Control de una planta no lineal de temperatura con redes neuronales

Control of a no lineal temperature plant with neural networks

Paula A. Ortiz Valencia[1] MSc, Alexander Arias Londoño[2] MSc.
1. Ingeniera en Instrumentación y Control. Docente de tiempo completo del ITM
2. Ingeniero Electrónico. Docente de tiempo completo del ITM.
paulaortiz@itm.edu.co; alexanderarias@itm.edu.co

Recibido para revisión 14 de Abril de 2009, aceptado 25 de Agosto de 2009, versión final 04 de Septiembre de 2009

Resumen— En este trabajo se presenta una comparación entre un control convencional y un control inteligente, los controladores desarrollados son un control PID y un control con redes neuronales para un sistema térmico, dentro del proyecto "Modulo didáctico de control - MODICO-". En la primera parte del artículo, se hace el planteamiento del modelo matemático y el modelo por medio de la identificación paramétrica de la planta de temperatura con mínimos cuadrados, estos modelos sirven para la caracterización de los controladores. La identificación es realizada con los datos tomados de la planta por medio de la tarjeta NI-6259 y la interfaz en labview. Para el control convencional, se diseña un control PID por medio del método del lugar geométrico de las raíces. El control neuronal se implementó con Redes Neuronales NARMA-L2, con cuatro neuronas, dos en la capa oculta, una en la capa de entrada y una en la capa de salida. La red neuronal es entrenada con seis algoritmos diferentes de entrenamiento, una vez entrenada la red, se selecciona el mejor método de entrenamiento y se valida el sistema con el control PID y el Control Neuronal.

Palabras Clave— Módulo Didáctico De Control, Identificación Paramétrica, Sensado, Control Neuronal, Control Pid.

Abstract— In this work a comparison appears between a conventional control and an intelligent control, the controllers developed are a control PID and a control with neural networks for a thermal system, inside the project "Didactic Control Module - MODICO". In the first part of the article, the exposition of the mathematical model of the plant is showed, and the model by means of parametric identification of the temperature plant is realized with least mean square, these models are used for the controller's characterization. The identification is realized taken information of the plant by means of the card NI -6259 card and the LabVIEW interface. For the conventional control, a control PID is designed by means of the method root locus. The control neuronal was implemented with NARMA-L2 neural networks; this net has four neurons, two in the secret layer, one in the layer of entry and one in the layer of exit. The network neuronal is trained with six different training algorithms, once the network trained, the best training method is selected and the system is validated with the control PID and the neural control.

Keywords— Control Didactic Module, Parametric Identification, Sensing, Neural Control, Pid Control.

I. INTRODUCCIÓN

Los sistemas de temperatura son necesarios en la mayoría de aplicaciones industriales. Los procesos manejados en las industrias tiene factores que influyen en los procesos y muchas veces no son tenidos en cuenta en los modelos lineales, además al ser trabajados en las zonas lineales se deben trabajar las variables sin la influencia de variables del contexto, por lo tanto en este trabajo se platea un modelo donde se analiza la no linealidad del proceso. Posteriormente se analiza el sistema con dos estrategias de control, un control PID convencional y un control con redes neuronales. Para el diseño del control PID convencional es necesario linealizar el sistema en un punto de operación, garantizando que el control trabaje bien en esta zona, pero cuando se aleja del punto de operación pierde efectividad el controlador, es por ello que es necesario diseñar un control que garantice el buen funcionamiento del control sin importar las no linealidades del proceso, por consiguiente se diseña un control con redes neuronales MPC (Model Predictive Control), esta estrategia se dio a conocer en primera instancia como Generalized Predictive Control en 1987 por D.W Clarke y sus colaboradores [1]. MPC ha sido una técnica control que ha sido fuente de investigación desde 1970, [14].

Las redes neuronales se caracterizan por su habilidad de aprender de los ejemplos en lugar de tener que programarse en un sentido convencional. Su uso posibilita que la dinámica de los sistemas complejos sea modelada y un control preciso sea logrado a través del entrenamiento, sin tener información a priori sobre los parámetros del sistema [12].

El trabajo está dividido en 5 secciones: En la primera sección se presentan las ecuaciones que rigen el sistema de temperatura y la identificación de las variables desconocidas por el método de mínimos cuadrados. En la segunda parte se muestra el software usado para la adquisición de los datos, las cuales posibilitaron la recolección de datos para el sistema de identificación. En la tercera parte se desarrolla un control convencional para el sistema linealizado en un punto de operación para el control PID. En la cuarta parte se diseña el control con redes neuronales y en la última sección se muestra el análisis de los resultados y las conclusiones.

II. MODELO MATEMÁTICO DEL SISTEMA A CONTROLAR

El modulo didáctico de control de temperatura consiste en un sistema cerrado similar al de una incubadora, con un área de 0.75 m^2 y un volumen de 0.1875 m^3, el material para la implementación del modulo es madera, color negro. A continuación se presenta el modelo matemático.

2.1. Ecuaciones Físicas del sistema

Las variables usadas en el modelo matemático se definen en la Tabla 1.

Tabla 1. Variables del sistema

Variable	Definición de las variables
H_e	Flujo de calor suministrado por el elemento de potencia
H_s	Flujo de calor en el sistema a controlar
H_M	Flujo de calor en el material (madera)
$\dot{Q}$	Velocidad del flujo de calor en el sistema a controlar
R_t	Resistencia térmica del material
R_{tb}	Resistencia térmica del bombillo
K_t	Conductividad térmica = $1/Rt$
C	Conductividad térmica
M	Masa del cuerpo
c	Calor especifico
T_a	Temperatura en el exterior del sistema (Temperatura ambiente)
T_s	Temperatura del sistema a controlar
T_b	Temperatura del elemento final de control

Ecuación de equilibrio térmico

$$H_e = H_s + H_M \quad (1)$$

H_M (Flujo de calor en el material (caja negra)): Se da por conducción, esta ley afirma que la velocidad de conducción de calor a través de un cuerpo por unidad de sección transversal es proporcional al gradiente de temperatura que existe en el cuerpo (con el signo cambiado). El factor de proporcionalidad se denomina conductividad térmica del material. Los materiales como el oro, la plata o el cobre tienen conductividades térmicas elevadas y conducen bien el calor, mientras que materiales como el vidrio o el amianto tienen conductividades cientos e incluso miles de veces menores; conducen muy mal el calor, y se conocen como aislantes. [2]

$$H_M = \frac{T_a - T_s}{R_t} \quad (2)$$

H_s (Flujo de calor en el sistema a controlar (aire seco)): Se da por convección, esta ley afirma que si existe una

diferencia de temperatura en el interior de un líquido o un gas, es casi seguro que se producirá un movimiento del fluido. Este movimiento transfiere calor de una parte del fluido a otra por un proceso llamado convección. El calentamiento de un sistema cerrado mediante un elemento generador de calor no depende tanto de la radiación como de las corrientes naturales de convección, que hacen que el aire caliente suba hacia el techo y el aire frío del resto del sistema se dirija hacia el elemento generador de calor. Debido a que el aire caliente tiende a subir y el aire frío a bajar, el elemento que suministra calor debe colocarse cerca del suelo (y los aparatos de aire acondicionado cerca del techo) para que la eficiencia sea máxima.

$$H_s = \frac{\dot{Q}}{A} \quad (3)$$

$$\dot{Q} = C\frac{d}{dt}T_s \quad (4)$$

$C = M * c$ Dónde . La masa del cuerpo es una relación entre el volumen y la densidad, la densidad depende de la temperatura y de la presión, en la Figura 1 se muestra la relación existente entre estas variables. [2]

El calor específico es la cantidad de calor necesaria para elevar la temperatura de una unidad de masa de una sustancia en un grado, esta depende de la temperatura, en la Figura 2 se muestra la relación existente entre estas dos variables.

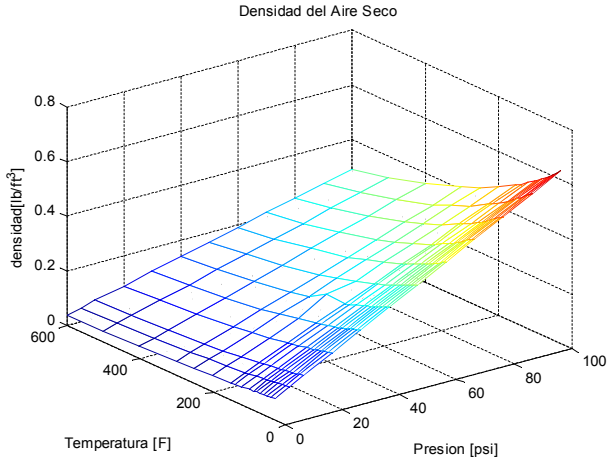

Figura 1. Densidad del aire seco en términos de la presión y la temperatura

Del sistema se mide la temperatura existente en la superficie de la planta y la presión y con ello se calcula la densidad y el calor especifico tomando los datos obtenidos en las Figura 1 y Figura 2.

