



UNIVERSIDAD DE QUINTANA ROO
DIVISIÓN DE CIENCIAS E INGENIERÍA

**Emulador de una Celda de Combustible tipo PEM con
Redes Neuronales Artificiales**

TESIS

Para obtener el grado de
Ingeniero en Sistemas de Energía

Presenta

Angel Amisadai Ba Villarreal

Director de Tesis

Dr. Víctor Manuel Sánchez Huerta

ASESORES

Dr. Gliserio Romeli Barbosa Pool

Dra. Beatriz Escobar Morales

Dr. José Hernández Rodríguez

Dr. Jorge Ovidio Aguilar Aguilar

Chetumal, Quintana Roo, México, Julio 2017



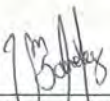


UNIVERSIDAD DE QUINTANA ROO
DIVISIÓN DE CIENCIAS E INGENIERÍA

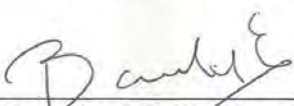
Tesis elaborada bajo la supervisión del Comité de Tesis y
aprobada como requisito parcial para obtener el grado de:

INGENIERO EN SISTEMAS DE ENERGÍA

COMITÉ DE TESIS

Director: 
Dr. Víctor Manuel Sánchez Huerta

Asesor: 
Dr. Gliserio Romeli Barbosa Pool

Asesor: 
Dra. Beatriz Escobar Morales



Chetumal, Quintana Roo, México, Julio 2017

Dedicatoria

Principalmente a Dios por haberme permitido llegar hasta este momento tan importante en mi formación profesional y también por permitirme tener como padres a Luis Armando Ba Chan y Leydi Marbella Villarreal Chan que juntos trabajaron duro hasta el cansancio de sus huesos por verme progresar, que con su apoyo incondicional y su amor han sido los pilares de mi vida. A mi madre que nunca dejo de apoyarme incluso en momento difíciles. A mi padre que en paz descansa, siempre trabajando por nosotros que hoy nos deja como enseñanza de la vida que la familia esta primero sin importar la distancia ni el tiempo, el amor a la familia lo es todo. Sin ustedes no hubiera logrado esta meta.

Agradecimientos

A mi madre y padre que siempre estuvieron para mí cuando más los necesite, sé todo lo que han pasado para poder darme lo que ahora tengo. Gracias por todo, siempre estaré agradecido con ustedes.

A mis amigos que formaron parte de esta etapa, que sin ellos tampoco podría haber logrado esta meta.

A todos mis profesores que tuve durante mi formación académica, de ellos aprendí mucho.

Y a todos aquellos que directamente o indirectamente me ayudaron a lo largo de este camino.

GRACIAS.

1 INTRODUCCIÓN	1
1.1 JUSTIFICACIÓN	2
1.2 OBJETIVO GENERAL	3
1.3 OBJETIVOS ESPECÍFICOS	3
1.4 ALCANCES	3
1.5 HIPÓTESIS	4
1.6 ORGANIZACIÓN DE LA TESIS	4
2 MARCO TEÓRICO	5
2.1 CELDAS DE COMBUSTIBLE.....	5
2.1.1 Tipos de Celdas de Combustible.....	6
2.1.2 Celda de Combustible de Membrana de intercambio de protones (PEMFC o celda PEM).....	7
2.2 REDES NEURONALES ARTIFICIALES	10
2.2.1 Redes Neuronales Biológicas	11
2.2.2 Redes Neuronales Artificiales.....	12
2.2.3 El modelo matemático de una neurona	14
2.2.4 Tipos de Funciones de activación	16
2.2.5 Arquitecturas Neuronales	18
2.2.5.1 Redes neuronales estáticas	18
2.2.5.2 Redes neuronales dinámicas o recurrentes	18
2.2.5.3 Redes neuronales unicapa	19
2.2.5.4 Redes neuronales multicapa.....	20
2.2.6 El perceptrón.....	21
2.2.7 El perceptrón multicapa.....	23
2.2.7.1 Estructuras de modelos lineales.....	25
2.2.7.1.1 Estructura Respuesta al impulso finito (FIR).....	27
2.2.7.1.2 Estructura ARX (AutoRegressive, External Input).....	28
2.2.7.1.3 Estructura ARMAX (AutoRegressive, Moving Average, External Input)	29
2.2.7.1.4 Estructura OE (Output Error)	29
2.2.7.2 Estructuras de modelos no lineales basadas en redes neuronales.....	30
2.2.7.2.1 NNFIR y NNARX.....	32
2.2.7.2.2 NNARMAX	32
2.2.7.2.3 NNOE.....	33
2.2.8 Métodos de Aprendizaje	34
2.2.8.1 Método por Error de Predicción	36
2.2.8.1.1 Búsqueda de un mínimo.....	36
2.2.8.1.2 El método del Gradiente	38
2.2.8.1.3 El método de Newton.....	38
2.2.8.1.4 El método de Gauss-Newton	40

2.2.8.1.5 El método de Levenberg-Marquardt.....	41
2.2.9 Selección del espacio de retardo	43
2.2.10 Redes Neuronales Recurrentes para Identificación.....	44
3 METODOLOGÍA.....	45
3.1 IDENTIFICACIÓN CON REDES NEURONALES ARTIFICIALES	45
3.1.1 Sistema bajo estudio	45
3.1.2 Análisis de los datos	47
3.1.3 Determinación del orden del sistema.	49
3.1.4 Caso 1 (sin retroalimentación de salidas pasadas)	51
3.1.4.1 Predicción del comportamiento eléctrico de la celda de combustible con la red neuronal NNARX.....	51
3.1.4.2 Validación del modelo	53
3.1.4.3 Análisis de los residuos	56
3.1.4.4 Predicción del comportamiento eléctrico de la celda de combustible con la red neuronal con la estructura NNOE.	58
3.1.4.5 Validación del modelo.	59
3.1.4.6 Análisis de los residuos	60
3.1.5 Caso 2 (considerando la retroalimentación de dos salidas pasadas)	62
3.1.5.1 Predicción del comportamiento eléctrico de la celda de combustible con la red neuronal NNARX utilizando dos salidas pasadas	63
3.1.5.2 Validación del modelo	64
3.1.5.3 Análisis de los residuos.....	66
3.1.5.4 Predicción del comportamiento eléctrico de la celda de combustible con la red neuronal con la estructura NNOE utilizando dos salidas pasadas.....	68
3.1.5.5 Validación del modelo	69
3.1.5.6 Análisis de los residuos.....	70
3.2 DESARROLLO DEL EMULADOR	72
3.2.1 Programación de la red neuronal.....	73
4 RESULTADOS.....	80
4.1 RESULTADOS DE SIMULACIÓN Y EXPERIMENTALES.....	80
4.1.1 Resultados de la identificación del sistema.	80
4.1.2 Resultados de la simulación del sistema.....	84
4.1.3 Resultados de la emulación del sistema	89
5 RESULTADOS.....	96
5.1 CONCLUSIONES Y TRABAJOS FUTUROS	96

Resumen

Este trabajo de investigación presenta un emulador de celdas de combustible de membrana de intercambio de protones, en el cual se estudia la aplicación de dos estructuras de redes neuronales artificiales recurrentes (RNAR) de tiempo discreto en tareas de carácter predictivo.

En las RNAR empleadas en este estudio se utilizó una estructura NNARX y una estructura NNOE, ambos con algoritmos de entrenamiento tipo recurrente y de ajuste de pesos Levenberg-Marquart (LM) para la predicción cinética del comportamiento eléctrico de una mono celda tipo PEM.

La validación de los resultados obtenidos por la RNAR se realizó con el software de Matlab y Labview. El emulador de la PEMFC se implementó en un sistema embebido de tiempo real programado mediante Labview.

En este trabajo se estudió un sistema MISO (Multiple Inputs – Single Output) con el fin de analizar y evaluar la potencialidad del uso de las RNAR para la predicción y emulación del comportamiento eléctrico y obtener un modelo de la PEMFC. El modelo construido de acuerdo a la metodología propuesta en este trabajo de investigación proporciona predicciones precisas del voltaje para todo el rango de funcionamiento de la PEMFC y, por lo tanto, la capacidad de representar el comportamiento variable de la celda.

Palabras Clave: Celdas de Combustible de Intercambio de Protones (PEMFC), Emulador, Redes Neuronales Artificiales, NNARX, NNOE.

Capítulo 1

1 Introducción

Las celdas de combustible de intercambio protónico (PEMFC, por sus siglas en inglés) son una tecnología dentro de la economía del hidrógeno (H^2) que se perfila como una alternativa de fuente de energía. La PEMFC produce agua y calor como subproducto, por lo que es considerado una fuente renovable y que opera a bajas temperaturas ($20^\circ C$ a $80^\circ C$ aproximadamente) lo que permite una rápida respuesta de arranque y que su construcción, transportación y seguridad sean más confiables [1,2,3].

Las características antes mencionadas hacen que el estudio de la PEMFC sea bastante difundido. Sin embargo, aunque actualmente se encuentran disponibles aplicaciones con celdas de combustibles, éstas aún se encuentran en su etapa de investigación. En la industria y en diversas instituciones se investigan algunos aspectos importantes que buscan perfeccionar la tecnología de las celdas de combustible. La investigación de nuevos materiales, la construcción de componentes, la instrumentación y los combustibles implicados en sistemas de generación eléctrica basados en celdas de combustible son los más mencionados en la actualidad. Debido al costo y otras razones antes mencionadas, tener la flexibilidad de poder contar con una celda de combustible virtual, mediante un emulador heurístico, permite realizar pruebas para diseño de equipos periféricos, tales como controladores de potencia o balance de planta.

Por otra parte, las redes neuronales artificiales son ampliamente usadas en diversos ámbitos de la ciencia e ingeniería principalmente porque permiten resolver problemas complejos, que muchas veces no son fáciles de resolver con técnicas tradicionales. Las características principales de la RNAR son: tratan incertezas de fácil adaptabilidad a nuevos datos y abstraen el conocimiento físico de los sistemas a ser modelados, ya que el modelo puede ser generado únicamente a partir de datos de entrada y salida [4]. Con el objetivo de superar deficiencias de los modelos matemáticos se ha incrementado el estudio en esta rama de la inteligencia artificial que utilizan modelos basados en redes neuronales artificiales. Un emulador de celdas de combustible tipo PEM basado en redes neuronales permite mediante un dispositivo electrónico y de software reproducir

respuestas estacionarias y dinámicas equivalentes a las celdas reales. La emulación de la PEMFC permitirá conocer el comportamiento de las celdas de combustible en régimen transitorio, ante perturbaciones que pudieran ocurrir durante su operación.

Este trabajo de investigación centra su estudio en el comportamiento estático y dinámico de una mono celda tipo PEM para la formulación de los parámetros necesarios de un modelo de un stack de celda de combustible. La información generada por el emulador puede ser importante para el diseño de celdas de combustibles y construcción de sistemas de acondicionamiento de potencia eficientes y la formulación de estrategias de control.

1.1 Justificación

Adquirir un sistema de generación eléctrica basado en celdas de combustible tiene un alto costo, debido a su instrumentación, manufactura y el combustible que se usa para su funcionamiento debido a que es una tecnología que aún está en la fase de investigación.

Por otra parte, tener un emulador de celdas de combustible permitirá agilizar procesos tanto de estudio como experimentación, ayudando a entender el comportamiento dinámico de la celda de combustible. El desarrollo de un modelo basado en RNAR ayudará a entender variables internas como los parámetros atribuidos en los materiales de construcción que van ligados al comportamiento dinámico y en la conversión de energía de la propia celda, así como también permitirá la mejora de dispositivos y sistemas en los cuales haya aplicaciones de celdas de combustible.

El uso de RNAR como estructura del emulador propicia que sea un sistema robusto ante variaciones de ruido, teniendo en cuenta que no se necesita conocimientos a priori del sistema original. Las técnicas de identificación de sistemas se emplean cada vez más para obtener modelos más realistas. Las redes neuronales usadas en la identificación de sistemas, son poderosas herramientas para mapear las relaciones no lineales entre las entradas y salidas de sistemas complejos y son capaces de representar sistemas dinámicos como los de las celdas de combustible.

Este trabajo busca desarrollar un emulador que integre en su estructura a las RNAR para que sea un modelo realista y describa de la mejor manera el funcionamiento de una celda de combustible tipo PEM y lograr con ello una mejor predicción en el voltaje de salida. En un futuro inmediato, el emulador puede ser usado como una herramienta docente. El diseño fue realizado con software práctico en el ámbito de investigación y en la industria para que la información sea fácil de interpretar y usar. Todo ello permita ser una herramienta para trabajar con redes neuronales por primera vez y conseguir la experiencia necesaria para ser capaz de trabajar más adelante con programas más complejos.

1.2 Objetivo General.

Desarrollar el código de cómputo e implementar los datos de entrenamiento para emular en tiempo real una celda de combustible tipo PEM, mediante redes neuronales artificiales recurrentes.

1.3 Objetivos Específicos

- Estudiar y analizar los fundamentos analíticos del comportamiento de las celdas de combustible tipo PEM, y de las estructuras de las redes neuronales artificiales.
- Desarrollar la estructura y el modelo de las redes neuronales que describa el comportamiento validado de una celda de combustible tipo PEM.
- Implementar el emulador en tiempo real instalando los diferentes elementos físicos y de software del emulador.
- Generar conocimiento en el uso de redes neuronales artificiales para el modelado de sistemas dinámicos y en el manejo de algoritmos de sistemas embebidos para el desarrollo de técnicas de inteligencia artificial.

1.4 Alcances

Este trabajo de tesis se desarrolló bajo el siguiente marco de trabajo:

- Se dispondrá de un software para realizar los algoritmos necesarios para el funcionamiento de la red neuronal.
- Se contará con tablas que contienen información sobre el funcionamiento de una celda de combustible tipo PEM.
- El modelo que se realizará será de con datos estáticos

- El entrenamiento y validación de la estructura de la red neuronal recurrentes se realizará por medio del software Matlab.
- La emulación de la red neuronal se hará por medio del software Labview y la tarjeta FPGA NI myRIO.
- El modelo final se implementará con Labview y la tarjeta FPGA NI myRIO para la emulación del comportamiento de la celda de combustible.

1.5 Hipótesis

Es posible emular el comportamiento eléctrico de una celda de combustible tipo PEM mediante redes neuronales artificiales recurrentes.

1.6 Organización de la tesis

El proyecto está dividido en cinco capítulos. En el capítulo dos incluye los conceptos teóricos más importantes sobre las celdas de combustible tipo PEM y las redes neuronales. En el capítulo tres se describen detalladamente la metodología de la identificación del sistema con redes neuronales artificiales recurrentes utilizando el paquete de “Neural Network Based System Identification toolbox” de Magnus Norgaard [41] y la emulación del modelo resultante en el software National Instruments Labview. En el capítulo cuatro se muestran los resultados obtenidos de la identificación del sistema, la simulación y la emulación del modelo y finalmente en el capítulo cinco se habla de las conclusiones y de las posibles mejoras para futuros trabajos.

Capítulo 2

2 Marco teórico

En este capítulo se describe el principio de operación de una PEMFC, así como los elementos que lo conforman. De igual forma en este capítulo se describen las redes neuronales artificiales, las estructuras más populares usadas para la identificación de sistemas dinámicos y se hará énfasis en los temas pertinentes a este trabajo de investigación.

2.1 Celdas de Combustible

Las celdas de combustible son una tecnología destinada para una función importante en el ámbito energético de las próximas décadas [5, 6]. Las celdas de combustible en principio son dispositivos que producen electricidad y calor, mediante la oxidación electroquímica de un combustible, el cual puede ser hidrógeno, sin estar involucrado en un proceso de combustión a altas temperaturas, ver la figura 2.1 [7, 8,9].

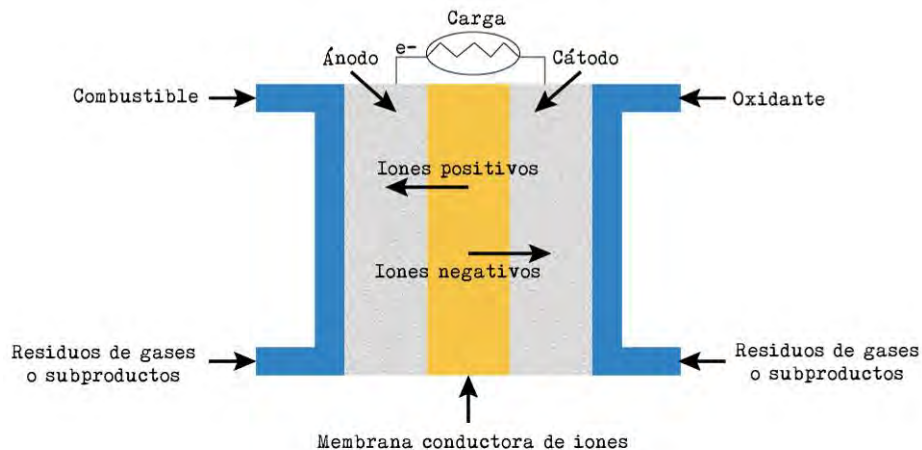
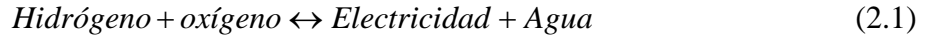


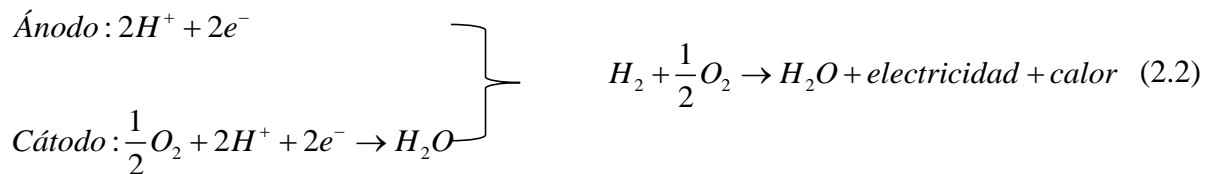
Figura 2.1 Esquema básico del funcionamiento de una celda de combustible.

Una celda de combustible (CC) es un dispositivo electroquímico que convierte la energía química de las reacciones de oxidación de un combustible y de reducción de un oxidante, en energía eléctrica, agua y calor. Como se describe en la ecuación 1, las CC pueden generar electricidad de manera inversa a la electrólisis del agua; en la electrólisis del agua se separa en dos componentes, el hidrógeno y el oxígeno, mientras que en las celdas de combustible se obtiene una corriente eléctrica como resultado de la reacción de estos dos gases [10, 11, 12].



A diferencia de las máquinas de combustión cuya eficiencia está regida por el ciclo de Carnot y limitada por la temperatura, la eficiencia teórica de las celdas de combustible está dada por las leyes de Faraday, que relacionan directamente la corriente producida en una reacción electroquímica con la cantidad de combustible que se utiliza, bajo la segunda ley de la termodinámica, la eficiencia máxima de las CC se obtiene de la división de la entalpia de la formación sobre la energía libre de Gibbs de formación, esta división arroja resultados teóricos de aproximadamente 80% [7].

Un sistema basado en celdas de combustible opera con dos tipos de gases, un combustible y un oxidante, que pasan a través de las superficies del ánodo y del cátodo que están opuestas al electrólito, respectivamente. Estos dos gases generan energía eléctrica por oxidación electroquímica del hidrógeno, y de la reducción electroquímica del oxígeno como consecuencia. La energía química almacenada en el enlace $H - H$ de la molécula de H_2 se convierte en energía eléctrica y vapor de agua. Los reactivos se transforman electroquímicamente, de acuerdo con las siguientes semireacciones de la ecuación 2, [5, 7, 13].



2.1.1 Tipos de Celdas de Combustible

Existe una gran variedad de tecnologías de celdas de combustible actualmente y están clasificadas de acuerdo al electrólito que emplean, temperatura de funcionamiento, etcétera. Dependiendo del electrólito, se producen diferentes reacciones químicas dentro de la propia celda, así como también se usan diferentes agentes catalíticos, rangos de temperatura, el combustible requerido, entre los factores más relevantes, por ende, determinan el tipo de aplicaciones en las que son usados [5, 6, 12, 14, 15]. A continuación, se presenta un resumen en la tabla 2.1 de las principales celdas de combustible.

Tabla 2.1 Clasificación de las celdas de combustible según su potencia.

<i>Tipos de Celdas de Combustible</i>	<i>Ion móvil</i>	<i>Temperatura de Operación</i>	<i>Eficiencia</i>	<i>Rango</i>	<i>Aplicaciones</i>
Alcalina (AFC)	OH^-	50 – 200°C	70%	10 a 100 kW	Usada en vehículos espaciales
Membrana de intercambio de protones (PEMFC)	H^+	25 – 100°C	≈ 40%	0,1 a 500 kW	Transporte, aplicaciones móviles y sistemas de CHP de baja potencia
Metanol directo (DMFC)	H^+	50 – 100°C	≈ 40%	Pocos mW a 100kW	Transporte, aplicaciones móviles y sistemas de CHP de baja potencia
Ácido fosfórico (PAFC)	H^+	≈ 200°C	> 40%	Superior a 10 MW	Sistemas CHP de 200 KW de potencia
Carbonato fundido (MCFC)	CO_3^{2-}	≈ 650°C	> 60%	100 MW	Adecuada para sistemas CHP de alta y media potencia
Óxido Sólido (SOFC)	O^{2-}	500 – 1000°C	> 70%	Superior a 100 kW	Adecuada para todos los niveles de potencia (baja, media y alta)

En este trabajo de investigación se propone el uso de los datos de celdas de combustible de membrana de intercambio de protones, la cual se detalla a continuación.

2.1.2 Celda de Combustible de Membrana de intercambio de protones (PEMFC o celda PEM)

Este tipo de celdas de combustible también llamadas celdas de membranas poliméricas (PEM), proporcionan una densidad energética muy alta y tienen la ventaja de ser ligeras y de dimensiones pequeñas, sí se compara con los otros antes mencionados. Las celdas tipo PEM usan como electrólito a un polímero sólido, conductor de iones y electrodos porosos de carbono. Las PEMFC

necesitan Hidrógeno y Oxígeno. [14, 16]. El electrólito sólido utilizado en este tipo de celdas tiene algunas ventajas con respecto a los electrólitos líquidos, como son: una alta densidad de corriente, la disminución en sus niveles de corrosión y una amplia vida útil.

El funcionamiento de las PEMFC se basa en las reacciones electroquímicas entre un combustible y un oxidante donde no interviene ningún ciclo de combustión. Para la generación de electricidad en la PEMFC se hace pasar el oxidante (oxígeno, que por lo general proviene del aire), a través del cátodo y, por la otra parte, se hace pasar el combustible (hidrógeno) a través del ánodo. El catalizador de platino acelera la reacción para que el átomo del hidrógeno se divida en un ion H^+ y un electrón. Los iones H^+ se dirigen hacia el cátodo, a través del electrolito, mientras que los electrones (e^-) circulan a través de un circuito externo donde se produce una corriente que puede ser utilizada para alimentar dispositivos eléctricos. Las moléculas del oxígeno se rompen en átomos en el cátodo, y se combinan con iones de hidrógeno y electrones con la ayuda del catalizador forman moléculas de H_2O y calor. Durante todo este proceso no se produce ni se emite ningún contaminante, en la imagen 2.2 se muestra el funcionamiento de la celda de combustible tipo PEM. [17, 18, 19].

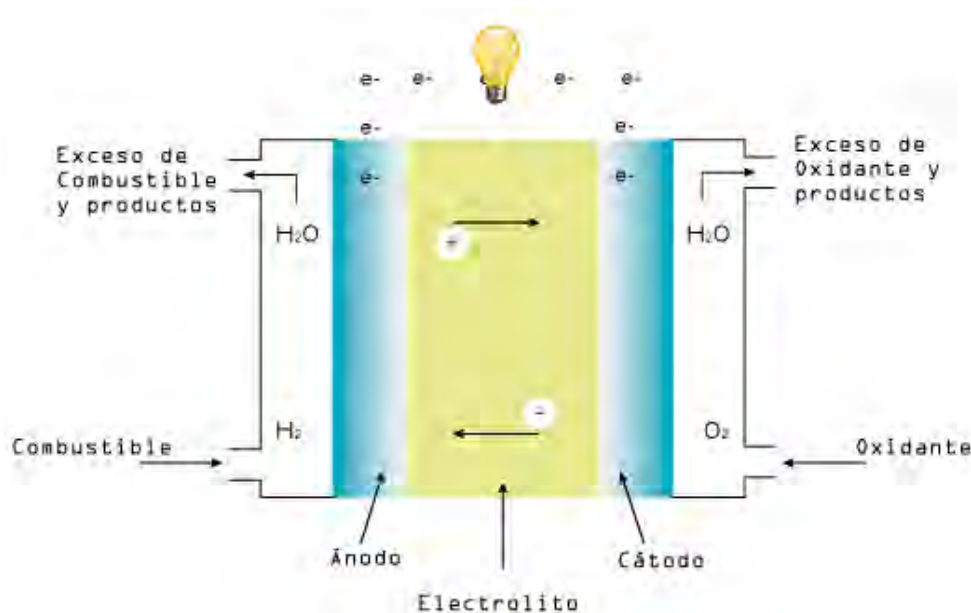


Figura 2.2 Funcionamiento de una celda de combustible de intercambio protónico (PEMFC).

La membrana conductora de iones (protones) consiste de una delgada película del polímero, ensamblada en medio de dos electrodos porosos impregnados con un electrocatalizador y que sirve como aislante electrónico. El Nafion es la membrana más utilizada en las PEMFC y es fabricada por DuPont. El Nafion tiene como base un polímero de polietileno, dicho polímero se modifica sustituyendo átomos de hidrógeno por átomos de flúor (perfluorinización), obteniendo un politetrafluoroetileno o PTFE (Teflón). Finalmente, el polímero PTFE se sulfona y un uno de los lados de la cadena se añade un grupo Sulfónico (SO_3). Este grupo se enlaza iónicamente, obteniéndose al final de la cadena lateral un ion SO_3^- . Y por esta razón, la estructura resultante también se denomina “ionómero” [1, 20, 21, 22, 23, 24]. En la siguiente figura 2.3 se muestra una membrana de Nafion y la representación esquemática de su estructura.

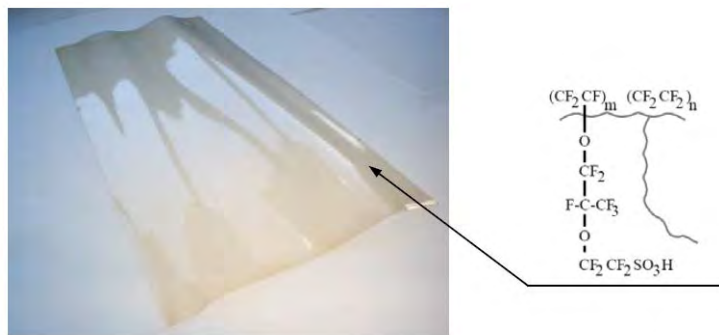


Figura 2.3 Membrana de Nafion y su representación estructural.

(Fuente: www.fuelcellstore.com).

La capa catalizadora es formada usando nanopartículas de platino que se adhieren a las estructuras más grandes de carbón. El platino queda homogéneamente distribuido en toda la superficie de la capa catalizadora que está en contacto con los reactantes. El Platino tiene la propiedad de catalizar reacciones sin degradarse. La capa catalizadora es aplicada a la membrana de intercambio protónico usando el método de laminado, aspersion o proceso de impresión. [17, 19, 25, 26].

Las celdas de combustible tipo PEM operan a temperaturas relativamente bajas (entre 25 y 80°C), lo cual permite que tenga un rápido arranque al necesitar menos tiempo de calentamiento. En consecuencia, tiene menor desgaste entre sus componentes del sistema, y por lo tanto aumenta la

duración del mismo. Estas celdas operan con un catalizador de platino el cual encarece el costo del sistema. Además, el catalizador es extremadamente sensible a la contaminación por CO , por lo que es necesario utilizar un reactor adicional para reducir el nivel del CO en el gas del combustible. La figura 2.4 muestra ejemplos de PEMFC [27, 28, 29].



Figura 2.4. Ejemplos de Celdas de Combustible tipo PEM.

(Fuente: www.fuelcellstore.com)

Las celdas tipo PEM se usan fundamentalmente para aplicaciones en el transporte e instalaciones estacionarias debido a su rapidez para el arranque, baja sensibilidad a la orientación, y su relación favorable entre peso y energía producida. Las celdas de combustible tipo PEM son especialmente adecuadas para su uso en vehículos.

2.2 Redes Neuronales Artificiales

La creciente demanda tecnológica en nuestra sociedad ha requerido de nuevos enfoques para resolver problemas tecnológicos cada vez más complejos. Las Redes Neuronales Artificiales (RNA) con su masivo paralelismo y su capacidad de aprender, prometen dar soluciones a estos problemas. Esto ha ocasionado que el conocimiento sobre redes neuronales se incremente de manera significativa en las últimas décadas, en particular el uso de redes neuronales recurrentes para el modelado de sistemas [30]. Las RNA presentan varias ventajas sobre las técnicas de modelado convencionales, ya que son capaces de modelar sistemas sin la necesidad de suposiciones sobre la naturaleza de los mismos y son capaces de aprender relaciones lineales y no lineales entre las variables a partir de un conjunto de ejemplos sin importar que los datos sean incompletos o contengan errores.

2.2.1 Redes Neuronales Biológicas

Las redes neuronales artificiales se basan en el funcionamiento abstracto de los sistemas nerviosos biológicos. Las neuronas biológicas son células nerviosas que constituyen los elementos primordiales del sistema nervioso central. Una neurona es capaz de recibir información de otras neuronas, procesarla y generar una nueva información que se enviará a otras neuronas con las que está conectada. Se estima que el cerebro está compuesto por más de diez billones de neuronas y que cada una está conectada a más de diez mil neuronas. La neurona biológica está compuesta por: un cuerpo celular o soma, axón y múltiples dendritas [30], como se muestra en la figura 2.5.

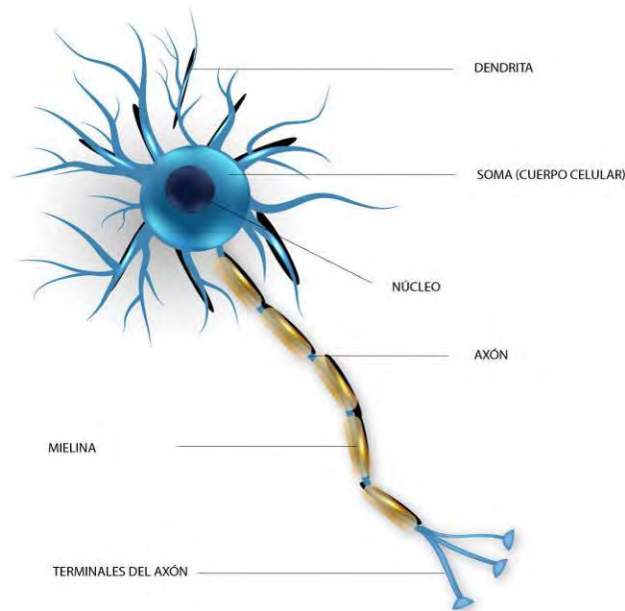


Figura 2.5. Neurona Biológica.

En algunas neuronas las uniones inician entre el axón y el soma para transmitir a lo largo del axón información electroquímica a otras células nerviosas. Cuando el axón está cerca de las células destino, se divide en muchas ramificaciones que forman la sinapsis con el soma o axones de otras células. La sinapsis puede ser excitatoria o inhibitoria según el neurotransmisor que se libere. Cada neurona recibe aproximadamente de 10,000 a 100,000 sinapsis y su axón realiza una cantidad similar de sinapsis [30].

Las señales que utilizan las neuronas son de dos tipos distintos: la eléctrica y la química. La señal de salida es eléctrica, mientras que la señal que se transmite entre los límites axómicos de una neurona y de las dendritas de las neuronas siguientes es de origen químico, de manera concreta se realiza mediante moléculas de sustancias transmisoras (neurotransmisores) que fluyen a través de sinapsis [31].

El funcionamiento de una neurona principalmente se basa en la diferencia de las concentraciones de iones de Sodio y de Potasio, esta diferencia en la membrana de la neurona produce un potencial de aproximadamente 70 mV negativos en el interior de la célula, lo cual es llamado potencial de reposo [31].

2.2.2 Redes Neuronales Artificiales

Las redes neuronales artificiales son modelos que tratan de emular las características y propiedades de las redes neuronales biológicas. Aunque no alcanzan la complejidad del cerebro; un modelo de red neuronal artificial tiene aspectos similares a las biológicas; y tienen la intención de construir sistemas de procesamiento de información paralela, distribuida y adaptativa, que puedan presentar un cierto comportamiento “inteligente”.

Las principales características de las redes neuronales artificiales destacan son las siguientes [32]:

1. *Auto-organización y adaptabilidad.* Utilizan algoritmos de aprendizaje adaptativo y auto-organizativo, por lo que las posibilidades de un procesamiento robusto y adaptativo son altas.
2. *Procesado no lineal.* Aumenta la capacidad de la red para aproximar funciones, clasificar patrones y aumenta su inmunidad frente al ruido. La red neuronal puede ser lineal o no lineal, esta característica es muy importante, especialmente si se intenta modelar sistemas generados mediante pautas no lineales.
3. *Procesado Paralelo.* Normalmente usan un gran número de nodos de procesamiento y con un alto nivel de interconectividad.

4. *Tolerancia ante fallos.* Una red neuronal es tolerante ante posibles fallas operacionales que solo afectan débilmente el rendimiento de estas. Esta propiedad es consecuencia de su naturaleza distribuida, de la información almacenada o procesada en la red neuronal.

La neurona artificial es una imitación del proceso de una neurona biológica y es la unidad fundamental para la operación de una red neuronal artificial. En la figura 2.6 se observa una neurona artificial en su forma general y su similitud con una neurona biológica.

