Redes neuronales para la estabilización simultánea con múltiples dominios acotados de estabilidad en control de procesos

Francisco J. Arteaga B.[1], Doris E. Sevilla H.[1], Pablo A. Franco N.[1], Raymundo Méndez[1], Guy O. Beale[2]
[1]Unidad de Investigación en Automatización Industrial,
Escuela de Ingeniería Eléctrica, Dpto. de Sistemas y Automática,
Universidad de Carabobo, Valencia, Venezuela. E-mail: farteaga@uc.edu.ve
[2] Electrical and Computer Engineering, George Mason University, Fairfax, Virginia, U.S.A.

Resumen

En este trabajo se presenta un procedimiento para la estabilización simultánea con múltiples dominios acotados (MDA) de sistemas con una entrada y una salida (SISO) usando redes neuronales artificiales. Se desarrolla una metodología basada en algoritmos con una función objetivo, una red backpropagation, una operación de optimización con varios subprogramas (neurosimulador, neuroposiciones y compensadores). Cada uno de estos subprogramas cumple con una tarea específica en la búsqueda del compensador estabilizante C. Los resultados obtenidos para el caso de diez modelos de plantas lineales de primer orden demuestran que el controlador basado en redes neuronales es capaz de estabilizar simultáneamente el sistema en sus diferentes dominios acotados de estabilidad.

Palabras clave: Estabilización simultánea, redes neuronales, dominio de estabilidad, retropropagación, función objetivo.

Neural netwoks for simultaneous stabilization with multiple bounded domains of stability in process control

Abstract

In this research a design procedure using artificial neural networks for simultaneous stabilization with multiple bounded domains (MBD) in SISO (single input single output) systems is presented. A methodology is developed based on algorithms with an objective function, a backpropagation network, an optimization operation with several subroutines (neurosimulator, neuropositions and compensators). Each of these subroutines performs an specific task in the search of the stabilizing compensator C. The results obtained for the case of ten models of first order linear plants show that the neural network based controller is capable of stabilizing the system within the different bounded domains of stability.

Key words: Simultaneous stabilization, neural networks, stability domain, objective function, backpropagation .

1. INTRODUCCIÓN

Una importante herramienta para el control de procesos dentro del campo de control robusto es la estabilización simultánea. Esta herramienta se aplica para encontrar respuesta al problema de estabilización de los sistemas lineales o no lineales con más de un modelo de planta, donde las incertidumbres que se presentan se derivan de la dinámica de la planta y son expresadas en un número discreto ó finito de modelos de dicha planta. La planta nominal esta representada por p_0 y las dinámicas del sistema o perturbaciones discretas provenientes de fallas de componentes del sistema están representadas por $p_1,..., p_r,$ formando así una familia de r plantas.

En este trabajo se plantea encontrar un compensador común c, basado en redes neuronales artificiales (RNA), capaz de estabilizar la familia de plantas $p_0, p_1, ..., p_r$, en sus dominios de estabilidad acotados.

El dominio de estabilidad de un sistema compensado es la región del plano s donde se encuentran todos los polos de lazo cerrado y en este estudio se hace necesario acotar dichos dominios debido a ciertas limitaciones físicas de los instrumentos que componen el sistema (actuadores, transductores, sensores, etc).

La asignación de este acotamiento se determinará según el factor de amortiguamiento (ξ) y el tiempo de reposo o de establecimiento (T_s) deseado por el sistema.

La implementación de sistemas de control en un proceso industrial puede llegar a ser costosa y compleja, por esta razón se hace necesario el uso de programas de computación para visualizar el comportamiento a través de un software que muestre la simulación de la operación del sistema.

2. REDES NEURONALES ARTIFICIALES PARA ESTABILIZACIÓN SIMULTÁNEA CON MDA

Existe una gran cantidad de problemas abiertos en control, uno de ellos es la estabilización simultanea con MDA que consiste en encontrar a un controlador que estabilice un conjunto finito de plantas p_i (i = 1, 2…,r), dentro de una región especifica D [1], [2]. Este problema es importante dentro del área del control robusto; por ejemplo, cuando se requiere la estabilidad de un sistema operando en diferentes modos, o cuando un modelo linealizado de un sistema no lineal operando en diferentes puntos de operación tiene un modelo diferente y se requiere diseñar un controlador que estabilice simultáneamente los diferentes modelos.

La solución al problema de encontrar un algoritmo eficiente y rápido para obtener un controlador que estabilice simultáneamente ***m*** plantas dentro de dominios de estabilidad acotados, ha sido una tarea ardua de investigadores especializados en este campo de control de sistemas y en este trabajo se presenta una solución empleando una poderosa herramienta como lo son las RNA, que aprovechando su capacidad para simular dinámicas y entrenarse mediante comportamientos logran resolver satisfactoriamente el problema planteado.

3. LAS REDES NEURONALES ARTIFICIALES

Las nuevas tendencias en control de sistemas dinámicos intentan aprovecharse de las redes neuronales como controladores no lineales avanzados. Trabajos recientes de investigación muestran que las redes neuronales son capaces de realizar tareas de control (discreto y continuo) con sistemas no lineales, debido a que las redes aprenden a través de entrenamiento en lugar de descripciones formales. Esto las ha hecho la opción preferencial para modelar procesos de variables con interrelaciones complejas. Motivado por lo anteriormente dicho se decide sustituir un compensador convencional por un algoritmo neuronal apropiado que pueda realizar de forma satisfactoria la estabilización de un proceso en estudio [3-6].

3.1. Definición

Las Redes Neuronales Artificiales, en general, son dispositivos procesadores (algoritmos realizados bajo software o hardware), que tratan de modelar o simular de manera parcial y muy simple, la estructura de la corteza cerebral animal y del Sistema Nervioso Central [4-6].

3.2. Taxonomía

Existen dos fases en toda aplicación de las Redes Neuronales: la Fase de Aprendizaje o Entrenamiento y la Fase de Prueba o de Operación. En la Fase de Entrenamiento, se usa un conjunto de datos o patrones de entrenamiento para determinar los pesos que definen el modelo neuronal. Una vez entrenada las Redes Neuronales se clasifican comúnmente en términos de sus correspondientes Algoritmos o Métodos de Entrenamiento [4], [5].

Dichos métodos se basan en la asignación de valores a los pesos sinápticos de la red, a su vez los valores de estos pesos pueden ser preestablecidos o entrenados adaptativamente.

3.3 Modelo de una Neurona

En el modelo de una Neurona (ver Figura 1) se trata de establecer un modelo matemático que represente el comportamiento de un elemento simple de procesado. Los elementos que conforman toda Neurona son los siguientes:

- Entradas (p)
- Pesos sinápticos (W)
- Función de Activación o de Transferencia (f)
- Salida (a)

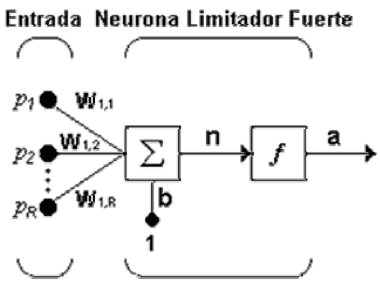

Figura 1. Modelo de una neurona.

