QÜESTIIÓ, Vol 19, 1, 2, 3 pp. 187-215, 1995

ANÁLISIS COMPARATIVO DE CÁLCULO DE PREVISIONES UNIVARIABLES Y FUNCIÓN DE TRANSFERENCIA, MEDIANTE LAS METODOLOGÍAS DE BOX-JENKINS Y REDES NEURONALES

DAVID DE LA FUENTE GARCÍA* y RAÚL PINO DÍEZ*

En este trabajo se presenta una comparación de los resultados obtenidos en el cálculo de previsiones de series temporales univariables y de función de transferencia, mediante la metodología de Box-Jenkins y la utilización de Redes Neuronales Artificiales. Los resultados obtenidos indican que la aproximación de las redes neuronales, ha sido muy satisfactoria, tanto en lo referente a la bondad de las previsiones obtenidas, como, sobre todo, por la facilidad de su utilización. En las Redes Neuronales Artificiales, o "Redes de Conocimiento", todavía no son muy conocidos los mecanismos matemáticos internos que justifican su funcionamiento, sin embargo, esto no es óbice para comprobar que los resultados que dan son muy interesantes. El problema principal es que, al menos por el momento, el desarrollo de las redes neuronales es heurístico, no existen reglas fijas que determinen la estructura de la red neuronal apropiada a cada caso de estudio. En este trabajo se propone la utilización de la metodología de Box-Jenkins como un paso previo al diseño de la estructura de la red neuronal.

Comparative analysis of univariate and transfer-function forecasting using neural networks and Box-Jenkins techniques.

Key words: Forecasting, Multivariate Time-Series, Neural Network, Back Propagation, Training.

* David de la Fuente García y Raúl Pino Diez. Escuela Técnica Superior de Ingenieros Industriales e Ingenieros Informáticos de Gijón. Universidad de Oviedo. Carretera de Castiello, s/n. 33204. GIJÓN (Asturias).

E-mail: David @etsiig.uniovi.es

- Article rebut el juliol de 1993.

- Acceptat l'abril de 1995.

1. INTRODUCCIÓN

El análisis de series temporales, es una herramienta estadística muy importante en el estudio del comportamiento de datos dependientes del tiempo y la predicción de valores futuros. Una serie temporal, es una secuencia de valores medidos en unidades de tiempo discretas o continuas.

Una serie temporal multivariable, consiste en una secuencia de valores de varias variables contemporáneas cambiando con el tiempo. Un caso aparte, es cuando las variables medidas están significativamente correlacionadas, por ejemplo, cuando atributos similares están siendo medidos en diferentes localizaciones geográficas. En la previsión de un valor nuevo para cada variable, la mejor predicción, se conseguirá teniendo en cuenta las variaciones de las otras variables.

Históricamente, podemos decir que en la década de los ochenta, había muchas técnicas disponibles para el análisis de series temporales, las cuales asumían relaciones lineales entre las variables (Box- Jenkins, 1976). Pero en el mundo real los datos no presentan relaciones simples y es difícil hacer predicciones seguras. Las relaciones lineales y sus combinaciones a menudo son inadecuadas para hacer previsiones. Por ello, parece necesario que sean modelos no lineales, los que sirvan para analizar los datos temporales del mundo real.

Los modelos lineales de series temporales presentan además, ciertas desventajas, como la imposibilidad de explicar cambios repentinos de amplitud muy grande y en intervalos irregulares de tiempo (Tong, 1983).

Estudios posteriores (Tiao y Tsay, 1989), analizaron otros problemas del modelado lineal multivariable. Para resolverlos, se utilizaron modelos estadísticos no-lineales como "modelos de umbral" y "modelos bilineales" sugeridos por Tong (1990). Por otra parte, Granger y Newbold (1986), sugirieron utilizar transformaciones no lineales de los datos originales antes de realizar el modelado lineal tradicional. Farmer y Sidorowich (1987), mejoraron significativamente las predicciones sobre series temporales caóticas utilizando métodos de aproximaciones locales.

Desgraciadamente, a pesar de que en la década pasada se avanzó considerablemente, la formulación de modelos no lineales razonables es una tarea extremadamente difícil, debido a las simplificaciones hechas en la etapa de modelado, por ejemplo, omitir parámetros que son desconocidos o que no parece que afecten directamente a los datos observados etc.

Por todo ello, como el objetivo es hacer buenas previsiones, buscamos la alternativa de la utilización de redes neuronales artificiales para el cálculo de predicciones cuantitativas.

En este trabajo, hacemos una somera descripción de la metodología de Box-Jenkins para modelado temporal univariable, así como funciones de transferencia para el caso de una entrada y una salida. A continuación, describimos la aproximación de Redes Neuronales para finalizar comparando con varios ejemplos, los resultados obtenidos mediante ambos enfoques, en el modelado univariante y función de transferencia de una entrada y una salida.

2. MODELADO UNIVARIANTE Y DE FUNCION DE TRANSFERENCIA DE BOX-JENKINS

Se sabe que el procedimiento ARIMA (autorregresivo, integrado, media móvil), identifica, calcula estimaciones y previsiones de modelos, utilizando la metodología descrita por Box-Jenkins (1976). Los procedimientos ARIMA, permiten modelar una serie temporal discreta como una función de términos autorregresivos y términos media móvil pudiendo incluirse en el modelo factores estacionales y no estacionales de cada tipo, llamados habitualmente: MA, AR, SMA y SAR.

El proceso de modelado de series temporales según Box-Jenkins, consta de varias fases:

1- Identificación y propuesta de un modelo para los datos observados.

2- Estimación de los parámetros y contrastes.

3- Crítica y diagnosis del modelo.

4- Validación del modelo. (Si no es válido se propone otro modelo en el paso 1).

5- Previsiones.

También se modelan, en su caso, la serie diferenciada o la no diferenciada para eliminar tendencias, mediante la utilización de las diferencias estacionales y no estacionales.

Existen en el mercado bastantes paquetes comerciales que utilizan la metodología de Box-Jenkins univariable, STATGRAPHICS, BMDP, RATS, TSP, SPSS etc. Algunos de ellos, utilizan funciones de transferencia, como el BMDP, y muy pocos modelado multivariable (varias entradas y varias salidas), por ejemplo, SCA y RATS (solo modelo AR). Incluso, existen algunos paquetes de cálculo automático de previsiones univariantes como son AUTOBOX, FOCA etc. y últimamente han aparecido sistemas expertos en previsión univariante, como el SCA EXPERT. Para la realización de este trabajo, en el caso de modelado univariante hemos escogido el programa

STATGRAPHICS por su sencillez mientras que para el modelado de función de transferencia, hemos utilizado el BMDP.