Reemplazando (2), (3) y (4) en (1) se encuentra el modelo del sistema

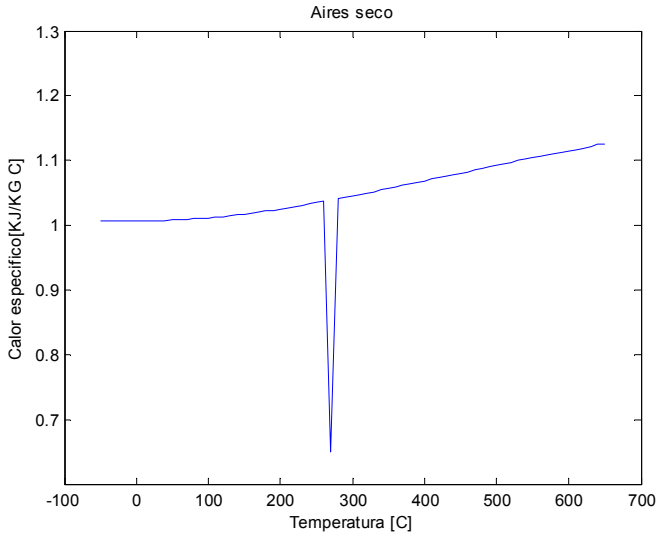

Figura 2. Calor especifico del aire seco vs la temperatura

$$H_e = \frac{C}{A}\frac{d}{dt}T_s + \frac{T_e - T_s}{R_t} \quad (5)$$

Donde R_t es un parámetro que depende T_a, T_s y H_e, este valor se calculó aplicando identificación de sistemas paramétrica por el método de mínimos cuadrados.

III. IDENTIFICCIÓN DE LA VARIABLE DESCONOCIDA R_t

3.1. Método de mínimos cuadrados

Este método es la base de distintos métodos paramétricos recursivos y no recursivos de identificación en el cual se trata de identificar los coeficientes θ_{ij} del sistema de ecuaciones propuesto en el modelo [7], [15]. Estas ecuaciones se representan como un sistema lineal:

$$\begin{aligned}
\hat{y}_1 &= \hat{\theta}_{11}x_1 + \hat{\theta}_{12}x_2 + \cdots + \hat{\theta}_{1r}x_n \\
\hat{y}_2 &= \hat{\theta}_{21}x_1 + \hat{\theta}_{22}x_2 + \cdots + \hat{\theta}_{2r}x_n \\
&\vdots \\
\hat{y}_r &= \hat{\theta}_{r1}x_1 + \hat{\theta}_{r2}x_2 + \cdots + \hat{\theta}_{rr}x_n
\end{aligned} \quad (6)$$

Donde r es el número de salidas del sistema y n es el número de entradas al sistema.

La ecuación (6) se puede representar como:

$$y = X\theta \quad (7)$$

$$z = X\theta + v \quad (8)$$

Donde $z=\begin{bmatrix} z(1) & z(2)\cdots z(n)\end{bmatrix}^T$ es el vector de la salida estimada del sistema, $\theta=\begin{bmatrix}\theta_0 & \theta_1\cdots\theta_n\end{bmatrix}^T$ es el vector de parámetros a estimar, $X=\begin{bmatrix}1 & \xi_1\cdots\xi_n\end{bmatrix}$ es la matriz de estados de la cual depende la señal de salida y $v=\begin{bmatrix} v(1) & v(2)\cdots v(n)\end{bmatrix}^T$ es el vector de la medición del error.

El objetivo de este método de identificación es minimizar la suma del error cuadrático cometido en k medidas, para ello se define el error como la diferencia entre el valor medido y el estimado, y se busca minimizar el índice de comportamiento J:

$$J=\frac{1}{2}(z-x\theta)^T(z-x\theta) \tag{9}$$

El valor de $\hat{\theta}$ que minimiza a $J(\theta)$ debe satisfacer que $\frac{\partial J}{\partial\theta}=0$ Derivando la ecuación (9) se tiene:

$$\frac{\partial J}{\partial\theta}=X^Tz+X^TX\hat{\theta} \tag{10}$$

$$\begin{aligned} X^Tz&=X^TX\hat{\theta}\\ X^T(z-X\hat{\theta})&=0\end{aligned} \tag{11}$$

Despejando $\hat{\theta}$ de la ecuación (11) se obtiene el valor estimado $\hat{\theta}$:

$$\hat{\theta}=(X^TX)^{-1}X^Tz \tag{12}$$

Con

$$E(v)=0 \qquad E(vv)=\sigma^2I \tag{13}$$

3.2. Pasos para la identificación

Para la aplicación de la técnica de identificación se deben seguir los siguientes pasos: Recolección de datos, Selección del modelo, Validación del modelo

Recolección de datos: En este caso se aplica una señal de excitación sobre la variable manipulada, PWM (Ver Figura 3), se registran los datos de las variables de salida (T_s, T_b, T_a). La respuesta obtenida se muestra de la Figura 4 a Figura 6.

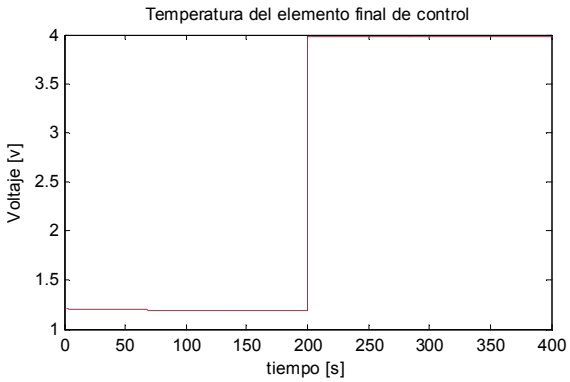

Figura 3. Señal de excitación

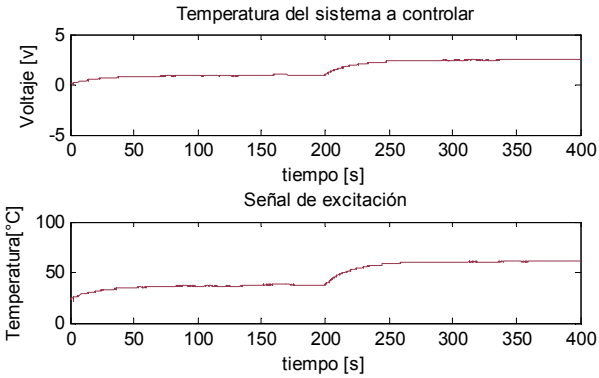

Figura 4. Respuesta del sistema

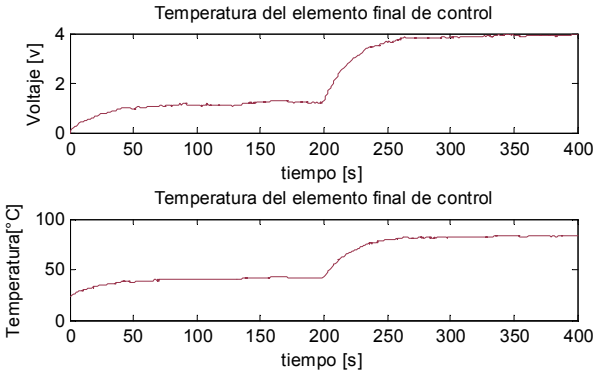

Figura 5. Respuesta del sistema calefactor

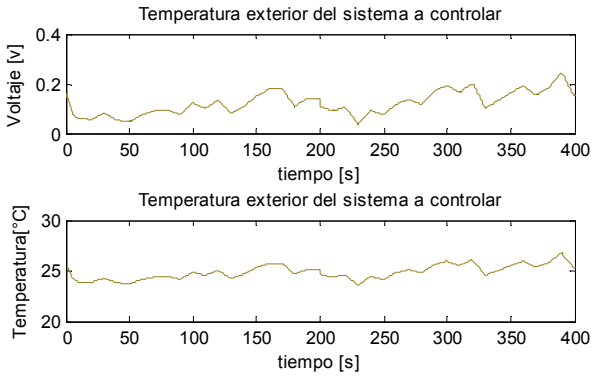

Figura 6. Temperatura en el exterior del sistema

Selección del modelo: Partiendo de las ecuaciones encontradas en el numeral 1, se calcula la variable desconocida R_t

• Con los datos obtenidos se determina la variable desconocida R_t, aplicando para ello un programa en Matlab. Para ello se despeja R_t de la ecuación (5)

• Se aplica el algoritmo de mínimos cuadrados

A continuación se explica el proceso seguido para implementar el algoritmo de mínimos cuadrados

• Se define la matriz de estado:

$$X = \begin{bmatrix} 1 & \dfrac{T_e(1) - T_s(1)}{H_e(1)} \\ \vdots & \vdots \\ 1 & \dfrac{T_e(n) - T_s(n)}{H_e(n)} \end{bmatrix} \tag{14}$$

• Se define el vector de parámetros a estimar:

$$\theta_1 = \begin{bmatrix} R_{t0} & R_{t(T_e - T_s)/H_e} \end{bmatrix} \tag{15}$$

• Se definen los vectores de las salidas estimadas, calculados a partir de los datos sensados.

$$Z_2 = \begin{bmatrix} R_t(1) & R_t(2) & \cdots & R_t(n) \end{bmatrix}^T \tag{16}$$

• Se calculan los parámetros a estimar usando las siguientes ecuaciones:

$$\hat{\theta}_1 = \left(X^T X\right)^{-1} X^T Z_1 \tag{17}$$

El resultado obtenido fue: $R_t = 0.78 + 0.2\dfrac{T_s - T_e}{H_e}$.

Validación del modelo: El resultado obtenido se muestra en el análisis de resultados.