3.4. Arquitectura de las Redes

En el caso de la arquitectura de la red neuronal, se busca establecer también un modelo matemático, pero en este caso, que represente el comportamiento de una gran cantidad de elementos simples de procesamiento, interconectados entre sí [5].

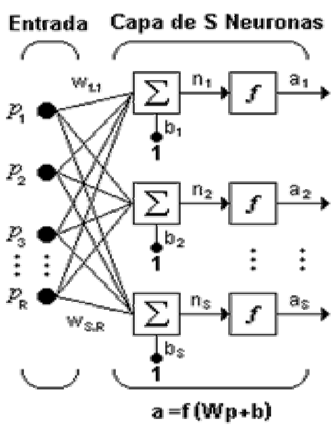

Figura 2. Red neuronal de dos capas.

Entre dos capas de Neuronas existe una red de pesos de conexión (Figura 2), es decir, una red de interconexión entre dichas Neuronas, la configuración de estas interconexiones es la que indica cómo se propagará la señal desde un elemento de procesamiento a otro o hacia sí mismo. La configuración de las interconexiones entre los elementos de procesamiento se puede clasificar en tres tipos, éstos son: Conexiones Feedforward, Conexiones Feedback y Conexiones Feedlateral [4], [5].

3.5. La Red Backpropagation

El algoritmo backpropagation es un tipo de aprendizaje supervisado, que emplea un ciclo propagación-adaptación de dos fases. Una vez que se ha aplicado un patrón a la entrada de la red como estímulo, éste se propaga desde la primera capa a través de las capas superiores de la red, hasta generar una salida. La señal de salida se compara con la salida deseada y se calcula una señal de error para cada una de las salidas.

Las salidas de error se propagan hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de la capa oculta que contribuyen directamente a la salida. Este proceso se repite, capa por capa, hasta que todas las neuronas de la red hayan recibido una señal de error que describa su contribución relativa al error total. Basándose en la señal de error percibida, se actualizan los pesos de conexión de cada Neurona, para hacer que la red converja [7-12].

Existen otros tipos de redes ya generalizadas y muy utilizadas como lo son la red ADALINE, red de base Radial, redes Competitivas, entre otras. En este trabajo se hace referencia a la red Backpropagation solamente debido a que fue la red apropiada utilizada para la solución del problema planteado.

4. LA ESTABILIZACIÓN SIMULTÁNEA

En la teoría de control robusto el objetivo es determinar un controlador simple para obtener estabilidad de lazo cerrado independientemente del modelo y las señales de incertidumbre. Un diseño de control robusto realimentado intenta disminuir los efectos de las perturbaciones de los modelos y reunir los requerimientos de lazo cerrado para la planta nominal y la familia de esta planta nominal [1], [13]. En el análisis de cada caso particular, el problema del control robusto puede declararse con mayor precisión especificando los tipos de modelos de planta, entradas del sistema, modelos de incertidumbre e índice de actuación que va a ser usado [1], [2].

Tal como se menciona entonces, el problema de la estabilización simultanea describe un caso especifico e importante del control robusto. En este caso las incertidumbres de la planta (o proceso) se representan como un numero finito de modelos de plantas pi (i = 1, 2…, r).

La estabilización simultanea esta íntimamente relacionada al problema de la estabilización fuerte, es decir, la estabilización de una planta por un compensador estable, y además con el problema de la estabilización biestable (estabilización con un compensador estable y de fase mínima) [2], [13], [14].

5. ESTABILIZACIÓN SIMULTÁNEA CON MÚLTIPLES DOMINIOS ACOTADOS DE ESTABILIDAD

La ventaja de emplear múltiples dominios en lugar de usar únicamente el dominio más pequeño D_0 radica en que se puede estar interesado en estabilizar cada planta p_i (i = 1, 2…, r) en alguna región de D_i que no es parte de D_0. En este caso la estabilización simultánea se realiza con respecto a cada uno de los dominios específicos para cada planta. Éste es un problema menos limitativo que considerar el mismo dominio D_0 para todas las plantas, y también más real. Las plantas p_i (i = 1, 2…, r) pueden ser simultáneamente estabilizables con respecto a cada D_i aún cuando no lo sean con respecto al más pequeño común dominio D_0. El uso de dominios acotados radica en el hecho de que existen limitaciones físicas en el compensador y los actuadores presentes en el sistema. Estas limitaciones resultan en una cota superior en la magnitud de la parte real negativa de los polos de lazo cerrado. La forma para estos dominios acotados se muestra en las Figuras 3 y 4, para el caso de dos modelos p_0, p_1 habrá dos dominios acotados D_0, D_1; para el caso de tres modelos p_0, p_1, p_2, habrá tres dominios acotados D_0, D_1, D_2; para el caso de cuatro modelos p_0, p_1, p_2, p_3 habrá cuatro dominios acotados D_0, D_1, D_2, D_3 y así sucesivamente [15].

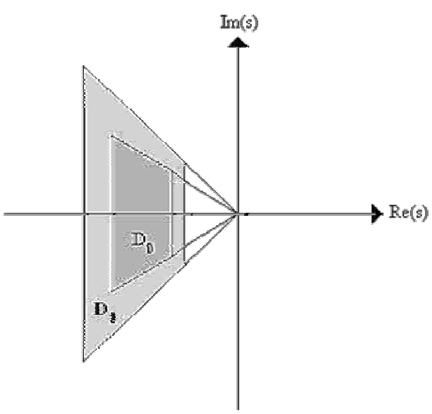

Figura 3. Dominios acotados de estabilidad D_0, D_1.

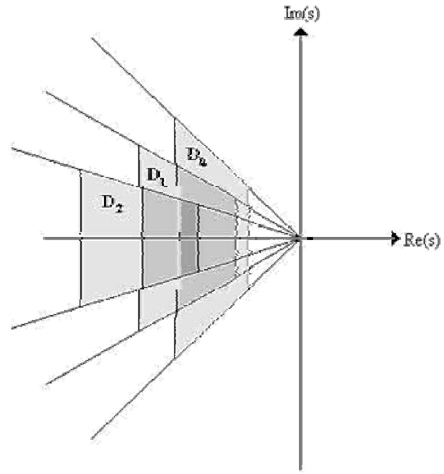

Figura 4. Dominios acotados de Estabilidad D_0, D_1, D_2.

En estabilización fuerte simultánea, es necesario que todas las plantas satisfagan la propiedad de paridad interlazada y para el caso de múltiples dominios acotados que satisfagan la extensión de la propiedad interlazada desarrollada por Arteaga-Bravo [13], Beale y Arteaga-Bravo [15].