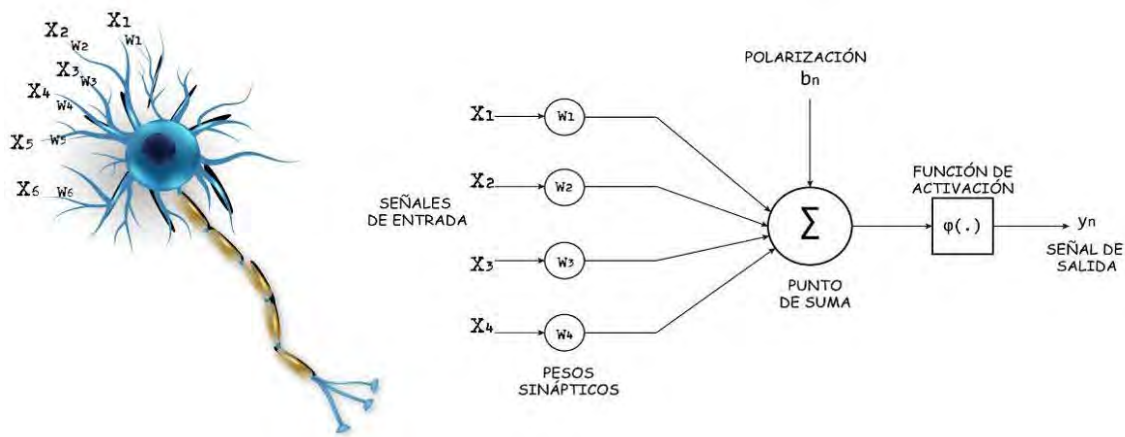


Figura 2.6. De la neurona biológica a la neurona artificial

De la imagen anterior se detalla el proceso biológico análogo con el sistema artificial.

- Las entradas $x_1, x_2, x_3, \dots, x_n$ se representan las señales que provienen de otras neuronas y que son capturas por las dendritas. Para las neuronas artificiales, las entradas pueden ser variables continuas o discretas, mientras que en las neuronas biológicas son pulsos discretos.
- Los pesos W_1 representa la intensidad de la sinápsis que conecta dos neuronas; tanto X_1 como W_1 son valores reales.
- φ es la función umbral que la neurona debe sobrepasar para activarse; este proceso ocurre biológicamente en el cuerpo de la célula.

En las siguientes secciones se omitirá la especificación de la palabra artificial ya que de ahora en adelante se trabaja exclusivamente este tipo de neuronas o redes.

2.2.3 El modelo matemático de una neurona

La neurona es la unidad fundamental del proceso de información en una red neuronal. En la figura 2.7 se muestra el modelo de la neurona; éste forma la base para el diseño de una red neuronal artificial. En la figura 2.7 se pueden identificar tres elementos básicos del modelo neuronal.

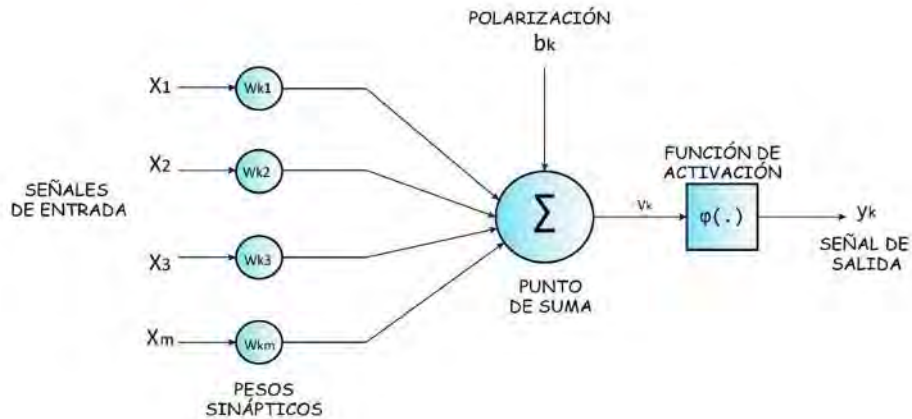


Figura 2.7. Modelo de una neurona artificial.

- *Los enlaces de conexión o un conjunto de sinapsis*, cada una de las cuales está caracterizada por un peso o ganancia sináptica. En específico, una señal x_j a la entrada de las sinapsis j conectada a la neurona k es multiplicada por un peso sináptico W_{kj} . Es importante notar que el primer subíndice corresponde a la neurona receptora, mientras que el segundo subíndice corresponde a la neurona emisora. Por lo que si $W_{kj} > 0$, entonces la conexión es excitatoria; así mismo, si $W_{kj} < 0$, la conexión es inhibitoria.
- *Sumador (Σ)*, suma las señales de entrada con su respectivo peso sináptico; las operaciones hasta aquí constituyen combinaciones lineales.
- *Función de activación*, sirve para limitar la amplitud de la salida de la neurona. La función de activación típicamente está normalizada dentro de un intervalo cerrado $[0,1]$ o alternativamente $[-1,1]$. Las funciones de activación son transformaciones no lineales.

El modelo neuronal de la figura 2.7 incluye una polarización externa, denotada b_k , esta polarización tiene como función incrementar o reducir la entrada total a la función de activación, dependiendo de si la polarización es positiva o negativa, respectivamente.

Es posible describir en términos matemáticos a la neurona k de la figura 2.7, por las siguientes ecuaciones.

$$u_k = \sum_{j=1}^m w_{kj} x_j \quad (2.3)$$

y

$$y_k = \varphi(u_k + b_k) \quad (2.4)$$

donde $x_1, x_2, x_3, \dots, x_m$ son las señales de entrada; $w_{n1}, w_{n2}, w_{n3}, \dots, w_{nm}$ son los pesos sinápticos de la neurona k , u_k es la combinación lineal de las entradas ponderadas; b_k es la polarización; $\varphi(\bullet)$ es la función de activación y finalmente y_k es la señal de salida de la neurona.

Se hace mención que la polarización es un parámetro externo de la neurona k , pero es posible considerarla como parte de las señales de entrada, de manera que se si se combinan con (2.3) y (2.4) se tiene lo siguiente:

$$v_k = \sum_{j=0}^m w_{kj} x_j \quad (2.5)$$

y

$$y_k = \varphi(v_k) \quad (2.6)$$

A v_k se le denomina potencial de activación. En la ecuación (2.5) se ha agregado una nueva sinapsis y su entrada es:

$$x_0 = +1 \quad (2.7)$$

Y su peso correspondiente es:

$$w_{k0} = b_k \quad (2.8)$$

Hasta este punto se ha mencionado a la función de activación, sin embargo, no se ha definido formalmente. A continuación, se analizan algunas de las funciones más usadas para la activación de redes neuronales.

2.2.4 Tipos de Funciones de activación

Las funciones de activación, denotadas por $\varphi(v)$, definen la salida de la neurona en término de un campo local inducido v . En este apartado se mencionan los básicos y más usados por las redes neuronales.

1. Función escalón o umbral. Para esta función de activación, descrita en la figura 2.8, se tiene lo siguiente:

$$\varphi(v) = \begin{cases} 1, & \text{sí, } v \geq 0 \\ 0, & \text{sí, } v < 0 \end{cases} \quad (2.9)$$

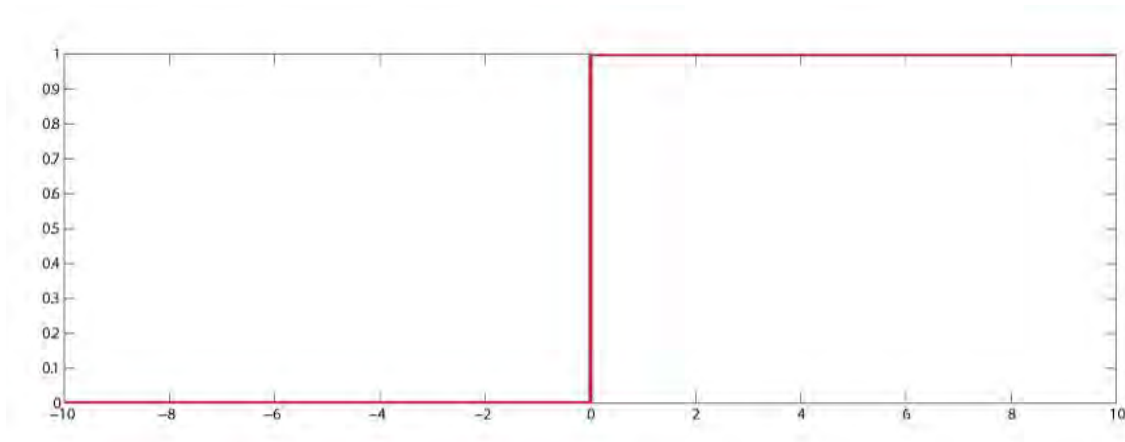


Figura 2.8. Representación de la función escalón.

2. Función lineal a tramos: Para esta función descrita en la figura 2.9 se tiene:

$$\varphi(v) = \begin{cases} 1, & v \geq +\frac{1}{2} \\ v, & +\frac{1}{2} > v > \frac{1}{2} \\ 0, & v \leq -\frac{1}{2} \end{cases} \quad (2.10)$$

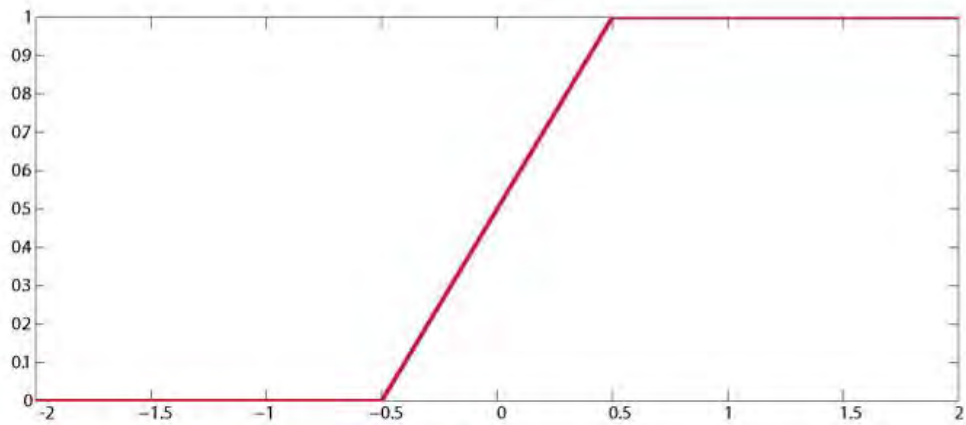


Figura 2.9. Representación de la función lineal a tramos.

3. Función sigmoideal: Ésta es la función más utilizada en redes neuronales artificiales. Es definida estrictamente creciente, con un comportamiento asintótico. Un ejemplo de la función sigmoideal es la función logística.

$$\varphi(v) = \frac{1}{1 + e^{-av}} \quad (2.11)$$

Donde a es el parámetro que determina la pendiente de la función sigmoideal como se muestra en la figura 2.10.

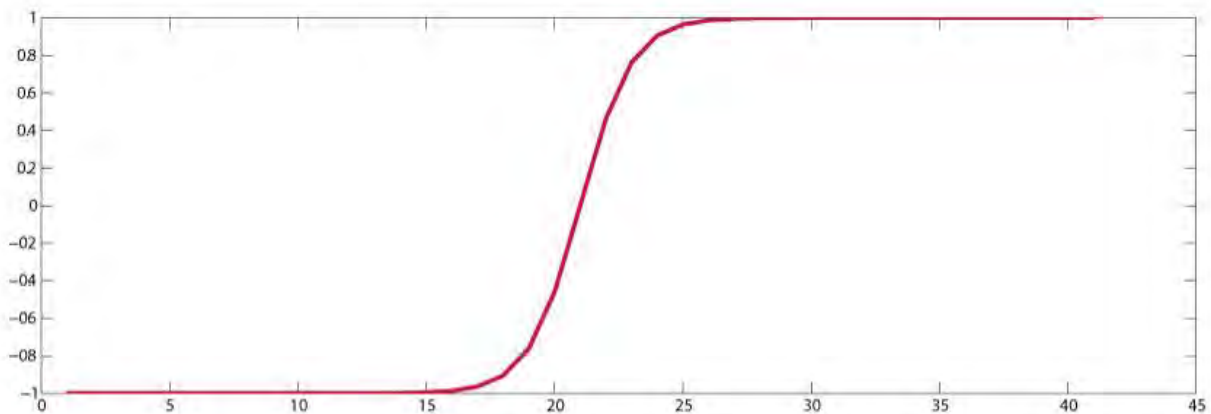


Figura 2.10. Representación de la función sigmoideal.

Las funciones de activación hasta aquí descritas toman valores en el intervalo cerrado $[0,1]$, sin embargo, tienen variantes que pueden tomar valores en el intervalo cerrado $[-1,1]$. La función escalón puede definirse como en (2.8).

$$\varphi(v) = \begin{cases} 1, & \text{sí, } v > 0 \\ 0, & \text{sí, } v = 0 \\ -1, & \text{sí, } v < 0 \end{cases} \quad (2.12)$$

Lo cual es comúnmente llamada signo. Para la forma correspondiente de la función sigmoideal se puede usar la función tangente hiperbólica, la cual está definida como:

$$\varphi(v) = \tanh(v) \quad (2.13)$$

2.2.5 Arquitecturas Neuronales

Las arquitecturas neuronales se pueden clasificar en dos grupos debido a su estructura. Las redes neuronales estáticas y las redes neuronales dinámicas o recurrentes.

2.2.5.1 Redes neuronales estáticas

Esta arquitectura también es conocida como red neuronal *feed forward* (hacia adelante) y la característica principal de esta red neuronal es que no existe conexión hacia atrás, autorrecurrente o lateral, y se utilizan en tareas de reconocimiento o clasificación de patrones [33].

2.2.5.2 Redes neuronales dinámicas o recurrentes

Las redes neuronales recurrentes, poseen una diferencia con respecto a la otra estructura antes mencionada, y es que estas contienen al menos tienen un lazo de retroalimentación. Estas redes neuronales pueden tener una sola capa oculta con cada una de sus neuronas retroalimentado de las señales de salida a la entrada, como se puede ver en la figura 2.11, donde por primera vez se menciona un operador de retardo unitario Z^{-1} . En el caso de las redes neuronales recurrentes tienen un profundo impacto en la capacidad de aprendizaje de la red y las hacen especialmente indicadas para el procesamiento de secuencias temporales.

Este tipo de estructura se seleccionó para este trabajo de investigación porque poseen la capacidad de aproximar funciones y por lo tanto su aplicabilidad ha sido usada en la rama de identificación de sistemas [34, 35].

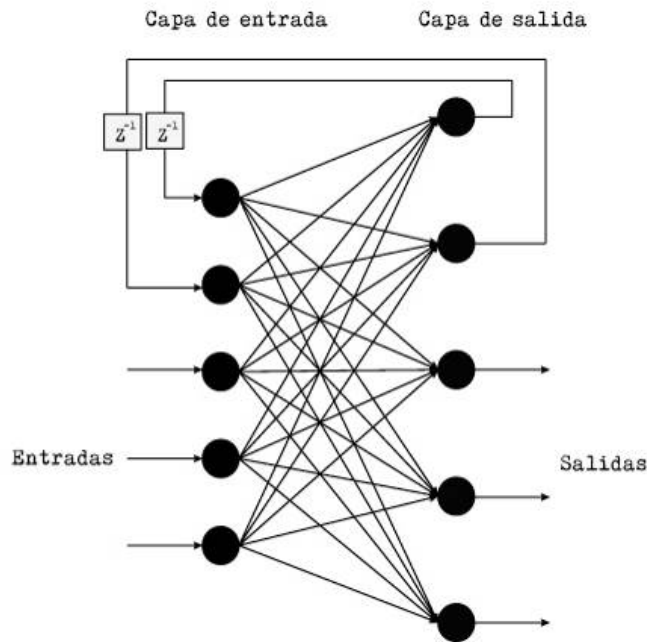


Figura 2.11. Redes neuronales recurrentes con neuronas ocultas.

2.2.5.3 Redes neuronales unicapa

Por lo general las redes neuronales se organizan por capas. En el caso más simple de una red neuronal es la red unicapa y solo establece una conexión directa entre neuronas de entrada y neuronas de salida por medio de la sinapsis, como se muestra en la figura 2.12. En esta figura se puede apreciar que cinco neuronas en la capa de entrada, así como en la capa de salida conectadas entre sí. A esta estructura se le conoce como unicapa porque solo tiene una capa conectada a los nodos (neuronas) de entrada; para esta designación no se toma en cuenta la capa que contiene los nodos de entrada.

Las redes neuronales unicapa se utilizan típicamente en áreas relacionadas con la autoasociación; por ejemplo, para regenerar información de entrada que se presentan a la red incompletas o distorsionadas.

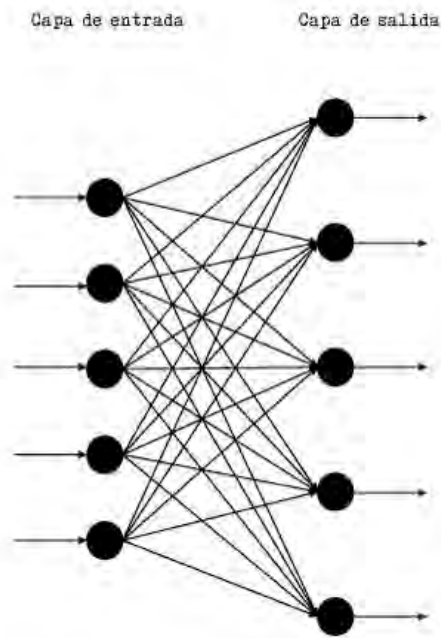


Figura 2.12. Red neuronal unicapa.

2.2.5.4 Redes neuronales multicapa

Las redes neuronales multicapa son aquellas que disponen de un conjunto de neuronas agrupadas en una o más capas ocultas o unidades ocultas. La figura 2.13 muestra una red neuronal multicapa totalmente conectada, en el sentido de que todos los nodos en cada capa están conectados a todos los nodos de la siguiente capa. En el caso de que falte alguna conexión entre algún nodo, se dice que la red es parcialmente conectada. Normalmente, todas las neuronas reciben señales de entrada de otra capa anterior, más cercana a la entrada de la red, y estas envían las señales de salida a una capa posterior, más cercana a la salida de la red.

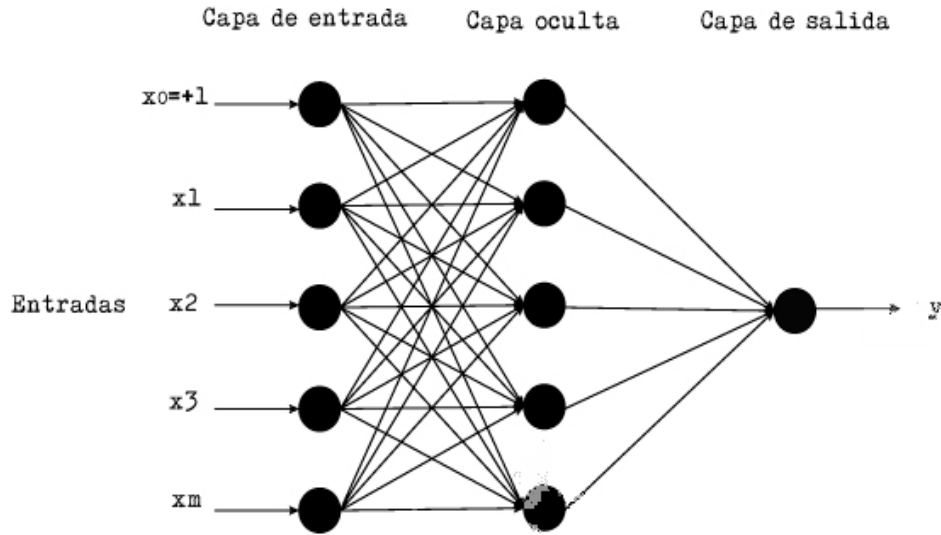


Figura 2.13. Red neuronal multicapa.

2.2.6 El perceptrón

La red tipo Perceptrón fue propuesta por el Psicólogo Rosenblatt en el año 1958 [36]. Su intención por ilustrar las propiedades fundamentales de los sistemas inteligentes de manera general, pero sin detallar a fondo las condiciones específicas y desconocidas para ciertos organismos biológicos [37].

El perceptrón es la forma más simple de una red neuronal usada para la clasificación de patrones que sean linealmente separables. En otras palabras, consiste en una neurona con pesos sinápticos ajustables y una polarización. El perceptrón que propuso el Psicólogo Rosenblatt en [36] fue el primer modelo de aprendizaje supervisado y también probó que, sí los patrones utilizados para entrenar el perceptrón pertenecían a dos clases linealmente separables, entonces el algoritmo del perceptrón convergerá y posicionará la superficie de decisión en un hiperplano entre las dos clases [37].

Con base al modelo neuronal de McCulloch-Pitts se construye el perceptrón como se muestra en la figura 2.14, donde la salida está dada por la siguiente ecuación:

$$y = \varphi(v) = \text{sign}(v)$$

$$v = \sum_{i=0}^m w_i x_i$$
(2.14)

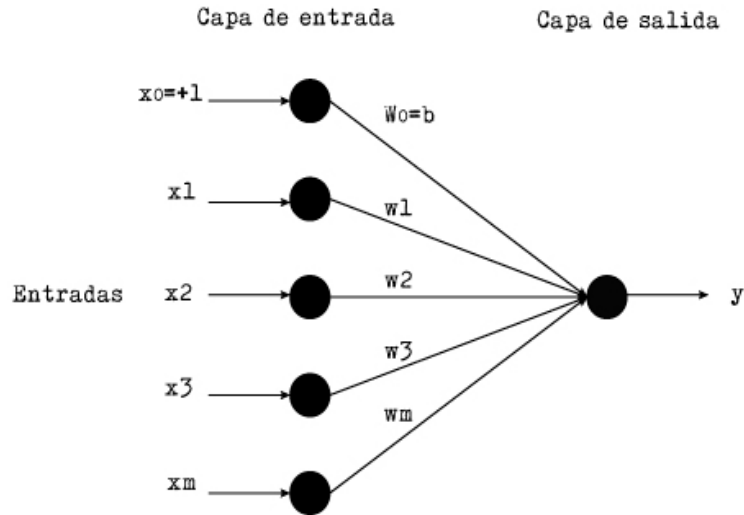


Figura 2.14. El perceptrón.

Para la adaptación de los pesos sinápticos del perceptrón w_1, w_2, \dots, w_m son clases linealmente separables. Considerando la salida del perceptrón y la salida deseada, es posible definir el error como:

$$e(k) = d(k) - y(k)$$

$$d(k) = \begin{cases} +1, & x(k) \in C_1 \\ -1, & x(k) \in C_2 \end{cases}$$
(2.15)

Con $d(k)$ es la salida deseada y $x(k)$ es el vector de entradas. Entonces la ley de adaptación de pesos queda como:

$$w(k+1) = w(k) + \eta(k)e(k)x(k)$$
(2.16)

Donde $w(k)$ es el vector de pesos. Esta adaptación de pesos puede lograrse utilizando un algoritmo de mínimos cuadrados o un algoritmo de gradiente descendiente, entre otros [37].

2.2.7 El perceptrón multicapa

El perceptrón multicapa fue desarrollado por P. Werbos en 1974 y posee una estructura con al menos una capa oculta; y su algoritmo de entrenamiento es basado en la corrección de error e implementa el gradiente distribuido en los diferentes componentes de la red. En 1969 Minsky y Papert publicaron un libro *Perceptrons: An introduction to computational geometry* [38], donde se presentaba un análisis detallado del perceptrón, en términos de sus capacidades y limitaciones. En especial, los problemas y las restricciones de que una red tipo perceptrón tiene sobre la capacidad a la hora de resolver problemas; y de su mayor desventaja en su incapacidad para resolver problemas de clasificación que no sean linealmente separables [37].

Las redes neuronales tipo perceptrón multicapa (MPL *Multilayer Perceptron Network por sus siglas en inglés*), han sido puestos en prueba satisfactoriamente para resolver diversos problemas por medio de algoritmos conocidos como *retropropagación* de la cual, se conocen dos etapas:

1. Etapa hacia adelante, este algoritmo tiene parámetros de la red fijos. Se presentan una señal de entrada a la red la cual se propaga hacia adelante para producir una salida.
2. Etapa hacia atrás, en este algoritmo el error entre la salida deseada y la red se propagan hacia atrás. Los parámetros de la red se modifican para minimizar el cuadrado del error.

La ecuación (2.14) describe el comportamiento de la MPL:

$$\hat{y}_i(t) = g_i[\varphi, \theta] = F_i \left[\sum_{j=1}^{n\eta} W_{i,j} f_j \left(\sum_{l=1}^{n\varphi} w_{j,l} \varphi_l + w_{j,0} \right) + W_{i,0} \right] \quad (2.17)$$

Donde θ , especifica el vector de los parámetros que contiene los parámetros ajustables de la red, es decir los pesos sinápticos y los bias $\{w_{j,l}, W_{i,j}\}$, aquí los bias también pueden interpretarse como pesos con valor siempre a 1.

El Perceptrón multicapa se diferencia por tres características únicas:

1. El modelo de cada neurona de la red incluye una función de activación no lineal, dado que la no-linealidad es suave (diferenciable en cualquier punto). Y la función de activación más utilizada es la tangente hiperbólica (2.8) [35].
2. La red contiene una o más capas ocultas que no son parte de las entradas o las salidas de la red. Estas neuronas ocultas permiten que la red “aprenda” tareas complejas por la extracción progresiva de las características principales de los patrones de entrada.
3. La red presenta alto grado de conectividad, que es determinado por las sinapsis de la misma red.

Estas características junto con la habilidad de aprender de la experiencia a través del entrenamiento del MLP da como resultado una herramienta de gran potencial en la computación [39, 40]. En la siguiente figura 2.15 se muestra la representación de una red tipo MLP y en la figura 2.16 se presenta el diagrama de bloques de una red neuronal.

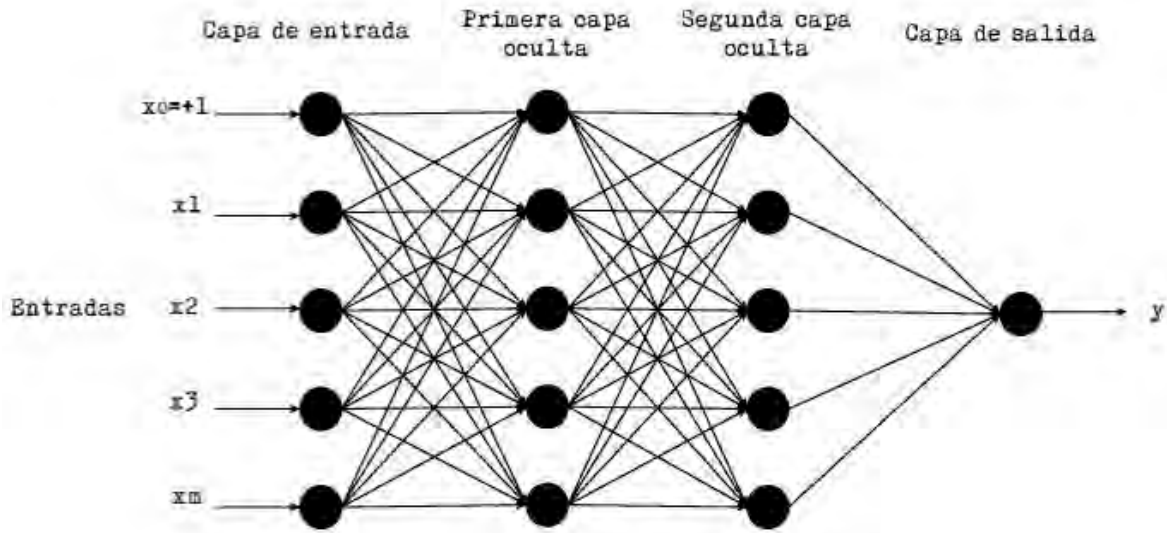


Figura 2.15. Red neuronal tipo Perceptrón multicapa.

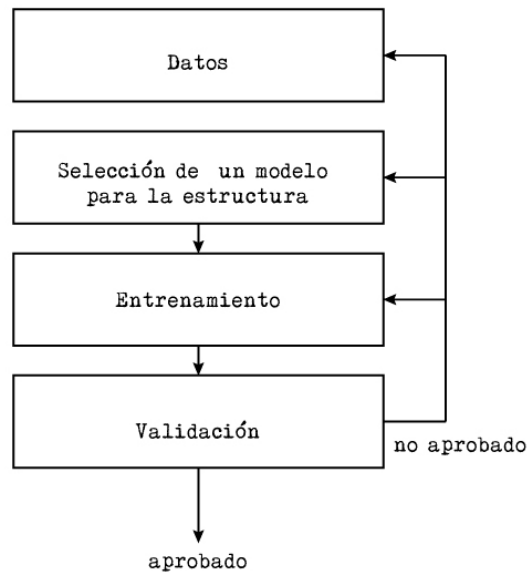


Figura 2.16. Diagrama de bloques del proceso para obtener la RNA.

2.2.7.1 Estructuras de modelos lineales

Considérese un sistema lineal representado por el siguiente modelo:

$$y(t) = G(q^{-1})u(t) + H(q^{-1})e(t) \quad (2.18)$$

Donde G y H son funciones de transferencia en el operador de retardo unitario q^{-1} , con:

$$q^{-d}x(t) = x(t-d) \quad (2.19)$$

Donde d es un múltiplo de periodo de muestreo y $e(t)$ es ruido blanco independiente de $u(t)$.

Además, considérese que se tiene un predictor de varianza mínima con la siguiente estructura:

$$\hat{y}(t|t-1, \theta) = H^{-1}(q^{-1}, \theta)G(q^{-1}, \theta)u(t) + [1 - H^{-1}(q^{-1}, \theta)]y(t) \quad (2.20)$$

Y escrito en la forma del predictor:

$$\hat{y}(q^{-1} | \theta) = \varphi^T(t) \theta \quad (2.21)$$

Donde θ es vector de parámetros y φ es el vector de regresión, el cual contiene entradas pasadas, salidas pasadas o señales derivadas a partir de entradas y salidas pasadas. Por lo tanto, la estructura general del modelo se puede escribir cómo y en la figura 2.17 se muestra su diagrama de bloques:

$$A(q^{-1})y(t) = (q^{-d}) \frac{B(q^{-1})}{F(q^{-1})}u(t) + \frac{C(q^{-1})}{D(q^{-1})}e(t) \quad (2.22)$$

donde:

$$\begin{aligned} A(q^{-1}) &= 1 + a_1q^{-1} + \dots + a_nq^{-n} \\ B(q^{-1}) &= b_0 + b_1q^{-1} + \dots + b_mq^{-m} \\ C(q^{-1}) &= 1 + c_1q^{-1} + \dots + c_kq^{-k} \\ D(q^{-1}) &= 1 + d_1q^{-1} + \dots + d_nq^{-i} \\ F(q^{-1}) &= 1 + f_1q^{-1} + \dots + f_rq^{-r} \end{aligned} \quad (2.23)$$

A, C, D y E son mónicos.

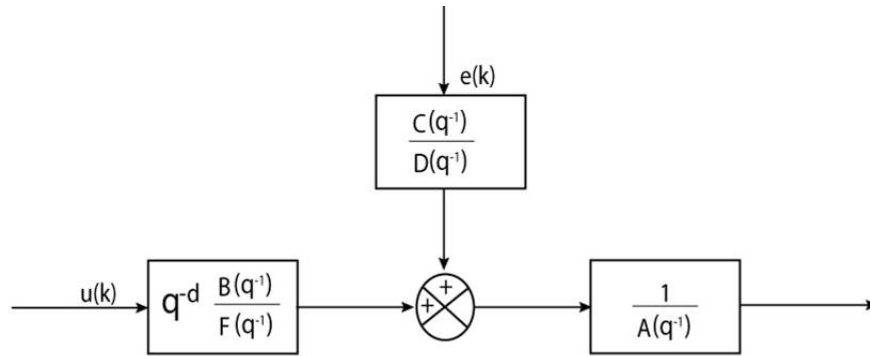


Figura 2.17. Diagrama a bloques para el modelo general.

2.2.7.1.1 Estructura Respuesta al impulso finito (FIR)

Es la estructura más simple y con una respuesta rápida donde su característica principal es la estabilidad, y se expresa de la siguiente forma:

$$\begin{aligned} G(q^{-1}, \theta) &= q^{-d} B(q^{-1}) \\ H(q^{-1}, \theta) &= 1 \end{aligned} \quad (2.24)$$

Y en su forma de predictor es:

$$\hat{y}(q^{-1} | \theta) = q^{-d} B(q^{-1}) u(t) \quad (2.25)$$

Y puede expresarse con su equivalente, en la forma de predictor, como se muestra a continuación

$$\hat{y}(q^{-1} | \theta) = \varphi^T(t) \theta \quad (2.26)$$

Donde $\varphi(t)$ es el vector de regresión y está definido por:

$$\varphi(t) = [u(t-d) \dots u(t-d-m)]^T \quad (2.27)$$

Correspondiendo, al vector de parámetros θ , y es compuesto en la forma:

$$\theta = [b_0 \dots b_m] \quad (2.28)$$

2.2.7.1.2 Estructura ARX (AutoRegressive, External Input)

Esta estructura puede tomar el nombre de *Controlled AutoRegressive Model* (CAR) en modelos de ecuaciones de error y en modelos de series paralelos. Y toma la forma:

$$\begin{aligned} G(q^{-1}, \theta) &= q^{-d} \frac{B(q^{-1})}{A(q^{-1})} \\ H(q^{-1}, \theta) &= \frac{1}{A(q^{-1})} \end{aligned} \quad (2.29)$$

En la forma del predictor:

$$\begin{aligned} \hat{y}(t | \theta) &= q^{-d} B(q^{-1}) u(t) + [1 - A(q^{-1})] y(t) \\ \hat{y}(t | \theta) &= \varphi^T(t) \theta \end{aligned} \quad (2.30)$$

con:

$$\begin{aligned} \varphi^T(t) \theta &= [y(t-1) \dots y(t-n), u(t-d), \dots, u(t-d-m)]^T \\ \theta &= [-a_1, \dots, -a_n, b_0 \dots b_m]^T \end{aligned} \quad (2.31)$$

Aunque G ahora tiene polos, y solo existe una relación algebraica entre las entradas pasadas y las salidas medidas. Por consiguiente, el predictor es considerado estable; incluso si el sistema es inestable. Esta es una de las características importantes de esta estructura.

