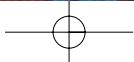
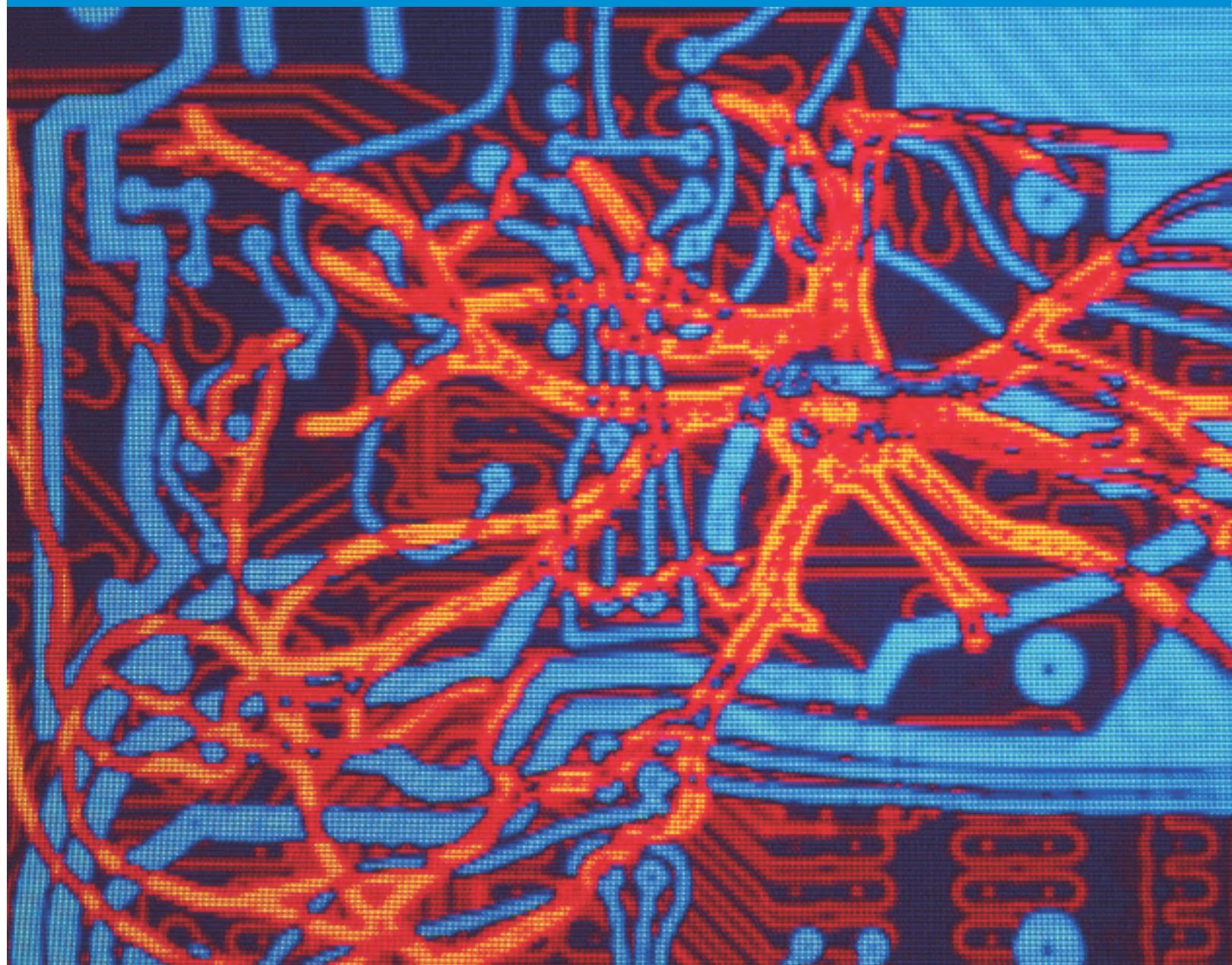
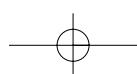


Identificación de sistemas no lineales mediante redes neuronales artificiales

JAVIER SEDANO FRANCO, ÁNGEL ALONSO ÁLVAREZ Y JOSÉ RAMÓN VILLAR FLECHA

**Características de las técnicas de identificación neurológica y su aplicación a un
modelo construido a partir de un sistema no lineal**





Introducción

¿Qué es la identificación?

Se puede definir la identificación de sistemas como los estudios de técnicas que persiguen la obtención de modelos matemáticos de sistemas dinámicos a partir de mediciones realizadas en el proceso: entradas o variables de control, salidas o variables controladas y perturbaciones. En la *figura 1* se muestra un croquis con estas ideas de forma gráfica.

El enfoque de la identificación se puede realizar en función de la estructura del modelo y de si tienen o no interpretación física sus parámetros. Podemos distinguir las siguientes estructuras: White-box, Gray-box y Black-box²⁰.

Estructura

Para realizar la modelización, es necesario partir de una observación de datos, compuesta de una salida en un instante de tiempo $y(t)$, obtenida a partir de una excitación en el mismo tiempo $u(t)$.

Por consiguiente, el grupo de observación estará formado por una colección finita de muestras:

$$\begin{aligned} Z^N = & \{u(0), y(0), u(1), y(1), u(2), \\ & y(2), \dots, u(N), y(N)\} = \\ & = \{[u(t), y(t)] \mid t = 1, \dots, N\} \end{aligned} \quad (1)$$

Los métodos de identificación¹⁵ tienen como fin encontrar un modelo matemático que nos relacione las colecciones de muestras de salidas con las de entrada. Podemos definir el grupo de entrenamiento de forma más general.

$$Z^N = \{[y(t), \varphi(t)] \mid t = 1, \dots, N\} \quad (2)$$

Se deberá estimar $y(t+1)$ a partir de $\varphi(t+1)$, donde $\varphi(t)$ es el vector de datos o vector de regresión. Por consiguiente, a partir de aquí lo que buscamos es una función de estimación $\hat{g}_N(t, \varphi(t))$, que nos permita determinar la salida estimada $\hat{y}(t)$, predictor, tal que:

$$\hat{y}(t) = \hat{g}_N(t, \varphi(t)) \quad (3)$$

Donde g será parametrizable, con número finito de parámetros; el vector de parámetros se llamará θ y la familia de funciones que describen el modelo es la denominada estructura del modelo. Entonces la función óptima consistirá en la obtención de un vector de parámetros θ adecuado, que se realizará en función de los $\hat{\theta}_N$ (parámetros estimados).

$$\hat{y}(t) = \hat{g}_N(t, \varphi(t)) = g(t, \hat{\theta}_N, \varphi(t)) \quad (4)$$

Sistemas no lineales

El modelo no lineal de ecuaciones diferenciales conocido como modelo Narnmax¹³ es una representación natural del muestreo de sistemas no lineales continuos; proporciona una representación válida para una amplia clase de sistemas lineales y tiene ventajas sobre las series de funciones, tales como las series de Volterra².