6. EXTENSIÓN DE LA PROPIEDAD DE PARIDAD ÍNTER-LAZADA PARA DOMINIOS ACOTADOS

Una extensión de la propiedad de paridad ínter-lazada ha sido desarrollado por Arteaga [13] donde se considera el caso de un dominio acotado D de estabilidad que se ilustra en la Figura 5. En este tipo de dominio, D^c representa el complemento de D y D^c_{+e} representa el complemento extendido de D. Entonces: Sea D un Dominio acotado de estabilidad y D^c_{+e} el complemento extendido de D. Ahora supongamos a $p \in R(s)$ y que $z1,..., zk$ son los ceros reales de p en D^c_{+e}. Si en cada uno de los intervalos $(zi, zi+1) \in D^c_{+e}$ con $i=1,...., k$, hay un número par de polos reales pi de $p(s)$, entonces allí existe un compensador $c(s)$ estable con respecto a D tal que el sistema de lazo cerrado (p,c) es estable con respecto a D.

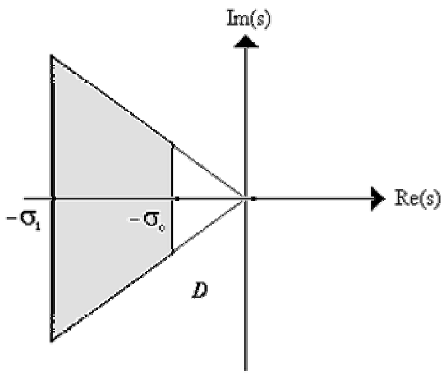

Figura 5. Dominio acotado de estabilidad.

La propiedad descrita es llamada la Extensión de la Propiedad de Paridad Interlazada (e.p.i.p.). Entonces se dice que una planta es *D*-Fuertemente Estabilizable si y solo si cumple con la e.p.i.p., esto quiere decir que e.p.i.p. es una condición necesaria y suficiente para la *D*-Estabilización fuerte de una planta **p**. Una prueba de este teorema se encuentra en Arteaga Bravo (1995) [13].

7. PLANTEAMIENTO DEL PROBLEMA

Para el planteamiento del problema de estabilización simultánea con MDA se tiene lo siguiente: sean

$$P_1 = \frac{N_1(s)}{D_1(s)}, \ldots\ldots, P_m = \frac{N_m(s)}{D_m(s)}$$

las plantas donde *Ni(s), Di(s)* ∈ *Â[s]*, son polinomios coprimos, (es decir polinomios que no tienen raíces en común en D^{c}_{+e}) para cada *i = 1, 2, ..., m. Â[s]* es el anillo de polinomios con coeficientes reales.

Ahora se propone un par:

Nc(s), Dc(s) ∈ *Â[s]*

de polinomios coprimos donde

$$C(s) = \frac{N_c(s)}{D_c(s)},$$

tales que:

$$Ni(s)Nc(s)+Di(s)Dc(s) = hi;\ i = 1, 2, \ldots, m \qquad (1)$$

Los *hi(s)* y *Dc(s)* son polinomios estables, esto es que la parte real de todas sus raíces es estrictamente negativa. En estabilización fuerte simultanea es necesario que todas las plantas satisfagan la propiedad de paridad interlazada [13], [16].

La Función Objetivo *F*, que se propone en este trabajo, para la estabilización simultánea con múltiples dominios acotados de estabilidad aplicando redes neuronales artificiales viene dada por:

$$F(C_k, P_i) = \mathop{\Phi}_{k=1}^{\#C}\{Nc_k np_i + Dc_k dp_i\},\ \forall i = 1,2,3,\ldots,n \qquad (2)$$

donde:
#C = número de Compensadores.
n = número de plantas

Y se desarrolla la función objetivo para cada planta p_i , *i* = 0,1,2,...., *n*;

Para resolver esta Función Objetivo se necesita un algoritmo compuesto por un conjunto de denominadores y numeradores de 2do Orden con coeficientes aleatorios que conformarán un conjunto de compensadores estables y de fase mínima y con la familia de plantas que se desean estabilizar, todos para entrenar las redes neuronales según criterios dados por la experiencia de los programadores: numero de capas, número de neuronas en cada capa, algoritmo de entrenamiento, la meta deseada o grado de error a alcanzar, etc. El target o función objetivo de las redes está dado con la finalidad de satisfacer la ecuación (1). Estos procedimientos son explicados con mayor detalle en las próximas secciones.

8. ALGORITMO USANDO LAS RNA

A continuación se presenta la implementación de las redes neuronales artificiales, usando el apoyo de los recursos computacionales y del software Matlab en la solución al problema de la estabilización simultánea con múltiples dominios acotados planteado en la sección 7. La implementación de redes neuronales está dividida en dos partes: *Metodología Aplicada a la solución de problema y Descripción de programas desarrollados para la solución del problema* (ver Figura 6).

9. METODOLOGÍA APLICADA A LA SOLUCIÓN DEL PROBLEMA

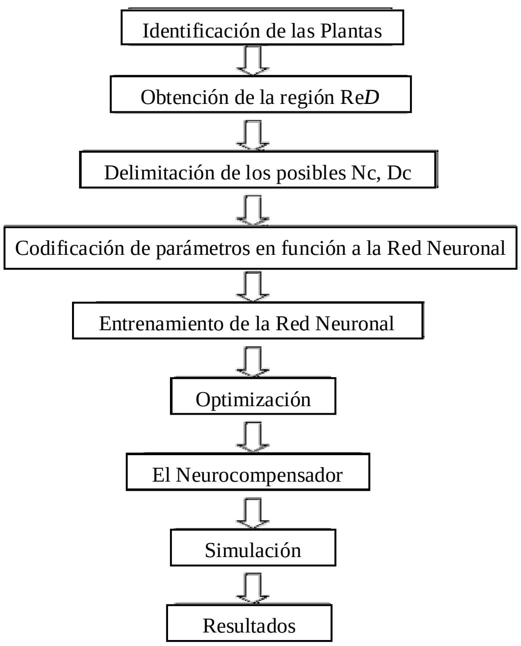

Figura 6. Etapas para la solución del problema.

9.1. Identificación de las Plantas

Se tiene un conjunto finito de *n* plantas coprimas y representativas de un sistema con sus dinámicas o fallas, donde el orden de las plantas alcanzan el segundo orden bipropias (como manera de simplificar el problema aproximando plantas de orden superior con plantas de segundo orden), expresadas como:

$$P_i = \frac{a_i s^2 + b_i s + c_i}{d_i s^2 + e_i s + f_i} \quad , \forall\ i=0,1,2,...,n \qquad (3)$$

9.2. Obtención de la Región ReD

Con los respectivos polos y ceros de la planta p_0, p_1, p_2, ..., p_n; $\forall\ i$ =1,2,..., *n* se obtiene la región Re*D:*

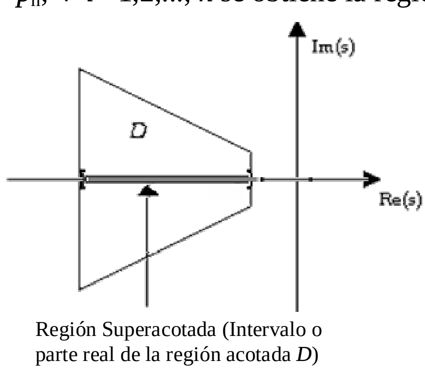

Figura 7. Región Re*D*.