IV. ADQUISICIÓN DE DATOS

Los datos mostrados en el inciso 3, son adquiridos por medio de la tarjeta NI USB-6259 [10]. El método de adquisición se muestra en la Figura 8 y su correspondiente implementación se muestra en la Figura 7. En el método de adquisición se utiliza la configuración de un canal como entrada análoga, se define el reloj para el muestreo, se muestrea continuamente hasta que el usuario para el modo de adquisición, al final se guarda todo en un archivo y se limpian todas las tareas y procesos abiertos en el sistema operativo [11].

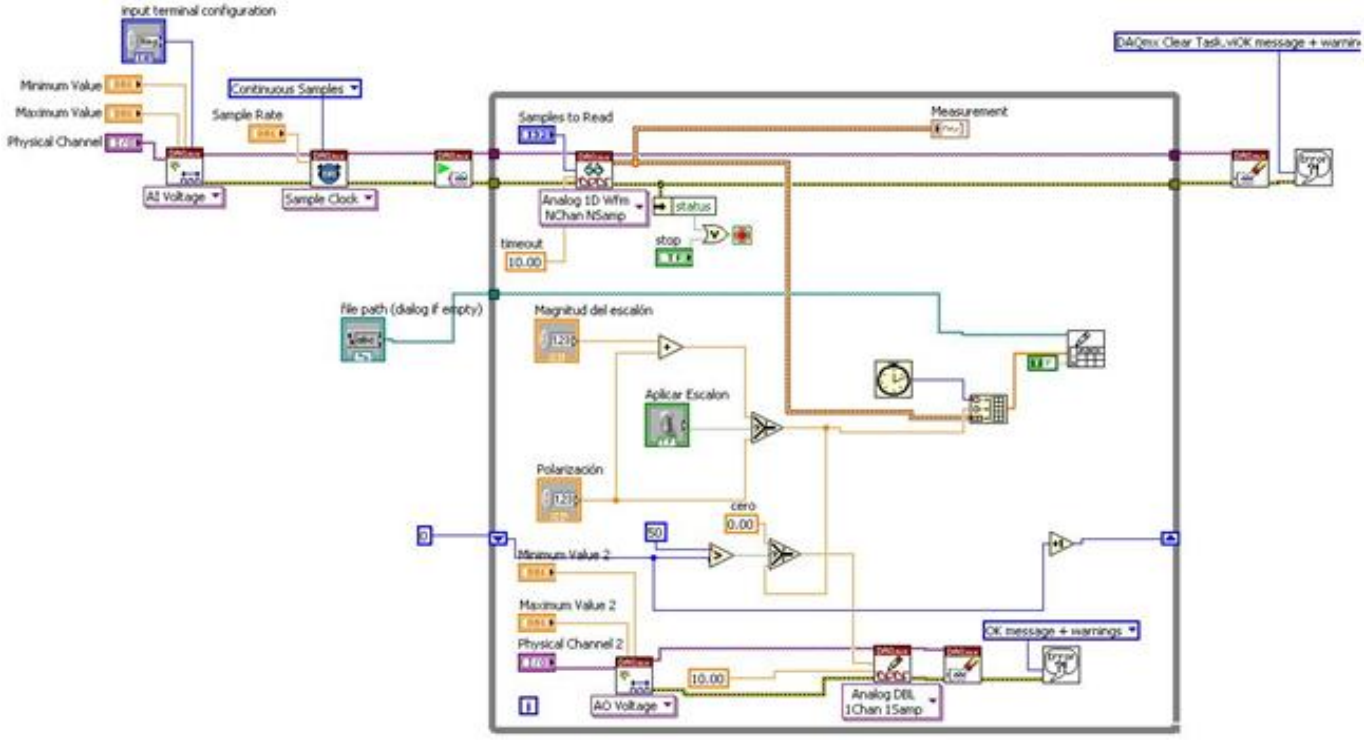

Figura 7. Adquisición de Datos con LabVIEW y la tarjeta NI USB-6259.

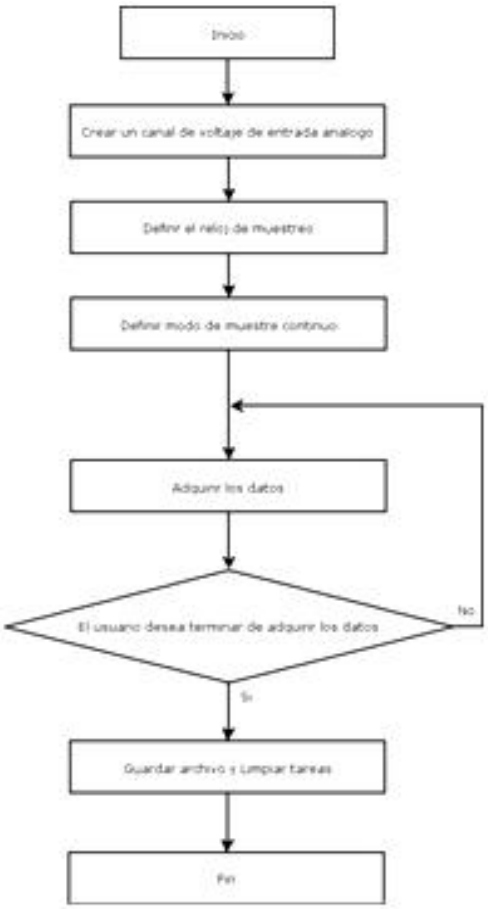

Figura 7. Método para la adquisición de datos.

V. DISEÑO DE CONTROLADORES

5.1. Control PID

Para la aplicación del control es necesario que el sistema sea lineal, por ello se procede a linealizar el sistema en un punto de operación. En este caso se desea que la temperatura alcance 23 ºC para ello se necesita que la señal de entrada sea de 1.1v. El sistema linealizado en el punto de operación es:

$$G_p(s) = \frac{0.9}{22s+1} \qquad (18)$$

El control se diseña aplicando técnicas del Lugar geométrico de las raíces, dando como resultado un control PI

$$G_c(s) = \frac{7.3521(s+0.2019)}{s} \qquad (19)$$

El comportamiento del control PID aplicado cerca al punto de operación se puede ver en la Figura 9, el sistema responde adecuadamente dentro de la zona de trabajo, donde se ha linealizado el sistema, presentando un sobre paso máximo de 5.2%, no presenta error en estado estable y un tiempo de establecimiento de 84 s.

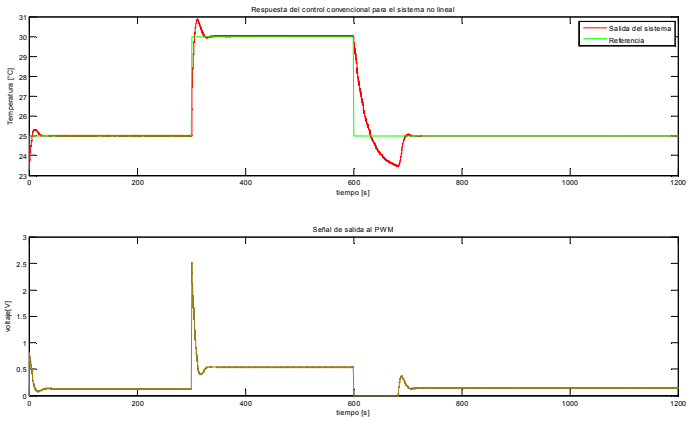

Figura 9. Respuesta del control convencional

5.2. REDES NEURONALES

Para el sistema no lineal se implementó un control con Redes Neuronales NARMA-L2, con 4 neuronas, dos en la capa oculta, una en la capa de entrada y una en la capa de salida. Este controlador es una variación de la linealización de un controlador realimentado. Un modelo de planta de temperatura no lineal es usado y mostrado en los incisos 2 y 3, la señal de control de entrada es ingresada a la red para forzar la salida a seguir la entrada de referencia. El modelo de la planta es entrenado con 6 métodos de entrenamientos diferentes que son: Gradiente Descendente (traingd), Gradiente Descendente con rata de aprendizaje adaptativa (traingda), Método BFGS quasi-Newton (trainbfg), Propagación hacia atrás con gradiente conjugado y actualización de Polack-Ribiere (traincp), Propagación hacia atrás secante de un paso (trainoss), Propagación hacia atrás de Levenberg Maquardt (trainlm). El control se reajusta al modelo de la planta y requiere pocas capacidades computacionales [8].

Antecedentes:

En los sistemas reales las no linealidades siempre aparecen, por ello las técnicas de control inteligente presentan ventajas al desarrollar control. El control es más robusto y se puede adaptar a un amplio rango de valores [6]. La perforación de hojas de metal es una actividad industrial muy aplicada para la fabricación en serie de partes metálicas y componentes en una gran variedad de industrias, en este proceso se han utilizado técnicas de control convencional y técnicas de control inteligente con redes neuronales. Las redes neuronales han mostrado una mejora en el control de este proceso [3]. Los algoritmos de control predictivo con redes neuronales son extensamente usados en la industria. La mayor parte de los algoritmos de control usan un modelo de proceso explícito para predecir el comportamiento futuro de una planta y debido a esto, el término el control predictivo de modelo (MPC) es utilizado a menudo. "El control MPC surge de la necesidad de superar los inconvenientes del control LQG, relacionados

con el manejo deficiente de las restricciones del proceso, incertidumbres del modelo y las no linealidades de los sistemas. El algoritmo MPC pretende optimizar el comportamiento futuro de la planta sobre un intervalo de tiempo (horizonte de predicción) a partir de la predicción de las salidas futuras del sistema para determinar el valor de las entradas sobre un horizonte de control, con el objeto de minimizar el error de las variables de salida respecto a su valor de referencia". El control con redes neuronales ha sido usado satisfactoriamente en procesos no lineales de intercambiadores de calor [9]. El análisis de sistemas de control de temperatura a nivel industrial es importante, además, estos sistemas presentan características como constantes de tiempo grandes y efectos de acople fuertes. En las máquinas de inyección de plástico se han diseñado controles PID con redes neuronales. Las neuronas se justan al sistema de temperatura multivariado [5].

