INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE INGENIERÍA MÉCANICA Y ELÉCTRICA
SECCIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

MODELO TÉRMICO CON UNA RED NEURONAL DE RETROPROPAGACIÓN DE UNA MÁQUINA ELÉCTRICA ROTATORIA

TESIS

QUE PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

PRESENTA:

ING. MIGUEL ANGEL MONROY CANALES

México, D.F. Diciembre de 2012

SIP-14

INSTITUTO POLITÉCNICO NACIONAL
SECRETARÍA DE INVESTIGACIÓN Y POSGRADO
ACTA DE REVISIÓN DE TESIS

En la Ciudad de **México, D. F.** siendo las **14:00** horas del día **21** del mes de **Noviembre** del **2012** se reunieron los miembros de la Comisión Revisora de la Tesis, designada por el Colegio de Profesores de Estudios de Posgrado e Investigación de **ESIME-ZAC.** para examinar la tesis titulada:

"MODELO TÉRMICO CON UNA RED NEURONAL DE RETROPROPAGACIÓN DE UNA MÁQUINA ELÉCTRICA ROTATORIA"

Presentada por el alumno:

MONROY	**CANALES**	**MIGUEL ANGEL**
Apellido paterno	Apellido materno	Nombre(s)

Con registro:

B	1	0	1	8	9	3

aspirante de:

MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

Después de intercambiar opiniones, los miembros de la Comisión manifestaron ***APROBAR LA TESIS***, en virtud de que satisface los requisitos señalados por las disposiciones reglamentarias vigentes.

LA COMISIÓN REVISORA

Los Directores(a) de tesis

DR. TADEUSZ NIEWIEROWICZ SWIECICKA

DR. LESZEK ZBIGNIEW KAWECKI ZLOTKOWSKA

Presidente

DR. DANIEL OLGUÍN SALINAS

Segundo Vocal

DR. LESZEK ZBIGNIEW KAWECKI ZLOTKOWSKA

Tercer Vocal

DR. JAIME JOSÉ RODRÍGUEZ RIVAS

Secretario

DR. DAVID ROMERO ROMERO

PRESIDENTE DEL COLEGIO DE PROFESORES

DR. MAURO ALBERTO ENCISO AGUILAR

I. P. N. SECCION DE ESTUDIOS DE POSGRADO E INVESTIGACION

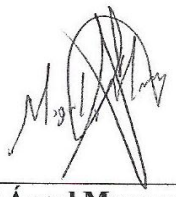
INSTITUTO POLITÉCNICO NACIONAL

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

CARTA CESIÓN DE DERECHOS

En la Ciudad de México, D.F. el día **21** del mes de **Noviembre** del año **2012**, el que suscribe **Miguel Ángel Monroy Canales** alumno(a) del Programa de **Maestría en Ciencias en Ingeniería Eléctrica**, con número de registro **B101893**, adscrito(a) a la **Sección de Estudios de Posgrado e Investigación de la ESIME unidad Zacatenco del IPN**, manifiesto(a) que es el (la) autor(a) intelectual del presente trabajo de Tesis bajo la dirección del (de la, de los) **Dr. Tadeusz Niewierowicz Swiecicka y del Dr. Leszek Kawecki Zlotkowska** y cede los derechos del trabajo titulado **Modelo Térmico con una Red Neuronal de Retropropagación de una Máquina Eléctrica Rotatoria**, al Instituto Politécnico Nacional para su difusión, con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expreso del (de la) autor(a) y/o director(es) del trabajo. Este puede ser obtenido escribiendo a las siguientes direcciones **mamc_124@hotmail.com**, **tniewi@ipn.mx** y **lkawecki@ipn.mx**. Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.



Miguel Ángel Monroy Canales

Dedicatoria

- A dios por llenar de bendiciones mi vida, por darme una hermosa familia y darme salud y sabiduría para concluir mis estudios de maestría.

- A mi esposa Paola por su incondicional apoyo, por impulsarme siempre a ser mejor persona, por todos los consejos y palabras de aliento durante los momentos difíciles y por estar siempre a mi lado en las buenas y en las malas, con todo mi amor le dedico este trabajo.

- A mis padres por ser el pilar fundamental de lo que soy, por su apoyo en mi educación, tanto académica como en la vida y por estar siempre a mi lado demostrando su solidaridad y amor.

- A mis hermanas, sobrinos, abuelitos y suegros por su cariño y motivación a seguir adelante y por el apoyo y comprensión brindada durante mis estudios de maestría.

- A mis amigos y compañeros por ayudarme a crecer personal y profesionalmente y por todas las anécdotas y vivencias que hemos compartido.

Agradecimientos

- A dios por haberme permitido concluir este proyecto tan importante y por haberme llenado de bendiciones durante esta etapa de mi vida.

- A mis asesores de tesis Dr. Tadeusz Niewierowicz Swiecicka y al Dr. Leszek Kawecki Zlotkowska por el tiempo invertido en mi trabajo y por el apoyo incondicional que me proporcionaron durante mi estancia en la SEPI.

- A los miembros de mi comisión revisora por sus acertadas recomendaciones para enriquecer este trabajo, especialmente al Dr. David Romero Romero por su confianza y consejos durante las materias que curse con él como en la revisión de mi trabajo.

- Al Dr. Ricardo Álvarez Salas, profesor investigador de la Universidad Autónoma de San Luis Potosí por su apoyo durante el periodo que estuve en el programa de movilidad académica nacional.

- A mi querido IPN por brindarme la oportunidad de regresar a la escuela que me formo a nivel licenciatura y por el apoyo económico que me brindo a través de la beca institucional de la beca del programa PIFI.

- Al Conacyt por el apoyo económico que recibí durante mis estudios de maestría.

- A todo el personal de la SEPI-ESIME-Zacatenco por el apoyo y comprensión que me brindaron durante en tiempo que estuve estudiando.

Resumen

En la actualidad la demanda de energía eléctrica en el mundo ha aumentado considerablemente debido al incremento de la actividad industrial, comercial y doméstica. Considerando que los motores eléctricos consumen gran porcentaje de dicha energía, ha cobrado gran relevancia su comportamiento energético. Estas condiciones han obligado a los diseñadores a crear motores eléctricos cada vez más eficientes con la finalidad de lograr el máximo aprovechamiento de la energía consumida, además de colaborar con el medio ambiente.

En este trabajo se presenta una estructura de red neuronal de retropropagación que permite simular el comportamiento térmico en el interior de una máquina eléctrica rotatoria con la finalidad de facilitar el estudio de dichas máquinas, inclusive para ser aplicada como herramienta de los diseñadores en la construcción de nuevas máquinas con mayor eficiencia y mejor comportamiento energético.

Para esta investigación fue necesario tomar la geometría de un motor de inducción doble jaula de ardilla de alta potencia para ajustar un motor virtual que permitiera generar datos para el entrenamiento de la red neuronal. Para ello se desarrolló un modelo matemático en dos dimensiones de espacio y una dimensión de tiempo de los procesos de transferencia de calor en la forma general de la ecuación diferencial parcial de segundo orden de tipo parabólico con condiciones iniciales y de frontera adecuadas a estos problemas. Dicha ecuación diferencial parcial de segundo orden se resolvió mediante el método de elemento finito. Posteriormente se definió una metodología para desarrollar una estructura de red neuronal y para entrenarla con los datos obtenidos del motor virtual obteniendo así las temperaturas al interior de la máquina en distintos puntos.

Es importante destacar que las redes neuronales han sido aplicadas para resolver diversos problemas en ingeniería, en particular el reconocimiento de patrones, sin embargo, hay poca evidencia en la literatura en donde se emplean para la solución de problemas en motores eléctricos siendo así esta una de las primeras investigaciones en donde se aplican para este fin dando buenos resultados.

El desarrollo de esta investigación se realizó utilizando como herramienta de programación y simulación el paquete computacional MATLAB®.

Abstract

At present the demand for electricity in the world has increased significantly due to the rise of industrial, commercial, domestic activities. Considering that electric motors consume a large percentage of that energy, its energy behavior has become important. These conditions have forced designers to make electric motors increasingly efficient in order to achieve maximum utilization of the energy consumed, as well as collaborating with the environment.

This paper presents a back-propagation neural network structure that allows us to simulate the thermal behavior inside a rotary electric machine in order to facilitate the study of such machines and even to be applied as a tool for designers in the construction of new machines with higher efficiency and better energy performance.

For this investigation was necessary to take the geometry of a high power double squirrel cage induction motor to develop a virtual engine motor that would generate data for training the neural network. To do this, was developed a mathematical model in two dimensions of space and one time dimension of the heat transfer processes in the general form of the partial differential equation of second order of parabolic type with initial and boundary conditions appropriate to these problems. This partial differential equation of second order was solved by the finite element method. Subsequently was defined a methodology to develop a neural network structure and to train it with data from the virtual motor thus obtaining the temperatures inside the machine.

It is important to note that neural networks have been applied to solve various engineering problems, in particular patterns recognition; however, there is little evidence in the literature where neural networks are used for solving problems in electric motors, being this one of the first investigations in which this order applies with good results.

The development of this research was conducted using as a tool for programming and simulation, the computer software MATLAB®.

Contenido

Índice de Figuras

Capítulo 2

Capítulo 3

Capítulo 4

Índice de Tablas

Capítulo 2

Capítulo 4

Simbología y Abreviaturas

SÍMBOLO	SIGNIFICADO
R	Resistencia óhmica
I	Corriente
C	Carbono
T	Temperatura absorbida a que se le somete un aislante.
O	Oxigeno
A	Sección transversal
H	Calor transferido por unidad de tiempo
t	Tiempo
ρ	Densidad
C	Calor específico
k	Conductividad térmica
$T(x,y,t)$	Temperaturas
$T_o(x,y)$	Temperatura inicial
"T_o"	Representa el vector de valores iniciales de temperatura.
$\nabla = \frac{\partial}{\partial x} + \frac{\partial}{\partial y}$	Operador del gradiente
$\vec{n}$	Vector normal a la superficie de transferencia de calor por convección.
α	Coeficiente de transferencia de calor por convección.
α_{rotor}	Coeficiente de transferencia de calor por convección en el rotor del lado del entrehierro.
$\alpha_{estator}$	Coeficiente de transferencia de calor por convección en el estator del lado del entrehierro.
h	Coeficiente de convección
$f(x,y)$	Intensidad de las fuentes internas de calor generado por pérdidas eléctricas y/o magnéticas en devanado y circuito magnético del estator y/o rotor.

SÍMBOLO	SIGNIFICADO
Ω	Dominio continuo de trasferencia de calor.
$\Omega_{Cu} \subset \Omega$	Dominio continuo de transferencia de calor correspondiente al devanado del motor.
$\Omega_{Fe} \subset \Omega$	Dominio continuo de trasferencia de calor correspondiente al circuito magnético del motor.
$\partial\Omega$	Fronteras del dominio continuo Ω
Cu	Cobre, devanado, circuito eléctrico.
Fe	Acero, circuito magnético.
V	Voltaje.
x_i	Vector de entradas.
ΔT	Diferencia de temperaturas.
HP	Potencia.
x, y	Coordenada del espacio, abscisa y ordenada respectivamente.
p, e, t	Representan matrices de los parámetros de la malla generada por el MEF.
c, a, d	Representan los vectores de los coeficientes de los pará metros para la EDP.
b	Representa la matriz de coeficientes de transferencia de calor conectiva de las condiciones de frontera establecidas.
i	Nodo i-ésimo
ΔQ	Calor transferido
Modelo matemático 2D+1	Modelo matemático en dos dimensiones de espacio y una dimensión de tiempo.
U_1	Matriz de temperaturas en todos los nodos de las mallas del segmento de corte axial del estator o rotor.
U_0	Vector de temperaturas iniciales.
∞	Infinito.

SÍMBOLO	SIGNIFICADO
Q_{de}	Pérdidas eléctricas en el devanado del estator.
Q_{cme}	Pérdidas eléctricas en el circuito magnético del estator.
Q_{dr}	Pérdidas eléctricas en el devanado del rotor.
Q_{cmr}	Pérdidas eléctricas en el circuito magnético del rotor.
$T_{1,2,3,4}$	Temperaturas generadas en nodos del estator y rotor
$T_{deseada}$	Temperatura generada por el motor virtual
$T_{obtenida}$	Temperatura generada por la simulación de la red neuronal.
w_i	Vector de pesos
o_i	Vector de salidas
$f(w^t x)$	Función de activación
net	Producto escalar de los vectores de pesos y entradas
λ	Constante
$error$	Error relativo de la simulación de la red neuronal

ABREVIATURAS	SIGNIFICADO
S.E.P.I.	Sección de Estudios de Posgrado e Investigación
E.S.I.M.E.	Escuela Superior de Ingeniería Mecánica y Eléctrica
I.P.N.	Instituto Politécnico Nacional
S.I.P.	Secretaría de Investigación y Posgrado
P.I.F.I.	Programa Institucional de Formación de Investigadores
MATLAB	MATrix LABoratory
EDP	Ecuación Diferencial Parcial
MEF	Método del Elemento Finito
MDF	Método de Diferencias Finitas
IEEE	American Institute of Electrical Engineers
RVP	Reunión de Verano de Potencia

ABREVIATURAS	SIGNIFICADO
ROC&C	Reunión de Otoño de Comunicaciones y Computación
Toolbox PDEtool	Partial Differential Equation Toolbox
GUI	Graphical User Interface
RNA	Redes Neuronales Artificiales
RPM	Revoluciones Por Minuto
EUA	Estados Unidos de América
Toolbox NNtool	Neural Networks Toolbox
ART	Adaptative Resonance Theory
TLU	Thrershold Logic Unit

CAPÍTULO 1

INTRODUCCIÓN

1.1 Generalidades

Este trabajo da a conocer la investigación realizada en el proyecto titulado “Aplicación de redes neuronales para la determinación y minimización de pérdidas generadas en máquinas eléctricas” registrado en la secretaria de investigación y posgrado del IPN.

El propósito de este proyecto de investigación es desarrollar un modelo en redes neuronales de retropropagación del comportamiento térmico de un motor de inducción doble jaula de ardilla con la finalidad de simular las temperaturas que se generan al interior de la misma en un intervalo de tiempo.

Los resultados de la investigación pueden servir a los diseñadores en el análisis de pérdidas eléctricas para hacer una mejor elección de los materiales aislantes en el diseño de motores eléctricos, ya que con la herramienta desarrollada es posible simular las temperaturas del motor en cualquier punto del rotor y del estator, además es posible determinar las temperaturas máximas, etc. Dependiendo de las necesidades del usuario como lo son temperaturas máximas en el motor o temperaturas en un punto especifico de la máquina.

La metodología que se propone para modelar térmicamente la máquina eléctrica mediante la red neuronal propuesta en la investigación, se desarrolló utilizando el paquete computacional MATLAB® como herramienta de simulación y programación. Debido a que no se contaba con mediciones de motores reales y considerando que se necesitarían muchos motores para obtener diversos conjuntos de datos, se trabajó con los datos de un motor de inducción doble jaula de ardilla de alta potencia para adaptar un motor virtual en la obtención de las temperaturas a través de sensores virtuales, obteniendo los conjuntos de entrenamiento de la red neuronal. Este motor virtual fue desarrollado en [1] y se adapto para generar los conjuntos de entrenamiento de la red neuronal propuesta.

Al contar con información suficiente para el entrenamiento fue posible simular distintas configuraciones de redes neuronales con distintos métodos de optimización a fin de obtener los resultados más exactos posibles.

Es importante destacar que las redes neuronales han sido aplicadas para resolver diversos problemas en ingeniería, en particular el reconocimiento de patrones; sin embargo, hay poca evidencia bibliográfica en donde las redes neuronales se emplean para la solución de problemas en motores eléctricos, por ello ésta se convierte en una de las primeras investigaciones donde las redes neuronales son aplicadas para el fin mencionado obteniendo buenos resultados.

1.2 Definición del problema

El modelo térmico de un motor eléctrico es igual de importante que el modelo mecánico y electromagnético debido a que todo calor generado en la máquina se considera pérdida de energía, lo cual ha cobrado importancia debido a los severos problemas ambientales que se presentan. Por ello es necesario construir motores de alta eficiencia en donde el modelado térmico de los diseños cobra gran relevancia.

Los motores eléctricos son los principales convertidores de energía eléctrica a energía mecánica, sin embargo durante esta conversión se genera energía calorífica que no se utiliza en dicho proceso, por lo tanto se considera como pérdida de energía [2, 3, 4,5].

Las pérdidas de energía se pueden clasificar de distintas formas, en lo que respecta a su naturaleza se pueden clasificar en cuatro grupos:

- Pérdidas en los conductores.

Estas pérdidas dependen del cuadrado de la corriente, es decir, se originan por la circulación de corriente eléctrica a través de un conductor, manifestándose en forma de calor tanto en el estator como en el rotor. Se dividen en dos zonas: estator (RI^2 en las bobinas del estator) y rotor (RI^2 en las bobinas del rotor) [3].

- Pérdidas en el núcleo magnético

Estas pérdidas tienen dos componentes, las pérdidas por corrientes de Eddy y las pérdidas por el fenómeno de histéresis, estas pérdidas son debidas a alteraciones del campo magnético en el material activo del estator y el rotor incluyendo las pérdidas superficiales en la estructura magnética del motor [6,7].

- Pérdidas por fricción y ventilación

Estas pérdidas son debidas a la fricción en los rodamientos y a las pérdidas por resistencia del aire al giro del ventilador y de otros elementos rotativos del motor. La fricción en los rodamientos es una función de las dimensiones de éste, de la velocidad, del tipo de rodamiento, de la carga y de la lubricación usada. Estas pérdidas quedan relativamente fijadas para un tipo de diseño, y debido a que constituyen un porcentaje pequeño de las pérdidas totales del motor, los cambios que se pueden hacer en el diseño para reducirlas no afectan significativamente la eficiencia del motor [8,9].

- Pérdidas adicionales

Son pérdidas residuales difíciles de determinar por medio de mediciones directas o de cálculos. La naturaleza de estas pérdidas es muy compleja, están en función de muchos factores de diseño y de fabricación del motor. Algunos de los elementos que influyen en éstas pérdidas son: el diseño del devanado, la relación entre la magnitud del entrehierro y la abertura de las ranuras, la relación entre el número de las ranuras del estator y del rotor, la inducción en el entrehierro, las condiciones en la superficie del rotor, el tipo de contacto superficial entre las barras y las laminaciones del rotor [2, 10]

Debido a estas pérdidas de energía se genera calor, que en exceso, puede disminuir considerablemente la vida útil de los motores, además debilita los aislamientos y como consecuencia disminuye la eficiencia energética de las máquinas.

Con la finalidad de reducir dichas pérdidas se han realizado diversos estudios para simular la transmisión de calor provocada por los motores eléctricos y para ello se ha

utilizado la computadora como herramienta aplicada a simular los modelos matemáticos de gran parte de los fenómenos que se presentan en máquinas eléctricas [11,12].

A través de los años los diseñadores han desarrollado modelos matemáticos de los motores cada vez más exactos que han ayudado a minimizar esas pérdidas de energía y para ello se han valido de diversas herramientas para simular campos de temperaturas, siendo el método de elemento finito (MEF) el más usado recientemente; sin embargo, para la ejecución de dichos métodos se requiere de mucho tiempo de cómputo así como de conocimientos sólidos de los modelos matemáticos y las ecuaciones que rigen a dichos modelos.

Considerando lo anterior, en este trabajo de investigación se propone una herramienta diferente para realizar estos estudios en máquinas eléctricas la cual consiste en una red neuronal que permite simular procesos térmicos al interior de la máquina de forma rápida, sin amplios conocimientos matemáticos y reduciendo ampliamente el tiempo de cómputo.

1.3 Objetivo de la tesis

Desarrollar un modelo térmico en redes neuronales de una máquina eléctrica rotatoria de inducción doble jaula de ardilla que permita conocer el comportamiento de las temperaturas generadas en el interior de la máquina de forma rápida y exacta, para apoyar a los diseñadores a implementar algoritmos de optimización y minimización de pérdidas electromagnéticas.

1.4 Justificación

En la industria, aproximadamente el 75% de la energía consumida es debida al uso de motores eléctricos. Es significativo el hecho de que los motores eléctricos, suministran (en su mayor parte) la energía que mueve los accionamientos industriales, por lo que la operación y conservación de los motores en la industria representa uno de los campos más fértiles de oportunidades en el ahorro de energía, que se traducen en una reducción en los costos de producción y en una mayor competitividad. Los mayores ahorros de energía eléctrica se obtienen cuando el motor opera a su máxima eficiencia. El

rendimiento máximo de un motor se obtiene cuando éste opera entre un 75% y un 95% de su potencia nominal [13].

La energía eléctrica es uno de los factores de producción y de costos más importantes. Estudios independientes han demostrado que aproximadamente el 30% de todos los motores eléctricos pueden funcionar de manera más eficaz, permitiendo ahorros de energía de hasta el 50% en condiciones de funcionamiento óptimas [13].

Entre las máquinas eléctricas con un potencial de ahorro comparativamente alto se incluyen bombas, ventiladores, compresores, cintas transportadoras, mezcladoras, molinos y extrusoras [13].

El ahorro de energía es un factor importante en el crecimiento económico de un país. Por ello, los países desarrollados han dado mayor importancia a diseñar motores cada vez más eficientes en el proceso de conversión. Aunque esta conversión de energía tiene una alta eficiencia, una pequeña mejora en la capacidad por el uso de motores más eficientes puede conducir a ahorros significativos de energía eléctrica [14].

El tema de la eficiencia de motores toma cada vez más importancia en el mundo, como lo muestran estudios realizados por diversas organizaciones a nivel global, las cuales resaltan la necesidad de reducir al máximo dichas pérdidas, que a gran escala representan grandes cantidades de dinero y afectan fuertemente los problemas ecológicos a nivel mundial [14].

En general, las máquinas eléctricas son eficientes y confiables cuando se usan apropiadamente, pero muchos factores en el diseño y operación de sistemas de maquinaria eléctrica pueden desperdiciar energía y causar fallas prematuras. Debido a su gran importancia económica y energética, es esencial optimizar la eficiencia y confiabilidad de los sistemas de máquinas eléctricas. Como ejemplo está un reporte del Centro de Investigación adjunto a la Comisión Europea que propone siete sistemas para que se pueda ahorrar cantidades sustanciales de energía, donde se identifica un potencial de ahorro total de 435TWh/año en Europa, equivalente a 200 millones de toneladas de emisiones de CO_2. Sorpresivamente, casi la mitad del potencial de ahorro reside en

sistemas de motores eléctricos, de los cuales aproximadamente el 80% del consumo en países desarrollados se debe a motores de inducción. Por lo anterior, el potencial de ahorro que puede obtenerse con estos procesos nos da una muestra clara de la importancia que tiene el tema y la investigación. Ahora bien, la mayoría de estos estudios se sitúan en la Comunidad Europea; sin embargo, es muy probable que en el futuro puedan verse resultados similares en países en vía de desarrollo. Además, es evidente que con estos procesos de ahorro, compañías industriales y usuarios individuales se podrán beneficiar ampliamente con maquinaria eléctrica bien diseñada y de alta eficiencia; lo que es más, el hecho de que sean ambientalmente amigables, sin duda es un argumento a favor tanto de las máquinas como del tema, argumento fácilmente utilizable para empresas o compañías de marketing [14].

1.5 Estado del arte

Este apartado se divide en dos partes, en la primera se presentan los trabajos relacionados con el modelado térmico de máquinas eléctricas rotatorias desarrollados a lo largo de la historia, en la segunda parte se dan a conocer las principales aplicaciones de las redes neuronales en máquinas eléctricas.

1.5.1 Antecedentes de estudios en máquinas eléctricas

Los inicios de la máquina eléctrica rotatoria pueden remontarse al tiempo que Faraday descubrió la ley de inducción electromagnética alrededor de 1831 y cuando Maxwell formuló las leyes de la electricidad (o las ecuaciones de Maxwell) en torno a 1860. El conocimiento había llegado por la invención de la máquina de inducción que tiene dos padres: Galileo Ferraris (1885) y Nikola Tesla (1886). En la patente de Ferraris, el rotor se hizo de un cilindro de cobre, mientras que en la Patente de Tesla el rotor se hizo de un cilindro ferromagnético provisto con barras en las bobinas [15].

Posteriormente hubo otros científicos que continuaron desarrollando diversos modelos, pero fue a partir de 1888 y principalmente en 1896, cuando el diseño de las máquinas eléctricas rotatorias se dio de forma funcional, por lo que se inició su comercialización. Para 1970 se le dio un auge a la nueva era de los motores eléctricos debido a los avances

tecnológicos de la época y a que se mejoró la calidad del acero, hubo un progreso en el diseño de los aislamientos y las características de construcción por lo que fue posible construir motores pequeños y con mejor rendimiento que los anteriores, ello repercutió en la economía de los costos de producción [15].

Lamentablemente los adelantos que se dieron en el diseño de motores no contemplaron la reducción de pérdidas eléctricas, el enfoque se dio básicamente en tratar de reducir los costos de producción en cuanto a la selección de materiales y no se enfocaron en mejorar la eficiencia energética.

Actualmente se sabe que en la conversión de energía eléctrica a energía mecánica una parte de esta energía se pierde en forma de calor en las piezas activas y constructivas de la maquina eléctrica y a este fenómeno se le considera como transferencia de calor, el cual forma parte de la teoría de calor que fue desarrollada en 1824 en Paris- Francia por el ingeniero Leonard Sadi Carnot [16].

En dicha teoría se expuso los dos primeros principios de la termodinámica en un documento llamado *"Reflexiones sobre la fuerza motriz del calor y sobre las máquinas capaces de desarrollar esta fuerza",* lamentablemente este trabajo no fue comprendido por los científicos de esa época [16].

Fue hasta 1834 cuando Emile Clapeyron de la Escuela Politécnica de Paris rescribió el trabajo de Carnot con un carácter rigurosamente matemático utilizando la representación gráfica de los procesos térmicos [17].

A partir de1840 se comenzó formular la Ley de la Conservación de la Energía, conocida como el primer principio de la termodinámica por varios investigadores entre ellos Julius Von Meyer y James Joule considerada como la más importante de la naturaleza. Aunque fueron Thompson y Clausius quienes escribieron los primeros enunciados formales de dicha ley en 1850 y 1851 [17].

Debido a la relación entre calor y energía descubierta por estos investigadores se le comenzó a prestar más atención a los procesos de intercambio térmico para la segunda

mitad del siglo XIX. Actualmente el análisis térmico es de vital importancia en el desarrollo de máquinas eléctricas debido a que se pretende construir máquinas más eficientes para resolver los problemas económicos y ambientales que se viven en nuestros días [1].

Los estudios térmicos de las máquinas eléctricas se han obtenido por técnicas analíticas, o por medio de circuitos térmicos equivalentes usando modelos con parámetros concentrados. Estos modelos se basan en un circuito eléctrico equivalente al modelo térmico, en el cual se supone que los devanados y los circuitos magnéticos del motor de inducción son cuerpos homogéneos, es decir, los representan con resistencias eléctricas y sólo se puede hacer el análisis de temperaturas en estado estable [1].

En los últimos años los diseñadores de máquinas eléctricas han tratado de mejorar la eficiencia energética de los motores eléctricos, se han auxiliado de modelos matemáticos para sus ensayos y pruebas, tomando como herramientas los avances en las técnicas de solución por medio de métodos numéricos (esquemas de diferencias finitas, técnicas numéricas e integrales y formulaciones variacionales) y de las crecientes velocidades de procesamiento en las computadoras.

Los métodos más comunes para resolver este tipo de modelos son el Método del Elemento Finito (MEF), y el Método de Diferencias Finitas (MDF). Estos métodos son aplicables a todos los problemas de mecánica del medio continuo, y problemas físicos en general, que sean definidos por ecuaciones diferenciales, y se pueden aplicar a elementos compuestos de diferentes materiales, con propiedad física distinta. Estos métodos son los más convenientes para el análisis de sistemas complejos [18].

El MEF se introdujo por primera vez para el análisis térmico del núcleo del estator de los grandes turbogeneradores por A.F. Armor y Chari en el año 1976. No obstante su trabajo esta limitado debido a que en su estudio no considera la influencia de calor en las bobinas del estator. En 1980 A.F. Armor empleó el MEF para resolver el flujo de calor transitorio, y para determinar en forma matricial el campo térmico en motores de inducción [18].

P.H. Mellor, D. Roberts y D.R. Turner en su trabajo publicado en 1991, *“Lumped Parameter thermal model for electrical machines of TEFC design*” calculan las

temperaturas de 3 motores de inducción de diferentes capacidades mediante el uso de pequeños termómetros con resistencia de platino colocados en el interior de los motores reduciéndolo a un modelo de ocho ecuaciones diferenciales lineales lo cual les permitió obtener las temperaturas en estado transitorio y en estado estable de una forma eficiente [19].

Driesen Johan, Belmans Ronnie y Hameyer Kay en su trabajo publicado en 2001, *" Finite-Element Modeling of Thermal Contact Resistances and Insulation Layers in Electrical Machines"* muestran el desarrollo matemático del MEF para modelar las temperaturas en un motor síncrono de imanes permanentes y muestran las pérdidas generadas en diversas partes de la máquina [20].

Se demuestra que el uso del MEF genera mejores resultados que con las anteriores herramientas usadas para simular campos de temperaturas [20].

El análisis térmico es un paso importante en el diseño de los motores de inducción. Sus principios requieren el cálculo de las corrientes inducidas en los rotores y, obviamente, este fenómeno está relacionado directamente con el calentamiento [21].

Las corrientes que circulan en la bobina del estator también calientan la máquina. Bastos J.P., Cabreira M., Sadowski N. y Arruda S. en 1997 con *"A Thermal Analysis of Induction Motors Using Weak Coupled Modeling"* presentan el análisis térmico de un motor de inducción en base a los fenómenos eléctricos y térmicos mediante un modelo acoplado en donde se propone tomar en cuenta los estados transitorios, los resultados obtenidos son comparados con cálculos previos mostrando así que es posible estimar temperaturas con este modelo [21].

1.5.2 Antecedentes de estudios con redes neuronales

Las simulaciones de redes neuronales no son un desarrollo reciente. Este campo fue establecido antes del desarrollo de las computadoras, pero su verdadero desarrollo tuvo lugar cuando las simulaciones por computadora fueron factibles por capacidad de procesamiento y bajo costo. Luego de un periodo inicial, las redes neuronales cayeron en

un periodo de desprestigio. Durante este periodo, cuando el soporte económico y computacional era mínimo, solo unos pocos investigadores consiguieron logros importantes. Estos pioneros fueron capaces de desarrollar una tecnología que sobrepasara las limitaciones identificadas por Minsky y Papert en 1969 [22], cuando publicaron un libro con el que sembraron un desencanto y frustración general en la comunidad científica contra las redes neuronales, que aceptó las conclusiones de estos investigadores sin un mayor análisis.

Actualmente, las redes neuronales son un campo en el cual resurgió el interés y en correspondencia se ha incrementado el financiamiento para investigar dicho campo. Por ello la historia de las redes neuronales puede dividirse en varios periodos.

1.5.2.1 Primeros intentos

Se realizaron algunas simulaciones usando lógica formal. McCulloch y Pitts (1943) [23] desarrollaron modelos de redes neuronales basados en su conocimiento de neurología. Dichos modelos tenían varios supuestos acerca del funcionamiento de las neuronas. Sus redes se basaban en neuronas simples, consideradas como dispositivos binarios con umbrales fijos. Los resultados de sus modelos fueron funciones lógicas elementales tales como "a o b" y "a y b". Dos grupos lo intentaron usando simulaciones computacionales. El primer grupo conformado por investigadores de IBM mantenía un contacto cercano con neurocientíficos de la universidad de McGill, estableciendo una tendencia interdisciplinaria que se mantiene en la actualidad [24].

1.5.2.2 Tecnología emergente y promisoria

No solo la neurociencia influía en el desarrollo de las redes neuronales, también los físicos y los ingenieros contribuían al progreso de las simulaciones de redes neuronales. Rosenblatt (1958) [25] revitalizó fuertemente el interés y la actividad en esta área cuando diseñó y desarrolló su Perceptrón. El Perceptrón tiene tres capas, incluida la capa intermedia denominada capa de asociación. Este sistema pudo aprender a conectar y asociar unas entradas dadas a una unidad de salida aleatoria. Otro sistema fue el ADALINE (Adaptive Linear Element) el cual fue desarrollado en 1960 por Widrow and Hoff (de la Universidad de Stanford) [26]. El ADALINE fue un dispositivo electrónico analógico

hecho de componentes simples, con un método de aprendizaje diferente al del Perceptrón, empleando una regla de aprendizaje basada en mínimos cuadrados (LMS) [24].

1.5.2.3 Periodo de frustración y desprestigio

En 1969 Minsky y Papert [22] escribieron un libro en el cual ellos generalizaban las limitaciones de un Perceptrón monocapa a sistemas multicapa. En el libro ellos decían: "nuestro intuitivo juicio es que la extensión (a sistemas multicapa) es una tarea estéril". El resultado de las afirmaciones de este libro fue eliminar la financiación para los investigadores que trabajaban con simulaciones de redes neuronales. Las conclusiones del libro de Minsky y Papert soportadas con el desencanto de los investigadores en el área, trajo como resultado un gran prejuicio contra la actividad de la misma [24].

1.5.2.4 Innovación

Aunque el interés público por las redes neuronales era mínimo, varios investigadores continuaron trabajando en el desarrollo de métodos computacionales basados en neuromorfología para problemas de identificación de patrones. Durante este periodo se crearon varios paradigmas, los cuales aun continúan con trabajos modernos, se puede mencionar a Steve Grossberg y Gail Carpenter quienes desarrollaron la teoría de la resonancia adaptativa, ART (Adaptive Resonance Theory) [27], Anderson y Kohonen (1982) [28] quienes desarrollaron técnicas para aprendizaje asociativo, así como Hopfield (1984) quien desarrolló una red neuronal haciendo un símil energético, Werbos (Paul Werbos 1982) [29] desarrolló y usó el método de aprendizaje conocido como retropropagación, que varios años después logró popularizarse; ésta es actualmente la arquitectura de red neuronal mejor conocida y más utilizada en las aplicaciones de redes neuronales. En esencia una red de retropropagación es un perceptrón con múltiples capas, con diferentes funciones de activación en las neuronas artificiales y con una regla de aprendizaje más robusta y confiable [24].

1.5.2.5 Resurgimiento

Durante el final de la década de los 70s y principios de los 80s, fue importante el resurgimiento del interés en el campo de las redes neuronales. Varios factores han influenciado este movimiento, tales como la aparición de libros y conferencias que han dado a conocer las bondades de esta técnica a personas de diferentes áreas. Introducción de cursos en los programas académicos de las principales universidades europeas y americanas. El financiamiento a proyectos de investigación en redes neuronales en Europa, EUA y Japón que han hecho que aparezcan una gran variedad de aplicaciones comerciales e industriales [24].

1.5.2.6 Actualidad

Se han realizado progresos muy significativos en el campo de las RNA, lo suficientes como para atraer una gran atención e interés en financiar investigaciones. Ya se encuentran comercialmente circuitos integrados basados en RNA y las aplicaciones desarrolladas resuelven problemas cada vez más complejos. Sin lugar a dudas, hoy es un periodo de transición y fuerte evolución para la tecnología en redes neuronales [24].

1.5.2.7 Otros trabajos

En 1986 Mc Clelland y Rumelhart publicaron un libro de dos volúmenes titulado: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* Este libro se considera un clásico en el área de redes neuronales y se puede decir que su aparición significó un nuevo impulso a la investigación en sistemas neuronales al mostrar las ventajas y desventajas de las redes neuronales artificiales (RNA) [30].

Para 1990, Kumpati S. Narendra y Kannan Parthasarathy realizaron una investigación cuyo trabajo fue publicado con el nombre de *"Identification and Control of Dynamical Systems Using Neural Networks"*, en éste plantean el uso de redes neuronales para identificar y controlar sistemas dinámicos no lineales y demuestran la eficiencia de las redes neuronales usando el algoritmo de retropropagacion en redes multicapa y redes recurrentes [31].

Kumpati S. Narendra y Kannan Parthasarathy publicaron en 1991 *"Gradient Methods for the Optimization of Dynamical Systems Containing Neural Networks"* explican el uso de los métodos de gradiente descendiente de sistemas estáticos y sistemas dinámicos usando redes neuronales multicapa, explican también el algoritmo de la retropropagacion y muestran resultados del uso de estos algoritmos con sistemas dinámicos no lineales [32].

Polycarpou Marios y Ioannou Petros en 1992 hacen del dominio público su investigación denominada *"Modelling, Identification and Stable Adaptative Control of Continuous-Time Nonlinear Dynamical Systems Using Neudal Networks"* en la cual muestran el desarrollo matemático para sintetizar y analizar redes neuronales basadas en identificación y diagramas de control. Se presenta un procedimiento de diseño de redes neuronales basado en teoría de estabilidad para modelar, identificar y controlar sistemas dinámicos no lineales continuos en el tiempo usando redes estáticas de tipo sigmoidal y las redes de funciones de base radial [33].

Penman J. y Yin C.M. publica en 1994 *"Feasibility of using unsupervised learning, artificial neural network for the condition monitoring of electrical machines"* utilizan los mapas auto-organizables de Kohonen para monitorear un motor de inducción con carga puesto que el análisis con sensores presentaba variaciones y era poco exacto. Demuestran la efectividad del uso de redes neuronales, pero muestran algunas limitaciones del aprendizaje no supervisado puesto que algunas decisiones pueden ser ambiguas [34].

Wishart Michael T. y Harley Ronald G. cuyo trabajo *"Identification and Control of Induction Machines Using Artificial Neural Networks"* fue publicado en 1995, en donde prueban exitosamente el uso de redes neuronales artificiales para identificación y control de motores de inducción con dos esquemas distintos. El primero consistió en controlar la corriente en el estator para seguir un valor deseado y el segundo fue controlar la velocidad en el rotor. Ambos esquemas muestran buenas respuestas en la simulación aunque hacen algunas recomendaciones para obtener mejores resultados en trabajos posteriores [35].

En 2004, Hammer M., Tozlovsky T. y Svoboda J. publican *"The Use of Neural Networks for the Life Prediction of Insulating Material of Electric Rotating Machines"* donde dan a conocer el uso de redes neuronales para la predicción de la vida útil del aislamiento Relanex que se aplica como aislamiento de los embobinados en máquinas eléctricas. Emplean el algoritmo de retropropagación para el entrenamiento con el método de Levenberg-Marquardt como método de optimización [36].

1.5.3 Trabajos relacionados desarrollados en la Sección de Estudios de Posgrado e Investigación

Por otro lado en la Sección de Estudios de Posgrado e Investigación (S.E.P.I.) desde 1998 se han realizado diversos trabajos enfocados a la reducción y minimización de pérdidas electromagnéticas modelando los campos de temperaturas generadas en estatores y rotores de motores eléctricos, aplicando el método de elemento finito y el método de diferencias finitas, como es el caso de los trabajos [1,37,38,53] aunque directamente el uso de redes neuronales para simular estos procesos se ha realizado recientemente por algunos profesores investigadores en la SEPI Zacatenco en el IPN quienes han publicado diversos trabajos como [39, 40, 41].

Cabe destacar que a nivel mundial son pocos los trabajos desarrollados en este sentido, puesto que las redes neuronales han tenido gran auge en otro tipo de aplicaciones como por ejemplo el reconocimiento de patrones.

1.6 Alcances

Este trabajo de tesis sienta las bases en la utilización de redes neuronales para simular procesos térmicos en máquinas eléctricas rotatorias puesto que los trabajos dedicados a simular los comportamientos térmicos únicamente se han trabajado con el método de diferencias finitas DF y el método de elemento finito, sin embargo en esta ocasión se utilizaron las redes neuronales por su rápida respuesta y su precisión.

Esta investigación constituye una herramienta de apoyo para los diseñadores de motores eléctricos, ya que es fácil de usar y no exige conocer ampliamente los modelos matemáticos, además es posible obtener los campos de temperaturas rápidamente.

Como consecuencia es posible desarrollar motores más eficientes y de mayor durabilidad debido a que se pueden seleccionar de mejor manera los materiales de aislamiento.