2.2.7.1.3 Estructura ARMAX (AutoRegressive, Moving Average, External Input)

Este modelo es aún más general que la estructura ARX y toma la siguiente forma:

$$\begin{aligned} G(q^{-1}, \theta) &= q^{-d} \frac{B(q^{-1})}{A(q^{-1})} \\ H(q^{-1}, \theta) &= q^{-d} \frac{C(q^{-1})}{A(q^{-1})} \end{aligned} \quad (2.32)$$

En la forma de predictor:

$$\begin{aligned} \hat{y}(t | \theta) &= q^{-d} \frac{B(q^{-1})}{C(q^{-1})} u(t) + \left[1 - \frac{A(q^{-1})}{C(q^{-1})} \right] y(t) \\ \hat{y}(t | \theta) &= q^{-d} B(q^{-d}) u(t) + [1 - A(q^{-d})] y(t) + [C(q^{-d}) - 1] \varepsilon(t, \theta) \\ \hat{y}(t | \theta) &= \phi^T(t, \theta) \theta \end{aligned} \quad (2.33)$$

Donde $\varepsilon(t, \theta) = y - \hat{y}(t | \theta)$, representan la predicción del error o residual. La forma del predictor y los vectores de parámetros son definidos como se muestra a continuación:

$$\begin{aligned} y(t | \theta) &= [y(t-1) \dots y(t-n), u(t-d) \dots u(t-d-m), \varepsilon(t, \theta), \dots, \varepsilon(t-k, \theta)]^T \\ \theta &= [-a_1, \dots, -a_n, b_0, \dots, b_m, c_1, \dots, c_k]^T \end{aligned} \quad (2.34)$$

Dada la presencia del polinomial C , el predictor tiene polos. Así que C debe tener sus raíces dentro del círculo unitario. Los polos también implican que el vector de regresión depende de los parámetros del modelo, lo cual hace más complicada la estimación [41].

2.2.7.1.4 Estructura OE (Output Error)

Esta estructura es utilizada cuando el único ruido que afecta al sistema es el ruido blanco de la medición. Y toma la siguiente forma:

$$y(t) = q^{-d} \frac{B(q^{-1})}{F(q^{-1})} u(t) + e(t) \quad (2.35)$$

Correspondiendo a la siguiente elección de G y H:

$$G(q^{-d}, \theta) = q^{-d} \frac{B(q^{-1})}{F(q^{-1})} \quad (2.36)$$

$$H(q^{-d}, \theta) = 1$$

Y su predictor toma la forma:

$$\hat{y}(t|\theta) = q^{-d} \frac{B(q^{-1})}{C(q^{-1})} u(t) \quad (2.37)$$

$$\hat{y}(t|\theta) = q^{-d} B(q^{-d}) u(t) + [1 - F(q^{-d})] \hat{y}(t|\theta) = \varphi^T(t|\theta)$$

donde:

$$\varphi(t, \theta) = \left[\hat{y}(t-1|\theta) \dots \hat{y}(t-r|\theta), u(t-d) \dots u(t-d-m) \right] \quad (2.38)$$

$$\theta = [-f_1 \dots -f_r, b_0 \dots b_m]^T$$

Para que el predictor sea estable, las raíces F deben estar dentro del círculo unitario [41].

2.2.7.2 Estructuras de modelos no lineales basadas en redes neuronales

Cuando se tiene como objetivo el modelado de la dinámica de un sistema no lineal, el problema de encontrar la estructura de un modelo tiende a dificultarse. Las MPL como se ha mencionado, tienen la capacidad de aproximar relaciones no lineales a partir de un conjunto de datos. Así que es natural pensar en las redes neuronales tipo perceptrón [41] ya que dentro de estos existe una

familia de estructuras de modelos que son convenientes usar para el modelado de sistemas dinámicos no lineales, y, habiendo ya hecho esta elección, la estructura del modelo básicamente se reduce en lo siguiente:

- Seleccionar las entradas de la red
- Seleccionar la arquitectura interna de la red.

Una aproximación muy común es usar las estructuras de los modelos lineales, pero con una estructura interna de red neuronal tipo MLP. Esto tiene ciertas ventajas como:

- La estructura es una extensión natural de modelos lineales conocidos.
- La arquitectura interna puede expandirse gradualmente con una gran flexibilidad, que es necesaria para modelar relaciones no lineales complejas.
- Las decisiones estructurales requeridas por el usuario son reducidas a un nivel razonable de manejar.

Las estructuras de modelos no lineales, son denotadas por:

$$y(t) = g[(\varphi, \theta), \theta] + e(t) \quad (2.39)$$

O en su forma de predictor:

$$\hat{y}(t|\theta) = g[\varphi(t, \theta), \theta] \quad (2.40)$$

$\varphi(t, \theta)$ es el vector de regresión, mientras que θ es el vector que contiene los parámetros ajustables en la red neuronal (pesos), g es la función realizada por la red neuronal. Dependiendo del tipo de vector de regresión que se seleccione, surgen diferentes estructuras no lineales. Sí el vector de regresión es seleccionado como un modelo ARX, la estructura es llamada NNARX como el acrónimo *Neural Network ARX*. De la misma manera para NNFIR, NNARMAX y NNOE.

2.2.7.2.1 NNFIR y NNARX

Como en el caso lineal, estos predictores son siempre estables, debido a que son puramente relaciones algebraicas entre la predicción, las entradas pasadas y las salidas pasadas. Esto resulta particularmente importante en el caso no lineal, debido a que el estudio de la estabilidad es mucho más complejo que para sistemas lineales. En la figura 2.18 se muestran los diagramas de bloques correspondientes a estas estructuras. La ausencia de problemas relacionados con la estabilidad, hace que estas estructuras, en particular la NNARX, sea la preferida cuando el sistema es determinístico o el nivel de ruido es insignificante [41]. En este trabajo de investigación dada las características de esta estructura, se decidió utilizarlo para generar los parámetros de la celda de combustible.

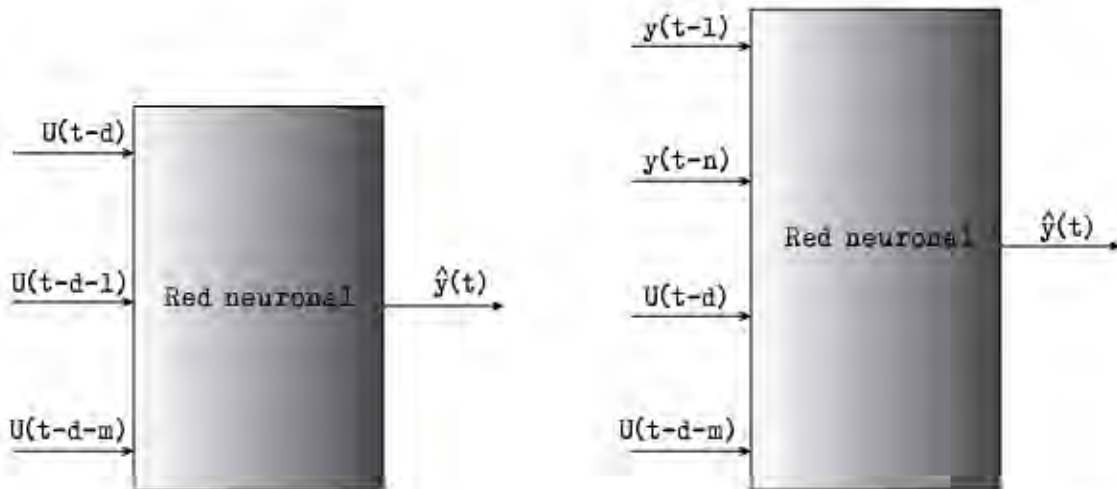


Figura 2.18. Diagrama a bloques de las estructuras NNFIR (izquierda) y NNARX (derecha).

2.2.7.2.2 NNARMAX

En esta estructura el vector de regresión está compuesto por salidas y por las entradas pasadas, así como de los errores de predicción pasados, estos últimos dependen de la salida del modelo y consecuentemente establecen una retroalimentación; lo cual puede provocar problemas de estabilidad. Es relevante considerar la estabilidad como una propiedad local, esto significa que un modelo NNARMAX es estable cuando opera en un régimen por calcular, mientras que puede ser inestable en otros [41]. En la figura 2.19 se muestra el diagrama a bloques de la estructura NNARMAX.

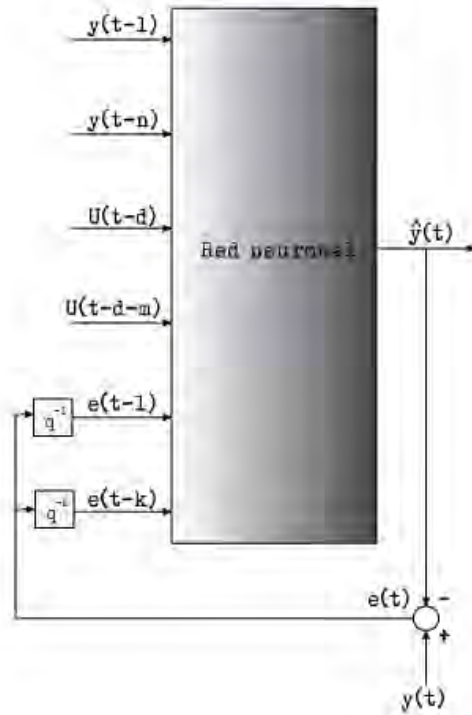


Figura 2.19. Diagrama a bloques de la estructura NNARMAX.

2.2.7.2.3 NNOE

En esta estructura algunos de sus regresores son predicciones de salidas pasadas y por lo tanto está sujeto a los mismos problemas que en la estructura NNARMAX [41]. Ver la figura 2.20.

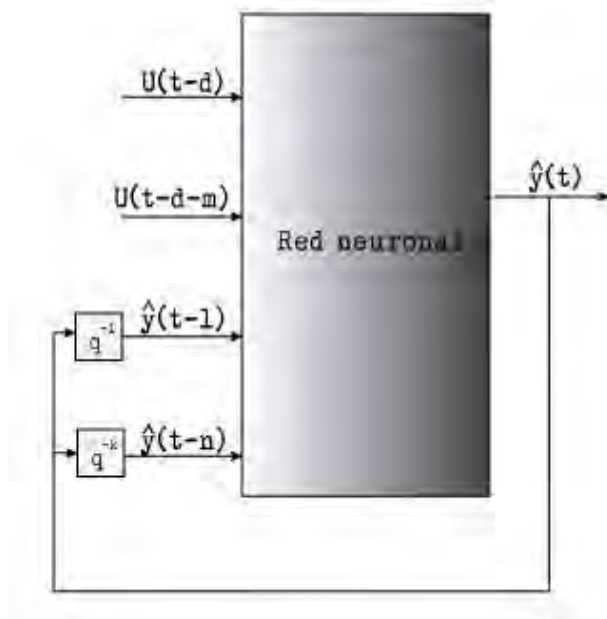


Figura 2.20. Diagrama a bloques de la estructura NNOE.

Existen muchas variantes o estructuras híbridas originadas de la combinación de estructuras lineales y no lineales eso dependerá del usuario y donde lo desea aplicar.

2.2.8 Métodos de Aprendizaje

El aprendizaje es un proceso donde se calculan los pesos de la red neuronal y estos son adaptados al ambiente en donde la red se desee utilizar. El tipo de aprendizaje está determinado por la regla o algoritmo que ajusta los valores de los pesos. No es muy difícil implementar un algoritmo de entrenamiento en una computadora, de manera que algunos softwares ya contienen paquetes para entrenamiento de datos.

El proceso de aprendizaje implica:

- Las redes neuronales son estimuladas por el medio donde se van utilizar.
- Las redes neuronales sufren cambios en sus pesos como resultado de esta estimulación.
- Las redes neuronales responden de una nueva forma al ambiente por que los cambios han ocurrido en su estructura interna.

De esta forma, al conjunto de reglas bien definidas para la solución del proceso de aprendizaje se le denomina algoritmo de aprendizaje. Los algoritmos de aprendizaje difieren unos de otros en la manera como ajustan los pesos de cada neurona.

En términos más formales, el proceso de aprendizaje o entrenamiento se establece como:

Dado un conjunto de datos:

$$Z^N = \{[u(t), y(t)], T = 1, \dots, N\} \quad (2.41)$$

Y un conjunto de modelos candidatos:

$$y(t) = \hat{y}(t|\theta) + e(t) = g[t|\theta] + e(t) \quad (2.42)$$

El propósito del entrenamiento es determinar un mapeo del conjunto de datos al conjunto de modelos candidatos:

$$Z^N \rightarrow \theta \quad (2.43)$$

Así que el modelo que es obtenido, provee predicciones que son en un sentido, muy cercanos a las salidas reales del sistema. Y para el caso de las redes neuronales el aprendizaje de la red se plantea como un problema de minimización de una determinada función de error. Y la medida de proximidad comúnmente utilizada es la del criterio del error medio cuadrático. Este es el algoritmo básico de aprendizaje que usa el perceptrón multicapa mediante el cual se van adaptando todos los parámetros de la red. El tipo de entrenamiento planteado para este trabajo de investigación es del tipo supervisado.

$$V_N(\theta, Z^N) = \frac{1}{2N} \sum_{t=1}^N \left[\left(y(t) - \hat{y}(t|\theta) \right) \right]^2 = \frac{1}{2N} \sum_{t=1}^N \varepsilon^2(t, \theta) \quad (2.44)$$

Se dice que este esquema es un método de error de predicción, ya que el objetivo como se dijo anteriormente es la de minimizar una suma sobre una norma de la predicción de errores. Por lo que este método corresponde a la estimación de máxima verosimilitud bajo el supuesto de que la distribución de la señal del error es Gaussiana. La característica más atractiva del criterio del error medio cuadrático es su simplicidad.

A continuación, se hablarán de algoritmos de aprendizaje basados en el criterio del error medio cuadrático.

2.2.8.1 Método por Error de Predicción

En este método de error por predicción, el objetivo es determinar los pesos de la red minimizando el criterio:

$$\hat{\theta} = \arg \min V^N(\theta, Z^N) \quad (2.45)$$

Cuando el criterio (2.34) es cuadrático en el error de predicción $\varepsilon(t, \theta)$, el entrenamiento es llamado mínimos cuadrados no lineales, que es un caso especial de optimización sin restricciones. Los problemas de mínimos cuadrados no lineales ocurren dentro de campos muy diversos, y existen muchos métodos para resolverlos. Como se muestra a continuación:

2.2.8.1.1 Búsqueda de un mínimo

Las series de Taylor de segundo orden del criterio en θ^* :

$$V_N(\theta, Z^N) = V_N(\theta^*, Z^N) + (\theta - \theta^*)^T V'_N(\theta^*, Z^N) + \frac{1}{2}(\theta - \theta^*)^T V''_N(\theta^*, Z^N)(\theta - \theta^*) \quad (2.46)$$

Donde la gradiente es definida por:

$$G(\theta^*) = V'_N(\theta^*, Z^N) = \left. \frac{dV_N(\theta, Z^N)}{d\theta} \right|_{\theta=\theta^*} \quad (2.47)$$

Y la derivada de segundo orden Hessiana, se define:

$$H(\theta^*) = V''_N(\theta^*, Z^N) = \left. \frac{d^2 V_N(\theta, Z^N)}{d\theta^2} \right|_{\theta=\theta^*} \quad (2.48)$$

Obsérvese que la notación informal $d\theta^2$ se ha utilizado para la derivada matricial $d\theta d\theta^T$. Una condición suficiente para que $\theta = \theta^*$ sea un mínimo de $V_N(\theta, Z^N)$ es que la gradiente sea igual a cero.

$$G(\theta^*) = 0 \quad (2.49)$$

Y que la matriz Hessiana sea definida positiva.

$$v^T H(\theta^*) v > 0 \quad (2.50)$$

Para todos los vectores distintos de cero. La búsqueda de un mínimo inicia desde una conjetura inicial sobre los parámetros. Los pesos se ajustan acorde al método de entrenamiento. Cuando el criterio, como aquí sea una función compleja de parámetros ajustables no lineales. El mínimo se encontrará usando un método de búsqueda iterativa. Este tipo de método generalmente toma la forma:

$$\theta^{i+1} = \theta^i + \mu^i f^i \quad (2.51)$$

Donde θ^i especifica la iteración actual (numero i), f^i la dirección de búsqueda y μ^i . Las iteraciones continúan hasta que θ^i sea lo suficientemente cercano al mínimo $\hat{\theta}$. El criterio tendrá en general más de un mínimo.

Desafortunadamente los métodos de búsqueda de este tipo no garantizan una convergencia con el mínimo que se espera. El mínimo que realmente se alcanza depende de la elección de los pesos iniciales.

2.2.8.1.2 El método del Gradiente

El principio del método del gradiente descendente, se encuentra en cada iteración para modificar los pesos sinápticos a lo largo de la dirección opuesta del gradiente. Es decir, la dirección de búsqueda es seleccionado como $f^i = -G(\theta^i)$.

$$\theta^{(i+1)} = \theta^{(i)} - \mu^{(i)} G^{(i)} \quad (2.52)$$

Se debe elegir el tamaño del paso $\mu^{(i)}$ adecuadamente y con esta selección de la dirección siempre es posible obtener una reducción del criterio $V_N(\theta^{(i+1)}, Z^N) \leq V_N(\theta^{(i)}, Z^N)$.

Al aplicar este método al entrenamiento de aprendizaje del perceptrón multicapa es útil ordenar los cálculos de la manera en cómo se utilizan en la estructura particular de la red. Este método es un caso del algoritmo de retropropagación o regla de la delta generalizada.

La convergencia del método depende únicamente del tamaño del paso $\mu^{(i)}$. Se pueden aplicar diferentes métodos para su selección. Por ejemplo, una búsqueda lineal suele dar la convergencia rápida en términos de iteraciones, pero requerirá de una gran cantidad de evaluaciones.

2.2.8.1.3 El método de Newton

Los métodos de gradiente únicamente se basan en una aproximación de primer orden al criterio para determinar la dirección de búsqueda. El método de Newton por otra parte es un método del siguiente paso. Este método determina una nueva iteración como el “minimizador” de una expansión de segundo orden del criterio acerca de una iteración actual.

$$\tilde{V}(\theta, Z^N) = V_N(\theta^{(i)}, Z^N) + (\theta - \theta^{(i)})^T G(\theta^{(i)}) + \frac{1}{2} (\theta - \theta^{(i)})^T H(\theta^{(i)}) [\theta - \theta^{(i)}] \quad (2.53)$$

Introduciendo la notación:

$$\psi(t, \theta) = \frac{d \hat{y}(t | \theta)}{d\theta} \quad (2.54)$$

Y la gradiente G y matriz Hessiana H está dado por:

$$\begin{aligned} G(\theta) &= V'_N(\theta, Z^N) = \frac{1}{N} \sum_{t=1}^N \psi(t, \theta) \psi^T(t, \theta) - \\ & y \quad (2.55) \\ H(\theta) &= V''_N(\theta, Z^N) = \frac{1}{N} \sum_{t=1}^N \psi(t, \theta) \left[y(t) - \hat{y}(t | \theta) \right] - \frac{1}{N} \sum_{t=1}^N \psi'(t, \theta) \varepsilon(t, \theta) \end{aligned}$$

El mínimo de (2.54) es encontrado en el punto donde $\tilde{V}'(\theta, Z^N) = 0$. Si la matriz Hessiana es simétrica, entonces corresponde a:

$$\begin{aligned} 0 &= G(\theta^{(i)}) + \frac{1}{2} \left[2H(\theta^{(i)})\theta - H(\theta^{(i)})\theta^{(i)} - H(\theta^{(i)})\theta^{(i)} \right] \\ &= G(\theta^{(i)}) + H(\theta^{(i)})[\theta - \theta^{(i)}] \end{aligned} \quad (2.56)$$

que conduce a la siguiente regla de actualización:

$$\theta^{(i+1)} = \theta^{(i)} - H^{-1}(\theta^{(i)})G(\theta^{(i)}) \quad (2.57)$$

Esta actualización corresponde al tamaño de paso $\mu^{(i)} = 1$, y la dirección de búsqueda se determina resolviendo la siguiente ecuación lineal:

$$H(\theta^{(i)})f^{(i)} = -G(\theta^{(i)}) \quad (2.58)$$

La dirección de búsqueda $f^{(i)}$, es referida frecuentemente como la dirección de Newton. En la práctica, el método debe complementarse con una búsqueda lineal, porque (2.54) es una aproximación al criterio real. La aproximación por lo general solo es válida en un cierto rango alrededor de la iteración actual y el paso completo puede traer una nueva iteración lejos del punto predicho por la misma aproximación. Este método es a veces llamado método de Newton Amortiguado.

En contraste con el método de la gradiente, uno no puede estar seguro de que este convergerá en lo absoluto cuando se encuentra cerca de un mínimo.

2.2.8.1.4 El método de Gauss-Newton

Este método utiliza la siguiente aproximación lineal del error de predicción

$$\varepsilon(t, \theta) = y(t) - \hat{y}(t | \theta).$$

$$\begin{aligned} \tilde{\varepsilon}(t, \theta) &= \varepsilon(t, \theta^{(i)}) + \left[\varepsilon'(t, \theta^{(i)}) \right]^T (\theta - \theta^{(i)}) \\ &= \varepsilon(t, \theta^{(i)}) - \left[\psi(t, \theta^{(i)}) \right]^T (\theta - \theta^{(i)}) \end{aligned} \quad (2.59)$$

Para modificar el criterio en el i -ésima iteración:

$$V_N(\theta, Z^N) \approx L(i)(\theta) = \frac{1}{2N} \sum_{t=1}^N \varepsilon^{-2}(t, \theta) \quad (2.60)$$

Cuando se evalúa $\theta = \theta^{(i)}$, la gradiente es el mismo que en el método de Newton:

$$G(\theta^{(i)}) = \left. \frac{dL^{(i)}(\theta)}{d\theta} \right|_{\theta=\theta^{(i)}} = \frac{1}{N} \sum_{t=1}^N \psi(t, \theta^{(i)}) \left[y(t) - \hat{y}(t | \theta^{(i)}) \right] \quad (2.61)$$

Pero la matriz Hessiana es diferente:

$$G(\theta^{(i)}) = \left. \frac{dL^{(i)}(\theta)}{d\theta} \right|_{\theta=\theta^{(i)}} = \frac{1}{N} \sum_{t=1}^N \psi(t, \theta^{(i)}) \psi^T(t | \theta^{(i)}) \quad (2.62)$$

$R(\theta)$ es llamado la matriz Hessiana Gauss-Newton, y por definición es positiva, además, tiene la propiedad de que requiere información de una derivada de primer orden. Análogamente al método de Newton, la actualización Gauss-Newton Hessiana se deriva como el minimizador de (2.60).

$$G(\theta^{(i+1)}) = \theta^{(i)} - R^{-1}(\theta^{(i)}) G(\theta^{(i)}) \quad (2.63)$$

En la práctica, la dirección del método de Gauss-Newton se calcula sin la inversa, pero resolviendo:

$$G(\theta^{(i+1)}) f^{(i)} = -G(\theta^{(i)}) \quad (2.64)$$

Sin embargo, el método Gauss-Newton es igual a la matriz Hessiana real, sí el segundo término no es tomado en consideración.

$$H(\theta^{(i+1)}) = R(\theta^{(i+1)}) - \frac{1}{N} \sum_{t=1}^N \psi(t, \theta^{(i)}) \varepsilon(t, \theta^{(i)}) \quad (2.65)$$

Sin embargo, este método aún está sujeto a ciertos problemas de mal acondicionamiento de variables como lo es también en el método de Newton.

2.2.8.1.5 El método de Levenberg-Marquardt

Este método es una modificación del método de Newton que elimina el cálculo de la matriz Hessiana. Y requiere un mayor esfuerzo computacional que los métodos de descenso del gradiente, pero es mucho más rápido que estos.

En este método la dirección se determina a través de una aproximación del criterio $L^{(i)}(\theta)$ que se toma válida solo en un rango cercano a la iteración actual. Si el mínimo de $L^{(i)}(\theta)$ está muy lejos de la iteración actual $\theta^{(i)}$ entonces se obtiene una dirección muy pobre. Intuitivamente, tiene más sentido buscar el mínimo $L^{(i)}(\theta)$ solo dentro del rango de la iteración actual. Seleccionando este rango con un radio $\delta^{(i)}$, el problema es minimizado como se muestra a continuación:

$$\theta^{(i+1)} = \arg \min L^{(i)}(\theta), \text{ con, } |\theta - \theta^{(i)}| \leq \delta^{(i)} \quad (2.66)$$

La regla de actualización que resuelve el problema de optimización es encontrada en [43]:

$$\begin{aligned} \theta^{(i+1)} &= \theta^{(i)} + f^{(i)} \\ \left[R(\theta^{(i)}) + \lambda^{(i)} I \right] f^{(i)} &= -G(\theta^{(i)}) \end{aligned} \quad (2.67)$$

Es una relación de tipo mónico entre $\lambda^{(i)}$ en un radio $\delta^{(i)}$ pero generalmente no es muy fácil encontrarlo. El método es bien conocido como algoritmo Levenberg-Marquardt después de dos contribuciones en [42] y [43]. Este algoritmo interpreta un rango de radio $\lambda^{(i)}$ y se interpreta como una región alrededor $\theta^{(i)}$ dentro de la cual la aproximación $L^{(i)}\theta$ puede ser considerado como una aproximación válida al criterio $V_N(\theta, Z^N)$. Este modelo tiene una región de confianza y es llamado, así como una interpretación del principio de [44, 45]. El método Levenberg-Marquardt es un método de región diseñado específicamente para los mínimos cuadrados no lineales.

A diferencia de los métodos anteriores, en el método que se habla actualmente no es común usar la búsqueda lineal. Este método básicamente está en contra de la filosofía de la región de confianza, ya que el tamaño del paso se ajusta automáticamente al radio de la región de confianza para obtener una mejor comprensión de la correspondencia entre el radio δ , y el parámetro λ . Para esto es útil hacer algunas consideraciones heurísticas sobre el impacto de la búsqueda en la dirección de búsqueda. Cuando $R(\theta^{(i)})\lambda I$ es sustituido por una matriz diagonal pura, la dirección de búsqueda

tiende a ser una gradiente negativa del criterio. Permitiendo que $\lambda \rightarrow \infty$ y la matriz diagonal domine sobre $R(\theta)$, conduciendo así a la gradiente con un tamaño de paso aproximado a cero. Por otro lado, sí $\lambda = 0$, se obtiene el método de Gauss-Newton ajustando λ , y a la dirección de búsqueda interpolando a la gradiente y la dirección de Gauss-Newton.

Existe una estrecha correspondencia entre la reducción del radio de la región de confianza y el incremento de λ (y viceversa). Sin embargo no existe una expresión cerrada para determinar un valor exacto de λ para un radio de región de confianza.

En este método existen dos maneras de lidiar con lo mencionado en los párrafos anteriores: métodos directos e indirectos. Los métodos directos ajustan la región de confianza directamente para luego utilizar un método iterativo para la determinación de los valores de λ . Por otro lado al ajustar λ directamente, sin prestar atención al tamaño actual de la región de confianza se representa un enfoque menos sofisticado.

Los métodos indirectos son similares en esencia al esquema propuesto en [37] pero para asegurar la convergencia, en [46] se sugirió seguir aumentándolo por factor, hasta que se produzca una reducción en el criterio, y cuando esto suceda entonces la iteración se acepta.

2.2.9 Selección del espacio de retardo

A menudo se les da más importancia a las estructuras usadas para el entrenamiento y a la estructura de la red neuronal. Sin embargo, una elección errónea del espacio de retardo o número de señales utilizadas como regresores pueden tener un impacto negativo en algunas aplicaciones. Un espacio de retardo implica que la dinámica esencial no será modelado y demasiado grande puede también ser un problema.

Por otra parte, si el espacio de retardo está debidamente determinado, el problema de selección de la estructura del modelo es sustancialmente reducido. Aunque algunas veces la generación del espacio de retardo puede resultar relativamente fácil a prueba y error, puesto que es necesario determinar un espacio de retardo suficientemente grande como un número adecuado de unidades ocultas.

2.2.10 Redes Neuronales Recurrentes para Identificación

Las redes neuronales recurrentes poseen una gran capacidad en la aproximación de funciones y su aplicabilidad ha sido extensa en el campo de la identificación y el control [47, 48, 49, 50], entre muchos otros. El proceso de identificación es muy importante para ciertas ramas de investigación, porque de allí se obtiene información de algún proceso, es decir, se puede conocer los parámetros de la estructura de una planta o sistema y obtener un modelo que se comporte igual. Existen procesos complicados como los biológicos, químicos o económicos en los que tener un modelo exacto es casi imposible.

La predicción o el pronóstico son áreas estrechamente relacionadas para modelado e identificación; mientras que los intentos de predecir estados futuros de un sistema son limitados por la precisión del modelo usado y el rango de correlación de proceso aleatorios que afecta al sistema. Mientras que la identificación con redes neuronales ha emergido como una herramienta efectiva para sistemas lineales y no lineales. Este enfoque o aproximación del modelo usa las versátiles características de las redes neuronales las hace ideales para la obtención de resultados teóricos estables.

El proceso de identificación de un sistema consiste en encontrar una descripción matemática que rija el comportamiento de un sistema estático o dinámico, es decir, crear un modelo matemático del sistema físico real a través de una serie de medidas de las señales de entrada y salida de dicho sistema.

La identificación del sistema puede ser aproximado en diferentes maneras, dependiendo del nivel de conocimiento a priori del sistema. Si la identificación es basada en datos medidos, asumiendo que no hay o sólo se tiene muy poco conocimiento del sistema, el proceso de identificación es llamado *blackbox modelling*. Los modelos de identificación caja negra son capaces de proporcionar una mejor aproximación para sistemas dinámicos, como una celda de combustible, sin la necesidad de conocer los parámetros físicos [46, 51].

En esta tesis, utiliza el enfoque del modelo de caja negra para conocer parámetros de un sistema de generación de energía basado en celdas de combustible y predecir su comportamiento dinámico.

Capítulo 3

3 Metodología

En este capítulo se describe el procedimiento que se ha seguido para desarrollar el modelo que emule las respuestas en estado estacionario y transitorio de una celda de combustible tipo PEM mediante redes neuronales artificiales recurrentes. De igual forma también se describe la identificación de la PEMFC y su implementación para emular el comportamiento eléctrico de la celda de combustible.

3.1 Identificación con Redes Neuronales Artificiales

3.1.1 Sistema bajo estudio

El sistema bajo estudio es una celda de combustible tipo PEM descrito en [52]. El experimento realizado con la celda de combustible denominado “Placa Alemana, Membrana Comercial, Diseño 2”, en donde la evaluación de la misma se realizó mediante mono celdas, y para cada mono celda se utilizaron pruebas potenciodinámicas; ya que estas, son una forma de rápida de determinar el comportamiento global de la celda, con ellas se puede caracterizar cualitativa y cuantitativamente el sistema, y también se puede determinar las regiones de operación óptima de la celda de combustible [52].

Cada mono celda consta de una curva de polarización que es una gráfica de la corriente contra el voltaje. La curva es obtenida variando la carga del circuito externo de una resistencia máxima a una resistencia mínima mediante un equipo automatizado [52].

En las figuras 3.1 y 3.2 se muestran las curvas de polarización para diferentes condiciones de presión y temperatura de la celda de combustible para cada mono celda durante la experimentación. Las curvas muestran que el desempeño de la celda de combustible, aumenta con el incremento de la presión y la temperatura.

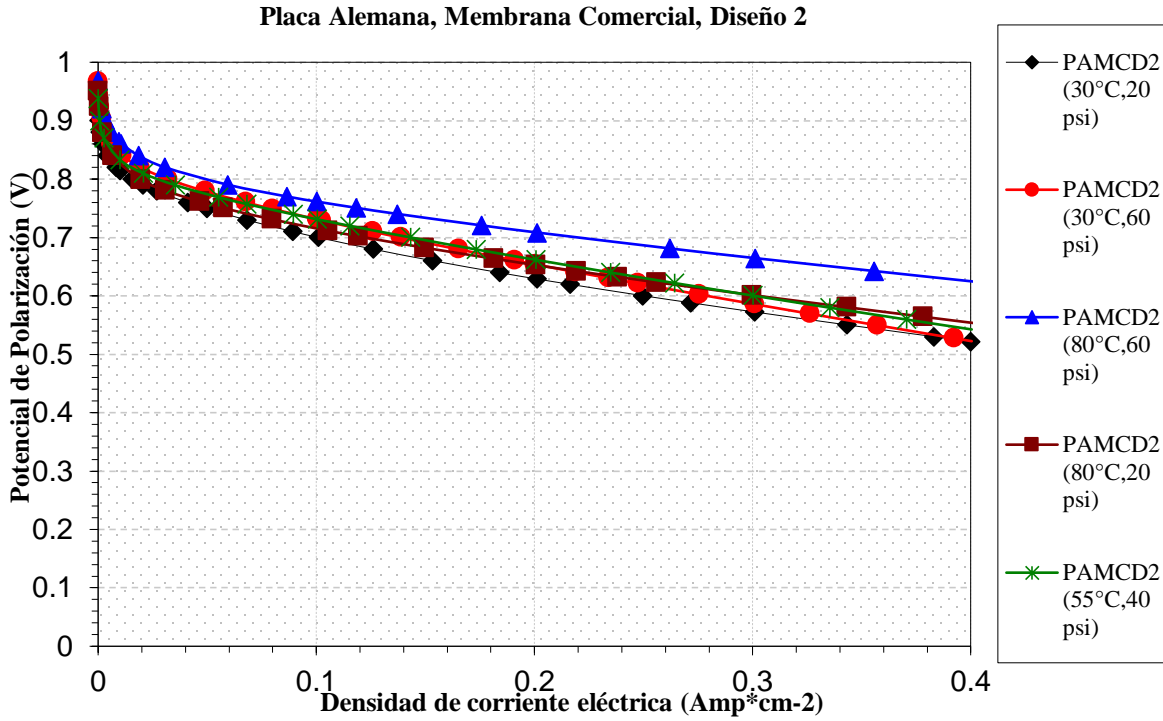


Figura 3.1. Curva de polarización para la Placa Alemana, Membrana Comercial, Diseño 2 [52].