La región super-acotada Re*D* (Figura 7) será un intervalo con un valor máximo y valor mínimo, es decir:

$$\mathrm{Re}D = [\mathrm{minReD}\,,\ \mathrm{maxReD}] \qquad (4)$$

9.3. Delimitación de los Posibles Nc, Dc

En esta etapa se contempla el conjunto de compensadores posibles que darán solución al problema. El compensador a elegir será Bipropio de orden inmediatamente superior al orden del conjunto de plantas a estabilizar, para este caso será de tercer orden.

Este conjunto de compensadores será aleatorio y los polinomios numeradores (Nc) y denominadores (Dc) tendrán sus raíces dentro de Re*D.*

Dado que Nc y Dc son polinomios de 3er orden, tendrán 3 raíces. Basado en esto, la forma de obtenerlos es consiguiendo tres valores aleatorios que representan las raíces dentro de Re*D.*

Los compensadores del conjunto generado aleatoriamente serán de la siguiente forma:

$$C = \frac{N_c}{Dc} = \frac{n_3 s^3 + n_2 s^2 + ns + n_0}{d_3 s^3 + d_2 s^2 + ds + d_0} \qquad (5)$$

9.4. Codificación de los parámetros en función de la Red Neuronal:

La red neuronal a utilizar es una red Backpropagation la cual se determinó luego de una serie de pruebas con diferentes tipos de redes resultando esta red como la más idónea.

Arquitectura:

- Numero de capas: se usan tres capas con #I neuronas en la capa oculta y #O neuronas en la capa de salida.

- Numero de entradas (#I) :

  #I = #CoNc + #CoDc + #Conp + #Codp (6)

  donde:
  #CoNc: Números de coeficientes de Nc
  #CDc: Números de coeficientes de Dc
  #Conp: Números de coeficientes de np
  #Codp: Números de coeficientes de dp

- Numero de salidas (#O):

$$\#O = deg(\mathrm{Dc}) + deg(\mathrm{dp}) + 1 \qquad (7)$$

donde:
deg(Dc): orden de Dc
deg(dp): orden de dp

- Se ajusta el rango de valores de cada entrada y del target, es decir, el máximo y el mínimo valor que ingresará por cada entrada.
- Las funciones de activación a utilizar: *tansig* para la capa oculta *purelin* para la capa de salida.

Parámetros prácticos:

- Meta planteada: El error mínimo deseado es de 1e-6.
- Rata de aprendizaje: Valor es variable y depende de la arquitectura de la red, se recomienda en ≅ 0.102.
- El algoritmo de optimización del error de la RN es el *trainlm* (entrenamiento por el algoritmo de Levenberg-Marquardt) por ser el más eficiente y rápido para el entrenamiento requerido.
- Numero de épocas: es el numero de iteraciones que la red utiliza para entrenar, se recomienda 2500.

9.5. Neuroplanta

Se entrena una red neuronal artificial Backpropagation para una planta y un conjunto de compensadores generados aleatoriamente con sus polos y ceros en la región Re*D* para resolver la función objetivo.

El número de redes a entrenar (#net) va a ser igual al numero de plantas (#p):

$$\mathbf{\#net = \#p} \qquad (8)$$

y las entradas serán los coeficientes de Nc, Dc, n*p*, d*p*.

En los siguientes esquemas se muestra de forma gráfica lo anteriormente descrito:

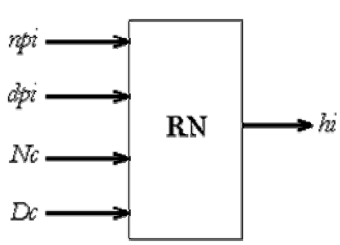

Figura 8. Forma general de representar la neuroplanta.

La salida de la red corresponde a los coeficientes del polinomio *hi*:

$$hi = h_5 s^5 + h_4 s^4 + h_3 s^3 + h_2 s^2 + hs + h_0 \qquad (9)$$

9.6. Optimización

La optimización consiste en un algoritmo computarizado que selecciona los compensadores más óptimos que hicieron que *hi* ubicara sus raíces dentro de la región *D*. Esta selección de los compensadores óptimos se realiza para cada planta, obteniéndose una cantidad de compensadores estabilizantes para cada planta.

La optimización consta de varios subprogramas (Neurosimulador, neuroposiciones, compensadores), cada subprograma cumple una función específica dentro de la búsqueda del compensador común C (ver Figura 9).

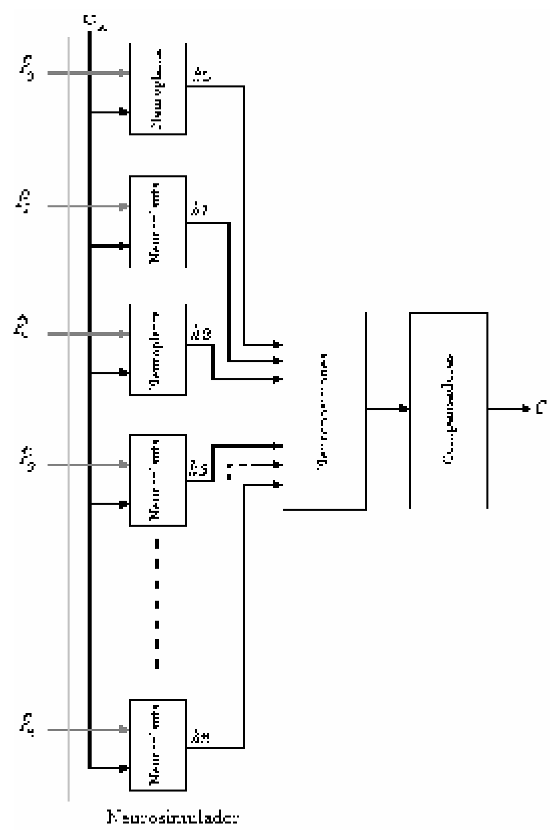

Figura 9. Optimizador.

Neurosimulador: Está diseñado para obtener la respuesta de cada una de las neuroplantas, estas respuestas son los *hi* correspondientes de cada planta con el conjunto de compensadores C_k.

Neuroposiciones: Este subprograma obtiene las posiciones de los compensadores pertenecientes al conjunto C_k que lograron estabilizar cada una de las plantas, es decir, cada planta tendrá un numero de compensadores que lograron estabilizarla dentro de la región *D* y este programa dará las posiciones de estos compensadores para cada planta.