1.7 Aportaciones

- Para simular el comportamiento térmico de la máquina rotatoria analizada, considerando diferentes valores de coeficientes de transferencia de calor por convección, fuentes de calor y tiempos se tomó un motor virtual en elementos finitos desarrollado en [1], el cual es capaz de recabar, procesar y almacenar información generada del modelo. A este motor virtual se le hicieron algunas adaptaciones para obtener los conjuntos de entrenamiento de la red neuronal que se propone.

- Se elaboró una metodología para simular campos de temperaturas en motores eléctricos con redes neuronales de retropropagación. Este procedimiento considera la elección de la mejor arquitectura, el número de neuronas, el método de optimización y la selección del algoritmo de entrenamiento.

- Se desarrollaron programas con el lenguaje de programación de MATLAB con base en las metodologías propuestas para la presente investigación. En primera instancia se obtienen los datos de la geometría del motor y datos de diseño de la máquina a estudiar, después se ejecuta el motor virtual para obtener los conjuntos de entrenamiento de la red neuronal, posteriormente se entrena la red neuronal hasta lograr reducir el error de entrenamiento y finalmente se simulan las temperaturas en los nodos seleccionados.

1.8 Publicaciones derivadas de la investigación

Artículos para congresos:

- M.A. Monroy Canales, T. Niewierowicz, L. Kawecki. **"Modelo Térmico en Redes Neuronales de una Maquina Eléctrica Rotatoria".** *Memorias de la Reunión de Otoño de Comunicaciones, Computación, Electrónica, Automatización, Robótica y exposición industrial del IEEE Sección México*, ROC&C 2011. Noviembre 27-Diciembre 3, 2011, Acapulco Guerrero, México.

- M.A. Monroy Canales, T. Niewierowicz, L. Kawecki. **"Aplicación de Redes Neuronales en el Modelado Térmico de Máquinas Eléctricas Rotatorias".** *Memorias de la Vigesimoquinta Reunión Internacional de Verano de Potencia, Aplicaciones Industriales y Exposición Industrial del IEEE Sección México*, RVP-AI 2012. 8-14 de Julio, 2012, Acapulco Guerrero, México.

- M.A. Monroy Canales, T. Niewierowicz, L. Kawecki... **"Simulador Neuronal de Campos de Temperaturas Generadas por Pérdidas Electromagnéticas en Máquinas Eléctricas Rotatorias".** *Memorias de la Reunión de Otoño de Comunicaciones, Computación, Electrónica, Automatización, Robótica y exposición industrial del IEEE Sección México*, ROC&C 2012. 11-15 Noviembre, 2012, Acapulco Guerrero, México.

- T. Niewierowicz, L. Kawecki., M.A. Monroy Canales. **"Determinación de Parámetros Térmicos de Motores Eléctricos aplicando Redes Neuronales".** *Memorias de la Reunión de Otoño de Comunicaciones, Computación, Electrónica, Automatización, Robótica y exposición industrial del IEEE Sección México*, ROC&C 2012. 11-15 Noviembre, 2012, Acapulco Guerrero, México.

1.9 Estructura de la tesis

En esta sección se presenta de forma general el contenido de cada uno de los capítulos que constituyen esta tesis:

Capítulo I. Se da una introducción sobre la importancia de diseñar motores eléctricos construidos cada vez más con mayor eficiencia energética, se presenta una breve definición del problema, se presenta el objetivo y se justifica el desarrollo de este trabajo. Se expone un breve estado del arte acerca de trabajos anteriores relacionados con modelos térmicos en máquinas eléctricas y las aplicaciones con redes neuronales en dichos fenómenos, además se describen las limitaciones del trabajo así como los alcances en la actualidad. Por ultimo se presentan las publicaciones derivadas de la investigación.

Capítulo II. En este capítulo se presenta una descripción detallada del modelo matemático de las temperaturas generadas por pérdidas electromagnéticas en máquinas eléctricas. Asimismo se dan las características y datos de la máquina de la cual se hizo el modelado y simulación, del mismo modo se presenta la herramienta utilizada para generar los datos del entrenamiento de la red neuronal.

Capítulo III. El tercer capítulo aborda el tema de las redes neuronales, las aplicaciones de estas y los algoritmos empleados en este trabajo. Se explica la metodología empleada para utilizar redes neuronales en la simulación de procesos térmicos en máquinas eléctricas.

Capítulo IV. En el cuarto capítulo se presentan las distintas simulaciones con diversas estructuras, números de neuronas y métodos de optimización, además se presenta el análisis de los resultados obtenidos a lo largo de la presente investigación.

Capítulo V. Finalmente en el capítulo quinto se dan a conocer las conclusiones acerca de este trabajo de investigación y se dan recomendaciones para investigaciones futuras.

Apéndice A – Teoría de redes neuronales

Este apéndice contiene los fundamentos teóricos de redes neuronales desde la definición, partes que componen una red neuronal, tipos de aprendizaje, clasificación de las mismas. Lo anterior con el objetivo de adentrarse en área de investigación.

Apéndice B – Regla delta generalizada

En este apartado se presenta la generalización que se le hizo a la regla delta para redes multicapa como es el caso de este trabajo de investigación.

Apéndice C - Método de Levenberg-Marquardt para el entrenamiento de redes neuronales

Este apéndice contiene el método de Levenberg-Marquardt y su aplicación a la optimización en el entrenamiento de redes neuronales artificiales para una mayor comprensión.

Apéndice D - Listado de programas

En este apéndice se muestra el código de los programas desarrollados en esta investigación.

Apéndice E - Temperaturas límites de los materiales aislantes

Este apartado incluye una tabla con los valores de temperatura de las clases de aislamiento así como materiales pertenecientes a cada grupo.

Apéndice F - Valores aproximados del coeficiente de transferencia de calor conectivo

Se muestra una tabla con las características de los coeficientes de transferencia de calor por convección y una lista de sus rangos para cada tipo de convección.

Apéndice G - Tipos de Redes Neuronales Artificiales

En este apéndice se muestra la clasificación de redes neuronales artificiales por su regla de aprendizaje, tipo de entrenamiento, arquitectura y aplicaciones.

Apéndice H - Funciones de entrenamiento contenidas en el Toolbox "Neural Networks" del software MATLAB

En este apéndice se resumen las funciones de entrenamiento contenidas en el Toolbox "Neural Networks" de MATLAB.

CAPÍTULO 2

MODELO MATEMÁTICO DE LAS TEMPERATURAS GENERADAS POR PÉRDIDAS ELECTROMAGNÉTICAS EN MÁQUINAS ELÉCTRICAS

2.1 Introducción

Durante la última década, el análisis térmico de las máquinas eléctricas ha comenzado a recibir mayor atención. De hecho, con las necesidades crecientes de la miniaturización, la eficiencia energética, la reducción de costos, y la necesidad de aprovechar al máximo nueva topologías y materiales, ahora es necesario analizar el circuito térmico a la misma medida que el diseño electromagnético. El aumento en la eficiencia energética repercute fuertemente en el impacto al medio ambiente como consecuencia de las emisiones de CO_2. Por otro lado también se disminuye el consumo de electricidad impactando fuertemente en la parte económica [42].

Existen diversas alternativas para modelar el comportamiento térmico en la fase de diseño de un motor eléctrico, por ejemplo, los modelos de parámetros concentrados para simular el proceso de transferencia de calor en el interior de la máquina y así, obtener las temperaturas en puntos del estator y rotor, sin embargo, la exactitud de este método es poco precisa puesto que los devanados y circuito magnético son representados con resistencias eléctricas y solo se puede hacer el análisis en estado estable [1].

Por otro lado el análisis térmico se ha realizado con otras técnicas como la que se basa en las corrientes del estator y el rotor. Existe un fuerte acoplamiento entre los fenómenos eléctricos y térmicos. El comportamiento eléctrico del dispositivo se obtiene calculando la fuerza electromotriz debida al acoplamiento de los circuitos eléctricos con el potencial magnético. Mediante la aplicación de esta metodología, los valores instantáneos de las corrientes en el estator y rotor se calculan. Estas son las fuentes térmicas. Este método puede simular diversos tipos de motor [21].

Con el propósito de obtener un modelo más robusto en cuanto a la exactitud en la simulación de temperaturas, en este trabajo se presenta un modelo matemático con parámetros distribuidos que se basa en la determinación de los campos de temperaturas generadas por pérdidas electromagnéticas, tanto en el estator como en el rotor de un motor de inducción. De esta forma es posible apreciar de mejor manera la distribución de las temperaturas en diversos puntos de la geometría del motor y además es posible ver el comportamiento de las mismas en el tiempo.

2.2 Análisis térmico de los motores eléctricos

Debido a los problemas de medio ambiente y de cambio climático que en los últimos años se han agudizado a nivel mundial, los diseñadores de motores eléctricos se han enfocado en desarrollar máquinas con mayor eficiencia energética y de menores volúmenes. Por ello ha cobrado importancia el análisis térmico del motor para garantizar la confiabilidad de los equipos. Dicho análisis permite localizar las temperaturas a lo largo de la geometría del motor, de esta forma es posible optimizar el diseño con la finalidad de elevar la funcionalidad y tiempo de servicio de las máquinas.

El diseño térmico debe asegurar que las temperaturas generadas en los devanados no excedan los límites permitidos por la clase de aislamiento [43].

La transferencia de calor en un motor de inducción depende del nivel y localización de las pérdidas, la geometría de la maquina y el método de enfriamiento.

La eliminación de calor y la distribución de temperaturas en el motor de inducción son los principales objetivos del diseño térmico. Para ello es necesario encontrar los lugares donde se presentan las temperaturas más altas en todos los puntos de la máquina, esto es fundamental para elegir los aislamientos adecuados y en consecuencia aumentar la vida útil del motor [43].

El modelado térmico de una máquina eléctrica es igualmente importante que el modelado electromagnético, sin embargo anteriormente al análisis de perdidas no se le consideraba

relevante porque el principal esfuerzo de diseño se dirigió hacia los costos iniciales de los materiales de las máquinas eléctricas.

La gran mayoría de los motores eléctricos son enfriados por aire, de modo que la temperatura ambiente o aire circundante determina la eficiencia del enfriamiento del motor. Las altas temperaturas gradualmente oxidan y carbonizan el material aislante, lo que reduce la capacidad del aislamiento.

2.2.1 Distribución de temperaturas en máquinas eléctricas

La distribución de las temperaturas dentro de un motor eléctrico es complejo de analizar con precisión, por el efecto tridimensional del problema, la correcta localización de las fuentes de calor y sus parámetros difíciles de predecir, como la resistencia térmica de contacto entre los devanados y las ranuras. El aspecto más importante del problema de la distribución de la temperatura es encontrar la ubicación de la temperatura más alta en el motor, dada una cierta distribución de las pérdidas y una tasa conocida de transferencia de calor [37].

La distribución de la temperatura en estado estacionario puede ser muy diferente de la distribución en estado transitorio y de hecho diferentes métodos de análisis pueden ser necesarios para los dos casos.

El modelado de estos sistemas térmicos se dificulta, ya que la temperatura no suele ser homogénea en los cuerpos, por lo que es necesario una descripción con magnitudes evaluadas en cada instante y en cada punto del espacio, lo que dará lugar a las ecuaciones diferenciales parciales (EDP) y, por tanto, a modelos con parámetros distribuidos.

En el apéndice E se muestran los tipos de aislamientos y las temperaturas permisibles de los materiales.

Para lograr mejores resultados en los modelos térmicos fue necesario desarrollar métodos para resolver las EDP de los modelos con parámetros distribuidos. Los métodos más comunes para resolver este tipo de modelos son el método de diferencias finitas y el método de elementos finitos. Estos métodos son aplicables a elementos compuestos de diferentes materiales, con propiedades físicas distintas [37].

Diferencias finitas es uno de los métodos numéricos ampliamente usados, a pesar de que este método determina las temperaturas de los puntos más calientes, el método no es tan flexible como el MEF en el manejo de las condiciones de frontera y las geometrías complejas [37].

La ventaja principal del MEF es, que la geometría de cualquier dispositivo se puede modelar, sin embargo, es muy exigente en términos de la configuración del modelo y el tiempo de cómputo [37].

2.2.2 Transferencia de calor en motores eléctricos

El proceso de transferencia de calor se da de tres formas: conducción, convección y radiación, sin embargo en las máquinas eléctricas cobran relevancia únicamente la conducción y la convección puesto que las características de la radiación se presentan en condiciones distintas a las de las propias máquinas eléctricas.

2.2.2.1 Conducción de calor

La conducción es el mecanismo de transferencia de calor en escala atómica a través de la materia por actividad molecular, por el choque de unas moléculas con otras, donde las partículas más energéticas le entregan energía a las menos energéticas, produciéndose un flujo de calor desde las temperaturas más altas a las más bajas. Los mejores conductores de calor son los metales. El aire es un mal conductor del calor. Los objetos malos conductores como el aire o plásticos se llaman aislantes [44].

La conducción de calor sólo ocurre si hay diferencias de temperatura entre dos partes del medio conductor. Para un volumen de espesor Δx, con área de sección transversal A y cuyas caras opuestas se encuentran a diferentes T_1 y T_2, con $T_2 > T_1$, como se muestra en al Fig. 2.1, se encuentra que el calor ΔQ transferido en un tiempo Δt fluye del extremo caliente al frío. Si se llama *H* al calor transferido por unidad de tiempo, la rapidez de transferencia de calor *H* = $\Delta Q / \Delta t$, está dada por la ***ley de la conducción de calor de Fourier*** [44].

$$H = \frac{dQ}{dt} = -kA\frac{dT}{dx} \tag{2.1}$$

Donde *k* (en $[W/m°C]$) se llama ***conductividad térmica*** del material, magnitud que representa la capacidad con la cual la sustancia conduce calor y produce la consiguiente variación de temperatura; y $\frac{dT}{dx}$ es el gradiente de temperatura. El signo menos indica que la conducción de calor es en la dirección decreciente de la temperatura.

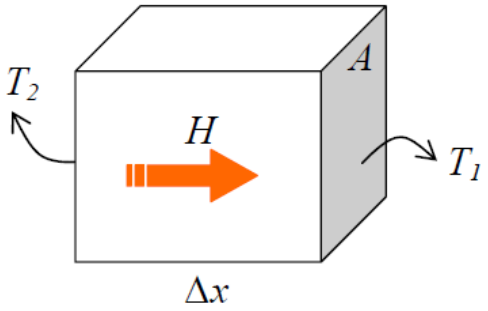


Fig. 2.1 Conducción de Calor

2.2.2.2 Convección de calor

La convección es el mecanismo de transferencia de calor por movimiento de masa o circulación dentro de la sustancia. Puede ser natural producida solo por las diferencias de densidades de la materia; o forzada, cuando la materia es obligada a moverse de un lugar a otro, por ejemplo el aire con un ventilador o el agua con una bomba. Sólo se produce en líquidos y gases donde los átomos y moléculas son libres de moverse en el medio [44].

En la naturaleza, la mayor parte del calor ganado por la atmósfera por conducción y radiación cerca de la superficie, es transportado a otras capas o niveles de la atmósfera por convección.

Un modelo de transferencia de calor *M* por convección, llamado ***ley de enfriamiento de Newton***, es el siguiente:

$$M = \alpha A(T_A - T) \tag{2.2}$$

Donde α se llama coeficiente de convección, en $\left[W / m^2 °C\right]$, *A* es la superficie que entrega calor con una temperatura T_A al fluido adyacente, que se encuentra a una temperatura T, como se muestra en el esquema de la Fig. 2.2.

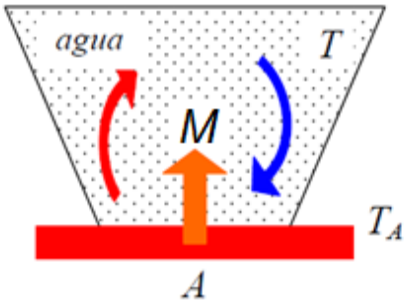


Fig. 2.2 Convección de Calor

El flujo de calor por convección es positivo (*H > 0*) si el calor se transfiere desde la superficie de área *A* al fluido ($T_A > T$) y negativo si el calor se transfiere desde el fluido hacia la superficie ($T_A < T$).

2.3 Motor eléctrico base del estudio

Para esta investigación se consideró un motor de inducción de alta potencia [37] para obtener los campos de temperaturas, aunque el procedimiento propuesto en esta tesis se puede considerar para diversos tipos de motor eléctrico de distintas dimensiones.

En la Tabla 2.1 se presentan los datos técnicos de operación del motor eléctrico así como los parámetros de diseño del rotor y estator [37].

Tabla 2.1 Datos técnicos y de diseño de la maquina eléctrica a analizar.

DATOS TÉCNICOS Y DE DISEÑO DE LA MAQUINA ELÉCTRICA A ANALIZAR	
Potencia de salida (HP)	250
Voltaje recomendado (V)	2300
Frecuencia (Hz)	60
Velocidad recomendada (rpm)	1779
Datos de diseño del núcleo del estator	
Diámetro interior(mm)	331
Diámetro exterior(mm)	549.7
Ranuras	48
Datos de diseño del núcleo del rotor	
Diámetro interior(mm)	119.85
Diámetro exterior(mm)	326.48
Ranuras	36

A continuación se presenta la máquina eléctrica rotatoria tomada de la referencia [37] para este trabajo además del rotor y estator que son las áreas activas de interés para el análisis de la máquina.

La geometría del rotor cuenta con doble ranura y el estator tiene ranuras de forma rectangular. Para nuestro estudio es importante proponer un modelo preciso, esto debido, a que el método numérico utilizado obtiene mejores resultados con una geometría detallada. Lo anterior se muestra en la Fig. 2.3.

a) **Motor eléctrico**

b) Ranuras del estator

b) Ranuras del rotor

Fig. 2.3 Máquina eléctrica a analizar

2.4 Metodología propuesta del trabajo de investigación

En este trabajo se propone desarrollar una herramienta en redes neuronales que permita la simulación de campos de temperatura en nodos seleccionados de la geometría del rotor y estator de un motor eléctrico, con la finalidad de facilitar estudios de comportamiento térmico, siendo una herramienta útil en la fase de diseño de motores eléctricos. Para mejor comprensión, el desarrollo de dicho modelo se divide en 7 fases. En la Fig. 2.4 se muestra el procedimiento a seguir.

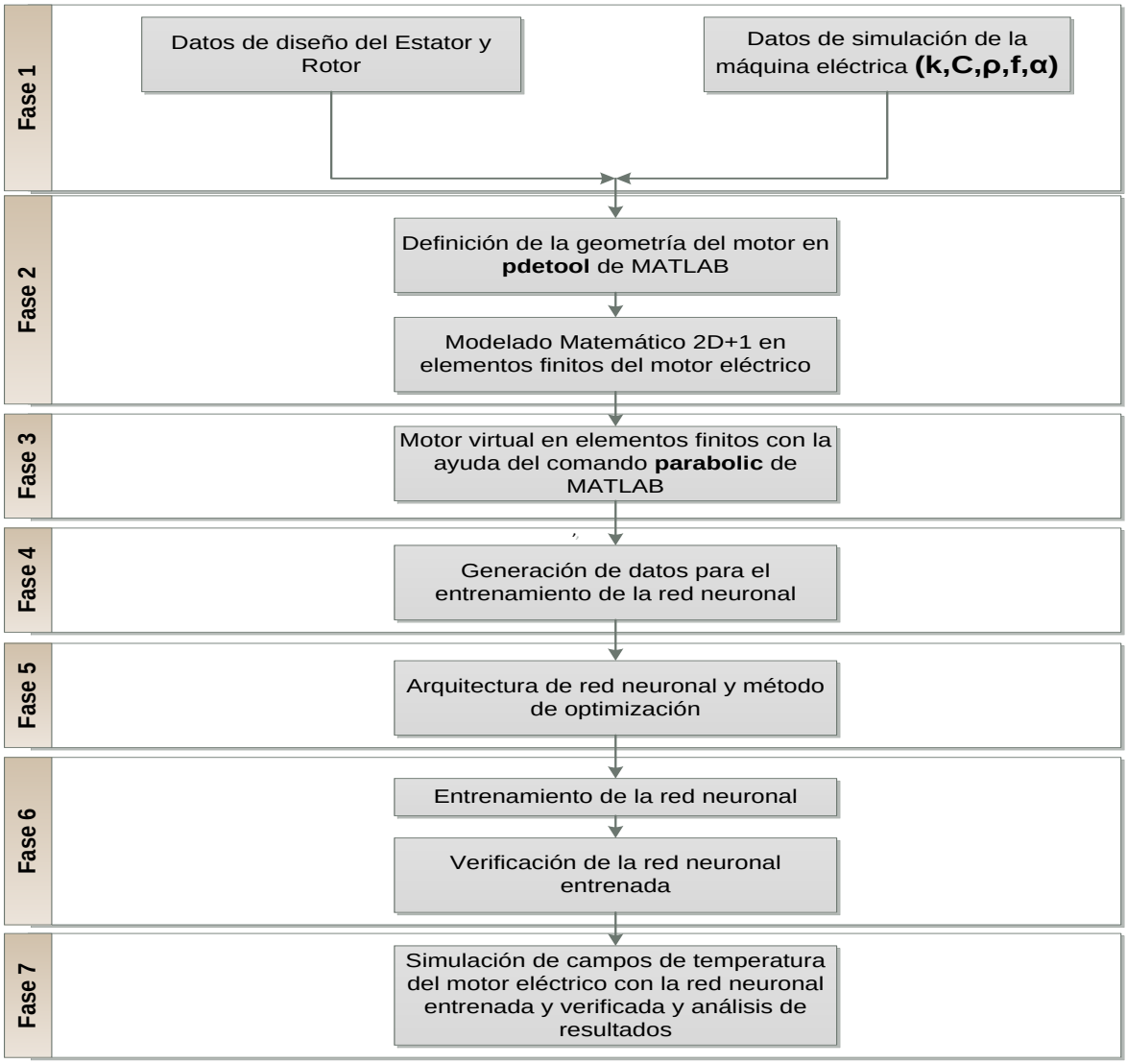


Fig. 2.4 Esquema general de la metodología propuesta en el trabajo de tesis

Fase 1

En principio se tienen que considerar los datos de diseño del motor a estudiar, los cuales corresponden a la geometría real del rotor y estator. También es necesario considerar las propiedades de los materiales que componen los devanados y circuito magnético así como las condiciones de frontera ya que en base a esta información se obtendrán los campos de temperaturas en un intervalo de tiempo definido para el rotor y el estator.

Fase 2

En esta etapa se realiza la simulación en elemento finito, para esto se determinó usar la interfaz grafica PDEtool de MATLAB [54] en donde inicialmente se tiene que generar la geometría del estator y del rotor empleando la información de la fase 1 y posteriormente se realiza el mallado de elementos finitos en el rotor y estator, comenzando la solución para determinar la distribución de temperaturas en el estator y rotor.

Fase 3

En esta fase se desarrolla un motor virtual para simular el comportamiento térmico del motor eléctrico considerando generadores de calor por pérdidas electromagnéticas y diferentes datos del coeficiente de transferencia de calor por convección α**.** Este coeficiente determina el tipo y velocidad del agente enfriador utilizado. Para ello se ejecuta el comando **parabolic** de MATLAB [54] que facilita la solución del modelo en elemento finito. Estos resultados servirán para entrenar la red neuronal.

Fase 4

En esta fase se generan distintos conjuntos de datos variando los parámetros antes mencionados para poder entrenar la red neuronal. A estos se les llama conjuntos de entrenamiento y serán as entradas y salidas de la red neuronal.

Fase 5

Una vez que el motor virtual genera un conjunto de datos apropiados del comportamiento térmico de la máquina en un intervalo de tiempo, se propone una arquitectura de la red neuronal para simular dicho comportamiento con el número de neuronas adecuado.

Así mismo se hace la selección del tipo de entrenamiento (método de optimización) que mejor se comporte en este tipo de problemas con el que se puedan obtener mejores resultados.

Fase 6

Ya que se tiene definida la arquitectura de la red neuronal y la forma de entrenamiento, se efectúan múltiples simulaciones hasta obtener el error de entrenamiento mas bajo posible para los diferentes casos planteados, enseguida se hace la verificación de la red neuronal, proceso que consiste en validar si los resultados de la red neuronal corresponden a los del motor virtual.

Fase 7

Por ultimo, ya con la red neuronal entrenada y verificada, se hacen simulaciones diversas del comportamiento térmico del motor, incluso el modelo puede ser usado para otras geometrías similares reduciendo ampliamente el tiempo de computo y obteniendo resultados bastante precisos.

2.5 Modelo matemático en 2D+1 de transferencia de calor

Como ya se mencionó en el capítulo uno el modelo matemático que se propone en este trabajo de investigación expresa el comportamiento térmico en estado transitorio y/o permanente dentro del espacio continuo representativo del núcleo del estator y/o rotor Ω de la máquina eléctrica rotatoria, que esta gobernada por la siguiente ecuación diferencial parcial de segundo orden de tipo parabólico representada en el espacio 2D+1, lo que

significa que este modelo es en dos dimensiones de espacio y una dimensión de tiempo y tiene la forma siguiente:

$$\rho C \frac{\partial T(x,y,t)}{\partial t} - \nabla \cdot (k \nabla T(x,y,t)) = f(x,y) \qquad \text{en} \quad \Omega \qquad (2.3)$$

Con la condición inicial:

$$T(x,y,0) = T_0(x,y) \qquad \text{para } x, y \in \Omega \qquad (2.4)$$

Y condición de frontera:

$$\vec{n} k \nabla T(x,y,t) + \alpha T(x,y,t) = 0 \qquad \text{en} \quad \partial\Omega \qquad (2.5)$$

Donde:

ρ - Densidad $\left[Kg / m^3 \right]$.

C - Calor especifico $\left[J / Kg°C \right]$.

k - Conductividad térmica $\left[W / m°C \right]$.

$T(x,y,t)$ - Temperaturas $\left[°C \right]$.

$\nabla = \frac{\partial}{\partial x} + \frac{\partial}{\partial y}$ - Operador gradiente.

$\vec{n}$ - Vector normal a la superficie de transferencia de calor por convección.

α - Coeficiente de transferencia de calor por convección $\left[W / m^2 °C \right]$.

$f(x,y)$ - Intensidad de las fuentes internas de calor generado por pérdidas eléctricas y/o magnéticas en devanado y circuito magnético del estator y/o rotor $\left[W / m^3 \right]$.

$\Omega_{Cu} \subset \Omega$ y $\Omega_{Fe} \subset \Omega$ - Dominio continuo de transferencia de calor.

$\partial\Omega$ - Fronteras del dominio continuo Ω.

t - tiempo $\left[seg \right]$.

$(x,y) \in \Omega$.

2.6 Solución en elemento finito del modelo de transferencia de calor

El Método de Elemento Finito es un procedimiento de análisis numérico que se emplea en la solución de una amplia variedad de problemas en la ingeniería cuyos fenómenos físicos estén gobernados por ecuaciones diferenciales parciales, principalmente definidas en ciertos dominios del espacio físico y temporal [18].

Las EDP que describen el caso investigado pueden ser resueltas mediante la aplicación del MEF, esta técnica permite una solución de los campos de temperaturas, aún con campos variables en el tiempo y con materiales heterogéneos y no isotrópicos, además de proporcionar una alternativa que es más adecuada para sistemas de geometría irregular [37].

En contraste con las técnicas por diferencias finitas, el MEF divide el dominio de solución en pequeñas regiones, o “elementos”, por lo que se puede desarrollar una solución aproximada para la EDP de cada uno de los elementos. La solución total se genera enlazando o “ensamblando” soluciones individuales sin descuidar la continuidad en las fronteras entre los elementos.

2.6.1 Suposiciones y Consideraciones

Para efectuar el análisis del proceso de termo-transferencia es necesario hacer las siguientes suposiciones y consideraciones:

- Se consideran las máquinas giratorias en las cuales el rotor cilíndrico está dispuesto dentro del estator.

- Se considera solamente la geometría del estator y rotor.

- El aislamiento eléctrico entre el devanado y el circuito magnético no influye en la distribución de temperaturas dentro del estator o rotor investigado.

- Se investigan solamente incrementos de las temperaturas en relación con la temperatura ambiente.

- Se considera, que el aire en el entrehierro es un aislamiento térmico efectivo entre el estator y rotor.

- Las partes investigadas en la máquina rotatoria están constituidas por el circuito magnético y el devanado con diferente generación de calor en estos materiales.

- En el punto de contacto del circuito magnético y devanado se aplica el principio de continuidad de las temperaturas.

- La transmisión de calor con el ambiente es por convección.

- Los gradientes de temperatura en dirección x, y son mucho mayores que en la dirección z.

- El estator y el rotor están compuestos con materiales de naturaleza isotrópica.

- La transmisión de calor entre el devanado y el circuito magnético es por conducción.

- En el entrehierro, el flujo del agente enfriador (por ejemplo flujo del aire del ventilador) enfría estator y rotor al mismo tiempo, por eso se considera que:

$$\alpha_{1rotor} = \alpha_{1estator} \tag{2.6}$$

2.6.2 Solución en elementos finitos del modelo de transferencia de calor utilizando simulación digital

Para resolver las EDP del modelo de transferencia de calor 2D+1 aplicando el método de elemento finito se utilizó la simulación digital con el software MATLAB mediante la librería PDEtool con la cual fue posible representar los campos de temperaturas generadas por pérdidas eléctricas y/o magnéticas en el núcleo del estator y rotor de máquinas eléctricas rotatorias.

Tomando en consideración las suposiciones anteriores, se utilizo la GUI de la librería PDEtool para simular la distribución de temperaturas en cada nodo del estator y del rotor de la máquina eléctrica en un intervalo de tiempo determinado.

En primer lugar se definió la geometría del estator y rotor del motor. Considerando la estructura uniforme y que hay simetría tanto en el estator como en el rotor, fue posible limitar el área de estudio a una pequeña parte en ambos casos sin afectar los resultados de las simulaciones. Por lo tanto, se generaron pequeños cortes axiales del estator y del rotor como lo muestran las Figs. 2.5 y 2.6 que se presentan a continuación:

Es importante considerar que la geometría debe realizarse de manera muy exacta respecto al motor que se prentende estudiar, debido a que la red neuronal es una herramienta que no conoce de la fisica de los procesos, unicamente aproxima los valores que se le dan en el entrenamiento, por lo tanto si la geometría no es exacta, la red neuronal arrojara resultados de acuerdo al conjunto de entrenamiento que se proporciono.

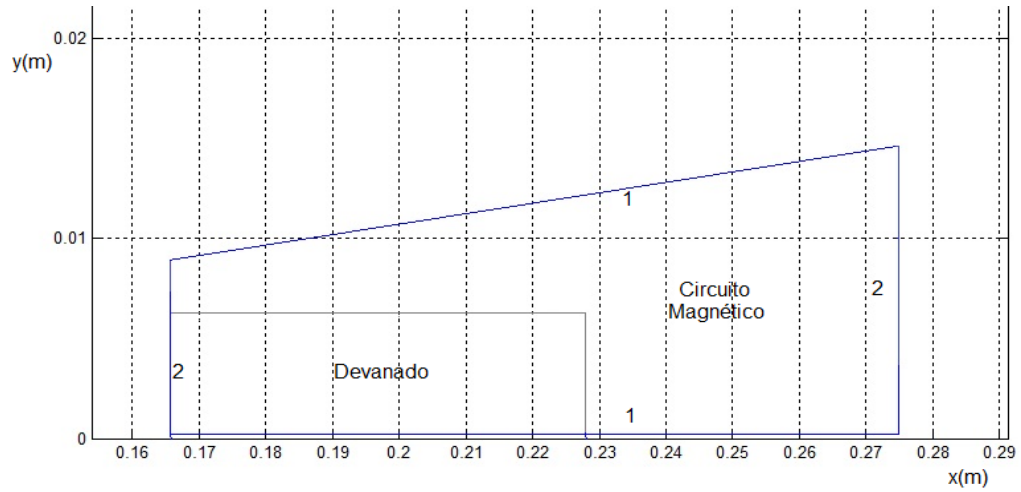


Fig. 2.5 Segmento de corte del estator investigado.

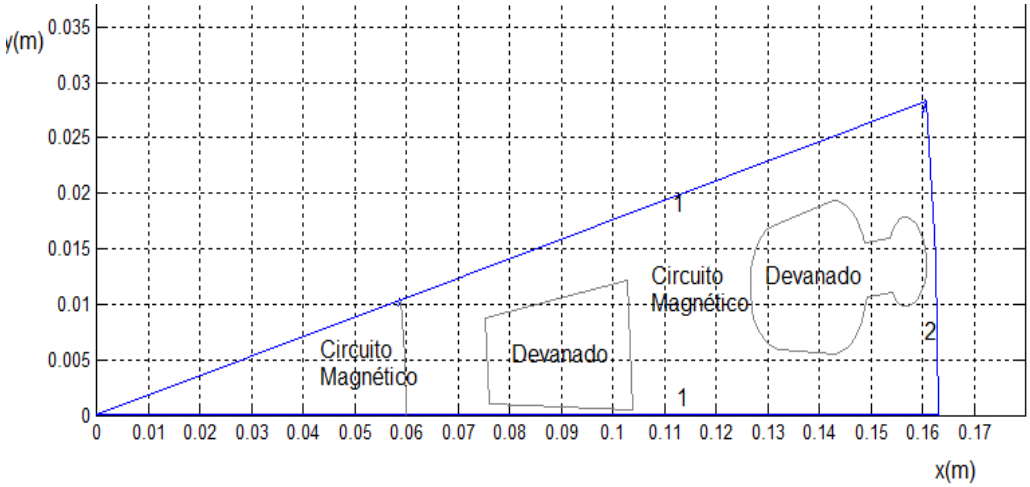


Fig. 2.6 Segmento de corte del rotor investigado.

La generación del mallado implica la division del segmento axial del estator y rotor del motor en un conjunto de elementos triangulares. El mallado que genera la interfaz gráfica a los segmentos de corte es el que muestra en las Figs. 2.7 y 2.8 compuesta por 72 nodos con 101 elementos para el estator y de 864 nodos con 1637 elementos para el rotor.

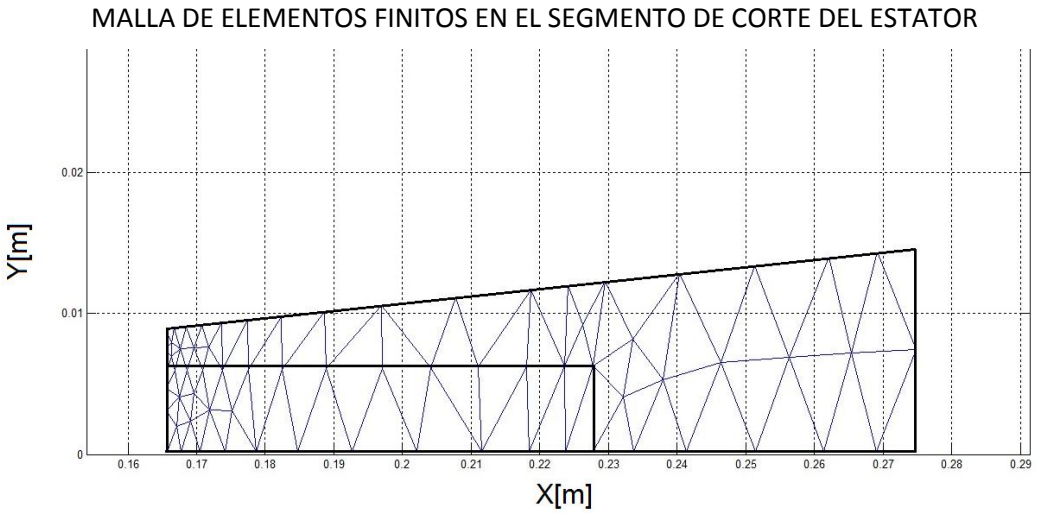


Fig. 2.7 Mallado en segmento de estator.

Es evidente que la geometría del rotor es más detallada que la del estator por considerar las partes curvas de los devanados, por lo que el número de nodos es mucho mayor en el rotor.

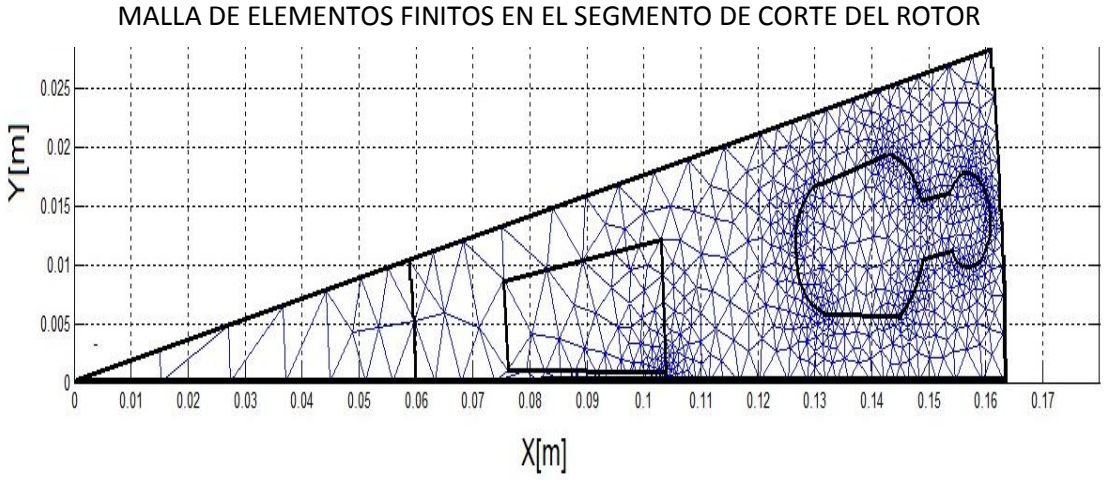


Fig. 2.8 Mallado en segmento de rotor.

En la solución de problemas más específicos se requiere hacer un mallado mucho más detallado para que la respuesta sea mas exacta aunque ello implica que converja

lentamente, ello implica que el número de elementos aumenta considerablemente y en consecuencia el tiempo de cómputo es mucho mayor.

Para darle solución al modelo es necerio especificar las propiedades de los materiales que forman parte de las piezas del rotor y el estator como lo son el cobre y el acero. Estos datos los tomamos de [37] y se muestran en la tabla siguiente:

Tabla 2.2 Propiedades de los materiales del motor electrico analizado.

Propiedad	Unidad	Valor	
k	$W/m°C$	Cu	386
		Fe	45
C	$J/Kg°C$	Cu	385.4
		Fe	480
ρ	Kg/m^3	Cu	8890
		Fe	7880

Donde:

k es la conductividad térmica

C es el calor específico

ρ es la densidad

Cabe destacar que las condiciones de frontera se dividen en dos tipos que se muestran en las Figs. 2.5 y 2.6 que serán 1 y 2.

Las condiciones tipo 1 serán cero debido a que separan partes simétricas y las tipo 2 serán tipo Neumann [11] y están representadas por medio de la ecuación 2.5 ya que describen la transferencia de calor por convección.

En el caso de los coeficientes de calor por convección para el lado del entrehierro y la carcasa se toman de un rango de valores que se encuentran en la tabla del apendice F.

Las fuentes internas de calor en devanados y circuitos magnéticos debidos a las pérdidas eléctricas y magnéticas en el estator y rotor que se usaron para la simulación que se presenta en las Figs. 2.9 y 2.10 se muestran en la siguiente tabla:

Tabla 2.3 Valores de las fuentes internas de calor considerados.

Fuente de Calor	Unidad	Valores	
$f(x,y)$	W/m^3	Cu	$0.75x10^6$
		Fe	$0.1x10^6$

Se eligieron estos valores para las fuentes internas de calor debido a que estan en el rango de valores que normalmente se utilizan en este tipo de trabajos.