2.1. Modelado Univariante

La forma básica del modelo que va a ser ajustado, es la siguiente:

$$W_t = \mu + \frac{\theta(B) \cdot \theta_s(B)}{\phi(B) \cdot \phi_s(B)} \cdot a_t$$

Este modelo, expresa los datos como una combinación de los valores de la serie y los valores de una entrada aleatoria, donde:

t	Índice de tiempo.
B	Operador de retardo.
W_t	Datos originales o una diferencia de estos datos.
μ	Término constante.
$\theta(B)$	Operador media móvil no estacional $(1 - \theta_1 B - \cdots - \theta_q B^q)$
$\phi(B)$	Operador autorregresivo no estacional $(1 - \phi_1 B - \cdots - \phi_p B^p)$
$\theta_s(B)$	Operador media móvil estacional $(1 - \theta_1 B^s - \cdots - \theta_Q B^{Qs})$
$\phi_s(B)$	Operador autorregresivo estacional $(1 - \phi_1 B^s - \cdots - \phi_P B^{Ps})$
a_t	Error aleatorio.

Este modelo se puede representar con la siguiente notación, $(\boldsymbol{p},\boldsymbol{d},\boldsymbol{q}) \times (\boldsymbol{P},\boldsymbol{D},\boldsymbol{Q})_s$, donde $\boldsymbol{p},\boldsymbol{d}$ y $\boldsymbol{q}$ representan los órdenes del término autorregresivo, de la diferencia y del término media móvil estacionales respectivamente, y $\boldsymbol{P},\boldsymbol{D}$ y $\boldsymbol{Q}$, los mismos órdenes pero estacionales. Por último, s representa la longitud de la estacionalidad.

La identificación y validación del modelo, se obtienen mediante el cálculo de las funciones de autocorrelación y autocorrelación parcial de la serie original y los residuos, respectivamente.

Para la estimación de los parámetros del modelo propuesto, se utiliza el algoritmo de Gauss- Marquardt, que es un método de optimización no lineal con restricciones y combina los algoritmos iterativos de Gauss-Newton y el de máxima pendiente.

El primer ejemplo que vamos a considerar para el modelado univariante, es la serie temporal que llamamos UN05.DAT, obtenida de la base de datos de Reilly, D.P. (1980), y cuyo gráfico representamos a continuación.

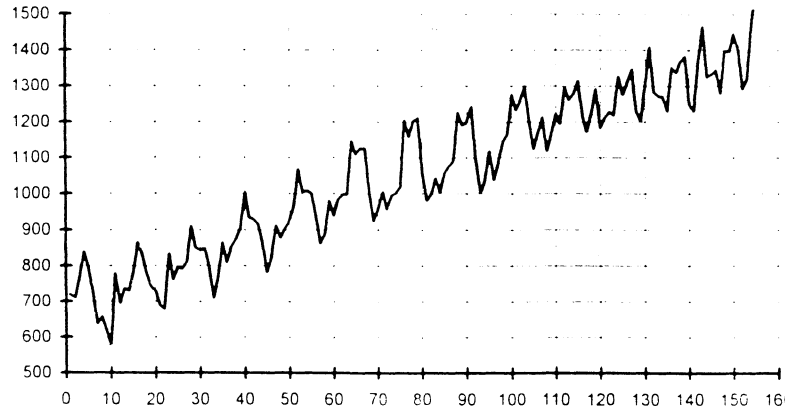

Figura 1.
Serie UN05.DAT.

Esta serie, consta de 155 valores, de los cuales hemos utilizado los primeros 134 como datos para al cálculo de las previsiones mediante la técnica de Box-Jenkins, el resto (21 valores), servirán para comprobar la bondad del método. Utilizando, como ya se ha comentado, el paquete STATGRAPHICS, se obtuvo el modelo:

$$(1 - 0.6B)(1 - B^{12})Z = \text{CONS} + \text{NOISE}$$

Los resultados que se ven en la figura 2, son la representación gráfica de las previsiones calculadas, comparadas con los valores reales de la serie UN05.

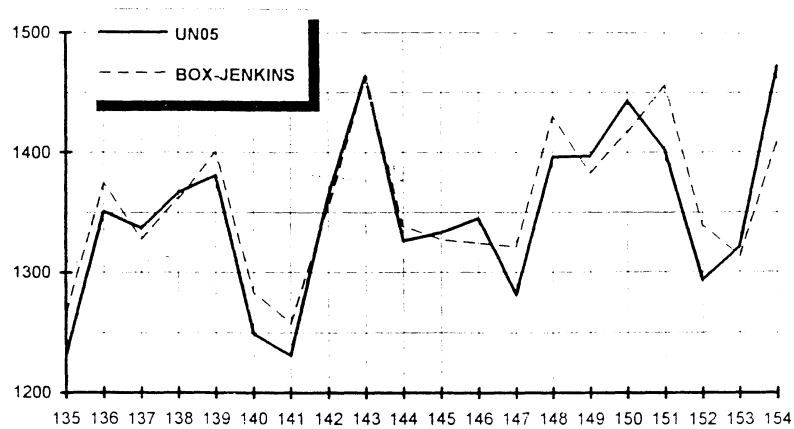

Figura 2.
Valores reales/Previsiones Box-Jenkins.

2.2. Modelado de Funciones de Transferencia

Previo a este tipo de modelado, habría que considerar el llamado análisis de intervención que correspondería, a efectos puntuales en la evolución de la serie temporal, o a efectos que se producen a partir de un determinado instante en el desarrollo de la serie (un ejemplo del primer caso, podría ser un día de fiesta en el cálculo del consumo de electricidad, y un ejemplo del segundo podría ser la entrada en vigor de una ley a partir de una determinada fecha). No obstante, al ser casos particulares del modelado multivariante, no entraremos en ellos.

El modelado de funciones de transferencia, pretende encontrar un modelo estocástico que relacione varias variables explicativas $X_1, X_2, \ldots, X_n$ (input) y una variable explicada (output) Y (Figura 3). El procedimiento de cálculo del modelo, implica los mismos pasos que en el modelado univariante: identificación tentativa, estimación de parámetros eficientemente y verificación del mismo. Resultando en este caso, una regresión dinámica pero ahora con regresores las series $X_1, X_2, \ldots, X_n$ y el ruido del sistema.

Figura 3.
Modelado de Función de Transferencia (Multiple Input Single Output).

El modelo de función de transferencia, se supone que es lineal, causal y discreto. Además, las entradas $X_1, X_2, \ldots, X_n$ son independientes y no existe realimentación entre ningún par de variables a relacionar. Por ello, el modelo propuesto tiene la siguiente expresión:

$$Y_t = \sum_{i=1}^{n} \frac{\omega_i(B)}{\delta_i(B)} \cdot X_{it} + \frac{\theta(B)}{\phi(B)} \cdot a_t$$

La herramienta básica para obtener el modelo adecuado, es la función de correlación cruzada; vamos a describir los pasos necesarios para el cálculo de una función de transferencia cuando tenemos una sola entrada, aunque es fácilmente generalizable y se encuentra en la literatura al uso.

Los pasos a realizar en la especificación, estimación y contraste de un modelo de función de transferencia son (Figura 4):

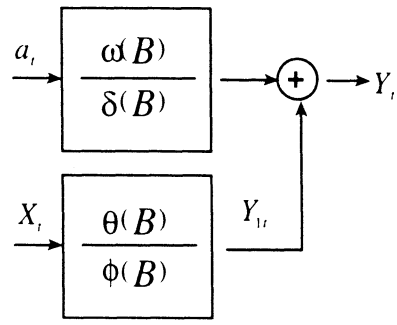

Figura 4.
Cálculo de la Función de Transferencia.