Compensadores: Este subprograma se encarga de determinar los compensadores que lograron estabilizar simultáneamente el conjunto de plantas en la región acotada *D*.

$$C(s) = \frac{n_3 s^3 + n_2 s^2 + ns + n_0}{d_3 s^3 + d_2 s^2 + ds + d_0} \qquad (10)$$

9.7. El Neurocompensador NC

Este paso consiste en simular mediante RN el compensador *C* obtenido en el paso anterior y se sustituir la ecuación del compensador por una red neuronal compensadora.

Datos: Para simular el compensador *C*, con una red neuronal, es necesario conocer el comportamiento de *C* ante la presencia de una señal de entrada. Para esto se forma un sistema compuesto por el compensador, la planta nominal y una señal de entrada rampa con pendiente igual a 1 (para conocer su comportamiento punto a punto). Entonces se toma muestra de su entrada y salida para hacer una base de datos que será suministrada a la red neuronal para su posterior entrenamiento (ver Figura 10).

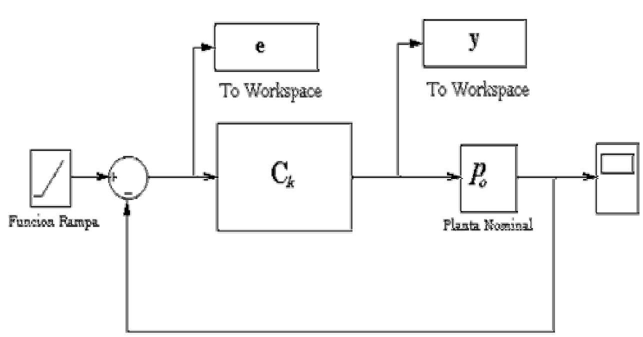

Figura 10. Obtención de datos e, y.

Entrenamiento: La red a entrenar es una red Backpropagation con: una entrada e, una salida *y*, donde y = target = t, la función tansig en la capa oculta, pureline en la capa de salida, un error de 1E-2 y el algoritmo de optimización Levenberg-Marquardt (trainlm). Esto se ilustra en la Figura 11.

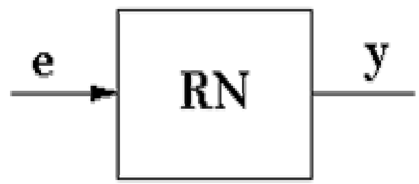

Figura 11a. Entradas y salidas de red neuronal.

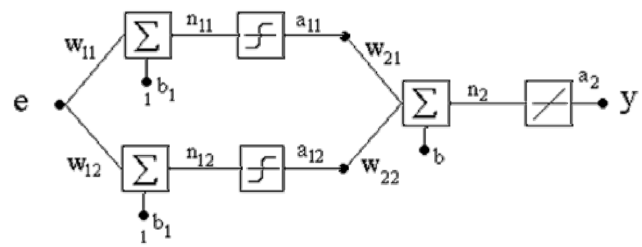

Figura 11b. Arquitectura interna de RN.

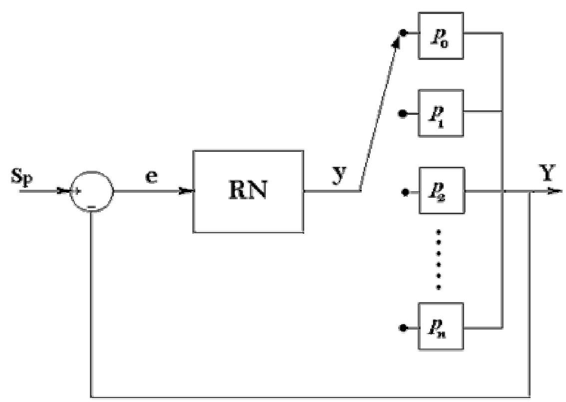

Figura 12. Neurocompensador como controlador de un sistema de lazo cerrado para n plantas p_0, p_1, p_2,..., p_n, $\forall\ i = 0,1,2,\ldots,n$.

9.8. Simulación

La simulación se realiza bajo el ambiente Simulink del software Matlab donde se configura el sistema representado por de la Figura 12.

9.9. Resultados

Estos valores son obtenidos por la simulación de los esquemas en Simulink y verificados por medio de las gráficas obtenidas, donde se observa la completa adhesión a la solución del problema planteado.

10. EJEMPLO REPRESENTATIVO

En esta sección se presenta la estabilización simultánea con MDA mediante un compensador para diez plantas propias de 1er orden $p_0, p_1, p_2, ..., p_9$. Esto se logro mediante la implementación de diez redes neuronales tipo backpropagation entrenadas cada una con un modelo de planta donde el vector de entrada p para el entrenamiento está compuesto por los coeficientes del numerador y denominador de las plantas *pi*, " *i=0,1,2, ..., 9* y por los coeficientes de los numeradores y de los denominadores de compensadores bipropios, de 2do orden y estables generados aleatoriamente con ceros y polos dentro de la región Súper-acotada (intervalo real dentro de la región acotada, ver Figura 13).

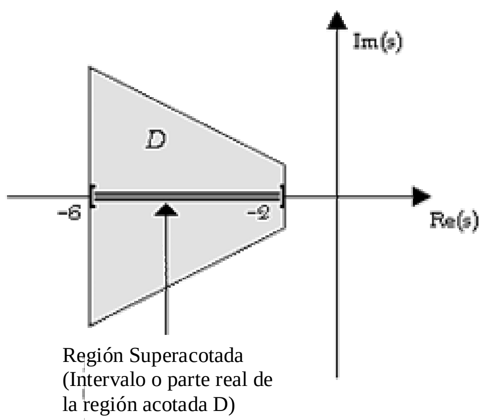

Figura 13. Región súper-acotada.

Las siguientes 10 plantas propias y de primer orden:

$$p_0 = \frac{5}{s+2};\quad p_1 = \frac{10}{s-6};\quad p_2 = \frac{4}{s+1}$$

$$p_3 = \frac{2}{s+1};\quad p_4 = \frac{3}{s-1};\quad p_5 = \frac{7}{s+3}$$

$$p_6 = \frac{9}{s-4};\quad p_7 = \frac{4}{s-2};\quad p_8 = \frac{1}{s+0.5}$$

$$p_9 = \frac{3}{s+1.5}$$

y los correspondientes dominios *D0, D1, D2, D3, D4, D5, D6, D7, D8, D9* están definidos como (ver Figura 14):