Aplicación de la red al sistema no lineal

La red neuronal se simuló para cada método de entrenamiento mencionados anteriormente, buscando lograr la meta de un error de 10^{-3}. Una vez lograda la meta se detuvo la simulación. Sin embargo, se puso una limitante de 100 épocas ó iteraciones para alcanzar esta meta, así que en algunos casos, se llegó a las 100 épocas sin lograr el objetivo. En la Figura 10 se muestra el esquema usada para la simulación de la red y en la Figura 11 la configuración de la red.

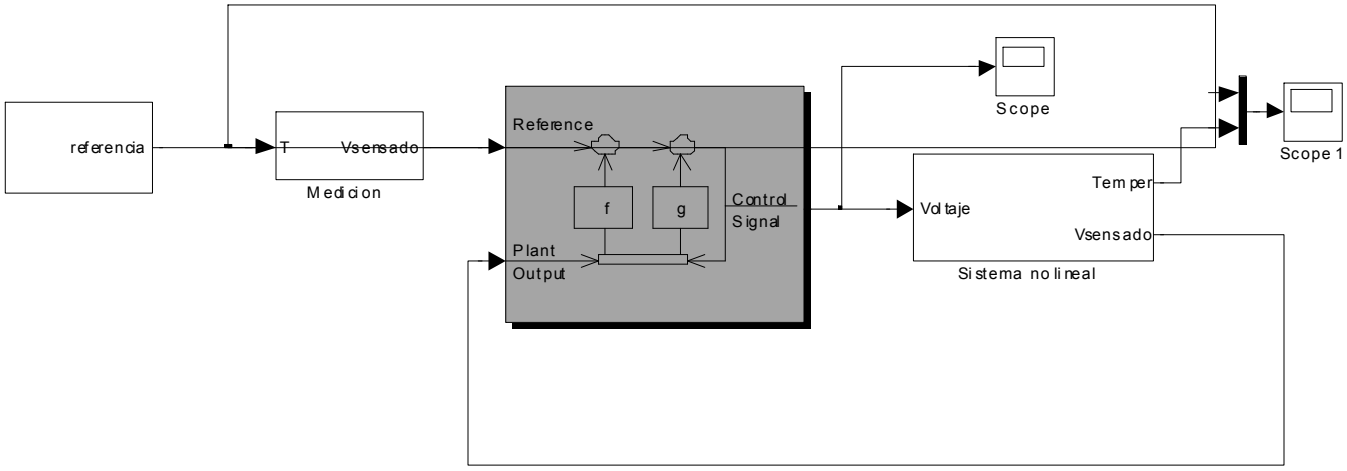

Figura 10. Red neuronal implementada en simulink.

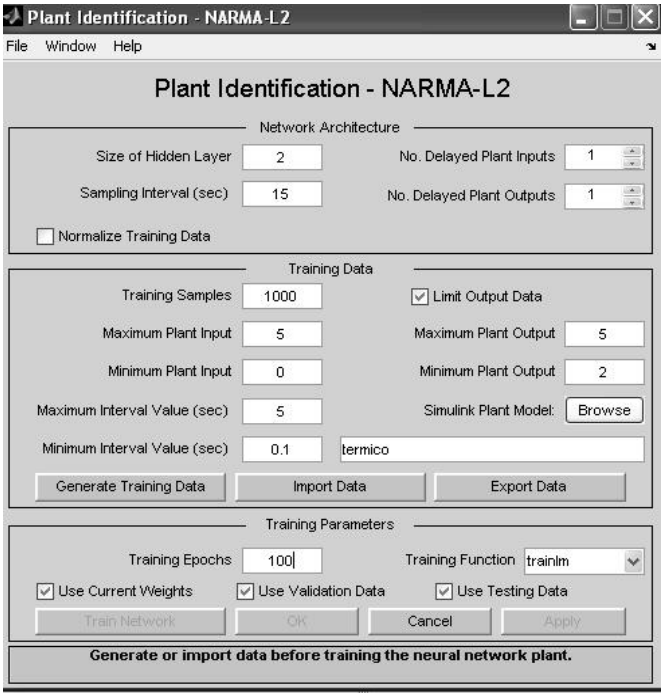

Figura 11. Configuración de la red

El resumen de los resultados obtenidos por cada método de entrenamiento se muestra En la Tabla 2 organizados según su rapidez de convergencia.

Tabla 2. Resumen de los resultados obtenidos con cada método.

Método	Número de épocas	¿Alcanzó la meta?	Error
trainlm	10	Sí	1.24153e-11
trainbfg	15	Sí	3.46837e-6
trainoss	37	No	0.0195382
traincgp	79	No	0.0152773
traingda	87	No	0.0104603
traingd	100	No	0.0626891

La respuesta del sistema ante los diferentes métodos de entrenamiento se muestra en la Figura 12

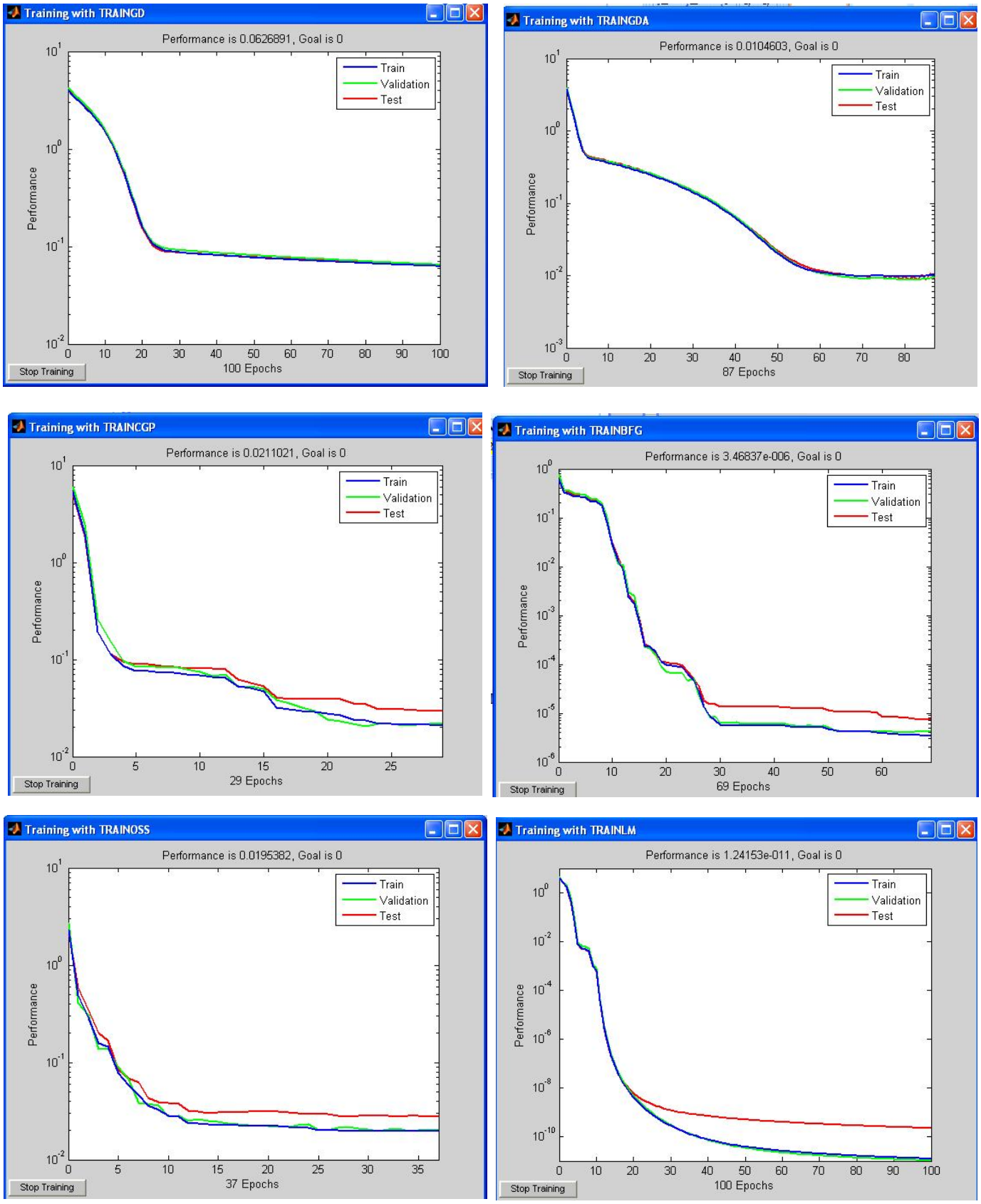

Figura 12. Respuesta obtenida usando los diferentes métodos de entrenamiento.