1) Se hace un análisis univariante de la serie $X(t)$.

2) Preblanqueado de la serie $Y(t)$ (esto es filtrar la serie $Y(t)$ en el modelo univariante de $X(t)$), y estimación inicial de los coeficientes de la respuesta $v(k)$ (o sea, cálculo de la función de correlación cruzada entre los residuos de la serie $X(t)$ y los obtenidos al filtrar la serie $Y(t)$).

3) Determinación de los órdenes de los polinomios de la función de transferencia, $w(B)$ y $\delta(B)$, a partir de la relación:

$$(1 - \delta_1 B^1 - \cdots - \delta_r B^r) \cdot (v_0 + v_1 B + \cdots) = (w_0 - w_1 B - \cdots - w_s B^s)$$

que nos da una orientación sobre los posibles valores de r y s. Un análisis detallado de los diferentes valores de r y s, nos permitirá obtener los más adecuados.

4) Determinación de los órdenes de los polinomios del término de error, $\phi(B)$ y $\theta(B)$, bien a través del análisis univariante de $Y(t)$ o bien estudiando el comportamiento de los residuos.

5) Etapa de estimación de los coeficientes del modelo: Mediante el procedimiento de mínimos cuadrados no lineales.

6) Etapa de validación, que consiste en comprobar que los residuos que se obtienen al relacionar $X(t)$ con $Y(t)$, son ruido blanco, si no ocurre esto, se debe de dar estructura al modelo de ruido que será, generalmente, muy semejante al modelo univariante de $Y(t)$. En esta última etapa de validación, se deben de utilizar, entre otros, los contrastes estadísticos **t** para la significación de los parámetros, matriz de correlación de los parámetros y contraste de ruido blanco y ausencia de correlación cruzada entre $X(t)$ y $a(t)$.

Como el objetivo del trabajo es comparar diferentes metodologías, vamos a utilizar un ejemplo muy conocido para el cálculo de la función de transferencia. A continuación, se muestran los resultados de la aplicación del método descrito, para el modelado de la función de transferencia en un ejemplo. En él, se trata de relacionar la concentración de CO_2 resultante, $Y(t)$, como salida de un horno de gas, en función de la tasa de gas que lo alimenta, $X(t)$. Las dos series $X(t)$ e $Y(t)$, constan de 296 datos, y aparecen representadas en la figura 5.

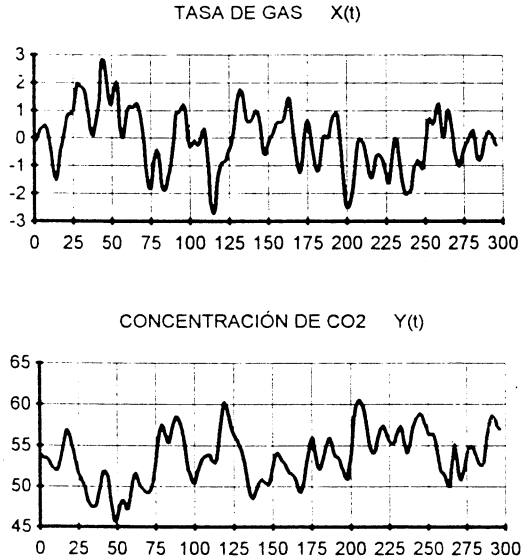

Figura 5.
Series Temporales $X(t)$ e $Y(t)$.

De la misma manera que se ha hecho en el caso univariante, utilizaremos los 250 primeros valores de las dos series temporales, $X(t)$ e $Y(t)$, para deducir el modelo y calcularemos 40 previsiones que comparadas con los valores reales nos darán una idea de la bondad del método. Utilizando el paquete comercial BMDP (programa P2T), se puede deducir el modelo:

$$(1-1.53B+0.63B^2)Y_t = 53.4 + \frac{-0.53B^3 - 0.38B^4 - 0.52B^5}{1-0.55B}X_t + a_t$$

Los resultados de las previsiones obtenidas, junto con los valores reales de la serie $Y(t)$, se pueden observar en la figura 6.

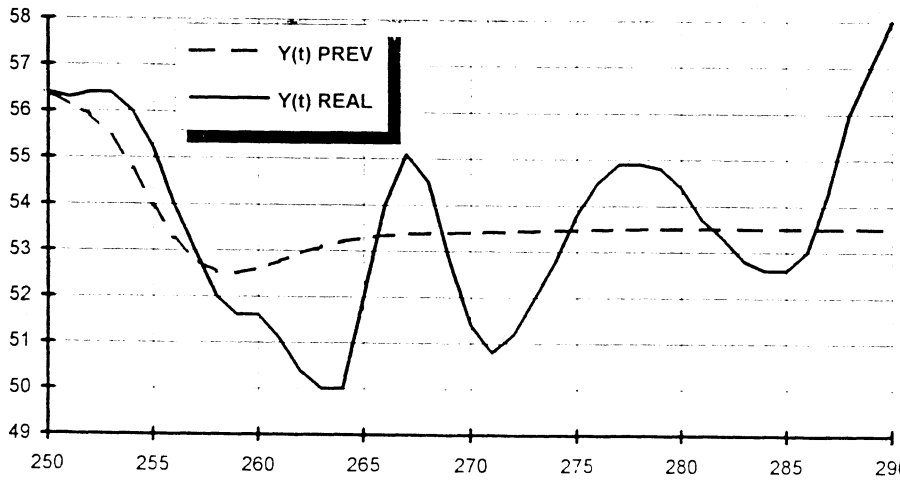

Figura 6.
Previsiones/Valores reales de $Y(t)$.

3. REDES NEURONALES

Las redes neuronales (o redes de neuronas artificiales), son modelos matemáticos simplificados de las redes de neuronas que constituyen el cerebro humano. Estos modelos, están compuestos por un conjunto de "neuronas artificiales" o conjunto de unidades que procesan e intercambian información. Las neuronas de una red, están estructuradas en distintas capas, de forma que una neurona de una capa está conectada con las de la capa siguiente, a las que puede enviar información.

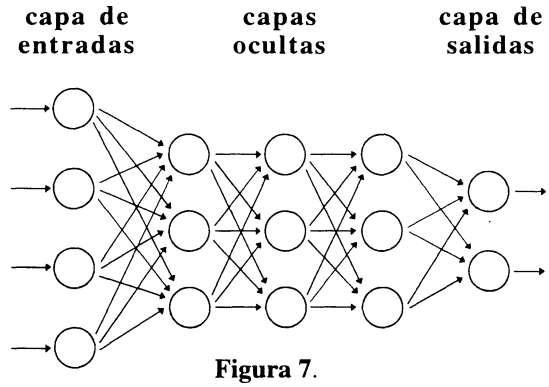

Figura 7.
Arquitectura de una Red Neuronal Artificial.