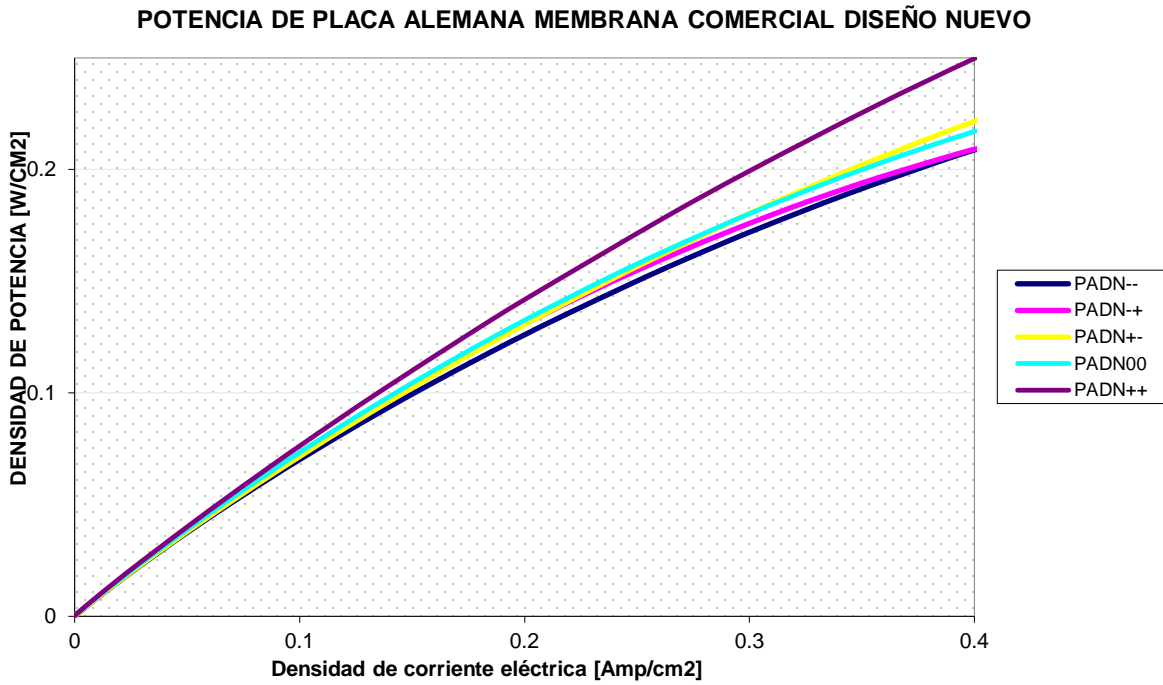


Figura 3.2. Curvas de polarización, densidad de corriente y densidad de potencia para la Placa Alemana, Membrana Comercial, diseño 2 [52].

Con esta información se prosiguió analizar los datos obtenidos de la celda de combustible tipo PEM para la identificación del sistema basado en redes neuronales recurrentes.

3.1.2 Análisis de los datos

Para que el proceso de identificación sea satisfactorio, es necesario que los datos utilizados para tal fin contengan información significativa sobre el sistema. Esto implica una cuidadosa selección de datos para la etapa de aprendizaje y la validación del sistema a identificar.

Para el análisis de los datos obtenidos de [52] del comportamiento dinámico del sistema bajo estudio se optó que esos datos sean utilizados conforme al orden de los experimentos realizados pero un módulo seguido de otro módulo, conforme la temperatura y la presión fueron aumentando, de manera que se tome el sistema como si estuviera en funcionamiento continuo y su demanda eléctrica creciera.

En el apéndice A, se resume la adquisición de los datos registrados por las gráficas de las figuras 3.1 y 3.2, los cuales son ordenados conforme se realizaron los experimentos en [52] para cada mono celda ordenados en dos tablas: Una tabla (tabla A1) de datos para patrones de entrenamiento o aprendizaje y la otra (tabla A2) de datos para patrones de validación.

En la estructura para la RNA la temperatura, la presión y la corriente serán considerados como datos de entrada del sistema y al voltaje como la salida del sistema. Esto selección se decidió porque la temperatura, la presión y la corriente influyen directamente al sistema de generación basado en celdas de combustible. Respecto a la salida (voltaje), en el sistema real es el resultado de la aplicación de las entradas.

Una vez hecho este análisis sobre cuales variables tomarían el papel de entradas y salidas del sistema y describan mejor su comportamiento se prosiguió introducirlos al software Matlab usando el paquete “Neural Network Based System Identification”.

En la figura 3.3 se muestran todos los datos de entrada ordenados para ser usados en la identificación del sistema.

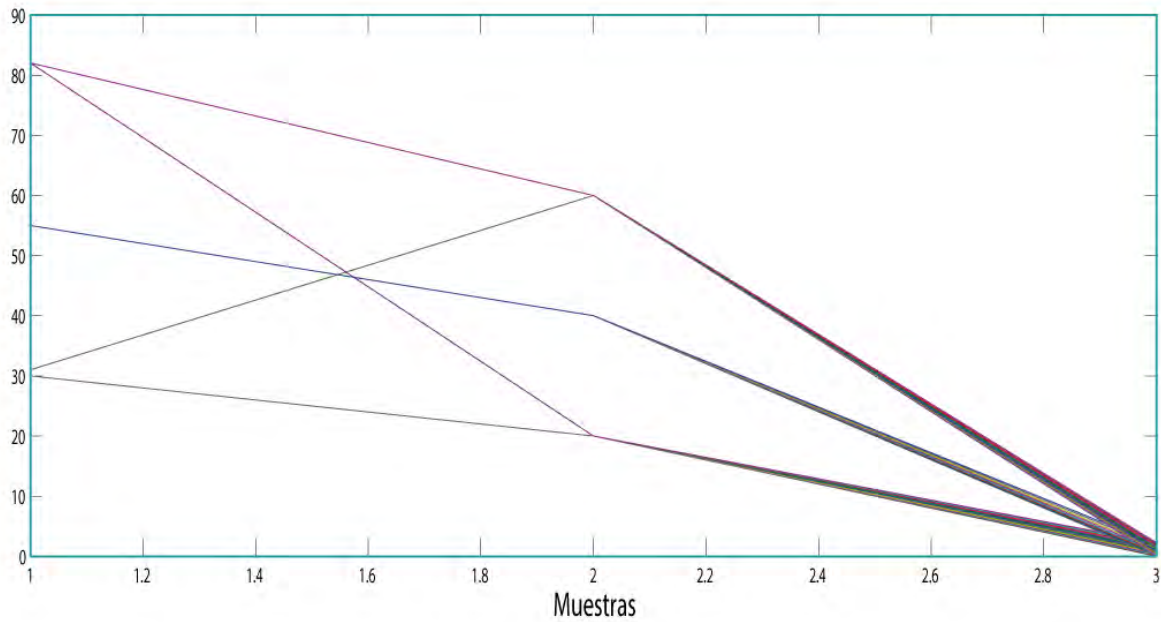
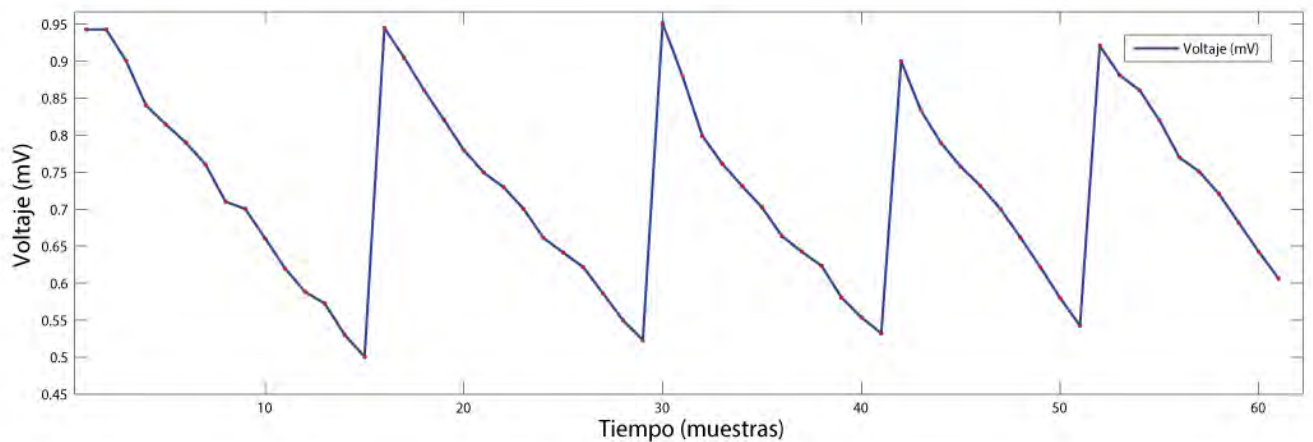


Figura 3.3. Grafica de cada uno de los datos de entrada (temperatura, presión y corriente) para la identificación.

Los datos mostrados en la figura 3.3 son usados para estimular a la red neuronal y estas ajusten sus parámetros para proveer el mejor resultado posible. En la figura 3.4 se muestra la gráfica de los valores de salida (voltaje).



3.4. Grafica de las salidas del sistema.

Los datos de figura 3.4, son el comportamiento deseado que se espera como resultado de la identificación basado en redes neuronales recurrentes. En especial, los datos de salida en las redes neuronales son usados para proveer una predicción del comportamiento eléctrico en el sistema identificado.

El siguiente paso en la definición de la estructura de la RNAR consiste en determinar el orden del sistema, el cual es descrito en la siguiente sección.

3.1.3 Determinación del orden del sistema.

La determinación del orden del sistema es utilizada para identificar el orden correcto del sistema, es decir, el número de las entradas y/o salidas pasadas. Si se seleccionan incorrectamente estos valores pueden ocasionar diversos problemas en el entrenamiento de la red neuronal. Por otra parte, si el orden seleccionado es menor al real, se pueden perder características importantes del sistema, lo cual provocaría una mala predicción como resultado. Aun tomando estas consideraciones, no siempre resulta exitosa la determinación de orden del sistema; aunque, siempre es mejor tener suficiente conocimiento físico del sistema a ser modelado [53].

El método propuesto en [41] para determinar el orden del sistema se utiliza para saber cuántas entradas y/o salidas pasadas necesita la arquitectura de red para proveer datos cercanos al real (ver figura 3.5).

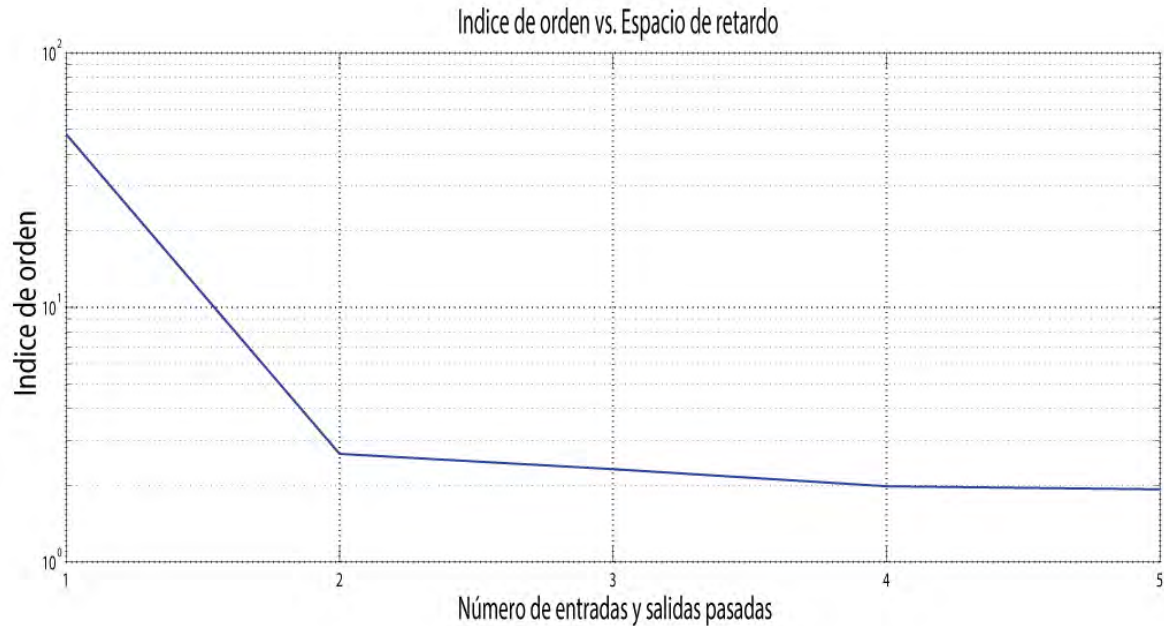


Figura 3.5. Determinación del número de entradas y/o salidas pasadas del sistema.

Aunque el algoritmo propuesto en [41] provea una gráfica como la mostrada en la figura 3.5, es complicado concluir algo seguro y aunque no parezca razonable suponer que el sistema pueda ser modelado por un modelo de segundo orden o que nos asegure un mejor rendimiento del sistema modelado, se realizaron pruebas heurísticas que permitan obtener un mejor resultado del sistema.

Con el fin de tomar todos los datos posibles del sistema, en los apartados siguientes se presenta la predicción del voltaje considerando dos casos:

- Caso uno, la predicción del voltaje con la estructura NNARX sin consideración de retroalimentación de salidas pasadas. La validación con la estructura NNOE sin consideración de la retroalimentación de las predicciones pasadas.
- Caso dos, la predicción del voltaje considerando la retroalimentación de dos salidas pasadas. Y la validación con la estructura NNOE considerando la retroalimentación de dos predicciones pasadas.

Para la etapa de entrenamiento o aprendizaje y validación del sistema a la red neuronal se le presentan un conjunto par de datos entradas-salidas, donde sus pesos sinápticos y bias son

ajustados conforme al algoritmo Levenberg-Marquardt (L-M) [54]. El entrenamiento de la red neuronal se realiza con la estructura NNARX, ya que esta estructura proporciona información más rápida y de un rendimiento más exacto por la retroalimentación de sus entradas o salidas externas reales del sistema en comparación con la estructura NNOE, que se realiza en ausencia de entradas y/o salidas reales del sistema [41]. La etapa de validación se lleva a cabo con la estructura de entrenamiento modificada a la estructura NNOE, es decir la salida de predicción es retroalimentada a la red como una salida pasada.

Con el fin de construir un modelo adecuado para la emulación del comportamiento dinámico de una celda de combustible se deben proporcionar predicciones precisas para una mejor predicción en la salida. Aunque la estructura NNOE tiene la capacidad de realizar predicciones a *varios pasos adelante*, se ha comprobado que es sensible a problemas de estabilidad y menos inmune al ruido del sistema, es por esta razón que para tener un mejor comportamiento este problema de inestabilidad se debe tratar durante la etapa de entrenamiento [55].

3.1.4 Caso 1 (sin retroalimentación de salidas pasadas)

Con el fin de conocer el comportamiento del modelo a identificar se plantea este primer caso, lo cual consta de la predicción del voltaje de la celda de combustible tipo PEM sin consideración de la retroalimentación o el retraso de las salidas pasadas al sistema.

3.1.4.1 Predicción del comportamiento eléctrico de la celda de combustible con la red neuronal NNARX

La red neuronal utilizada para esta aplicación es una de tipo Perceptrón Multicapa con una estructura no lineal y es utilizado para modelar un sistema MISO (Múltiples Entradas Una Salida por sus siglas en inglés).

La configuración de la estructura NNARX se muestra en la figura 3.6. lo cual consta de dos capas, una capa oculta o intermedia y una capa de salida. Se utilizaron 10 neuronas en la capa oculta y una neurona en la capa de salida; para las neuronas de la capa oculta se usó una de función de activación sigmoideal como la mostrada en la figura 2.3, y para la neurona de la capa de salida de una función lineal.

El error medio cuadrático considerado es de $1e-3$ y se alcanzó en la iteración 247. El vector de regresión utilizado es el que se define en 4.1, y se calculó durante la etapa de entrenamiento (algoritmo de entrenamiento y validación, apéndice B).

$$\varphi = [y(t)u_1(t-1)u_2(t-1)u_3(t-1)]^{-T} \quad (4.1)$$

En (4.1) se utilizan tres entradas pasadas y una salida actual para la estimación de la red neuronal; en este apartado no se consideró el orden del sistema calculado con el método propuesto en [41] porque puede no ser tan efectivo incluyendo más entradas o salidas al sistema y puede ser suficiente utilizando una buena elección en los datos en el entrenamiento de la red neuronal. Siguiendo con (4.1), es necesario agregar las entradas pasadas externas $u_1(k-1)u_2(k-1)u_3(k-1)$, los cuales corresponden a la Temperatura ($^{\circ}\text{C}$), la Presión (psi) y la corriente (mA), por lo que se considera un sistema MISO. Es necesario considerar un sistema de este tipo para obtener el mejor resultado del sistema, puesto que el sistema de generación de energía basado en celdas de combustible tiene un comportamiento dinámico y es por ello que se necesita contar todos los parámetros que influyan directamente en el sistema para obtener una mejor identificación del sistema estudiado. En la figura 3.6 se muestra la estructura general de la red neuronal utilizada.

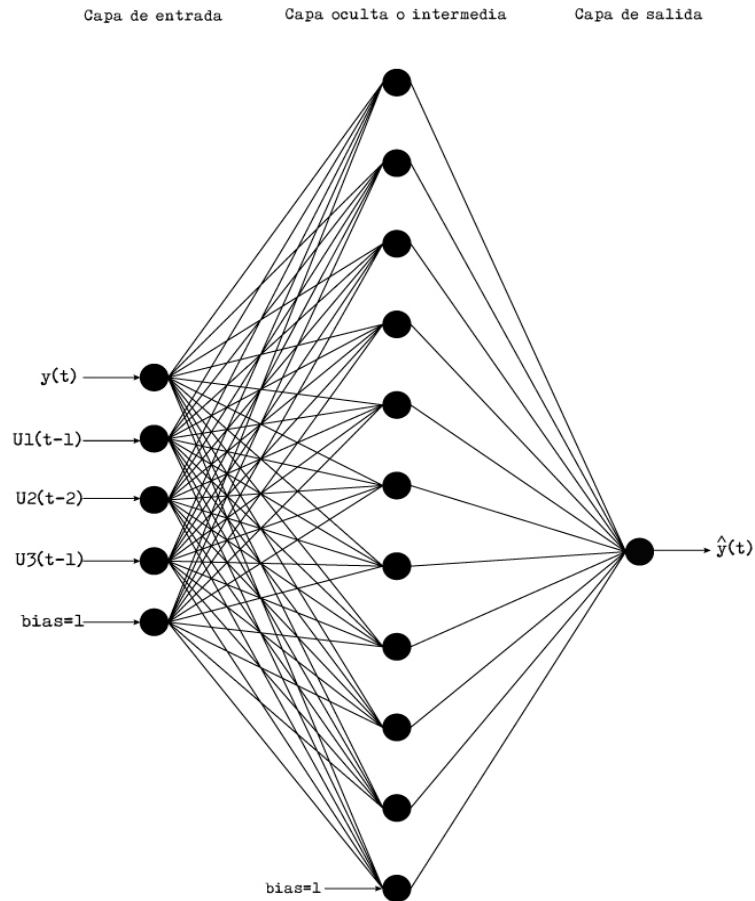


Figura 3.6. Estructura de la red utilizada.

El entrenamiento de la red neuronal se realizó fuera de línea por los datos estáticos que tenemos del sistema [52]. Una vez finalizado el entrenamiento (cuando se alcanzó el error medio cuadrático deseado) de la red neuronal NNARX con el algoritmo que se muestra en el apéndice B y se obtienen los pesos de las conexiones entre neuronas que permiten realizar la predicción del voltaje.

3.1.4.2 Validación del modelo

Con la finalidad de evaluar el aprendizaje de la RNA, se realiza la validación del sistema mediante la simulación del modelo obtenido haciendo la comparación entre su salida obtenida (la predicción del voltaje) y la salida deseada del sistema (datos del apéndice A, tabla A2). Los datos de validación son datos que no fueron utilizados durante el entrenamiento de la red neuronal.

Para la etapa de la validación se realiza en el paso 6 del algoritmo mostrado en el apéndice B. Dando como resultado la figura 3.7.

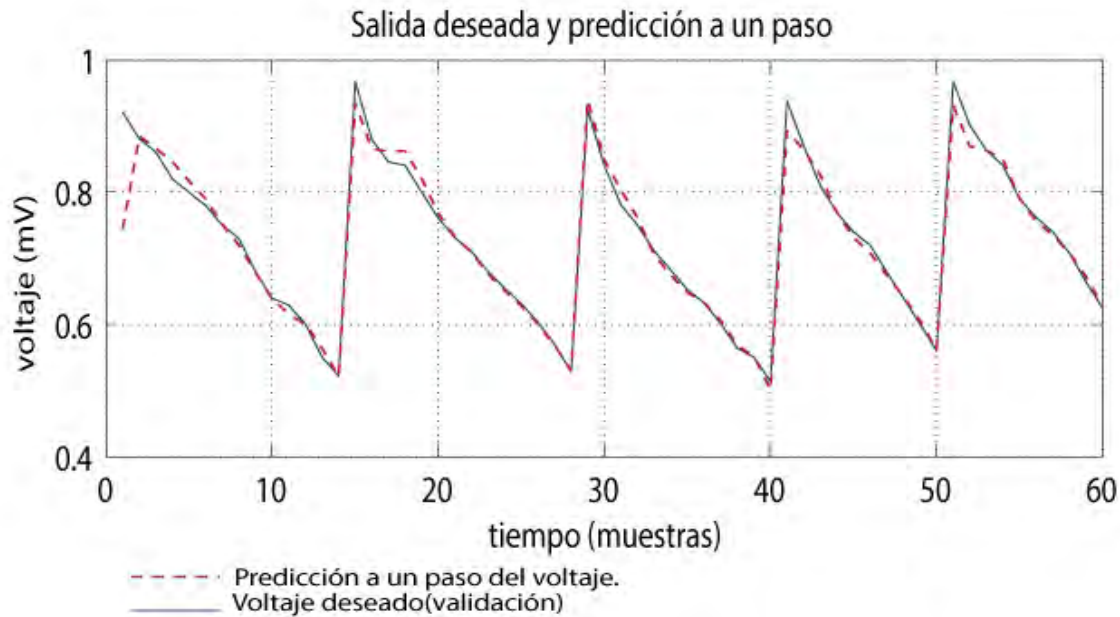


Figura 3.7 Grafica de Validación (línea azul continua) vs grafica de predicción a un paso (línea discontinua roja).

En la figura 3.7 se muestra la comparación entre las gráficas los datos de validación y de los datos de predicción del voltaje. Cabe mencionar que la estructura NNARX produce una predicción de *un paso adelante*. Esto se debe porque se requiere parámetros reales del sistema a identificar.

A continuación, la evolución del error medio cuadrático se muestra en la figura 3.8 y en la figura 3.9 la evolución del error de predicción, ambos durante el entrenamiento de la red.

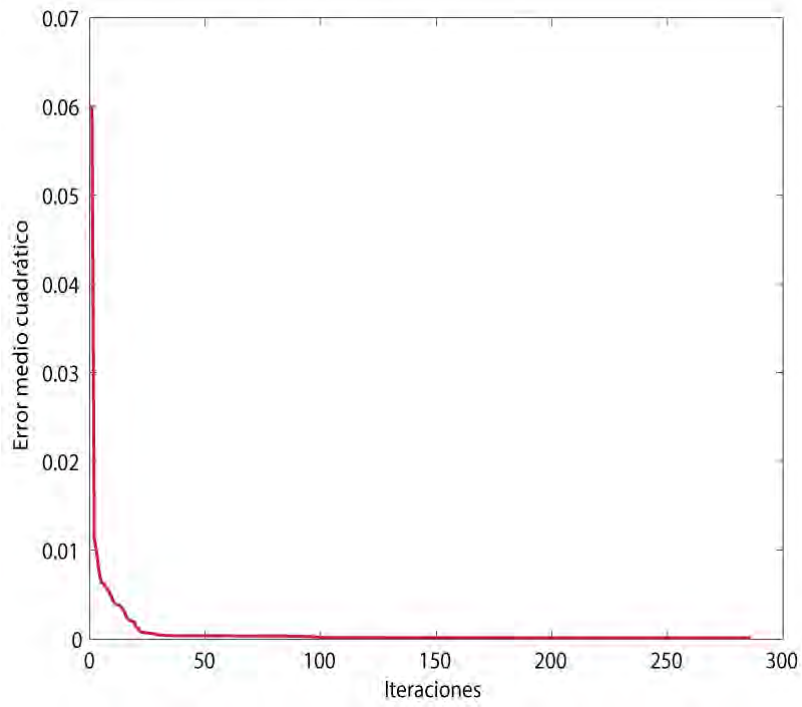


Figura 3.8. La evolución del error medio cuadrático a lo largo de las iteraciones.

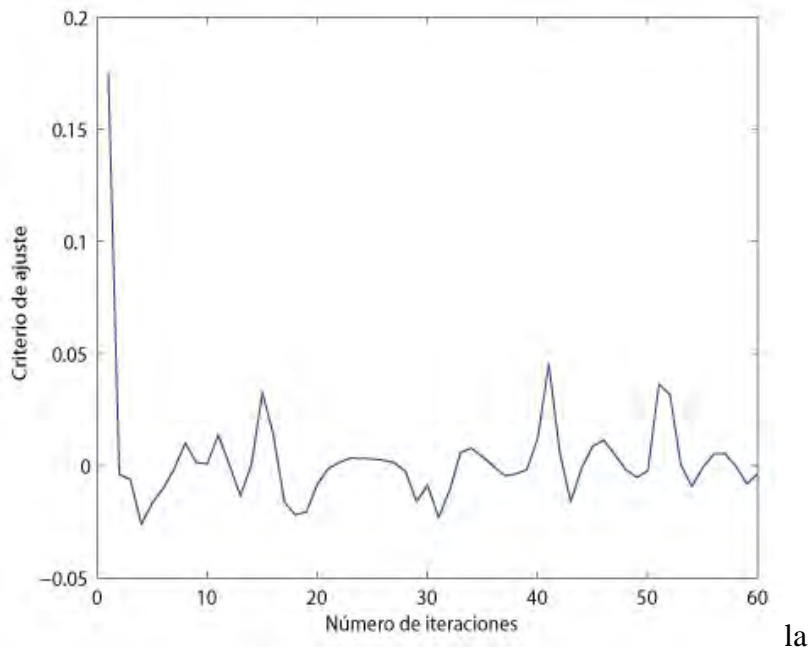


Figura 3.9. Evolución del error de predicción durante el entrenamiento de la red neuronal.

3.1.4.3 Análisis de los residuos

El análisis de los residuos permite constatar que los parámetros seleccionados para el entrenamiento de la red neuronal fueron los óptimos para el sistema y estos determinan la validez del resultado de la identificación, según la expresión:

$$\varepsilon(t) = \varepsilon(t, \theta) = y(t) - \hat{y}(t) \quad (4.2)$$

Siendo θ el vector de parámetros del modelo, $y(t)$ la respuesta real del sistema y $\hat{y}(t)$ la respuesta estimada por el modelo para la misma entrada.

Idealmente, estos residuos deben ser independientes de la entrada. Si no sucede así, significa que hay componentes en $\varepsilon(t)$ que proceden de la entrada $u(t)$, lo cual a su vez significa que el modelo no es capaz de describir completamente la dinámica del sistema.

Por otra parte, el modelo será más exacto en cuanto más se acerquen a cero los términos de la correlación. En la figura 3.10 se muestran los errores de correlación. Los errores de correlación miden la influencia de las muestras pasadas de las entradas sobre el valor presente de la salida (voltaje de la celda de combustible). Sin embargo, como la salida producida por el sistema es dinámico y variable en el tiempo (en un experimento a largo plazo) el modelo resultante por lo menos debe tener el orden del sistema calculado (orden del sistema) para cumplir con su objetivo.

En la figura 3.10 se muestra el comportamiento de cada término; estos términos se mantienen cercano al cero a pesar de que en algunos puntos logran salirse del margen de validez (líneas rojas discontinuas) y otros se quedan al cercanos al límite del margen de validez.

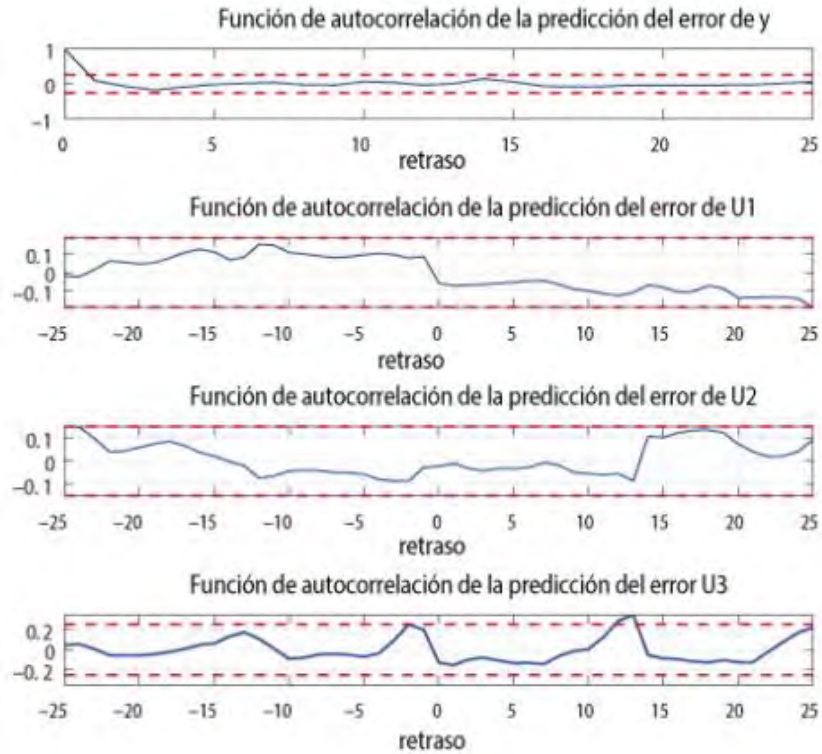
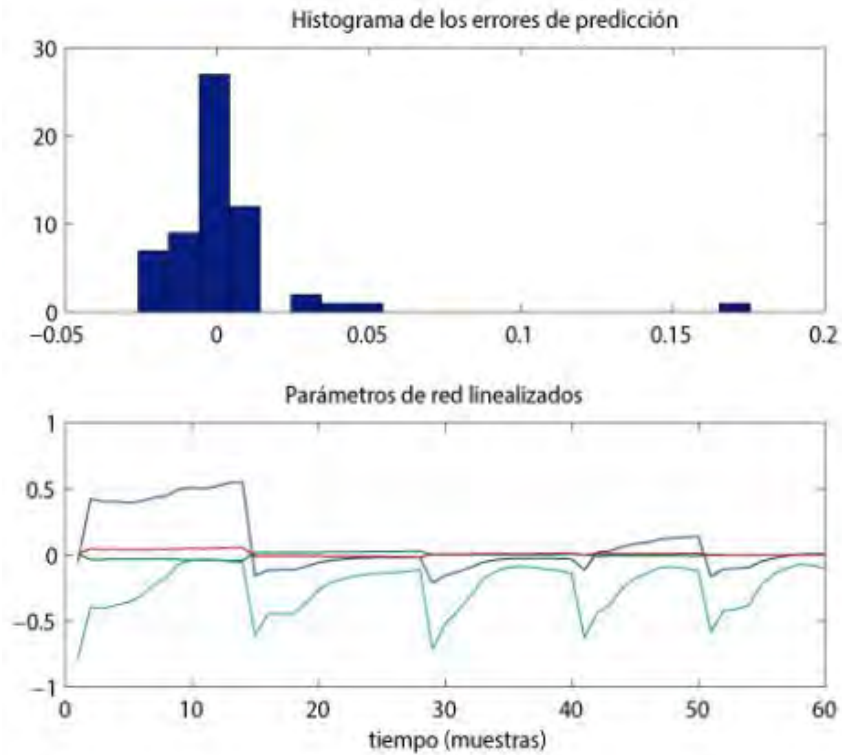


Figura 3.10. Grafica de los errores de correlación para las entradas y salida.

En la figura 3.11, se presentan el histograma de errores, y representan la distribución de los datos de las entradas y los datos de la salida. El histograma contiene la desviación estándar de las entradas y salidas pasadas los cuales se concentran alrededor de la media.



3.11. Histograma de errores de predicción.

3.1.4.4 Predicción del comportamiento eléctrico de la celda de combustible con la red neuronal con la estructura NNOE.

La estructura NNOE es utilizada para una predicción *a varios pasos adelante* utilizando los parámetros de entrenamiento de la estructura NNARX.

Con el fin de conocer como es el comportamiento eléctrico de la celda de combustible y cómo afectaría dicho comportamiento si la demanda eléctrica de la celda de combustible se prolongará más tiempo se usa la estructura actual. En la figura 3.12 se muestra la estructura de red NNOE.

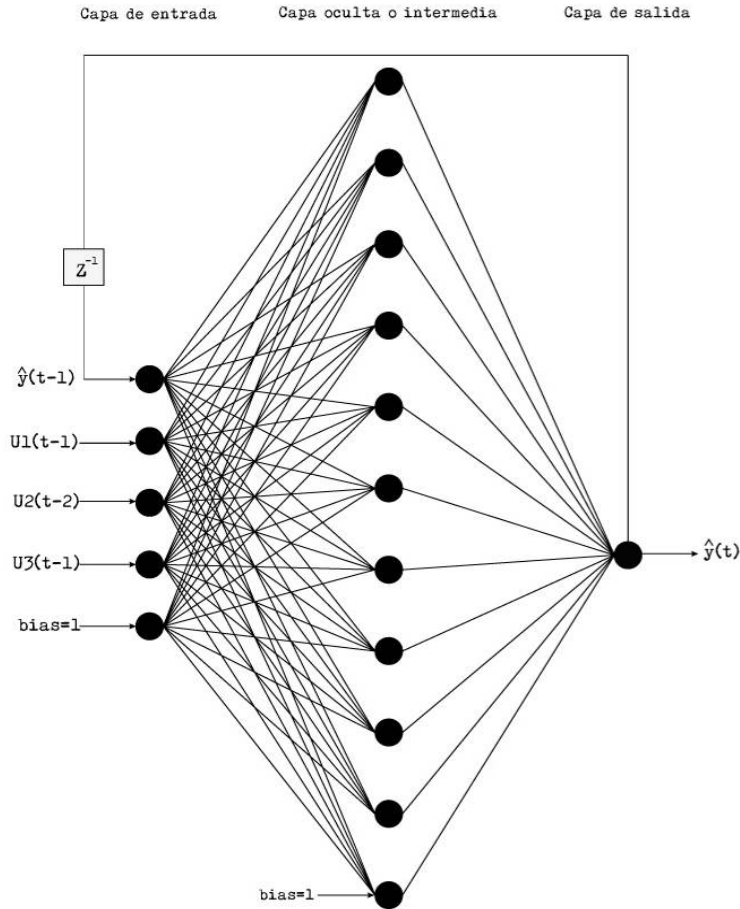


Figura 3.12. Estructura de la estructura NNOE.