Explicación de los resultados obtenidos con los diferentes métodos de entrenamientos

A partir de la Tabla 2 puede verse que ninguno de los algoritmos de gradiente descendente logró la meta propuesta (traingd y traingda). La razón principal por la que estos métodos no lograron el error de 10^{-3}, se puede encontrar en el hecho de haber utilizado funciones de transferencia sigmoidales en la primera capa. Las funciones sigmoidales se caracterizan por que su pendiente se aproxima a cero a medida que la entrada se hace más grande. Esto ocasiona un problema cuando se utilizan gradientes descendentes para entrenar estas redes, pues el gradiente puede tener una magnitud muy pequeña y por lo tanto ocasionar pequeños cambios en los pesos y las bias, aún cuando estos están muy lejos de sus valores óptimos. Otra razón por la cual los algoritmos traingd y traingda son lentos, se debe a que utilizan técnicas heurísticas en vez de técnicas de optimización numérica.

Con el método de gradiente conjugado traincgp no se logró el objetivo, el sistema llegaba a un punto de estabilidad de convergencia más rápido que en el anterior método de entrenamiento, con una duración de 79 épocas. En los algoritmos de gradiente conjugado se hace una búsqueda a lo largo de direcciones conjugadas, lo que generalmente produce convergencias más rápidas que con los algoritmos de gradiente descendente. Dado que las diferencias entre los métodos de este grupo son pequeñas, es posible que se deban a que la inicialización de la red se hizo de manera aleatoria, lo cual pudo haber favorecido más a unos métodos que a otros.

Con el método de BFGS ó trainbfg, se obtuvo la segunda convergencia más rápida de todas (15 épocas), lo cual era de esperarse, ya que los métodos newtonianos generalmente convergen más rápido que los métodos de gradiente conjugado. La razón por la cual no es el más rápido de todos se encuentra en el hecho de que con este algoritmo se requiere calcular la matriz de Hessian en cada iteración, lo cual lo hace muy pesado computacionalmente y ello se traduce en menor rapidez de convergencia.

Un método que soluciona este problema es el algoritmo de Levenberg-Marquardt ó trainlm. Este algoritmo está diseñado para aproximar velocidades de entrenamiento de segundo orden sin necesidad de tener que calcular la matriz de Hessian. Esta es la razón por la cual este algoritmo fue significativamente el más rápido de los seis utilizados, logrando la meta en tan solo 10 épocas, aunque se dejo corriendo las 100 épocas para ver el alcance de convergencia y el error, con el cual se logró un error mínimo de 1.24153e-11.

Por último, con el algoritmo de Propagación hacia atrás secante de un paso trainoss se esperaba un buen desempeño, casi tan bueno como el de trainlm, puesto que es un algoritmo quasi-newtoniano, que no almacena completamente la matriz de Hessian en cada iteración. Sin embargo éste no logro alcanzar el objetivo.

A continuación se muestra el algoritmo Levenberg-Marquardt [13], para una red neuronal de dos capas como la que se muestra en la Figura 13. La primera capa es sigmoidal y la segunda es lineal.

Para la red neuronal mostrada en la Figura 13, se tiene que:

$$x=\begin{bmatrix}1\\x_1\\x_2\\\vdots\\x_n\end{bmatrix} \quad y=\begin{bmatrix}y_1\\y_2\\\vdots\\y_m\end{bmatrix} \quad a^1=\begin{bmatrix}a_1^1\\a_2^1\\\vdots\\a_{n_1}^1\end{bmatrix} \quad a^2=\begin{bmatrix}a_1^2\\a_2^2\\\vdots\\a_m^2\end{bmatrix} \qquad (20)$$

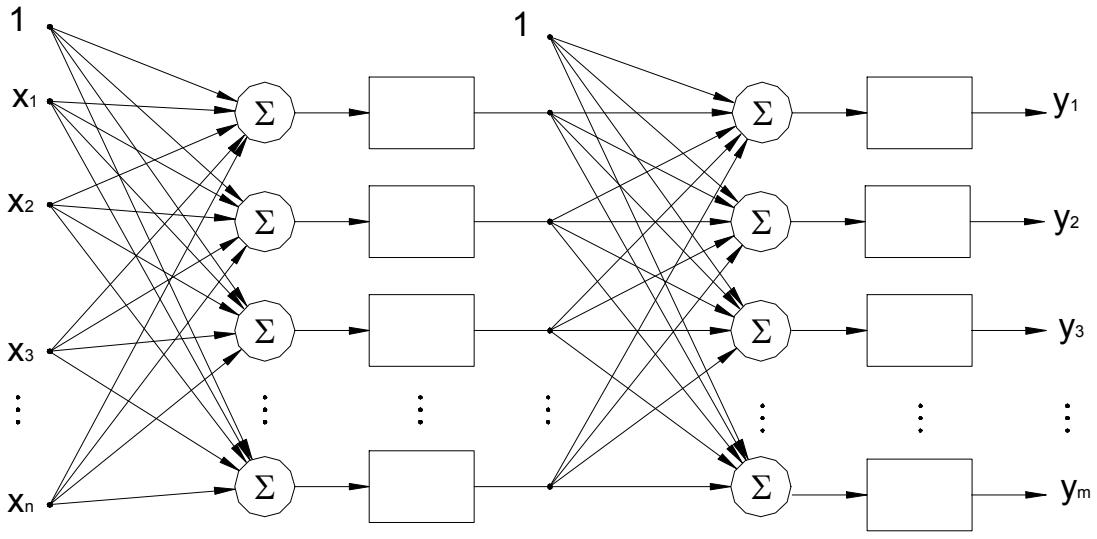

Figura 13. Red neuronal que se entrenará con el algoritmo de Levenberg-Marquardt. La primera capa es sigmoidal y la segunda es lineal.

$$f^1\left(a^1\right)=\begin{bmatrix}1\\ f_1^1\left(a^1\right)\\ f_2^1\left(a^1\right)\\ \vdots\\ f_{n_1}^1\left(a^1\right)\end{bmatrix}\quad f^2\left(a^2\right)=\begin{bmatrix}1\\ f_1^2\left(a^2\right)\\ f_2^2\left(a^2\right)\\ \vdots\\ f_m^2\left(a^2\right)\end{bmatrix} \qquad (21)$$

$$W^1=\begin{bmatrix} w_{01}^1 & w_{02}^1 & \cdots & w_{0n_1}^1\\ w_{11}^1 & w_{12}^1 & \cdots & w_{1n_1}^1\\ \vdots & \vdots & \ddots & \vdots\\ w_{n1}^1 & w_{n2}^1 & \cdots & w_{nn_1}^1\end{bmatrix}\quad W^2=\begin{bmatrix} w_{01}^2 & w_{02}^2 & \cdots & w_{0m}^2\\ w_{11}^2 & w_{12}^2 & \cdots & w_{1m}^2\\ \vdots & \vdots & \ddots & \vdots\\ w_{n_1 1}^2 & w_{n_1 2}^2 & \cdots & w_{n_1 m}^2\end{bmatrix} \qquad (22)$$

Donde w_{ij}^1 es el factor de peso para la capa 1, de la entrada i y la neurona j.

La ecuación que define la red neuronal es:

$$y=f^2\left(W^{2\mathrm{T}}\,f^1\left(W^{1\mathrm{T}}x\right)\right) \qquad (23)$$

Para el entrenamiento de la red neuronal, la función objetivo es la función de error del tipo:

$$J\left(\Theta\right)=\frac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{m}\left(d_{ij}-y_{ij}\right)^2 \qquad (24)$$

d_{ij} Es la salida deseada en la neurona j, para un vector de entradas x_k.

y_{ij} Es la salida real de la neurona de salida j, para un vector de entradas x_k.

m Es el número de neuronas de salida (igual al número de salidas de la red neuronal)

p Es el número total de patrones de entrenamiento utilizados para entrenar la red.

$$\Theta=\left[\,w_0^{1T}\quad w_1^{1T}\quad\cdots\quad w_{n_1}^{1\,T},\quad w_0^{2T}\quad w_1^{2T}\quad\cdots\quad w_m^{2\,T}\right]\in\Box^{S} \qquad (25)$$

$$w_i^{1T}=\begin{bmatrix}w_{0i}^1\\ w_{1i}^1\\ \vdots\\ w_{ni}^1\end{bmatrix}$$ Son los factores de peso para todas las entradas y la neurona i.