D0={s: -7<Re(s)<-2, |Im(s)|≤0.952 |Re(s)|}

D1={s: -6<Re(s)<-1, |Im(s)|≤1.395 |Re(s)|}

D2={s: -6.5<Re(s)<-1.5, |Im(s)|≤1.4|Re(s)|}

D3={s: -6.3<Re(s)<-1.8, |Im(s)|≤1.518|Re(s)|}

D4={s: -7<Re(s)<-2, |Im(s)|≤2.29 |Re(s)|}

D5={s: -8<Re(s)<-0.5, |Im(s)|≤1.772 |Re(s)|}

D6={s: -7.5<Re(s)<-1.9, |Im(s)|≤1.632 |Re(s)|}

D7={s: -6.5<Re(s)<-1, |Im(s)|≤1.441 |Re(s)|}

D8={s: -8.5<Re(s)<-0.5, |Im(s)|≤1.98 |Re(s)|}

D9={s: -10<Re(s)<-1, |Im(s)|≤2.3|Re(s)|}

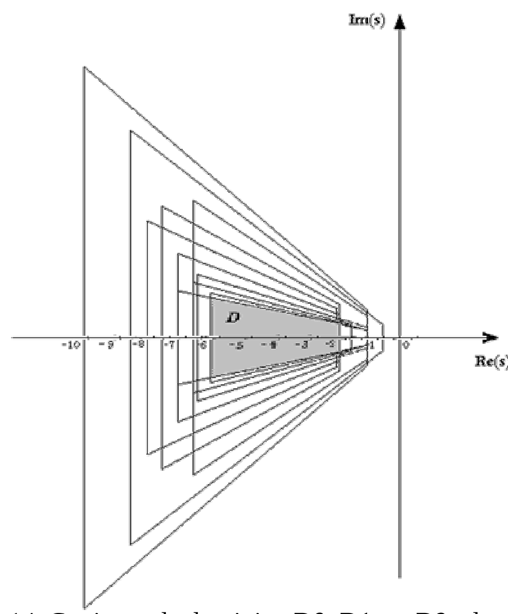

Figura 14. Conjunto de dominios *D0, D1,….,D9* y la región de intersección de los dominios *D* cuyo intervalo real es [-6, -2].

Los valores reales dados por $s\,i^-$ y $s\,i^+$ (con i = 0, 2,...,9) y quedan establecidos por el rango de valores de Tiempos de Establecimiento (Ts) especificados según el criterio del 2%, y la pendiente de la recta es representada por la tangente del ángulo que determina el arcoseno de factor de amortiguamiento (ξ) con el eje real del semiplano izquierdo.

El algoritmo se realizó en Matlab ®, determinando un conjunto de 3 compensadores que estabilizan el juego de diez plantas representativas de un sistema con sus dinámicas o perturbaciones, con tan solo diez redes neuronales tipo backpropagation (una para cada planta) entrenadas con el algoritmo de optimización Levenberg-Marquardt y donde cada red posee nueve neuronas en su capa oculta y cuatro en la capa de salida (una neurona para cada entrada), usando una rata de aprendizaje de 0.102 y con una meta de 1e-6. Las funciones de activación utilizadas fueron Tansig en la capa oculta y purelin en la capa de salida.

El conjunto de controladores estabilizantes dentro de la región acotada *D* fue:

$$C_1 = \frac{1.0s^2 + 10.2299s + 25.6896}{1.0s^2 + 6.7032s + 10.3927}$$

$$C_2 = \frac{1.0s^2 + 7.3497s + 13.5027}{1.0s^2 + 8.0478s + 15.0420}$$

$$C_3 = \frac{1.0s^2 + 7.1837s + 11.8534}{1.0s^2 + 7.2448s + 10.6167}$$

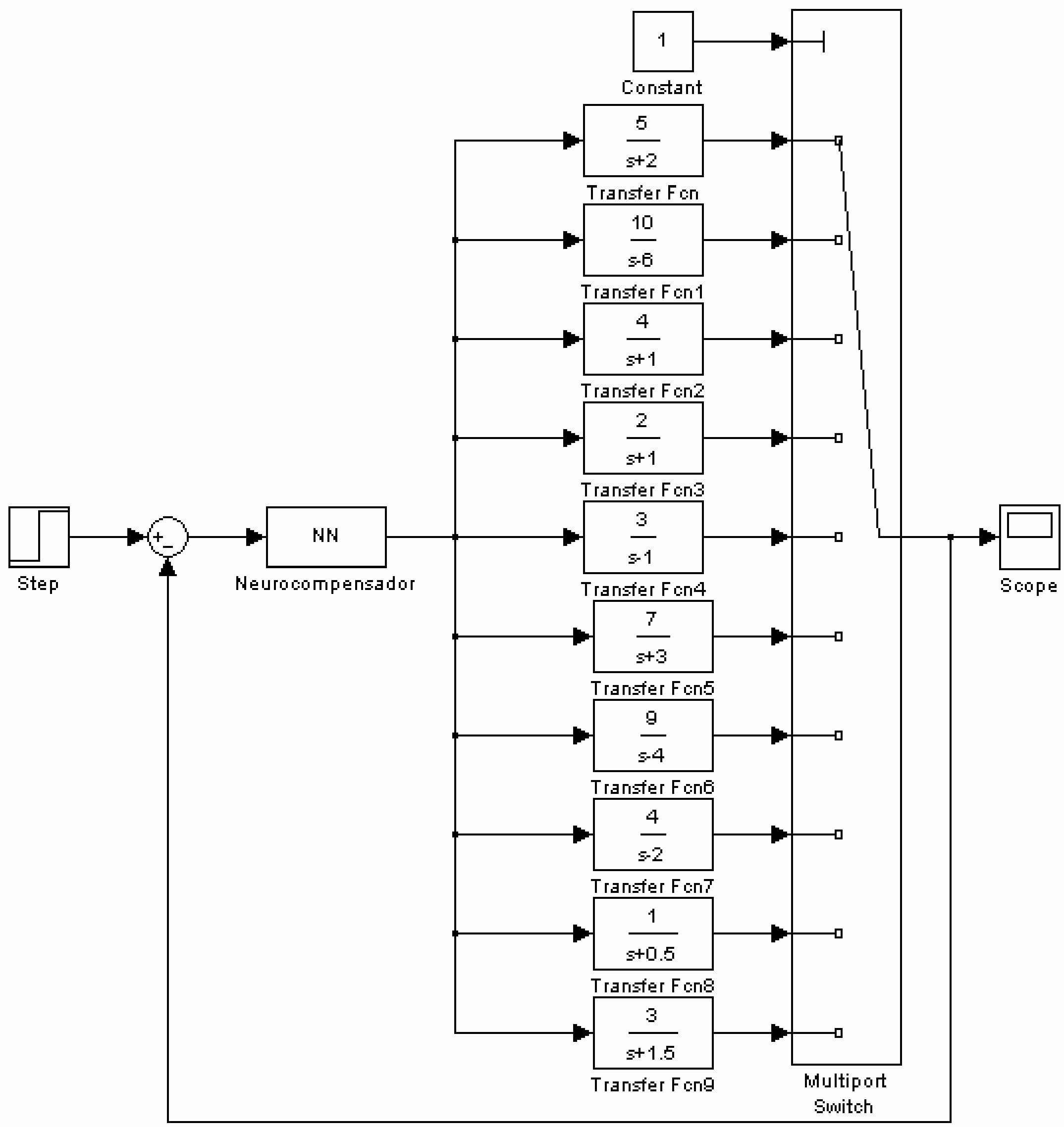

Figura 15. Estabilización simultánea con MDA de 10 plantas de primer orden con un neurocompensador.