Desde un punto de vista práctico, existe la necesidad de aproximar el comportamiento de las entradas y salidas a modelos simples y de acuerdo con el teorema de Stone-Weirerstars, los modelos de salida afín y polinómicos son adecuados. Estos modelos son los siguientes:

Modelo general no lineal: Las representaciones de las entradas y salidas que expanden la salida presente en función de sus términos pasados proporcionan modelos que satisfacen una gran clase de sistemas no lineales. El modelo general es el mostrado en la ecuación (5), donde F es una función no lineal. Al modelo anterior se le conoce como Armax¹, por su semejanza con el modelo lineal representado en la ecuación (6).

$$y(k) = F(y(k-1), K, y(k-n_y), u(k-1), K, u(k-n_u)) \quad (5)$$

$$\begin{aligned} y(k) = & a_0 + \sum_{i=1}^{n_y} a_i y(k-i) + \\ & + \sum_{i=1}^{n_u} b_i u(k-i) \end{aligned} \quad (6)$$

Un sistema invariante en el tiempo y discreto no lineal se puede representar por el modelo (5) en una región próxima al punto de equilibrio, conforme a dos condiciones suficientes [13]: Que la función de respuesta del sistema es finitamente realizable y con la premisa que un modelo linealizado existe, si el sistema trabaja cerca del punto de equilibrio elegido.

Varios modelos no lineales conocidos se pueden identificar con el modelo Narnmax, considerándose casos especiales del modelo; a saber, el modelo bilineal, el modelo de salida afín y modelo racional. El modelo bilineal indica que, de forma general, la función causal continua se puede aproximar bien por sistemas bilineales dentro de cualquier intervalo limitado de tiempo [7]. Algunos ejemplos pueden ser las columnas de destilación[5] y los procesos de control termal o nuclear [17]. El modelo general bilineal viene definido por la ecuación (7).

$$\begin{aligned} y(k) = & a_0 + \sum_{i=1}^{n_y} a_i y(k-i) + \sum_{i=1}^{n_u} b_i u(k-i) \\ & + \sum_{i=1}^{n_y} \sum_{j=1}^{n_u} c_{ij} y(k-i) u(k-j) \end{aligned} \quad (7)$$

Los modelos de salida afín y racional surgen a la hora de cumplir las condiciones de la función de respuesta del sistema en lo relativo a la función polinómica y a su límite²¹. Así, se tendrá que las ecuaciones generales para estos modelos son, bien la ecuación de diferencias racional (8), donde r es el orden del sistema y $a(\cdot)$ y $b(\cdot)$ son los polinomios de grado finito, bien la ecuación de diferencias afín (9), donde los $a_i(\cdot)$, con $i=0, 1, \dots, r+1$, son polinomios de grado finito.

$$\begin{aligned} y(k) = & \frac{b(y(k-1), K, y(k-r), \\ & u(k-1), K, u(k-r))}{a(y(k-1), K, y(k-r), \\ & u(k-1), K, u(k-r))} \end{aligned} \quad (8)$$

$$\begin{aligned} y(k) = & \sum_{i=1}^r \frac{a_i(u(k-1), K, u(k-r))}{a_0(u(k-1), K, u(k-r))} \\ & y(k-i) + \frac{a_{r+1}(u(k-1), K, u(k-r))}{a_0(u(k-1), K, u(k-r))} \end{aligned} \quad (9)$$

Identificación no lineal a partir de redes neuronales

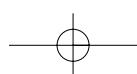
Introducción

Las redes neuronales artificiales (ANN) son herramientas de procesado de información, cuyo funcionamiento está basado en operaciones sencillas, realizadas en paralelo por células elementales, llamadas neuronas.

Las ANN son sistemas formados por neuronas, unidas entre sí, formando redes. La neurona, nodo, célula o unidad, será el elemento base del proceso, encontrándose conectada a n entradas, cada una con su factor de peso w ; siendo la función de activación de la neurona, la salida y . En la *figura 2* se muestra un ejemplo de neurona con las funciones de activación más comunes en identificación de sistemas.

Arquitectura

La arquitectura de una ANN va a depender de: Si las entradas a la neurona son externas o internas, proceden de otras neuronas, $U = \{u_0, u_1, \dots, u_N\}$; de cómo el factor de peso w_{ij} se calcula en el proceso de aprendizaje; de la función de activación f_i , la cual puede asumir cualquier



RESUMEN

En este artículo se discute la representación de sistemas no lineales en tiempo discreto y el procedimiento de identificación de éstos a partir de estructuras Narmax (*Nonlinear Autoregresive Moving Average with eXogenous inputs*) mediante la utilización de redes neuronales artificiales ANN. Se van a presentar las características de las ANN, arquitectura, funcionamiento, método de aprendizaje, en los procesos de identificación de sistemas. También se va a construir un modelo basándose en un sistema no lineal, representado a partir de ecuaciones de diferencias de segundo orden, que representa el comportamiento de una columna de destilación de metanol. Para la identificación del sistema se usará un método no paramétrico, a partir de ANN entrenadas con el método de Levenberg-Marquardt y optimizadas de acuerdo con la estrategia de reducción por neurocirugía OBS.

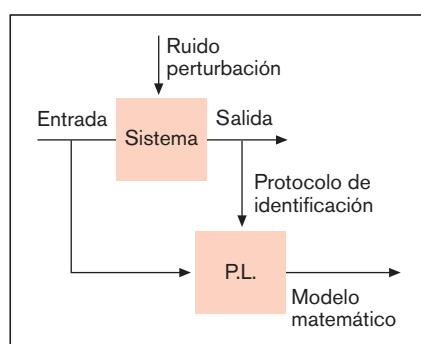
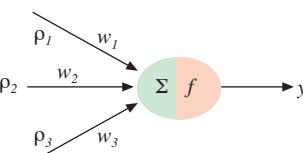


Figura 1. Proceso de identificación sobre un sistema para la obtención de un modelo matemático, mediante la realización de mediciones y captura de datos en el sistema.

forma, siendo las de mayor interés, las capaces de discriminar entre rangos de valores; las más usuales son: lineal, simétrica, saturada triangular, logarítmica, tangente hiperbólica, (figura 2); de cómo las neuronas se distribuyen por capas, siendo una capa un conjunto de neuronas semejantes con el mismo comportamiento.