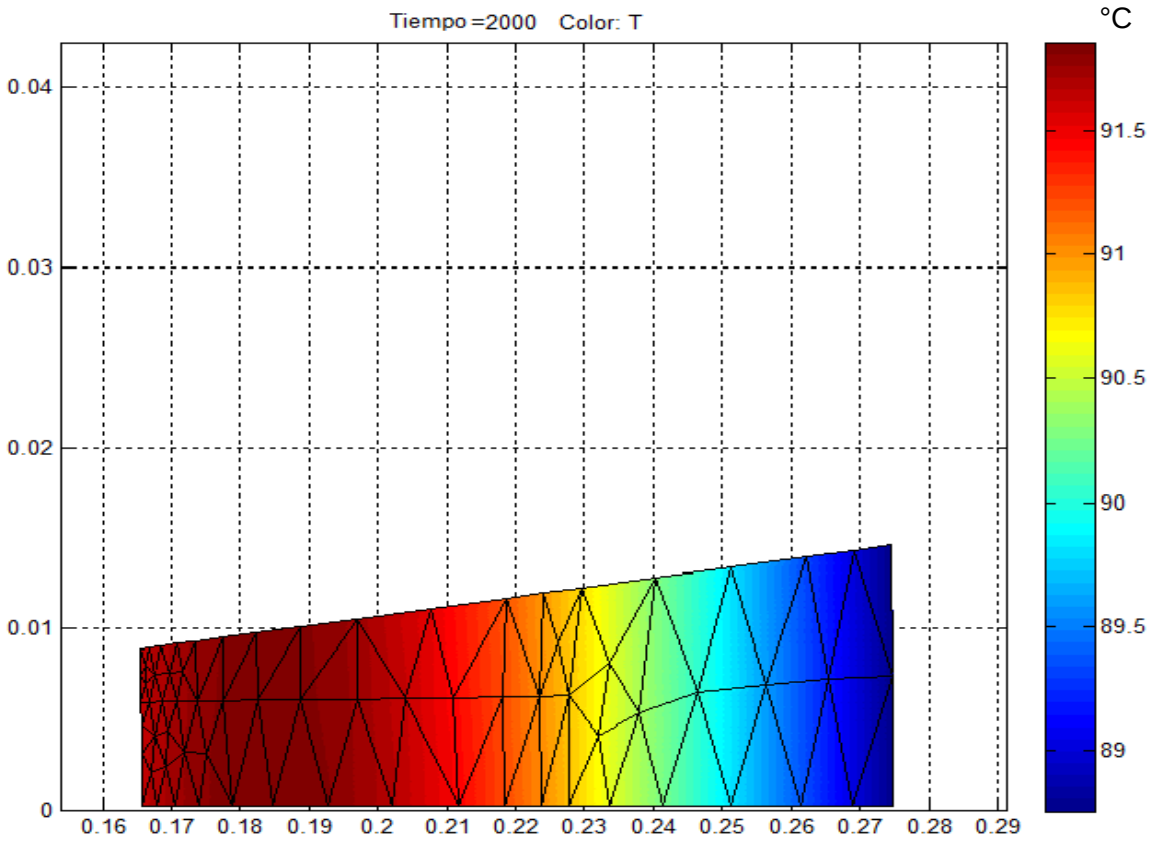


Fig. 2.9 Resultado de la simulacion en el segmento de corte del estator.

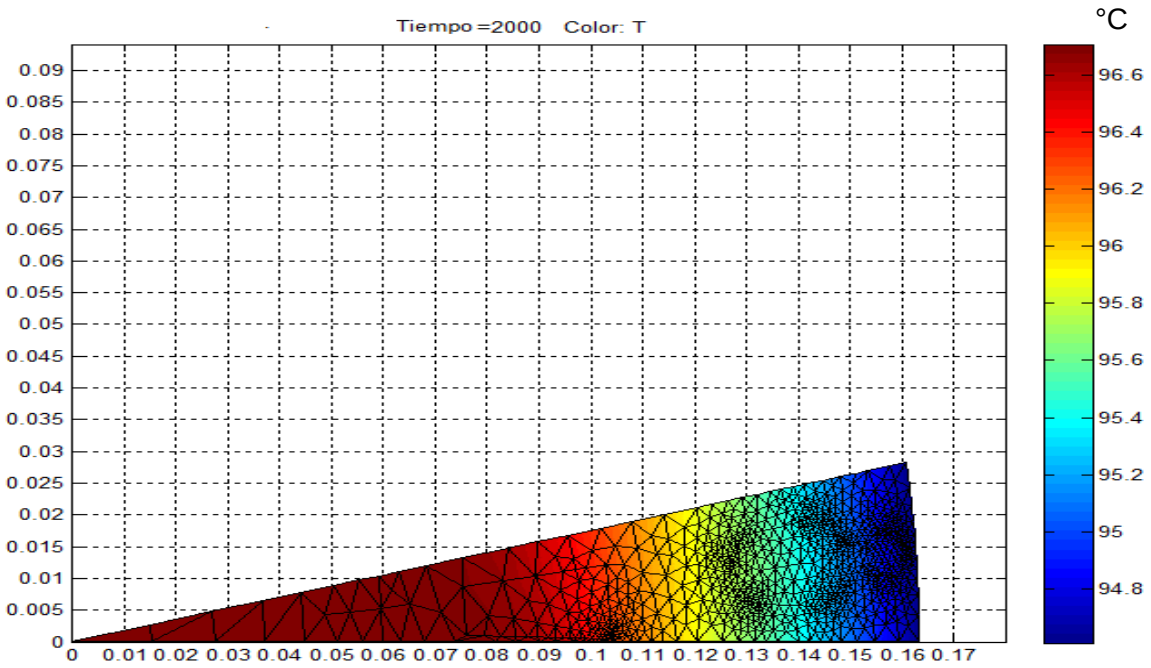


Fig. 2.10 Resultado de la simulacion en el segmento de corte del rotor.

De esta manera es posible simular el comportamiento térmico en el motor en un rango de tiempo; sin embargo, para lograr el objetivo del trabajo es necesario hacer el procedimiento anterior cientos de veces para tener un conjunto de entrenamiento para la red neuronal, puesto que se prentende obtener valores con distintas combinaciones de los valores de fuentes de calor, coeficientes de transferencia de calor por convección y tiempos, por lo que es necesario generar un motor virtual que nos permita tener los valores de tempreraturas del motor de forma rápida.

2.7 Motor virtual para la determinación de temperaturas generadas en el núcleo del estator y rotor de la maquina eléctrica rotatoria

Considerando la simulación anterior mediante el MEF y con el apoyo del Toolbox PDEtool de MATLAB es posible obtener una serie de matrices y vectores relacionados con la geometría del motor. De esta manera se puede representar el modelo matemático 2D+1 de transferencia de calor mediante un motor virtual en elemento finito para determinar

campos de temperaturas generadas por pérdidas eléctricas y/o magnéticas en máquinas eléctricas rotatorias.

Se propone un motor virtual en elemento finito aprovechando el comando **Parabolic** de MATLAB que tiene la finalidad de recabar, procesar y almacenar la información generada del modelo matemático presentado en 2.5 representando las temperaturas del motor en los instantes de tiempo t.

$$U_1 = parabolic(U_0, tiempo, b, p, e, t, c, a, f, d) \tag{2.7}$$

Donde:

U_1 - Es la matriz de temperaturas en todos los nodos de las mallas del segmento de corte del estator o rotor determinado para cada instante de tiempo.

U_0 - Es el vector de temperaturas iníciales.

b - Es la matriz que introduce los valores de coeficientes de transferencia de calor convectiva (α) de las condiciones de frontera establecidas.

p, e, t - Representan las matrices de los parámetros de la malla generada por el MEF.

c, a, d - Representan los vectores de las propiedades de los materiales involucrados además de los parámetros para la EDP.

f - Es el vector que describe la generación de calor, en el devanado y circuito magnético generado por pérdidas eléctricas y magnéticas del estator o rotor.

En la Fig. 2.11 se muestra el diagrama se presenta la metodología que sigue el motor virtual para generar campos de temperaturas con los datos que previamente se obtuvieron con la utilización de la librería PDEtool.

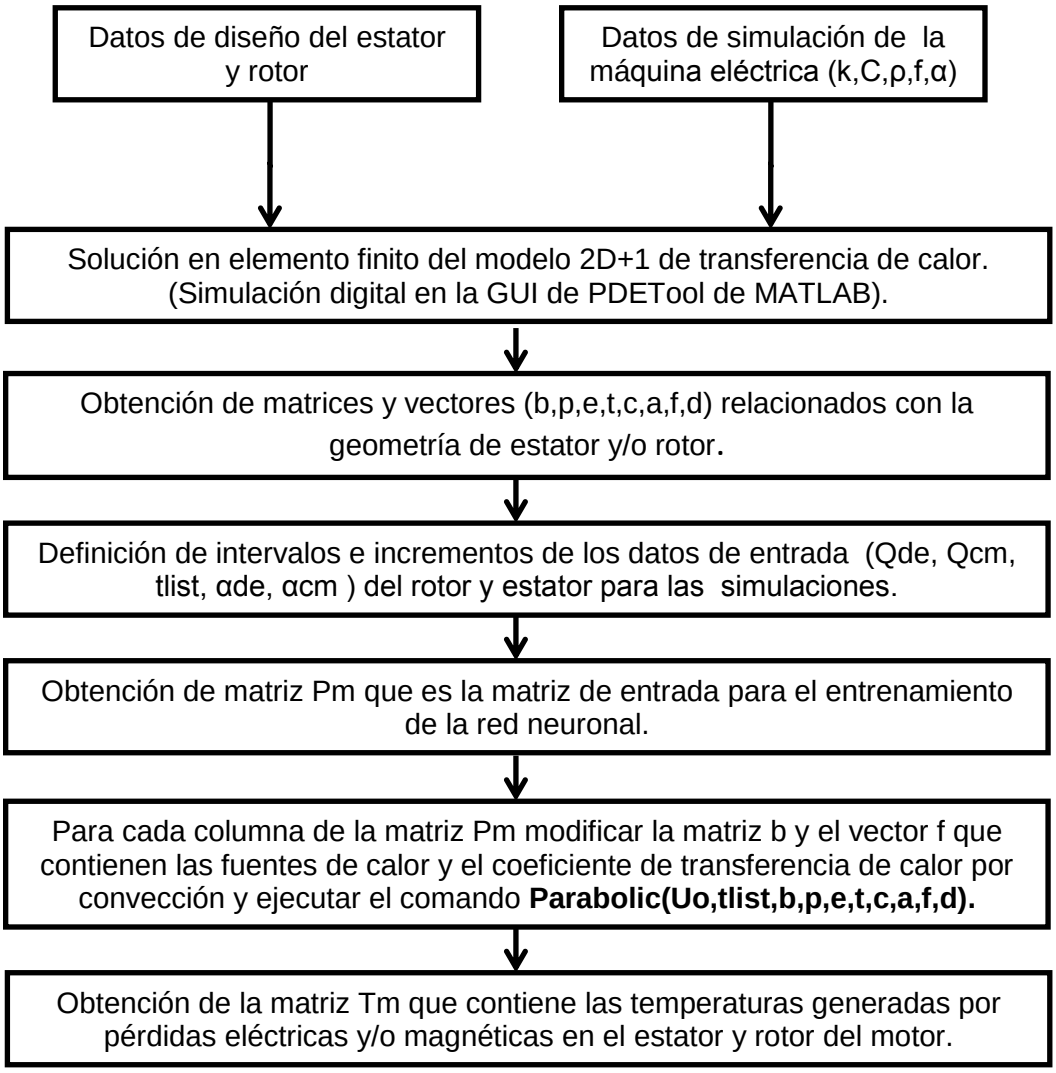


Fig. 2.11 Esquema a bloques del motor virtual en elemento finito

El programa desarrollado para el diagrama mostrado en la Fig. 2.11 se muestra en el apéndice D.

CAPÍTULO 3

RED NEURONAL PARA LA SIMULACIÓN DE PROCESOS DE TRANSFERENCIA DE CALOR GENERADOS POR PÉRDIDAS ELECTROMAGNÉTICAS

3.1 Introducción

En este capítulo, se plantea la metodología para generar una red neuronal que permite simular el comportamiento térmico de un motor de inducción doble jaula de ardilla, de la misma manera que se hace mediante el uso del MEF, pero de forma mas rápida y asegurando mayor facilidad para los usuarios.

Los conceptos generales de redes neuronales son muy conocidos ya en la literatura por lo que se muestran en el apéndice A. En este capitulo se presenta la aplicación de las redes neuronales al problema de transferencia de calor en máquinas eléctricas en el modelo propuesto, para simular los campos de temperatura de la máquina mostrada en el punto 2.3.

3.2 Generalidades de las redes neuronales artificiales

Existen máquinas de cómputo capaces de realizar 100 millones de operaciones por segundo, pero no son capaces de entender el significado de las formas visuales o de distinguir entre distintas clases de objetos. Los sistemas de computación secuencial son exitosos en la resolución de problemas matemáticos o científicos, en la creación, manipulación y mantenimiento de bases de datos, en comunicaciones electrónicas, en el procesamiento de textos, gráficos y auto edición, etc., pero definitivamente tienen una gran incapacidad para interpretar el mundo [45].

Esta dificultad de los sistemas de cómputo que trabajan bajo la filosofía de los sistemas secuenciales, ha hecho que un gran número de investigadores centre su atención en el desarrollo de nuevos sistemas de tratamiento de la información, que permitan solucionar

problemas cotidianos, tal como lo hace el cerebro humano; este órgano biológico cuenta con varias características deseables para cualquier sistema de procesamiento digital, tales como:

1. Es robusto y tolerante a fallas, diariamente mueren neuronas sin afectar su desempeño.
2. Es flexible, se ajusta a nuevos ambientes por aprendizaje, no hay que programarlo.
3. Puede manejar información difusa, con ruido o inconsistente.
4. Es altamente paralelo.
5. Es pequeño, compacto y consume poca energía.

El cerebro humano constituye una computadora muy notable, es capaz de interpretar información imprecisa suministrada por los sentidos a un ritmo increíblemente veloz. Logra discernir un susurro en una sala ruidosa, un rostro en un callejón mal iluminado y leer entre líneas un discurso; lo más impresionante de todo es que el cerebro aprende sin instrucciones explícitas de ninguna clase a crear las representaciones internas que hacen posibles estas habilidades [45].

Basados en la eficiencia de los procesos llevados a cabo por el cerebro, e inspirados en su funcionamiento, varios investigadores han desarrollado desde hace más de 60 años la teoría de las Redes Neuronales Artificiales (RNA), las cuales emulan las redes neuronales biológicas, y se han utilizado para aprender estrategias de solución basadas en ejemplos de comportamiento típico de patrones; estos sistemas no necesitan programarse, ellos generalizan y aprenden de la experiencia [45].

Los sistemas de cómputo tradicional procesan la información en forma secuencial; un computador serial consiste por lo general de un solo procesador que puede manipular instrucciones y datos que se localizan en la memoria. El procesador lee, y ejecuta una a una las instrucciones en la memoria; este sistema serial es secuencial pues todo sucede en una sola secuencia determinística de operaciones. Las RNA no ejecutan instrucciones, responden en paralelo a las entradas que se les presentan. El resultado no se almacena en una posición de memoria, este es el estado de la red para el cual se logra equilibrio. El

conocimiento de una red neuronal no se almacena en instrucciones, el poder de la red está en su topología y en los valores de las conexiones (pesos) entre neuronas [36].

Las RNA son una teoría que aún esta en proceso de desarrollo, su verdadera potencialidad no se ha alcanzado todavía; aunque los investigadores han desarrollado potentes algoritmos de aprendizaje de gran valor práctico, las representaciones y procedimientos de que se sirve el cerebro, son aún desconocidas. Tarde o temprano los estudios computacionales del aprendizaje con RNA acabarán por converger a los métodos descubiertos por evolución, cuando eso suceda, un gran número de datos empíricos concernientes al cerebro comenzarán súbitamente a adquirir sentido y se tornarán factibles muchas aplicaciones desconocidas de las redes neuronales [46].

3.3 Principales aplicaciones de las redes neuronales artificiales

- **Automóviles:** Sistemas de piloto automático, detección de fallas por reconocimiento exterior de vibraciones.
- **Bancos:** Lectura de cheques y otros documentos, evaluación de aplicaciones de créditos.
- **Electrónica:** Predicción de secuencia de códigos, distribución de elementos en CI, control de procesos, análisis de fallas, visión artificial, reconocimiento de voz.
- **Finanzas:** Transacción real de los bienes, asesoría de los prestamos, previsión en la evolución de precios, seguimiento de hipotecas, análisis de uso en línea de crédito, evaluación del riesgo en créditos, identificación de falsificaciones, Interpretación y reconocimiento de firmas.
- **Manufactura:** Control de la producción y de procesos, análisis y diseño de productos, diagnostico de fallas en el proceso y la maquinaria, identificación de partículas en tiempo real, inspección de calidad mediante sistemas visuales, análisis de mantenimiento de máquinas.
- **Medicina:** Análisis de células portadoras de cáncer mamario, análisis de electroencefalograma y de electrocardiograma, reconocimiento de infartos mediante electrocardiogramas. Diseño de prótesis, optimización de tiempos de trasplante, reducción de gastos hospitalarios.

- **Robótica:** Control dinámico de trayectoria, robots elevadores, controladores, sistemas ópticos.
- **Seguridad:** Códigos de seguridad adaptativos, criptografía, reconocimiento de huellas digitales, control de accesos.
- **Telecomunicaciones:** Compresión de datos e imágenes, automatización de servicios de información, traslación en tiempo real de lenguaje hablado.
- **Transporte:** Diagnostico de frenos en camiones, sistemas de ruteo y seguimiento de flotas.
- **Voz:** Reconocimiento de voz, compresión de voz, clasificación de vocales, transformación de texto escrito a voz.

3.4 Estructura de red neuronal propuesta

Lo anterior sirve para conocer la herramienta tan poderosa que resultan ser las Redes Neuronales Artificiales en la solución de problemas diversos, como se pudo notar, se pueden aplicar a diversas áreas de la investigación, no obstante su principal aplicación hasta ahora es el reconocimiento de patrones. En esta investigación se aplican las redes neuronales para aproximar una función, es decir, ajustar las curvas del comportamiento de la temperatura en el rotor y estator de un motor eléctrico en un intervalo de tiempo definido previamente.

Para ello se necesita conocer los datos de entrada y salida que se generan con en el motor virtual del capítulo 2 y que se emplean como conjunto de entrenamiento.

La Fig. 3.1 muestra el esquema de la red neuronal propuesta con las entradas y salidas que se consideraron para esta investigación.

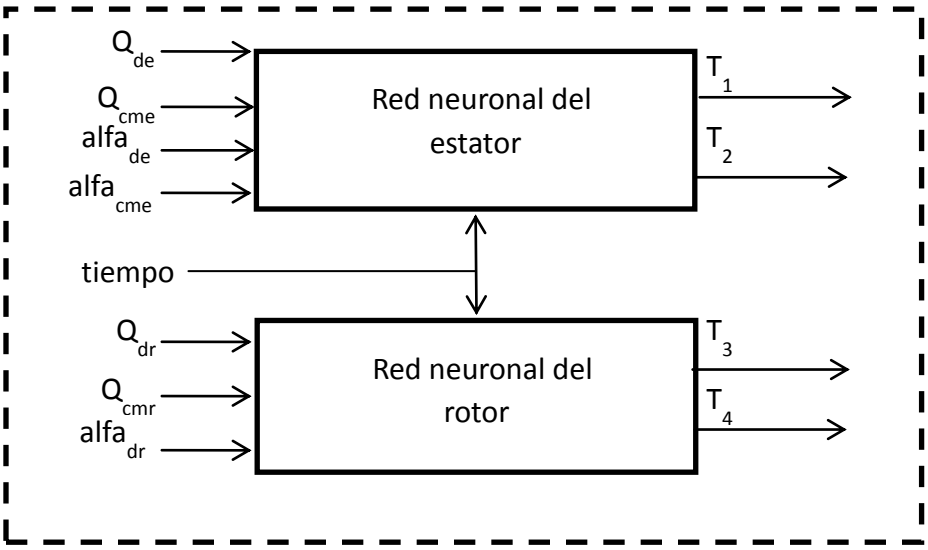


Fig.3.1 Diagrama a bloques de la red neuronal propuesta

Dónde:

Las entradas:

Q_{de} - pérdidas eléctricas en el devanado del estator.

Q_{cme} - pérdidas magnéticas en el circuito magnético del estator.

$alfa_{de}$ - coeficiente de transferencia de calor por convección por el lado del devanado del estator.

$alfa_{cme}$- coeficiente de transferencia de calor por convección por el lado del circuito magnético del estator.

Q_{dr} - pérdidas eléctricas en el devanado del rotor.

Q_{cmr} - pérdidas magnéticas en el circuito magnético del rotor.

$alfa_{dr}$ - coeficiente de transferencia de calor por convección por el lado del devanado del rotor.

t – tiempo.

Las salidas:

T_1 – Temperatura generada en un nodo del estator
T_2 – Temperatura generada en otro nodo del estator
T_3 – Temperatura generada en un nodo del rotor
T_4 – Temperatura generada en otro nodo del rotor

3.5 Selección de los nodos de medición de temperaturas

La elección de un nodo en concreto en donde lo que interesa son los cambios de temperatura se puede interpretar como la colocación de un sensor virtual dentro del motor virtual en un lugar preciso. Para el caso 1 se eligieron dos nodos para el rotor y dos nodos para el estator de la máquina tomando en cuenta lugares donde físicamente sería más fácil colocar los sensores de temperatura y considerando una buena exactitud en la simulación.

En general se pueden simular las temperaturas en cualquiera de los puntos del motor de acuerdo a las necesidades, pueden también determinarse donde están las temperaturas máximas en el estator y/o rotor.

En el caso 2 se eligieron dos nodos para el rotor y dos nodos para el estator de la máquina en puntos intermedios con la finalidad de demostrar que es posible determinar la temperatura en cualquier punto del motor.

A continuación se presentan los nodos que fueron seleccionados para este estudio de acuerdo a la referencia [39]. La Fig. 3.2 muestra los nodos del estator para el primer caso, en la Fig. 3.3 se presentan los nodos del rotor para el primer caso, la Fig. 3.4 expone los nodos del estator para el segundo caso y la Fig. 3.5 muestra los nodos del rotor para el segundo caso.

En el primer caso del estator se eligió el nodo 9 para tener una medición en el lado de la carcasa y el nodo 10 para tener una medición en el lado del entrehierro.

En el primer caso del rotor se eligieron los nodos 168 y 177 del lado del entrehierro debido a que por su ubicación sería más probable poner sensores en esa parte debido a que en la parte del núcleo sería imposible hacer mediciones físicas.

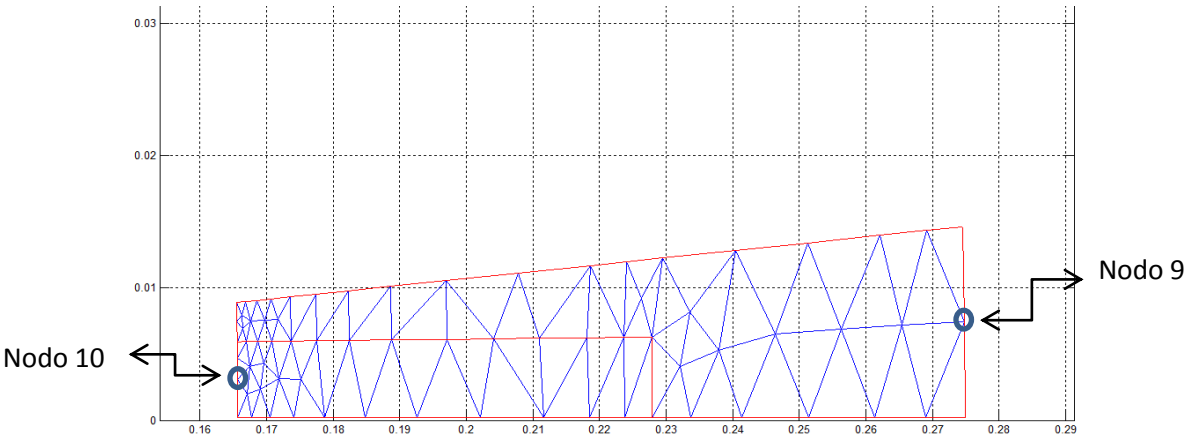


Fig.3.2 Nodos seleccionados del estator en el caso 1

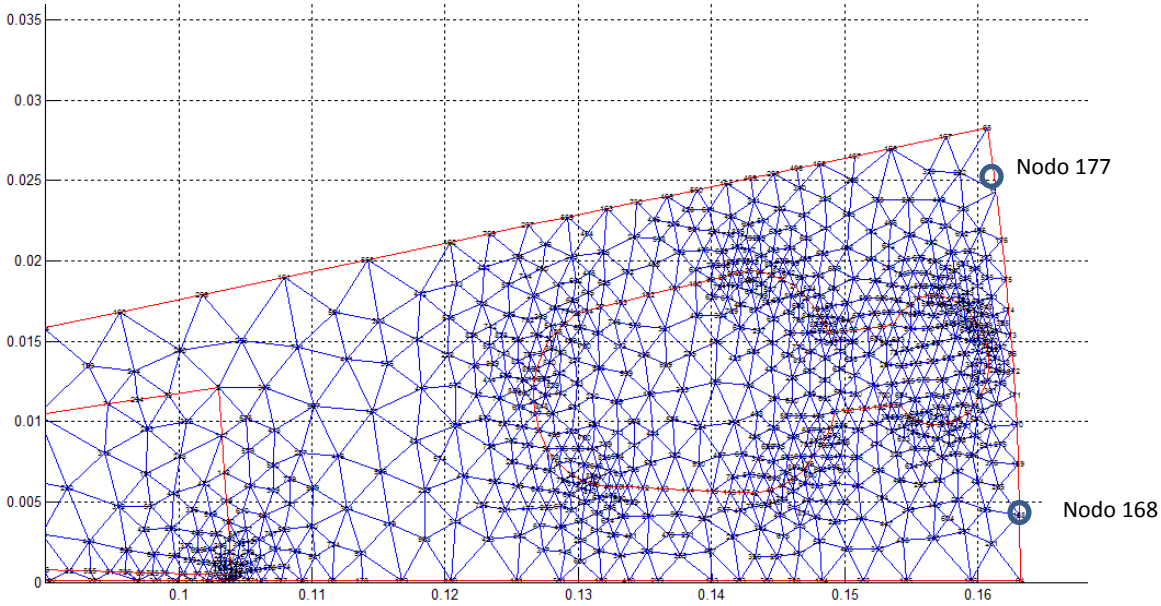


Fig.3.3 Nodos seleccionados del rotor en el caso 1

En el segundo caso del estator se eligieron los nodos 20 y 55 de manera heurística para tener mediciones en el interior del estator.

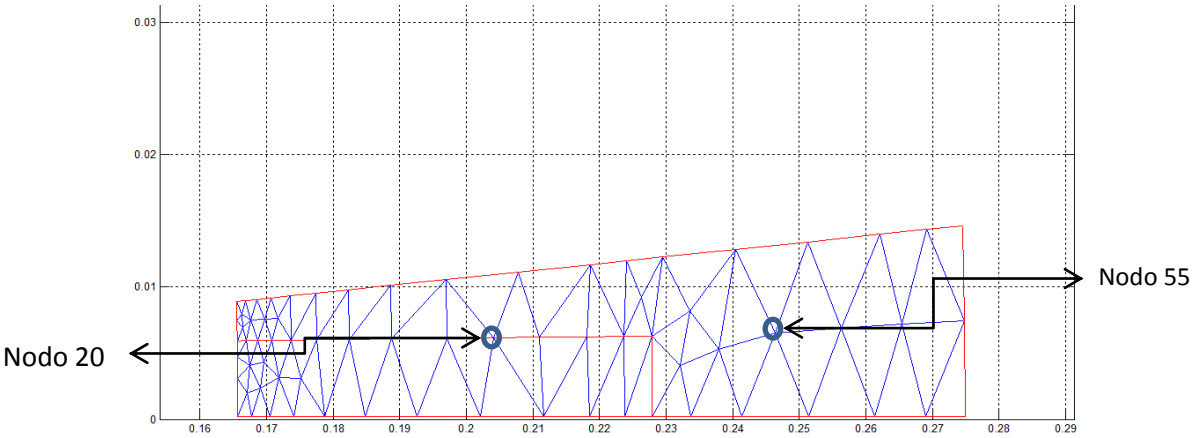


Fig.3.4 Nodos seleccionados del estator en el caso 2

En el segundo caso del rotor se eligieron los nodos 346 y 485 con la finalidad de conocer las temperaturas en puntos internos.

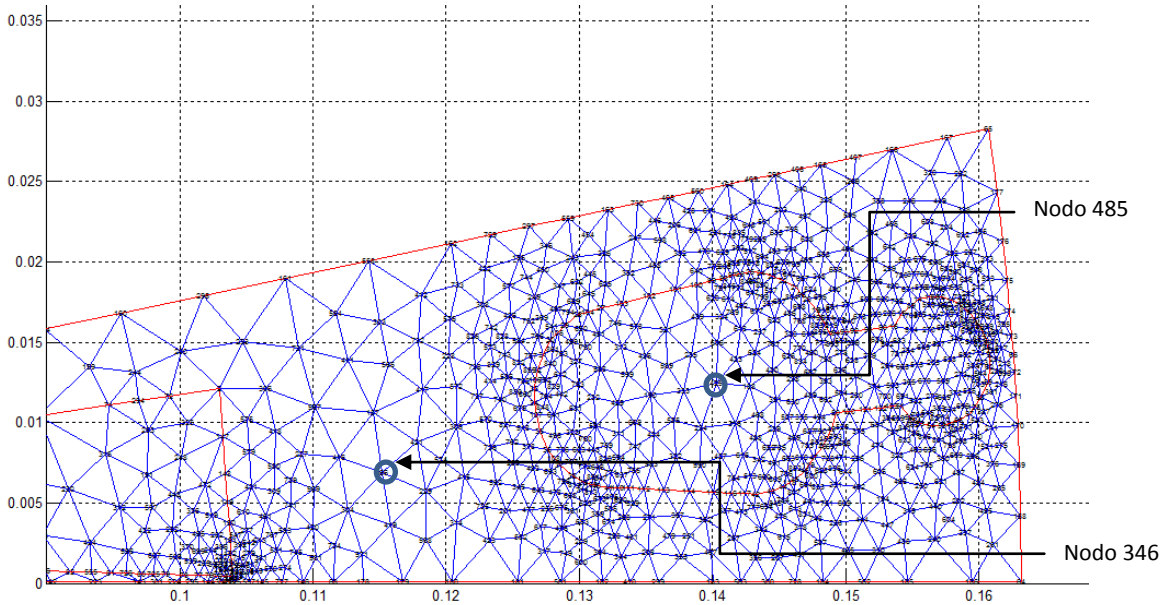


Fig.3.5 Nodos seleccionados del rotor en el caso 2

3.6 Algoritmo de entrenamiento de la red neuronal propuesta

Como se puede ver en el apéndice A, se denomina aprendizaje al proceso de configuración de la red para que las entradas produzcan las salidas deseadas a través del fortalecimiento de las conexiones. Para ello existen diversos algoritmos que se pueden emplear dependiendo las necesidades del problema a resolver. (Ver Apéndice H)

En esta investigación se recurrió al algoritmo de retropropagación del error debido a que por el tamaño de los conjuntos de entrenamiento fue necesario emplear una red multicapa. A continuación se describe el algoritmo de forma detallada para su mejor comprensión.

3.6.1 El algoritmo de retropropagación

El nombre de retropropagación resulta de la forma en que el error es propagado hacia atrás a través de la red neuronal, en otras palabras el error se propaga hacia atrás desde la capa de salida. Esto permite que los pesos sobre las conexiones de las neuronas ubicadas en las capas ocultas cambien durante el entrenamiento.

El algoritmo de retropropagación se da al hacer una generalización a la regla delta que se muestra a continuación.

3.6.1.1 Regla Delta

La regla de aprendizaje delta sólo es válida para las funciones de activación continua y en el modo de entrenamiento supervisado, tal es el caso de este trabajo de investigación. La señal de aprendizaje de esta regla se denomina delta y se define como sigue:

$$r = \left[d_i - f(w_i^t x) \right] f'(w_i^t x) \tag{3.1}$$

El termino $f'(w_i^t x)$ es la derivada de la función de activación $f(net)$ calculada para $net = w_i^t x$. La regla delta se explica en la Fig. 3.6. Esta regla de aprendizaje puede ser fácilmente derivada de la condición del error cuadrático entre o_i y d_i. Calculando el vector gradiente de con respecto a w_i del cuadrado del error se define como:

$$E = \frac{1}{2}(d_i - o_i)^2 \tag{3.2}$$

El cual es equivalente a:

$$E = \frac{1}{2}\left[d_i - f(w_i^t x)\right]^2 \tag{3.3}$$

De esta forma se encuentra el valor del vector gradiente del error:

$$\nabla E = -(d_i - o_i) f'(w_i^t x) x \tag{3.4}$$

Las componentes del vector gradiente son:

$$\frac{\partial E}{\partial w_{ij}} = -(d_i - o_i) f'(w_i^t x) x_j \tag{3.5}$$

Puesto que la minimización del error requiere un cambio de peso para estar en la dirección del gradiente negativo, se toma:

$$\Delta w_i = -\eta \nabla E \tag{3.6}$$

Donde η es una constante positiva. De las ecuaciones anteriores se obtiene:

$$\Delta w_i = \eta (d_i - o_i) f'(net) x \tag{3.7}$$

O por el simple ajuste de pesos se convierte en:

$$\Delta w_{ij} = \eta(d_i - o_i) f'(net) x_j \tag{3.8}$$

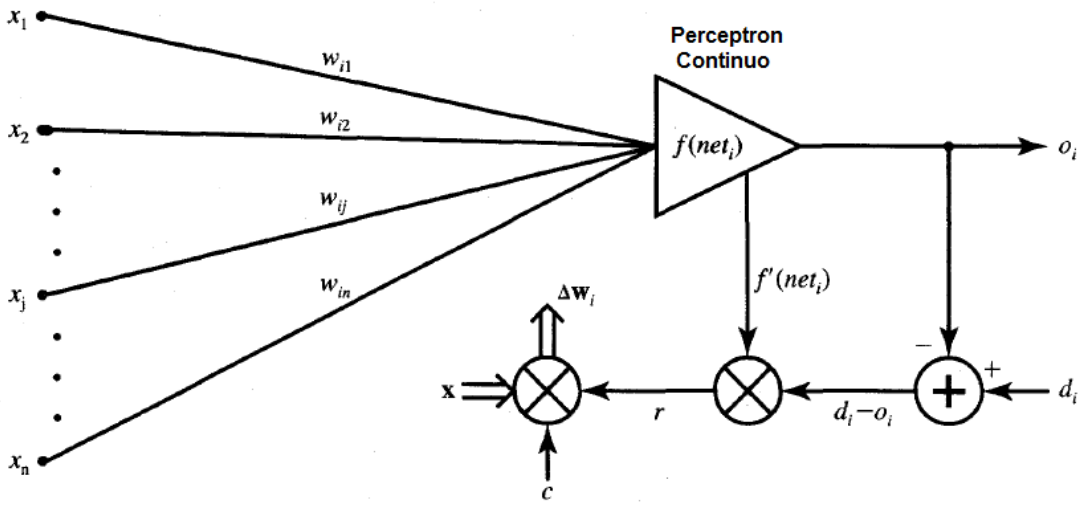


Fig.3.6 Regla de aprendizaje delta

Se puede ver que el ajuste de pesos en las formulas 3.7 y 3.8 se calculan en base a la minimización del error cuadrático.

Considerando el uso de la regla general de aprendizaje (Ver apéndice A), el ajuste de pesos se convierte en:

$$\Delta w_i = c(d_i - o_i) f'(net) x \tag{3.9}$$

Por lo tanto, se puede ver que (3.9) es idéntica a (3.7), ya que c y η se han asumido para ser constantes arbitrarias. Los pesos se inicializan a los valores de este método de entrenamiento.

La regla delta fue introducida hace poco tiempo para el entrenamiento de redes

neuronales (McClelland y Rumelhart, 1986) [47]. Esta regla es paralela a la regla de aprendizaje del perceptrón discreto. También se le conoce como la regla de aprendizaje del perceptrón continuo. La regla de aprendizaje delta puede ser generalizada para redes multicapa como es el caso de este trabajo de investigación, por lo tanto se hace una breve descripción de la generalización de la regla delta que se describe detalladamente en el apéndice B.

3.6.1.2 Algoritmo de retropropagación resumido

El diagrama de flujo de la Fig. 3.7 se puede explicar de la siguiente manera:

Para el conjunto de pares de entrenamiento P:

$$\{z_1, d_1, z_2, d_2, ..., z_p, d_p\} \tag{3.10}$$

Donde z_i es $(1x1)$, d_i es $(Kx1)$, $i = 1, 2..., P$ Tomando en cuenta que la primer componente de cada z_i es de valor -1 los vectores de entrada que han sido argumentados. El tamaño $J-1$es seleccionado para las capas ocultas con salidas y. Tomando en cuenta también que la J -ésima componente de y es de valor -1, desde que las salidas de la capa oculta han sido argumentadas; y es $(Jx1)$ y o es $(Kx1)$.

Paso 1: Se seleccionan $\eta > 0$, $E_{\max}$

Los pesos W y V se inicializan con pequeños valores aleatorios; W es (KxJ), V (VxI).

$$q \leftarrow 1, p \leftarrow 1, E \leftarrow 0 \tag{3.11}$$

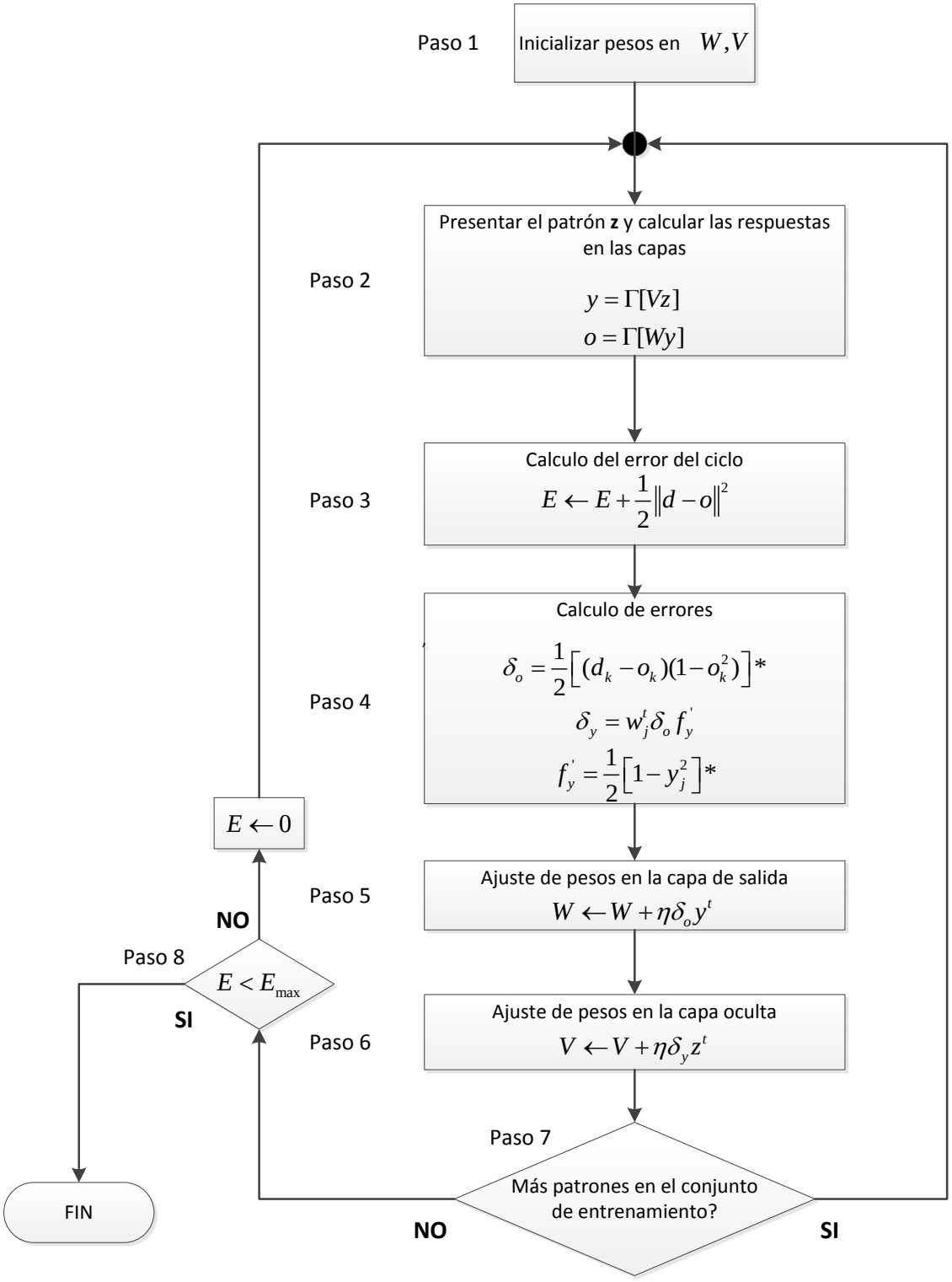


Fig. 3.7 Diagrama de flujo del algoritmo de retropropagación con 2 capas

Paso 2: La etapa de entrenamiento comienza aquí (Ver nota 1)

Se presentan las entradas y se calculan las salidas de las capas:

$$z \leftarrow z_p, d \leftarrow d_p \tag{3.12}$$

$$y_j \leftarrow f(v_j^t z) \text{ Para } j = 1, 2..., J \tag{3.13}$$

Donde v_j, un vector columna, es la j'esima fila de V, y

$$o_k \leftarrow f(w_k^t y) \text{ Para } k = 1, 2..., K \tag{3.14}$$

Donde w_k, un vector columna, es la k'esima fila de W.