11. RESULTADOS

El esquema en lazo cerrado del compensador C_1 simulado por una red neuronal (Neurocompensador) para una entrada escalón unitario y realimentación unitaria con cada uno de los sistemas se muestra en la Figura 15. Esta figura es la representación en Simulink del sistema en lazo cerrado del compensador Neurocompensador con la entrada de un escalón unitario, trabajando simultáneamente para las diez plantas activadas por medio de un selector.

Aquí se presentan las respuestas a una entrada escalón unitario para cada una de las plantas con el Neurocompensador (Figuras 16 a la 24):

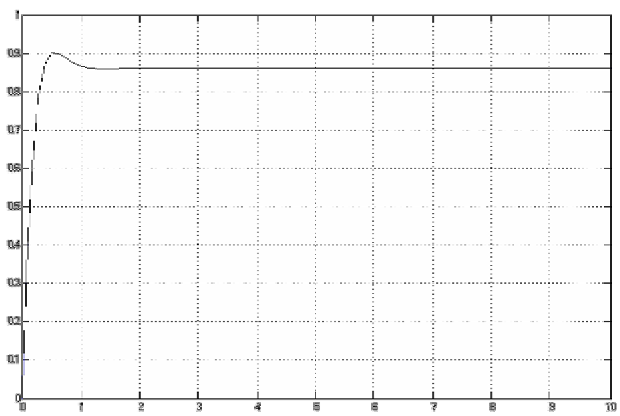

Figura 16. Respuesta temporal para una entrada de escalón unitario NC_1, p_0 .

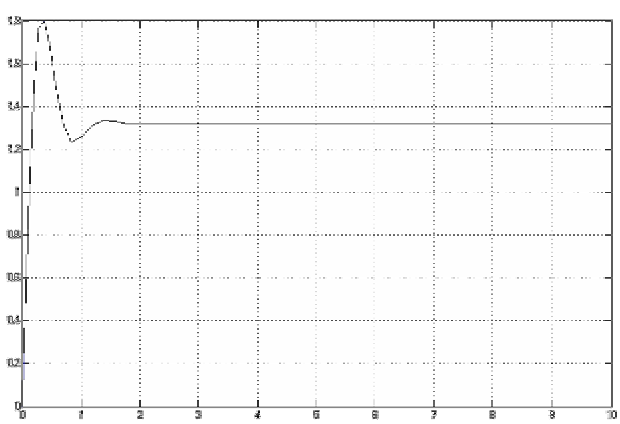

Figura 17. Respuesta temporal para una entrada de escalón unitario NC_1, p_1 .

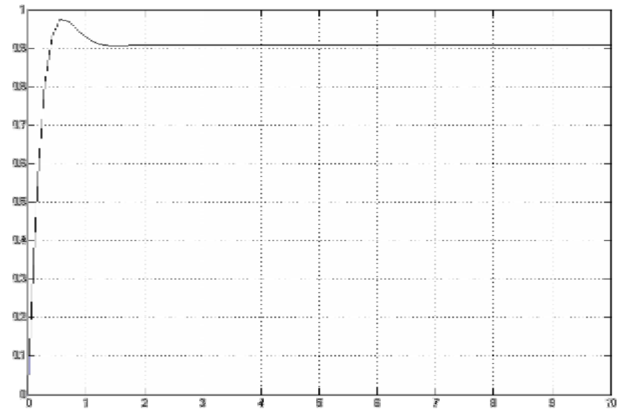

Figura 18. Respuesta temporal para una entrada de escalón unitario NC_1, p_2 .

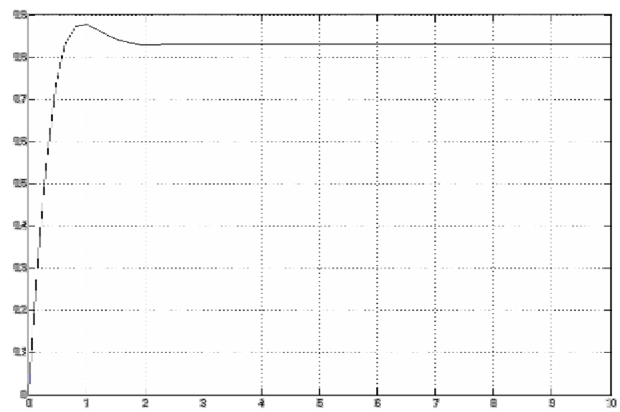

Figura 19. Respuesta temporal para una entrada de escalón unitario NC_1, p_3 .

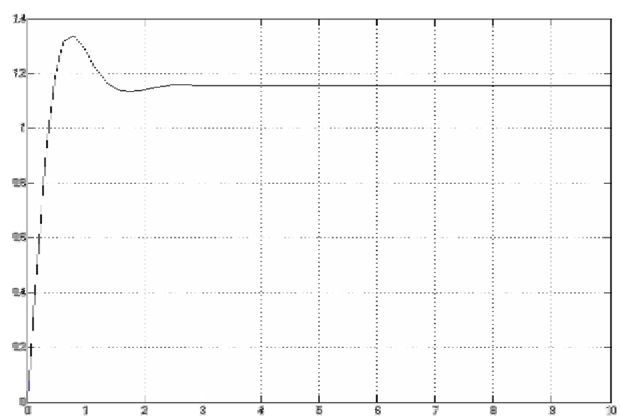

Figura 20. Respuesta temporal para una entrada de escalón unitario NC_1, p_4 .

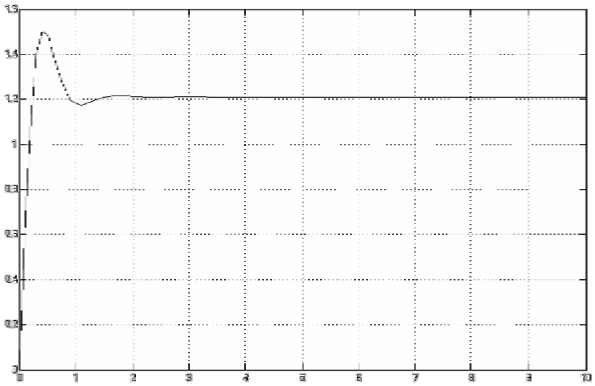

Figura 21. Respuesta temporal para una entrada de escalón unitario NC_1, p_5 .

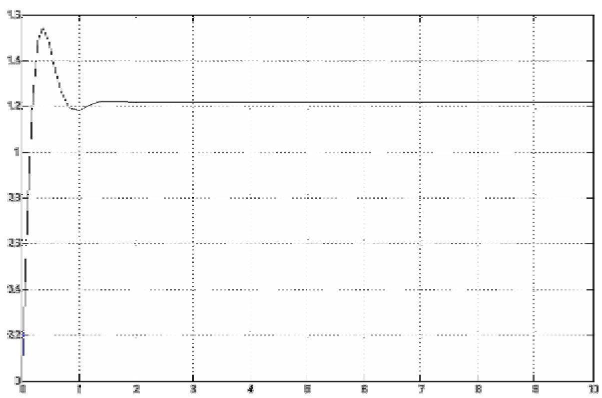

Figura 22. Respuesta temporal para una entrada de escalón unitario NC_1, p_6 .