Funcionamiento

Si se parte de una sucesión de datos de entrada $u(t)$ será posible obtener su salida correspondiente $y(t)$ a partir de la función



Neurona que proporciona un valor de salida de $y = f_i(\sum_{j=1}^N w_{ij} \varphi_j)$, donde f_i es la función de la neurona.

Tipos: Lineal: $f(x) = x$; Simétrica $f(x) = 1$ si $x \geq 0$ y $f(x) = -1$ si $x < 0$;

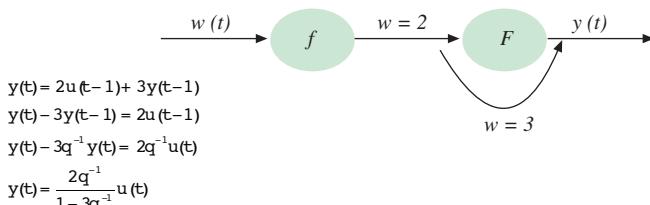
Tang. Hiperbólica: $f(x) = [2/(1+e^{-x})] - 1$.

Sigmoidal: $f(x) = (1+e^{-x})^{-1}$

Figura 2. Representación de neurona y tipos de función.

Figura 3. Red recursiva de primer orden.

F y f son funciones lineales



$$y_i = f_i(\sum_{j=1}^N w_{ij} u_j)$$

(figura 2); si el vector de factores de peso w se encuentra fijo, entonces la red se encuentra en la fase de reconocimiento; por el contrario, si se le permite modificar sus factores de peso, se encuentra en la fase de aprendizaje. En la figura 3 se presenta un ejemplo de una red recursiva de primer orden que se encuentra en la fase de reconocimiento.

Por otro lado, la fase de aprendizaje va a ser el proceso a partir del cual la ANN obtiene los parámetros que mejor se adaptan a la asociación de entradas y salidas. Se puede realizar una clasificación básica de este aprendizaje basándose en la forma en que se modela el entorno; así se tendrá el aprendizaje:

No supervisado. No disponemos de información de la respuesta prevista, tampoco se conoce el error cometido.

Por refuerzo. Se dispone de información del error cometido, aunque ésta sea imprecisa.

Supervisado. Se dispone de un grupo de entrenamiento completo, o sea, datos de entrada y datos de salida (1); la salida de la red es la respuesta esperada para esa entrada.

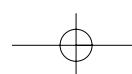
ANN, en los procesos de identificación

Tres preguntas importantes, relativas a las características de la arquitectura más apropiada, surgen cuando se usan ANN en procesos de identificación, las cuales son: número de capas, número de neuronas por capa y funciones de activación.

Muchos autores han investigado al respecto, Kolmogorov, Hecht-Nielsen, Kurkova, Hornik, Cybenko, concluyendo que las ANN de dos capas, con funciones de tipo tangente hiperbólica o sigmoidal en la capa oculta, van a ser predictores universales^{12,3}.

La red de perceptrón multicapa (MLP) tiene la capacidad de modelar las relaciones funcionales simples, también como las complejas. Ha sido probada, a lo largo de un gran número de aplicaciones prácticas, por Demuth & Beale⁴.

Está formada por dos capas, una oculta y otra de salida (tres si se considera la capa de entrada, capa transparente), con funciones de activación, lineales o tangentes hiperbólicas (f, F). Los factores (vector θ o de forma alternativa matrices w y W) son los parámetros ajustables de la red y están escogidos de un



grupo de entrenamiento, que es el grupo de entradas y de salidas $u(t)$ e $y(t)$ a identificar. En la figura 4 se muestra una red MLP con una capa de salida y otra capa oculta.

En cambio, no hay trabajos tan concluyentes en lo relativo al número de neuronas por capa, teniendo en cuenta que este número es adecuado para evitar sobreajustes en los datos y evitar que la estructura del modelo elegida contenga demasiados factores de peso. Por consiguiente, se deberán tratar estos posibles excesos a través de una estrategia de reducción OBS (neurocirujano óptimo)^{9,10}.

Otro aspecto importante será conocer cuál de los algoritmos permite un aprendizaje más rápido, para determinar el conjunto óptimo de factores de peso en la ANN. Cada algoritmo se caracteriza por cómo se busca y se selecciona la dirección y el paso; algunos de los algoritmos más usados en la identificación son: método de Lenvenberg-Marquardt[6], método recursivo de predicción de error de Gauss-Newton¹⁵, versión por tandas del algoritmo de retropropagación¹¹ y la versión recursiva de retropropagación¹¹.

Método de aprendizaje

A partir de la figura 2, se definen todas las variables asociadas a la red; donde:

$$u_i(t) = (u_1(t), u_2(t), u_3(t), K, u_n(t))$$

es el vector de entradas del patrón de entrenamiento;

$$y_i(t) = (y_1(t), y_2(t), y_3(t), K, y_m(t))$$

es el vector de salida esperado;

$$f_1(\cdot), f_2(\cdot), K, f_j(\cdot)$$

son las funciones de activación de las neuronas en la capa oculta;

$$F_1(\cdot), F_2(\cdot), K, F_i(\cdot)$$

son las funciones de activación de la capa de salida;

$$w_{01}, w_{02}, K, w_{jl}$$

son los pesos de las entradas a la capa oculta, y

$$W_{01}, W_{02}, K, W_{ij}$$

son los pesos de la capa oculta a la capa de salida.

Las polarizaciones o *bias* son neuronas unidas al resto con valor 1 y proporcionan niveles de activación independientes de los datos de entrenamiento; sus pesos se modifican de la misma forma que el resto, w_{j0} y W_{j0} .

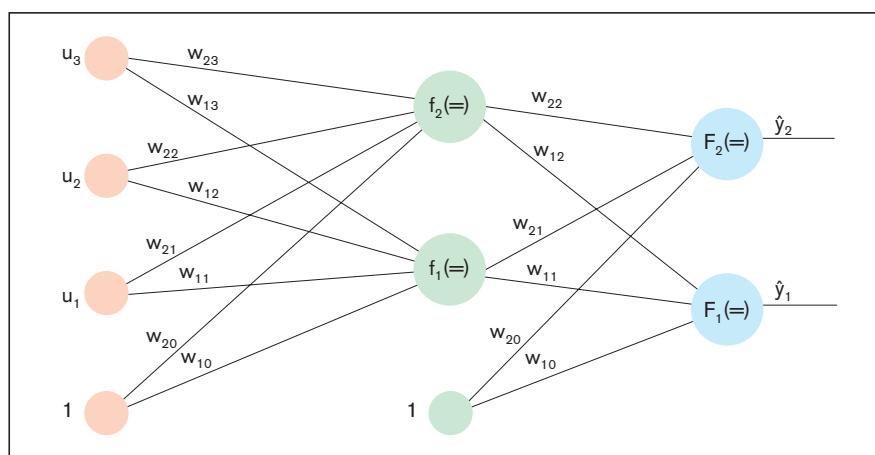


Figura 4. Red MLP conectada de dos capas (una oculta y una de salida), con tres entradas y dos unidades de salida.