Tal como se refleja en la figura 7, la arquitectura más habitual consiste en una capa de neuronas de entrada que recibe la información "del exterior", una serie de capas intermedias (u "ocultas") y una capa de salidas, que proporciona "al exterior" el resultado del trabajo de la red.

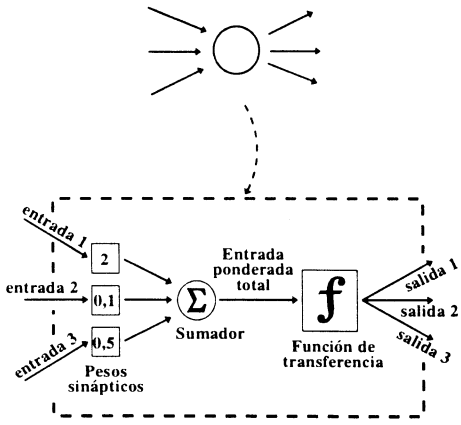

Figura 8.a.

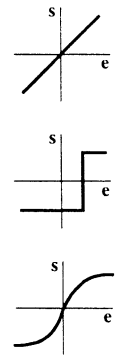

Figura 8.b.

Figura 8.
Funcionamiento de una neurona y tipos de función de transferencia.

Cada neurona, tal como se muestra en la figura 8.a, constituye una "unidad de procesamiento" de información, convierte un conjunto de señales de entrada en una salida que es difundida a las neuronas de la capa siguiente. Esta conversión se realiza en dos etapas: primero, cada una de las señales de entrada es multiplicada por un coeficiente de ponderación ("peso sináptico") atribuido a la conexión; todos los productos son sumados para obtener una cantidad denominada "entrada ponderada total". En una segunda fase, cada unidad utiliza una función de transferencia entrada-salida, o función de activación, que transforma la entrada ponderada total en una señal de salida que es la que se difunde a las neuronas de la capa siguiente. La función de transferencia puede ser de tres tipos, (Lippmann, 1987):

1. ***Lineal***. La actividad de salida es proporcional a la entrada ponderada total.

2. ***De umbral***. La salida queda fija a uno de dos niveles, dependiendo si la entrada ponderada total es mayor o menor que cierto valor crítico denominado "umbral".

3. ***Sigmoide***. La salida varía de forma continua dependiendo de la entrada ponderada total, pero esta dependencia no es lineal.

Habitualmente, se suele utilizar la sigmoide como función de transferencia cuando se trata de aplicar la tecnología de redes neuronales al procesado de señales no-lineales (Lapedes y Farber, 1987), aunque es necesario tener presente que las tres son aproximaciones bastante burdas de la actividad de las neuronas reales.

El proceso de aplicación de la tecnología de redes neuronales artificiales a un problema concreto, consta de tres etapas fundamentales:

1. **Diseño de la Red**. Es necesario decidir la arquitectura que va a tener la red , lo cual implica determinar el número de neuronas de la capa de entradas, el número de capas ocultas y las neuronas que contendrá cada una de ellas, y, por último, el número de neuronas de la capa de salidas. La arquitectura de la red dependerá, como es lógico, del problema concreto que se quiera resolver.

   En el caso que nos ocupa, queremos aplicar la tecnología de redes neuronales artificiales a la previsión de series temporales, basándonos en los datos históricos de la serie, por lo tanto, el número de neuronas de la capa de entrada coincidirá con el número de datos anteriores de la serie que son necesarios para calcular un valor concreto. La capa de salida estará compuesta por una sola neurona cuya salida será el valor de la previsión que queremos calcular. En cuanto al número de capas ocultas y las neuronas de estas capas, no existen reglas fijas que determinen los valores óptimos. Un número muy pequeño de capas ocultas puede hacer que el proceso de entrenamiento se alargue excesivamente, mientras que demasiadas capas ocultas llevan a que se produzca una memorización de los datos, en lugar de la deducción de los patrones que se derivan de los datos, con lo que se falsea la predicción (Richeson y Zimmermann, 1994).

   Por otra parte, el número de neuronas en cada capa oculta, dependerá de la complejidad del problema a estudiar, aunque algunos autores dan ciertas recomendaciones que van desde: la utilización de complicadísimas fórmulas para su cálculo (Zurada, 1991), o indicar que el número de neuronas de la capa oculta debe ser como mínimo el 75% del número de neuronas de entrada (Salchenberger, 1992), hasta aplicar la teoría matemática clásica de Kolmogorov para calcular el número de neuronas como $2k+1$, siendo "k" el número de neuronas de la capa de entrada (Zaremba, 1990).

   Hemos comprobado que el número de neuronas de la capa oculta, siempre que esté entre unos valores mínimo (por ejemplo el 75% de las neuronas de entrada) y máximo (5 veces ese mismo número), sólo influye en la velocidad de entrenamiento de la red, por lo que en nuestro caso hemos partido de un valor inicial ($2k+1$), que posteriormente se ha ajustado, si durante el proceso de entrenamiento de la red se comprueba que mejora los resultados.

2. **Entrenamiento de la Red Neuronal Artificial**. El entrenamiento de una Red Neuronal consiste en la utilización de un algoritmo (generalmente se usa el al-

goritmo "Back Propagation") para ajustar los pesos sinápticos de las conexiones entre las neuronas.

El proceso consiste en presentar a la red inicial una batería de casos de entrenamiento, que se construyen utilizando los datos reales disponibles (el pasado de la serie temporal), tal como se comenta en el apartado siguiente. Cada uno de estos casos está compuesto por una serie de valores de entrada y el valor de salida correspondiente. Se asignan los valores de entrada a las neuronas de la capa de entradas, y se obtiene al final un valor de salida de la red neuronal. Esta respuesta se compara con la deseada u objetivo, mediante una función de error que da una medida de la eficacia de la configuración actual de pesos sinápticos de la red. El objetivo del aprendizaje es minimizar esta función de error.

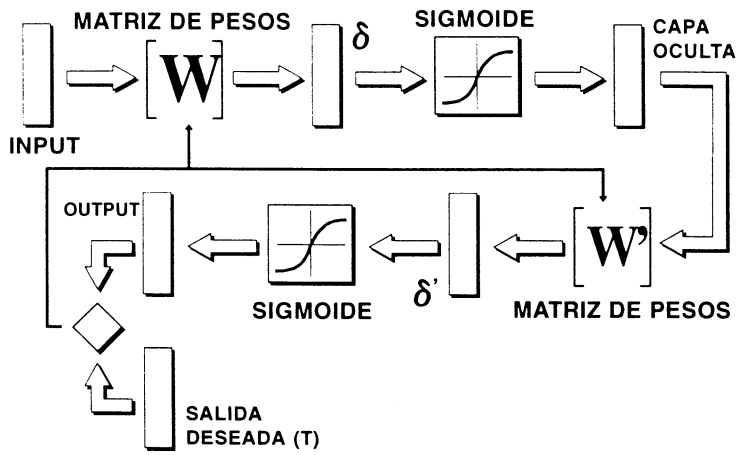

Figura 9.
Algoritmo "Back-Propagation".