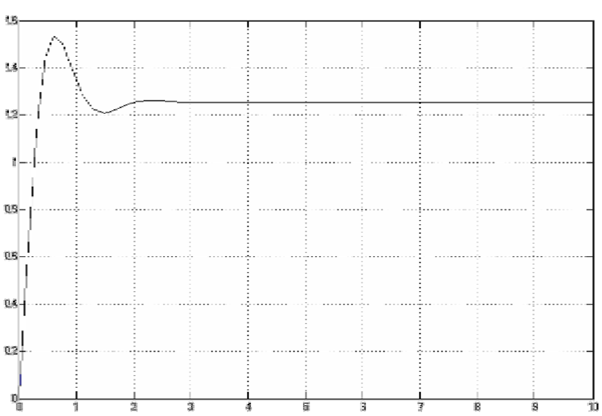

Figura 23. Respuesta temporal para una entrada de escalón unitario NC_1, p_7 .

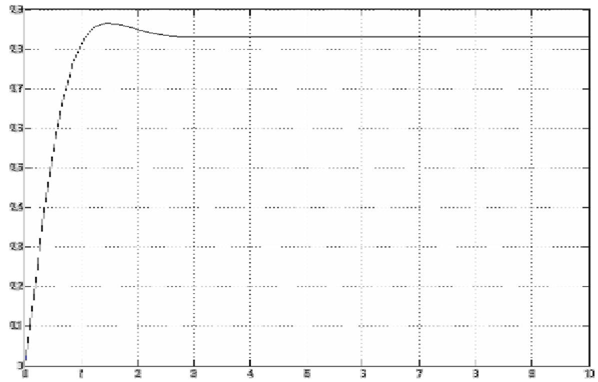

Figura 18. Respuesta temporal para una entrada de escalón unitario NC_1, p_8 .

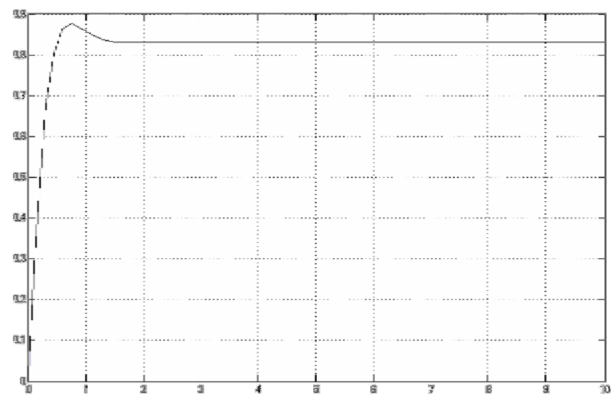

Figura 24. Respuesta temporal para una entrada de escalón unitario NC_1, p_9 .

12. CONCLUSIONES

En este trabajo se ha presentado una muestra de un conjunto de metas propuestas para satisfacer el problema de la estabilización simultanea con múltiples dominios acotados de estabilidad usando un diseño de software basado en redes neuronales artificiales. Se desarrolló un algoritmo capaz de encontrar un compensador que estabilice simultáneamente un conjunto de modelos de plantas de un sistema usando las redes neuronales, se sustituyó el compensador encontrado, por el algoritmo anteriormente mencionado, por una red neuronal que ofrece buenos resultados, reduciendo el tiempo de establecimiento Ts y las oscilaciones del sistema compensado. Se obtuvieron resultados satisfactorios para un ejemplo ilustrativo de estabilización simultanea con RNA para diez plantas propias y de primer orden, con múltiples dominios de estabilidad.

En base a lo anteriormente expuesto se puede decir que el uso de algoritmos de computación neuronal demuestra su efectividad para trabajar en problemas que hasta los momentos son complejos de trabajar de manera analítica. Como tema de investigación posterior se puede también analizar el diseño de RNA para sistemas con múltiples entradas y salidas.

13. REFERENCIAS

[1] Peter, Dorato. "Robust Control". IEEE Press, New York, 1987.

[2] Blondel, Vincent. "Simultaneous Stabilization of Linear System", Lecture Notes in Control and Information Sciences, volume 191. Springer-Verlag, Berlin, 1994.

[3] Aguilera, L. "Técnicas de Programación lógica y diagramación, Lenguaje de programación BASIC", Academia Americana, 1995.

[4] Martínez, C. Álvarez, M. Arteaga, F. "Software Didáctico para el Aprendizaje de Redes Neuronales Artificiales", Universidad de Carabobo, Facultad de Ingeniería Eléctrica, 2000.

[5] Martin, H. Howard, D. and Beale M., "Neural network design" Math Works Inc.1998.

[6] Demuth, H. and Beale, M. "Neural Network Toolbox for use with MATLAB", Math Works Inc.1998.

[7] C. Saavedra, F. Izaurieta,, "Redes Neuronales Artificiales", Departamento de Física, Universidad de Concepción, Chile.

[8] Acosta, M. Zuluaga, C. y Salazar, H. "Tutorial de Redes Neuronales Artificiales", Universidad Tecnológica de Pereira, Facultad de Ingeniería Eléctrica, Colombia 2000.

[9] Padern, Xavier. "Redes Neuronales. Introducción", publicación en Internet, 1999.

[10] Méndez, R. "Predicción de la curva de destilación ASTM D86 para gasolinas automotrices empleando Redes Neuronales en la Refinería El Palito", Universidad de Carabobo, Área de estudio de postgrado, Tesis de Maestría, Facultad de Ingeniería, Abril de 2003.

[11] B. Martín del Brío, y A. Sanz, M. "Redes Neuronales y Sistemas Difusos", segunda edición, Universidad de Zaragoza, editorial RAMA Madrid, España, 2002.

[12] Vega, A. y Andina, D. "Tutorial de Redes Neuronales", Universidad Politécnica de Madrid, Departamento de Señales, Sistemas y Radio Comunicaciones, SSR, 2001.

[13] Arteaga-Bravo, F.J. "Simultaneous Stabilization with Multiple Bounded Domains of Stability", PhD Dissertation, George Mason University, Fairfax, Virginia, USA, February 1995.

[14] Arteaga-Bravo, F.J. Contramaestre, M. y Vizcaya, M. "Desarrollo de software para la estabilización simultanea con múltiples dominios acotados de estabilidad con el método de Factorización", aceptado para ser publicado en Revista INGENIERÍA UC, Universidad de Carabobo, Facultad de Ingeniería, 2003.

[15] Beale, G.O. and Arteaga-Bravo, F.J. "Simultaneous Stabilization with Multiple Bounded Stability Domains", Automatika, Vol.37, pp 91-98, 1996.

[16] Muñoz, S. Fernández, G. Sánchez, R. and Mayol M.M., "Strong simultaneous stabilization using evolutionary strategies" Facultad de Ingeniería de la UNAM, Mexico, 1998.