1. Propagación hacia la capa de salida

El vector de entradas $u(t)$ se propaga hacia la capa oculta que cuenta con j neuronas, la excitación de una neurona oculta (s_j) quedará definida en la ecuación (10); la salida de la neurona oculta quedará como función del vector de entradas asociado a la función de activación (11).

$$s_j = \sum_{i=1}^n w_{ji} u_i + w_{j0} \quad (10)$$

$$y_j = f_j(s_j) \quad (11)$$

Trasladando lo anterior a la excitación de la neurona de salida, ésta queda definida por las ecuaciones (12) y (13).

$$s_i = \sum_{j=1}^q W_{ij} y_j + W_{i0} \quad (12)$$

$$\hat{y}_i = F_i(s_i), \text{ salida estimada } i. \quad (13)$$

Por consiguiente, la formula general de la red MLP puede formarse por la combinación de las ecuaciones anteriores de (10) a (13); como:

$$\hat{y}_i(w, W) = F_i \left(\sum_{j=1}^q W_{ij} y_j(w) + W_{i0} \right) =$$

$$= F_i \left(\sum_{j=1}^q W_{ij} f_j \left(\sum_{l=1}^n w_{jl} u_l + w_{j0} \right) + W_{i0} \right)$$

2. Modificación del valor de los pesos de la capa oculta a la capa de salida

La estrategia utilizada es la aplicación del criterio cuadrático medio, mediante la minimización del mismo, por lo que la función de pérdidas o de error dependerá de este criterio, ecuación (14). Entonces el error entre la salida estimada y la salida ideal será $y(t) - \hat{y}(t)$, que aplicado en cada neurona de la capa de salida dará lugar al error en la neurona, ecuación (15).

$$E = V_N(\theta, z^N) = \frac{1}{2N} \sum_{i=1}^N [y_i(t) - \hat{y}_i(t|\theta)]^T [y_i(t) - \hat{y}_i(t|\theta)] \quad (14)$$

$$\varepsilon_i(t) = y_i(t) - \hat{y}_i(t) \quad (15)$$

La función de error se minimiza a partir de la elección de la dirección de búsqueda, opuesta a la dirección del gradiente del error E , respecto a los pesos W_{ij} ; por consiguiente, el vector de pesos se modifica en función de una tasa de aprendizaje η , de la que depende en gran medida la velocidad y tiempo de entrenamiento; a este método se le conoce como de descenso del gradiente, siendo la dirección del gradiente la indicada en la ecuación (16). Se define

$$\delta_i = -\frac{\partial E}{\partial s_i}$$

como la sensibilidad de la neurona de salida i ante el vector de entrada y , dado que es posible la utilización de la regla de la cadena, debido a que las funciones de activación son diferenciales, la sensibilidad puede reescribirse en la ecuación (17).

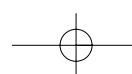
$$\nabla E = \frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial s_i} \frac{\partial s_i}{\partial W_{ij}} \quad (16)$$

$$\delta_i = (y_i(t) - \hat{y}_i(t)) \frac{\partial F_i}{\partial s_i} = (y_i(t) - \hat{y}_i(t)) F'_i(s_i) \quad (17)$$

Si se representa la

$$\frac{\partial s_i}{\partial W_{ij}}$$

mediante la ecuación (18), entonces ∇E (16) se puede reescribir en la ecuación (19).



$$\frac{\partial s_i}{\partial W_{ij}} = \frac{\partial}{\partial W_{ij}} \left(\sum_{j=1}^q W_{ij} y_j + W_{i0} \right) = y_j \quad (18)$$

$$\nabla E = \frac{\partial E}{\partial W_{ij}} = -(y_i(t) - \hat{y}_i(t)) F_i(s_i) y_j \quad (19)$$

Si el vector de pesos se modifica proporcionalmente en dirección contraria al gradiente y se tiene en cuenta el momento μ , con el fin de reducir el tiempo de aprendizaje y la aparición de mínimos locales, haciendo que la variación de valor de los pesos se realice en la misma dirección, excepto ante cambios de sentido global. La regla de actualización podrá formularse conforme a la ecuación (20), donde.

$$\Delta W_{ij} = -\eta \frac{\partial E}{\partial W_{ij}} = -\eta \nabla E$$

$$W_{ij}(t) = W_{ij}(t-1) + \Delta W_{ij}(t-1) + \mu(W_{ij}(t-1) - W_{ij}(t-2)) \quad (20)$$

Aplicado al cálculo de los pesos de la capa oculta a la capa de salida, en función de la activación de la neurona, se tendrá: funciones de activación lineal, ecuaciones (21) a (22), funciones de tangente hiperbólica, ecuaciones (23) a (24).

$$\hat{y}_i = F_i(s_i) = s_i \quad (21)$$

$$F_i = 1$$

$$W_{ij}(t) = W_{ij}(t-1) + \eta(y_i(t-1) - \hat{y}_i(t-1))y_j + \mu(W_{ij}(t-1) - W_{ij}(t-2)) \quad (22)$$

$$\hat{y}_i = F_i(s_i) = \frac{2}{1 + e^{-2s_i}} - 1 \quad (23)$$

$$F_i = (F_i(s_i) + 1)(1 - F_i(s_i)) = (\hat{y}_i + 1)(1 - \hat{y}_i)$$

$$\nabla E = -(y_i(t) - \hat{y}_i(t))(\hat{y}_i(t) + 1)(1 - \hat{y}_i(t))y_j \quad (24)$$

$$W_{ij}(t+1) = W_{ij}(t) + \eta \nabla E + \mu(W_{ij}(t) - W_{ij}(t-1))$$

3. Modificación del valor de los pesos de la capa de entrada a la capa oculta

La función de pérdidas o de error en este caso se corresponde con el cálculo a partir de la salida estimada en la capa oculta, la diferencia estriba en que en este caso no se conoce la salida (salida real), pero sí la estimada.

$$E = V_N(\theta, z^N) \quad (25)$$

$$\varepsilon_i(t) = y_i(t) - \hat{y}_i(t) = y_i(t) - F_i(s_i) \quad (26)$$

$$\varepsilon_i(t) = y_i(t) - F_i \left(\sum_{j=1}^q W_{ij} y_j + W_{i0} \right) \quad (27)$$

$$\frac{\partial E}{\partial w_{jl}} = -\sum_{i=1}^m (y_i(t) - \hat{y}_i(t)) \frac{\partial \hat{y}_i(t)}{\partial s_i} \frac{\partial s_i}{\partial y_j} \frac{\partial y_j}{\partial w_{jl}} \quad (28)$$

$$\frac{\partial E}{\partial w_{jl}} = -\sum_{i=1}^m (y_i(t) - \hat{y}_i(t)) F_i(s_i) W_{ij} f_j(s_j) u_i \quad (29)$$