Paso 3: El valor del error es calculado:

$$E \leftarrow \frac{1}{2}(d_k - o_k) + E \text{ Para } k = 1, 2..., K \tag{3.15}$$

Paso 4: El error de los vectores δ_o y δ_y de ambas capas es calculado.

El vector δ_o es $(Kx1)$ y el vector δ_y es $(Jx1)$.

La señal del error en términos de la capa oculta en este paso es:

$$\delta_{yj} = \frac{1}{2}(1 - y_j^2)\sum_{k=1}^{K} \delta_{ok} w_{kj} \text{ Para } j = 1, 2..., J \tag{3.16}$$

Paso 5: Se ajustan los pesos de la capa de salida:

$$w_{kj} \leftarrow w_{kj} + \eta\delta_{ok} y_j \text{ Para } k = 1, 2..., K \text{ y para } j = 1, 2..., J \quad (3.17)$$

Paso 6: Se ajustan los pesos de la capa oculta:

$$v_{ji} \leftarrow v_{ji} + \eta\delta_{yj} z_i \text{ Para } j = 1, 2..., J \text{ y para } i = 1, 2..., I \quad (3.18)$$

Paso 7: SI $p < P$ entonces $p \leftarrow p+1, q \leftarrow q+1$, y regresa al paso 2, de lo contrario ir al paso 8.

Paso 8: El ciclo de entrenamiento se ha completado.

Para $E < E_{\max}$ termina la sesión de entrenamiento. Se obtienen los valores finales de W, V, q y E.

Si $E > E_{\max}$, entonces $E \leftarrow 0$, $p \leftarrow 1$, y se inicia el nuevo ciclo de entrenamiento a partir del paso 2.

Nota 1: Para mejores resultados, los patrones pueden ser elegidos aleatoriamente del conjunto de entrenamiento.

En el caso de este trabajo de investigación el procedimiento para el entrenamiento consiste en obtener las matrices de entrada Pm y de salida Tm del motor virtual, posteriormente se genera la red neuronal y se comienza a entrenar con el algoritmo de retropropagación.

La Fig. 3.8 muestra un diagrama a bloques del entrenamiento propuesto para la red neuronal con el conjunto de entrenamiento antes mencionado.

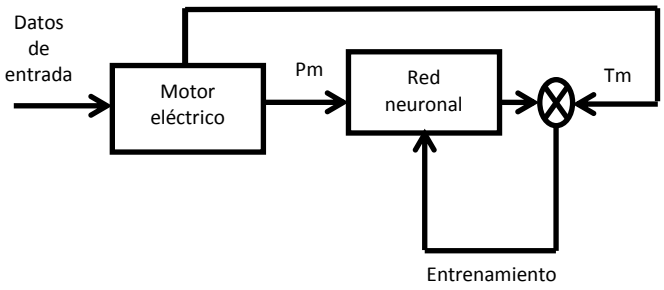


Fig. 3.8 Diagrama a bloques del entrenamiento de la red propuesta

3.7 Verificación de la red neuronal propuesta

Después del proceso de entrenamiento los pesos de las conexiones en la red neuronal quedan fijos. Como paso siguiente se debe comprobar si la red neuronal puede resolver nuevos problemas, del tipo general, para los que ha sido entrenada. Por lo tanto, con el propósito de validar la red neuronal se requiere de otro conjunto de datos, denominado *conjunto de validación.*

Cada ejemplo del conjunto de evaluación contiene los valores de las variables de entrada, con su correspondiente solución tomada; pero ahora esta solución no se le es otorgada a la red neuronal. Luego se compara la solución calculada para cada ejemplo de validación con la solución conocida.

Los conjuntos de datos para la verificación de la red neuronal entrenada se generan del mismo motor virtual que genera los datos para el entrenamiento de la red propuesta, pero los datos son diferentes de los usados para el entrenamiento, aunque están dentro del rango anterior. Para ello se calcula el error máximo de la siguiente manera:

$$error = abs\left[\frac{T_{deseada} - T_{obtenida}}{T_{deseada}} \cdot 100\right] \tag{3.19}$$

Dónde:

$T_{deseada}$ - Temperatura generada por el motor virtual.

$T_{obtenida}$ - Temperatura generada por la simulación de la red neuronal entrenada.

3.8 Diseño de la red neuronal propuesta usando el software MATLAB

Aprovechando el modo interactivo del Toolbox Neural Network [51] del sistema computacional MATLAB, se plantea de esta manera llevar a cabo el proceso de diseño de la red neuronal propuesta. Para ello se llevan a cabo los procedimientos que a continuación se establecen.

3.8.1 Recolección de datos

Se refiere esencialmente a contar con un conjunto de datos de entrenamiento, dicho conjunto esta formado por las matrices Pm para las entradas y el conjunto Tm para las salidas, puesto que se trata de un aprendizaje supervisado. Ambas matrices se generan con el motor virtual en elemento finito descrito en el Capítulo 2.

3.8.2 Creación de la red

Para crear la red neuronal era necesario ejecutar el comando ***nntool*** para introducir las matrices Pm como datos de entrada y Tm como salida de acuerdo al diagrama de la Fig. 3.15.

3.8.3 Configuración de la red

En la configuración de la red se definieron los parámetros con los cuales opera la red neuronal. A continuación se enlistan las propiedades que se definen en este apartado:

3.8.3.1 Nombre de la red

Se asigna un nombre a la red, en este trabajo se tomaron los nombres de acuerdo a la arquitectura y a la función de entrenamiento (ejemplo: trainlm_2capas20 que significa que fue una red de 2 capas con 20 neuronas en la primera capa y entrenada con la función trainlm que indica que fue entrenada con el algoritmo de Levenberg-Marquardt).

3.8.3.2 Tipo de red

Se asigna el tipo de red que se desea crear, en este caso como ya se aclaró anteriormente se eligió trabajar con redes de **retropropagación** para todos los casos simulados. En el apéndice G se muestra una clasificación de las redes neuronales.

Para comprender de mejor manera el algoritmo de retropropagación se desarrolló un programa de acuerdo al diagrama de flujo del algoritmo en la Fig. 3.7, este programa se encuentra en el Apéndice D-1.

3.8.3.3 Datos de entrada

Dentro del conjunto de entrenamiento se encuentran los datos de entrada, en este trabajo se refieren a la matriz **Pm** que resulta del motor virtual desarrollada en el capítulo 2.

En el caso del estator esta matriz esta conformada por la combinación de los vectores:

Q_{de} - pérdidas eléctricas en el devanado del estator.
Q_{cme} - pérdidas magnéticas en el circuito magnético del estator.
$alfa_{de}$ - coeficiente de transferencia de calor por convección por el lado del devanado del estator.
Alfa $_{cme}$- coeficiente de transferencia de calor por convección por el lado del circuito magnético del estator.
t – tiempo.

En el caso del rotor esta matriz esta conformada por la combinación de los vectores:

Q_{dr} - pérdidas eléctricas en el devanado del rotor.
Q_{cmr} - pérdidas magnéticas en el circuito magnético del rotor.
$alfa_{dr}$ - coeficiente de transferencia de calor por convección por el lado del devanado del rotor.
t – tiempo.

3.8.3.4 Datos de salida

Dentro del conjunto de entrenamiento se encuentran los datos de salida, en este trabajo se refieren a la matriz **Tm** que resulta del motor virtual desarrollado en el Capítulo 2.

En el caso del estator esta matriz esta conformada por los vectores de temperaturas generadas por el motor virtual en los nodos seleccionados:

T_1 – Temperatura generada en un nodo del estator
T_2 – Temperatura generada en otro nodo del estator

En el caso del rotor esta matriz esta conformada por los vectores de temperaturas generadas por el motor virtual en los nodos seleccionados:

T_3 – Temperatura generada en un nodo del rotor
T_4 – Temperatura generada en otro nodo del rotor

3.8.3.5 Función de entrenamiento

El Toolbox Neural Networks de MATLAB contiene diversas funciones de entrenamiento. Es necesario seleccionar la función con la que se desea optimizar el entrenamiento de la red neuronal. En esta investigación se hicieron pruebas con todas las contenidas en la

librería para el algoritmo de retropropagación, con la finalidad de corroborar la función que mejor comportamiento tiene en la corrección del error.

En el Apéndice H se presenta un resumen de cada una de las funciones de entrenamiento contenidas en el Toolbox Neural Networks de MATLAB.

3.8.3.6 Número de capas

Continuando con el proceso de configuración de la red, era necesario también definir el número de capas que formarían la red. La literatura para ecuaciones de termo-transferencia de calor nos dice que se utilizan redes de 2 y 3 capas [39, 40, 41].

Por ello, en esta investigación se trabajó con redes neuronales de 2 y 3 capas con diversas combinaciones de neuronas hasta alcanzar una mayor reducción en el error.

Es importante aclarar que no existe una metodología para determinar el número de capas y neuronas, en este trabajo se toman basados en anteriores investigaciones como se menciono pero básicamente se hace de forma heurística.

3.8.3.7 Cantidad de neuronas por capa

Fue necesario de igual manera establecer el número de neuronas en cada capa, puesto que, como se mencionó en el anterior punto de esta tesis, para disminuir el error también fue necesario hacer pruebas con cantidades de neuronas diversas hasta encontrar la combinación que diera el error mas bajo de forma heurística, los resultados se muestran en el capitulo 4.

3.8.3.8 Función de activación en cada capa

Finalmente en el proceso de configuración de la red es necesario seleccionar la función de activación en cada capa de la red, de acuerdo al principio de funcionamiento del algoritmo de retropropagación, en las capas ocultas la función es continua unipolar y

continua bipolar (tipo sigmoidal) son las habituales y en la capa de salida regularmente se emplea la función rampa unipolar (tipo lineal) [48].

3.8.3.9 Entrenamiento de la red

El proceso que se lleva mayor tiempo es el del entrenamiento de la red; sin embargo, es importante considerar la capacidad de cómputo con la que se cuenta puesto que de ello depende el tiempo que se lleva el entrenamiento.

En esta investigación se realizaron diversos procesos de entrenamiento con diversas arquitecturas, número de neuronas, funciones de entrenamiento, número de capas, etc. Para finalmente encontrar el conjunto adecuado que permitiera tener los valores mas aproximados al MEF.

Es importante comentar que la inicialización de pesos se realiza de manera aleatoria por lo que en cada sesión de entrenamiento se inició con distintos pesos hasta reducir el error al valor deseado.

Se comprobó que el algoritmo de Levenberg-Marquardt fue la función de entrenamiento que mejores resultados dio durante la etapa del entrenamiento de la red. Este algoritmo se explica en el Apéndice C en su forma aplicada al entrenamiento a redes neuronales.

También se demuestra que la red de 3 capas con 25 neuronas en la primera capa y 10 neuronas en la segunda capa fue la arquitectura con la que se logro reducir el error de entrenamiento más apegado al MEF.

Todo ello se muestra detalladamente en el capítulo 4 de este trabajo de tesis.

3.8.4 Simulación de la red entrenada

Una vez que la red está entrenada y validada, se puede hablar de que la red se encuentra lista para simularse mediante el comando **sim** del Toolbox Neural Networks de MATLAB [51] y obtener los campos de temperatura de los puntos seleccionados de la máquina eléctrica de una forma rápida, confiable y sobre todo sin tener un amplio conocimiento de las ecuaciones térmicas o electromagnéticas del motor eléctrico.

3.9 Diagrama de flujo del algoritmo propuesto para la red neuronal

A continuación se presenta un diagrama de flujo del algoritmo desarrollado en este trabajo de tesis.

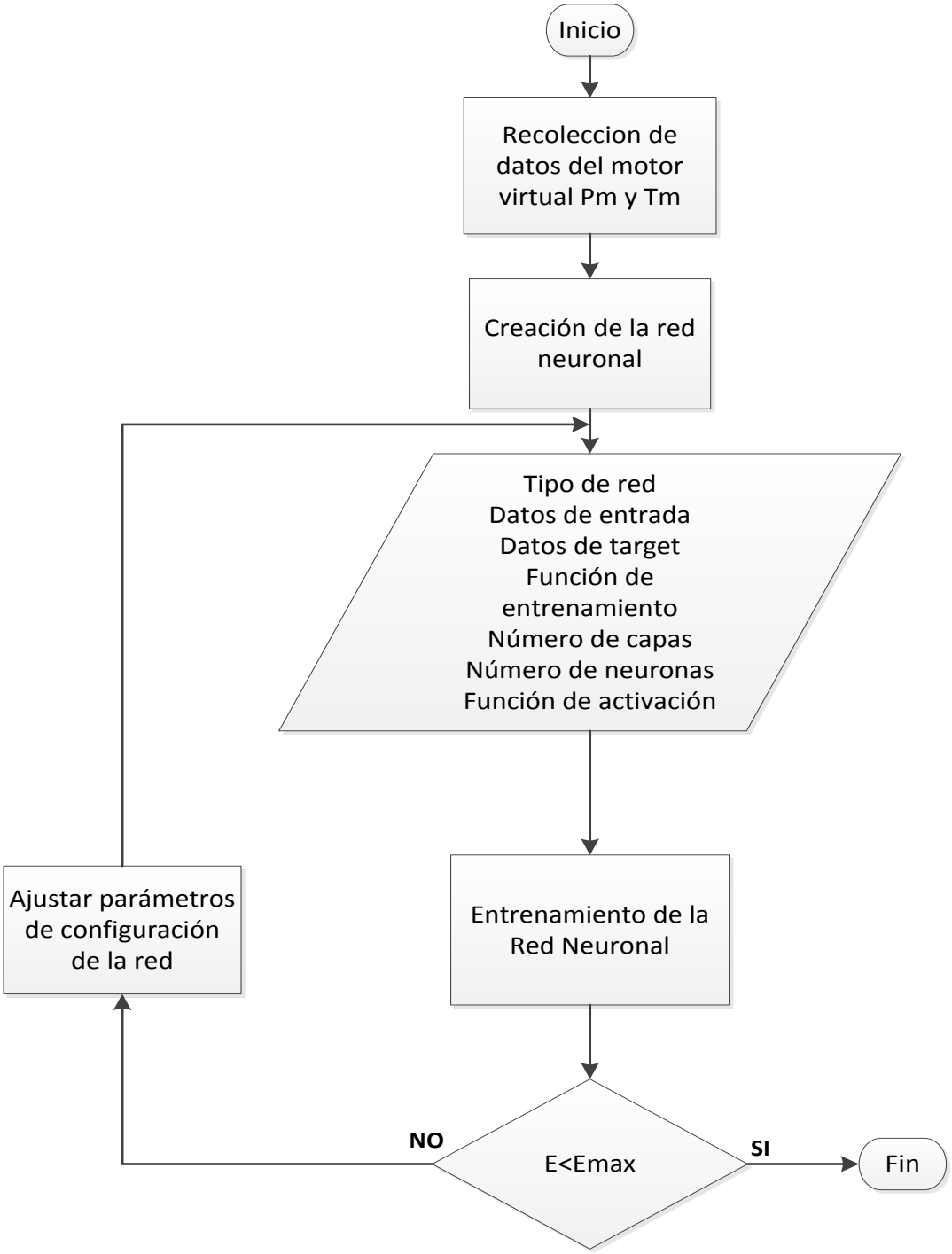


Fig. 3.9 Diagrama de flujo del algoritmo propuesto de red neuronal

CAPÍTULO 4

SIMULACIÓN DEL SISTEMA PROPUESTO Y ANÁLISIS DE RESULTADOS

4.1 Introducción

En este capítulo se presentan los resultados obtenidos de la metodología propuesta para simular los campos de temperatura de un motor de inducción doble jaula de ardilla con redes neuronales así como el análisis de los resultados obtenidos.

Para mejor comprensión de la metodología mostrada en la Fig. 3.9 se explica paso a paso desde la recolección de datos hasta la simulación de las que mejores respuestas dieron.

En el Capítulo 3 se definió que la regla de aprendizaje seria el algoritmo de retropropagación, sin embargo también fue necesario obtener lo siguiente:

1. El algoritmo de optimización que mejor se adecúa al problema.
2. El número de neuronas y de capas necesarias para lograr la reducción del error.
3. Verificación de la red neuronal entrenada.

A continuación se detalla la forma como se hizo y las consideraciones que se tomaron para la elección de la arquitectura final.

4.2 Selección del algoritmo de optimización

El primer paso era definir el algoritmo de optimización que mejor comportamiento tuviera en cuanto al tiempo y a la reducción del error de entrenamiento. Para ello se eligió un conjunto de entrenamiento bastante grande con la finalidad de ver el comportamiento que cada algoritmo tiene ante dicho conjunto de datos.

Para determinar el algoritmo de optimización adecuado al problema, se realizaron diversos entrenamientos para 2 redes neuronales que se eligieron en base a las

referencias [39,40,41] que son comunes en este tipo de problemas y que ciertamente se realizo de forma heurística, la primera es una red de dos capas con 20 neuronas en la primera capa y 2 neuronas en la segunda capa, la segunda red que se eligió fue de 3 capas con 20 neuronas en la primera capa, 10 neuronas en la segunda capa y 2 neuronas en la tercera capa. Se realizaron ejercicios de entrenamiento con los diferentes métodos contenidos en la librería Neural Networks de MATLAB los cuales se detallan en el Apéndice H.

4.2.1 Conjunto de entrenamiento

En el caso investigado, se simularon los procesos de transferencia de calor con los siguientes datos:

Para el caso del estator:

$Q_{de} = 0.5x10^6 - 2.0x10^6$ W/m^3;
$\Delta Q_{de} = 0.5x10^6$;
$Q_{cme} = 0.5x10^5 - 2.0x10^5$ W/m^3;
$\Delta Q_{cme} = 0.5x10^5$;
α_{de} = 50-500 W / (m^2 K);
$\Delta\alpha_{de}$= 50;
α_{cme} = 50-500 W / (m^2 K);
$\Delta\alpha_{cme}$= 50;
t = [10 50 100 300 600 900 1200 1500];

Para obtener la matriz Pm fue necesario ejecutar el comando **combvec** de MATLAB el cual permite hacer una combinación de vectores.

$$Pm= (Q_{de} * Q_{cme} * \alpha_{de} * \alpha_{cme} * t) \quad (4.1)$$

La matriz Tm corresponde al resultado que se obtiene de resolver mediante el MEF, cada columna de la matriz Pm para cada nodo seleccionado.

Por lo tanto el tamaño del conjunto de entrenamiento depende del tamaño de los vectores de entrada ya que se hace la combinación de todos los datos.

Debido a lo anterior, este conjunto de datos representa una matriz de entrada Pm de 5x18432 y una matriz de salida Tm de 2x18432, es decir, se trata de un conjunto de entrenamiento bastante grande.

Para el caso del rotor:

$Q_{dr} = 0.5x10^6 - 2.0x10^6$ W/m^3;
$\Delta Q_{dr} = 0.5x10^6$;
$Q_{cmr} = 0.5x10^5 - 2.0x10^5$ W/m^3;
$\Delta Q_{cmr} = 0.5x10^5$;
$\alpha_{dr} = 50\text{-}500$ W / (m^2 K);
$\Delta\alpha_{dr} = 50$;

La matriz Pm se calcula de la misma forma que para el estator, en este caso el tiempo se considera igual que para el estator como lo indica la Fig. 3.1, y la diferencia es que solo existe un coeficiente de transferencia de calor por convección quedando de la siguiente manera:

$$Pm = (Q_{dr} * Q_{cmr} * \alpha_{dr} * t) \quad (4.2)$$

Igualmente se trata de un conjunto de datos grande, una matriz de entrada Pm de 5x1536 y una matriz de salida Tm de 2x1536, no obstante, al ser un vector menos que en el caso del estator, se disminuye considerablemente el tamaño del conjunto de entrenamiento.

4.2.2 Resultados de las simulaciones estator

- Primer caso:

Arquitectura de dos capas, con 20 neuronas en la primera capa y dos en la segunda capa.

La estructura de esta red se muestra en la Fig. 4.1 la creada por **nntool** de MATLAB y la de la Fig. 4.2 generada en **simulink**.

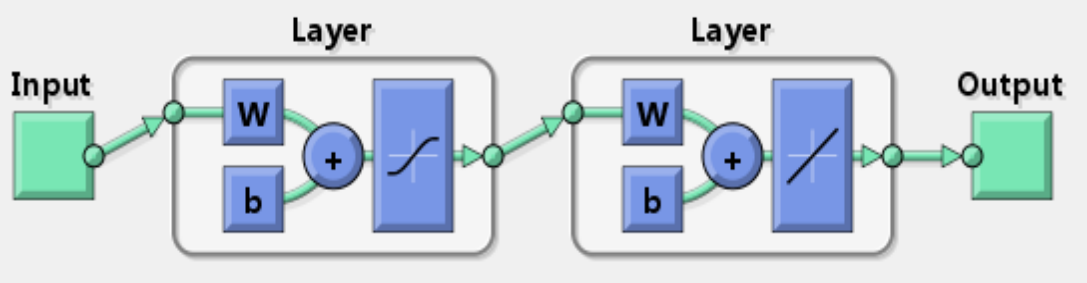


Fig.4.1 Arquitectura de dos capas generada por nntool de MATLAB para el primer caso del estator

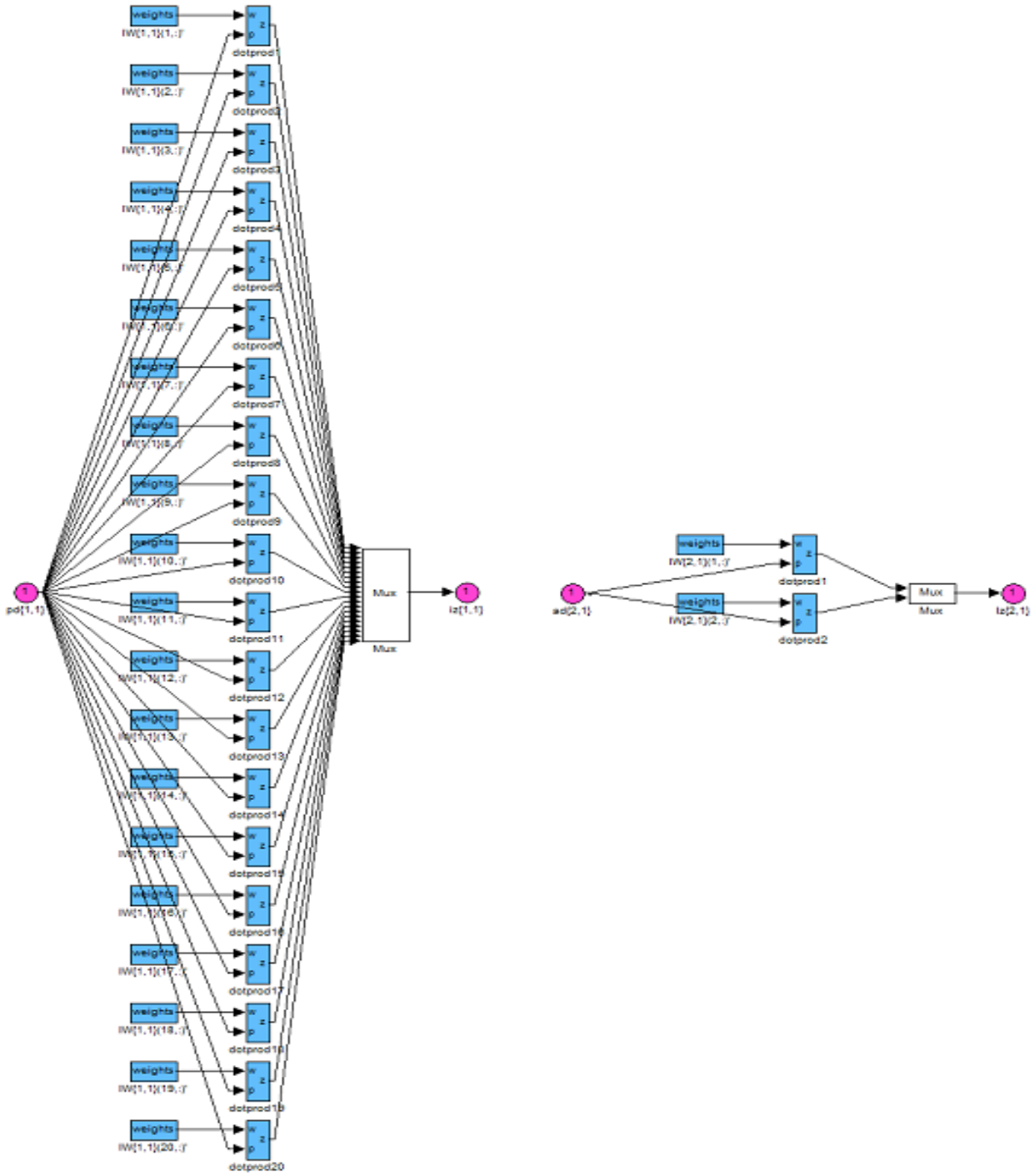


Fig. 4.2 Arquitectura de dos capas generada por Simulink de MATLAB para el primer caso del estator

Para seleccionar el algoritmo de entrenamiento se realizaron diez entrenamientos de cada función de entrenamiento para la arquitectura de dos capas con 20 neuronas en la primera capa y dos neuronas en la segunda capa mostrada en la Fig. 4.2.

En la Fig. 4.3 se muestra el entrenamiento que redujo más el error y fue el de la función trainlm, que corresponde al algoritmo de Levenberg-Marquardt.

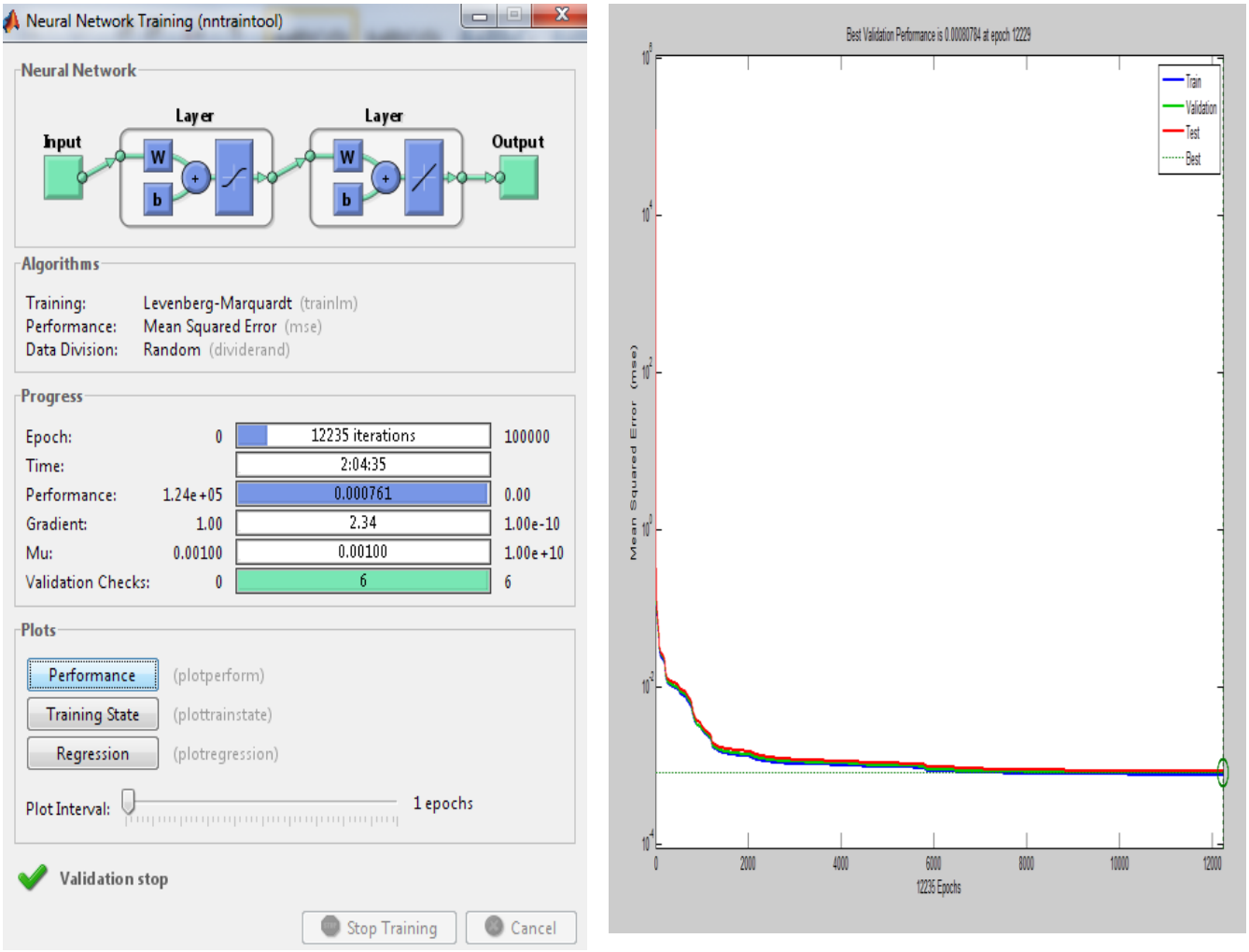


Fig. 4.3 Entrenamiento con el algoritmo de Levenberg-Marquardt para el primer caso del estator

La Tabla 4.1 contiene los resultados de las mejores simulaciones de cada función de entrenamiento para la arquitectura de dos capas con 20 neuronas en la primera capa y dos neuronas en la segunda capa mostrada en la Fig. 4.2.

Tabla 4.1 Resultado de las funciones de entrenamiento para el primer caso del estator

Entrenamiento	Iteraciones	Tiempo	Error
Trainlm	12235	02:04:35	0.000762
Traincgb	1392	00:06:32	0.218
Traincgp	976	00:04:50	0.65
Traincgf	1487	00:06:54	0.699
Trainscg	966	00:05:57	0.944
Trainrp	3464	00:10:15	1.09
Trainbfg	100000	53.16.04	3.38
Trainoss	403	00:02:24	4.29
Trainbr	5121	01:03:25	36.2
Traingdx	131	00:00:17	97.9
Traingda	171	00:00:24	246
Traingd	6	00:00:01	53300
Traingdm	6	00:00:01	54200
Trainr	100	01:07:05	50400000

Debido al tamaño del conjunto de datos para el entrenamiento se notan los altos tiempos que cada función se llevó en entrenar la red, en especial el método de Levenberg-Marquardt. Sin embargo, para los fines de la investigación, se considera prioridad la reducción del error, sin importar el tiempo de entrenamiento o el número de iteraciones porque la finalidad es reducir el error lo más posible. Se nota también que las funciones traingd y traingdm truncan el entrenamiento a un segundo de haberlo iniciado y el error es muy elevado. Se nota que el trainbfg ejecuta un gran número de iteraciones pero no logra disminuir el error.

El hecho de que no se disminuya el error con los demás métodos no significa que no sirvan, simplemente no se hizo el ajuste a cada uno, en esta investigación de realizaron los entrenamientos con los parámetros con los que aparecen precargados automáticamente en el software MATLAB.

- Segundo caso:

Arquitectura de tres capas, con 20 neuronas en la primera capa, 10 neuronas en la segunda capa y dos neuronas en la tercera capa. Figs. 4.4 y 4.5

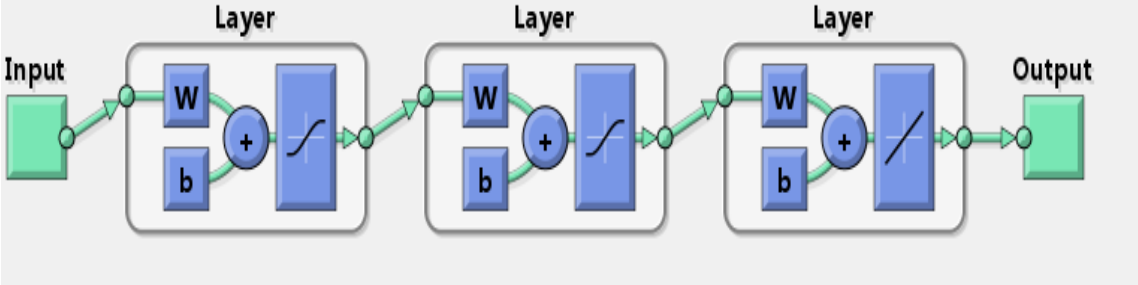


Fig.4.4 Arquitectura de tres capas generada por nntool de MATLAB para el segundo caso del estator

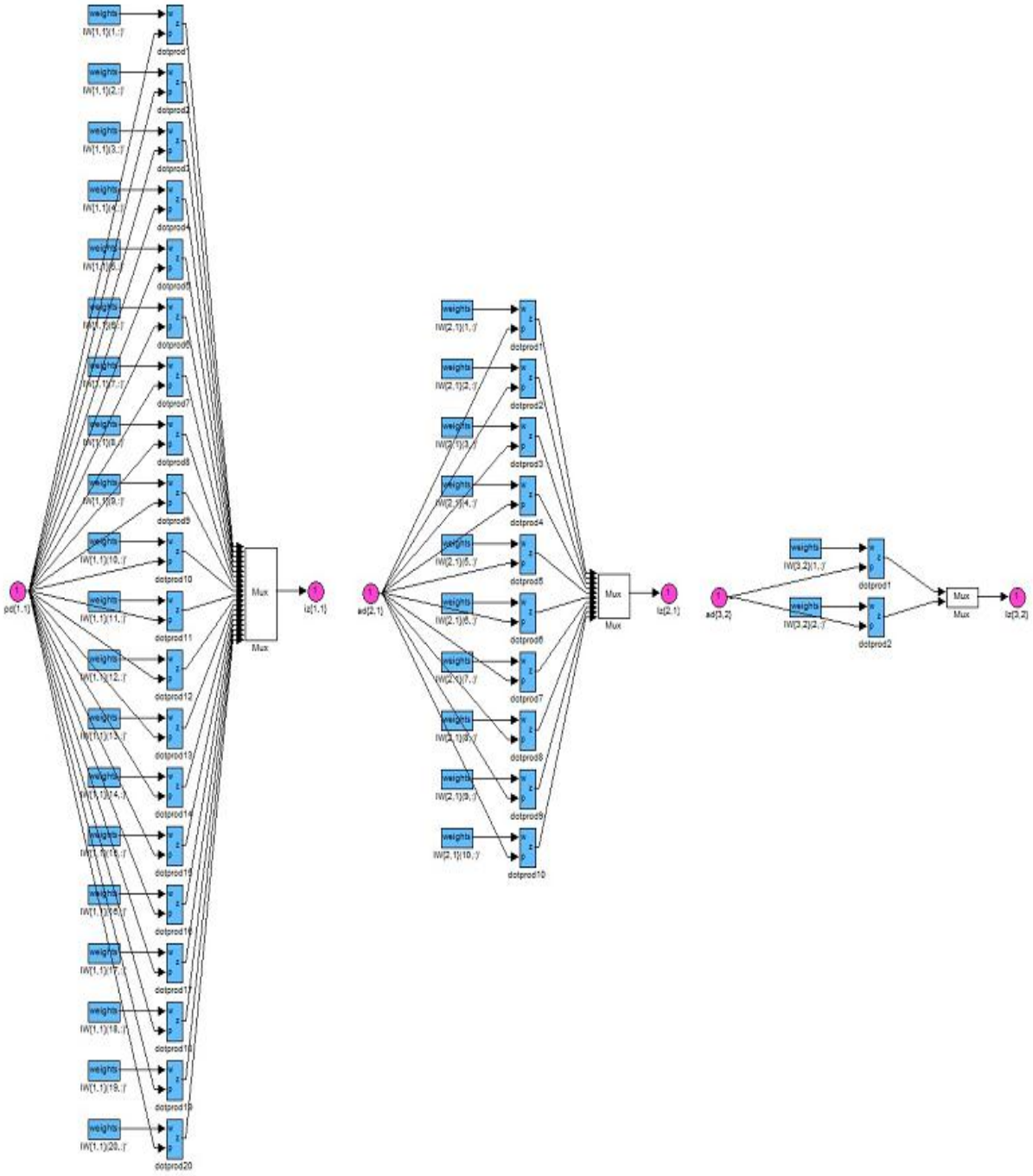


Fig.4.5 Arquitectura de tres capas generada por Simulink de MATLAB para el segundo caso del estator

Para seleccionar el algoritmo de entrenamiento se realizaron diez simulaciones de cada función para la arquitectura de tres capas con 20 neuronas en la primera capa, 10 neuronas en la segunda y dos neuronas en la tercera capa mostrada en la Fig. 4.5. En la Fig. 4.6 se muestra el mejor resultado que igualmente que en el caso anterior fue el de Levenberg-Marquardt.

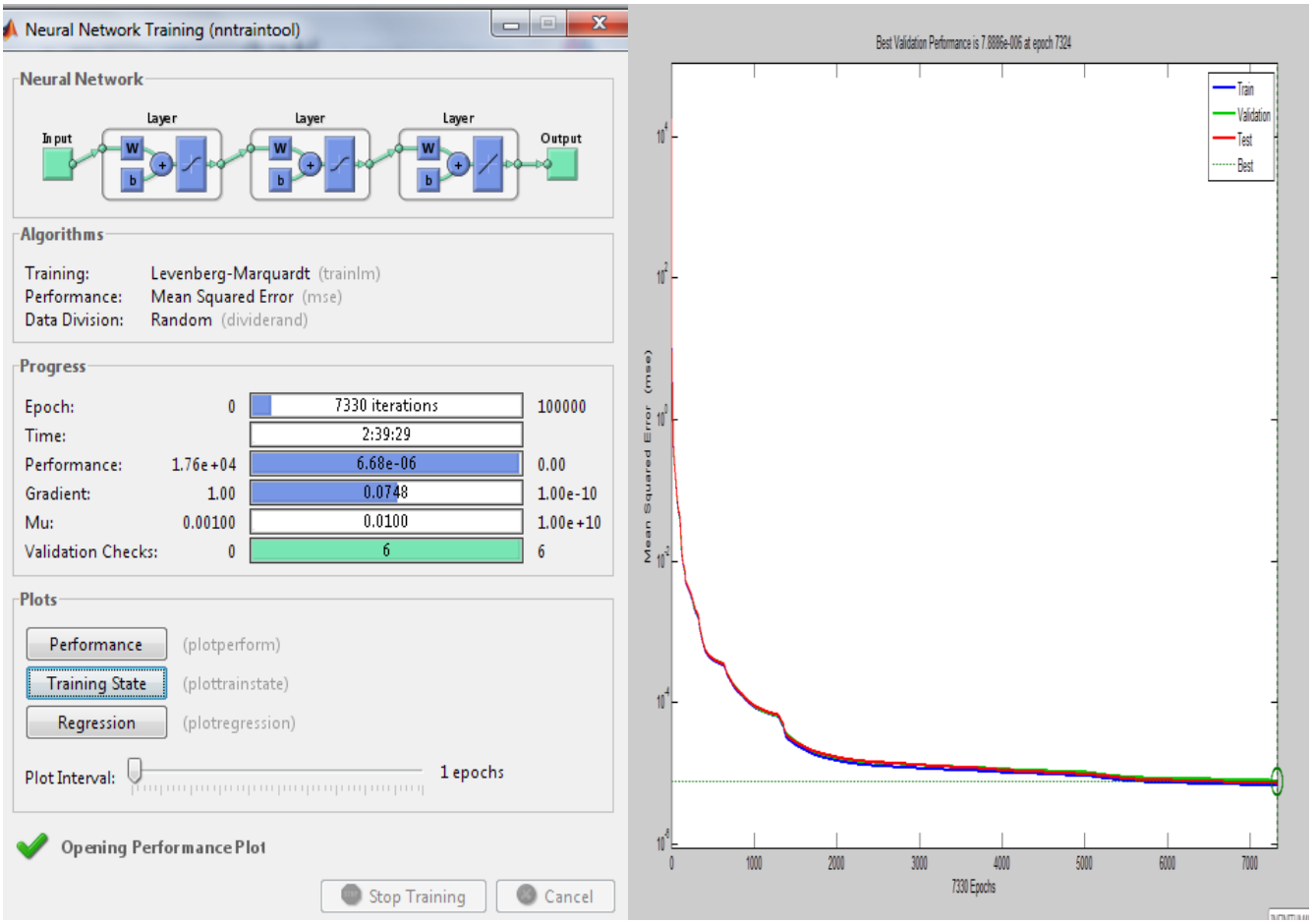


Fig. 4.6 Entrenamiento del algoritmo de Levenberg-Marquardt para el segundo caso del estator

La Tabla 4.2 contiene los resultados de las mejores simulaciones de cada función de entrenamiento para la arquitectura de tres capas, con 20 neuronas en la primera capa, 10 neuronas en la segunda capa y dos neuronas en la tercera capa mostrada en la Fig. 4.5.