Una vez calculado este error, se procede a realizar la fase "hacia atrás" ("backward") variando los pesos sinápticos de los enlaces en función de la magnitud del error cometido y de una constante ε llamada "coeficiente, tasa o ritmo de aprendizaje" que varía entre 0 y 1. La elección de ε es importante, ya que un valor bajo, da como resultado una lenta convergencia pues implicará variar los pesos muy poco, mientras que un valor excesivamente alto, puede conducir a oscilaciones en los pesos de la red, con lo que nunca se alcanzarían los pesos óptimos. Un valor usual de partida para ε es 0,3 aunque puede ser ajustado ligeramente durante el proceso de aprendizaje de la red.

Una vez ajustados los pesos sinápticos, se vuelven a presentar los casos a la red y se vuelve a calcular el error para de acuerdo con él, reajustar los pesos.

Este proceso continuará hasta que el error obtenido esté dentro de unos límites previamente fijados como aceptables, en este momento, el entrenamiento de la red habría finalizado. Una forma sencilla de acelerar el entrenamiento de la red, es añadir un término de momento "***m***" a la hora de ajustar los pesos, que recoja información sobre el último ajuste realizado (Chapman, 1994). Un valor habitual de partida suele ser $m = 0,7$, aunque también puede ser ajustado a cualquier valor entre 0 y 1 durante el proceso de entrenamiento de la red.

3. **Utilización de la Red Neuronal en "Modo Recuerdo"**. Una vez entrenada, la Red Neuronal está en condiciones de ser utilizada. Para ello, no hay más que presentar a la red un caso determinado (por ejemplo los últimos datos disponibles de una serie temporal) para que, utilizando los pesos sinápticos encontrados durante el proceso de entrenamiento, calcule la salida (la previsión del dato siguiente de la serie temporal).

La mayor parte de las aplicaciones de redes neuronales, se realizaron en lenguajes de programación convencionales como FORTRAN ó C. Sin embargo, de forma análoga a lo sucedido con los sistemas expertos, ante la expansión de la tecnología y las dificultades que presentan algunas de la etapas de diseño y entrenamiento de las redes, diversas instituciones y compañías, han empezado a comercializar ***entornos de desarrollo*** o "shells" que facilitan diversos tipos de arquitecturas y algoritmos de entrenamiento. Nosotros, hemos utilizado el paquete comercial denominado "NEUROSHELL" en aplicaciones sobre previsión, aunque existen otros como "ANSIM", "BACK PROPAGATION", etc, para ordenadores personales.

3.1. Modelado Univariante

En este primer caso, hemos tomado la serie UN05 (ya utilizada en este mismo trabajo) que consta de 155 datos, y hemos calculado previsiones utilizando la tecnología de redes neuronales. En el apartado siguiente de este artículo, se comparan estas previsiones con las obtenidas utilizando la metodología de Box-Jenkins.

Siguiendo los pasos indicados anteriormente, la primera fase consiste en el diseño de la red neuronal, es decir, número de neuronas de entrada, número de capas ocultas y número de neuronas en cada una de ellas, así como el número de neuronas de la capa de salidas. Como se ha comentado anteriormente, el número de capas ocultas influye en la velocidad del proceso de entrenamiento, en nuestro caso, teniendo en cuenta el problema que intentamos resolver, es suficiente con una sola capa oculta, ya que las pruebas que se han hecho con dos o tres no mejoraban significativamente los resultados obtenidos con una pero sí aumentaba de forma considerable el tiempo de entrenamiento. Por lo tanto, trabajaremos con redes neuronales del tipo "***d*−*n*−*s***",

llamando "d" al número de neuronas de la capa de entrada, "n" número de neuronas de la capa oculta, y "s" el número de neuronas de la capa de salida.

El valor de "s" es evidente, sólo queremos pronosticar un valor de la serie temporal utilizando para ello una serie de valores anteriores de la misma; por ello sólo habrá una neurona en la capa de salidas, por lo tanto, será una red del tipo "$\boldsymbol{d-n-1}$". El valor de "d", neuronas de la capa de entrada, dependerá del número de valores anteriores de la serie temporal que son necesarios para que la red deduzca un patrón o un modelo, de forma que pueda calcular el valor de salida correspondiente. El cálculo de "d" se realiza mediante pruebas sucesivas con varios diseños y escogiendo aquel con el que se obtienen los mejores resultados.

Se puede empezar con una red $5-11-1$. Cinco neuronas de entrada implica que cada valor de la serie temporal dependerá directamente de los cinco valores anteriores, ponemos 11 neuronas (resultado de aplicar la fórmula $2d+1$) en la capa oculta como número inicial, y una de salida. Esta arquitectura se muestra en la figura 10.

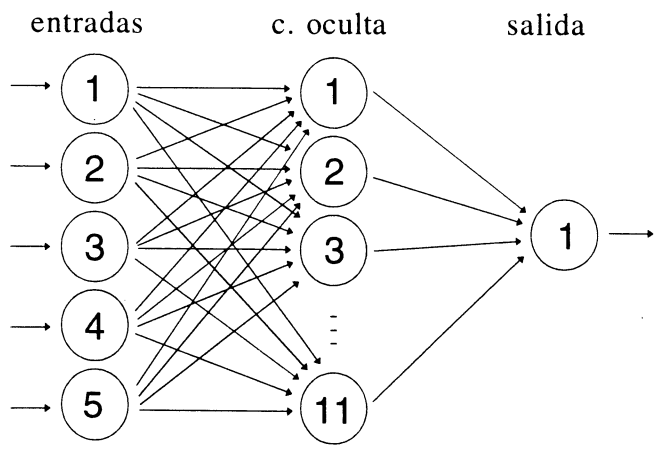

Figura 10.
Arquitectura de la red $5-11-1$.

Para el proceso de entrenamiento de esta red, disponemos de los 134 valores históricos de la serie temporal UN05. En algunas ocasiones, es útil hacer un procesado previo de los datos de entrada antes de presentarlos a la red para el entrenamiento, por ejemplo, para series que presentan grandes variaciones e irregularidades aleatorias, es conveniente trabajar con la serie en logaritmos de los datos de entrada, o efectuar un alisado exponencial en el caso de variaciones aleatorias menores (Kimoto *et al*, 1990). Además, es también útil normalizar los datos de entrada antes de presentarlos

a la red neuronal, esta normalización se utiliza para reducir el rango del conjunto de valores y que sean los apropiados para la función de transferencia o activación que se está usando. Generalmente se normalizan los datos entre 0 y 1 para prevenir el efecto de la saturación de la función de transferencia (sigmoide).