Sustituyendo la sensibilidad de la neurona de la salida i (19), en la ecuación (29); la dirección del gradiente del error respecto a los pesos se reescribe, ecuación (30); siendo la sensibilidad de la neurona j la expresada en la ecuación (31).

$$\frac{\partial E}{\partial w_{jl}} = -f_j'(s_j) \sum_{i=1}^m W_{ij} \delta_i u_i \quad (30)$$

$$\delta_j = f_j'(s_j) \sum_{i=1}^m W_{ij} \delta_i \quad (31)$$

Ésta es la esencia del método de entrenamiento de retropropagación, el cual se basa en distribuir los errores que aparecen en las neuronas de salida hacia las neuronas ocultas con las que existe conexión. Por consiguiente, los pesos de las capas de entrada a las capas ocultas se modifican conforme a la siguiente expresión:

$$w_{jl}(t+1) = w_{jl}(t) + \eta \delta_j u_l + \mu(w_{jl}(t) - w_{jl}(t-1)) \quad (32)$$

Aplicado al cálculo de los pesos de la capa de entrada a la capa oculta, para una función de activación tangente hiperbólica en esta capa, los pesos se modificarán de acuerdo con la ecuación (34).

$$f_j = (f_j(s_j) + 1)(1 - f_j(s_j)) = (y_j + 1)(1 - y_j) \quad (33)$$

$$w_{jl}(t+1) = w_{jl}(t) + \eta(y_j(t) + 1)(1 - y_j(t)) \sum_{i=1}^m W_{ij} \delta_i u_i + \mu(w_{jl}(t) - w_{jl}(t-1)) \quad (34)$$

Procesos de identificación

Si se parte de una observación (1), el objetivo del proceso de identificación

consistirá en determinar los factores de peso a ajustar (especificados por w , W o θ), partiendo de la observación Z^N .

Entonces el aprendizaje supervisado deberá encontrar el vector de parámetros θ , mediante la aproximación al error de predicción; basándose en la medida de la proximidad, en términos del criterio de error de media de cuadrados, ecuación (14), donde el vector de pesos θ será minimizado $V_N(\theta, Z^N)$, ecuación (35). El esquema de minimización iterativa se define en la ecuación (36), donde $f(t)$ es la dirección de búsqueda y $\mu(t)$ es el tamaño del paso.

$$\hat{\theta} = \arg \min_{\theta} V_N(\theta, Z^N) \quad (35)$$

$$\theta(t+1) = \theta(t) + \mu(t) f(t) \quad (36)$$

Selección y estructuras del modelo

Después de realizar la observación Z^N , debe elegirse el grupo de regresores y la estructura de red; para la selección de los regresores, como natural extensión de la identificación lineal, nos apoyamos en los modelos lineales [18]. Las estructuras de los modelos se definirán por tres parámetros. $\pi(t)$: vector de regresión, θ vector de parámetros o pesos y $g(\cdot)$ función de la neurona.

NNARX:

Vector de regresión:

$$\varphi(t) = [y(t-1) Ky(t-n_a) u(t-n_k) Ku(t-n_b - n_k + 1)]^T \quad (37)$$

Predictor:

$$\hat{y}(t|\theta) = \hat{y}(t|t-1, \theta) = g(\varphi(t), \theta) \quad (38)$$

NNOE:

Vector de regresión:

$$\varphi(t) = [\hat{y}(t-1|\theta) \hat{y}(t-n_a|\theta) u(t-n_k) u(t-n_b - n_k + 1)]^T \quad (39)$$

Predictor:

$$\hat{y}(t|\theta) = g(\varphi(t), \theta)$$

NNARMAX

Variante 1

Vector de regresión:

$$\varphi(t) = [y(t-1) y(t-n_a) u(t-n_k) Ku(t-n_b - n_k + 1) \varepsilon(t-1) \varepsilon(t-n_c)]^T$$

$$= [\varphi_1^T(t) \varepsilon(t-1) \varepsilon(t-n_c)]^T \quad (40)$$

Donde $\varepsilon(t)$ es el error de predicción,

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

Predictor:

$$\hat{y}(t|\theta) = g(\varphi_1(t), \theta) + (C(q^{-1}) - 1)\varepsilon(t)$$

$$C(q^{-1}) = 1 + c_1 q^{-1} + L + c_{nc} q^{-nc}$$

Variante 2

Vector de regresión:

$$\varphi(t) = [y(t-1) \dots y(t-n_a) u(t-n_k)]$$

$$Ku(t-n_b - n_k + 1) \varepsilon(t-1) \quad (41)$$

$$K\varepsilon(t-n_c)]^T$$

Predictor:

$$\hat{y}(t|\theta) = g(\varphi(t), \theta)$$

NNFIR

Vector de regresión: (42)

$$\varphi(t) = [u(t-n_k) \dots u(t-n_b - n_k + 1)]^T$$

Predictor:

$$\hat{y}(t) = [u(t-n_k) Ku(t-n_b - n_k + 1)]^T$$

Ensayo. Columna de destilación para la separación de la mezcla de metanol y agua

El proceso de destilación presenta un comportamiento no lineal entre la concentración y el flujo de vuelta, que se produce en la parte superior de la columna; para más detalles ver [8]. Por consiguiente, la ecuación de diferencias de segundo orden, que representa la estructura del modelo, será tomada como si fuera de caja negra y a partir de aquí, excitada con una señal distribuida de forma uniforme con cinco muestras constantes.

$$\begin{aligned} y(t) = & 0.353y(t-1) + 0.54u(t-1) - \\ & - 1.65u(t-2) - 1.68u^2(t-1) + 1.8u^2(t-2) \\ & + 0.71y(t-1)u(t-1) - 0.3y(t-2) \\ & u(t-2) + 2 + \nu(t) \quad (43) \end{aligned}$$

Al sistema se le somete a diferentes tipos de ruidos, con el objeto de ver cómo el entrenamiento de la ANN y el posterior reentrenamiento, una vez que se ha obtenido la estructura óptima de red (OBS), va a ser capaz de predecir la salida.