Tabla 4.2 Resultado de las funciones de entrenamiento para el segundo caso del estator

Entrenamiento	Iteraciones	Tiempo	Error
Trainlm	7330	02:39:29	6.68E-06
Trainbr	10000	03:30:44	0.236
trainscg	989	00:05:32	0.353
Trainrp	4013	00:11:17	0.115
traincgb	648	00:03:50	0.375
traincgf	343	00:01:52	0.729
traingda	73	00:00:09	390
Trainbfg	18531	08:46:33	2.11
traincgp	716	00:04:12	0.665
trainoss	441	00:00:03	3.42
traingdx	132	00:00:19	107
traingdm	6	00:00:01	53300
Traingd	6	00:00:01	54200
Trainr	12	01:23:56	7041000000

Los resultados de la tabla muestran claramente que el algoritmo de Levenberg-Marquardt nuevamente reduce el error mucho más que los demás métodos, sin embardo el tiempo sigue siendo demasiado alto. En este caso el aumento de una capa hace que se reduzca el error considerablemente respecto al caso 1.

En base a los resultados mostrados en las simulaciones anteriores es evidente que el algoritmo de Levenberg-Marquardt (trainlm) es la función de entrenamiento que mejores rendimientos tuvo para el conjunto de entrenamiento mostrado en 4.2.1 y de la arquitectura de las Figs. 4.2 y 4.5.

Cabe destacar que se hicieron diez simulaciones en cada una de las 14 funciones de entrenamiento por lo que quedo comprobado que después de 140 pruebas el algoritmo de Levenberg-Marquardt (trainlm) es el que reduce mas el error de entrenamiento y será empleado para las posteriores simulaciones en el estator.

Nuevamente se aclara que no se hicieron ajustes a los demás métodos y es posible que algún otro método se ajuste mejor al problema, sin embargo para fines de esta investigación no se hacen esas consideraciones

4.2.3 Resultados de las simulaciones rotor

- Primer caso:

Arquitectura de dos capas, con 20 neuronas en la primera capa y dos en la segunda capa. La estructura de esta red se muestra en la Fig. 4.7 la creada por **nntool** de MATLAB y la de la Fig. 4.8 generada en **simulink.**

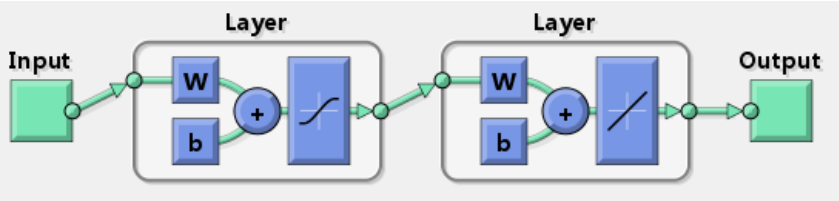


Fig.4.7 Arquitectura de dos capas generada por nntool de MATLAB para el primer caso del rotor

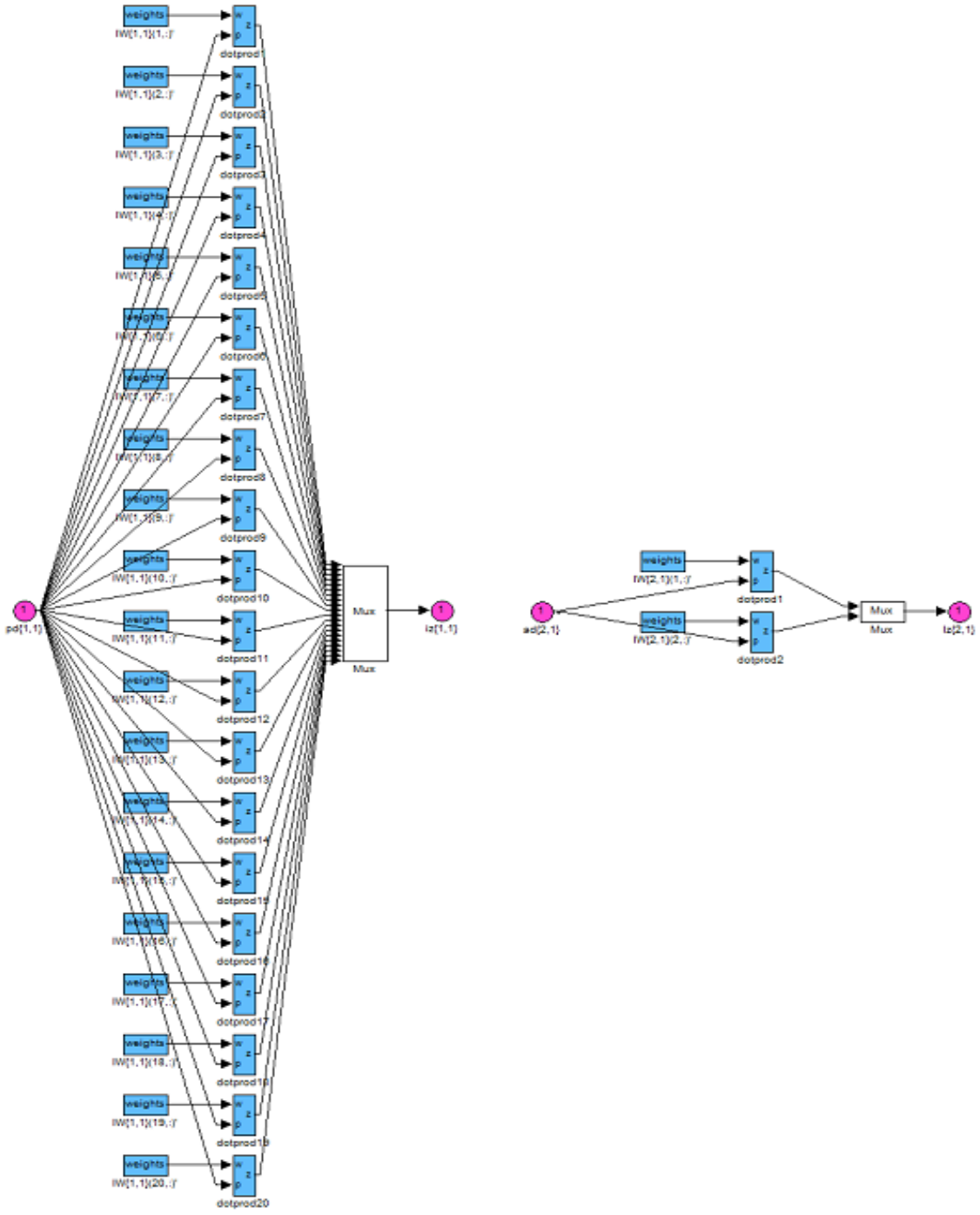


Fig.4.8 Arquitectura de dos capas generada por Simulink de MATLAB para el primer caso del rotor

Para seleccionar el algoritmo de entrenamiento se realizaron diez entrenamientos de cada función de entrenamiento para la arquitectura de dos capas con 20 neuronas en la primera capa y dos neuronas en la segunda mostrada en la Fig. 4.8. En la Fig. 4.9 se muestra el mejor entrenamiento que fue el de Levenberg-Marquardt.

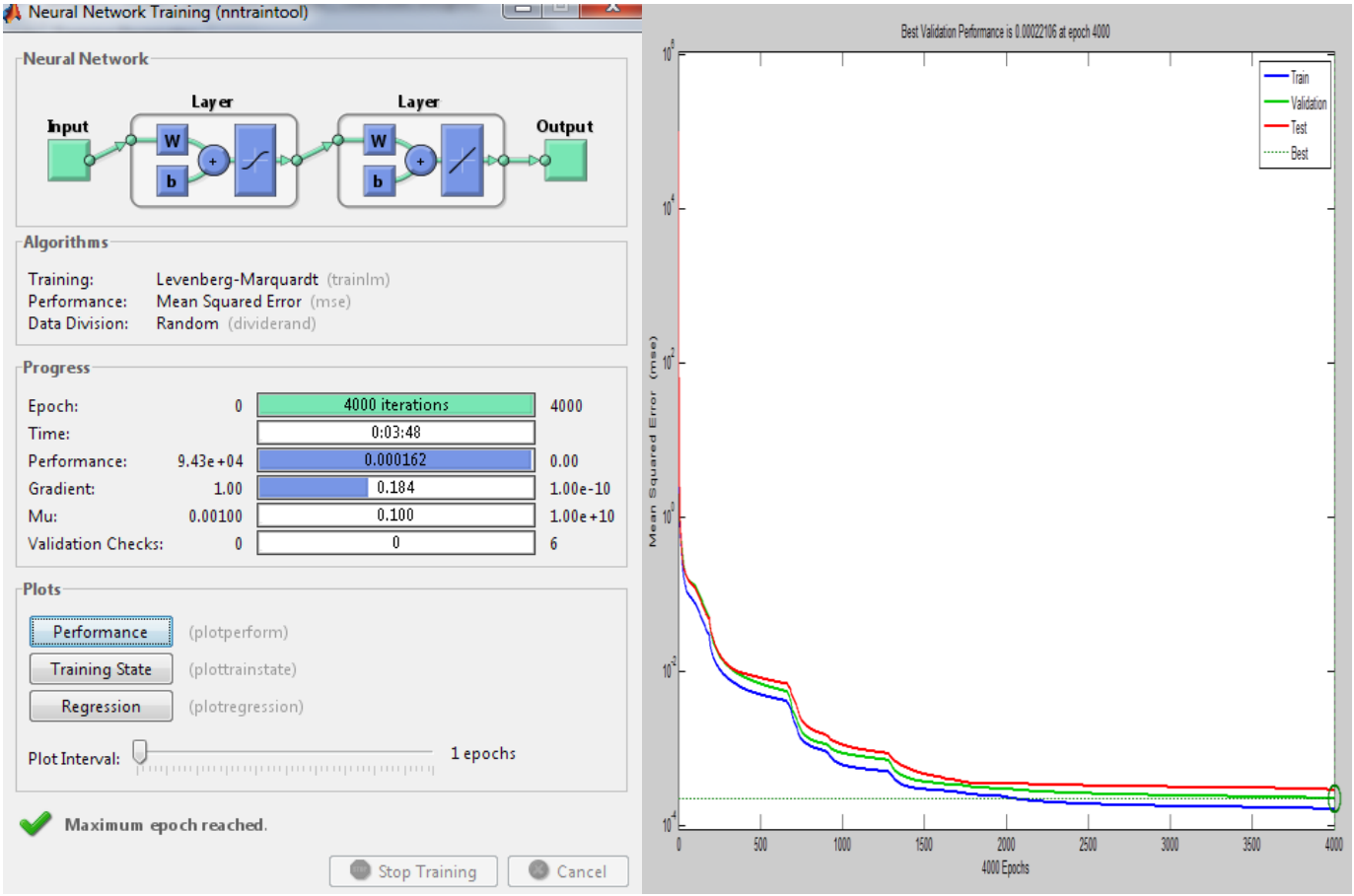


Fig. 4.9 Entrenamiento del algoritmo de Levenberg-Marquardt para el primer caso del rotor

La Tabla 4.3 contiene los resultados de las mejores simulaciones de cada función de entrenamiento para la arquitectura de dos capas con 20 neuronas en la primera capa y dos neuronas en la segunda capa mostrada en la Fig. 4.8.

Tabla 4.3 Resultados de las funciones de entrenamiento para el primer caso del rotor

Entrenamiento	Iteraciones	Tiempo	Error
trainlm	4000	00:03:48	0.000162
trainbr	646	00:00:36	1.06
traincgp	439	00:00:17	2.04
trainrp	806	00:00:18	2.42
trainscg	326	00:00:12	3.36
traincgb	334	00:00:13	3.64
traincgf	283	00:00:12	4.35
trainbfg	10000	00:11:42	6.36
trainoss	223	00:00:13	11.2
traingda	225	00:00:04	109
traingdx	113	00:00:03	129
traingd	6	00:00:01	94900
traingdm	6	00:00:01	99100
trainr	85	00:37:11	57300000

En la tabla 4.3 se observa que el tiempo de entrenamiento se reduce ampliamente comparado con los resultados que se obtuvieron en el estator, esto se debe a la reducción en el conjunto de entrenamiento pues en el estator las matrices de entrada y salida contenían 18432 columnas mientras que en este caso del rotor solo es de 1536 columnas, Es evidente nuevamente que el algoritmo de Levenberg-Marquardt es el que disminuye el error en mayor grado.

- Segundo caso:

Arquitectura de tres capas, con 20 neuronas en la primera capa, 10 neuronas en la segunda capa y dos neuronas en la tercera capa. Figs. 4.10 y 4.11

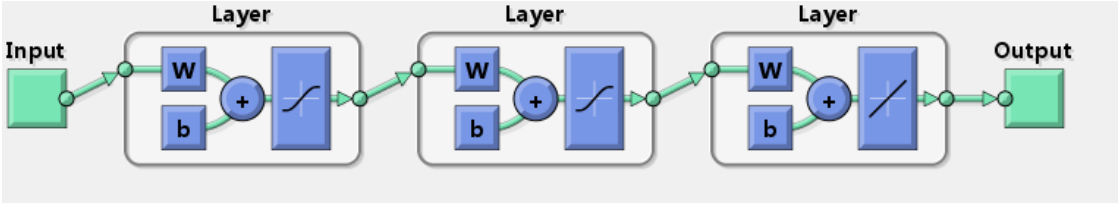


Fig.4.10 Arquitectura de tres capas generada por nntool de MATLAB para el segundo caso del rotor

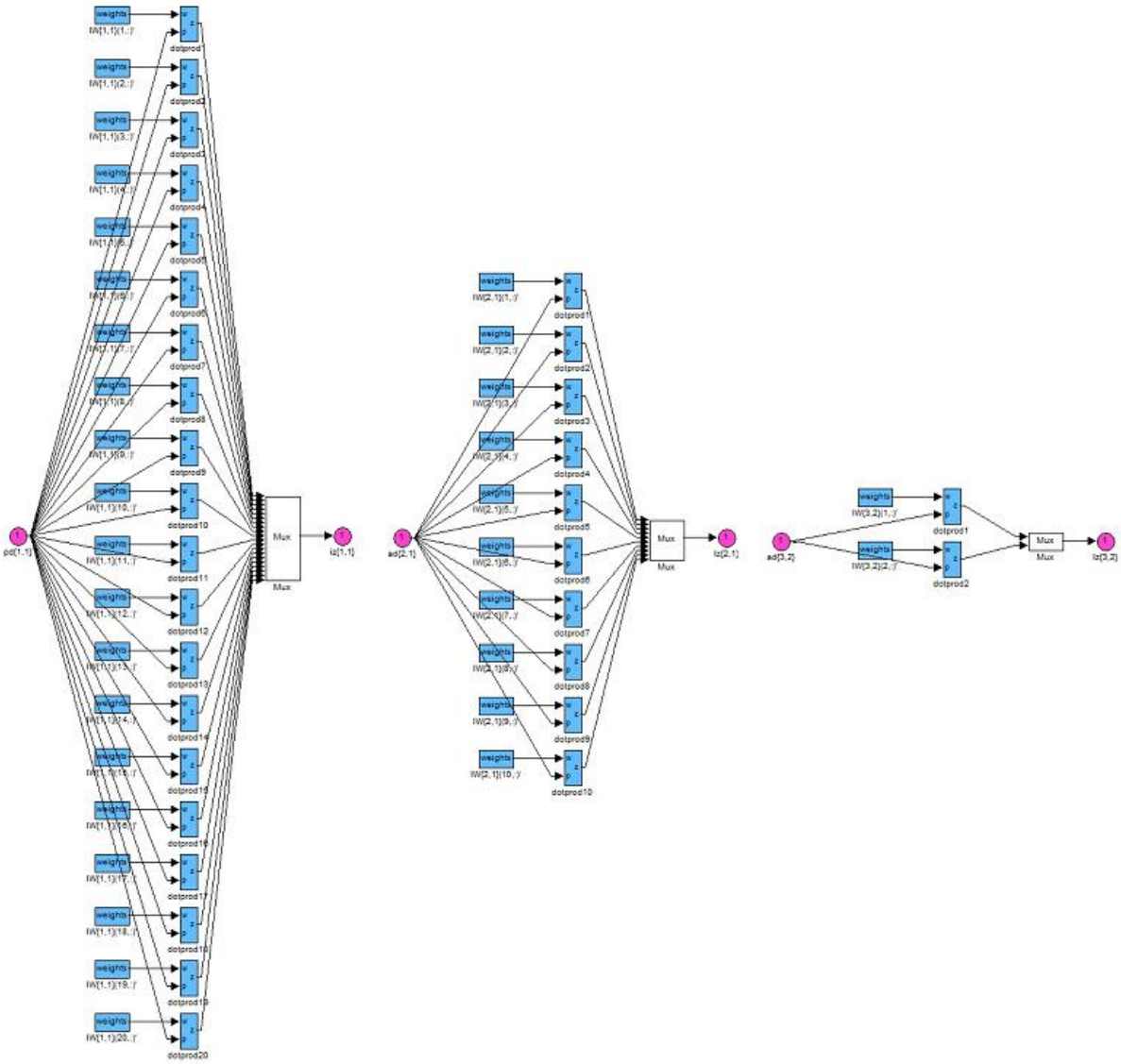


Fig.4.11 Arquitectura de tres capas generada por Simulink de MATLAB para el segundo caso del rotor

Para seleccionar el algoritmo de entrenamiento se realizaron diez entrenamientos de cada función de entrenamiento para la arquitectura de tres capas con 20 neuronas en la primera capa, diez neuronas en la segunda y dos neuronas en la tercera capa mostrada en la Fig. 4.11. En la Fig. 4.12 se muestra el mejor entrenamiento de Levenberg-Marquardt.

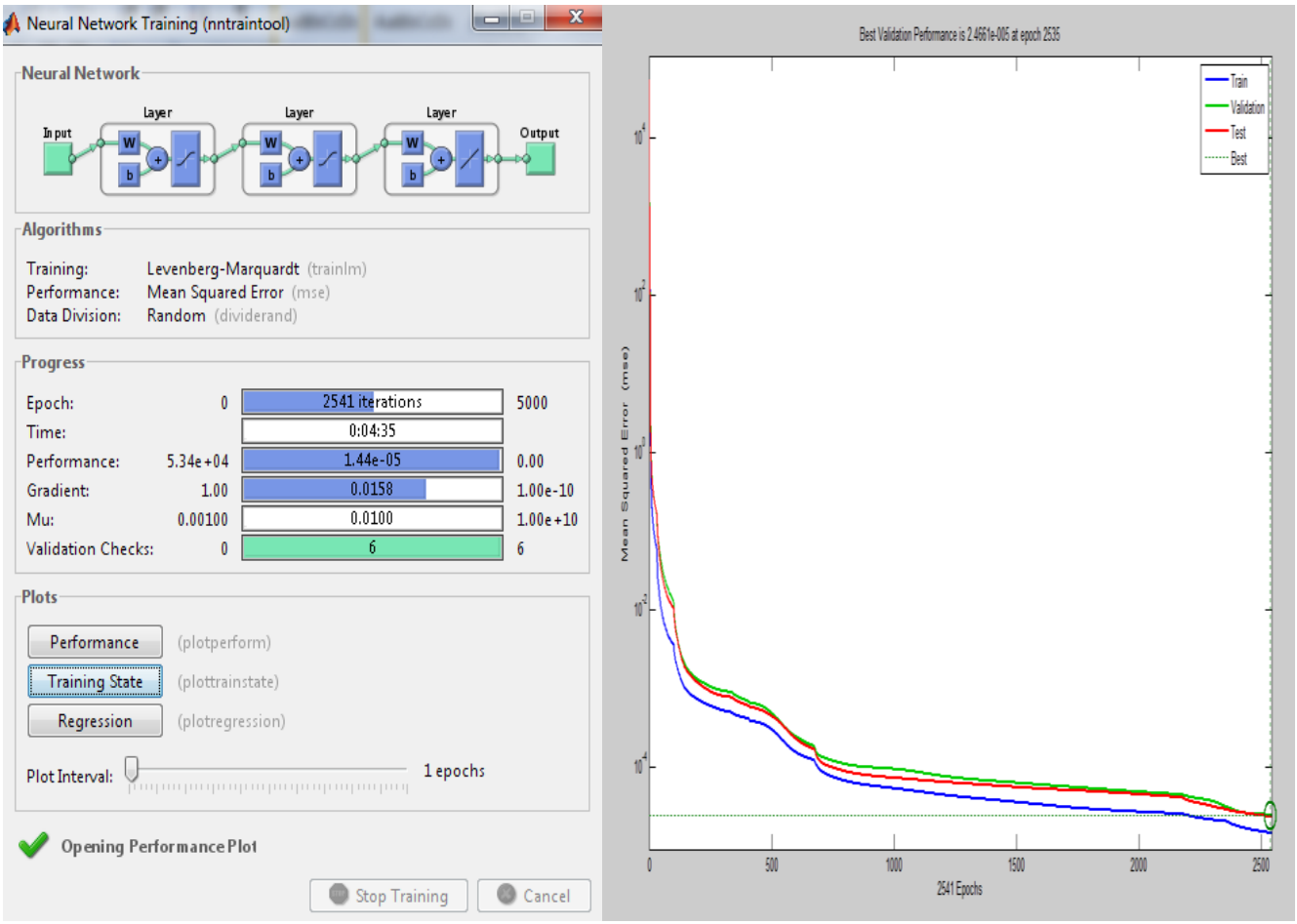


Fig. 4.12 Entrenamiento del algoritmo de Levenberg-Marquardt para el segundo caso del rotor

La Tabla 4.4 contiene los resultados de las mejores simulaciones de cada función de entrenamiento para la arquitectura de tres capas, con 20 neuronas en la primera capa, 10 neuronas en la segunda capa y dos neuronas en la tercera capa mostrada en la Fig. 4.11.

Tabla 4.4 Resultado de las funciones de entrenamiento para el segundo caso del rotor

Entrenamiento	Iteraciones	Tiempo	Error
trainlm	2541	00:04:35	0.0000144
trainbr	3771	00:08:50	0.0328
trainscg	448	00:00:27	1.33
trainrp	536	00:00:19	1.47
traincgb	340	00:00:18	1.89
traincgf	314	00:00:17	1.9
traingda	170	00:00:05	2.38
trainbfg	25000	00:41:18	2.54
traincgp	304	00:00:18	2.7
trainoss	167	00:00:11	20.9
traingdx	140	00:00:04	61.8
traingdm	6	00:00:01	16300
traingd	6	00:00:01	54000
trainr	3	00:00:20	41300000

Nuevamente se muestra que la red de tres capas disminuye el error en la mayoría de la funciones, sin embargo se aumenta el tiempo de entrenamiento. También se nota que el algoritmo de Levenberg-Marquardt es el que por mucho reduce el error.

En base a los resultados mostrados en las simulaciones anteriores es evidente que el algoritmo de Levenberg-Marquardt (trainlm) es la función de entrenamiento que mejores rendimientos tuvo para el conjunto de entrenamiento de la tabla 4.4 y de la arquitectura de las Figs. 4.8 y 4.11.

Cabe destacar que se hicieron diez simulaciones en cada una de las 14 funciones de entrenamiento por lo que quedo comprobado que después de 140 pruebas el algoritmo de Levenberg-Marquardt (trainlm) es el que reduce mas el error de entrenamiento y será empleado para las posteriores simulaciones en el estator.

4.3 Selección de la arquitectura

En el punto anterior se eligió el método de optimización que mas reduce el error con 2 configuraciones de red neuronal, una de dos capas y otra de tres capas, sin embargo, ahora ya con el método de Levenberg-Marquardt como método de optimización, es necesario ver que otras configuraciones pueden disminuir el error de entrenamiento.

Para ello se procedió a realizar diversos entrenamientos con diversas configuraciones seleccionadas arbitrariamente, pero con un nuevo conjunto de entrenamiento con una menor cantidad de datos, esto con la finalidad de ver el comportamiento de la red al momento de ser entrenada.

4.3.1 Conjunto de entrenamiento

Para el caso del estator:

$Q_{de} = 0.5x10^6 - 1.0x10^6$ W/m^3;
$\Delta Q_{de} = 0.25x10^6$;
$Q_{cme} = 1.0x10^5 - 2.0x10^5$ W/m^3;
$\Delta Q_{cme} = 0.5x10^5$;
α_{de} = [50 100 250 400] W / (m^2 K);
α_{cme} = [50 100 250 400] W / (m^2 K);
t = [10 50 150 300 700 1000 2000] s;

Este conjunto de datos representa una matriz de entrada Pm de 5x1008 y una matriz de salida de 2x1008, es decir, se trata de un conjunto de entrenamiento mucho mas pequeño que el conjunto empleado para determinar el método de optimización.

Para el caso del rotor:

Q_{dr} = 0.5x10^6 - 1.0x10^6 W/m^3;
ΔQ_{dr} = 0.25x10^6;
Q_{cmr} = 1.0x10^5 - 2.0x10^5 W/m^3;
ΔQ_{cmr} = 0.5x10^5;
α_{dr} = [50 100 250 400] W / (m^2 K);

Igualmente se trata de un conjunto de datos pequeño, una matriz de entrada Pm de 4x252 y una matriz de salida de 2x252.

4.3.2 Arquitectura del estator

Se eligieron varias configuraciones con diversas cantidades de capas y neuronas en base a las referencias que han tratado este tipo de problemas. Se hicieron 10 entrenamientos con cada una de las arquitecturas que se muestran en la tabla 4.5 y a continuación se muestra la que mejores resultados generó.

- Arquitectura de tres capas, con 25 neuronas en la primera capa, 10 neuronas en la segunda capa y dos neuronas en la tercera capa. Figs. 4.13 y 4.14

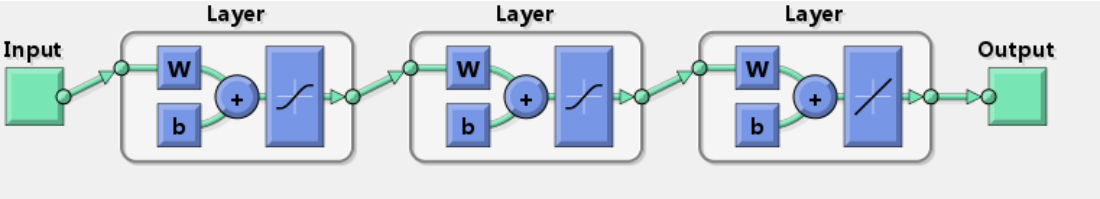


Fig.4.13 Arquitectura de tres capas generada por nntool de MATLAB de la red neuronal que mejores resultados obtuvo para el estator

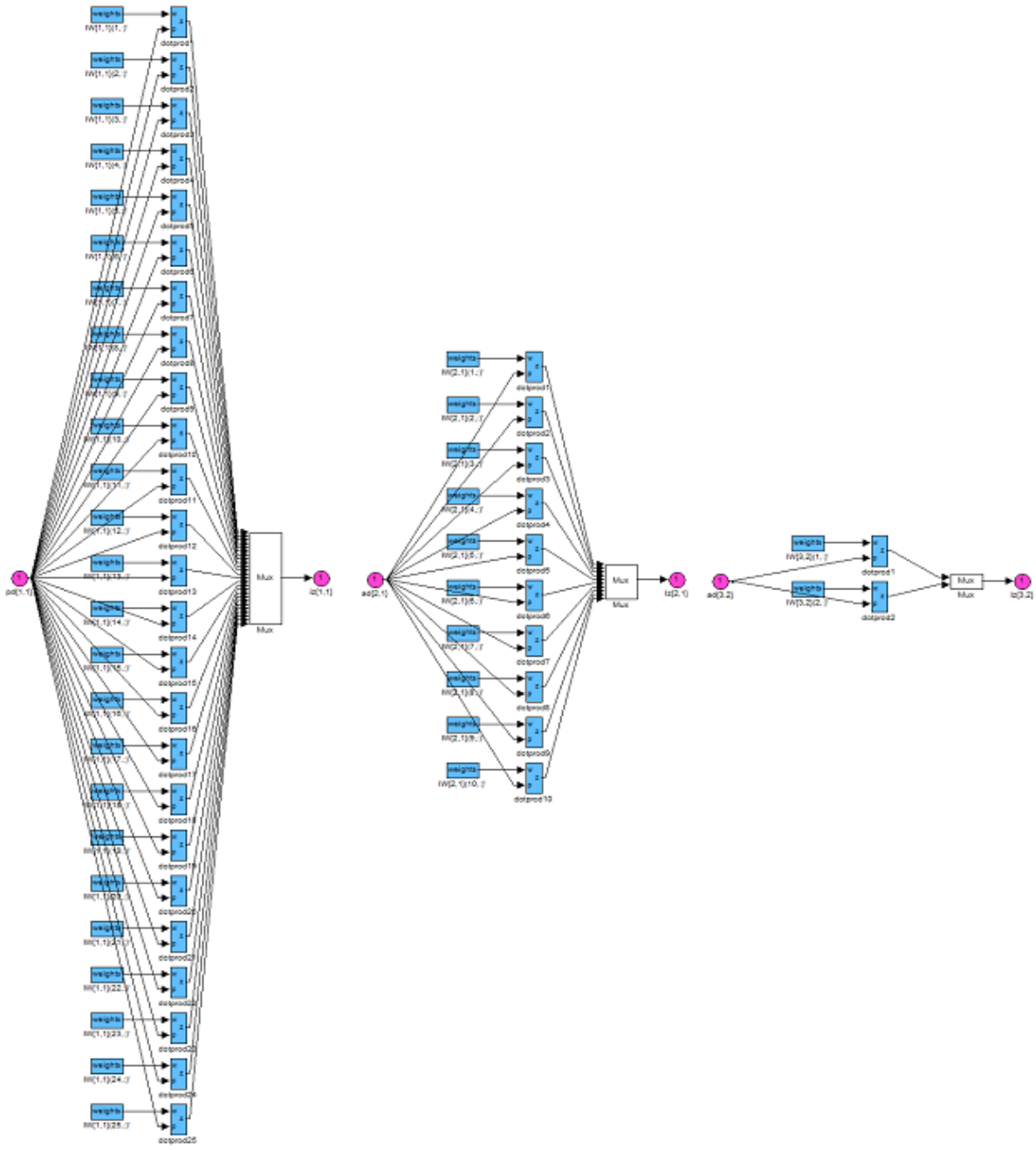


Fig.4.14 Arquitectura de tres capas generada por Simulink de MATLAB de la red neuronal que mejores resultaos obtuvo para el estator

Como lo muestra la Fig. 4.14 la arquitectura de tres capas con 25 neuronas en la primera capa, diez en la segunda capa y dos en la tercer capa es la que logro reducir más el error mismo que se muestra en la Fig. 4.15.

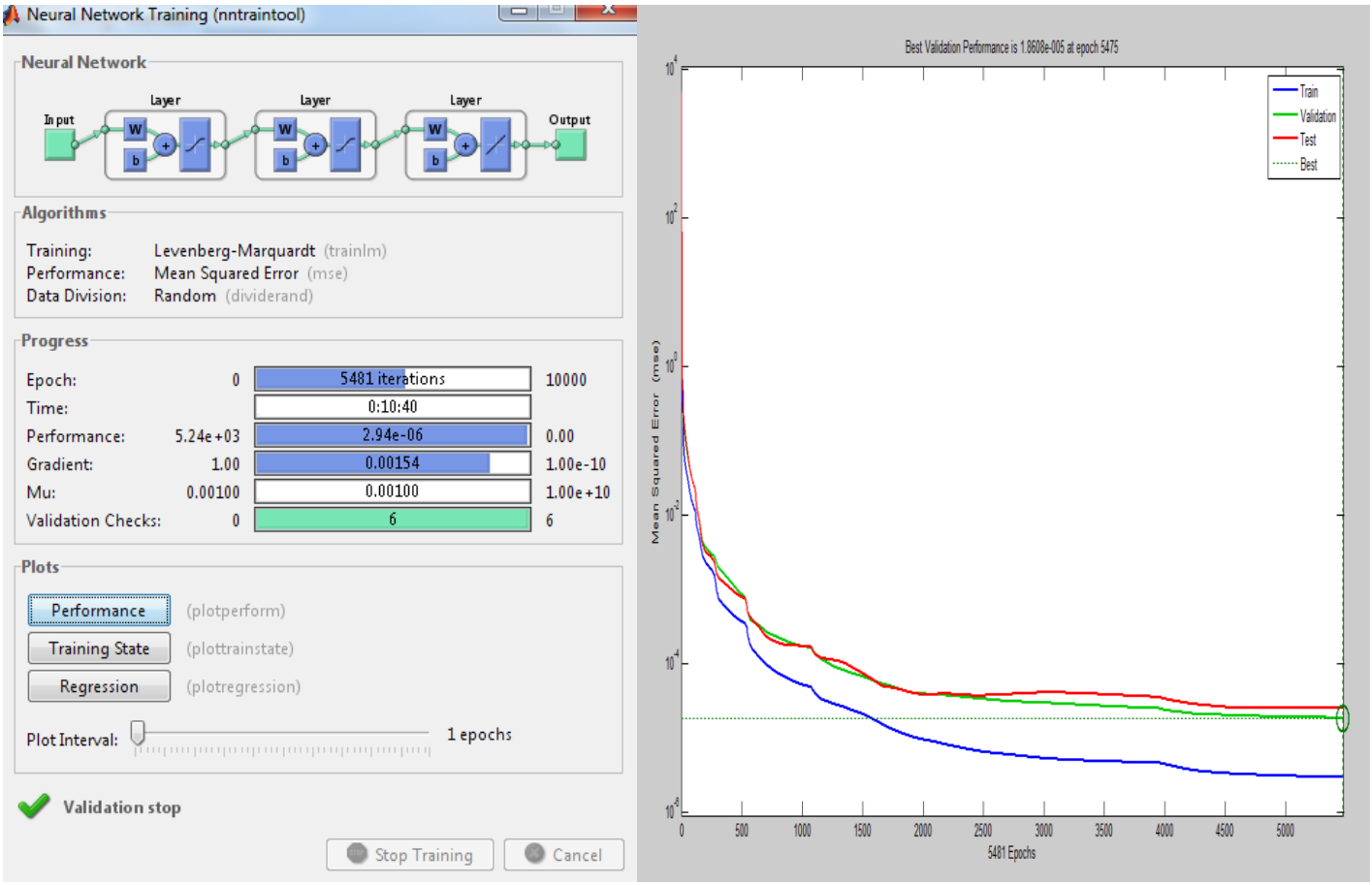


Fig. 4.15 Entrenamiento con mejor rendimiento de la red neuronal para el estator

La Tabla 4.5 contiene los resultados de las mejores simulaciones de cada arquitectura simulada para el estator. Es importante recordar que la capa de salida consta de dos neuronas por lo que en la tabla ya no aparece.

Tabla 4.5 Entrenamientos de diversas arquitecturas para estator

ESTATOR			
Levenberg-Marquardt	**Iteraciones**	**Tiempo**	**Error**
3 capas(25 y 10)	5481	00:10:40	2.94E-06
3 capas(35 y 20)	5717	00:14:19	9.11E-06
3 capas(35 y 10)	1376	00:08:36	9.20E-06
3 capas(35 y 5)	2833	00:08:08	1.09E-05
3 capas(25 y 5)	2681	00:07:19	2.45E-05
3 capas(30 y 10)	1616	00:08:55	5.87E-05
3 capas(20 y 5)	1435	00:05:12	3.66E-05
3 capas(20 y 10)	1431	00:08:79	9.14E-05
2 capas(30)	10364	00:08:23	5.31E-05
2 capas(40)	2076	00:03:31	8.50E-05
2 capas(35)	4871	00:07:15	9.09E-05
2 capas(25)	3383	00:03:07	2.00E-04
2 capas(20)	440	00:01:15	7.62E-04
2 capas(15)	997	00:00:40	8.17E-03
2 capas(10)	1389	00:00:28	7.24E-02

La tabla 4.5 muestra que el cambio en el conjunto de datos disminuye drásticamente el tiempo del entrenamiento y evidentemente el error es bastante menor. Se puede observar que el número de neuronas es determinante en el resultado y que ciertamente su mayor número implica mayor tiempo de procesamiento de los datos, pero, también se observa que no necesariamente la mayor cantidad de neuronas garantiza la reducción del error, ni la reducción en el tiempo de entrenamiento.

4.3.3 Arquitectura del rotor

Igualmente se eligieron las mismas configuraciones que en el estator, con diversas cantidades de capas y neuronas en base a las referencias que han tratado este tipo de problemas. Se hicieron 10 entrenamientos con cada una de las arquitecturas que se muestran en la tabla 4.6 y a continuación se muestra la que mejores resultados generó.