Teniendo en cuenta este diseño de la red, se construye el conjunto de casos de entrenamiento; cada uno de ellos se compone de cinco valores de entrada que determinan un valor de salida. Si llamamos $X1, X2, \ldots, X134$ a los valores históricos (ya normalizados entre 0 y 1) de la serie temporal, se podrían construir 129 casos:

$$
\begin{array}{ccccccc}
X1, & X2, & X3, & X4, & X5 & \longrightarrow & X6 \\
X2, & X3, & X4, & X5, & X6 & \longrightarrow & X7 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
X129, & X130, & X131, & X132, & X133 & \longrightarrow & X134
\end{array}
$$

Con estos 129 casos, comienza el proceso de entrenamiento de la red. Mediante las sucesivas pasadas de los casos por la red, y los ajustes que se van haciendo a los correspondientes pesos sinápticos de los enlaces entre las neuronas, se intenta reducir la diferencia entre la salida real y la salida que genera la red. El proceso termina cuando los errores entre la salidas real y generada por la red entran dentro de un intervalo previamente fijado como aceptable, o bien cuando, transcurrido un período de tiempo considerable, no se observan disminuciones en los errores, que quedan estabilizados en valores fuera del intervalo previamente fijado como aceptable.

Cuando el proceso de entrenamiento ha terminado, la red está preparada para ser utilizada en modo recuerdo, los enlaces entre las neuronas de las distintas capas tendrán unos pesos sinápticos que representan el modelo o patrones de la serie temporal, de forma que se puede pronosticar un valor de salida introduciendo los valores de entrada correspondientes. En nuestro caso, hemos introducido de nuevo los 129 casos para que la red calcule las 129 salidas que componen la serie UN05PR, que podemos comparar con la serie original UN05. Utilizamos para ello el Error Cuadrático Medio (MSE) calculado según la expresión:

$$MSE = \frac{\sum_{i=1}^{n} (X_i - S_i)^2}{n}$$

donde X_i son los valores de la serie UN05, S_i los datos pronosticados que pertenecen a la serie UN05PR, y n el total de casos de entrenamiento (en esta primera red, 129).

Ya hemos comentado que la obtención de la arquitectura óptima de la red en cuanto a número de neuronas de entrada, en la capa oculta y en la salida, en un principio sólo es posible mediante un proceso iterativo, entrenando sucesivamente una red tras otra. En nuestro caso hemos comenzado por la red 5-11-1, la red ha sido entrenada utilizando los 129 casos que se pueden construir con los valores históricos

de la serie, y, cuando el entrenamiento ha finalizado (en este caso, se comprobó que después de más de 1.000.000 de pasadas de los casos de entrenamiento por la red, los pesos sinápticos de los enlaces apenas variaban, de forma que la función de error que se intenta minimizar, quedaba estabilizada en un valor determinado), se introdujeron a la red ya entrenada, los mismos 129 casos para que la red dedujera la salida correspondiente para cada uno de ellos, y se calculó el error cuadrático medio (MSE), para comprobar la bondad de la red 5-11-1.

El mismo proceso se ha seguido para entrenar una segunda red, con la arquitectura 6-13-1 (cada valor de la serie UN05 depende de los seis valores anteriores), 7-15-1, 8-17-1, etc. (el número de neuronas de la capa oculta se ha calculado siempre utilizando la expresión $2d + 1$, ya comentada), hasta la red 15-31-1, que ha sido la última que se ha entrenado. Todas ellas han utilizado un mínimo de 1.000.000 de pasadas con el programa "NeuroShell", hasta que se llegaba a un instante en el que la función de error se estabilizaba. En ese momento, se considera la red como suficientemente entrenada y apta para realizar previsiones. Después de calcular la serie UN05PR que pronosticaba cada red, se calculó el error cuadrático medio (MSE), obteniéndose los valores que se recogen en la tabla 1.

Tabla 1

Error Cuadrático Medio

Número de neuronas de entrada	**Arquitectura de la red entrenada**	**Número de casos de entrenamiento**	**Error Cuadrático Medio MSE**
5	5-11-1	129	0.00525371
6	6-13-1	128	0.00321710
7	7-15-1	127	0.01257860
8	8-17-1	126	0.00430616
9	9-19-1	125	0.00519821
10	10-21-1	124	0.00578450
11	11-23-1	123	0.01380814
12	**12-25-1**	**122**	**0.00047366**
13	13-27-1	121	0.00101259
14	14-29-1	120	0.00191643
15	15-31-1	119	0.00364024

Tabla 2

Pesos sinápticos finales de la Red 12-25-1

		NEURONAS DE LA CAPA DE ENTRADAS												SALIDA
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**
	1	0.10	−0.18	0.25	0.14	0.67	0.55	0.55	0.32	0.15	−0.52	−0.42	0.78	0.54
	2	0.39	−0.25	−0.01	0.10	0.59	0.71	0.60	0.79	0.26	−0.16	−0.03	0.44	0.54
	3	0.20	−0.10	−0.10	0.20	0.72	0.31	0.27	0.00	−0.08	−0.64	0.24	0.83	0.42
N	**4**	0.04	−0.55	−0.25	0.32	0.68	0.34	0.25	−0.22	0.16	−0.31	0.34	1.13	0.20
E	**5**	0.46	0.33	0.23	0.34	0.50	0.18	0.40	−0.11	0.53	0.58	0.48	0.32	−0.62
U	**6**	−0.23	−0.50	−0.86	0.41	0.60	0.81	−0.36	−0.51	0.07	−0.27	−0.25	−1.44	0.01
R	**7**	0.46	0.32	0.20	0.20	0.45	−0.01	0.55	0.31	0.24	0.27	0.64	0.07	−0.18
O	**8**	1.57	2.30	2.10	−1.18	−2.15	−1.71	−1.12	0.24	0.24	1.73	1.69	3.15	1.85
N	**9**	0.54	1.99	0.78	−0.74	−1.49	−0.64	−0.51	0.92	−0.14	0.32	0.08	−3.93	−2.33
A	**10**	0.12	0.81	0.67	−0.02	−0.13	−0.07	0.22	0.06	0.33	0.45	0.66	−1.44	−0.65
S	**11**	0.29	0.61	0.40	0.03	−0.64	−0.14	0.07	0.62	0.33	0.87	0.67	−1.47	−0.58
	12	−0.32	−0.29	−0.18	−0.06	0.82	0.63	0.76	0.12	−0.57	−0.96	−0.07	1.18	0.91
C	**13**	−0.06	−0.96	−0.26	0.25	0.57	0.40	0.63	−0.36	−0.53	−0.50	−0.14	1.84	0.60
A	**14**	−0.30	−0.50	−0.93	0.30	0.75	0.46	−0.18	−0.62	0.11	0.05	−0.28	−1.27	−0.14
P	**15**	0.48	0.68	0.19	0.46	−0.02	0.20	0.00	0.30	0.41	0.68	0.53	−0.88	−0.55
A	**16**	0.38	0.44	0.31	0.39	0.27	0.46	0.39	0.61	0.06	0.27	0.41	−0.34	−0.02
	17	0.24	0.31	0.31	0.50	0.13	0.30	0.22	0.55	0.59	0.14	0.13	0.42	0.02
O	**18**	0.35	−0.26	0.23	0.45	0.26	0.07	0.34	0.09	−0.27	−0.36	0.35	0.78	0.31
C	**19**	−0.32	−0.49	−0.37	0.22	0.73	0.99	0.94	0.63	−0.51	−0.88	−0.52	0.82	1.26
U	**20**	−0.08	0.49	−0.14	0.21	0.49	0.61	0.85	0.95	0.09	−0.29	−0.10	0.31	0.84
L	**21**	0.32	0.30	0.42	0.01	0.04	0.39	0.39	0.32	0.37	0.63	0.52	−0.64	−0.29
T	**22**	0.33	0.06	0.31	0.20	0.49	0.32	0.24	0.03	0.14	−0.21	0.40	0.53	0.07
A	**23**	0.26	0.22	0.62	0.68	0.55	0.19	0.15	0.15	0.21	1.01	0.36	−0.16	−0.80
	24	0.36	0.06	0.45	0.48	0.33	0.55	0.56	0.51	0.35	0.29	0.39	−0.14	−0.04
	25	0.54	0.36	0.10	0.42	0.28	0.24	0.09	0.32	0.16	1.04	0.40	−0.32	−0.65