S Es el número total de elementos en Θ, y es igual a:

$$S=\left(n+1\right)\cdot n_1+\left(n_1+1\right)\cdot m \qquad (26)$$

Donde: n es el número de entradas en la red neuronal.

m Es el número de salidas de la red neuronal.

n_1 Es el número de neuronas de la capa oculta

La ecuación (24) se puede escribir también así:

$$J\left(\Theta\right)=\frac{1}{2}\varepsilon^{\mathrm{T}}\varepsilon \qquad (27)$$

Donde

$$\varepsilon=\left[\,e_1^1\quad\cdots\quad e_m^1\;\middle|\;e_m^2\quad\cdots\quad e_m^2\;\middle|\;\cdots\;\middle|\;e_m^p\quad\cdots\quad e_m^p\,\right]^{\mathrm{T}} \qquad (28)$$

Con $e_j^i=d_{ij}-y_{ij}$

El problema es encontrar un valor de Θ tal que $J\left(\Theta\right)$ sea lo más pequeño posible. Se comienza entonces por linealizar ε alrededor de un valor de Θ, utilizando series de Taylor hasta la primera derivada:

$$\hat{\varepsilon}\left(\Theta\right)=\varepsilon\left(\Theta^k\right)+\left.\frac{\partial\varepsilon}{\partial\Theta}\right|_{\Theta=\Theta^k}\cdot\left(\Theta-\Theta^k\right) \qquad (29)$$

La ecuación (29) se puede reescribir como:

$$\hat{\varepsilon}\left(\Theta\right)=\varepsilon\left(\Theta^k\right)-\left.\frac{\partial\varepsilon}{\partial\Theta}\right|_{\Theta=\Theta^k}\cdot\Theta^k+\left.\frac{\partial\varepsilon}{\partial\Theta}\right|_{\Theta=\Theta^k}\cdot\Theta \qquad (30)$$

Sea $y=\varepsilon\left(\Theta^k\right)-A\cdot\Theta^k$ y $x=W$.

Con $A=-\left.\frac{\partial\varepsilon}{\partial\Theta}\right|_{\Theta=\Theta^k}$

Entonces se tiene que

$$\hat{\varepsilon}\left(\Theta\right)=y-A\cdot x \qquad (31)$$

Reemplazando (31) en (27):

$$J\left(\Theta\right)=\frac{1}{2}\left(y-A\cdot\Theta\right)^{\mathrm{T}}\left(y-A\cdot\Theta\right) \qquad (32)$$

Este es un problema de mínimos cuadrados cuya solución es:

$$\Theta^{k+1}=\Theta^k-\left(\frac{\partial\varepsilon}{\partial\Theta}^{\mathrm{T}}\frac{\partial\varepsilon}{\partial\Theta}\right)^{-1}\cdot\frac{\partial\varepsilon}{\partial\Theta}\cdot\varepsilon\left(\Theta^k\right) \qquad (33)$$

Para aplicar el método de Levenberg-Marquardt se hace el siguiente cambio:

$$\Theta^{k+1} = \Theta^k - \left(\frac{\partial \varepsilon}{\partial \Theta}^{\mathrm{T}} \frac{\partial \varepsilon}{\partial \Theta} + \mu I \right)^{-1} \cdot \frac{\partial \varepsilon}{\partial \Theta} \cdot \varepsilon\left(\Theta^k\right) \quad (34)$$

Donde μ se conoce como parámetro de entrenamiento. También suele utilizarse la siguiente ecuación:

$$\Theta^{k+1} = \Theta^k - \left(\frac{\partial \varepsilon}{\partial \Theta}^{\mathrm{T}} \frac{\partial \varepsilon}{\partial \Theta} + \mu D \right)^{-1} \cdot \frac{\partial \varepsilon}{\partial \Theta} \cdot \varepsilon\left(\Theta^k\right) \quad (35)$$

Donde D es la diagonal de $\frac{\partial \varepsilon}{\partial \Theta}^{\mathrm{T}} \frac{\partial \varepsilon}{\partial \Theta}$.

Para desarrollar el algoritmo de Levenberg-Marquardt para la red neuronal de la Figura 13, es necesario entonces calcular la derivada $\frac{\partial \varepsilon}{\partial \Theta}$, que equivale a una matriz de la siguiente forma:

$$\frac{\partial \varepsilon}{\partial \Theta} = \begin{bmatrix} \frac{\partial \varepsilon(1)}{\partial \Theta} \\ \frac{\partial \varepsilon(2)}{\partial \Theta} \\ \vdots \\ \frac{\partial \varepsilon(p)}{\partial \Theta} \end{bmatrix} \quad (36)$$

Recordar que p es el número de parámetros de entrenamiento.

$$\frac{\partial \varepsilon(k)}{\partial \Theta} = \begin{bmatrix} \frac{\partial \varepsilon_1(k)}{\partial \Theta_1} & \frac{\partial \varepsilon_1(k)}{\partial \Theta_2} & \cdots & \frac{\partial \varepsilon_1(k)}{\partial \Theta_S} \\ \frac{\partial \varepsilon_2(k)}{\partial \Theta_1} & \frac{\partial \varepsilon_2(k)}{\partial \Theta_2} & \cdots & \frac{\partial \varepsilon_2(k)}{\partial \Theta_S} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \varepsilon_m(k)}{\partial \Theta_1} & \frac{\partial \varepsilon_m(k)}{\partial \Theta_2} & \cdots & \frac{\partial \varepsilon_m(k)}{\partial \Theta_S} \end{bmatrix} \text{con } k = 1, 2, ..., p \quad (37)$$

$$\frac{\partial \varepsilon(k)}{\partial \Theta} = \begin{bmatrix} \frac{\partial \varepsilon(k)}{\partial \Theta_1} & \frac{\partial \varepsilon(k)}{\partial \Theta_2} & \cdots & \frac{\partial \varepsilon(k)}{\partial \Theta_S} \end{bmatrix} \quad (38)$$

Recordar que S es el número total de elementos en Θ.

Para un parámetro de entrenamiento k

$$\frac{\partial \varepsilon}{\partial \Theta_j} = \frac{\partial \varepsilon}{\partial y} \cdot \frac{\partial y}{\partial a^2} \cdot \frac{\partial a^2}{\partial \Theta_j} \text{ para } \left(n+1\right) \cdot n_1 < j \leq S \ (\ \Theta_j = w^2_{rs}) \quad (39)$$

$$\frac{\partial \varepsilon(k)}{\partial \Theta_j} = \frac{\partial \varepsilon}{\partial y} \cdot \frac{\partial y}{\partial a^2} \cdot \frac{\partial a^2}{\partial f^1} \cdot \frac{\partial f^1}{\partial a^1} \cdot \frac{\partial a^1}{\partial \Theta_j} \text{ para } 1 \leq j \leq \left(n+1\right) \cdot n_1 \ (\ \Theta_j = w^1_{pq}) \quad (40)$$

Θ_j Corresponde la entrada j del vector Θ, que es igual a la entrada (p, q) de la matriz de pesos w^1 o a la entrada (r, s) de la matriz w^2, dependiendo del valor de j.

Se desarrollarán las ecuaciones (39) y (40), según el caso:

Caso 1: $\Theta_j = w^2_{rs}$ Ecuación (39)

$$\frac{\partial \varepsilon}{\partial y} = -\begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} \quad (41)$$

$$\frac{\partial y}{\partial a^2} = \begin{bmatrix} \dot{f}^2_1\left(a^2_1\right) & 0 & \cdots & 0 \\ 0 & \dot{f}^2_2\left(a^2_2\right) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dot{f}^2_m\left(a^2_m\right) \end{bmatrix} \quad (42)$$

$$\frac{\partial a^2}{\partial \Theta_j} = \begin{bmatrix} 0 \\ \vdots \\ f^1_r \\ \vdots \\ 0 \end{bmatrix} \rightarrow \textit{fila} \quad s \quad (43)$$

Finalmente:

$$\frac{\partial \varepsilon}{\partial \Theta_j} = -\dot{f}^2_s\left(a^2_s\right) \cdot f^1_r\left(a^1_r\right) \quad (44)$$

Caso 2: $\Theta_j = w^1_{pq}$ Ecuación (40)

En el caso anterior ya se calcularon $\frac{\partial \varepsilon}{\partial y}$ y $\frac{\partial y}{\partial a^2}$.

Se calculan las derivadas que hacen falta:

$$\frac{\partial a^2}{\partial f^1} = w^{2T} \quad (45)$$

$$\frac{\partial f_1}{\partial a^1} = \begin{bmatrix} \dot{f}_1^1\left(a_1^1\right) & 0 & \cdots & 0 \\ 0 & \dot{f}_2^1\left(a_2^1\right) & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \dot{f}_{n_1}^1\left(a_{n_1}^1\right) \end{bmatrix} \quad (46)$$

$$\frac{\partial a^1}{\partial \Theta_j} = \begin{bmatrix} 0 \\ \vdots \\ x_p \\ \vdots \\ 0 \end{bmatrix} \rightarrow fila \quad q \quad (47)$$

$$\frac{\partial \varepsilon(k)}{\partial \Theta_j} = -\left[\, \dot{f}_1^2\left(a_1^2\right) \quad \dot{f}_2^2\left(a_2^2\right) \quad \cdots \quad \dot{f}_m^2\left(a_m^2\right)\right] \left[\, w_{q1}^1 \dot{f}_q^1\left(a_q^1\right) x_p \quad w_{q2}^1 \dot{f}_q^1\left(a_q^1\right) x_p \quad \cdots \quad w_{qm}^1 \dot{f}_q^1\left(a_q^1\right) x_p \,\right]^T \quad (48)$$

Finalmente, dado que se conoce el tipo de funciones que tendrá cada capa, se pueden hallar las derivadas $\dot{f}_i^1\left(a_i^1\right)$ y $\dot{f}_i^2\left(a_i^2\right)$:

$$f_i^1\left(a_i^1\right) = \frac{1}{\left(1+e^{-a_i}\right)} \quad (49)$$

$$\dot{f}_i^1\left(a_i^1\right) = \frac{e^{-a_i}}{\left(1+e^{-a_i}\right)^2} \quad (50)$$

$$f_i^2 = a_i^2 \quad (51)$$

$$\dot{f}_i^2 = 1 \quad (52)$$

Entonces la ecuación (44) queda así:

$$\frac{\partial \varepsilon}{\partial \Theta_j} = f_r^1\left(a_r^1\right) \quad (53)$$

Y la ecuación (48) queda así:

$$\frac{\partial \varepsilon(k)}{\partial \Theta_j} = -\left(w_{q1}^1 \dot{f}_q^1\left(a_q^1\right) x_p + w_{q2}^1 \dot{f}_q^1\left(a_q^1\right) x_p + \cdots + w_{qm}^1 \dot{f}_q^1\left(a_q^1\right) x_p \right) \quad (54)$$

Así restaría reemplazar los resultados de (53) o (54) (según sea el caso) en (38), y calcular los valores para cada uno de los parámetros de entrenamiento de la red neuronal, para poder entrenar la red a partir de la ecuación (34).