Para la estructura de red NNOE, se tomaron los parámetros obtenidos en el entrenamiento de la estructura NNARX y solo se realizó la parte de la validación, por lo que la predicción es conocido como *varios pasos adelante* con una retroalimentación de la predicción $[\hat{y}(t-1)]$. Los resultados obtenidos con la red NNOE se muestran a continuación.

3.1.4.5 Validación del modelo.

A continuación, se muestran los resultados obtenidos con la red neuronal.

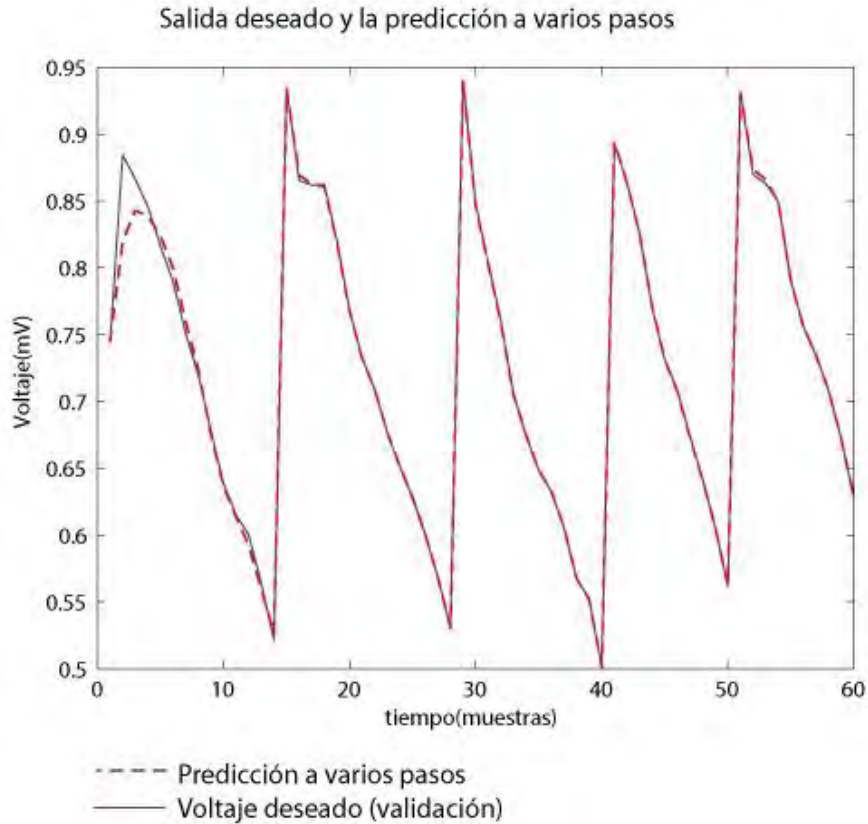


Figura 3.13. Gráfica de validación (línea azul continua) vs gráfica de predicción con la red NNOE (línea discontinua roja).

En la figura 3.13 se muestra la comparación entre las gráficas los datos de validación (nuevos datos) y de los datos de predicción del voltaje. También se puede apreciar que el comportamiento de la estructura actual es más similar con el que se compara y esto es logrado al hacer la retroalimentación de su predicción, de manera que haciendo esto sus parámetros tienen más oportunidad de tomar valores cercanos a los datos originales.

3.1.4.6 Análisis de los residuos

Las gráficas de correlación de los términos se muestran en la figura 3.14, y en ellas se pueden apreciar que tienen un mejor comportamiento de manera que se encuentran con más forma dentro del margen de validez.

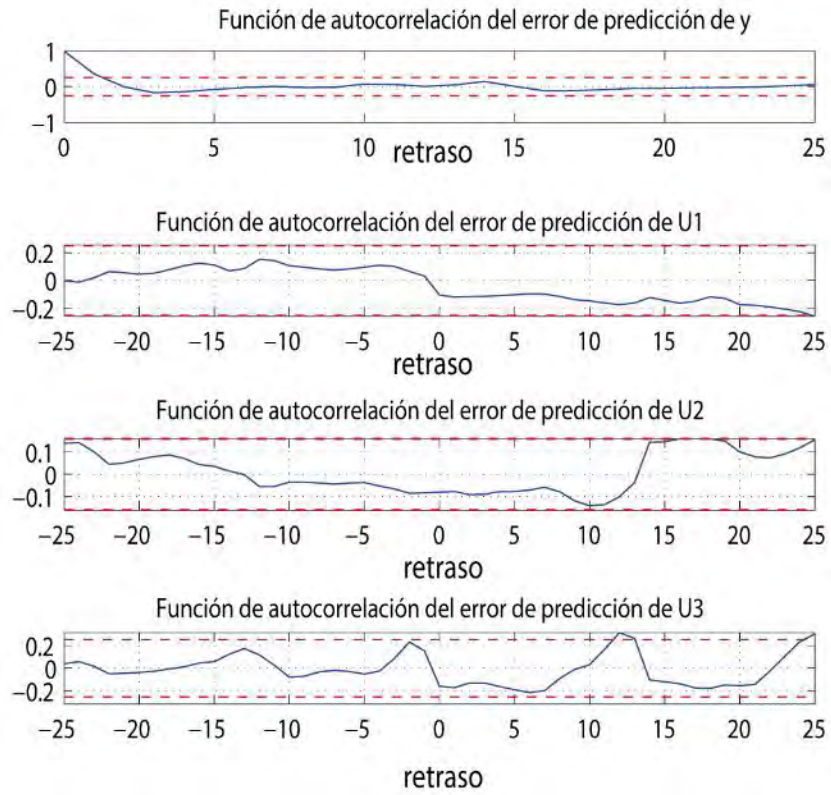


Figura 3.14. La gráfica de los errores de correlación para entradas y salida.

En la figura 3.15, se muestra el histograma de errores donde la distribución de los datos de las entradas y de las salidas son más cercanos a la media en comparación con la estructura de red NNARX.

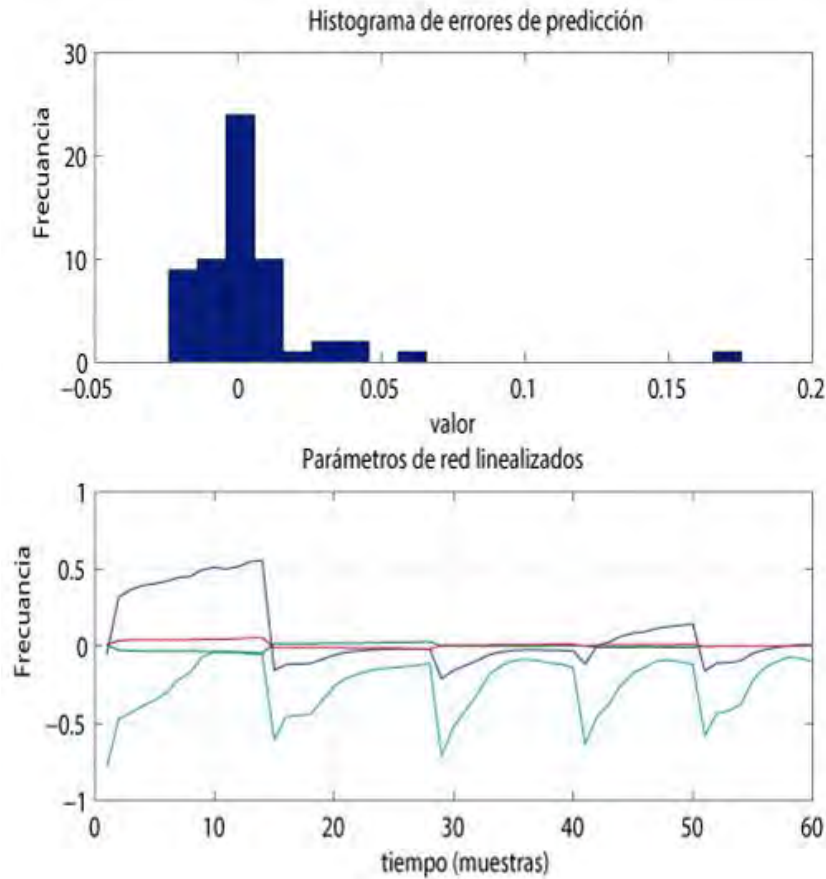


Figura 3.15. Histograma de errores de predicción.

3.1.5 Caso 2 (considerando la retroalimentación de dos salidas pasadas)

Para el segundo caso se consideran dos salidas pasadas en retroalimentación del sistema. En este apartado se realizan los mismos pasos que en caso 1 y utilizando las estructuras de red NNARX y NNOE, para el entrenamiento y validación respectivamente, pero esta ocasión tomando en cuenta dos retrasos de la salida actual para la red neuronal NNARX y en la validación con estructura NNOE tomando en consideración la retroalimentación de la predicción atrasada dos veces.

En resumen, para este caso 2 tenemos: una salida actual y dos en retraso de la misma con tres entradas pasadas para la red neuronal NNARX. Para la red neuronal NNOE una predicción actual y dos en retraso, utilizando tres entradas pasadas.

Para que nuestro sistema logre estar lo suficientemente caracterizado surge la necesidad de considerar el orden del sistema para un mejor modelo resultante. De acuerdo con el orden calculado

al menos debe contener de dos entradas y/o salidas retrasadas de la salida actual aplicadas al sistema, sin embargo, es posible agregarle más de dos entradas o salidas si se requiere.

3.1.5.1 Predicción del comportamiento eléctrico de la celda de combustible con la red neuronal NNARX utilizando dos salidas pasadas

La configuración usada para este apartado de la estructura de la red NNARX se muestra en la figura 3.16. En esta configuración se utilizaron 10 neuronas en la capa oculta y una neurona en la capa de salida; para la capa oculta se usó una función de activación tangente hiperbólica para cada neurona, y en la neurona de salida, se usó una función de activación lineal.

En esta configuración, se utilizó un error medio cuadrático $1e-8$ y se alcanzó en la iteración 500 y el vector de regresión utilizado es el que se define en 4.3.

$$\varphi = [y(t-1) y(t-2) u_1(t-1) u_2(t-1) u_3(t-1)]^{-T} \quad (4.3)$$

En (4.3) se utilizan dos salidas pasadas $y(t-1) y(t-2)$, y para la estimulación de la red neuronal se usaron tres entradas externas $u_1(t-1) u_2(t-1) u_3(t-1)$, los cuales corresponde a la Temperatura ($^{\circ}\text{C}$), la Presión (psi) y la Corriente (mA).

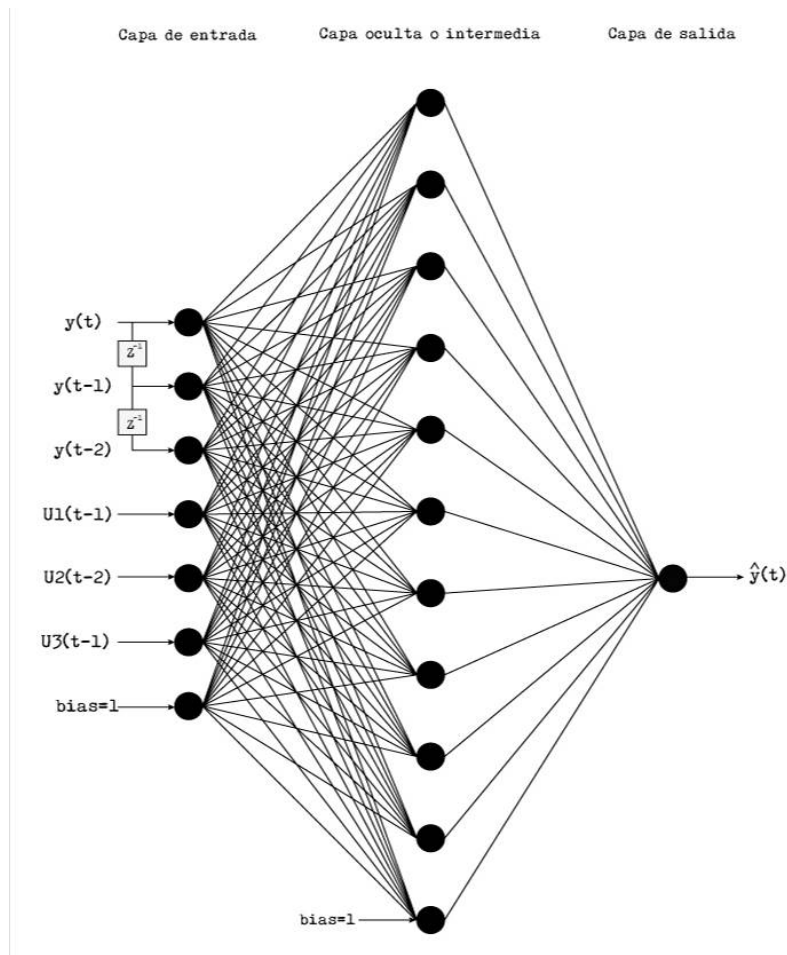


Figura 3.16. Estructura de la red neuronal.

De la misma manera que en el caso anterior, el entrenamiento se realizó fuera de línea para los datos estáticos que tenemos del sistema estudiado en [52]. Una vez que se alcanzó el error medio cuadrático o concluido con las iteraciones establecidas con el algoritmo del apéndice E se realiza la predicción y seguidamente se hace la validación del modelo obtenido.

3.1.5.2 Validación del modelo

Se hace la comparación entre los datos de validación y los datos de la salida deseada. Dando como resultado la figura 3.17.

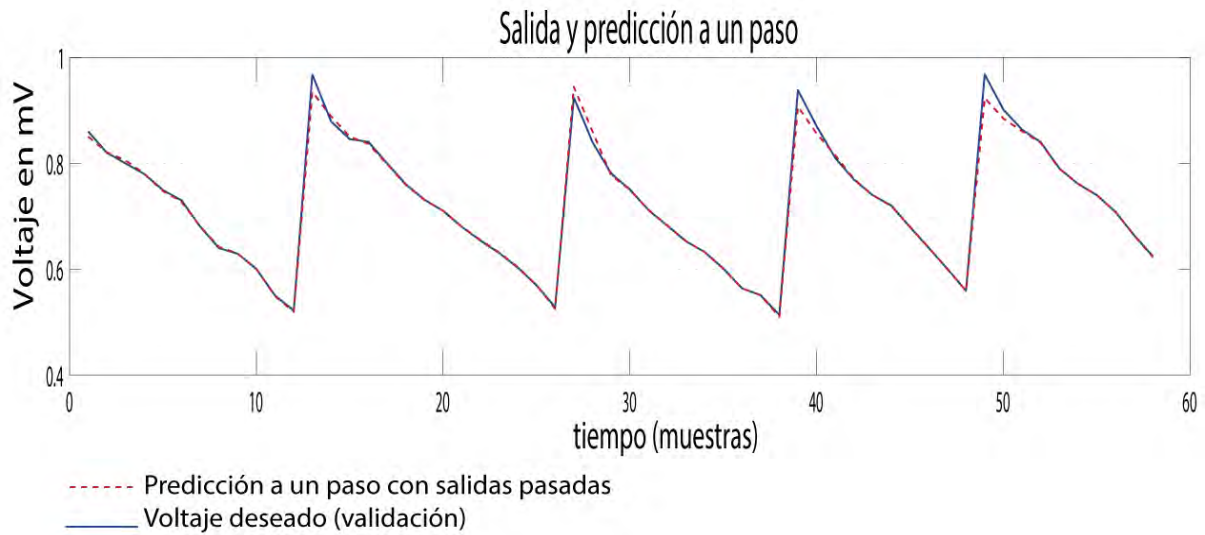


Figura 3.17. Grafica de Validación (línea azul continua) vs grafica de predicción a un paso (línea discontinua roja).

Al utilizar más componentes en el entrenamiento de la red neuronal ocasiona una mejor estimulación en los parámetros internos del sistema que genera en consecuencia una mejor respuesta en la predicción del voltaje y más cercanos al voltaje deseado (datos de validación). Para complementar los resultados mostrados en la figura 3.17 se presentan las gráficas de la evolución del error a lo largo de entrenamiento y las gráficas de los errores de correlación en las figuras 3.18 y 3.19, respectivamente.

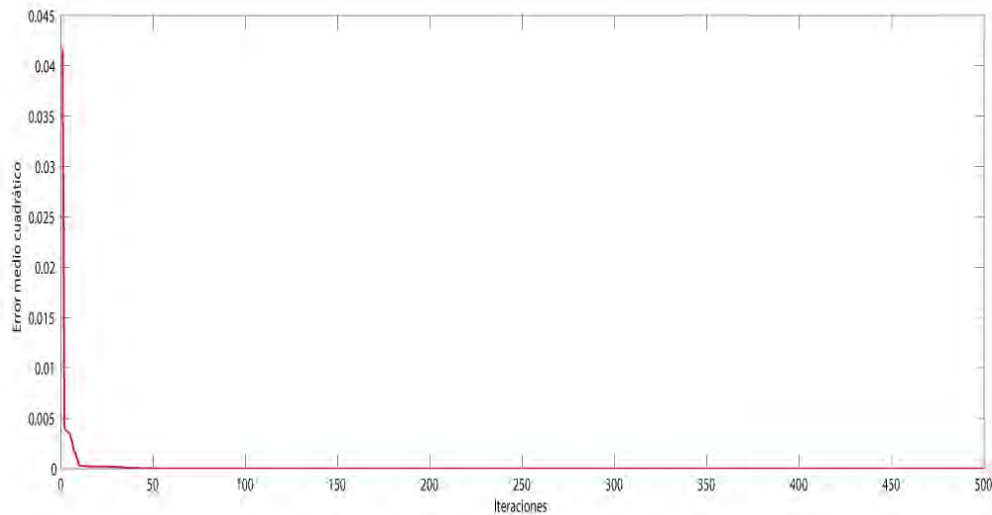


Figura 3.18. Evolución del error medio cuadrático a lo largo de las iteraciones.

Cuando se utilizan más salidas o entradas pasadas en el sistema, se necesitan más iteraciones para llegar al error medio cuadrático propuesto, pero, sí en el comportamiento de la predicción llegará a estar muy cercano al deseado, entonces no es necesario agregarle más iteraciones durante el entrenamiento. Es necesario mencionar que se debe al menos entrenar a la red neuronal dos veces y máximo tres, para no llegar a un sobre entrenamiento y que exista un sesgo en los valores resultantes.

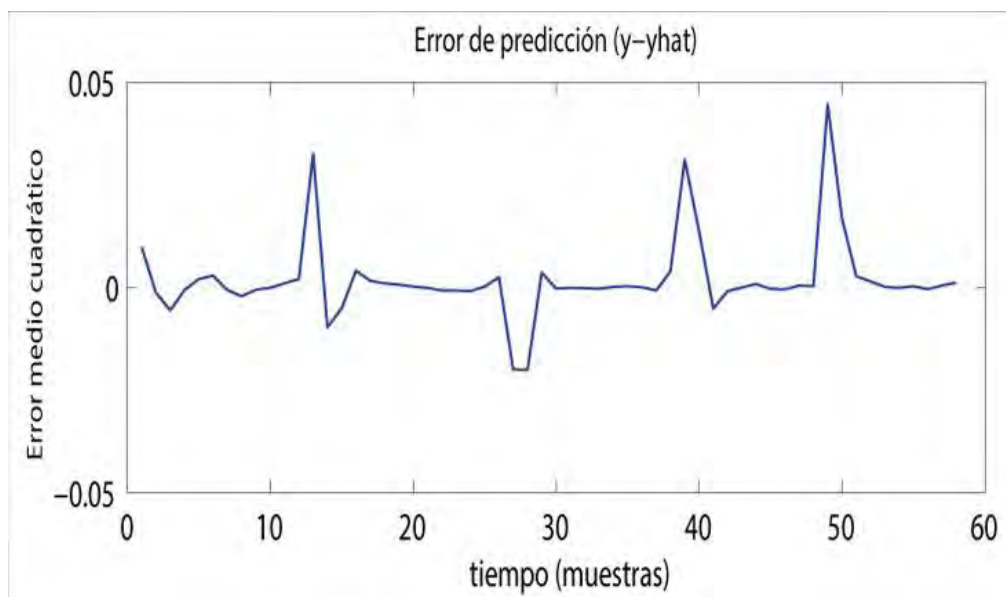


Figura 3.19. Evolución del error de predicción durante el entrenamiento de la red neuronal.

3.1.5.3 Análisis de los residuos.

En la figura 3.20 los errores de correlación mantienen un comportamiento más cercano al cero dentro del margen de validez (líneas rojas discontinuas), esto es resultado de considerar a las salidas pasadas en el entrenamiento por lo que, en la predicción muestra un mejor comportamiento. Los errores de correlación pueden mejorarse si se usan más salidas pasadas y/o entradas pasadas en el entrenamiento si se desea. En la figura 3.21 se muestra el histograma de errores que de igual manera tienen más concentración en la media por lo que para este sistema es suficiente haber utilizado las dos salidas pasadas para complementar su comportamiento.

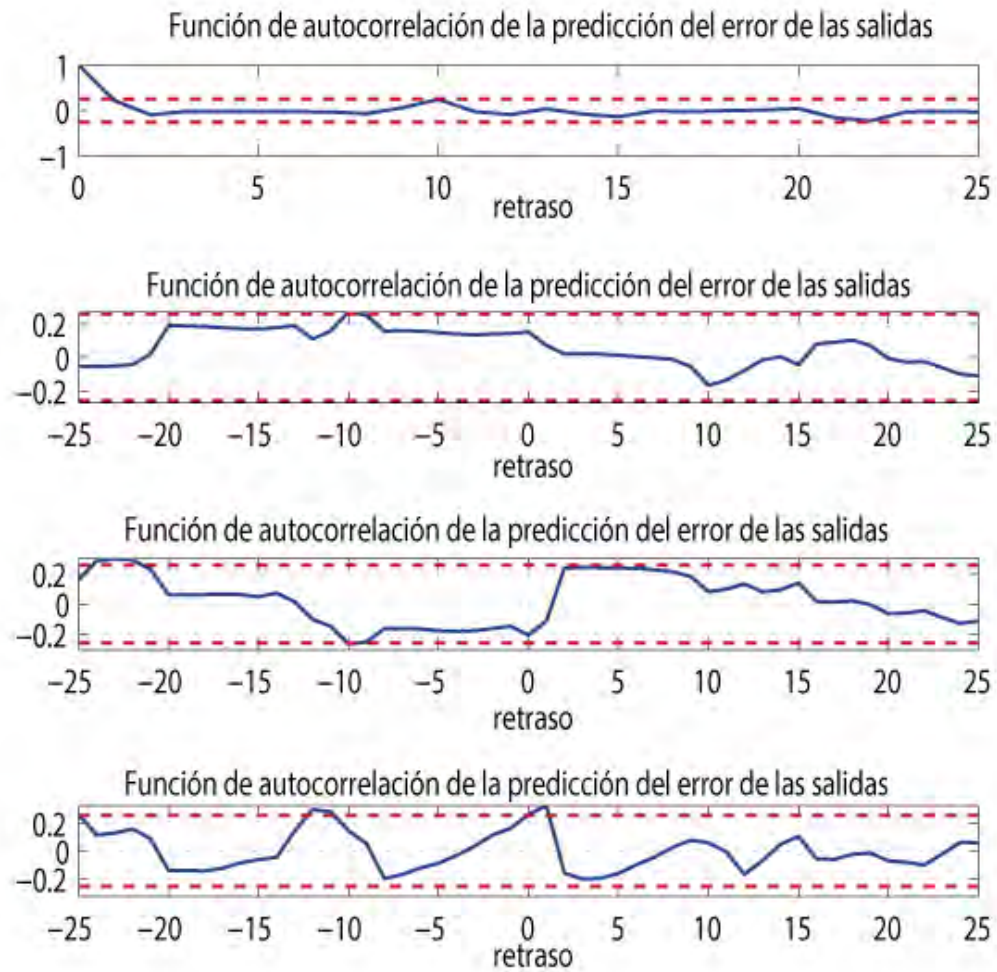


Figura 3.20. Gráfica de los errores de correlación para las entradas y salidas.

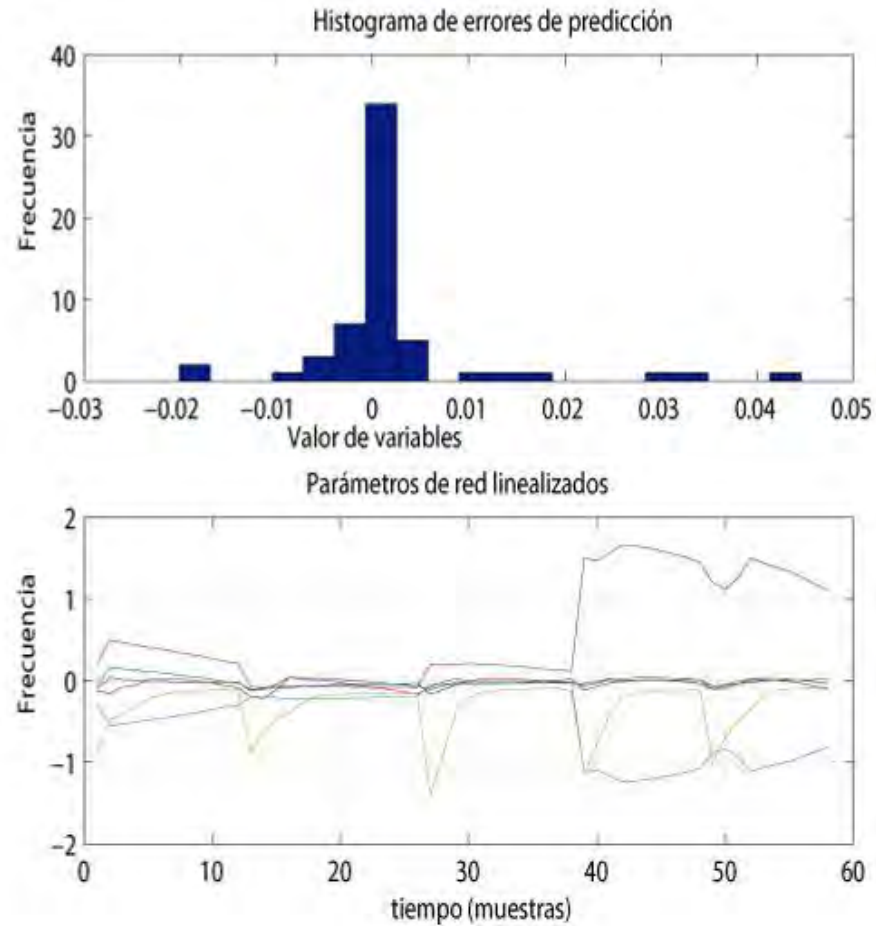


Figura. 3.21. Histograma de errores de predicción.

3.1.5.4 Predicción del comportamiento eléctrico de la celda de combustible con la red neuronal con la estructura NNOE utilizando dos salidas pasadas

Con la estructura NNOE se realiza la retroalimentación de la predicción en una salida y se atrasa en el tiempo dos veces (dos retardos unitarios). En la figura 3.22 se muestra la estructura de la red NNOE (algoritmo de validación del apéndice G).

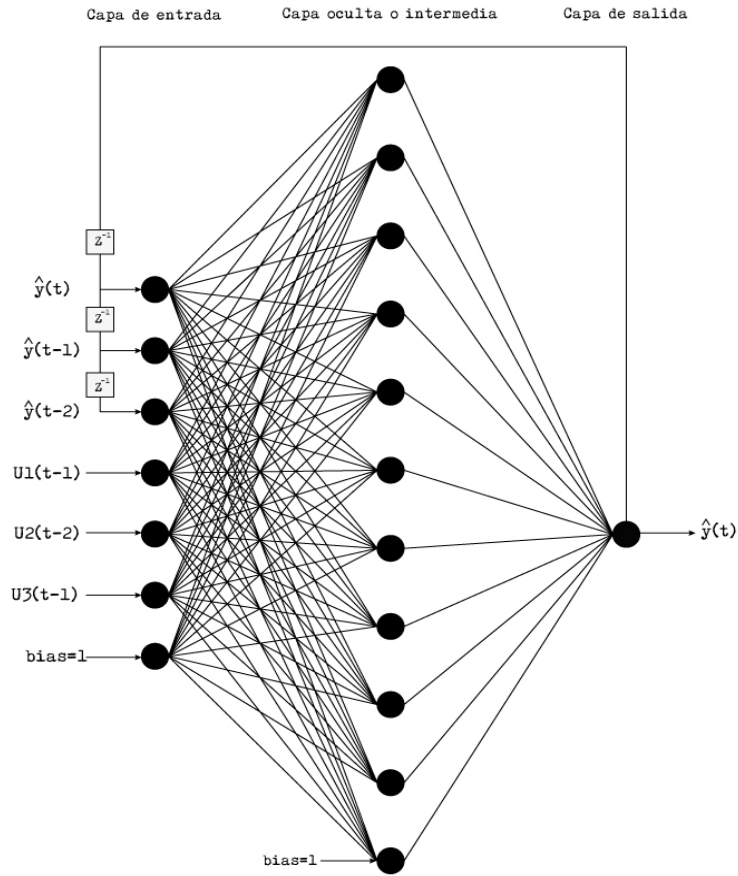


Figura 3.22. Estructura de red NNOE con retroalimentación y retrasada dos veces.

Al realizar la validación con la estructura actual tenemos los siguientes resultados.

3.1.5.5 Validación del modelo

En la siguiente figura 3.23 se muestran las comparaciones entre las gráficas de la salida deseada y de la predicción del voltaje. Se ha mencionado que para este caso 2, la retroalimentación de la predicción y del atraso de la misma en el sistema juega un papel importante en el comportamiento de la predicción del voltaje.

En la figura 3.23 se muestra la mejor predicción obtenida. Los resultados revelan que el modelo NNOE proporciona una mejor predicción utilizando de los parámetros calculados con el método propuesto en [41].

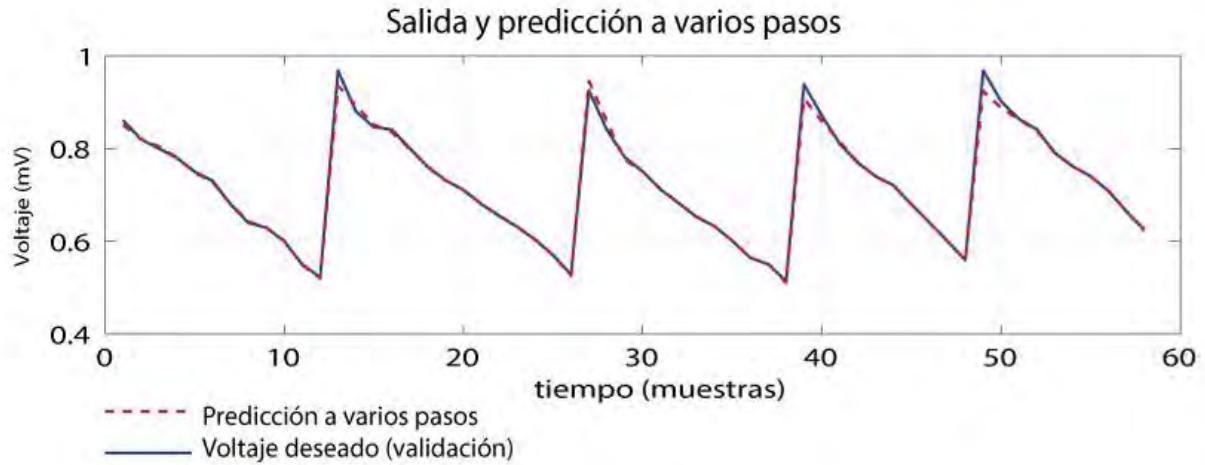


Figura 3.23. Gráfica de la salida (datos de validación, línea azul continua) vs la predicción a varios pasos (línea roja discontinua).

En complemento a la validación del sistema se muestran las figuras de la evolución del error de predicción y de los errores de correlación en las figuras 3.24 y 3.25, respectivamente.

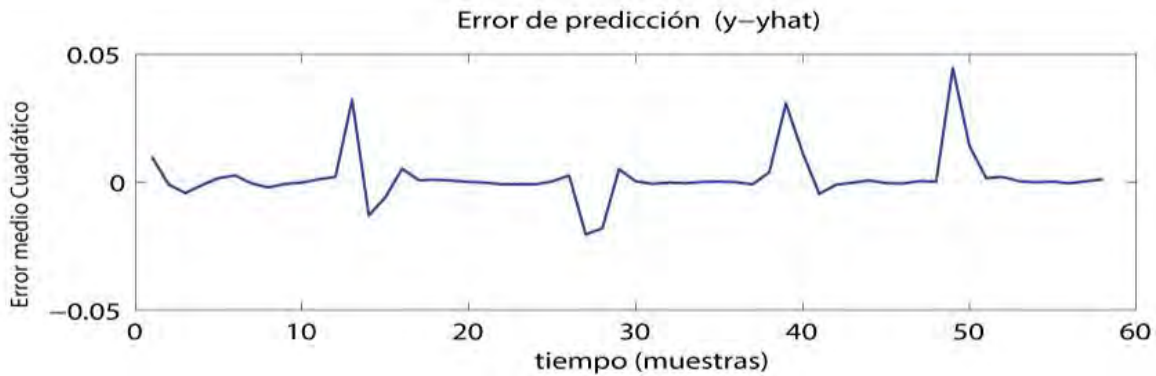


Figura 3.24. Evolución del error de predicción.