Tal como se puede comprobar en la tabla, el error cuadrático medio más bajo se obtiene para la red 12-25-1 (cada valor de la serie depende directamente de los doce valores anteriores). Este resultado parece lógico ya que la serie UN05 tiene una estacionalidad de orden doce, tal como se comprobó al hacer el estudio de la serie para calcular el modelo según Box-Jenkins. Una vez que se ha decidido que el valor óptimo de neuronas de entrada es 12, se puede intentar afinar más, variando ligeramente la tasa de aprendizaje, ε, y/o el momento, m. Se puede incluso intentar aumentar o disminuir ligeramente el número de neuronas de la capa oculta para comprobar si se reducen aún más los errores o se produce una convergencia más rápida de los pesos sinápticos de los enlaces. En este caso concreto se ha comprobado que se produce una ligerísima mejoría si se ajusta como tasa de aprendizaje el valor $\varepsilon = 0.3$ y el momento a $m = 0.2$, mientras que no se observa mejoría apreciable alguna variando el número de neuronas de la capa oculta. Los valores de los pesos sinápticos definitivos después de todo el proceso de entrenamiento de la red neuronal 12-25-1, son los que aparecen recogidos en la tabla 2, en la que se puede ver el peso sináptico del enlace entre cada una de las 25 neuronas de la capa oculta y cada una de las 12 neuronas de la capa de entradas, y la única neurona de la capa de salidas.

En la figura 11, se puede comprobar el ajuste de la serie UN05 deducida por la red neuronal 12-25-1, con la serie UN05 real.

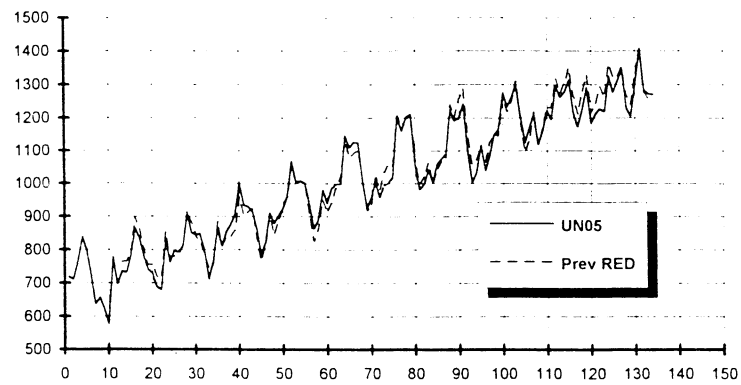

Figura 11.
Serie calculada por la Red 12-25-1 y serie UN05 real.

Una vez finalizado el proceso de entrenamiento de la red neuronal, se pasa al cálculo de previsiones, utilizando la red en "modo recuerdo", es decir, se introducen los nuevos casos a la red con los pesos sinápticos ya definitivos que se han ajustado durante el proceso de entrenamiento de la misma. En concreto, se introducen 20 casos para que la red calcule las previsiones $S135, S136, \ldots, S154$, y que representamos de la siguiente forma:

$X123\ X124\ X125\ X126\ X127\ X128\ X129\ X130\ X131\ X132\ X133\ X134$ ¿S135?
$X124\ X125\ X126\ X127\ X128\ X129\ X130\ X131\ X132\ X133\ X134\ X135$ ¿S136?
$X125\ X126\ X127\ X128\ X129\ X130\ X131\ X132\ X133\ X134\ X135\ X136$ ¿S137?

$X142\ X143\ X144\ X145\ X146\ X147\ X148\ X149\ X150\ X151\ X152\ X153$ ¿S154?

De esta manera de obtienen los 20 valores que aparecen en la figura 12 junto con los valores reales de la serie UN05.

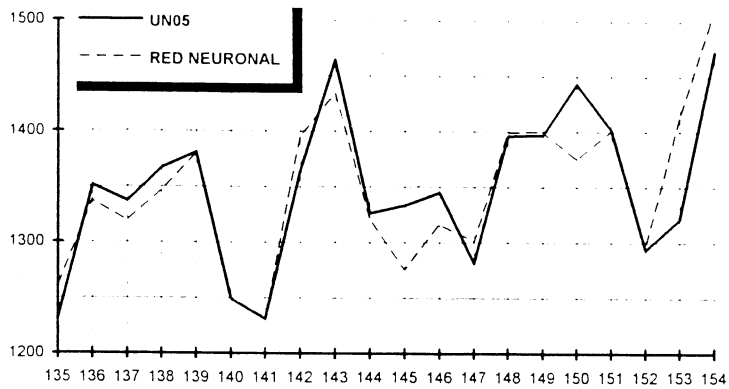

Figura 12.
Valores reales/Previsiones RED NEURONAL.

3.2. Modelado de Funciones de Transferencia

Nuestro objetivo es pronosticar los valores de una serie $Y(t)$ que dependen directamente de los de otra serie $X(t)$. Como vimos en el apartado 2.2, utilizamos el ejemplo del horno, en el cual tenemos como datos 296 valores históricos de las series $X(t)$ e $Y(t)$, y que están representados en la figura 5, de los cuales los primeros 250, nos servirán para entrenar a la red neuronal.

Al final del proceso, el modelo de red neuronal, hará las veces de función de transferencia entre $X(t)$ e $Y(t)$, de forma que para cada valor o serie de valores de $X(t)$, se tendrá un valor de $Y(t)$. Utilizaremos los últimos 46 datos de Y, para comprobar la bondad de las previsiones que se calcularon mediante la red neuronal, y las obtenidas utilizando la metodología de Box-Jenkins.

La red neuronal que se va a construir será de tipo $n - d - 1$, es decir, será una red con "n" nodos de entrada, "d" nodos en la capa oculta y 1 nodo de salida. Los valores óptimos de n y d, se obtienen haciendo pruebas de la misma forma que en el caso univariante. El objetivo, es reducir el valor del error cuadrático medio (MSE).