La estructura de la red vendrá definida: por el número de neuronas en la capa oculta (se han utilizado de 10 a 12 neuronas); por el número de parámetros (factores de peso), se han utilizado parámetros arbitrarios de inicio y rearrancado varias veces con el fin de evitar mínimos locales; por la función que debe desarrollar la neurona, tangente hiperbólica en la capa oculta y lineal en la capa

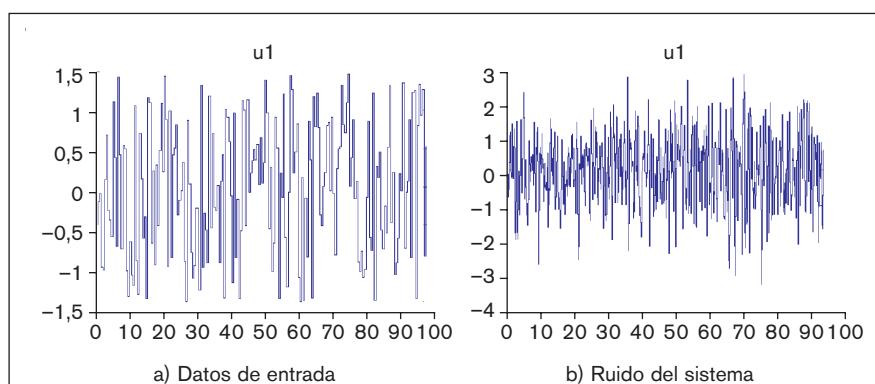


Figura 5. a) Datos de entrada generados con distribución uniforme escalados entre $-1,5$ y $1,5$, de cinco muestras constantes con media cero y varianza $0,708$. b) Ruido del sistema $\nu(t) = (0,5/1 - 1,2z^{-1} + 0,7z^{-2})e$ intervalo de muestreo $0,1$ sg. $e(t)$ es ruido blanco gaussiano.

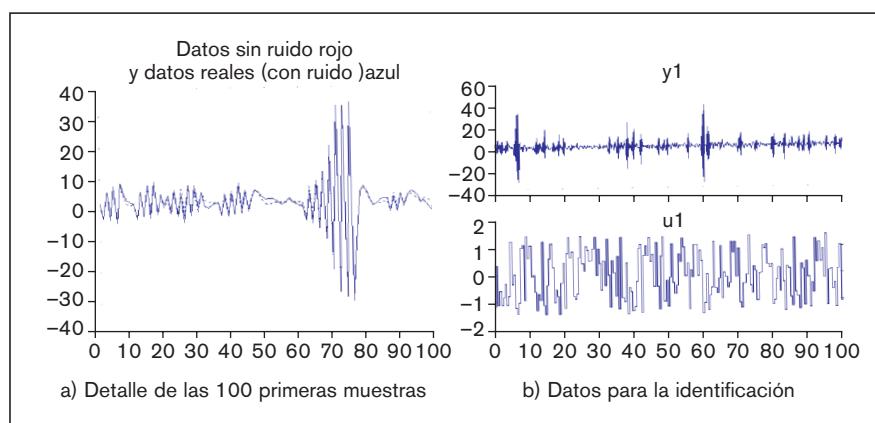


Figura 6. a) Detalle de las 100 primeras muestras de datos de salida con y sin ruido. b) Datos para la identificación. Se generan dos colecciones: los primeros 500 serán de estimación y los 500 restantes serán de validación; ambos se deberán escalar antes del entrenamiento a un valor con media cero y varianza 1.

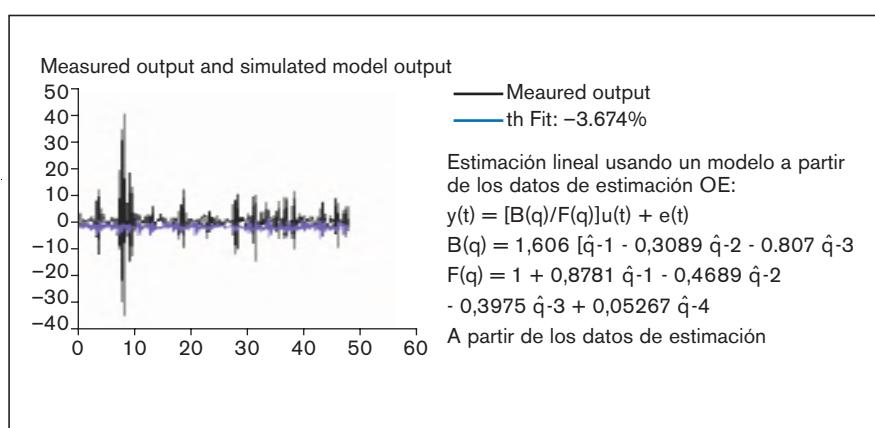
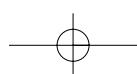


Figura 7. Representa la comparación entre los datos de salida real con la estimada, utilizando un modelo OE con órdenes en el polinomio $B=3$ y $F=4$; a partir del retraso uno (3 4 1). Se observa que no es capaz de aproximarse en nada, lo que garantiza la fuerte no linealidad del sistema.

de salida; por el orden del modelo (n_a, n_b, n_c) y por los algoritmos de búsqueda y dirección, en este caso se ha utilizado el método de Levenberg-Marquardt, estándar de minimización que utiliza el criterio de error de media cuadrada; en

el primer entrenamiento se añade un término de regularización¹⁴.

La arquitectura óptima se obtiene en base a la regularización, ya que el entrenamiento obedece a criterios que toman mínimos locales y es común que se



Ruido	Órdenes del modelo na, nb, nc, nk y estructura	Fe sobre datos de estimación	Fe sobre datos de validación	Fe sobre datos de validación finales
Sin ruido	2 4 2 1 / (10 th-1)	99,63	97,19	97,19
0,6* $e(t)$	2 4 2 1 / (10 th-1)	85,97	64,10	78,62
2* $e(t)$	2 4 2 1 / (10 th-1)	69,78	-28,74	47,61
2* $e(t)$	3 4 0 1 / (12 th-1)	68,32	33,12	67,95
$(2/1-1,2z^{-1}+0,7z^{-2})e(t)$	3 4 0 1 / (12 th-1)	67,34	36,89	52,13
$(2/1-1,2z^{-1}+0,7z^{-2})e(t)$	2 4 2 1 / (10 th-1)	85,93	-5,98	-12,76

Tabla 1. Otros ensayos con diferentes valores de ruido sobre el mismo sistema.