- Arquitectura de tres capas, con 25 neuronas en la primera capa, 10 neuronas en la segunda capa y dos neuronas en la tercera capa. Figs. 4.16 y 4.17

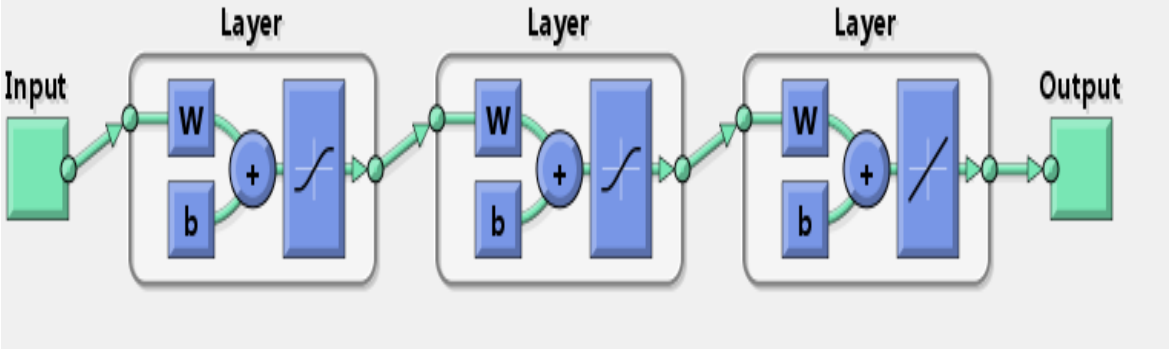


Fig.4.16 Arquitectura de tres capas generada por nntool de MATLAB de la red neuronal que mejores resultados obtuvo para el rotor

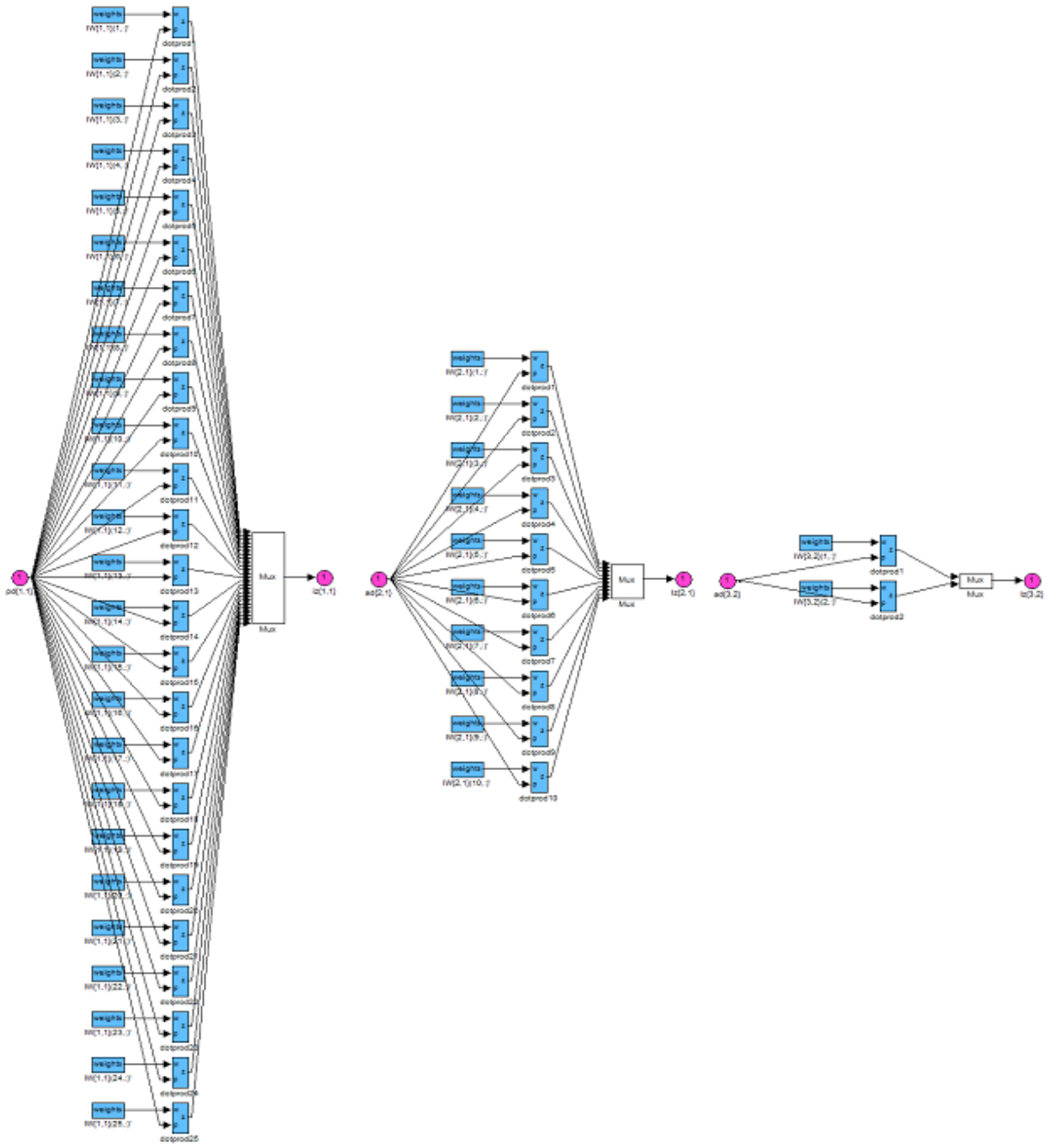


Fig.4.17 Arquitectura de tres capas generada por Simulink de MATLAB de la red neuronal que mejores resultados obtuvo para el rotor

Como lo muestra la Fig. 4.17 la arquitectura de tres capas con 25 neuronas en la primera capa, diez en la segunda capa y dos en la tercer capa es la que logro reducir mas el error mismo que se muestra en la Fig. 4.18.

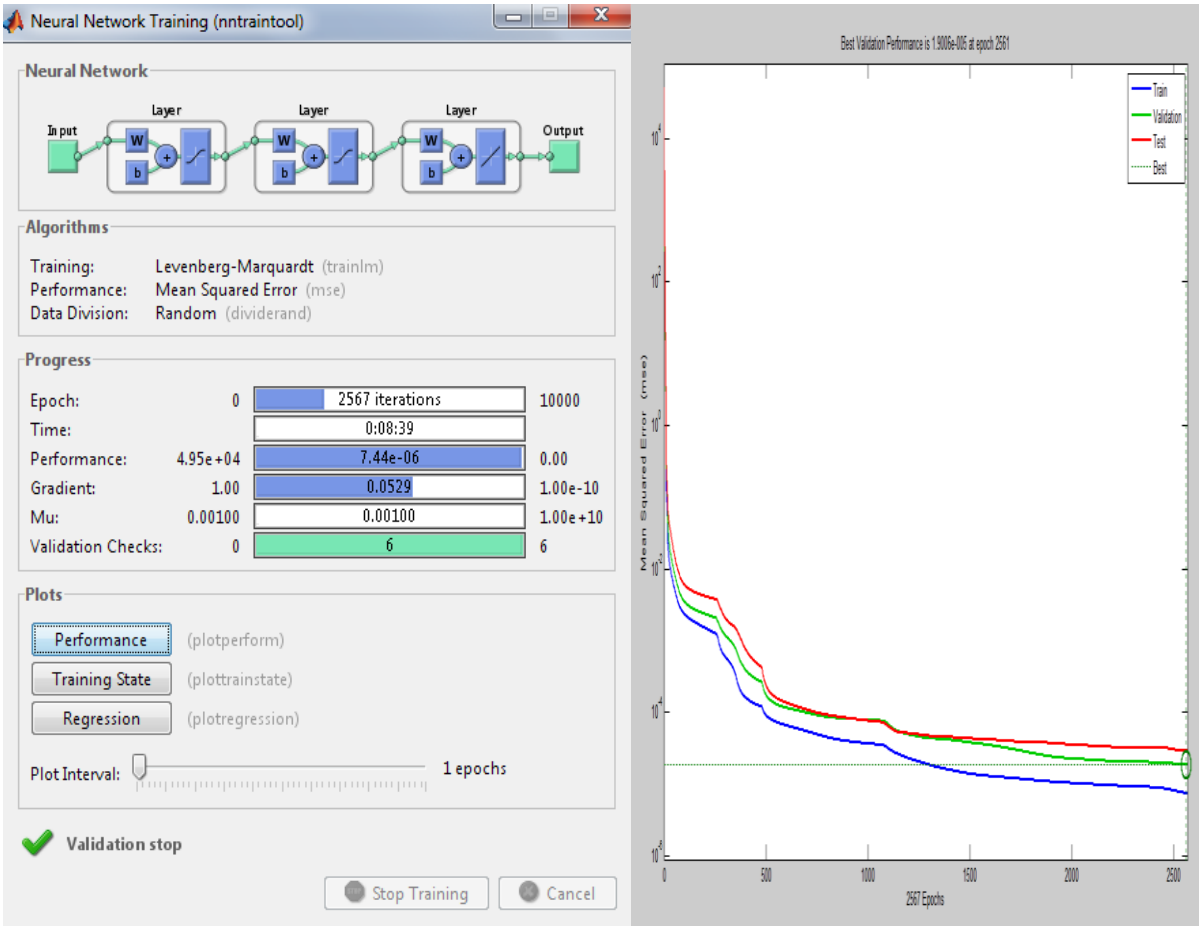


Fig. 4.18 Entrenamiento con mejor rendimiento de la red neuronal para el rotor

La Tabla 4.6 contiene los resultados de las mejores simulaciones de cada arquitectura simulada para el estator. Es importante recordar que la capa de salida consta de dos neuronas por lo que en la tabla ya no aparece.

Tabla 4.6 Entrenamientos de diversas arquitecturas para rotor

ROTOR			
Levenberg-Marquardt	**Iteraciones**	**Tiempo**	**Error**
3 capas(25 y 10)	2567	00:08:39	7.44E-06
3 capas(35 y 20)	517	00:04:19	8.01E-06
3 capas(35 y 10)	1376	00:08:36	9.20E-06
3 capas(35 y 5)	2833	00:08:08	1.09E-05
3 capas(25 y 5)	2289	00:04:24	1.10E-05
3 capas(30 y 10)	1326	00:03:55	1.18E-05
3 capas(20 y 5)	1825	00:02:15	1.36E-05
3 capas(20 y 10)	2541	00:04:35	1.44E-05
2 capas(30)	10000	00:12:27	3.39E-05
2 capas(40)	2076	00:03:31	8.50E-05
2 capas(35)	4871	00:07:15	9.09E-05
2 capas(25)	3383	00:04:07	1.00E-04
2 capas(20)	4000	00:03:48	1.62E-04
2 capas(15)	1997	00:01:40	2.17E-03
2 capas(10)	789	00:00:35	1.14E-02

La tabla 4.6 muestra claramente que al reducirse el conjunto de datos para el rotor, respecto al estator, el tiempo de entrenamiento se disminuye e igualmente el número de iteraciones en la mayoría de las configuraciones. También se nota que no siempre el número de neuronas es determinante en la reducción del error.

4.4 Entrenamiento de la red neuronal para el caso de los nodos internos

Como se vio en la sección 3.5, se llevo a cabo también una prueba para dos nodos internos en el estator y para dos nodos en el rotor.

También en las secciones anteriores de este capítulo se seleccionó el método de Levenberg-Marquardt como algoritmo de optimización y la arquitectura de 3 capas con 25, 10 y 2 neuronas respectivamente como arquitectura que mejores resultados dio.

Entonces para validar la metodología se entreno una red con el mismo conjunto de datos de 4.3.1 para conocer el comportamiento térmico en nodos internos del estator y rotor alcanzando los siguientes entrenamientos.

4.4.1 Para el caso del estator

Se realizó 10 veces el entrenamiento del conjunto. La Fig. 4.19 muestra el mejor obtenido para el caso del estator.

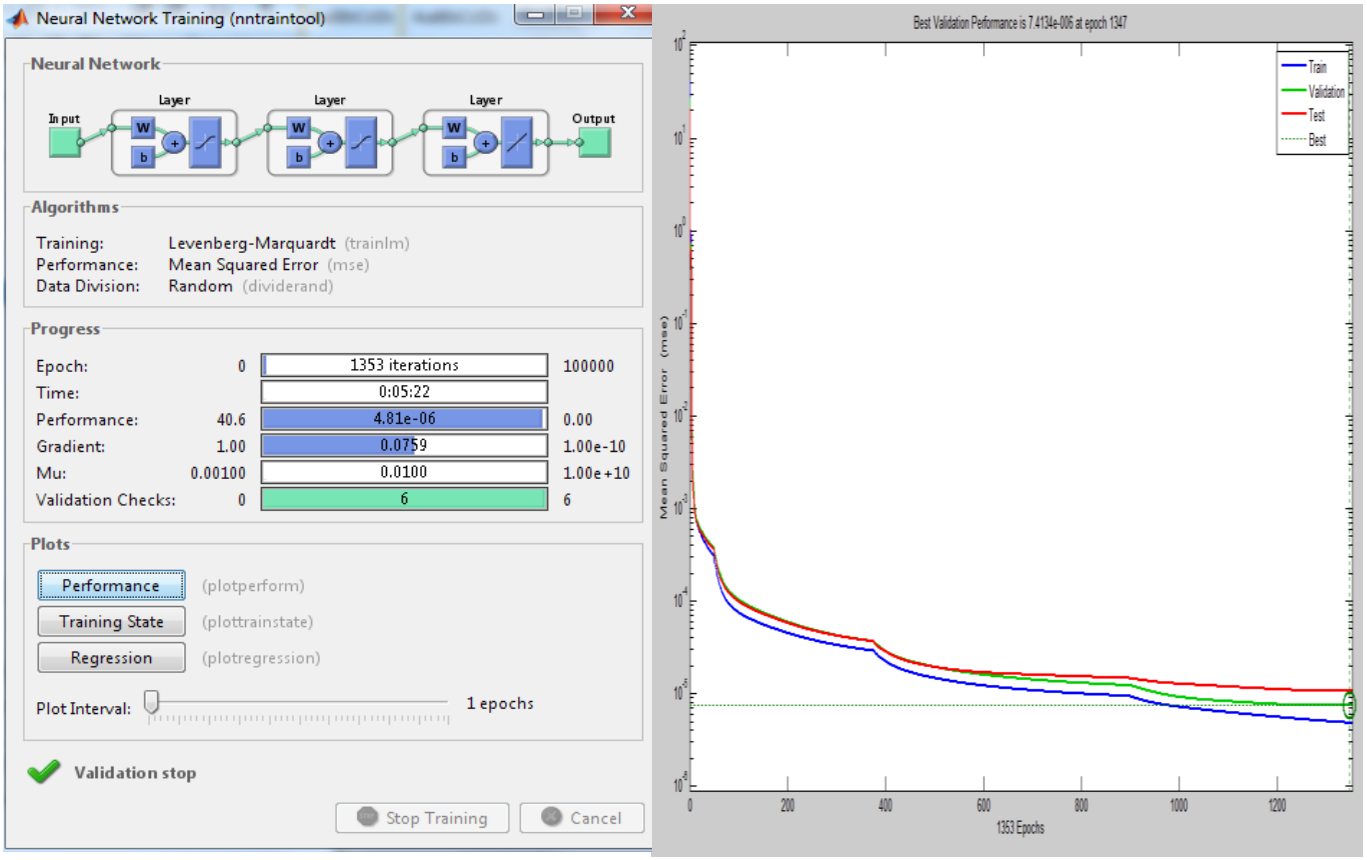


Fig. 4.19 Entrenamiento con mejor rendimiento de la red neuronal para el estator en nodos internos

4.4.2 Para el caso del rotor

Se realizó 10 veces el entrenamiento del conjunto. La Fig. 4.20 muestra el mejor obtenido para el caso del rotor.

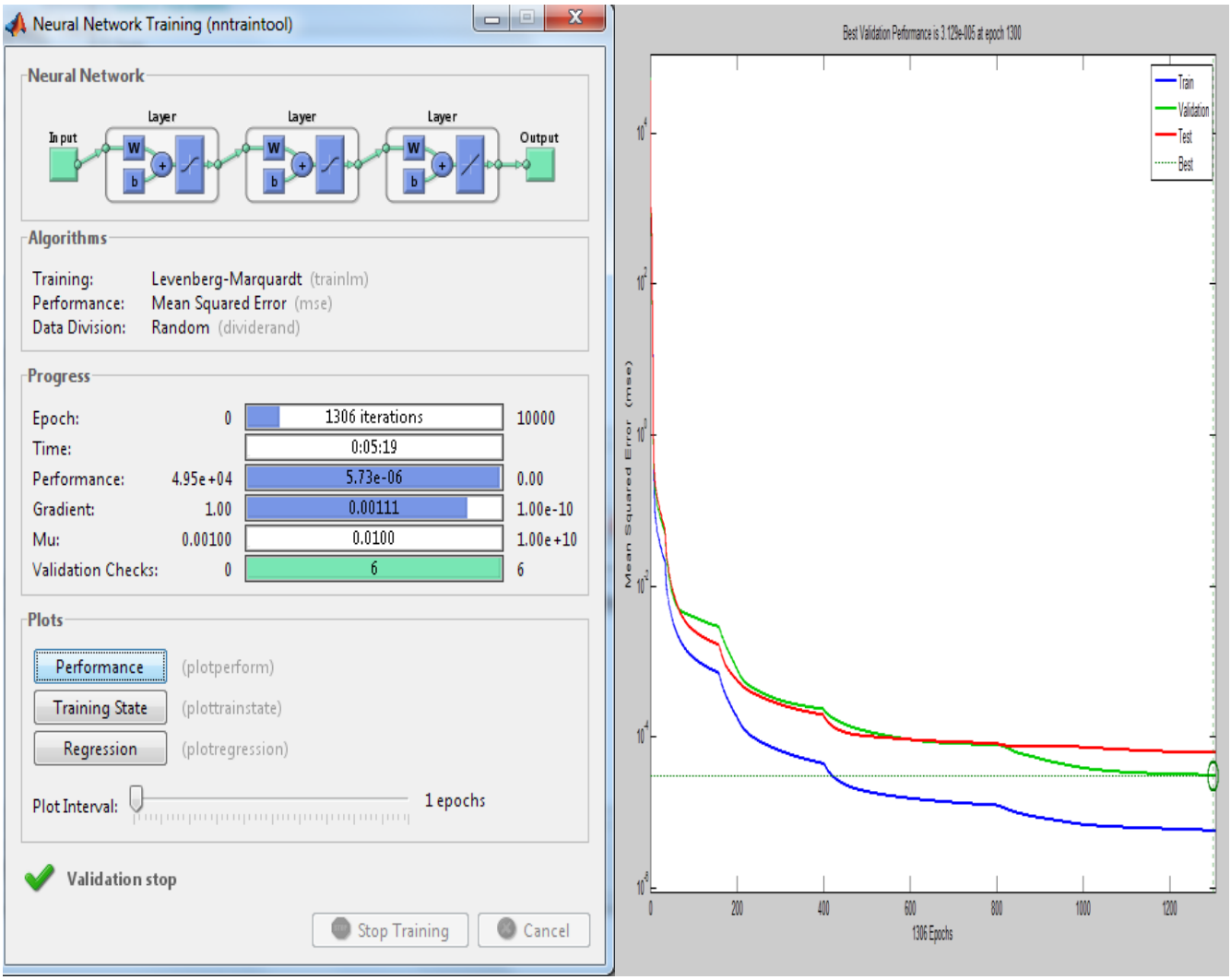


Fig. 4.20 Entrenamiento con mejor rendimiento de la red para el rotor en nodos internos

4.5 Verificación de la red neuronal entrenada

Como se vio en el capítulo 3, fue necesario llevar a cabo un proceso de validación de la red neuronal entrenada y obtener un error relativo.

Para este caso se llevo a cabo con el siguiente conjunto de datos:

Q_{de} = [0.6x10^6 0.8x10^6 0.95x10^{6}8] W/m^3;
Q_{cme} = [1.1x10^5 1.4 x10^5 1.75 x10^5] W/m^3;
α_{de} = [70 125 200 300] W / (m^2 K);
α_{cme} = [70 125 200 300] W / (m^2 K);
Q_{dr} = [0.6x10^6 0.8x10^6 0.95x10^{6}8] W/m^3;
Q_{cmr} = [1.1x10^5 1.4 x10^5 1.75 x10^5] W/m^3;
α_{dr} = [70 125 200 300] W / (m^2 K);

Los tiempos t de medición de temperaturas quedan iguales porque estos valores dependen solamente del usuario.

El proceso de verificación de la red neuronal se incluyó en el programa desarrollado por lo que automáticamente da el error máximo en todo el conjunto de datos como resultado del proceso de validación de la red neuronal.

4.5.1 Verificación de los nodos externos

En la Tabla 4.7 se observa el porcentaje de error que tiene la red neuronal respecto a la salida deseada (matriz Tm del motor virtual). T1 y T2 representan las temperaturas censadas en los nodos seleccionados del estator. T3 y T4 son las temperaturas censadas en los nodos seleccionados del rotor.

Tabla 4.7 % de errores en verificación y error máximo en nodos externos

T	1	2	3	% Error Max
T1	0.2070	1.1700	2.1658	2.5983
T2	0.6956	0.1459	0.1357	0.9228
T3	0.0253	0.2443	0.5354	1.0556
T4	0.0275	0.2456	0.5315	1.0572

En las Figs. 4.21 se observa el error máximo en el estator y en la Fig. 4.22 el error correspondiente al rotor. En este caso se muestra que el error es mayor al principio y después se reduce ampliamente, esto es porque los primeros datos son los del inicio del arranque y por lo tanto son los mas imprecisos, sin embargo, cuando avanza la verificación se va eliminando el error.

La exactitud depende en gran medida de la parte que se esté verificando del conjunto de datos, pues los datos de arranque arrojan mayores errores.

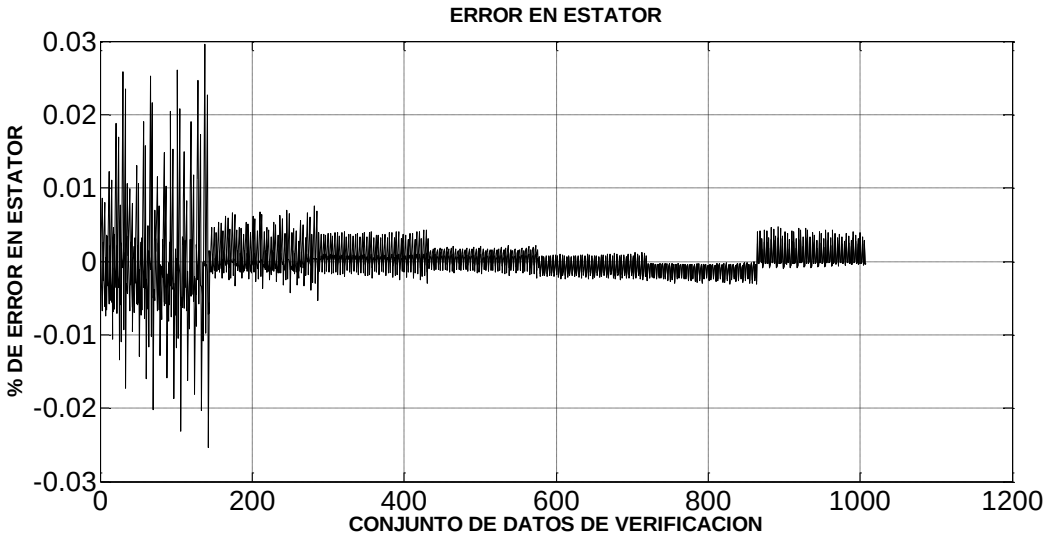


Fig. 4.21 Errores en la determinación de temperaturas del estator en nodos externos

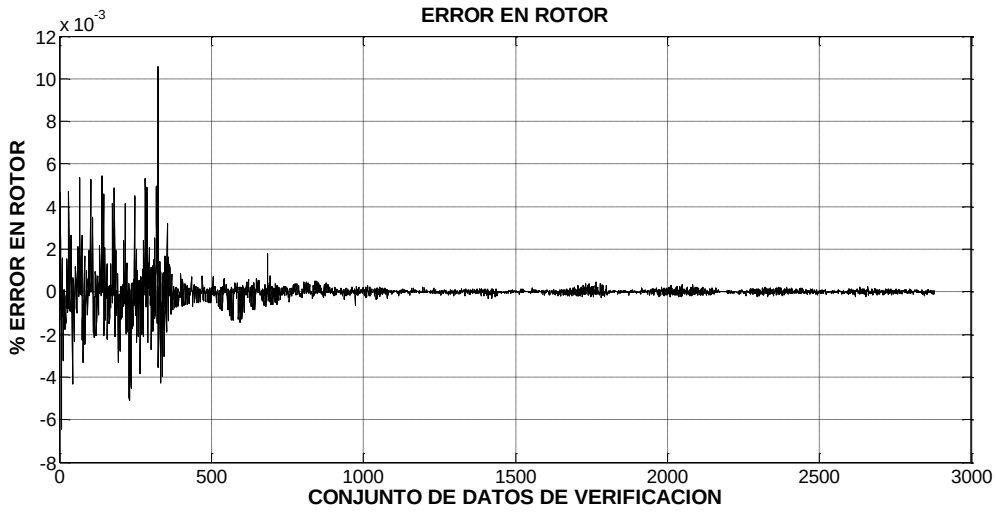


Fig. 4.22 Errores en la determinación de temperaturas del rotor en nodos externos

4.5.2 Verificación de los nodos internos

En la Tabla 4.8 se observa el porcentaje de error que tiene la red neuronal respecto a la salida deseada (matriz Tm del motor virtual). T1 y T2 representan las temperaturas censadas en los nodos seleccionados del estator. T3 y T4 son las temperaturas censadas en los nodos seleccionados del rotor.

Tabla 4.8 % de errores en verificación y error máximo en nodos internos

T	1	2	3	% Error Max
T1	0.4520	1.4580	2.9812	3.0133
T2	0.3443	0.9959	1.3452	2.8918
T3	0.8992	1.1039	0.9823	1.5780
T4	0.4536	0.3453	1.0023	1.4901

En las Figs. 4.23 se observa el error máximo en el estator y en la Fig. 4.24 el error correspondiente al rotor. En este caso se muestra que el error es mayor al principio y después se reduce ampliamente, esto es porque los primeros datos son los del inicio del arranque y por lo tanto son los mas imprecisos, sin embargo, cuando avanza la verificación se va eliminando el error.

La exactitud depende en gran medida de la parte que se esté verificando del conjunto de datos, pues los datos de arranque arrojan mayores errores.

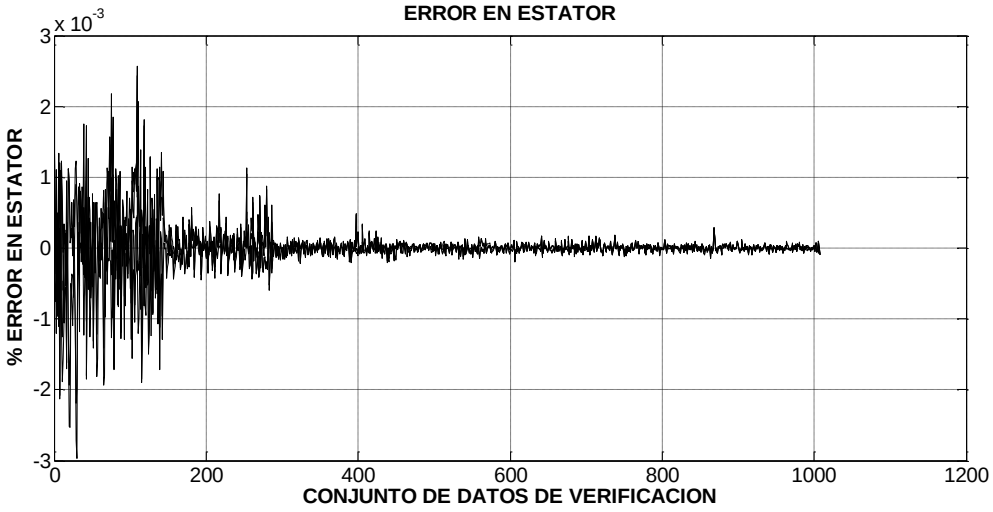


Fig. 4.23 Errores en la determinación de temperaturas del estator en nodos internos

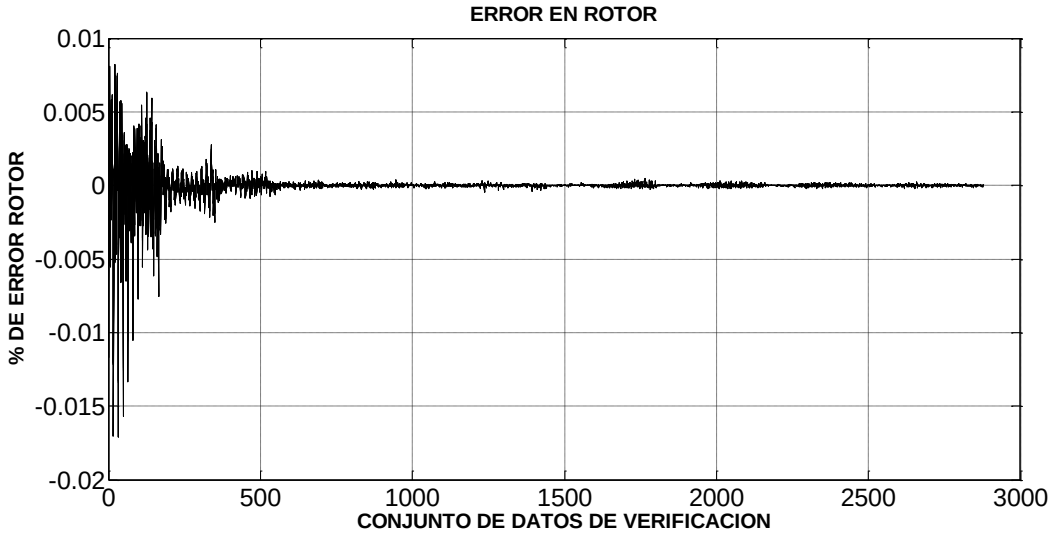


Fig. 4.24 Errores en la determinación de temperaturas del rotor en nodos internos

En esta investigación se consideraron varios conjuntos de entrenamiento de tamaños diversos, generados con el motor virtual en elemento finito y se simularon con la función de entrenamiento y la arquitectura que ya se describieron anteriormente obteniendo resultados muy similares a los anteriores.

El objetivo principal de este trabajo es plantear una metodología para simular campos de temperatura en máquinas eléctricas rotatorias y con el procedimiento anterior queda comprendida la secuencia por lo que esta de mas colocar los datos de cada simulación.

CAPÍTULO 5

CONCLUSIONES Y SUGERENCIAS

5.1 Introducción

En este capítulo de la tesis se exponen las conclusiones y las contribuciones de la metodología propuesta: “Modelo Térmico con una Red Neuronal de Retropropagación de una Máquina Eléctrica Rotatoria” y se aportan sugerencias para trabajos futuros.

5.2 Conclusiones

- Al realizar este trabajo se pudo concluir que el método presentado permite conocer las temperaturas al interior de la maquina eléctrica de forma rápida, simplemente basta con tener los datos de entrada para poder determinar las temperaturas con una buena exactitud.

- La metodología propuesta en este trabajo puede ser útil para los diseñadores de máquinas eléctricas rotarias, ya que se pueden encontrar los coeficientes de transferencia de calor óptimos para las diferentes partes de la máquina eléctrica, considerando distintas condiciones térmicas de operación a las que pueda estar sometida la máquina eléctrica rotatoria.

- La estructura de red neuronal propuesta por este trabajo puede ser apropiada para cualquier tipo de motor, ya sea de alta o baja potencia sin importar su dimensión, al solo modificar la geometría del devanado y el circuito magnético del estator y rotor de la máquina eléctrica rotatoria que se requiera estudiar y realizar el entrenamiento correspondiente.

- Se concluye que el método de Levenberg-Marquardt es el algoritmo de optimización que mejor reduce el error de entrenamiento en este tipo de problemas con redes de retropropagación como fue el caso de esta investigación. Sin embargo esto no significa

que los demás algoritmos no puedan ser empleados bajo alguna estrategia de sintonización que permita ajustar las curvas obteniendo buenos resultados.

- El uso de la red neuronal ya entrenada representa una gran ventaja comparada con el MEF puesto que la velocidad de la respuesta de la red neuronal es casi instantánea y la del MEF puede tardar mucho más.

- El motor virtual representa una herramienta poderosa para simular las temperaturas en las máquinas eléctricas, incluso pudiera ser utilizada por los diseñadores para facilitar el estudio térmico de las máquinas eléctricas.

- La solución del modelo 2D+1 de transferencia de calor mediante la herramienta "Toolbox PDEtool" del paquete computacional MATLAB facilita en gran manera la obtención del conjunto de entrenamiento de la red neuronal.

- Este trabajo sienta las bases en el uso de redes neuronales en el modelado térmico de máquinas eléctricas rotatorias por lo que es importante intentar darle continuidad a estos trabajos, ya que como se menciono anteriormente, las redes neuronales son una poderosa herramienta que puede aplicarse en todas las áreas para resolver problemas.

- Un conjunto de datos muy grande hace que el entrenamiento se haga lento, al reducir los datos se agiliza el proceso, sin embargo, en esta investigación lo relevante fue disminuir el error, sin considerar el tiempo de entrenamiento.

- Se concluye que la red de tres capas con 25 neuronas en la primera capa, 10 neuronas en la segunda capa y 2 neuronas en la tercera capa fue la arquitectura que dio los mejores resultados para estator y para rotor. Esto implica que un mayor número de neuronas no necesariamente garantiza disminuir el error ni el tiempo en el entrenamiento, por ello se deben hacer diversos entrenamientos para obtener la arquitectura que mejor disminuye el error.

- También se deduce que entre mas puntos se quiera abarcar, mas lento será el proceso de entrenamiento puesto que al ser muchos puntos es más difícil lograr ajustar la curva.

- Dado que hay pocas investigaciones de redes neuronales aplicadas a determinar campos de temperaturas en máquinas eléctricas, el número de capas y neuronas se determinó de forma heurística.

- Se concluye que es posible determinar las temperaturas en cualquier punto de la máquina puesto que se logro obtener las temperaturas en puntos internos del estator y rotor de la máquina estudiada.

- En el proceso de verificación, el error que se considera ya es el de simulación y no tiene que ver con el error del entrenamiento, pero finalmente resulta mucho más rápido y sencillo el uso de la herramienta en redes neuronales que el MEF.

- Para simular el comportamiento térmico de la máquina rotatoria analizada, considerando diferentes valores de coeficientes de transferencia de calor por convección, fuentes de calor y tiempos se tomó un motor virtual en elementos finitos desarrollado en [1], el cual es capaz de recabar, procesar y almacenar información generada del modelo. A este motor virtual se le hicieron algunas adaptaciones para obtener los conjuntos de entrenamiento de la red neuronal que se propone.

- Se elaboró una metodología para simular campos de temperaturas en motores eléctricos con redes neuronales de retropropagación. Este procedimiento considera la elección de la mejor arquitectura, el número de neuronas, el método de optimización y la selección del algoritmo de entrenamiento.

- Se desarrollaron programas con el lenguaje de programación de MATLAB con base en las metodologías propuestas para la presente investigación. En primera instancia se obtienen los datos de la geometría del motor y datos de diseño de la máquina a estudiar, después se ejecuta el motor virtual para obtener los conjuntos de entrenamiento de la red neuronal,

posteriormente se entrena la red neuronal hasta lograr reducir el error de entrenamiento y finalmente se simulan las temperaturas en los nodos seleccionados.

5.3 Sugerencias para futuros trabajos

- Se recomienda trabajar con modelos en tres dimensiones, para el análisis de campos de temperaturas generados por pérdidas de energía y ver las complicaciones o ventajas que puedan generarse al tratar de aplicar la metodología de redes neuronales.

- Se sugiere validar la metodología generando una geometría drásticamente distinta y simulando el comportamiento térmico bajo distintas condiciones de operación.

- Se sugiere desarrollar la metodología propuesta con una herramienta de redes neuronales distinta al paquete computacional MATLAB con el propósito de comparar los resultados obtenidos y validar la metodología, o en su defecto proponer los cambios que se consideren necesarios.

- Se recomienda hacer ajustes a cada método de optimización para ver el comportamiento de cada método y ver si alguno da mejores resultados que el de Levenberg-Marquardt en tiempo y en la disminución del error.

- Se recomienda implementar la metodología para trabajo discontinuo y ver la respuesta que tienen las redes neuronales.

REFERENCIAS

[1] **Juárez Balderas Edgar Alfredo**, *Control Optimo del Sistema de Enfriamiento de Motores de Inducción,* Tesis de grado, SEPI/ESIME/IPN, 2009.

[2] **Chapman Sthepen J**., *Máquinas Eléctricas*, Ed. Mc. Graw Hill, Tercera edición, 2000.

[3] **Engelmann Richard H. y Middendorf, William H.,** *Handbook of electric motors,* Ed. Dekker, 1995.

[4] **Cortés Cherta Manuel,** *Curso moderno de máquinas eléctricas rotativas, Tomo 1, La máquina eléctrica en general,* Ed. Editores técnicos asociados, 1970.

[5] **Sawhney A. K.,** *A course in electrical machine design*, Ed. Dhanpat Rai & Sons, 1984.

[6] **Hayt William Jr.,** *Teoría Electromagnética*, Ed. Mc. Graw Hill, 1991.

[7] **Strages N. y Dymond J. H.**, *How design influences the temperature rise of motors on inverter drives,* Energy Conversion, IEEE Transaction,Nov. 2003, Vol. 39, IEEE, 2003, pp.1584-1591.

[8] **Mugglestone J. , Pickering S.J. y Lampard, D.,** *Effect of geometric changes on the flow heat transfer in the end region of TEFC induction motor,* University of Nottingham, U.K., IEE, 1999.

[9] **Glen C.N.,** *Stray load losses in induction motors; a challenge to academia,* U.K., 2000.

[10] **Smith A.C. Edey K.,** *Influence of manufacturing processes on iron losses,* Electrical machines and drives, 1995. Seventh International Conference, 1995, pp. 77-81.

[11] **Guyer Erick C.**, *Handbook of applied thermal design,* Ed. Mc. Graw Hill, 2000.

[12] **Chapra Steven C. y Raymond P. Canale,** *Métodos numéricos para ingenieros*, Ed. Mc. Graw Hll, Cuarta edición, 2003.

[13] **Siemens Energy,** *Soluciones en eficiencia energética*, Siemens España, 2010

[14] **Cortés Camilo, Deprez Wim, Driesen Johan y Pérez Jhon,** *Determining electrical loss in electromagnetically-modelled induction motors using the finite element method,* Revista Ingeniería e Investigación, vol. 28, No. 3, Diciembre 2008, pp. 64-74.

[15] **A.V. Ivanov-Smolenki,** *Máquinas Eléctricas*, Tomo 2, Editorial Mir. Moscú, 1980

[16] **J.M. Smith y H.C. Van Ness,** *Introducción a la Termodinámica en Ingeniería Química,* Ed. Mc. Graw Hill, 1980.

[17] **Pérez Cruz Justo R.,** *La Termodinámica de Carnot a Clausius,* Universidad de la Laguna, 2008.

[18] **Ortega Herrera José A. y Hernández Gómez, Luis H.,** *Análisis del Método del Elemento Finito y sus Aplicaciones a la Ingeniería*, Vol. 6. Serie en Ciencias e Ingeniería. Editorial. Biblioteca SEPI ESIME Zacatenco. IPN. Novena Impresión

[19] **Mellor P.H., Roberts D. y Turner D.R.**, *Lumped Parameter Thermal Model for Electrical Machines of TEFC Design,* IEE Proceedings-B, Vol.138, No.5, September 1991.

[20] **Driesen Johan, Belmans Ronnie J.M. y Hameyer Kay.,** *Finite-Element Modeling of Thermal Contact Resistances and Insulation Layers in Electrical Machines,* IEEE Transactions on Industry Applications, Vol. 37, No.1, January-February 2001.

[21] **Bastos J.P., Cabreira M., Sadowski N. y Arruda S.**, *A Thermal Analysis of Induction Motors Using Weak Coupled Modeling,* IEEE, Transactions on Magnetics, Vol.33, No.2, March 1997.

[22] **Minsky M. y Papert S.**, *Perceptrons*, MIT Press, Cambridge, MA, 1969.

[23] **McCulloch W. S. y Pitts W. A.**, *A logical calculus of the ideas immanent in nervous activity*, Bulletin of Mathematical Biophysics 5, 1943.

[24] **Matich Damian Jorge**, *Redes Neuronales: Conceptos Básicos y Aplicaciones,* Universidad Tecnologica Nacional de Argentina, Marzo 2001.

[25] **Rosenblatt F.,** *The Perceptron: A probabilistic model for information storage and organization in the brain*, Psychological Review 65, 1958.

[26] **Widrow B.**, *An Adaptive Adaline Neuron Using Chemical Memistors*, Stanford Electronics Laboratories Technical Report 1553-2, October 1960.

[27] **Carpenter Gail A. y Grossberg Stephen**, *Adaptative Resonance Theory*, MIT Press, 1998.

[28] **Kohonen Teuvo**, *The Self-Organizing Map*, Springer, 1982.

[29] **Werbos Paul J.,** *Backpropagation Through Time: What It Does and How to Do IT* Proceedings of the IEEE, Vol. 78, No. 10. (1990), pp. 1550-1560.

[30] **Ponce Cruz, Pedro,** *Inteligencia Artificial con Aplicaciones a la Ingeniería,* Ed. Alfaomega, Julio 2010.

[31] **Kumpati S., Narendra y Kannan, Parthasarathy**, *Identification and Control of Dynamical Systems Using Neural Networks*, IEEE Transactions on Neural Networks, Vol.1, No.1, March 1990.

[32] **Kumpati S., Narendra y Kannan, Parthasarathy,** *Gradient Methods for the Optimization of Dynamical Systems Containing Neural Networks,* IEEE Transactions on Neural Networks, Vol.2,No.2, March 1991.

[33] **Polycarpou, Marios y Ioannou, Petros.,** *Modelling, Identification and Stable Adaptative Control of Continuous-Time Nonlinear Dynamical Systems Using Neudal Networks,* University of Southern California, 1992 ACC/WA2.

[34] **Penman J. y Yin C.M.**, *Feasibility of using unsupervised learning, artificial neural network for the condition monitoring of electrical machines*, IEE, Proceedings-Electr., Power Applications, Vol.141, No.6, November 1994.