Verificación de la red en el sistema a controlar

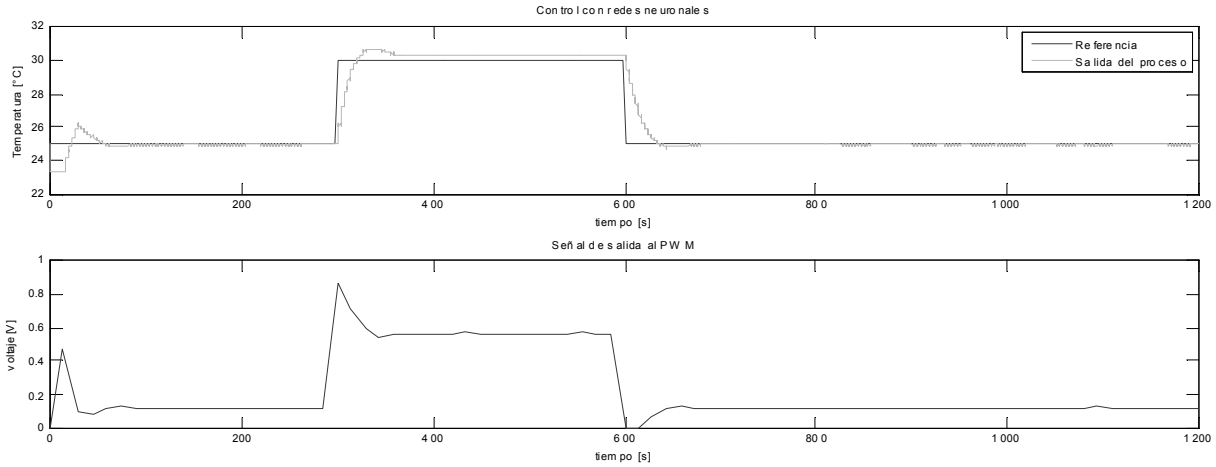

Figura 14. Respuesta del sistema con la red neuronal

El comportamiento del sistema lo observamos en la Figura 14, se puede ver que el sistema responde adecuadamente ante los cambios de los escalones, en el escalón de 300 s a 600 s presentando un sobre paso máximo de un 2% , un error en estado estable del 2% y el tiempo de establecimiento de 31 s. En la gráfica inferior de la Figura 14, se presenta la respuesta del actuador o elemento final de control, el cual no presenta efecto wind-up, y tiene un comportamiento suave, este hace que los elementos tengan mayor durabilidad lo cual a nivel industrial se traduce en economía de gastos de operación y mantenimiento.

Análisis de resultados

Modelo matemático identificado con mínimos cuadrados: Para analizar si el modelo obtenido es confiable, se procede a realizar dos pruebas, la primera es la verificación del modelo

en la cual se compara los datos obtenidos reales con los datos obtenidos con el sistema identificado, el resultado se muestra en la Figura 15, En la Figura 16 se muestra el error obtenido entre la señal estimada y la señal encontrada con los datos reales, medidos en el sistema, con estos valores se calcula el índice de desempeño, J= 0.0137.

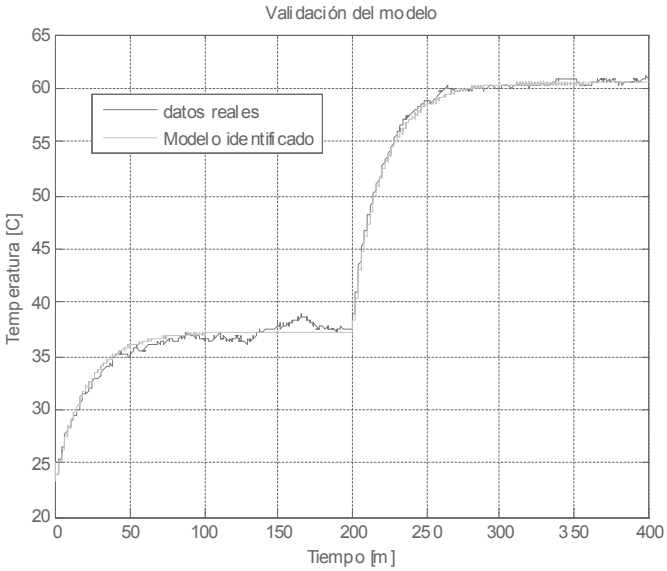

Figura 15. Resultado de la identificación paramétrica

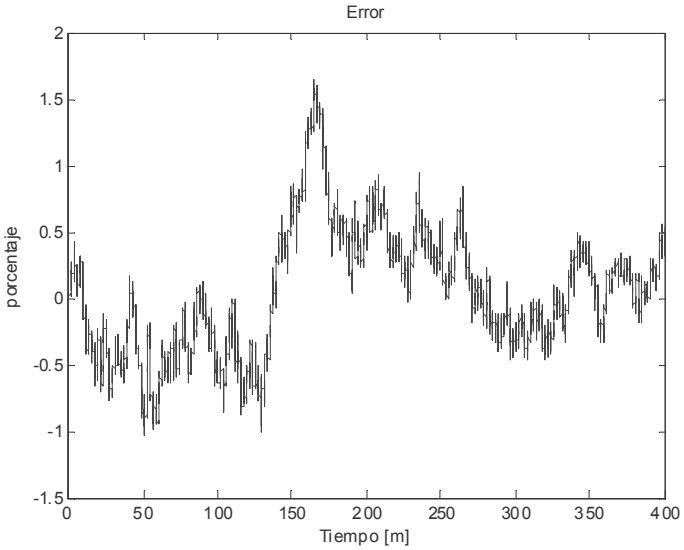

Figura 16. Error entre el valor real y estimado

6. La segunda prueba es realizar un comprobación del modelo, para ello se aplica otra señal de excitación diferente a la que se identifico el modelo y se compara con el sistema construido, el resultado se puede ver en la Figura 17, con estos valores se calcula el índice de desempeño, J= 0.03. Como era de esperarse el índice de desempeño aumenta, pero sigue siendo un índice muy bueno, por lo tanto se puede decir que el sistema es confiable.

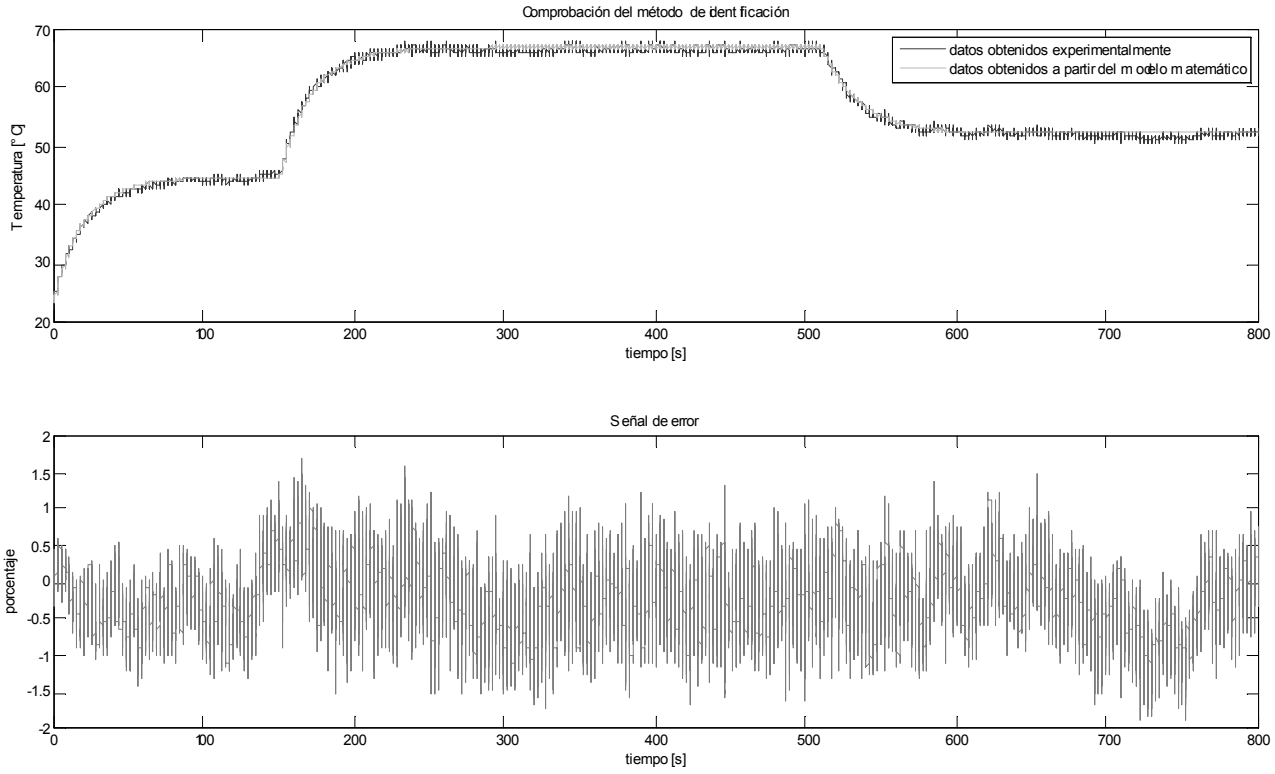

Figura 17. Comprobación del método de identificación

• Comparación entre el control convencional y el control con redes neuronales

Para analizar los resultados se valida el sistema ante la misma señal de entrada tanto para el control convencional como el diseñado con redes neuronales, el resultado obtenido se muestra en la Figura 18, aunque los resultados obtenidos por los dos controladores es buena, el control con redes presenta un mejor comportamiento, mejorando la estabilidad y disminuyendo el tiempo de respuesta del sistema. El resumen de la respuesta obtenida por los controladores diseñados se muestra en la Tabla 3.