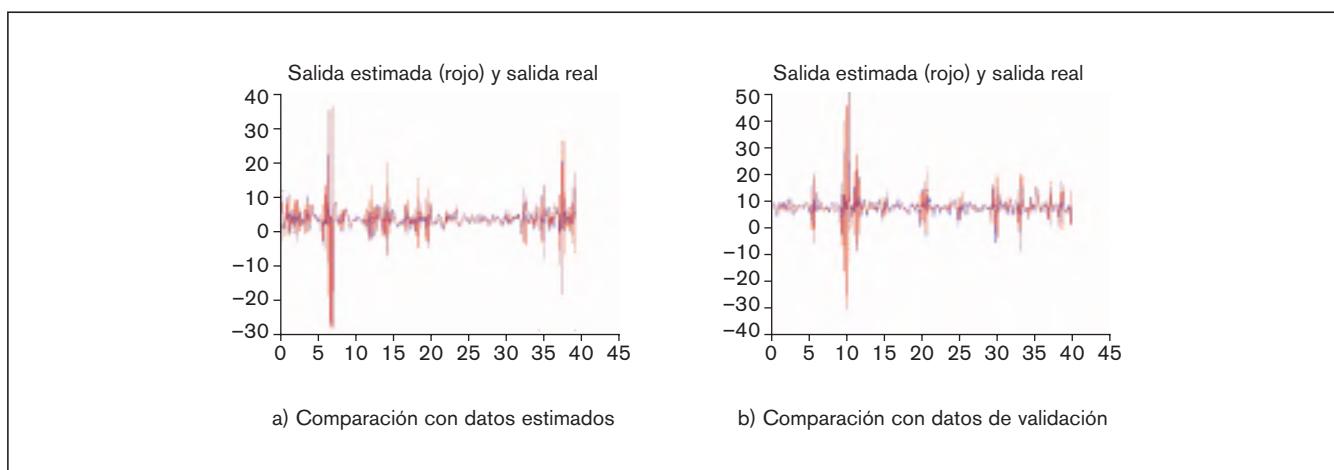


Figura 8. a) Comparación de datos de salida real con datos estimados a partir de la predicción de un paso adelante, el porcentaje de la salida que se explica con el modelo (fe) es del 86,78%. b) Comparación de datos de salida real con datos de validación estimados a partir de la predicción de un paso adelante, fe del 76,14%. La red ha sido entrenada usando el método de Lenverberg-Marquardt usando un máximo de 400 iteraciones y regularización de 1e-3, con 10 neuronas en la capa oculta con función tangente hiperbólica (th) y una capa de salida con función lineal (l); el orden del modelo es (2 4 2 1).

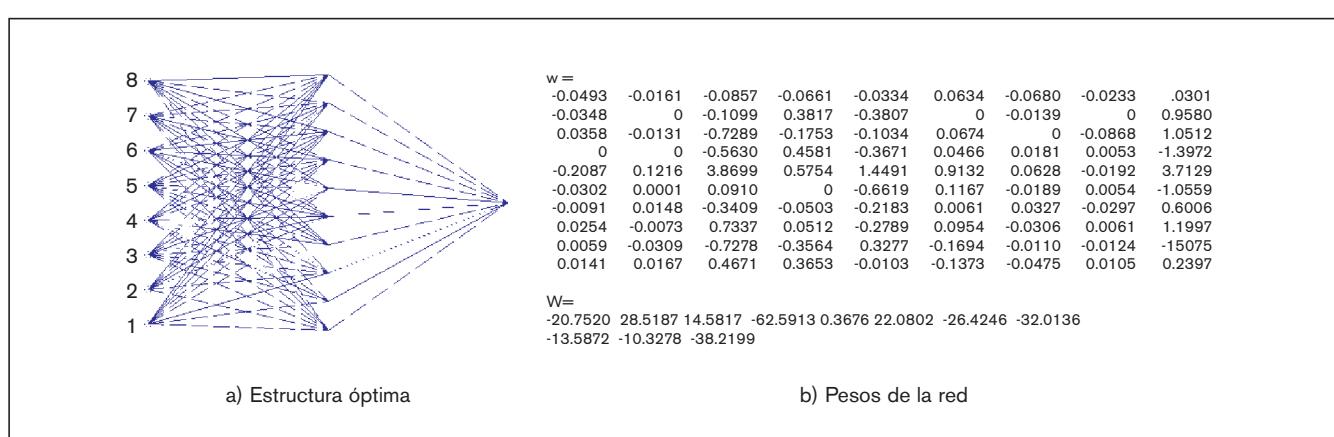


Figura 9. a) Estructura óptima después de utilizar la estrategia del neurocirujano OBS y el reentrenamiento sin regularización. b) Pesos de la red.

pueda acabar en un mínimo local no válido. Por consiguiente, la regularización suaviza los criterios y elimina, en gran medida, los mínimos locales; es decir, se deben eliminar los pesos superfluos y buscar el mínimo error sobre los datos de validación (para la validación de la red).

La estrategia que se ha utilizado después de observar que la validación cru-

zada sobre los datos de validación no es satisfactoria ha sido utilizar la estrategia OBS, que consiste en encontrar los errores de prueba, estimación más fiable de la generalización del error y seleccionar la ANN con error, sobre la colección de validación, más pequeño.

Por último, una vez que la ANN cuenta con la arquitectura óptima, se ha vuelto a realizar un nuevo ejercicio de

entrenamiento, pero sin regularización, recalibrando los parámetros o pesos que definen la ANN. El ensayo se ha realizado usando Matlab con System Identification Toolbox¹⁶ y Neural Network based System Identification Toolbox¹⁹. Los resultados gráficos de estos ensayos se muestran en las figuras 5, 6, 7, 8, 9, 10 y algunos datos significativos se reflejan en la tabla 1.

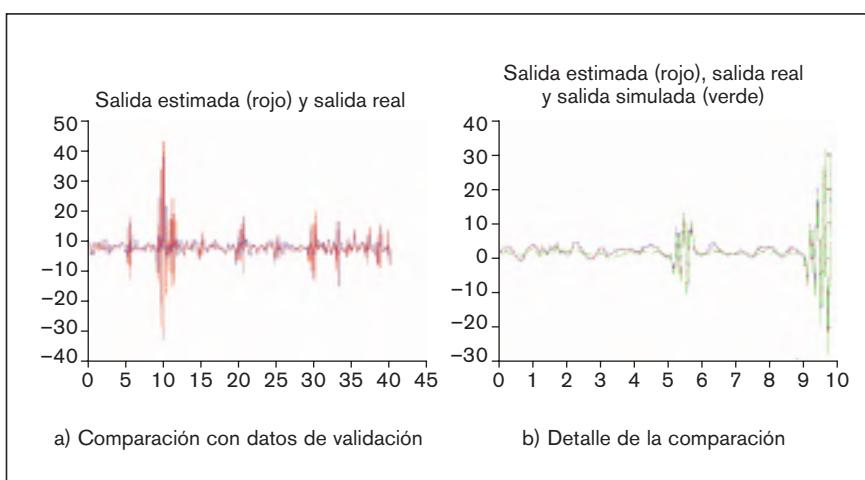


Figura 10. a) Comparación de datos de salida real (datos de validación) con datos estimados a partir de la predicción de un paso adelante, fe es del 79,81%. b) Detalle de la comparación de datos de salida real (datos de validación) estimados a partir de la predicción de un paso adelante y mediante simulación.