3.1.5.6 Análisis de los residuos

En la figura 3.24 se muestran los errores de correlación y estos se encuentran dentro del margen de validez (líneas rojas discontinuas) y es debido a la consideración de las salidas pasadas para este apartado.

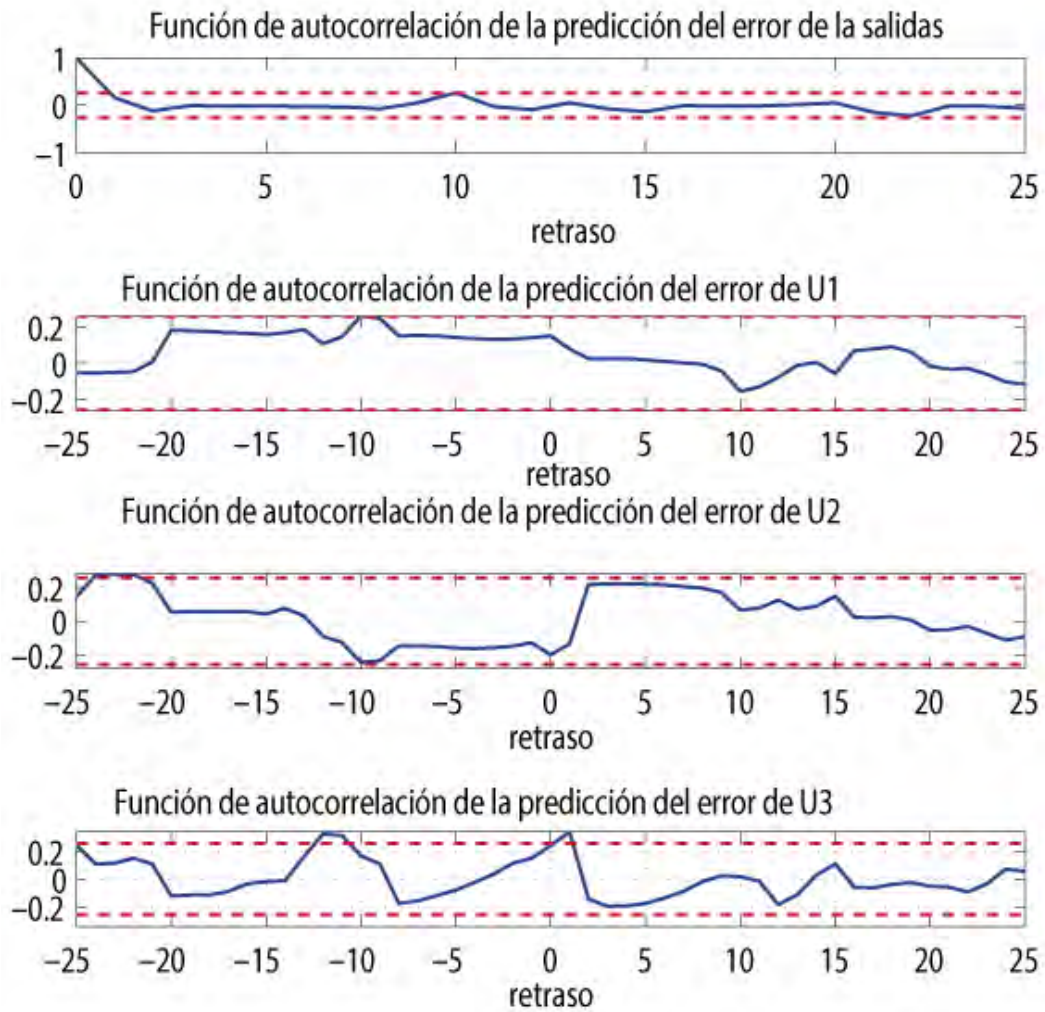


Figura 3.24. Gráfica de los errores de correlación para las entradas y salidas.

En la figura 3.25, la media y la desviación estándar muestran como son la distribución de los datos de las entradas y las salidas para la estructura NNOE que están más concentrados a la media.

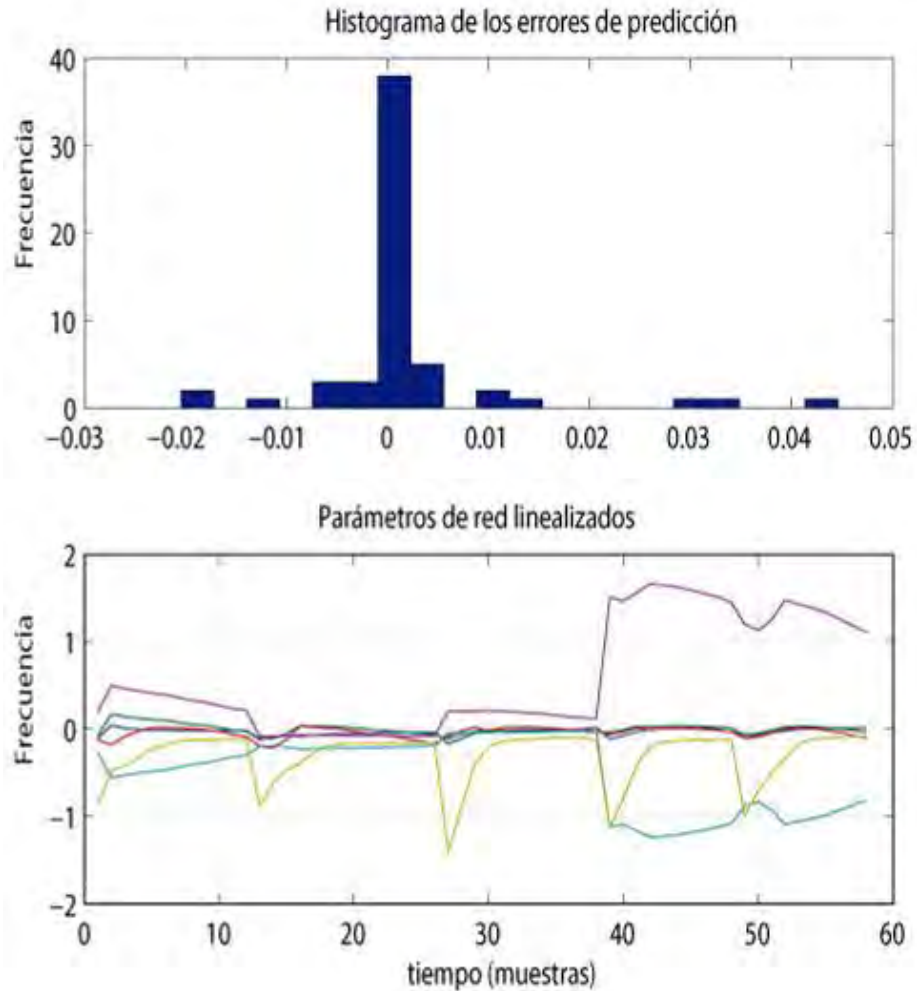


Figura 3.25. Histograma de errores de predicción.

El rendimiento causado por el ajuste del orden del sistema y el número de iteraciones del algoritmo muestra un importante resultado en las predicciones y el comportamiento en general, de manera que en conjunto estos modelos son capaces de proveer un esquema completo del sistema real. Se puede concluir que el modelo resultante utilizando estas dos estructuras de red son lo suficientemente completos para caracterizar un sistema de generación de energía con celdas de combustible con un comportamiento dinámico.

3.2 Desarrollo del emulador

En esta sección se describe el desarrollo del emulador, definiendo cada uno de los elementos que se utilizaron. Se describe primero la estructura interna de la programación realizada en la

plataforma Labview. Posteriormente se hace mención de la etapa de la salida de las señales por parte del sistema embebido basado en la tarjeta NI myRIO.

3.2.1 Programación de la red neuronal

La programación en Labview es creado mediante un proyecto conformado en dos partes: La primera parte consta de un VI alojado en el procesador de la computadora porque contiene más funciones y operaciones a utilizar, este VI realiza la lectura de los parámetros de la red neuronal. La segunda parte del proyecto consta otro VI alojado en el procesador de la tarjeta NI myRIO y este juega un papel importante porque contiene un lazo cerrado llamado *Control & simulation* que realiza las iteraciones para que la red neuronal NNARX-NNOE pueda realizar las operaciones pertinentes con los patrones de validación y los pesos sinápticos. Este segundo VI corre en tiempo real y contiene otros VI's dentro de él, o también conocidos como SubVI's. La estructura del proyecto se muestra en la figura 3.26.

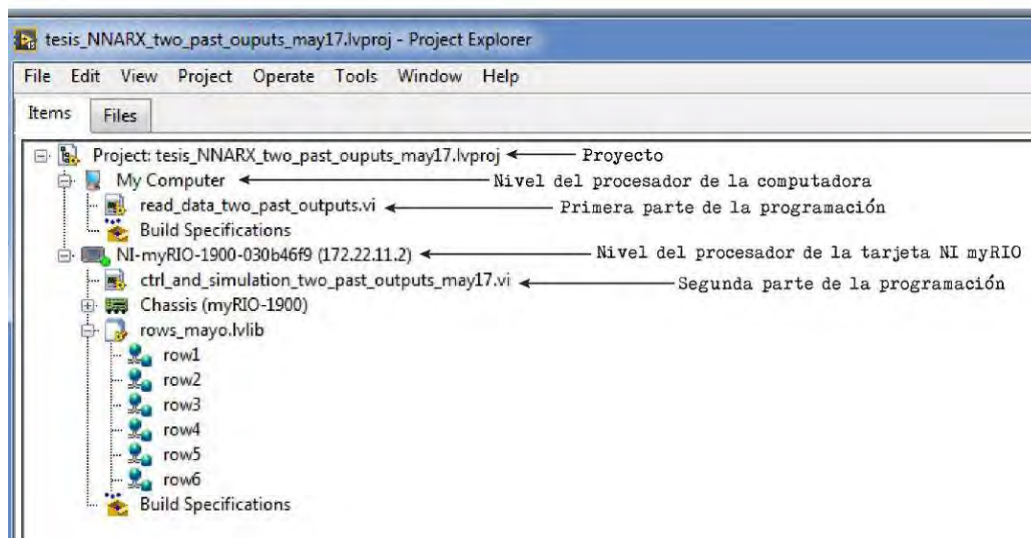


Figura 3.26. Estructura de los algoritmos del proyecto.

La programación de Labview comienza a partir de la lectura de los datos por parte del primer algoritmo (primer VI) de la figura 3.27 del vector PHI. Estos datos están ordenados en una matriz transpuesta para facilitar el manejo por el algoritmo.

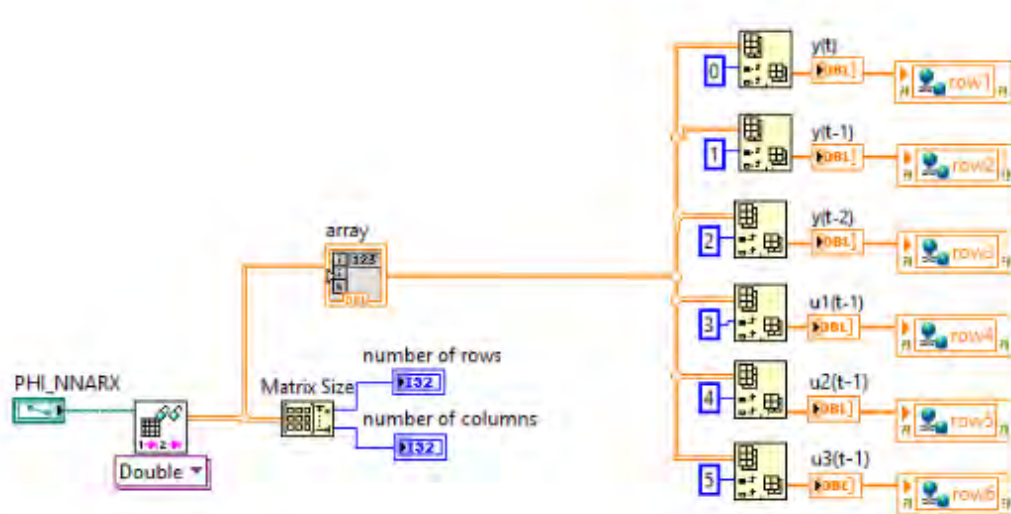


Figura 3.27. Primera parte de la programación en Labview (Lectura de datos de PHI).

En esta primera parte de la programación se leen los patrones de validación (apéndice A, tabla A2) como archivos tipo .txt, estos patrones de validación son los mismos que fueron utilizados en el apartado de la identificación del sistema. El programa lee el archivo .txt y lo descompone en vectores fila para posteriormente almacenarlos en *variables compartidas*, esto para que puedan ser introducidas a nivel del procesador de la tarjeta FPGA o al nivel del procesador de la computadora según se requiera.

La segunda parte de la programación muestra VI (figura 3.28), que se aloja en el procesador de la tarjeta FPGA NI myRIO y las funciones del programa son extraer cada dato de las *variables compartidas* en cada iteración.

Este VI tiene un panel frontal donde se muestran tablas de los datos que son leídos por el programa, mismos que se van introduciendo a un SubVI que juega el papel de la red neuronal y finalmente se crea una tabla en el panel frontal donde se alojan los datos de la predicción de la red neuronal, así como su gráfica.

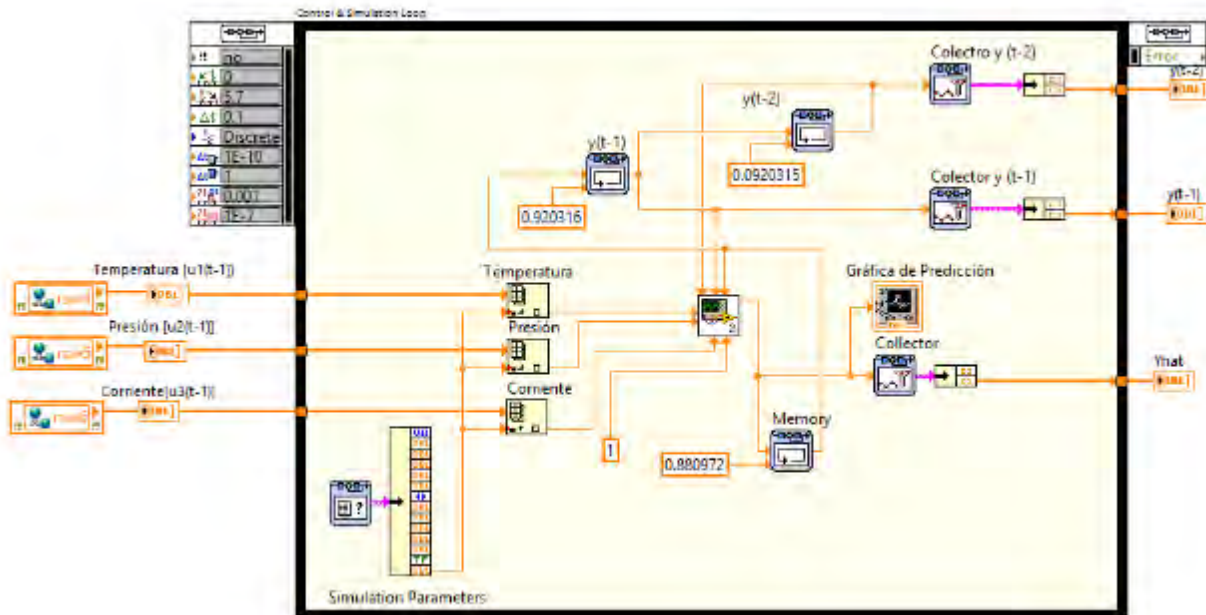


Figura 3.28. Segunda parte de la programación. (localizado al nivel del procesador de la tarjeta FPGA NI myRIO).

El programa de la figura 3.28 contiene un SubVI que toma el papel de red neuronal que a su vez dentro de esta contiene otros subVI's y en cada iteración los datos que son extraídos de las *variables compartidas* son introducidos a estos otros subVI's de la red neuronal. Los subVI's que están dentro del SubVI de la red neuronal toman el papel de las neuronas de la capa oculta y de la capa de salida.

Las conexiones que tienen estos subVI's unos con otros son la representación de la sinapsis y sirven para hacer las operaciones pertinentes de suma y multiplicación de los pesos sinápticos con sus funciones de activación.

En la figura 3.29 se muestran los subVI's que representan a las neuronas y sus conexiones con los vectores de entradas y salidas del sistema.

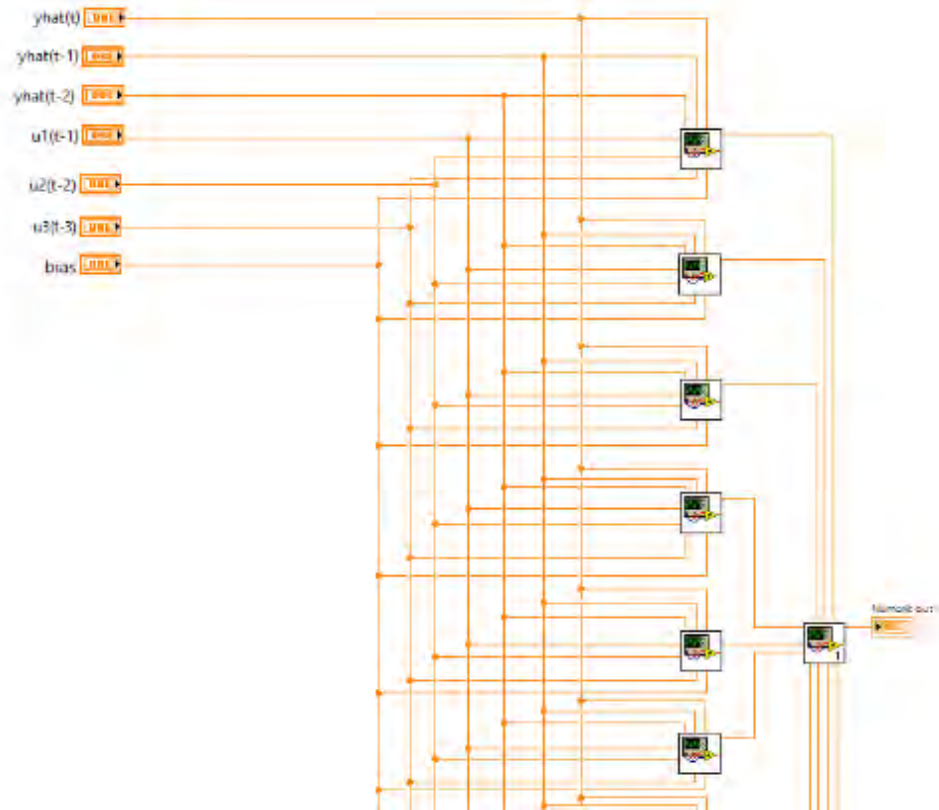


Figura 3.29. Estructura de la red neuronal (Capa oculta, 11 neuronas y para la capa de salida, una neurona).

Cada subVI que representa a una neurona contiene una estructura como se muestra en la figura 3.30, y representa las operaciones que se deben llevar a cabo, para esto, cada salida y entrada son multiplicados por su respectivo peso sináptico (apéndice C, peso de la capa oculta $W1$).

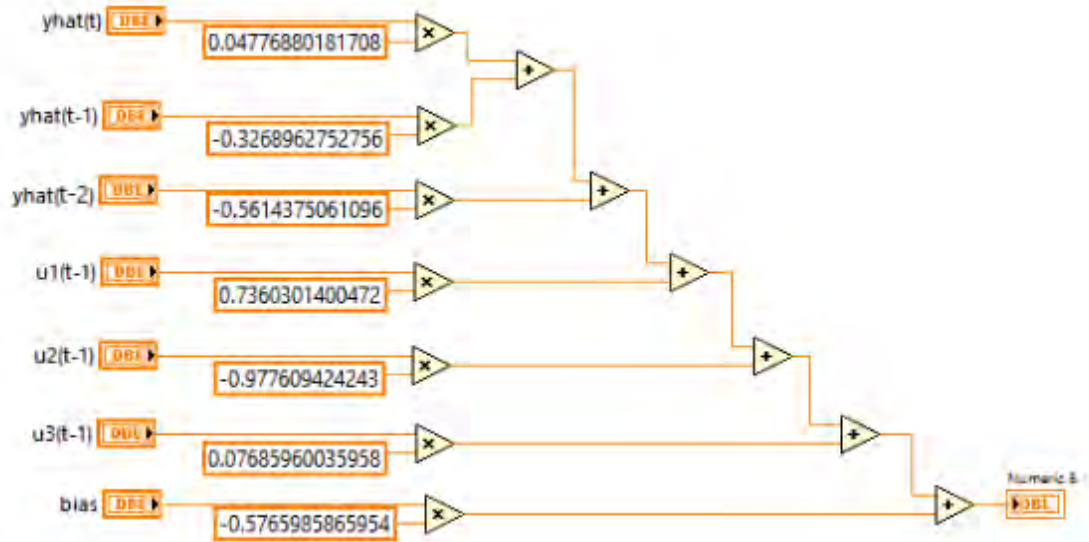


Figura 3.30. Estructura interna de cada neurona.

Y como parte final se muestra la parte frontal del VI principal (figura 3.31), que consta de vectores fila y una gráfica, donde al finalizar las iteraciones se gráfica el comportamiento final de la predicción del voltaje.

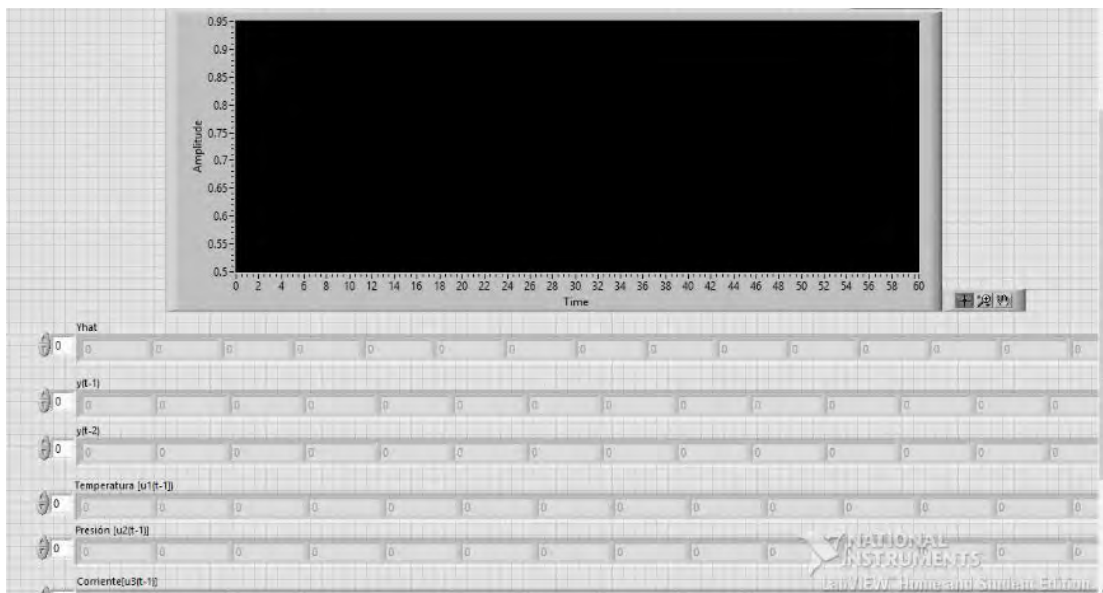


Figura 3.31. Panel frontal de la segunda parte de la programación.

En la figura 3.32 se muestra la implementación del sistema, que consta de un ordenador conectado a la tarjeta FPGA y un osciloscopio para medir la señal de voltaje de salida que provee la misma tarjeta myRIO.

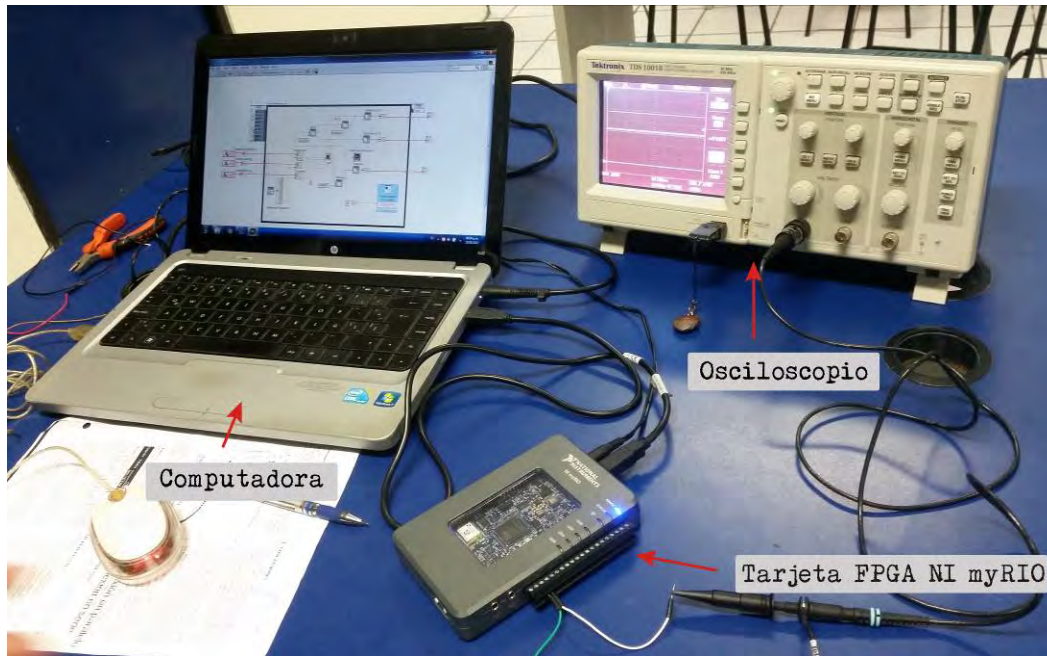


Figura 3.32. Implementación del sistema.



Figura 3.33. Tarjeta FPGA NI myRIO.



Figura 3.34. Osciloscopio.

Capítulo 4

4 Resultados

4.1 Resultados de simulación y experimentales

Con base a la metodología en el apartado anterior de la identificación del sistema y la emulación del sistema, se presentan en esta sección los resultados de simulación y experimentales de la predicción del comportamiento estacionario o transitorio de las celdas de combustible tipo PEM. El modelo neuronal que se obtuvo presentó las siguientes características:

- Se fundamenta con principios teóricos y es basado mediante datos experimentales reales de una celda de combustible.
- Define la operación de una celda de combustible tipo PEM.

4.1.1 Resultados de la identificación del sistema.

En la figura 4.1 se muestra el resultado de la predicción del voltaje a *un paso adelante*, con base al orden calculado del sistema y considerando las salidas pasadas en atraso para la estructura NNARX, esto ocasionó que el resultado fuera más cercano al deseado, por lo que el modelo identificado es capaz de predecir el comportamiento en estado estacionario o dinámico de las celdas de combustible tipo PEM. Los resultados de validación de la estructura NNARX, ofrecen más confiabilidad para realizar la parte experimental.

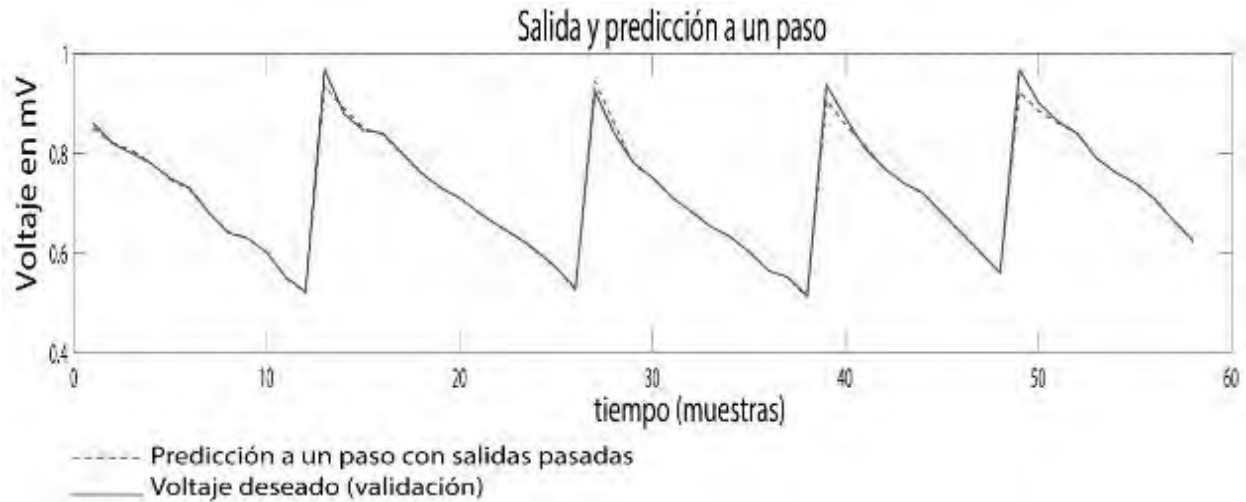


Figura 4.1. Gráfica de la predicción comportamiento del voltaje (línea roja discontinua) vs gráfica de del comportamiento deseado del sistema basado en celdas de combustible tipo PEM para la estructura NNARX.

En la figura 4.2 se muestran los errores de correlación los cuales se encuentran dentro del margen de validez (líneas rojas discontinuas) y se consideran aceptables. Por último, se muestra la predicción del voltaje con la estructura NNOE.

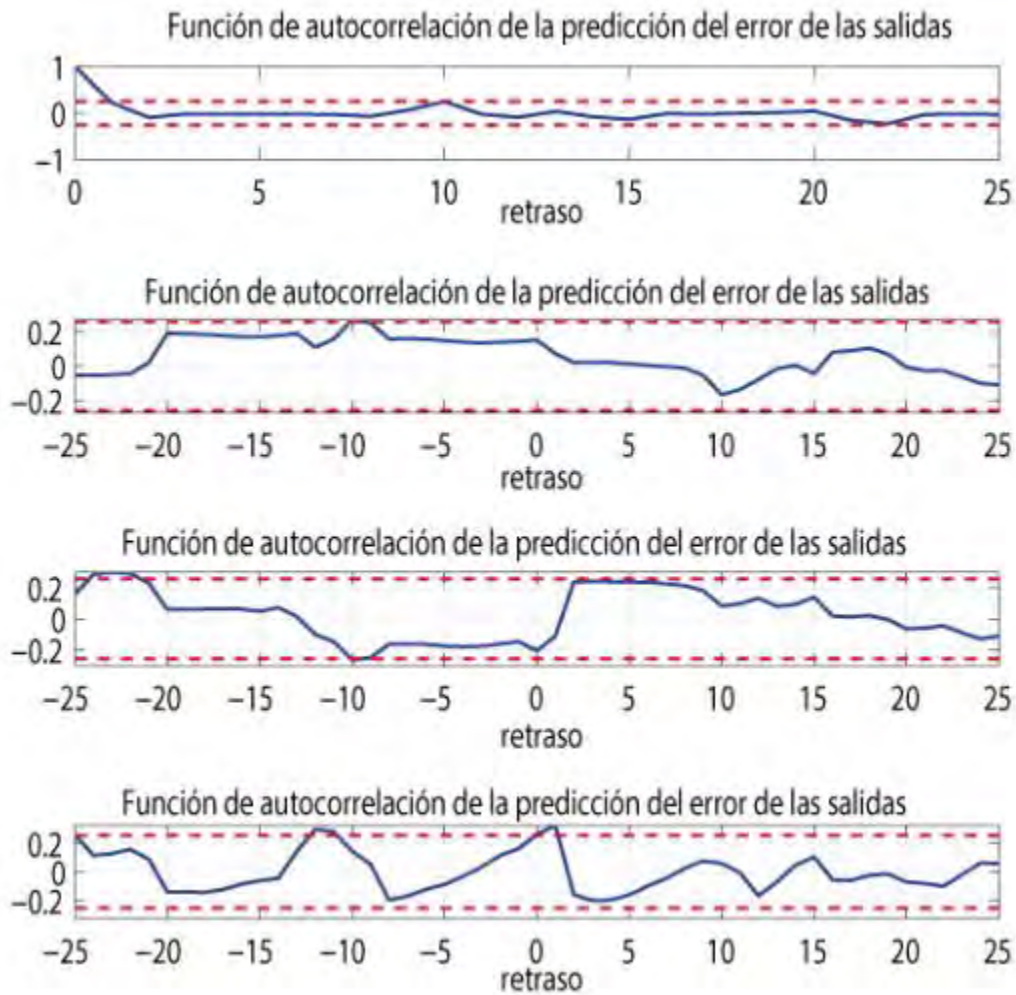


Figura 4.2. Función de auto correlación de la predicción del error.

En la figura 4.3 se muestra la predicción del voltaje, y a diferencia de la estructura NNARX, esta estructura retroalimenta su predicción como una salida y las retrasa dos veces para proporcionar un mejor resultado. La gráfica de predicción es muy cercana a la gráfica del voltaje deseado.

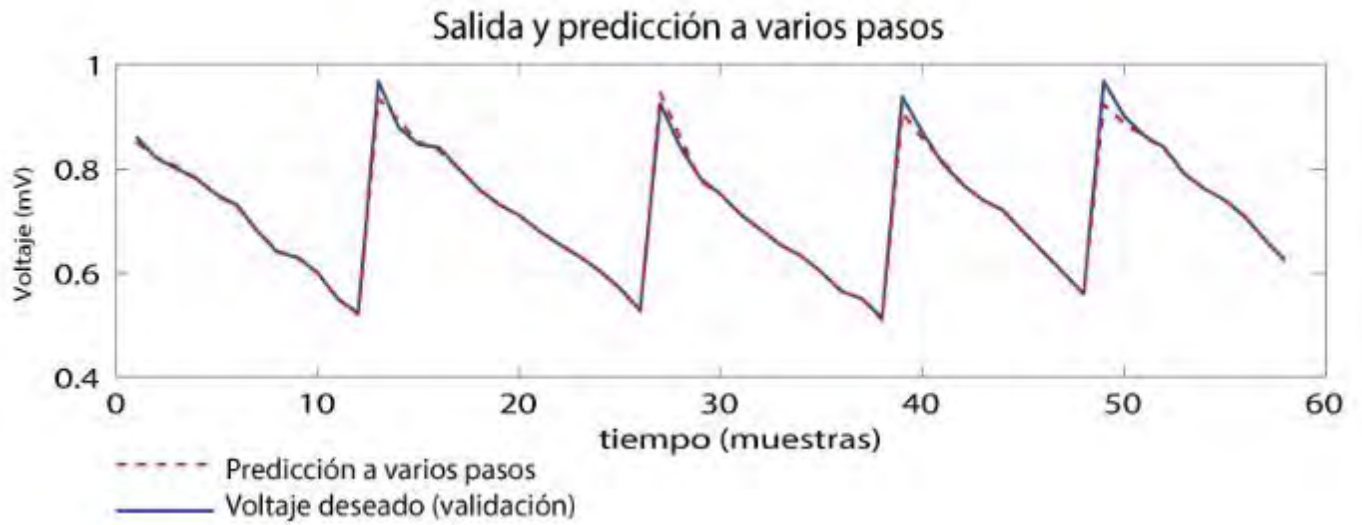


Figura 4.3. Gráfica de la salida (datos de validación, línea azul continua) vs la predicción a varios pasos (línea roja discontinua) para la estructura NNOE.