Control	Ess(%)	Ts (s)	Mp(%)	J
Redes	5	65	2.5	0.025
PID	0	84	5.2	0.085

Tabla 3. Comparación de los controles diseñados

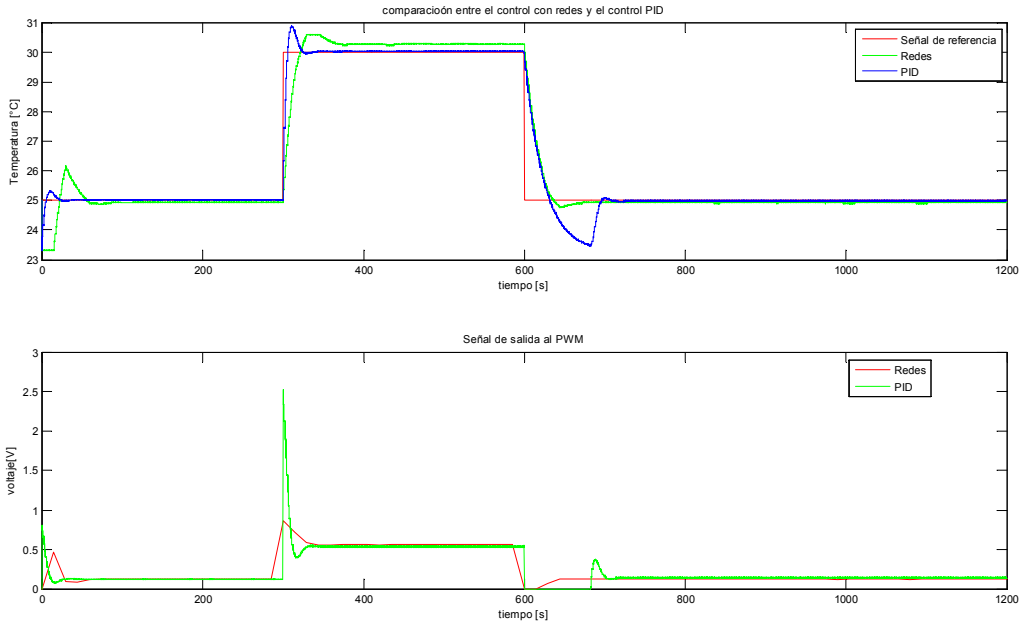

Figura 18. Comparación entre la red neuronal y el control convencional

VII. CONCLUSIONES

El modelo matemático y los controladores diseñados en este proyecto sirven como base para los procesos de enseñanza de control con plantas industriales dada su presentación teórico-práctica, la cual va desde la aproximación a problemas reales hasta la identificación y control aplicando simulación y modelamiento matemático.

El proyecto MODICO ha servido para plantear una nueva metodología de trabajo para los futuros diseñadores de controladores en la ciudad de Medellín, esto, gracias a la didáctica planteada en el Instituto Tecnológico Metropolitano y el modelo pedagógico basado en competencias y aplicación de nuevas tecnologías en procesos de control.

El control neuronal NARMA-L2 mostró mejor desempeño que los controladores convencionales PID, principalmente en el estado estacionario. Este tipo de controladores evita en gran medida el efecto campana sobre los actuadores de control.

El control convencional es una buena solución para un sistema, siempre y cuando este se trabaje cerca de la zona de trabajo, es decir, cerca del punto donde se linealizó el sistema.

Para un sistema no lineal el control convencional PID funciona adecuadamente para el punto de operación en el cual fue linealizado, al alejarse de este punto pierde confiabilidad el control.

El modelo del sistema térmico se modelo por medio de los datos obtenidos en la tarjeta de adquisición NI-6259 y LabVIEW, la simulación se realizó en Matlab y Simulink, lo cual permitió validar los controladores propuestos y analizar la dinámica del sistema.

REFERENCIAS

[1] W, Clarke, Mohtadi, C. and P. S. Tuffs, 1984. Generalized Predictive Control.

[2] Tools and Basic Information for Design, Engineering and Construction of Technical Applications, En: http://www.engineeringtoolbox.com/air-temperature-pressure-density-d_771.html Consulta: Marzo, 2009.

[3] Slomp, J. and Klingenberg, W., 2004. A Proposal to Use Artificial Neural Networks for Process Control of Punching/Blanking Operations. Flexible

Automation and Intelligent Manufacturing. FAIM2004, Toronto, Canada.

[4] Journal Code: X0014A. 1999 Diseño de la red neuronal. Univ. of Osaka. VOL.43rd; NO.; PAGE. 287-288

[5] Huailin, S. and Youguo, P., 2005. Decoupled Temperature Control System Based on PID Neural Network. En: ACSE 05 Conference, 19-21 December 2005, CICC, Cairo, Egypt.

[6] Kawasaki, M., 1999. Control of Temperature of Plant by Neural-Network. En: Proceedings of the Annual Conference of the Institute of Systems, Control and Information Engineers. VOL.43rd; pp. 287-288

[7] Lyashevskiy, S. andChen, Y., 1996. Nonlinear Identification of Aircraft , En: IEEE International Conference.

[8] Hagan, M., Demuth, H. and De Jesus, O., 2002. An Introduction to the Use of Neural Networks in Control Systems. En: International Journal of Robust and Nonlinear Control, Vol. 12, No. 11, pp. 959-985.

[9] Jalili-Kharaajool, M. and Babak, N. 2004., s.a. Neural network based predictive control of a heat exchanger nonlinear process. En: Journal of electrical & electronics engineering. Vol. 4. pp. 1219-1226.

[10] Manual NI USB-6259. Disponible en: http://sine.ni.com/ds/app/doc/p/id/ds-20/lang/en. Consultado el 26 de Marzo de 2009.

[11] NI USB-622x/625x OEM. USER GUIDE. M Series USB-6221/6225/6229/6251/6255/6259 OEM Devices. Disponible en: http://www.ni.com/pdf/products/us/371910a.pdf . Consultado el 26 de Marzo de 2009.

[12] Parker, 1995. Optimal algorithms for adaptive networks: Second order backpropagation, second order direct propagation and second order Hebbian learning. En: D.B. Parker IEEE 1st Int. Conf. on Neural Networks, vol.2, pp.593- 600, San Diego, CA.

[13] Ranga, N., Deodhare, D. and Nagabhushan, P., 2002. Parallel Levenberg-Marquardt-basede Neural Network Training on Linux Clusters Disponible en: http://www.ee.iitb.ac.in/~icvgip/PAPERS/248.pdf

[14] Soloway, D. and J. Haley, P., 1997. Neural Generalized Predictive Control, A Newton-Raphson Implementation. NASA Langley Research Center. Hampton : s.n.,

[15] Vélez, C.M., 2003. Identificación en Línea y Simulación de Sistemas Dinámicos, En: II Congreso de la Asociación Colombiana de Automática, Bucaramanga, pp. 69-73.

[16] Xu, J., Daniel, H.O., and Zheng, Y., 2004. A constructive algorithm for feedforward neural networks. En: Institute fo System Science, East China Normal University. 6 p. Disponible en: http://www.ee.unimelb.edu.au/conferences/ascc2004/proceedings/papers/P97.pdf

Universidad Nacional de Colombia Sede Medellín
Facultad de Minas
Escuela de Ingeniería de Sistemas

Grupos de Investigación

Grupo de Investigación en Sistemas e Informática

Categoría A de Excelencia Colciencias
2004 - 2006 y 2000.

GIDIA: Grupo de Investigación y Desarrollo en Inteligencia Artificial

Categoría A de Excelencia Colciencias
2006 – 2009.

Grupo de Ingeniería de Software

Categoría C Colciencias 2006.

Grupo de Finanzas Computacionales

Categoría C Colciencias 2006.

Centro de Excelencia en Complejidad

Colciencias 2006

Escuela de Ingeniería de Sistemas
Dirección Postal:
Carrera 80 No. 65 - 223 Bloque M8A
Facultad de Minas. Medellín - Colombia
Tel: (574) 4255350 Fax: (574) 4255365
Email: esistema@unalmed.edu.co
http://pisis.unalmed.edu.co/