[35] **Wishart, Michael T. y Harley, Ronald G.,** *Identification and Control of Induction Machines Using Artificial Neural Networks,* IEEE, Transactions on Industry Applications, Vol.31, No.3, May-June 1995.

[36] **Hammer, M.; Tozlovsky, T. y Svoboda J.**, *The Use of Neural Networks for the Life Prediction of Insulating Material of Electric Rotating Machines,* International Conference on Solid Dielectrics, Toulouse, France, July 5-9, 2004.

[37] **Cordero Ramírez, Jorge Jefrey**, *Optimización del Sistema de Enfriamiento Externo de Motores de Inducción con Trabajo Discontinuo*, Tesis de grado, SEPI/ESIME/IPN, 2011.

[38] **Yudiche Barbosa David de Jesus**, *Método Termométrico para Determinación de Pérdidas Electromagnéticas en Motores de Inducción*, Tesis de grado, SEPI/ESIME/IPN, 2011.

[39] **Niewierowicz T., Kawecki L. y Nepieralska E.**, *Determination of Electromagnetic Losses in Electrical Motors Applying Neural Networks,* IEEE Latin America Transactions, vol.9, No.5, September 2011.

[40] **Niewierowicz T., Kawecki L. y Nepieralska E.**, *Análisis de Errores en la Determinación de Pérdidas Generadas en Máquinas Eléctricas Rotatorias*, Memorias, IEEE Sección México, RVP-AI/2006, Acapulco Gro., México, pp. 1-6 (GEN-02), (2006).

[41] **Niewierowicz T**., *Simulador Neuronal de Temperaturas Generadas por Pérdidas Electromagnéticas en Motores Eléctricos*, Memorias, IEEE Sección México, RVP-AI/2007, Acapulco Gro., México, pp. 1-6 (AI-07), (2007).

[42] **Staton, David A. y Cavagnino, Andrea**, *Convection Heat Transfer and Flow Calculations Suitable for Electric Machines Thermal Models,* IEEE, Transactions on Industrial Electronics, Vol.55, No.10, October 2008.

[43] **Boldea Ion and Nasar Syed A.,** *The Induction Machine Handbook,* CRC Press, 2002.

[44] **S/A,** *Mecanismos de Transferencia de Calor,* www.dgeo.**udec**.cl/~juaninzunza/docencia/fisica/cap14.pdf, S/F.

[45] **Acosta, María Isabel; Salazar, Harold y Zuluaga, Camilo,** *Tutorial de Redes Neuronales,* Universidad Tecnologica de Pereira, 2000.

[46] **Ruiz Carlos Alberto y Basualdo Marta Susana,** Redes Neuronales.: Conceptos Basicos y Aplicaciones, UTN Argentina, 2001.

[47] **Zurada, Jacek M.**, *Introduction to Artificial Neural Systems,* West Publishing Company, 1992

[48] **Gupta Madan, M., Ji, Liang and Homma, Noriyasu**, *Static and Dynamic Neural Networks,* Wiley- Interscience, 2003

[49] **Freeman, James A. y Skapura, David M.,** *Neural Networks, Algorithms, Applications and Programming Techniques,* Addison Wesley Publishing Company, 1991.

[50] **Luenberger, David G. y Ye, Yinyu**, *Linear and Nonlinear Programming,* Springer, 2008

[51] **Demuth H. y Beale M.**, MATLAB User's Guide, Neural Network Toolbox, The MathWorks, 2009.

[52] **Haykin Simon**, *Neural Networks: A Comprehensive Foundation,*Prentice Hall, 1999.

[53] **Torres Sánchez Javier**, *Determinación de la Distribución de Temperatura en Motores de Inducción Jaula de Ardilla Utilizando Métodos Experimentales*, Tesis de grado, SEPI/ESIME/IPN, 2007.

[54] **MATLAB, Handle Graphics, and Real Time Workshop**, *Partial Differential Equations toolbox User´s Guide*, Copyright 1984-1997 by the Math Works, Inc.

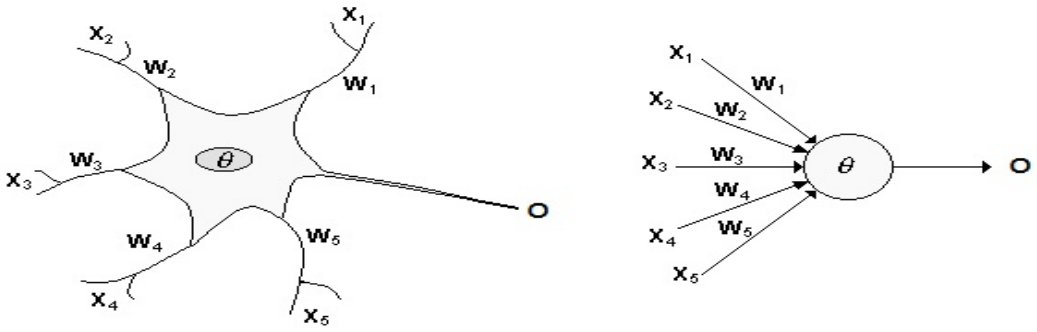
APÉNDICE A

TEORIA DE REDES NEURONALES

Definición de una red neuronal artificial

Las RNA se definen como sistemas de mapeos no lineales cuya estructura se basa en principios observados en los sistemas nerviosos humanos y animales. Constan de un número grande de procesadores simples ligados por conexiones con pesos. Las unidades de procesamiento se denominan neuronas. Cada unidad recibe entradas de otros nodos y genera una salida escalar simple que depende de la información local disponible guardada internamente o que llega a través de las conexiones con pesos. Pueden realizarse muchas funciones complejas dependiendo de las conexiones.

En la Fig. A.1 se muestra la analogía de la neurona biológica con la neurona artificial.



a) **b)**

Fig. A.1 a) Neurona biología, b) Neurona artificial

De la observación detallada del proceso biológico se han hallado las siguientes analogías con el sistema artificial:

- Las entradas X_i representan las señales que provienen de otras neuronas y que son capturadas por las dendritas.

- Los pesos W_i son la intensidad de la sinapsis que conecta dos neuronas; tanto X_i como W_i son valores reales.
- θ es la función umbral que la neurona debe sobrepasar para activarse; este proceso ocurre biológicamente en el cuerpo de la célula.
- O es la salida.

Elementos que componen una red neuronal artificial

Las redes neuronales están constituidas por neuronas interconectadas y arregladas en capas. Los datos ingresan por medio de la "*capa de entrada*", pasan a través de la "*capa oculta*" y salen por la "*capa de salida*". Cabe mencionar que la capa oculta puede estar constituida por varias capas como lo podemos ver en la Fig. A.2.

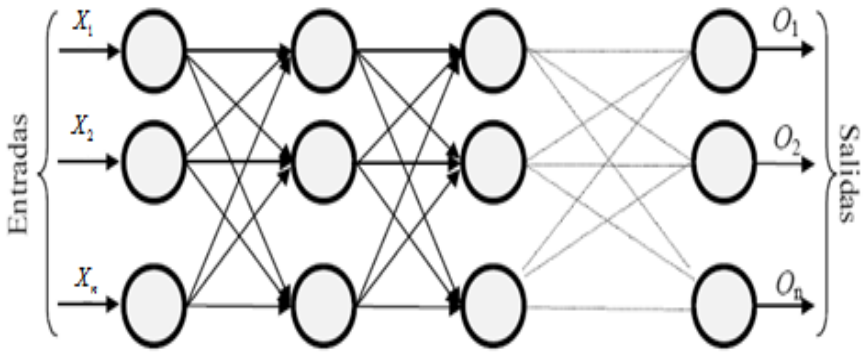


Fig. A.2 Estructura de una red neuronal multicapa.

La neurona

Cada modelo de neurona consta de un elemento de procesamiento con conexiones sinápticas de entrada y una salida simple. El flujo de las señales de entrada de las neuronas se considera unidireccional. El símbolo general de una neurona se muestra en la Fig. A.3. En esta representación se muestra un conjunto de pesos y el nodo.

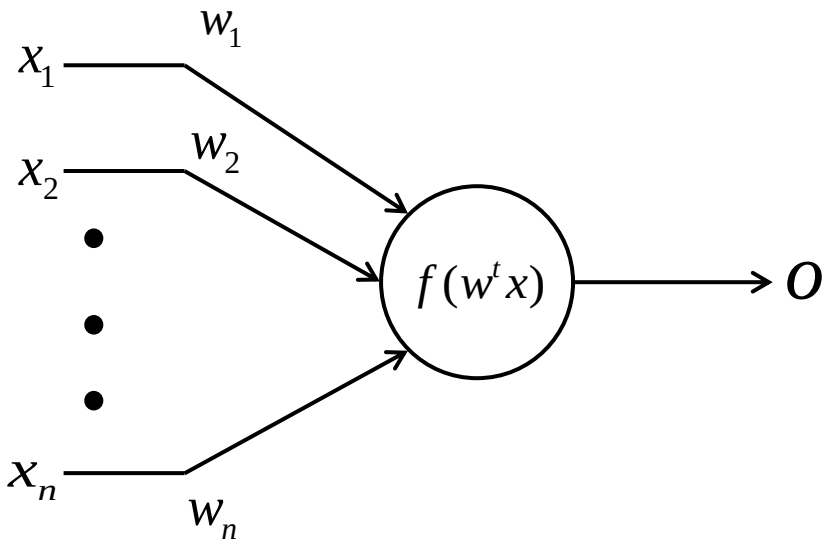


Fig. A.3 Estructura general de una neurona

La señal de salida de la neurona esta dada por la siguiente relación:

$$o = f(w^t x) \tag{A.1a}$$

O bien,

$$o = f\left(\sum_{i=1}^{n} w_i x_i\right) \tag{A.1b}$$

Donde w es el vector de pesos definido como:

$$w = \left[w_1\ w_2 \cdots w_n\right]^t$$

Y x es el vector de entradas:

$$x = \left[x_1\ x_2 \cdots x_n\right]^t$$

La función de activación

La función $f(w^t x)$ se refiere a una función de activación. La variable net se define como el producto escalar de los vectores de pesos y entradas:

$$net = w^t x \tag{A.2}$$

La estructura de neurona mostrada en la Fig. A.3 y descrita por las ecuaciones (A.1) y (A.2) es la mas conocida en la literatura de redes neuronales, sin embargo, diferentes clases de redes neuronales artificiales hacen uso de diferentes definiciones de $f(net)$.

Se nota en (A.1) que la neurona es como un nodo de procesamiento que realiza la operación de suma de sus entradas, o el producto escalar para obtener net.

Posteriormente, se realiza la operación no lineal $f(net)$ a través de su función de activación. Las funciones típicas de activación utilizadas son:

$$f(net) = \frac{2}{1+e^{(-\lambda net)}} - 1 \tag{A.3a}$$

Y también,

$$f(net) = \operatorname{sgn}(net) = \begin{cases} +1, net > 0 \\ -1, net < 0 \end{cases} \tag{A.3b}$$

Donde $\lambda > 0$ en (A.3a) es proporcional a la ganancia de la neurona determinado la pendiente de la función continua $f(net)$ cercana a $net = 0$. La función de activación continua es mostrada en la Fig. A.4 para varios λ. Se puede notar que mientras $\lambda \to \infty$, el limite de la función continua se convierte en la $\operatorname{sgn}(net)$, la cual esta definida en (A.3b).

Estas funciones de activación son llamadas, función continua bipolar (A.3a) y función binaria bipolar (A.3b). La palabra "bipolar" se utiliza para indicar que las respuestas positivas y negativas de las neuronas son producidas por esta definición de la función de activación.

Al desplazar y ampliar las funciones de activación bipolares definidas por (A.3), la función de activación continua unipolar y la función de activación binaria unipolar pueden ser obtenidas respectivamente de la siguiente manera:

$$f(net) = \frac{1}{1+e^{(-\lambda net)}} \tag{A.4a}$$

Y también,

$$f(net) = \begin{cases} 1, net > 0 \\ 0, net < 0 \end{cases} \quad \text{(A.4b)}$$

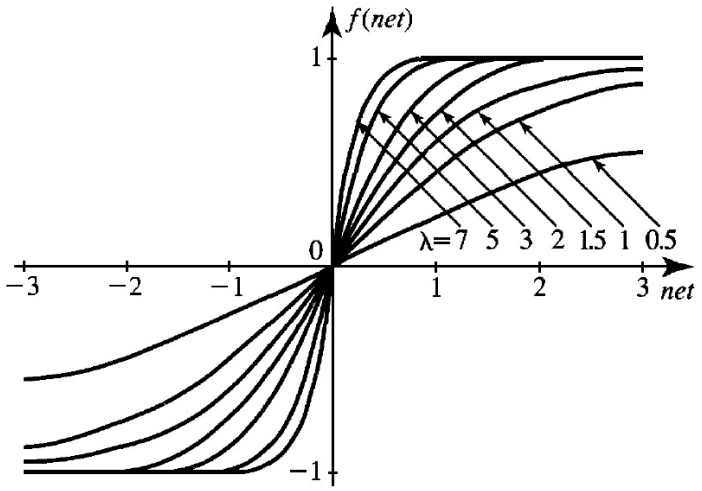


Fig. A.4 Función de activación continua bipolar

La función (A.4a) se muestra en la Fig. A.5. Una vez mas, la función binaria unipolar es el limite de la $f(net)$ en (A.4a) cuando $\lambda \to \infty$. A las funciones de activación de limitación suave (A.3a) y (A.4a) se les conoce como características sigmoidales, en contraparte de las funciones de activación de limitación dura dadas en (A.3b) y (A.4b). Las funciones de activación de limitación dura describen el modelo de neuronas discreto.

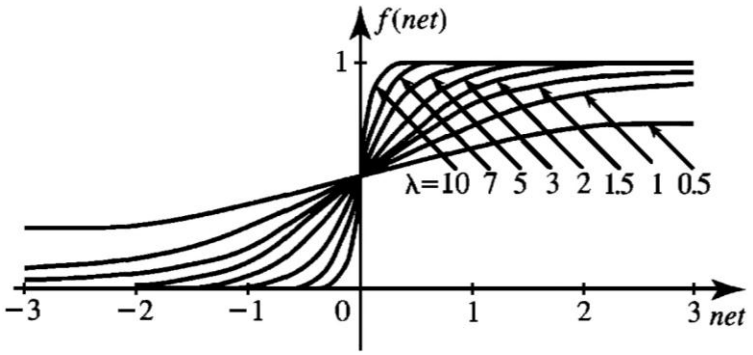


Fig. A.5 Función de activación continua unipolar

La mayoría de las neuronas emplean funciones de activación bipolares. Algunas arquitecturas de redes neuronales o aplicaciones, sin embargo, requieren específicamente las respuestas de neuronas unipolares. Esencialmente, cualquier función $f(net)$ que es monótonamente creciente y continua tal que $net \in R$ y $f(net) \in (-1,1)$ se puede utilizar en lugar de (A.3a) para el modelado neuronal. Algunos modelos neuronales que a menudo implican alguna forma de realimentación requieren el uso de otro tipo de no linealidad que la definida en (A,3) y (A,4).

Un ejemplo de función de activación rampa unipolar se muestra en la Fig. A.6.

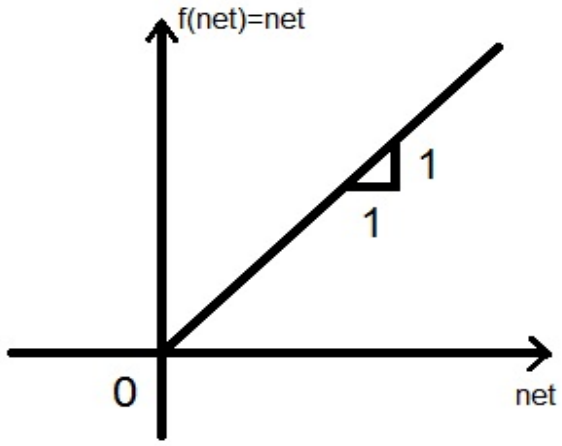


Fig. A.6 Función de activación rampa unipolar

Si la función de activación de la neurona tiene la forma binaria bipolar de (A.3b), el símbolo de la Fig. A.3 puede ser sustituido por el diagrama mostrado en la Fig. A.7, que en realidad es un diagrama a bloques de una neurona con función discreta que muestra la suma realizada por el nodo de suma y el umbral de limitación dura, realizado por la unidad lógica de umbral (TLU). Este modelo consta de los pesos sinápticos, un nodo de suma, y el elemento TLU.

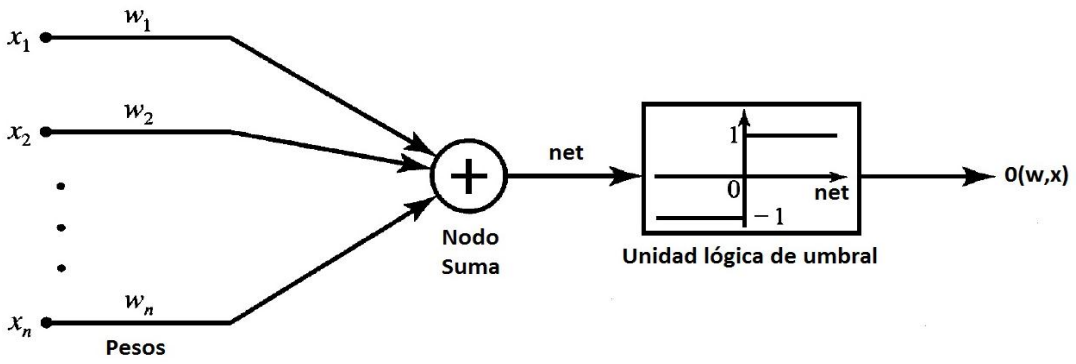


Fig. A.7 Perceptrón discreto (binario)

En el caso de la función de activación continua, como en (A.3a), el modelo usado se muestra en la Fig. A.8, la neurona es representada como amplificador sumador de saturación de alta ganancia que amplía su señal de entrada $w^t x$. Los modelos de las Figs. A.7 y A.8 pueden ser llamados perceptrones discretos (binario) y perceptrones continuos, respectivamente. El perceptrón discreto, introducido por Rosenblatt (1958), fue la primera máquina de aprendizaje y puede ser considerado como un precursor de muchos de los modelos de red neuronal que se utilizan hoy. Además, su estudio proporciona una visión considerable en la naturaleza de los sistemas neurales artificiales.

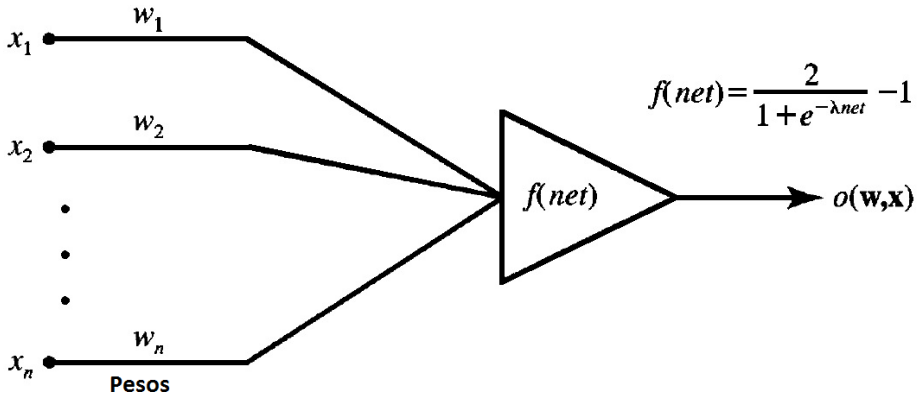


Fig. A.8 Perceptrón continuo

La salida

Como se vio anteriormente, las salidas de las neuronas son discretas o continuas. Dada una capa de m neuronas, sus valores de salida $o_1, o_2, \ldots, o_m$ pueden estar dispuestos en un vector de salida de una capa:

$$o = \begin{bmatrix} o_1 & o_2 \cdots & o_m \end{bmatrix}^t \tag{A.5}$$

Donde o_i es la señal de salida de la $i'esima$ neurona. El dominio de vectores o es definido en un espacio de dimensión m como el siguiente para $i = 1, 2, ..., m$:

$$(-1,1)^m \equiv \left\{ o \in R^m, o_i \in (-1,1) \right\} \tag{A.6a}$$

O bien,

$$(0,1)^m \equiv \left\{ o \in R^m, o_i \in (0,1) \right\} \tag{A.6b}$$

Para las funciones de activación continuas unipolares y bipolares definidas como en (A.3a) y (A.4a), respectivamente.

Aprendizaje y adaptación de redes neuronales artificiales

El aprendizaje en los seres humanos y los animales es un proceso inferido, no es posible ver qué pasa directamente y se supone que se ha producido mediante la observación de los cambios en el rendimiento.

En redes neuronales artificiales se denomina aprendizaje al proceso de configuración de la red para que las entradas produzcan las salidas deseadas a través del fortalecimiento de las conexiones.

Una forma de llevar esto a cabo, es a partir del establecimiento de pesos conocidos con anterioridad, y el otro método implica el uso de técnicas de retroalimentación y patrones de aprendizaje que cambian los pesos hasta encontrar los adecuados.

Una red neuronal debe aprender a calcular la salida correcta para cada vector de entrada en el conjunto de datos. Este proceso de aprendizaje se denomina: proceso de entrenamiento o acondicionamiento. El conjunto de datos sobre el cual se basa este proceso es, por ende, llamado: conjunto de datos de entrenamiento.

La topología de la red y las diferentes funciones de cada neurona no pueden cambiar durante el aprendizaje, mientras que los pesos sobre cada una de las conexiones si pueden hacerlo; el aprendizaje de una red neuronal significa: adaptación de los pesos.

En otras palabras el aprendizaje es el proceso por el cual una red neuronal modifica sus pesos en respuesta a una información de entrada. Los cambios que se producen durante el mismo se reducen a la destrucción, modificación y creación de conexiones entre las neuronas. En los sistemas biológicos existe una continua destrucción y creación de conexiones entre las neuronas. En los modelos de redes neuronales artificiales, la creación de una nueva conexión implica que el peso de la misma pasa a tener un valor distinto de cero. De la misma manera, una conexión se destruye cuando su peso pasa a ser cero.

En general, existen dos tipos de aprendizaje en redes neuronales artificiales:

a) Aprendizaje supervisado.
b) Aprendizaje no supervisado.

Aprendizaje supervisado

En el aprendizaje supervisado se asume que en cada instante de tiempo cuando la entrada es aplicada, la respuesta deseada del sistema es proporcionada por el maestro. Esto se ilustra en la Fig. A.10, la distancia p [d, o] entre la respuesta obtenida y la respuesta deseada sirve como una medida de error y se utiliza para corregir los parámetros externos de la red.

Debido a que se asume que los pesos son ajustables, el maestro puede implementar un esquema de recompensa y castigo para adaptar los pesos de la matriz W en la red. Este modo de aprendizaje es muy penetrante. Además, se utiliza en muchas situaciones de aprendizaje natural. Un conjunto de entrada y patrones de salida llamados un conjunto de entrenamiento es necesario para este modo de aprendizaje.

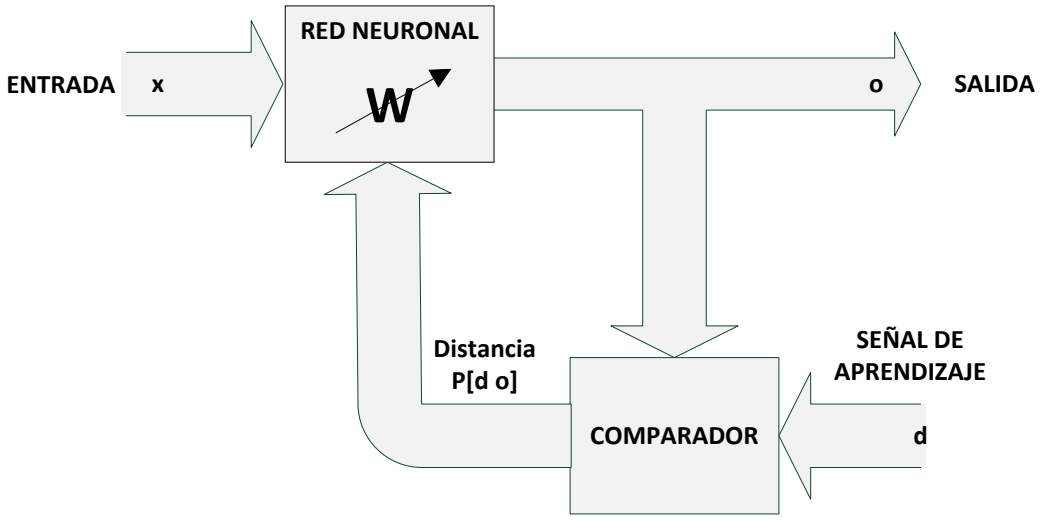


Fig. A.9 Diagrama a bloques del aprendizaje supervisado

Por lo general, en el aprendizaje supervisado se premian las clasificaciones o asociaciones exactas y se castiga a aquellas que producen respuestas inexactas. El maestro estima el error negativo de la dirección del gradiente y reduce el error correspondiente. En muchas situaciones, las entradas, las salidas y el cálculo del gradiente son deterministas, sin embargo, la minimización del error se realiza de manera aleatoria. Como resultado, la mayoría de los algoritmos de aprendizaje supervisado reducen a minimización estocástica del error en espacios multi-dimensionales de pesos.

Aprendizaje no supervisado

En el aprendizaje sin supervisión, la respuesta deseada no se conoce, por lo tanto, la información de error explícito no se puede utilizar para mejorar el comportamiento de la red, dicho aprendizaje se muestra en la Fig. A.10, dado que no se dispone de información

en cuanto a la corrección o incorrección de las respuestas, el aprendizaje de alguna manera debe llevarse a cabo sobre la base de las observaciones de las respuestas a las entradas que se tienen.

En este tipo de aprendizaje, las redes deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada. Existen varias posibilidades en cuanto a la interpretación de la salida de estas redes, que dependen de su estructura y del algoritmo de aprendizaje empleado.

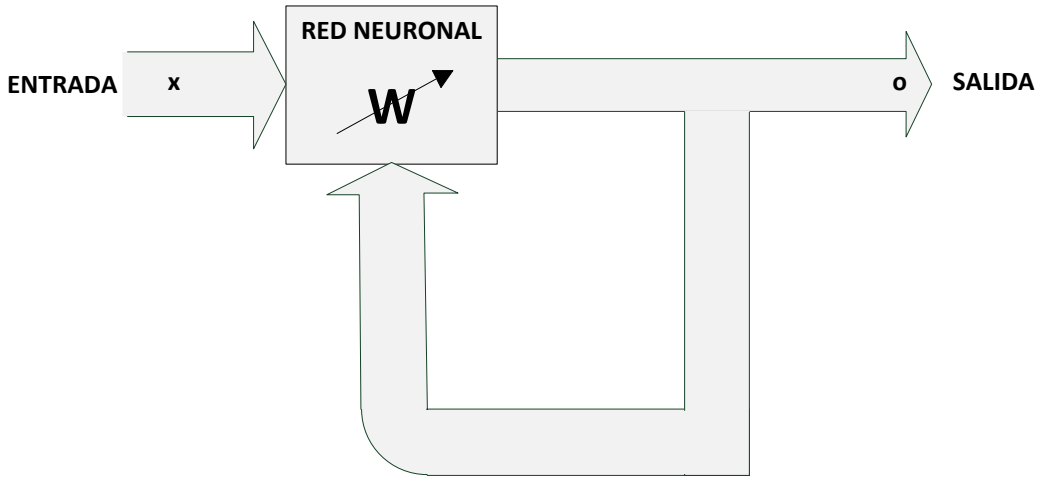


Fig. A.10 Diagrama a bloques del aprendizaje no supervisado

La técnica de aprendizaje no supervisado se utiliza a menudo para llevar a cabo la agrupación como la clasificación no supervisada de los objetos, sin proporcionar información acerca de las clases reales. Este tipo de aprendizaje responde con un mínimo de información *a priori* disponible.

Modelos de redes neuronales artificiales

El modelo o arquitectura de una red neuronal consiste en la organización y disposición de las neuronas en la misma, formando capas o agrupaciones de neuronas alejadas de la entrada y salida de dicha red. En este sentido, los parámetros fundamentales de la red

son: el número de capas, el número de neuronas por capa, el grado de conectividad y el tipo de conexiones entre neuronas.

Redes monocapa

En las redes monocapa, se establecen conexiones entre las neuronas que pertenecen a la única capa que constituye la red. Las redes monocapas se utilizan generalmente en tareas relacionadas con lo que se conoce como auto-asociación (regenerar información de entrada que se presenta a la red de forma incompleta o distorsionada).

Redes multicapa

Las redes multicapas son aquellas que disponen de un conjunto de neuronas agrupadas en varios (2, 3, etc.) niveles o capas. En estos casos, una forma para distinguir la capa a la que pertenece una neurona, consistiría en fijarse en el origen de las señales que recibe a la entrada y el destino de la señal de salida. Normalmente, todas las neuronas de una capa reciben señales de entrada desde otra capa anterior (la cual está más cerca a la entrada de la red), y envían señales de salida a una capa posterior (que está más cerca a la salida de la red). A estas conexiones se las denomina *conexiones hacia adelante o feedforward*.

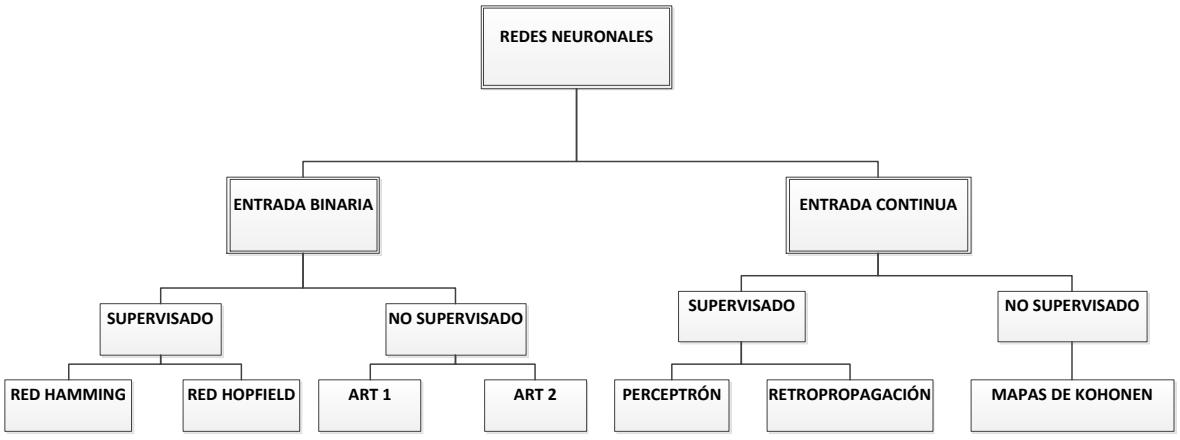


Fig. A.11 Clasificación básica de RNA

Sin embargo; en un gran número de estas redes también existe la posibilidad de conectar la salida de las neuronas de capas posteriores a la entrada de capas anteriores; a estas conexiones se las denomina *conexiones hacia atrás o feedback.*

Estas dos posibilidades permiten distinguir entre dos tipos de redes con múltiples capas: las redes con conexiones hacia adelante o *redes feedforward*, y las redes que disponen de conexiones tanto hacia adelante como hacia atrás o *redes feedforward/feedback*.

APÉNDICE B

REGLA DELTA GENERALIZADA

La arquitectura de la red de dos capas considerada a continuación se muestra en la Fig. B.1, estrictamente hablando, dos capas del procesamiento de neuronas. Sin embargo, si se cuenta por las capas de nodos, la red también puede ser considerada como una red de tres capas. La i-ésima columna de las señales se entiende en como una respuesta de una capa de salida inexistente (entrada) de la capa de neuronas. Por lo tanto se puede referir a la arquitectura de la Fig.B.1 Como una red de dos capas de neuronas, o red de tres capas de nodos.

El término "capa", hace referencia al número real de capas neuronales existentes y procesadas. Por lo tanto, no se contará a las terminales de entrada como capas. Así, la red de la Fig. B.1 es una red de dos capas. También debemos observar que una red de N capas tiene $N-1$ capas de neuronas cuyas salidas no son accesibles.

Las capas con neuronas cuyas salidas o resultados no son directamente accesibles se denominan capas internas u ocultas. Por lo tanto, todas excepto la capa de salida son las capas ocultas. Puesto que la salida de la capa j de las neuronas no es accesible desde la entrada y la salida la red de la Fig.B.1 puede ser nombrada como la única red de capa oculta.

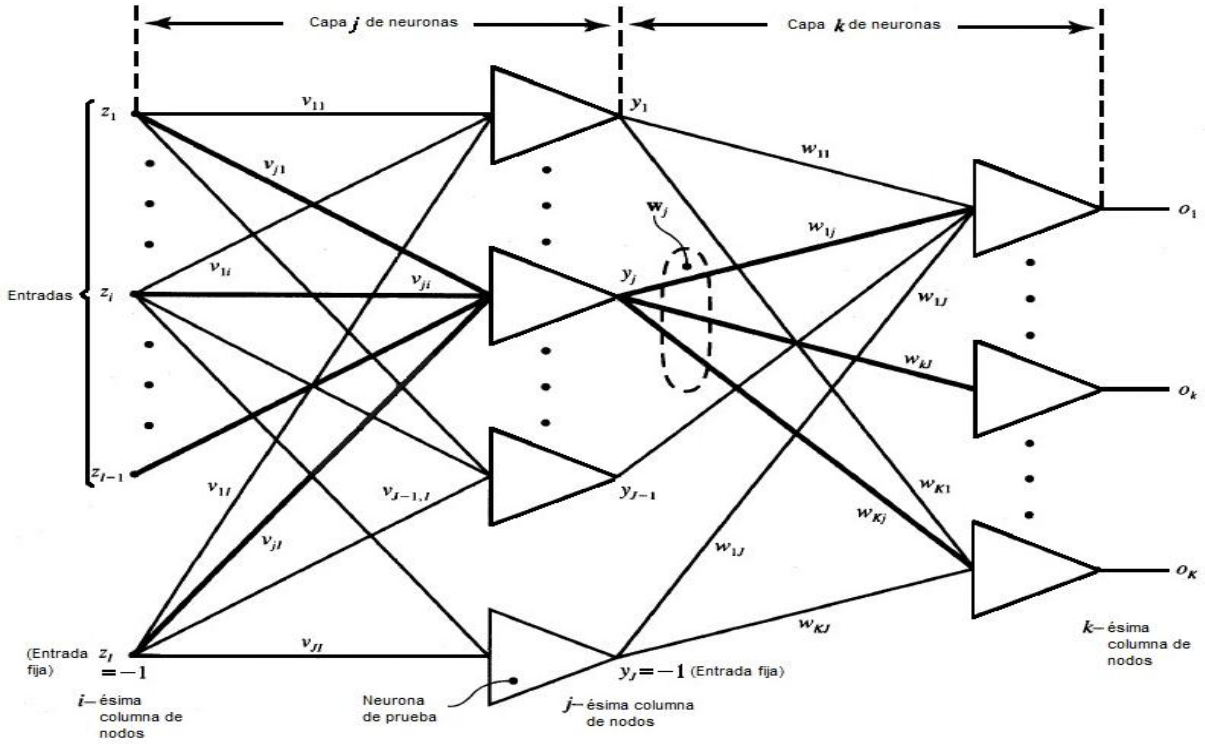


Fig. B.1 Red multicapa con dos capas de perceptrones continuos

Los pesos calculados conducen ahora desde el nodo i hacia el nodo j, como se muestra en la Fig. B.1.

La fórmula del gradiente negativo para la capa oculta ahora se lee:

$$\Delta v_{ji} = -\eta \frac{\partial E}{\partial v_{ji}}, \qquad \text{Para } j = 1, 2, ..., J \text{ y } i = 1, 2, ..., I \qquad \text{(B.21a)}$$

Y la fórmula de la componente del gradiente del error:

$$\frac{\partial E}{\partial v_{ji}} = \frac{\partial E}{\partial \left(net_j \right)} \cdot \frac{\partial \left(net_j \right)}{\partial v_{ji}} \qquad \text{(B.21b)}$$

Se toma en cuenta que las entradas a la capa son z_i para $i = 1, 2, ..., I$. Sobre la base de la relación (B.8), el segundo término en el producto (4.21b) es igual a z_i y podemos expresar el ajuste de peso de manera similar a (B.10) como

$$\Delta v_{ji} = \eta \delta_{yj} z_i \tag{B.21c}$$

Donde δ_{yj} es el término para la señal de error de la capa oculta con salida y. Este término para la señal de error es producido por la j-ésima neurona de la capa oculta, donde $j = 1, 2, ..., J$. El término para la señal de error es igual a

$$\delta_{yj} = -\frac{\partial E}{\partial \left(net_j \right)}, \text{ Para } j = 1, 2, ..., J \tag{B.22}$$

En contraste con la excitación de la capa de salida de las neuronas net_k, que afecta únicamente a la k-ésima neurona de salida, la net_j, contribuye ahora a *cada* componente de error en la suma de error que contiene los términos especificados en la expresión de K (B,4). El término para la señal de error δ_{yj}, en el nodo j puede ser computado como sigue:

$$\delta_{yj} = -\frac{\partial E}{\partial y_j} . \frac{\partial y_j}{\partial \left(net_j \right)} \tag{B.23a}$$

Donde

$$\frac{\partial E}{\partial y_j} = \frac{\partial}{\partial y_j} \left(\frac{1}{2} \sum_{k=1}^{K} \left\{ d_k - f \left[net_k \left(y \right) \right] \right\}^2 \right) \tag{B.23b}$$

Y, obviamente, el segundo término de (B.23a) es igual a

$$\frac{\partial y_j}{\partial\left(net_j\right)} = f_j\,'(net_j) \tag{B.23c}$$

Cálculos de rutina de (B.23b) dan como resultado

$$\frac{\partial E}{\partial y_j} = \sum_{k=1}^{K}\left(d_k - 0_k\right)\frac{\partial}{\partial y_j}\left\{ f\left[netk\left(y\right)\right]\right\} \tag{B.24a}$$

Cálculo de la derivada de la expresión entre llaves (4.24a) se obtiene

$$\frac{\partial E}{\partial y_j} = \sum_{k=1}^{K}\left(d_k - 0_k\right) f\,'(net_k)\frac{\partial(net_k)}{\partial y_j} \tag{B.24b}$$

Podemos simplificar la expresión de arriba con la forma compacta que se presenta a continuación, utilizando la expresión (4.14) de δ_{ok} y (4.5b) de net_k.

$$\frac{\partial E}{\partial y_j} = \sum_{k=1}^{K}\delta_{ok} w_{kj} \tag{B.24c}$$

Combinando (B.23c) y (B.24c) se obtiene como resultado el reordenamiento δ_{yj} expresado en (B.23a) de la forma

$$\delta_{yj} = f_j\,'(net_j)\sum_{k=1}^{K}\delta_{ok} w_{kj}, \quad \text{Para} \quad j = 1, 2, ..., J \tag{B.25}$$

El ajuste del peso (B.21c) en la capa oculta ahora se convierte en

$$\Delta v_{ji} = \eta\, f_j\,'(net_j) z_i \sum_{k=1}^{K}\delta_{ok} w_{kj}, \quad \text{Para} \begin{matrix} j = 1, 2, ..., J \\ i = 1, 2, ..., I \end{matrix} \tag{B.26a}$$

Donde los términos $f_j'(net_j)$ se van a calcular ya sea desde (B.16d) o desde (B.18b) como en el caso de una simple regla delta de entrenamiento. La fórmula (B.26a) expresa la llamada regla delta generalizada.