La figura 4.3 muestra los errores de autocorrelación del error de predicción de las entradas y salidas de la estructura NNOE. Para este caso particular, al usar los mismos datos de entrenamiento que la estructura NNARX y solo ajustar sus parámetros finales provocan un mejor resultado y en la gráfica de errores de correlación se aprecian mucha más controlados dentro del margen de validez.

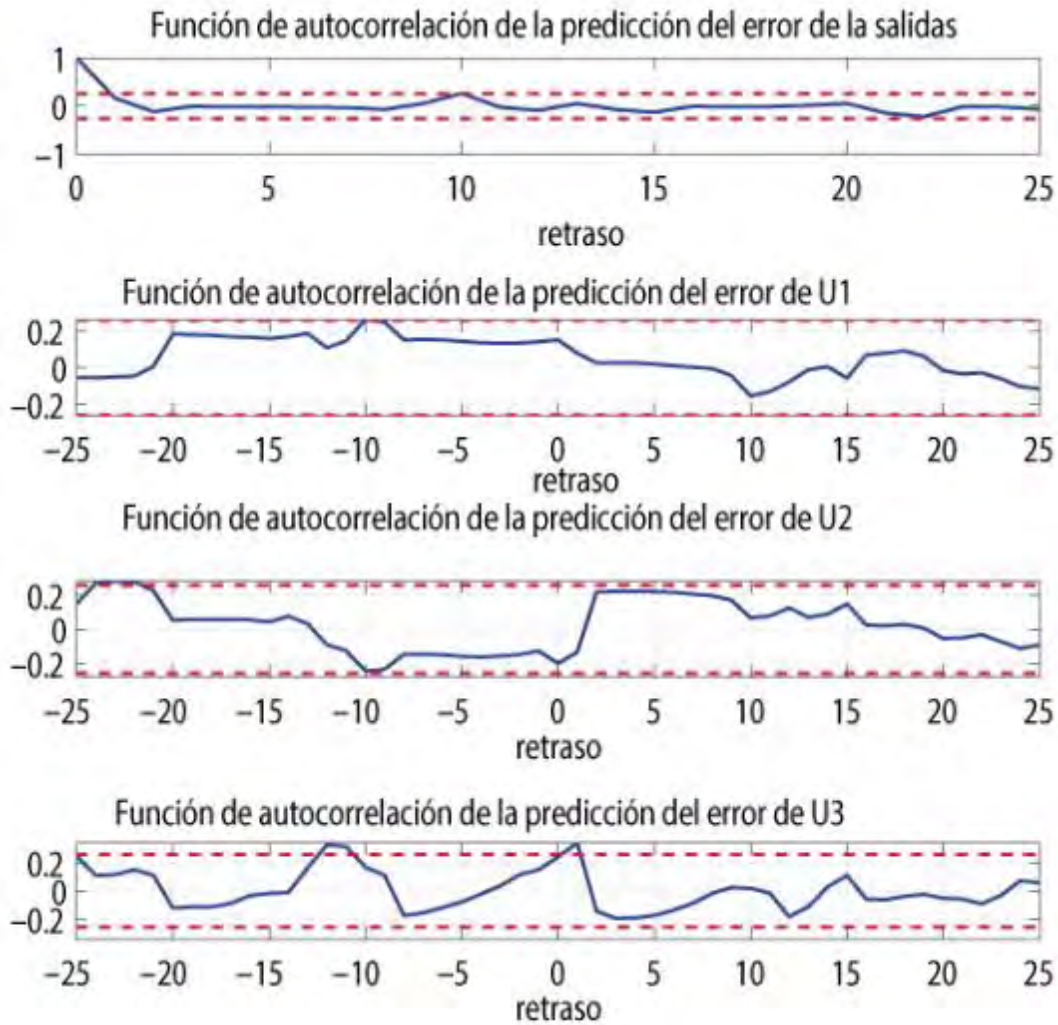


Figura 4.3. Gráfica de los errores de correlación para las entradas y salidas.

4.1.2 Resultados de la simulación del sistema.

La figura 4.4 muestra las lecturas de los patrones de entradas y salidas para poder realizar la predicción del voltaje como se muestra en la figura 4.5, y posteriormente poder ser emulados en la tarjeta NI myRIO en tiempo real.

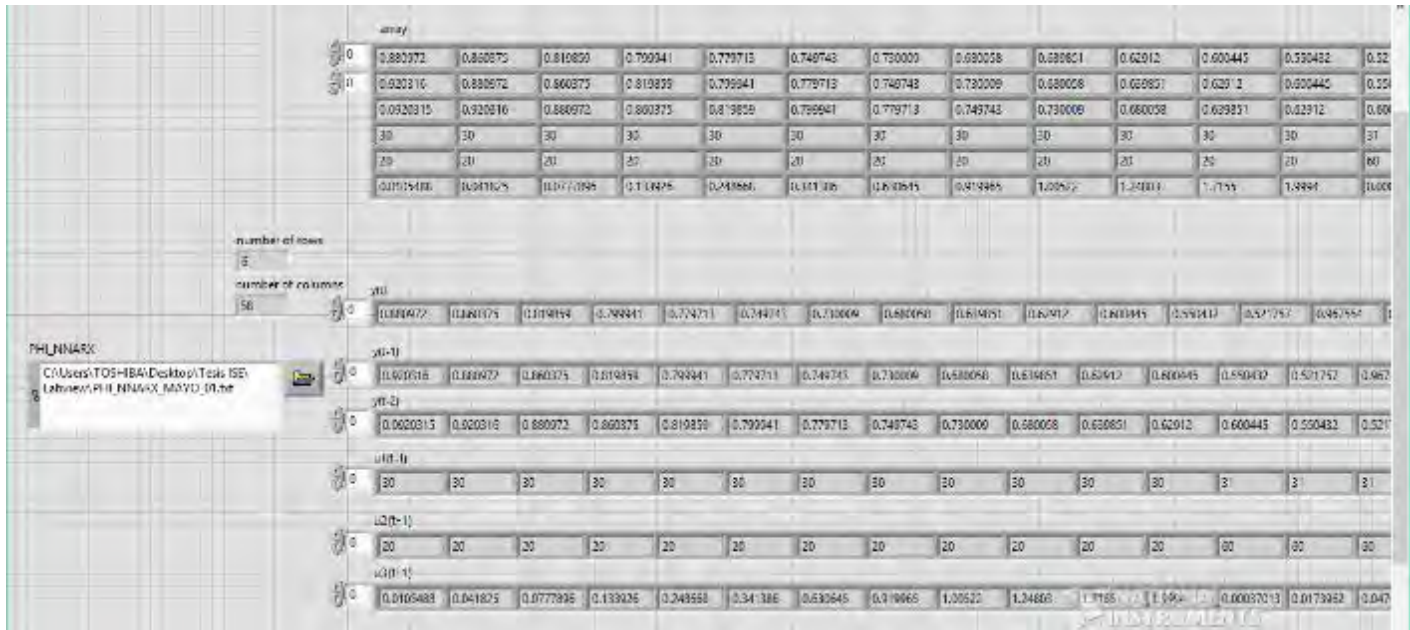


Figura 4.4. Lectura de los datos para la estructura NNARX y NNOE.

La grafica mostrada en la figura 4.5 es el resultado de la prediccion del voltaje con la estructura NNARX.

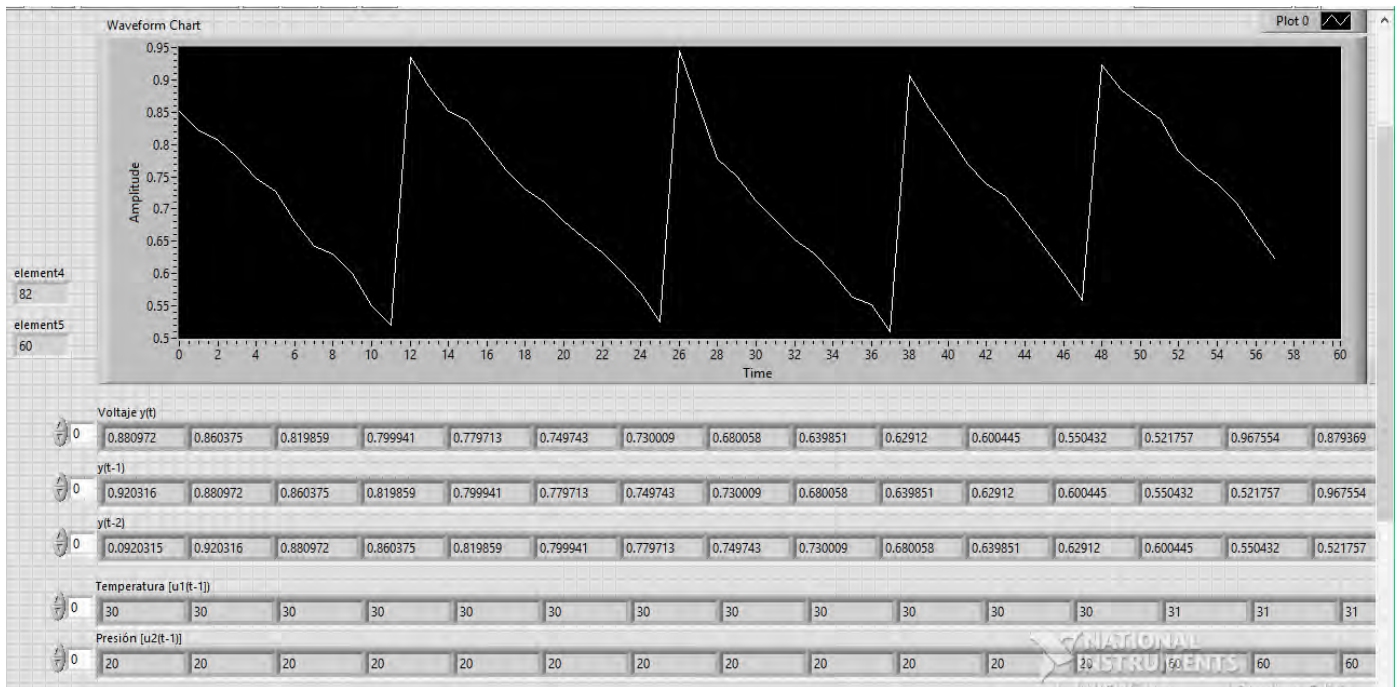


Figura 4.5. Gráfica de la simulación del voltaje de prediccion con la estructura NNARX.

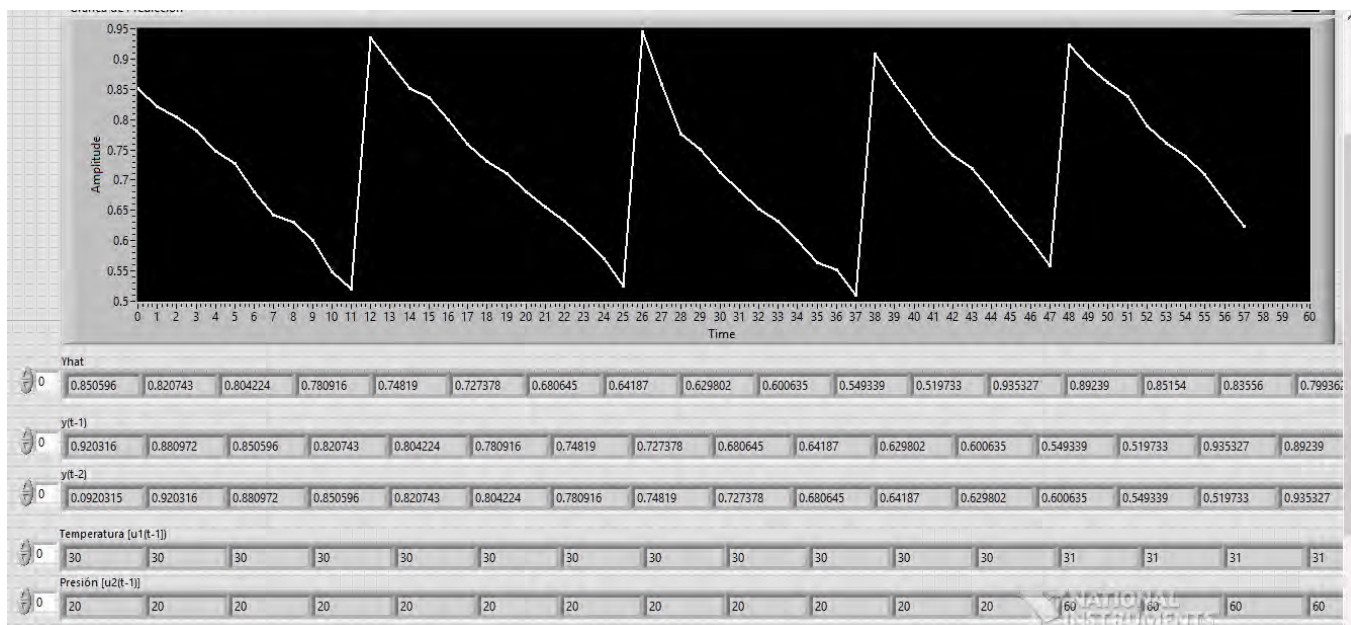


Figura 4.6. Gráfica de la simulación del voltaje de predicción con la estructura NNOE.

Con la finalidad de validar mejor nuestro sistema se anexó un apartado de cosimulación. Con el motivo de poseer un sistema completo y poder realizar pruebas a nivel de simulación. Para este apartado se usa la estructura de predicción NNOE del programa Labview en lazo cerrado con el Multisim.

El Multisim fue utilizado para crear un circuito resistivo o carga que proporciona una corriente, esta corriente se introduce al programa de Labview en cada iteración del algoritmo como una entrada pasada a la red neuronal. Al introducir esta corriente generada por el circuito resistivo se complementa el algoritmo para poder proporcionar la predicción del voltaje, los programas dependen uno del otro para que el algoritmo funcione.

PHI_NNARX		Resistencias_past_outputs		number of rows												
C:\Users\TOSHIBA\Desktop\Tesis ISE\Labview\PHI_NNARX_MAYO_01.bt		C:\Users\TOSHIBA\Desktop\Tesis ISE\Labview\Resistencias_past_outputs.bt		6												
				number of columns												
				58												
array																
0	0.880972	0.860375	0.819859	0.799941	0.779713	0.749743	0.730009	0.680058	0.639851	0.62912	0.600445	0.550432	0.521757	0.967554	0.879369	0.845698
0	0.920316	0.880972	0.860375	0.819859	0.799941	0.779713	0.749743	0.730009	0.680058	0.639851	0.62912	0.600445	0.550432	0.521757	0.967554	0.879369
0	0.0920315	0.920316	0.880972	0.860375	0.819859	0.799941	0.779713	0.749743	0.730009	0.680058	0.639851	0.62912	0.600445	0.550432	0.521757	0.967554
30	30	30	30	30	30	30	30	30	30	30	30	30	31	31	31	31
20	20	20	20	20	20	20	20	20	20	20	20	20	60	60	60	60
0	0.0105488	0.041825	0.0777896	0.133926	0.248668	0.341386	0.630645	0.919965	1.00522	1.24803	1.7155	1.9994	0.000370133	0.0173962	0.0476238	0.0549647
u1(t-1)																
0	30	30	30	30	30	30	30	30	30	30	30	31	31	31	31	
u2(t-1)																
0	20	20	20	20	20	20	20	20	20	20	20	60	60	60	60	
resistencias																
0	83.5141	20.5708	10.5394	5.97299	3.13557	2.19617	1.15756	0.739221	0.636529	0.504092	0.350011	0.275299	1409.65	55.6185	18.4649	15.3862

Figura 4.5. Lectura de datos para la simulación de la estructura NNOE en lazo cerrado con MULTISIM.

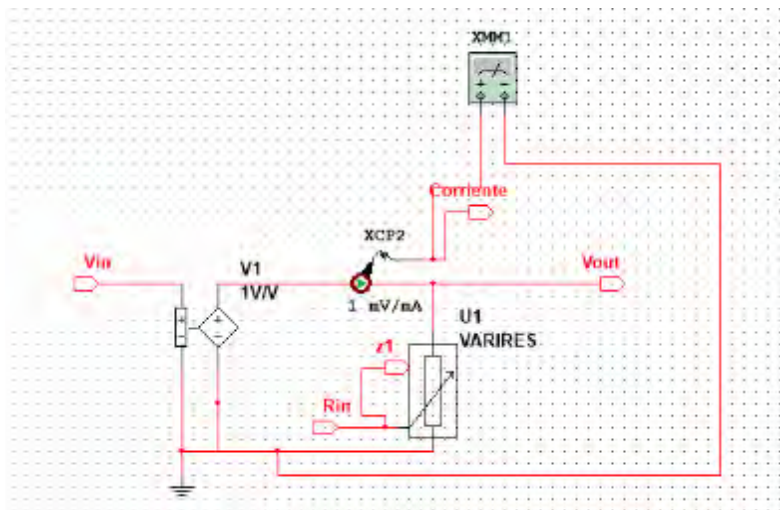


Figura 4.6. Circuito resistivo en MULTISIM usado en lazo cerrado con Labview.

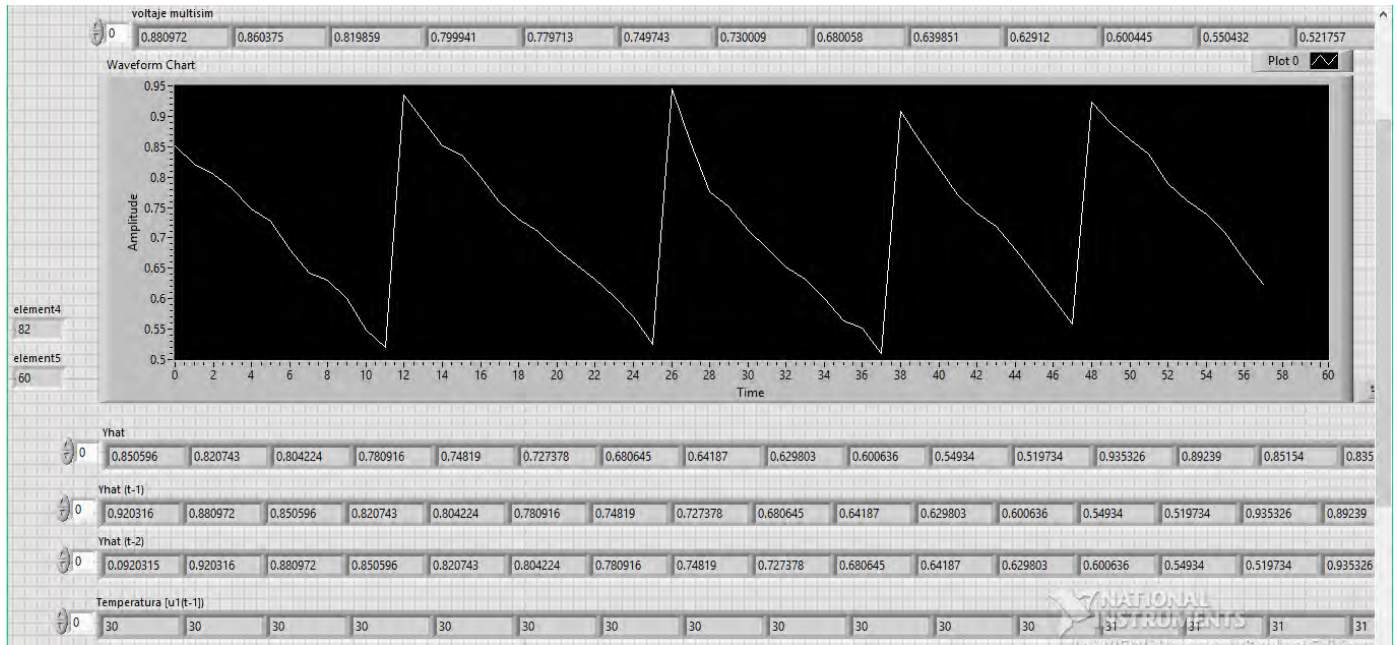


Figura 4.7. Gráfica del comportamiento del voltaje en lazo cerrado con MULTISIM con la estructura NNOE.

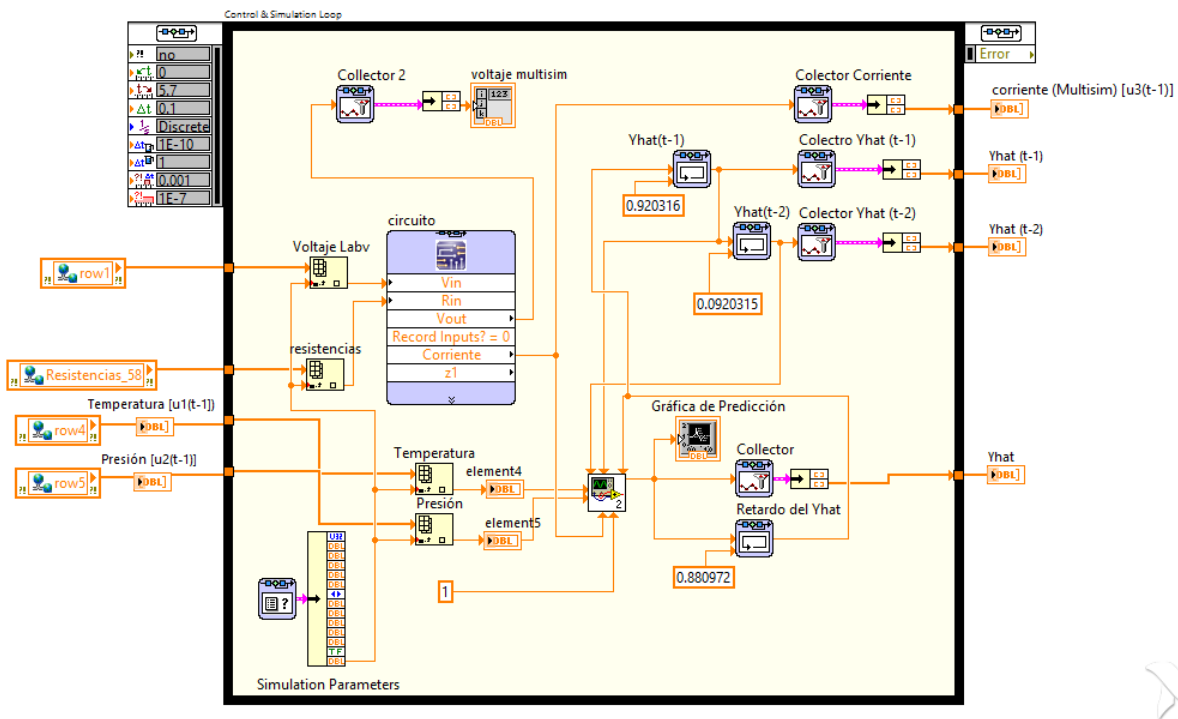


Figura 4.8. Algoritmo interno de Labview con lazo cerrado con MULTISIM.

4.1.3 Resultados de la emulación del sistema

Las siguientes figuras muestran los resultados finales utilizando el algoritmo NNARX-NNOE en el programa Labview en conjunto con la tarjeta NI myRIO. Los resultados presentados corresponden al sistema emulado con dos periodos de muestreo diferentes. Las gráficas fueron tomadas del osciloscopio. Para tener una mejor percepción de la señal del voltaje en el osciloscopio se multiplicó por un factor de diez y es por esta razón que en los datos que se muestran en el osciloscopio pueden llegar a tomar valores cercanos a 10 V cuando en el sistema real los datos máximos se acercan a 1 V. Las primeras tres figuras (4.9, 4.10 y 4.11) muestran la emulación del sistema en un tiempo total de 5.7 segundos que corresponden a 57 muestras procesadas con un tiempo de muestreo de 100 ms.

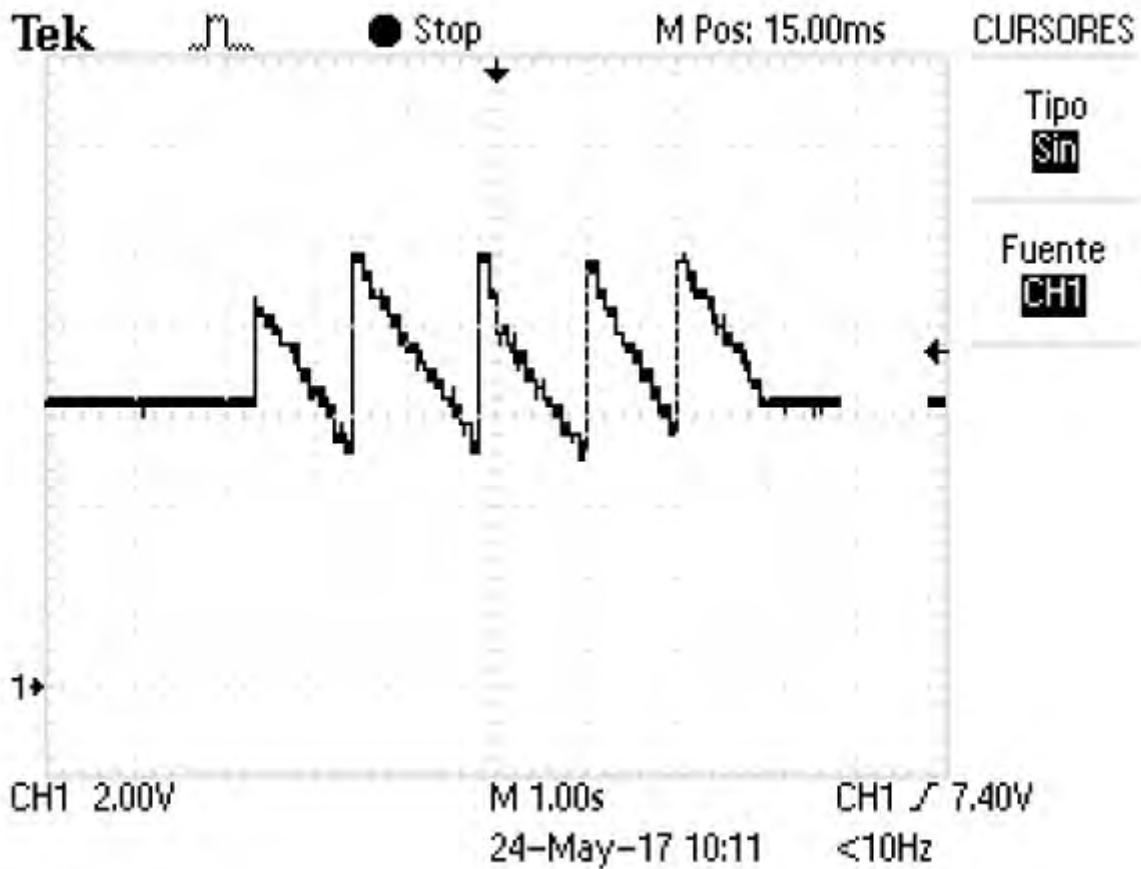


Figura 4.9. Gráfica de la emulación del voltaje desde el osciloscopio.

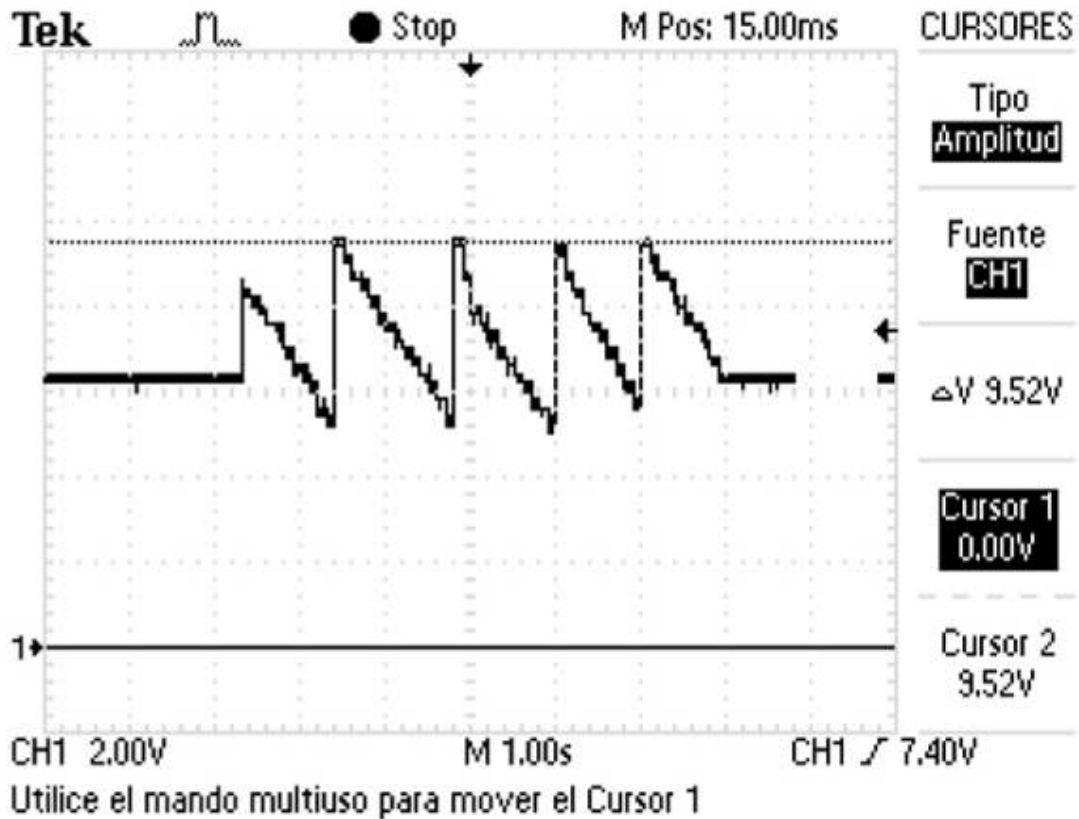


Figura 4.10. Gráfica de la emulación con amplitud del voltaje multiplicado por 10.

Los valores máximos en la emulación difieren en 7 décimas de los valores resultantes de la simulación. La diferencia de los resultados experimentales y simulados son aceptables considerando que representan un error de 7.35 %.

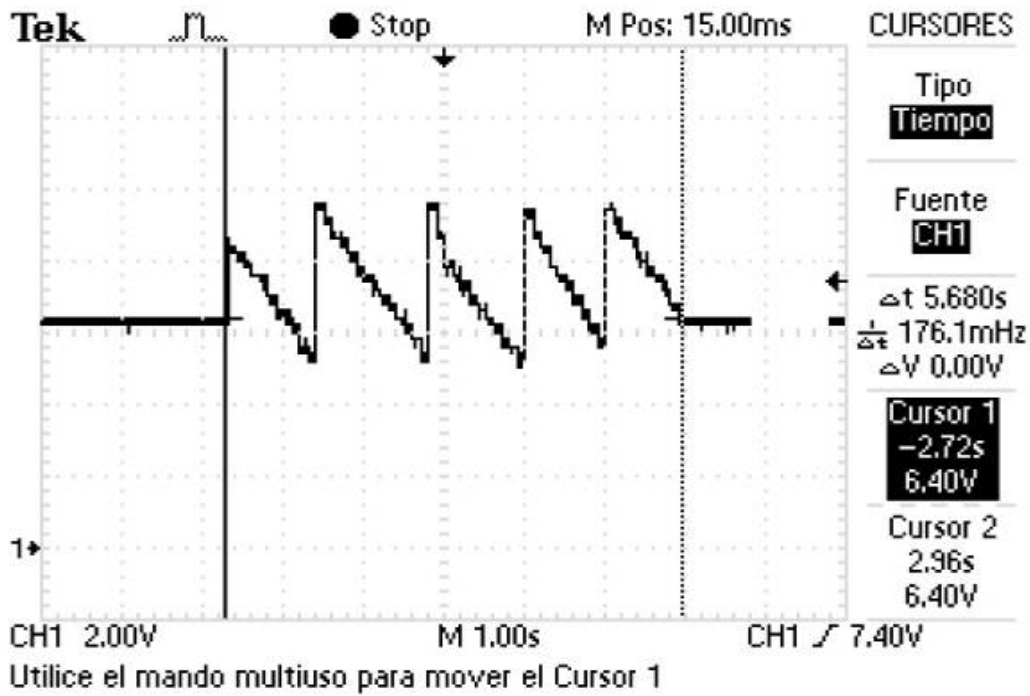


Figura 4.11.

Gráfica de la emulación con marcadores de tiempo con un total de ≈ 5.7 segundos.

Las siguientes tres figuras (4.12, 4.13 y 4.14) muestran la emulación del sistema en un tiempo total de 57 ms que corresponden a 57 muestras procesadas con un tiempo de muestro de 10 ms.