Conclusiones y trabajos futuros

Se ha mostrado cómo el modelo Narmax es una representación general de los sistemas finitos realizables, donde se incluyen el sistema lineal, el bilineal, el de salida afín y el racional. Se han expuesto las características más importantes de las ANN, cuando participan en procedimientos de identificación. Se ha demostrado cómo se realiza la actualización de los pesos en el proceso de aprendizaje de la ANN. Se ha ilustrado un modelo de la columna de destilación del metanol, que se ha desarrollado usando técnicas de identificación neurológica óptima; red neurológica que se ha entrenado y a la que se le ha aplicado OBS para predecir el comportamiento de la concentración, en función del flujo de vuelta; de igual forma se ha observado la capacidad de predicción del modelo, dependiendo del ruido existente en el sistema.

Nuestro trabajo futuro se desarrollará en la búsqueda de una metodología que conste de una serie de reglas, etapas, pautas y decisiones, capaces de hacer que el modelo final sea representativo del modelo identificado. Este procedimiento de identificación, que irá desde el diseño del experimento, la adquisición y tratamiento de los datos, la elección de la estructura del modelo y la selección de los parámetros, hasta la fase de validación del modelo, deberá tener como aptitudes, la capacidad de poderse ejecutar de forma automática.

Referencias

- S. A. Billings e I. J. Leontaritis, "Parameter estimation techniques for nonlinear systems", Proc. 6th IFAC Symposium on Identification and System Parameter Estimation, Washintong, USA, 1982. pp. 505-510.
- S. Chen y S. A. Billings, "Representations of nonlinear systems: The Narmax model", Taylor & Francis Ltd. Int. J. Control, Vol. 49, N.º 3, 1989. pp. 1013-1032.
- G. Cybenko, "Aproximation by superpositions of sigmoidal function", Math. Control, Signals and System, vol. 2, n.º 4, 1989
- H. Demuth & M. Beale "Neural Network Toolbox: For use with Matlab". The MathWorks Inc., Natick, MA, 2005.
- M. España e I. D. Landau, Automatica, Vol. 14, 345, 1978.
- R. Fletcher "Practical Methods of Optimization", Wiley, 1987.
- M. Fließ, y D. Normand-Cyrot, "On the approximation of nonlinear systems by some simple state-space models", Proc. 6th IFAC Symposium on Identification and System Parameter Estimation, Washintong, USA, 1982. pp. 511-514.
- R. Haber and L. Keviczky, "Nonlinear System Identification-Input-output Modelling approach", Vol 2, Kluwer Academic Publishers 1999.
- L. K. Hansen & M. W. Pedersen "Controlled Growth of Cascade Correlation Nets", Proc. ICANN 94, Sorrento, Italia, Eds. M. Marinaro & P. G. Morasso, pp. 797-800, 1994.
- B. Hassibi, D. G. Stork "Second Order Derivatives for Network Pruning: Optimal Brain Surgeon", NIPS 5, Eds. S. J. Hanson et al., 164, San Mateo, Morgan Kaufmann, 1993.
- J. Hertz, A. Krogh & R. G. Palmer "Introduction to the Theory of Neural Computation", Addison-Wesley, 1991.
- K. Hornik, K. Stinchcombe, M. and White H. "Multilayer Feedforward Networks are Universal Approximators", Neural Networks, vol. 2 n.º 5, 1989.
- I. J. Leontaritis, y S. A. Billings, Int. J. Control, Vol. 41, 303, 1985.
- J. Larsen & L. K. Hansen "Generalization Performance of Regularized Neural Network Models", Proc. of the IEEE Workshop on Neural networks for Signal Proc. IV, Piscataway, New Jersey, pp. 42-51, 1994.
- L. Ljung, "System Identification. Theory for the user", Prentice Hall, 1999.
- L. Ljung. "System Identification Toolbox: For use with Matlab". The MathWorks 2005.
- R. Mohler, "Bilinear Control Processes", Academic Press, 1973.
- M. Nørgaard, O. Ravn, N. K. Poulsen, L. K. Hansen "Neural networks for Modelling and Control of Dynamic Systems", Springer-Verlag, Londres, UK, 2000.
- M. Nørgaard. "Neural network Based System Identification Toolbox" Report Technical Technico 00-E-891, Department of Automation Technical University of Dinamarca, 2000.
- J. Sedano, J. R. Villar "Aproximación a la teoría de la identificación de sistemas, "system identification", estado del arte ", Técnica Industrial 1/05, pp. 30-36, 2005
- E. D. Sontag, "Polynomial response maps", Lecture Notes in Control and Information Sciences 13, Springer-Verlag, 1979.

AUTORES

Javier Sedano Franco

jsedano@ubu.es

Ingeniero técnico industrial en Electrónica e ingeniero de Organización Industrial por la Universidad de Burgos. Ha participado en el desarrollo y la dirección de cerca de una treintena de proyectos de desarrollo tecnológico y desarrollo precompetitivo financiados en programas regionales y nacionales, siendo prototipos la mayoría de los casos. Formador de profesionales en todas las disciplinas de las tecnologías productivas y de la dirección de proyectos. Ex responsable del departamento de Sistemas Electrónicos del área de I+D del ITCL y actualmente profesor asociado del área de Tecnología Electrónica de la Universidad de Burgos.

Ángel Alonso Álvarez

Doctor ingeniero de Telecomunicación por la Universidad Politécnica de Madrid. Catedrático de Universidad en la Universidad de León en el área de Ingeniería de Sistemas y Automática. Director de investigación en el Instituto de Automática y Fabricación. Autor de diversas publicaciones y trabajos de investigación. Actualmente trabaja en el desarrollo de un sistema basado en el conocimiento para la ingeniería de control.

José Ramón Villar Flecha

diejf@unileon.es

Ingeniero industrial. Obtuvo el grado de doctor en Ingeniería Informática por la Universidad de León en el programa de Tecnologías de la Información del año 2002. Ha trabajado en diversos ámbitos de la ingeniería, como ingeniero de proyectos en la consultoría Istrón, como consultor en el Instituto de Automática y Fabricación de la Universidad de León, en el desarrollo de proyectos de investigación con financiación pública y privada. Tiene publicados artículos internacionales de prestigio, así como una treintena de contribuciones a congresos tanto nacionales como internacionales.