El ajuste de los pesos que conducen a la neurona j en la capa oculta es proporcional a la suma ponderada de todos los valores δ en la capa adyacente siguiente de nodos conectando la neurona j con la salida. Los pesos que se abren en abanico desde el nodo j son por sí mismos los factores de ponderación. Los pesos que afectan δ_{yj} de la j-ésima neurona oculta se han destacado en la capa de salida de la Fig. B.1. Todos los errores de las capas de salida $\delta_{ok} w_{kj,}$ para $k = 1, 2, ..., K$ contribuyen al ajuste de pesos resaltado v_{ji} para $i = 1, 2, ..., I$ de la capa oculta. Los pesos modificadas de la capa oculta se pueden expresar ahora como

$$v'_{ji} = v_{ji} + \eta f_j'(net_j) z_i \sum_{k=1}^{K} \delta_{ok} w_{kj,} \quad \text{Para} \quad j = 1, 2, ..., J \quad \text{y} \qquad (B.26b)$$

$$i = 1, 2, ..., I$$

El ajuste de peso de la capa oculta basado en el principio de la regla delta generalizada para la red de la Fig. B.7 puede sucintamente ser declarado en notación vectorial como

$$V' = V + \eta \delta_y z^t \qquad (B.27)$$

Donde

$$z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_I \end{bmatrix},$$

$$V = \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1I} \\ v_{21} & v_{22} & \cdots & v_{2I} \\ \vdots & \vdots & \cdots & \vdots \\ v_{J1} & V_{J2} & \cdots & v_{JI} \end{bmatrix}$$

Y δ_y es el vector de columna con entradas δ_{yj} dada por (B.25).

Definiendo ahora la columna j-ésima de la matriz W como w_j, vector δ_y, se puede expresar de la siguiente forma compacta

$$\delta_y = w_j^t \delta_o f'_y \tag{B.28}$$

Donde f'_y; es el vector de columna con entradas f'_{yj}; expresado para cada una de las neuronas de capa oculta $1,2,...,J$, para funciones de activación unipolares y bipolares, respectivamente, como

$$f'_{yj} = y_j(1-y_j) \tag{B.29}$$

$$f'_{yj} = \frac{1}{2} y_j(1-y_j^2) \tag{B.29b}$$

Vector δ_o, que se utiliza en (4.28) se define como en (B.20).

Comparación de la regla delta de entrenamiento (B.20) para el ajuste de los pesos de la capa de salida y la regla delta generalizada (B.27) para el ajuste de los pesos de la capa oculta indicando que ambas fórmulas son bastante uniformes. La diferencia significativa se da en los subíndices, que se refieren la ubicación de los pesos y las señales de entrada, y en la forma en que la señal de error del vector δ se calcula.

El vector δ_o, contiene entradas escalares (B.17) o (B.18a). Cada componente del vector δ, es simplemente la diferencia entre los valores de salida deseados y los tiempos reales de la derivada de la función de activación. El vector δ_y, sin embargo, contiene entradas que son productos escalares $w_j^t \delta_o f'_{yj}$ expresando la suma ponderada de las señales de contribución de error δ_o, producida por la siguiente capa. La regla de aprendizaje delta generalizada propaga el error hacia atrás por una capa, permitiendo que el mismo proceso se repita para cada capa anterior a la capa j. En la siguiente sección vamos a formalizar el método de entrenamiento para capas de redes neuronales.

APÉNDICE C

MÉTODO DE LEVENBERG-MARQUARDT PARA EL ENTRENAMIENTO DE REDES NEURONALES

El algoritmo de retropropagación (EBP) ha sido una mejora significativa en la investigación de redes neuronales pero tiene una tasa de convergencia débil.

Muchos esfuerzos se han hecho para acelerar el algoritmo EBP. Todos estos métodos conducen a resultados poco aceptables. El algoritmo Levenberg-Marquardt (LM) se produjo por el desarrollo de métodos dependientes del algoritmo EBP.

Éste da un buen intercambio entre el algoritmo de velocidad de Newton y la estabilidad del método de pendiente descendente, éstos son dos teoremas básicos del algoritmo LM. Se ha hecho un intento para acelerar el algoritmo LM con la modificación del índice de desempeño modificado y el cálculo del gradiente, aunque es incapaz de reducir el error de oscilación. Se ha producido otro esfuerzo con la tasa de atenuación variable para reducir el error de oscilación, pero el algoritmo que se propuso tenía poca velocidad en comparación con el algoritmo LM estándar.

En el algoritmo de EBP, el índice de rendimiento de $F(w)$ al ser minimizado se define como la suma de errores cuadráticos entre los resultados previstos y los resultados simulados de la red, como a continuación se describe:

$$F(w) = e^{T|}e \tag{1}$$

Donde $w = \left[w1, w2, ...w_N \right]$ consta de todos los pesos de la red, e es el vector de error que comprende el error de para todos los ejemplos de entrenamiento.

Al entrenar con el método LM, el incremento de pesos Δw se puede obtener de la siguiente forma:

$$\Delta w = \left[J^T J + \mu I \right]^{-1} J^T e \tag{2}$$

Donde J es la matriz Jacobiana, μ es la tasa de aprendizaje que será actualizada utilizando β en función del resultado. En particular, μ se multiplica por la tasa de descomposición $\beta\left(0<\beta<1\right)$ cuando $F(w)$ disminuye, mientras que μ se divide por β siempre que $F(w)$ aumenta en un nuevo paso.

El proceso de entrenamiento LM estándar puede ser ilustrado en los siguientes pseudo-códigos,

1. Inicializar los pesos y el parámetro $\mu(\mu=.01$ es apropiado).
2. Calcular la suma de los errores cuadráticos sobre todas entradas $F(w)$.
3. Resolver (2) para obtener el incremento de pesos Δw
4. Recalcular la suma de errores cuadráticos $F(w)$.

Usando $w+\Delta w$ como el nuevo w, y seguir con:

Si el nuevo $F(w)<F(w)$ en el paso 2, entonces,

$$w=w+\Delta w$$
$$\mu=\mu\cdot\beta(\beta=.1)$$

Volver al paso 2

De lo contrario

$$\mu={}^{\mu}\!/_{\beta}$$

Volver al paso 4

Considerando que el índice de rendimiento es $F(w)=e^Te$ utilizando el método Newton tenemos como:

$$W_{K+1}=W_K-A_K^{-1}\cdot g_K \tag{3}$$

$$A_k = \nabla^2 F(w)\,|_{w=w_k} \tag{4}$$

$$g_k = \nabla F(w)\,|_{w=w_k} \tag{5}$$

$$\left[\nabla F(w)\right]_j = \frac{\partial F(w)}{\partial w_j} = 2\sum_{i=1}^{N} e_i(w).\frac{\partial e_i(w)}{\partial w_j} \tag{6}$$

El gradiente se puede escribir como:

$$\nabla F(x) = 2J^T e(w) \tag{7}$$

Donde

$$J(w) = \begin{bmatrix} \frac{\partial e_{11}}{\partial w_1} & \frac{\partial e_{11}}{\partial w_2} & \cdots\cdots & \frac{\partial e_{11}}{\partial w_N} \\ \frac{\partial e_{21}}{\partial w_1} & \frac{\partial e_{21}}{\partial w_2} & \cdots\cdots & \frac{\partial e_{21}}{\partial w_N} \\ \vdots & & & \\ \frac{\partial e_{KP}}{\partial w_1} & \frac{\partial e_{KP}}{\partial w_2} & \cdots\cdots & \frac{\partial e_{KP}}{\partial w_N} \end{bmatrix} \tag{8}$$

$J(w)$ Es la matriz Jacobiana.

En seguida se debe hallar la matriz Hessiana. Los elementos k, j del rendimiento de la matriz Hessiana se encuentran con la siguiente formula:

$$\left[\nabla^2 F(w)\right]_{k,j} = \frac{\partial^2 F(w)}{\partial w_k \partial w_j}$$

$$= 2\sum_{i=1}^{N}\sum_{i=1}^{n} X_i^2 \left\{ \frac{\partial e_i(w)}{\partial w_k}\frac{\partial e_i(w)}{\partial w_j} + e_i(w).\frac{\partial^2 e_i(w)}{\partial w_k \partial w_j} \right\} \tag{9}$$

La matriz Hessiana puede entonces expresarse como sigue:

$$\nabla^2 F(W) = 2J^T(W)\cdot J(W) + S(W) \tag{10}$$

Donde

$$S(w) = 2\sum_{i=1}^{N} e_i(w).\nabla^2 e_i(w) \tag{11}$$

Si suponemos que $S(w)$ es pequeña, se puede aproximar la matriz Hessiana como:

$$\nabla^2 F(w) \cong 2J^T(w)J(w) \tag{12}$$

Usando (12) y (4) se obtiene el método de Gauss-Newton como:

$$\begin{aligned} &W_{k+1} = \\ &W_k - \left[2J^T(w_k)\cdot J(w_k)\right]^{-1} 2J^T(w_k)e(w_k) \\ &\cong W_k - \left[J^T(w_k)\cdot J(w_k)\right]^{-1} J^T(w_k)e(w_k) \end{aligned} \tag{13}$$

La ventaja de Gauss-Newton es que no requiere cálculo de segundas derivadas.

Hay un problema en el método de Gauss-Newton, se trata de la matriz $H = J^T J$ que puede no ser invertible. Esto se puede superar mediante el uso la siguiente modificación.

La matriz Hessiana se puede escribir como:

$$G = H + \mu I \tag{14}$$

Suponemos que los valores propios y los vectores propios de H son $\{\lambda_1, \lambda_2, ..., \lambda_n\}$ y $\{Z_1, Z_2, ..., Z_n\}$

Entonces:

$$\begin{aligned} Gz_i &= \left[H + \mu I\right] z_i \\ &= Hz_i + \mu z_i \\ &= \lambda_i z_i + \mu z_i \\ &= \left(\lambda_i + \mu\right) z_i \end{aligned} \tag{15}$$

Por lo tanto, los vectores propios de G son los mismos que los vectores propios de H, y los valores propios de G son $(\lambda_i + \mu)$. La matriz G es definida positiva, aumentando μ hasta $(\lambda_i + \mu) < 0$ para todo i por lo tanto, la matriz será invertible.

Esto conduce al algoritmo Levenberg-Marquardt:

$$W_{K+1} = W_K - \left[J^T(w_K) J(w_k) + \mu I \right]^{-1} J^T(w_K) e(w_K) \tag{16}$$

$$\Delta W_K = \left[J^T(w_K) J(w_K) + \mu I \right]^{-1} J^T(w_K) e(w_K) | \tag{17}$$

Como es conocido, el parámetro de aprendizaje, μ es ilustrador de pasos de movimientos de salida real a la salida deseada. En el método LM estándar, μ es un número constante. Sin embargo, para mejores resultados se modifica μ como:

$$\mu = 0.01 e^T e \tag{18}$$

Donde e es a la matriz k × 1 por lo tanto, $e^T e$ es a 1 × 1 tanto $\left[J^T J + \mu I \right]$ es invertible.

Por lo tanto, si la salida real está lejos de salida deseada o similar, los errores son grandes, así, converge a la salida deseada con pasos grandes.

Del mismo modo, cuando la medida de error es pequeño, entonces, la salida real se aproxima a la salida deseada con pasos suaves. Por lo tanto el error oscilación se reduce grandemente.

APÉNDICE D

LISTADO DE PROGRAMAS DESARROLLADOS

Este programa de estator obtiene el error de una red neuronal entrenada ejecutando otras subrutinas previamente desarrolladas y contando con una red neuronal entrenada.

Estator

```
clear all
clc
load estator_nuevosdatos
q1=[500000 750000 1000000];
q2=[100000 150000 200000];
alphaext=[50 100 250 400];
alphaent=[50 100 250 400];
tiempo=[10 50 150 300 700 1000 2000];
pm=combvec(q1,q2,alphaext,alphaent,tiempo);
[n,m]=size(pm);
temp= zeros(m,2);
u0=zeros(1,m);
for i=1:m
    tlist=0:1:pm(5,i);
    b=vectorb3(pm(3,i),pm(4,i),b);
    f=funcionf(pm(1,i),pm(2,i));
    u1=parabolic(u0(1,i),tlist,b,p,e,t,c,a,f,d);
    temp(i,1)=u1(9,pm(5,i)+1);
    temp(i,2)=u1(10,pm(5,i)+1);

    i
end
```

```
deseadas=temp';
obtenidas=sim(trainlm1_trescapas25y10,pm)';
dif=(deseadas-obtenidas);
[fil,col]=size(dif);
for k=1:fil
   for i=1:col
      error(k,i)=dif(k,i)/deseadas(k,i);
   end
end
errores=abs(error)*100;
emax=max(errores);
```

Este programa de rotor obtiene el error de una red neuronal entrenada ejecutando otras subrutinas previamente desarrolladas y contando con una red neuronal entrenada.

Rotor

```
clear all
clc
load rotor_prueba1_2nodos_nuevosdatos
q1=[500000 750000 1000000];
q2=[100000 150000 200000];
alpha=[50 100 250 400];
tiempo=[10 50 150 300 700 1000 2000];
pm=combvec(q1,q2,alpha,tiempo);
[n,m]=size(pm);
temp= zeros(2,m);
u0=zeros(1,m);
for i=1:m
    tlist=0:1:pm(4,i);
```

```
    b=vectorb4(0,pm(3,i),b);
    f=funcionf(pm(1,i),pm(2,i));
    u1=parabolic(u0(1,i),tlist,b,p,e,t,c,a,f,d);
    temp(1,i)=u1(170,pm(4,i)+1);
    temp(2,i)=u1(177,pm(4,i)+1);
    i
end
deseadas=temp';
obtenidas=sim(trainlm1_trescapas25y10,pm)';
dif=(deseadas-obtenidas);
[fil,col]=size(dif);
for k=1:fil
   for i=1:col
      error(k,i)=dif(k,i)/deseadas(k,i);
   end
end
errores=abs(error)*100;
emax=max(errores);
```

Este programa determina la matriz b del estator, que introduce los valores de coeficiente de transferencia de calor por convección.

```
function[b]=vectorb3(alfacme,alfade,b)
walfa=double(num2str(alfacme))';
walfa1=double(num2str(alfade))';
lwfa=length(walfa);
lwfa1=length(walfa1);
%load b
b7=b(:,7);
b7(3)=lwfa1;
b8=b(:,8);
b8(3)=lwfa1;
```

```
b6=b(:,6);
b6(3)=lwfa;
if alfacme<1000
   b6(9)=49;
else
   b6(9)=48;
end
if alfade<1000
   b7(9)=49;
   b8(9)=49;
else
   b7(9)=48;
   b8(9)=48;
end
for i=5:(4+lwfa)
   b6(i)=walfa(i-4);
end
for i=5:(4+lwfa1)
   b7(i)=walfa1(i-4);
   b8(i)=walfa1(i-4);
end
b(:,6)=b6;
b(:,7)=b7;
b(:,8)=b8;
```

Este programa determina la matriz b del rotor, que introduce los valores de coeficiente de transferencia de calor por convección.

```
function[b]=vectorb4(alfacme,alfade,b)
walfa=double(num2str(alfacme))';
walfa1=double(num2str(alfade))';
lwfa=length(walfa);
```

```
lwfa1=length(walfa1);
%load mc
b56=b(:,56);
b56(3)=lwfa;
b57=b(:,57);
b57(3)=lwfa;
b58=b(:,58);
b58(3)=lwfa;
b64=b(:,64);
b64(3)=lwfa;
b65=b(:,65);
b65(3)=lwfa1;
if alfacme<1000
   b56(9)=49;
   b57(9)=49;
   b58(9)=49;
   b64(9)=49;
else
   b56(9)=48;
   b57(9)=48;
   b58(9)=48;
   b64(9)=48;

end
if alfade<1000
   b65(9)=49;
else
   b65(9)=48;
end
for i=5:(4+lwfa)
   b56(i)=walfa(i-4);
   b57(i)=walfa(i-4);
```

```
   b58(i)=walfa(i-4);
   b64(i)=walfa(i-4);

end
for i=5:(4+lwfa1)
   b65(i)=walfa1(i-4);

end
b(:,56)=b56;
b(:,57)=b57;
b(:,58)=b58;
b(:,64)=b64;
b(:,65)=b65;
```

Función que describe la generación de calor

```
function[f]=funcionf(q1,q2)
%q1 - generacion de calor en devanado
%q2 - generacion de calor en circuito magnetico
%f - char array, generacion de calor para "parabolic"
f1='(Q1)+(0).*(0.0)!(Q2)+(0).*(0.0)!(Q2)+(0).*(0.0)!(Q1)+(0).*(0.0)';
z1=num2str(q1);
z2=num2str(q2);
f2=strrep(f1,'Q2',z2);
f=strrep(f2,'Q1',z1);
```

Programa que determina la distribución de las temperaturas por pérdidas eléctricas y magnéticas en el núcleo del estator realizado por la interfaz gráfica *pdetool* de MATLAB.

```
function pdemodel
[pde_fig,ax]=pdeinit;
pdetool('appl_cb',9);
set(ax,'DataAspectRatio',[3.2362030827321888 1 47.105500450856582]);
set(ax,'PlotBoxAspectRatio',[1 1 1]);
set(ax,'XLim',[0.15395507812499998 0.29135742187500002]);
set(ax,'YLim',[0 0.042457886676876]);
set(ax,'XTick',[ 0,...
 0.01,...
 0.02,...
 0.029999999999999999,...
 0.040000000000000001,...
 0.050000000000000003,...
 0.059999999999999998,...
 0.070000000000000007,...
 0.080000000000000002,...
 0.089999999999999997,...
 0.10000000000000001,...
 0.11,...
 0.12,...
 0.13,...
 0.14000000000000001,...
 0.14999999999999999,...
 0.16,...
 0.17000000000000001,...
 0.17999999999999997,...
 0.18999999999999997,...
 0.19999999999999998,...
```

```
 0.20999999999999996,...
 0.21999999999999997,...
 0.22999999999999998,...
 0.23999999999999999,...
 0.24999999999999997,...
 0.26000000000000001,...
 0.26999999999999996,...
 0.27999999999999997,...
 0.28999999999999998,...
 0.29999999999999999,...
 0.31,...
 0.31999999999999995,...
 0.32999999999999996,...
 0.33999999999999997,...
 0.34999999999999998,...
]);
set(ax,'YTick',[ 0,...
 0.01,...
 0.02,...
 0.029999999999999999,...
 0.040000000000000001,...
 0.050000000000000003,...
 0.059999999999999998,...
 0.070000000000000007,...
 0.080000000000000002,...
 0.089999999999999997,...
 0.10000000000000001,...
 0.11,...
 0.12,...
 0.13,...
 0.14000000000000001,...
 0.14999999999999999,...
```

```
 0.16,...
 0.17000000000000001,...
 0.17999999999999997,...
 0.18999999999999997,...
 0.19999999999999998,...
 0.20999999999999996,...
 0.21999999999999997,...
 0.22999999999999998,...
 0.23999999999999999,...
 0.24999999999999997,...
 0.26000000000000001,...
 0.26999999999999996,...
 0.27999999999999997,...
 0.28999999999999998,...
 0.29999999999999999,...
 0.31,...
 0.31999999999999995,...
 0.32999999999999996,...
 0.33999999999999997,...
 0.34999999999999998,...
]);
pdetool('gridon','on');

% Geometry description:
pdecirc(0,0,0.27489150000000001,'C1');
pdecirc(0,0,0.1656715,'C2');
pdepoly([ 0.16405142028554426,...
 0.22781302052349789,...
 0.22781302052349789,...
 0.16405142028554426,...
],...
[ 0.0059307450790967164,...
```

```
 0.0062809443303948678,...
 -0.0063262287163386166,...
 -0.0059760294650404652,...
],...
 'P1');
pdepoly([ 7.6569678407314989e-005,...
 0.28001531393568146,...
 0.28001531393568146,...
],...
[ 0.00020673813169981049,...
 0.01492343032159261,...
 0.00020673813169981049,...
],...
 'P2');
set(findobj(get(pde_fig,'Children'),'Tag','PDEEval'),'String','(C1-C2)*P2')

% Boundary conditions:
pdetool('changemode',0)
pdesetbd(8,...
'neu',...
1,...
'100',...
'0')
pdesetbd(7,...
'neu',...
1,...
'100',...
'0')
pdesetbd(6,...
'neu',...
1,...
'200',...
```

```
'0')
pdesetbd(5,...
'neu',...
1,...
'0',...
'0')
pdesetbd(4,...
'neu',...
1,...
'0',...
'0')
pdesetbd(3,...
'neu',...
1,...
'0',...
'0')

% Mesh generation:
setappdata(pde_fig,'Hgrad',1.3);
setappdata(pde_fig,'refinemethod','regular');
setappdata(pde_fig,'jiggle',char('on','mean',''));
pdetool('initmesh')

% PDE coefficients:
pdeseteq(2,...
'386!386',...
'0!0',...
'(750000)+(0).*(0.0)!(150000)+(0).*(0.0)',...
'(8890).*(385.4)!(8890).*(385.4)',...
'0:2000',...
'0.0',...
'0.0',...
```

```
'[0 100]')
setappdata(pde_fig,'currparam',...
['8890!8890    ';...
'385.4!385.4  ';...
'386!386      ';...
'750000!150000';...
'0!0          ';...
'0.0!0.0      '])

% Solve parameters:
setappdata(pde_fig,'solveparam',...
str2mat('0','1000','10','pdeadworst',...
'0.5','longest','0','1E-4','','fixed','Inf'))

% Plotflags and user data strings:
setappdata(pde_fig,'plotflags',[1 1 1 1 1 1 7 1 0 1 0 2001 1 0 0 0 0 1]);
setappdata(pde_fig,'colstring','');
setappdata(pde_fig,'arrowstring','');
setappdata(pde_fig,'deformstring','');
setappdata(pde_fig,'heightstring','');

% Solve PDE:
pdetool('solve')
```

Programa que determina la distribución de las temperaturas por pérdidas eléctricas y magnéticas en el núcleo del rotor realizado por la interfaz gráfica *pdetool* de MATLAB.

```
function pdemodel
[pde_fig,ax]=pdeinit;
pdetool('appl_cb',9);
set(ax,'DataAspectRatio',[1.9121661721068248 1 10.623145400593472]);
set(ax,'PlotBoxAspectRatio',[1 1 2]);
set(ax,'XLim',[0 0.17999999999999999]);
set(ax,'YLim',[0 0.094134078212290501]);
set(ax,'XTick',[ 0,...
 0.01,...
 0.02,...
 0.029999999999999999,...
 0.040000000000000001,...
 0.050000000000000003,...
 0.059999999999999998,...
 0.070000000000000007,...
 0.080000000000000002,...
 0.089999999999999997,...
 0.10000000000000001,...
 0.11000000000000001,...
 0.12000000000000001,...
 0.13,...
 0.14000000000000001,...
 0.15000000000000002,...
 0.16,...
 0.17000000000000001,...
 0.18000000000000002,...
 0.19,...
 0.20000000000000001,...
```

```
]);
set(ax,'YTick',[ 0,...
 0.0050000000000000001,...
 0.01,...
 0.014999999999999999,...
 0.02,...
 0.025000000000000001,...
 0.029999999999999999,...
 0.035000000000000003,...
 0.040000000000000001,...
 0.044999999999999998,...
 0.050000000000000003,...
 0.055,...
 0.059999999999999998,...
 0.065000000000000002,...
 0.070000000000000007,...
 0.074999999999999997,...
 0.080000000000000002,...
 0.085000000000000006,...
 0.089999999999999997,...
 0.095000000000000001,...
 0.10000000000000001,...
 0.10500000000000001,...
 0.11000000000000001,...
 0.115,...
 0.12000000000000001,...
 0.125,...
 0.13,...
 0.13500000000000001,...
 0.14000000000000001,...
 0.14500000000000002,...
 0.15000000000000002,...
```

```
 0.15500000000000003,...
 0.16,...
 0.16500000000000001,...
 0.17000000000000001,...
 0.17500000000000002,...
 0.18000000000000002,...
 0.185,...
 0.19,...
 0.19500000000000001,...
 0.20000000000000001,...
]);
pdetool('gridon','on');

% Geometry description:
pdecirc(0,0,0.16324,'C1');
pdecirc(0,0,0.059924999999999999,'C2');
pdepoly([ 0.075378511986686131,...
 0.10295160027972894,...
 0.1039706495946756,...
 0.076048571810212712,...
],...
[ 0.0087201842011408511,...
 0.012097197028597786,...
 0.00044941005966429685,...
 0.0010613653722530798,...
],...
 'P1');
pdepoly([ 0.16079820887276461,...
 0.16058383455847167,...
 0.16035206453949691,...
 0.15973432227399528,...
 0.15928159300097214,...
```

0.1585938012390965,...
0.15792716432942705,...
0.15717637468857945,...
0.1563028817596657,...
0.15548909423984617,...
0.15427569791634221,...
0.15370778365030155,...
0.1488283510012621,...
0.14845272284201502,...
0.14782792895911132,...
0.14717163613672163,...
0.14628028082547945,...
0.14537152980955542,...
0.14451543258532357,...
0.14367343859589576,...
0.14283849622387007,...
0.14211591305263072,...
0.13164198287836107,...
0.12993025510754388,...
0.12893076642068627,...
0.12790683041174472,...
0.12730036558881602,...
0.12687630946304768,...
0.12665581688664587,...
0.12664218032948835,...
0.12707092895807415,...
0.12763272496200592,...
0.12854523512545424,...
0.12952074316787457,...
0.13038389200950851,...
0.13136974413920857,...
0.13220468651123432,...

0.14326908583669051,...
0.14440208857654066,...
0.14534563100182843,...
0.14622241640061989,...
0.14695534365913904,...
0.14766711606545202,...
0.14829473574058688,...
0.14893269950300156,...
0.14930410183707774,...
0.15418353448611719,...
0.15483982730850687,...
0.15565737397585072,...
0.15645376579098838,...
0.15728165654561205,...
0.15849176039923826,...
0.15932999524114172,...
0.16015741931811878,...
0.16071828196675736,...
0.16086167342938312,...
],...
[0.014238694411705602,...
0.015113329304828504,...
0.015661573101770545,...
0.016419700579626217,...
0.016867395654431089,...
0.017375742718547971,...
0.017642288715491092,...
0.017820255010144594,...
0.017825051573501376,...
0.017672637098691497,...
0.01683552304525977,...
0.015973664062058676,...

0.015546769020829455,...
0.016163644153648536,...
0.017002371987228794,...
0.017675909436077362,...
0.018410098874237939,...
0.018817897216217652,...
0.019149084875755295,...
0.019319071823843764,...
0.019408458416207648,...
0.019264023287962805,...
0.017129413750550271,...
0.016736005098640899,...
0.016080040123070922,...
0.015178284407044674,...
0.014231835394836478,...
0.01330134507575835,...
0.012145012382636527,...
0.011250429123101044,...
0.0095011593368552315,...
0.0083320507338115954,...
0.0073560602284772642,...
0.0067104504926063053,...
0.0062986624773440775,...
0.0060600441933785689,...
0.0059706576010146823,...
0.0055579731415005794,...
0.0057383153292880689,...
0.0059832991796700766,...
0.0064660944239433339,...
0.0070175210040936244,...
0.0078107486514176795,...
0.0085153965964521131,...

```
 0.0096270359933919927,...
 0.010634136994912193,...
 0.011061032036141415,...
 0.010387494587292843,...
 0.0099717168987480988,...
 0.0097977402773771153,...
 0.0097889540407378184,...
 0.01013847628653951,...
 0.010536681501805662,...
 0.011583679236151024,...
 0.012526138575076701,...
 0.01351329121018432,...
],...
 'P2');
pdepoly([ 1.9632727226881864e-005,...
 0.16156482959158855,...
 0.16404356508603027,...
],...
[ 8.799630086477247e-005,...
 0.028408265046245072,...
 7.618870006518598e-005,...
],...
 'P3');
set(findobj(get(pde_fig,'Children'),'Tag','PDEEval'),'String','(C1+C2)*P3')

% Boundary conditions:
pdetool('changemode',0)
pdesetbd(65,...
'neu',...
1,...
'150',...
'0')
```

```
pdesetbd(64,...
'neu',...
1,...
'0',...
'0')
pdesetbd(58,...
'neu',...
1,...
'0',...
'0')
pdesetbd(57,...
'neu',...
1,...
'0',...
'0')
pdesetbd(56,...
'neu',...
1,...
'0',...
'0')

% Mesh generation:
setappdata(pde_fig,'Hgrad',1.3);
setappdata(pde_fig,'refinemethod','regular');
setappdata(pde_fig,'jiggle',char('on','mean',''));
pdetool('initmesh')

% PDE coefficients:
pdeseteq(2,...
'386!45!45!386',...
'0!0!0!0',...
'(750000)+(0).*(0.0)!(100000)+(0).*(0.0)!(100000)+(0).*(0.0)!(750000)+(0).*(0.0)',...
```

```
'(8890).*(385.4)!(7880).*(480)!(7880).*(480)!(8890).*(385.4)',...
'0:2000',...
'0.0',...
'0.0',...
'[0 100]')
setappdata(pde_fig,'currparam',...
['8890!7880!7880!8890        ';...
'385.4!480!480!385.4        ';...
'386!45!45!386              ';...
'750000!100000!100000!750000';...
'0!0!0!0                    ';...
'0.0!0.0!0.0!0.0            '])

% Solve parameters:
setappdata(pde_fig,'solveparam',...
str2mat('0','2456','10','pdeadworst',...
'0.5','longest','0','1E-4','','fixed','Inf'))

% Plotflags and user data strings:
setappdata(pde_fig,'plotflags',[1 1 1 1 1 1 7 1 0 1 0 2001 1 0 0 0 0 1]);
setappdata(pde_fig,'colstring','');
setappdata(pde_fig,'arrowstring','');
setappdata(pde_fig,'deformstring','');
setappdata(pde_fig,'heightstring','');

pdetool('solve')
```

APENDICE D-1

Programa que resuelve la red neuronal de dos capas con el algoritmo de retropropagación

```
clear all
clc
('RETROPROPAGACION CON DOS CAPAS')
x=[1 0 -1; -2 1.5 1;0 -0.5 0.5; -1 -1 -1];
lam=1;
c=0.1;
[n,m]=size(x);
w=rand(n,m,m);
v=rand(n,m,m-1);
d=[-1 1];
e=zeros(n,m,m-1);
emax=0.1;
for a=1:m
    for f=1:m
        for p=1:m-1
            neto=w(:,a,f)'*x(:,f);
            o=(2/(1+exp(-lam*neto)))-1;

            nety=v(:,a,f)'*o(f)
            y=(2/(1+exp(-lam*nety)))-1

            fnety=(1/2)*(d(f)-y)*(1-y^2);

            fneto=(1/2)*(1-y^2);

            w(:,a,f+1)=c*(d(f)-o)*fneto*x(:,f)+w(:,a,f);
            e(:,a,f+1)=(0.5*(d(f)-y)^2)+e(:,a,f);
        end
```

```
    end
end
    if e(n,m,m-1)>emax
       w(n,m,m)=w(m+1,:,:);
    else end

 ('Las matrices de pesos son:')
w
v
 ('El error es:')
e(n,m,m-1)
```

APÉNDICE E

TEMPERATURAS LÍMITES DE LOS MATERIALES AISLANTES

CLASE DE AISLAMIENTO	*TEMPERATURA ADMISIBLE PARA UNA TEMPRERATURA AMBIENTE DE 40°C*	*TEMPERATURA MÁXIMA DEL PUNTO MÁS CALIENTE*	*DESCRIPCIÓN DE LOS MATERIALES*
Y(O)	50°C	90°C	*Algodón, seda, papel, u otros materiales orgánicos, ni impregnados ni sumergidos en materiales líquidos aislantes.*
A	65°C	105°C	*1.- Cualquiera de los materiales anteriores sumergidos o impregnados en dieléctricos líquidos. 2.- Esmaltes y barnices aplicados a los conductores. 3.- Películas y láminas de acetato de lulosa u otros productos celulósicos. 4.- Materiales moldeados y laminados que tengan relleno celulósico o resina fenólicas u otras resinas de propiedades similares.*
E	80°C	120°C	*Aislamiento compuesto de materiales que, por experiencia o por ensayos reconocidos, muestran poseer una estabilidad térmica que les permite soportar temperaturas 15°C superiores a las de los materiales de la clase A (nylon, poliéster, esmaltes de CPV y resinas fenólicas, triacetatos de celulosa, tereftalato de polietileno,etc.).*
B	90°C	130°C	*Aislamiento compuesto de materiales inorgánicos (mica, fibra de vídrio, amianto, etc.) combinados con pequeños porcentajes de materiales de la clase A.*
F	115°C	155°C	*Los materiales del grupo anterior con un aglutinante que posee una estabilidad térmica tal que pueden ser utilizados a una temperatura 25°C superiores a los materiales de la clase B (barnices sintéticos de siliconas y poliuretanos).*
H	140°C	180°C	*1.- Mica, asbesto, fibra de vidrio y materiales inorgánicos similares con sustancias aglomerantes a base de compuestos de silicona. 2.- Compuestos de silicona en forma de goma o resina o materiales con propiedades dieléctricas y de temperaturas equivalentes.*
C	Sin límite de temperatura		*Materiales exclusivamente inorgánicos: mica pura, porcelana, vidrio, cuarzo y materiales similares en forma pura (lana de vidrio, cintas aislantes, etc.).*

APÉNDICE F
VALORES APROXIMADOS DEL COEFICIENTE DE TRANSFERENCIA DE CALOR CONECTIVO

MECANISMO		h, Btu/hr ft^2 °F	h, W/(m^2 °K)
Convección libre	Gases en general	0.35 -8.80	2 – 50
	Aire	1 - 10	5 – 50
	Líquidos en general	8.80 - 176.11	50 - 1 000
	Agua	8.80 - 26.41	50 – 150
Convección forzada	Gases en general	4.40 - 52.83	25 – 300
	Aire	5 - 50	25 – 250
	Líquidos en general	3.52 - 3 522	20 - 20 000
	Agua	50 - 3 000	250 - 15 000
Convección con cambio de fase, ebullición.	Agua en ebullición	500 - 5 000	25 000 - 25 000
Convección con cambio de fase, condensación.	Vapor de agua en condensación	1 000 - 20 000	5 000 - 100 000
Aceites		10-120	60 -700

COEFICIENTE GLOBAL DE TRANSFERENCIA DE CALOR			
CORRIENTE CÁLIDA	CORRIENTE FRÍA	Btu/hr ft^2 °F	h,W/(m^2 °K)
Agua	Agua	140 – 280	86 - 1 400
Solventes Orgánicos	Agua	45 – 130	215 - 645
Gases	Agua	2.6 – 45	13 - 215
Aceites livianos	Agua	60 -160	300 - 770
Aceites pesados	Agua	10 – 45	50 - 215
Solventes Orgánicos	Aceites livianos	20 – 70	100 - 345
Agua	Salmuera	105 – 210	515 - 1 030
Solventes Orgánicos	Salmuera	26 – 90	130 - 430
Gases	Salmuera	2.6 – 45	13 - 215
Solventes Orgánicos	Solventes Orgánicos	20 – 62	100 - 300
Aceite pesado	Aceite pesado	8 – 44	40 - 215
Vapor	Agua	260 – 700	1 290 - 344
Vapor	Aceites livianos	44 – 140	215 - 690
Vapor	Aceite pesado	9 – 80	40 - 390
Vapor	Solventes Orgánicos	105 -210	515 - 1 030
Vapor	Gases	3.5 – 35	17 -170

Donde 1 Btu/ft^2 hr °F = 5.678 W/m^2 °K = 4.882 kcal/hr m^2°C

APÉNDICE G
TIPOS DE REDES NEURONALES ARTIFICIALES

Aprendizaje	Regla de Aprendizaje	Arquitectura	Algoritmo de Aprendizaje	Aplicaciones
Supervisado	Corrección del error	Perceptrón mono-capa o multicapa	Retropropagación Adaline y Madaline Perceptrones	Clasificación de patrones Aproximación de funciones Predicción, Control, etc.
	Boltzmann	Recurrente	Algoritmo de aprendizaje Boltzmann	Clasificación de patrones
	Hebbiana	Multicapa Feed-forward	Análisis lineal de discriminante	Análisis de datos Clasificación de patrones
	Competitiva	Competitiva	Aprendizaje de Vector de cuantización	Compresión de datos Categorización de clases
		Redes ART	Mapas ART	Categorización de clases Clasificación de patrones
No supervisado	Corrección del error	Multicapa Feed-forward	Proyección Sammon	Análisis de datos
	Hebbiana	Feed-Forward o Competitiva	Análisis de componentes principales	Análisis de datos Compresión de datos
		Red Hopfield	Aprendizaje de memoria asociativa	Memoria Asociativa
	Competitiva	Competitiva	Aprendizaje de Vector de cuantización	Categorización Compresión de datos
		Mapas de Kohonen	Mapas de Kohonen	Categorización Análisis de datos
		Redes ART	ART1, ART2	Categorización
	Corrección del error y Competitiva	Red RBF	Algoritmo de aprendizaje RBF	Clasificación de patrones Aproximación de funciones Predicción, Control, etc.

APÉNDICE H

FUNCIONES DE ENTRENAMIENTO CONTENIDAS EN EL TOOLBOX "NEURAL NETWORKS" DEL SOFTWARE MATLAB

- **traincgf:** Es una función que entrena redes multicapa con retropropagación, actualizando W y b de acuerdo al método de gradiente conjugado de Fletcher-Reeves el cual tiene los requerimientos más pequeños de almacenaje de todos los algoritmos de gradiente conjugado.

- **traincgp:** Es una función que entrena redes multicapa con retropropagación, actualizando W y b de acuerdo al método de gradiente conjugado de Polak-Ribiére el cual tiene los requerimientos de almacenaje ligeramente más grandes que el de Fletcher-Reeves, además, tiene una mayor capacidad de convergencia en algunos problemas.

- **traincgb:** Es una función que entrena redes multicapa con retropropagación, actualizando W y b de acuerdo al método de gradiente conjugado de Powell-Beale el cual tiene los requerimientos de almacenaje ligeramente más grandes que el de Polak-Ribiére, además, tiene una mayor velocidad de convergencia.

- **trainscg:** Es una función que entrena redes multicapa con retropropagación, actualizando W y b de acuerdo al método de gradiente conjugado escalado el cual es el único algoritmo de este tipo que no requiere línea de búsqueda, además de ser un algoritmo de entrenamiento de propósito general bastante bueno.

- **trainrp:** El algoritmo Resilient backpropagation introducido por Riedmiller y Braun en 1993, se basa en la estrategia de adaptar los parámetros del algoritmo de aprendizaje en el transcurso del mismo.

  Para ajustar el parametro se tiene en cuenta el signo de los gradientes anterior y actual, es decir, calcula el cambio para cada uno de los pesos de forma separada en función de la topología de la superficie del error.

- **trainbfg:** El método de Newton es una alternativa a los métodos de gradiente conjugado para una optimización rápida. El método de Newton converge a menudo más rápido que los métodos de gradiente conjugado. Por desgracia, es complejo y costoso para calcular la matriz hessiana de las redes neuronales feedforward. Hay una clase de algoritmos que se basa en el método de Newton, pero que no requiere el cálculo de los derivados del segundo. Estos son llamados métodos de secante.

- **trainoss:** Dado que el algoritmo BFGS requiere más espacio de almacenamiento y cómputo en cada iteración de los algoritmos de gradiente conjugado, existe la necesidad de una aproximación secante con menores requisitos de almacenamiento y computación. El paso secante (OSS) método es un intento de cerrar la brecha entre los algoritmos de gradiente conjugado y la cuasi-Newton (secante) algoritmos. Este algoritmo no almacena la matriz hessiana completo, se supone que en cada iteración, el anterior de Hesse fue la matriz de identidad. Esto tiene la ventaja adicional de que la dirección de búsqueda nueva se puede calcular sin calcular una matriz inversa.

- **traingdx:** Es una función de formación en red que se actualiza el peso y el sesgo de los valores de acuerdo con el impulso de pendiente de descenso y una tasa de aprendizaje adaptativo.

- **traingda:** Es una función de para cada iteración todos los vectores de entrenamiento.

- **traingd:** Es un algoritmo de aprendizaje supervisado que se usa para entrenar redes neuronales artificiales. El algoritmo consiste en minimizar un error (comúnmente cuadrático) por medio de descenso de gradiente, por lo que la parte esencial del algoritmo es cálculo de las derivadas parciales de dicho error con respecto a los parámetros de la red neuronal.

- **trainlm:** Es un método que interpola entre el algoritmo de Gauss-Newton y el método de gradiente descendiente. Es más robusto que el Gauss-Newton, lo que significa que en muchos casos se encuentra una solución, incluso si comienza muy lejos de los mínimos de final. Por buen comportamiento y las funciones razonables parámetros de partida, este

método tiende a ser un poco más lento que el Gauss-Newton. Levenberg-Marquardt también se puede ver como Gauss-Newton con una región de confianza enfoque.