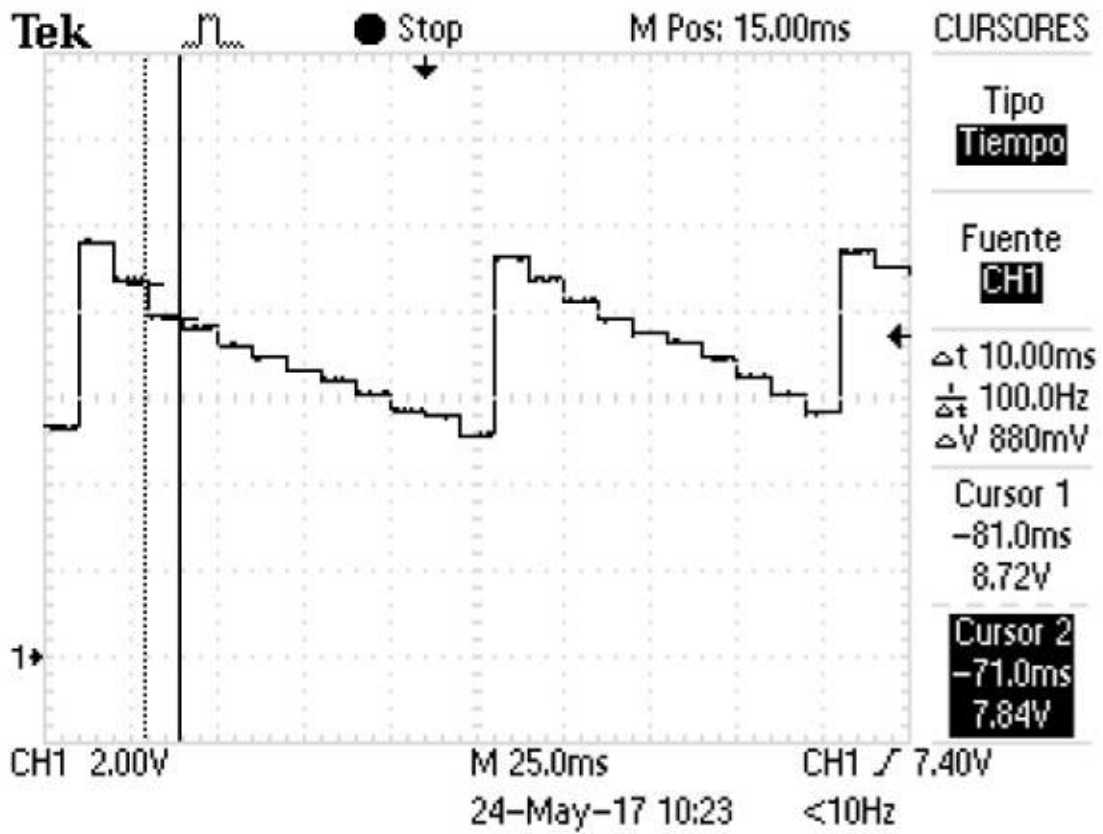


Figura 4.12. Detalle del periodo de muestreo correspondiente a 10 ms.

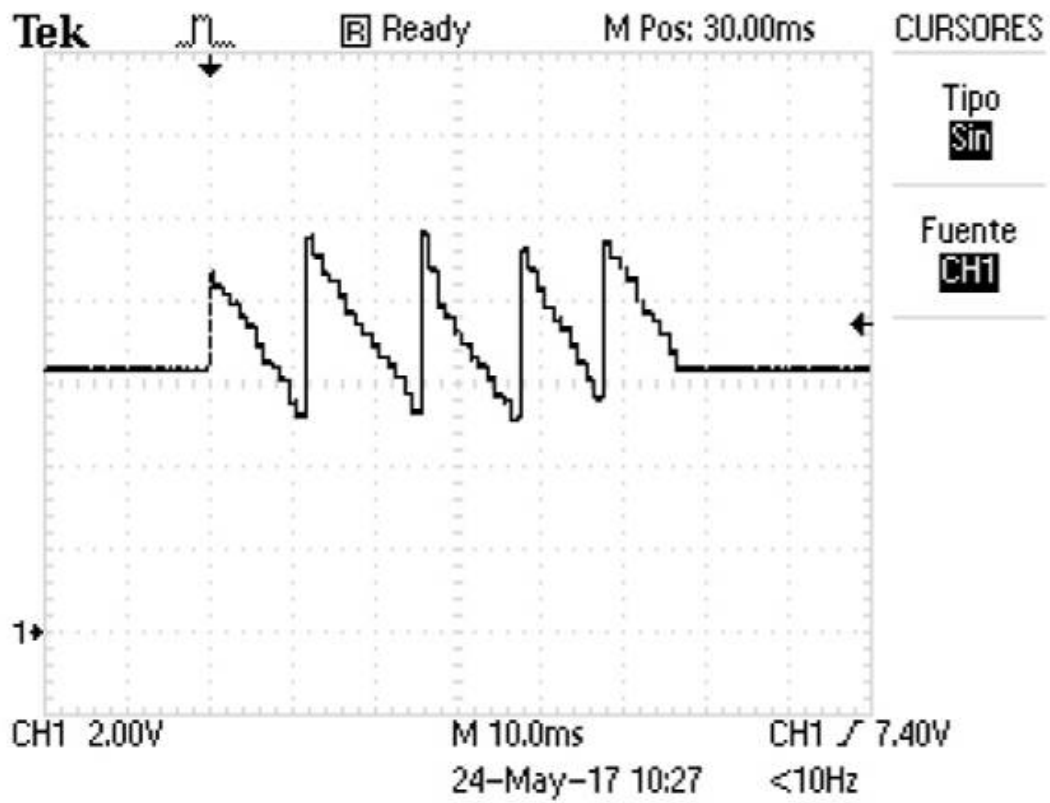


Figura 4.13. Gráfica del voltaje para un tiempo total de 57 ms.

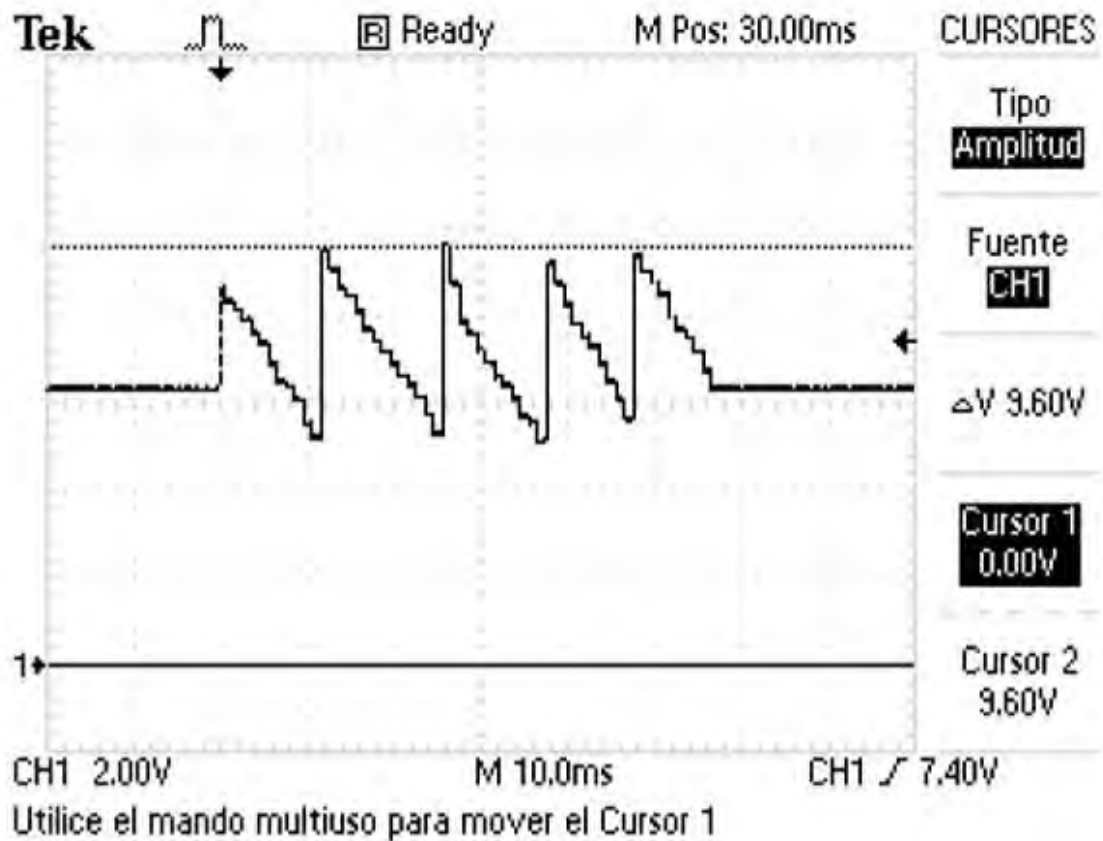


Figura 4.14. Gráfica de resultados de la emulación del voltaje.

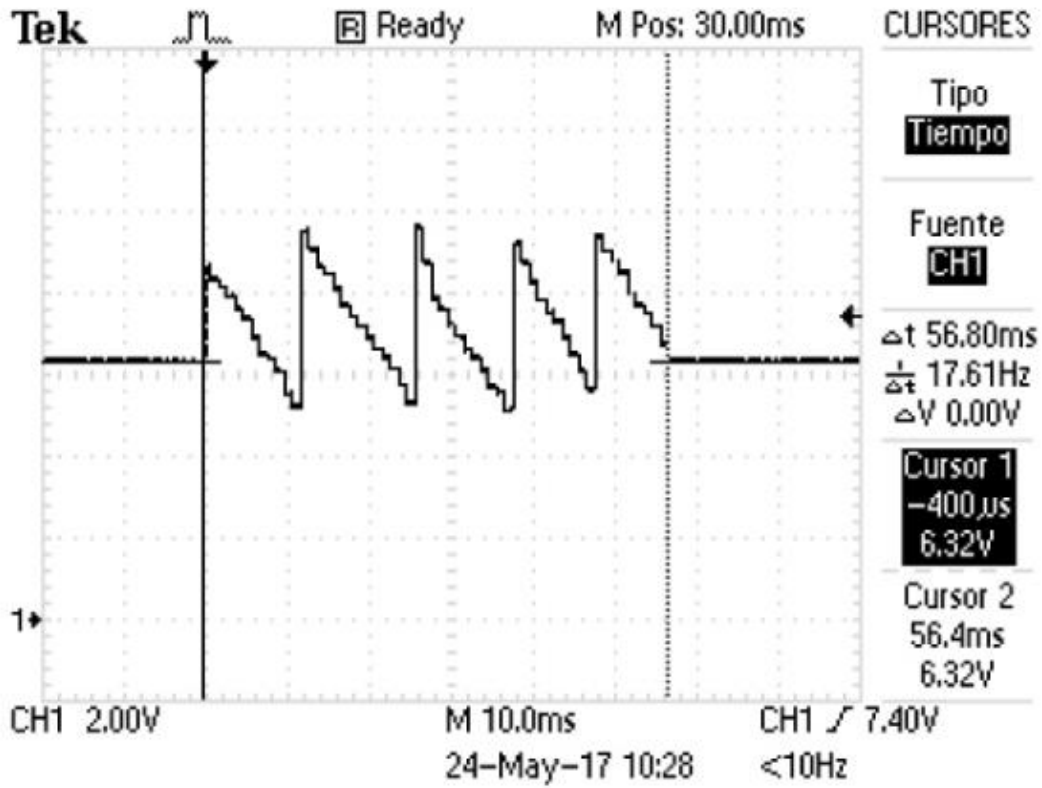


Figura 4.15. Gráfica de los resultados de la emulación en un total de 57 ms.

Capítulo 5

5 Resultados

5.1 Conclusiones y Trabajos Futuros

Este trabajo de investigación presentó una metodología para la identificación de sistemas no lineales empleando dos estructuras de redes neuronales (NNARX y NNOE) para obtener un emulador de una celda de combustible tipo PEM.

Se realizó un estudio teórico de las redes neuronales artificiales y de los algoritmos de entrenamiento más frecuentemente utilizados, también se habló de las características más importantes de las redes neuronales recurrentes que propiciaron su elección para este trabajo de investigación.

También se describe la manera en cómo se utilizó la plataforma Labview y la tarjeta NI myRio para la emulación de las señales de voltaje en la salida y así como de la programación del programa.

La realización de este trabajo permitió obtener una perspectiva del campo de estudio y aplicación de las redes neuronales. Finalmente se puede decir que los resultados obtenidos con las dos estructuras empleadas son capaces de proporcionar predicciones altamente precisas del voltaje de la celda de combustible. En particular los resultados revelaron que la estructura NNARX reproduce el voltaje de la celda de combustible prácticamente sin error y por otra parte se demostró que el comportamiento variable en el tiempo de la celda de combustible puede ser bien aproximado por un modelo NNOE.

El modelo neuronal puede ser mejorado si se consideran más parámetros que estimulen su comportamiento ya que estos pueden afectar de manera positiva a la hora de la emulación del sistema.

El modelo resultante tiene las siguientes características:

- Es válido para valores de temperatura, presión, corriente y voltaje cercanos a los presentados en este trabajo de investigación. El modelo tiene como única salida el voltaje, respondiendo a cambios en la demanda de corriente, presión y temperatura.

Se cumple con el objetivo de desarrollar un modelo simple, de fácil evaluación y con un grado de error aceptable. De igual forma, se realizó una comparación del comportamiento real y el de simulación y se comprobó que el desempeño del modelo neuronal utilizado es bastante cercano al sistema real.

TRABAJOS FUTUROS

Para los trabajos futuros se recomiendan:

1. Incluir en el modelo neuronal más parámetros que influyen directamente en el comportamiento de la celda de combustible para poder evaluar si existe una mejoría de los resultados obtenidos, entre más parámetros se utilicen para el sistema, más real será el modelo resultante.
2. Los parámetros utilizados en el algoritmo de entrenamiento de la red neuronal pueden mejorarse aumentando el número de capas ocultas y neuronas, así también como el número de las iteraciones y el tamaño de error medio cuadrático.
3. Se recomienda incrementar el grado del sistema aumentando el número de entradas y salidas pasadas al sistema para no perder características importantes del sistema real.

Bibliografía

- [1] Larminie, James; Dicks, Andrew; McDonald, Maurice S. Fuel cell systems explained. Chichester, UK: J. Wiley, 2003.
- [2] Álvarez gallegos et. al; Introducción a la identificación de procesos, CINVESTAV-IPN, 1983.
- [3] T. Romero et. al, Impedance spectroscopy as a tool in the evaluation of MEA's, Journal of Power Sources, 2003.
- [4] Ballini, Rosangela; VON ZUBEN, Fernando J. Application of neural networks to adaptive control of nonlinear systems: GW Ng, Control Systems Centre Series, UMIST, UK, 1997, ISBN: 0-86380-214-1. 2000.
- [5] DOE, U. S. Fuel cell handbook. 2004.
- [6] Huerta González Guadalupe de Rosa, Chávez López Ernesto, Velázquez Morales Blanca. Hidrógeno introducción a la energía limpia. Ed. Universidad Autónoma de la ciudad de México (UACM). México (2009).
- [7] Rozo M. Sandra Q., Juan E. Tibaquirá G. Celdas de combustible tipo membrana de intercambio protónico. Revista Scientia Et Technica vol. XIII, número 037 pp. 279-283 (2007)
- [8] S. Gottesfeld, C. F. Keller, S. Moller-Holst, A. Redondo, "Fuel Cells–Green Power", Office of Advanced Automotive Technologies, U.S. Department of Energy (2002).
- [9] EG&G Service, Persons, Inc., "Fuel Cell Handbook", Fifth Edition, USA, Department of Energy, Office of Fossil Energy, October 2000.
- [10] Kreuer, Klaus-Dieter (ed.). Fuel cells: selected entries from the encyclopedia of sustainability science and technology. Springer Science & Business Media, 2012.

- [11] Steele, Brian CH; HEINZEL, Angelika. Materials for fuel-cell technologies. Nature, 2001, vol. 414, no 6861, p. 345-352.
- [12] DE CIENCIA, Libros. tecnología No. 2, “Tecnología Solar-Eólica-Hidrógeno-Celdas de Combustible como Fuentes de Energía”. Libro electrónico Primera Edición, 2009.
- [13] Escobar Morales Beatriz. Estudio del efecto del método de desarrollo de la mea en el funcionamiento de una celda de combustible tipo PEM. Unam. Tesis de doctorado, (2007).
- [14] IEA, OECD. Prospects for Hydrogen and Fuel Cells. 2005.
- [15] Boletín, Fuel Cell Catalist, vol 4, numero 2. (2004).
- [16] J. Lin, A. Trivisonno, R. Wycisk, y Peter N. Pintauro. ECS Transactions 3, Proton Exchange Membrane Fuel Cells 6, 63-71, (2006).
- [17] Jianlu Z., Et al. Journal of Power Sources, Volume 160, 872-891, (2006).
- [18] K. Suárez-Alcántara, A. Rodríguez-Castellanos, R. Dante and O. Solorza-Feria. Journal of Power Sources 157, 114-120, (2006).
- [19] Xuewei Z., and Pengfei S. Electrochemistry Communications 8, 1229-1234, (2006).
- [20] Mathias Ulbricht. Polymer 47, 2217-2262, (2006).
- [21] E. Chalkova, et al. ECS Transactions 3, Proton Exchange Membrane Fuel Cells 6, 73-82, (2006).
- [22] B. Baradie, J.P. Dodelet, P. Guay, Journal of Electroanalytical Chemistry 489, 209 (1998).

- [23] Mingqiang Li, et al. *Electrochemical and Solid-State Letters* 9, A92-A95 (2006).
- [24] V. Ramani., H.R. Kunz, J.M. Fenton. *Journal of Membrane Science* 232, 31–44, (2004).
- [25] Minsuk K., Jin-Nam P., Hyuk K., Seongmin S. and Won-Ho L. *Journal of Power Sources*, Available online 22, (2006).
- [26] D. Raistrick, et al. *Symposium on Diaphragms, separators and ion exchange membranes proceedings*, 86, 172 (1986).
- [27] Lymberopoulos N. *Hydrogen production from renewables*. Pág. 5-18, 87. Project Technical Assistant Framework Contract (EESD Contract N: NNE5-PTA-2002-003/1). UE (2005).
- [28] J. A. Kolde , B. Bahar, M. S. Wilson, T. A. Zawodzinski y S. Gottesfeld, *The electrochemical Society Proceedings Series*. NJ. 193-195.
- [29] G. García, et al. *Catalysis Today* 116, 415-421, (2006).
- [30] P.K Simpson, *Artificial Neural Systems: Foundation, Paradigms, Applications and Implementations*, Pergamon press, New York, USA., 1990.
- [31] Hilerá José R y Martínez Víctor J. “*Redes Neuronales Artificiales*” Addison Wesley Iberoamericana Rama EUA 1995.
- [32] S. Haykin, *Neural Networks: A comprehensive Foundation*, Second edition, Prentice Hall, New Jersey, USA., 1999.
- [33] Z.Feng and A.N.Michel, *Robustness Analysis of a Class of Discrete-Time Systems with Applications to Neural Networks*, Proc. of American Control Conference, 3479-3483, San Deigo, 1999.

- [34] J.A.K. Suykens, J. Vandewalle, B. De Moor, NLq Theory: Checking and Imposing Stability of Recurrent Neural Networks for Nonlinear Modelling, *IEEE Transactions on Signal Processing* (special issue on neural networks for signal processing), Vol.45, No.11, 2682-2691, 1997.
- [35] G.A.Rovithakis and M.A.Christodoulou, Adaptive Control of Unknown Plants Using Dynamical Neural Networks, *IEEE Trans. on Syst., Man and Cybern.*, Vol. 24, 400-412, 1994.
- [36] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain", *Psychological review*, vol. 65, pp. 396-408,1958.
- [37] S. Haykin, *Neural Networks: A comprehensive Foundation*, Second edition, Prentice Hall, New Jersey, USA., 1999.
- [38] M.L. Minisky, S. A. Papert, *Perceptrons*, Cambridge, MA, MIT Press, 1969.
- [39] N. E. Cotter, "The Stone-Weirtrass theorem and its application to neural networks", *IEEE transactions on Neural Networks*, vol. 1, no. 4, pp. 290-295, 1990.
- [40] G. A. Rovitakis and M. A. Chistodoulou, *Adaptive Control with Recurrent High-Order Neural Networks: Theory and Industrial Applications*, Springer-Verlag,London, 2000.
- [41] M. Nørgaard, O. Ravn, N. K. Poulsen, L. K. Hansen, *Neural Networks for Modelling and Control of Dynamic Systems*, London, U.K.: Springer-Verlag, 2003.
- [42] Levenberg, K. (1944). A method for solution of certain nonlinear problems in least squares. *Quart. Appl. Mathematics*, 2, 164-168.
- [43] Marquardt, D. (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal Appl. Mathematics*, 11(2), 164-168.

- [44] Moré, J. J. (1983). Recent developments in algorithms and software for trustregion methods. In *Mathematical Programming, the State of the Art: Bonn 1982*. Springer-Verlag, Berlin, Germany.
- [45] Dennis, J. E. and More, J. J. (1977). Quasi-Newton methods, motivation and theory. *SIAM Review*, 19(1), 46-89.
- [46] L. Ljung, "Perspectives on system identification", *Annu. Rev. Control*, vol. 34, no. 1, pp.1-12, Apr. 2010.
- [47] J.C.Willems, Least Squares Optimal Control and Algebraic Riccati Equations, *IEEE Trans. Automat. Contr.*, vol.16, 621-634, 1971.
- [48] Wen Yu, Passive Equivalence of Chaos in Lorenz System, *IEEE Trans. Circuits and Systems, Part I*, Vol.46, No.7, 876-879, 1999
- [49] J.A.K. Suykens, J. Vandewalle, B. De Moor, NLq Theory: Checking and Imposing Stability of Recurrent Neural Networks for Nonlinear Modelling, *IEEE Transactions on Signal Processing* (special issue on neural networks for signal processing), Vol.45, No.11, 2682-2691, 1997.
- [50] G.A.Rovithakis and M.A.Christodoulou, Adaptive Control of Unknown Plants Using Dynamical Neural Networks, *IEEE Trans. on Syst., Man and Cybern.*, Vol. 24, 400-412,1994.
- [51] C. Alippi and V. Piuri, "Neural modeling of dynamic systems with nonmeasurable state variables," *IEEE Trans. Instrum. Meas.*, vol. 48, no. 6, pp. 1073–1080, Dec. 1999.
- [52] G. Barbosa Pool, "Diseño, construcción y evaluación de una celda de combustible de intercambio protónico", *Licenciatura, Universidad de Quintana Roo*, 2004.
- [53] He, X. and Asada, H. (1993). A new method for identifying orders of inputoutput models for nonlinear dynamical systems. In *Proc. of the American Control Conference*, San Francisco, California, pp. 2520-2523.

[54] M. H. Beale, M. T. Hagan, and H. B. Demuth, *Neural Network Toolbox User's Guide-R2012a*. Natick, MA, USA: The MathWorks Inc., 2012.

[55] F. da Costa Lopes, E. H. Watanabe, and L. G. B. Rolim, "A recurrent neural approach for modeling non-reproducible behavior of PEM fuel cell stacks," in *Proc. IEEE ICIT*, Cape Town, South Africa, Feb. 2013, pp. 679–683.

Apéndice A

Patrones de entrenamiento y patrones de validación para la red neuronal NNARX y la red neuronal NNOE.

Estos patrones de entrenamiento y patrones de validación son usados en el software Matlab en archivos tipo .txt para cada tabla, que son utilizados en el algoritmo que se muestra en el apéndice 2.

Tabla A1. Patrones de Entrenamiento o aprendizaje.

30	20	0.00030844	0.9428866
30	20	0.00246755	0.9002125
30	20	0.02085082	0.8402096
30	20	0.04978287	0.8143707
30	20	0.10271185	0.7901353
30	20	0.20542375	0.7599797
30	20	0.44631855	0.7102753
30	20	0.504861	0.7002234
30	20	0.766607	0.6603243
30	20	1.08122	0.6198702
30	20	1.3580175	0.5882963
30	20	1.5042815	0.5724476
30	20	1.9156845	0.5300817
30	20	2.214073	0.5004812
31	60	0.00141884	0.9447367
31	60	0.00777279	0.9044675
31	60	0.03078272	0.8606217
31	60	0.09611115	0.8206608
31	60	0.245213	0.7796517
31	60	0.3998052	0.7493111
31	60	0.512017	0.7296391
31	60	0.694061	0.7002851
31	60	0.954511	0.6611877
31	60	1.0943595	0.6413307
31	60	1.237354	0.6217203
31	60	1.504775	0.5861995
31	60	1.785768	0.5501238
31	60	2.002419	0.5228666
82	20	0.00024676	0.9511501
82	20	0.00987021	0.8798621
82	20	0.0981469	0.7989538
82	20	0.233122	0.7613363
82	20	0.39776945	0.7312424
82	20	0.5959755	0.7021968
82	20	0.9072575	0.6637161
82	20	1.096704	0.6424406
82	20	1.2795495	0.6232003
82	20	1.716121	0.5804644
82	20	2.001	0.5538239
82	20	2.2375765	0.5321785
55	40	0.00382471	0.8999041
55	40	0.05046145	0.8332412

55	40	0.1776021	0.7897036
55	40	0.34107745	0.7573896
55	40	0.5016535	0.7313657
55	40	0.716269	0.7002234
55	40	1.0035535	0.661866
55	40	1.321251	0.6220903
55	40	1.676517	0.5799711
55	40	2.0022955	0.5426003
82	60	0.00820461	0.9206245
82	60	0.02967232	0.8809104
82	60	0.05262055	0.860745
82	60	0.1528649	0.8200442
82	60	0.4329321	0.7699699
82	60	0.591534	0.7506061
82	60	0.8785105	0.7205122
82	60	1.3098385	0.6810448
82	60	1.7783035	0.6423174
82	60	2.24245	0.60618

Tabla A2. Patrones de Validación

30	20	0.00092533	0.0920315
30	20	0.00518186	0.9203162
30	20	0.01054879	0.880972
30	20	0.04182501	0.860375
30	20	0.0777896	0.8198591
30	20	0.1339264	0.7999405
30	20	0.2486676	0.7797135
30	20	0.3413859	0.7497428
30	20	0.6306445	0.7300091
30	20	0.9199655	0.6800581
30	20	1.005219	0.6398506
30	20	1.2480265	0.6291204
30	20	1.715504	0.6004448
30	20	1.999396	0.5504322
31	60	0.00037013	0.5217566
31	60	0.01739625	0.9675538
31	60	0.04762377	0.8793687
31	60	0.05496475	0.845698
31	60	0.1605143	0.8407646
31	60	0.33830145	0.8001255
31	60	0.5016535	0.7607197
31	60	0.6288555	0.7314891
31	60	0.825828	0.710522
31	60	1.0013945	0.6801198
31	60	1.169435	0.6544042
31	60	1.3772645	0.6309704
31	60	1.6317925	0.6029115
31	60	1.9616425	0.5697958
82	20	0.00178898	0.527985
82	20	0.03263338	0.9245713
82	20	0.1533584	0.8401479
82	20	0.28728485	0.7807001

82	20	0.527069	0.7504212
82	20	0.74736	0.711632
82	20	1.0027515	0.6825865
82	20	1.1907175	0.6526775
82	20	1.4987295	0.6324505
82	20	1.8910705	0.6013082
82	20	2.0377665	0.5640608
82	20	2.437325	0.5504938
55	40	0.00030844	0.5139248
55	40	0.01357154	0.9381998
55	40	0.1035138	0.8703652
55	40	0.27593405	0.8100539
55	40	0.4468121	0.7691681
55	40	0.5765435	0.739876
55	40	0.8663575	0.7202039
55	40	1.1738765	0.679688
55	40	1.5007655	0.6402823
55	40	1.8530085	0.6006298
82	60	0.00030844	0.559559
82	60	0.01610078	0.9679238
82	60	0.0479939	0.9015075
82	60	0.0921631	0.8636433
82	60	0.2970933	0.8403946
82	60	0.5006665	0.7901352
82	60	0.685733	0.7613363
82	60	1.005836	0.7401226
82	60	1.505824	0.708302
82	60	2.004948	0.6643944
82	60	2.00495	0.6243719


```

        title('Auto-correlation function of prediction error')
    else
        title(['Autocorrelation coefficients for prediction error (output # ' ...
            num2str(ii) ' )']);
    end
    grid

    for i=1:nu,
        subplot(nu+1,1,i+1);
        UEcross=crosssco(E(ii,:),U(i,1:N),M);

        plot([-M:M], UEcross, 'b-'); hold on
        plot([-M M],[conf -conf;conf -conf], 'r--');hold off
        xlabel('lag')
        title(['Cross-correlation coef. of u' num2str(i) ' and prediction error'])
        ymax=min(5*conf,max([abs(UEcross)]));
        axis([-M M -ymax ymax]);
        grid
    end
    subplot(111)
    drawnow
end

% ----- Extract linear model from network -----
dy2dx=zeros(outputs*(inputs+1),N);

% Matrix with partial derivative of each output with respect to each of the
% outputs from the hidden neurons
for t=1:N,
    dy2dy1 = W2(:,1:hidden);
    for j = H_output',
        dy2dy1(j,:) = W2(j,1:hidden)*(1-Yhat(j,t).*Yhat(j,t));
    end

% Matrix with partial derivatives of the output from each hidden neurons with
% respect to each input:
dy1dx = W1;
for j = H_hidden',
    dy1dx(j,:) = W1(j,:)*(1-y1(j,t).*y1(j,t));
end

% Matrix with partial derivative of each output with respect to each input
d1 = (dy2dy1 * dy1dx)';
d1(inputs+1,:)=d1(inputs+1,:)+W2(:,hidden+1)';
dy2dx(:,t) = d1(:);
end

figure(si+2*outputs+1)
subplot(212)
plot(dy2dx(1:outputs*inputs,:))
title('Linearized network parameters')
xlabel('time (samples)')
grid
for ii=1:outputs,
    subplot(2,outputs,ii);

```

```
    hist(E(ii,:),20)
end
subplot(2,outputs,1);
title('Histogram of prediction errors')
subplot(111)
figure(si+1)
```


Apéndice C

Tablas de los pesos sinápticos para la capa oculta y la capa de salida.

Los pesos que fueron obtenidos durante el entrenamiento de la red neuronal NNARX en el software Matlab se presentan en las siguientes tablas:

Tabla C1. Pesos de la Capa Oculta:

	yhat(t)	yhat(t-1)	yhat(t-2)	Presión	Temperatura	Corriente	bias
neurona 1	0.047768802	-0.326896275	-0.561437506	0.73603014	-0.977609424	0.0768596	-0.57659859
neurona 2	0.034983843	0.010270571	-0.117345467	-0.237295484	0.001455121	-0.145849228	-0.03810294
neurona 3	-0.023800428	0.138567519	0.251596869	-0.02300123	0.022415789	-0.003165101	-0.67701373
neurona 4	0.128734259	0.204138161	0.197944341	-0.100966748	-0.211287427	0.238010652	0.1310983
neurona 5	0.447472365	0.329899711	0.389091856	-0.003147371	0.003909361	3.817550574	1.12844336
neurona 6	-3.280269928	-1.425231241	8.154607619	-0.026486781	-0.265116624	-0.607044294	0.3028985
neurona 7	0.043849899	-0.195703194	-0.345540854	0.078331185	-0.015572306	0.039417723	-0.95564546
neurona 8	0.606157901	0.053792011	0.34974766	-0.008184872	0.295421324	-1.592999949	0.17893654
neurona 9	0.024936201	-0.144969205	-0.201523874	0.090724405	0.130584411	0.104095626	0.06738786
neurona 10	-0.072080414	0.181558771	0.448191282	-0.122507396	0.474895753	-0.273988066	0.03425897

Tabla C2. Pesos de la capa de salida.

neurona 1	neurona 2	neurona 3	neurona 4	neurona 5	neurona 6	neurona 7	neurona 8	neurona 9	neurona 10	neurona 11
-1.80503111	-0.32112868	-6.89600766	-0.24190933	-2.29193735	-1.21048532	-4.80637143	0.66882672	0.21145119	0.4785264	0.51119421

Apéndice D

Programa de validación para la estructura NNOE solo una predicción como retroalimentación.

```
load NNARX_TRAINED_TESIS2017_MAY01.mat %carga el espacio de trabajo y los
parametros de la red entrenada
[Yhat2,NSSE]=nnvalid('nnoe',NetDef,NN,W1,W2,Yu,Uu);
figure(1)
sample=1:1:60;
plot(sample,Yhat,sample,Yhat2)
```


Apéndice F

Programa de validación para la red neuronal NNOE con retroalimentación de dos predicciones pasadas.

```
load MAYO_01.mat %carga el espacio de trabajo y los parametros de la red
entrenada
[Yhat2,NSSE]=nnvalid('nnoe',NetDef,NN,W1,W2,Yu,Uu);
figure(1)
sample=1:1:58;
plot(sample,Yhat,sample,Yhat2)
```

Yhat
Yhat2
PHI

Apéndice G

Pesos sinápticos para la capa oculta y la capa de salida.

Los Pesos que fueron calculados durante el entrenamiento de la red neuronal NNOE en el software Matlab.

Tabla G1. Pesos de la capa oculta.

	yhat(t)	yhat(t-1)	yhat(t-2)	Presión	Temperatura	Corriente	bias
neurona 1	0.047768802	-0.326896275	-0.561437506	0.73603014	-0.977609424	0.0768596	-0.57659859
neurona 2	0.034983843	0.010270571	-0.117345467	-0.237295484	0.001455121	-0.145849228	-0.03810294
neurona 3	-0.023800428	0.138567519	0.251596869	-0.02300123	0.022415789	-0.003165101	-0.67701373
neurona 4	0.128734259	0.204138161	0.197944341	-0.100966748	-0.211287427	0.238010652	0.1310983
neurona 5	0.447472365	0.329899711	0.389091856	-0.003147371	0.003909361	3.817550574	1.12844336
neurona 6	-3.280269928	-1.425231241	8.154607619	-0.026486781	-0.265116624	-0.607044294	0.3028985
neurona 7	0.043849899	-0.195703194	-0.345540854	0.078331185	-0.015572306	0.039417723	-0.95564546
neurona 8	0.606157901	0.053792011	0.34974766	-0.008184872	0.295421324	-1.592999949	0.17893654
neurona 9	0.024936201	-0.144969205	-0.201523874	0.090724405	0.130584411	0.104095626	0.06738786
neurona 10	-0.072080414	0.181558771	0.448191282	-0.122507396	0.474895753	-0.273988066	0.03425897

Tabla G2. Los pesos para la capa de salida:

neurona 1	neurona 2	neurona 3	neurona 4	neurona 5	neurona 6	neurona 7	neurona 8	neurona 9	neurona 10	neurona 11
-1.80503111	-0.32112868	-6.89600766	-0.24190933	-2.29193735	-1.21048532	-4.80637143	0.66882672	0.21145119	0.4785264	0.51119421