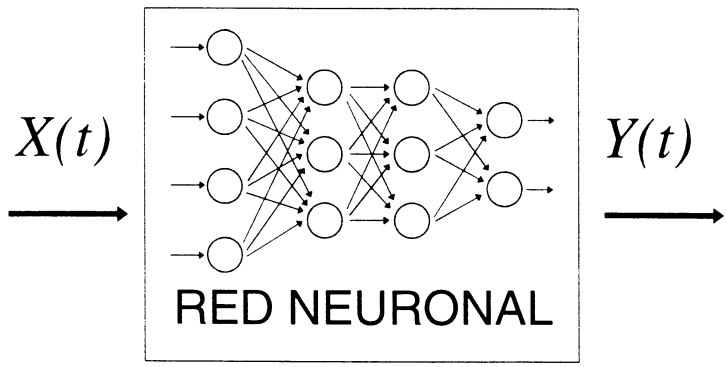

Figura 13.
Red Neuronal como función de transferencia.

La primera red que vamos a entrenar, es la que llamaremos RED 30, que indica que en cada caso utilizado para el entrenamiento, se utilizan 3 valores de la serie X, el del instante t, el de $t-1$ y el de $t-2$ ($X_t, X_{t\ 1}$ y $X_{t\ 2}$), para pronosticar el valor de Y retrasado cero períodos (Y_t). En este primer caso se dispone de un conjunto de 248 casos de entrenamiento:

$$
\begin{array}{llll}
X1, & X2, & X3 & \longrightarrow Y3 \\
X2, & X3, & X4 & \longrightarrow Y4 \\
\multicolumn{4}{c}{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots} \\
\multicolumn{4}{c}{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots} \\
X248, & X249, & X250 & \longrightarrow Y250
\end{array}
$$

Se seguirá un proceso similar al llevado a cabo en el modelado univariante para intentar llegar a la arquitectura óptima de la red neuronal, con una diferencia, además de ir aumentando el número de neuronas de la capa de entradas, se tendrá en cuenta también la posibilidad de que la respuesta pueda estar retrasada algún período respecto a la entrada. Según esto, probaremos con redes de 3, 4 o 5 neuronas en la capa de entrada, y para cada una de ellas, consideraremos que la salida puede estar retrasada

desde 0 a 4 períodos. Los resultados del proceso de entrenamiento de las diferentes redes neuronales aparecen recogidos en la tabla 3.

Tabla 3

Error Cuadrático Medio para diferentes configuraciones de la red neuronal.

		NÚMERO DE NEURONAS DE LA CAPA DE ENTRADAS		
		3	**4**	**5**
NÚMERO DE PERÍODOS DE RETRASO	**0**	0.02285230	0.01333500	0.00728100
	1	0.01524960	0.00556036	0.00234480
	2	0.00298400	0.00312900	0.00076860
	3	0.00324960	0.00117510	0.00074710
	4	0.00203520	0.00220270	0.00431380

En dicha tabla se observa que el valor mínimo del error cuadrático medio se obtiene para la RED53 (se trata de una red con 5 neuronas en la capa de entrada y en la que la salida se retrasa 3 períodos), el diseño de la red aparece en la figura 14. Una vez escogida la arquitectura idonea, se fue variando el número de neuronas de la capa oculta desde las 11 que se consideraron inicialmente (resultado de aplicar la expresión $2d+1$), hasta el 14, número de neuronas con el que se observó un menor valor del error, después de un tiempo considerable de entrenamiento. Los valores de los pesos sinápticos finales entre los enlaces de la red53, aparecen recogidos en la tabla 4.

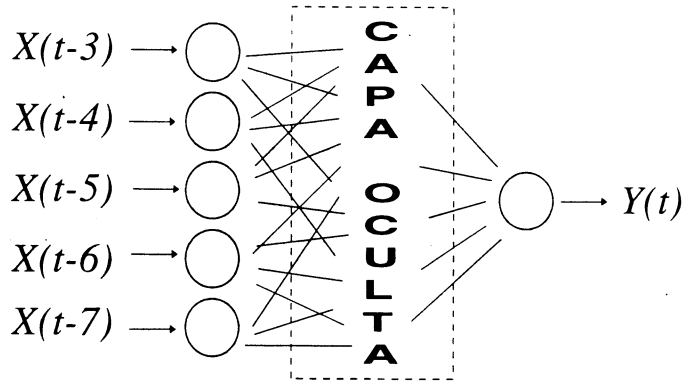

Figura 14.
Arquitectura de la RED53.

Tabla 4

Pesos sinápticos finales de la red 5-14-1.

		CAPA DE ENTRADAS					SALIDA
		1	**2**	**3**	**4**	**5**	**1**
		$X(t-3)$	$X(t-4)$	$X(t-5)$	$X(t-6)$	$X(t-7)$	$Y(t)$
	1	0.46	0.69	0.60	1.03	−0.46	−0.07
C	**2**	−6.13	−1.24	−0.12	0.46	1.34	2.53
A	**3**	0.57	1.10	0.24	1.37	−0.40	0.37
P	**4**	1.75	1.84	3.02	−0.27	0.16	−1.87
A	**5**	1.18	−0.73	0.51	−0.98	4.68	−1.39
	6	0.75	0.77	0.80	0.68	1.67	0.13
	7	0.78	0.55	0.29	0.89	−0.52	−0.05
O	**8**	0.64	0.55	0.90	0.95	0.90	0.18
C	**9**	0.41	0.60	0.69	1.29	0.10	0.32
U	**10**	0.85	1.19	0.31	0.87	−0.14	0.54
L	**11**	0.64	0.60	1.02	1.03	0.85	0.22
T	**12**	2.43	−2.12	−4.92	0.18	−0.86	2.69
A	**13**	−0.74	−0.95	−1.18	−0.17	−0.39	0.31
	14	−2.65	−2.05	−1.58	0.02	−0.17	1.27

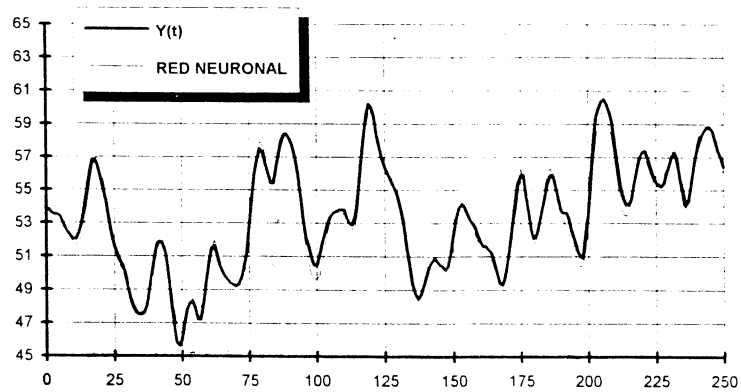

Figura 15.
Serie $Y(t)$ real y serie deducida por la red53.

En la figura 15, se puede comprobar el ajuste conseguido por la red neuronal con respecto a la serie $Y(t)$ original. Si ahora se calculan las previsiones introduciendo a la red53 entrenada, los 20 casos que proporcionan como salida las previsiones de $Y251$ a $Y290$:

$$
\begin{array}{llllllll}
X244, & X245, & X246, & X247, & X248 & \longrightarrow & ¿Y251? \\
X245, & X246, & X247, & X248, & X249 & \longrightarrow & ¿Y252? \\
X246, & X247, & X248, & X249, & X250 & \longrightarrow & ¿Y253? \\
\ldots & & & & & & \\
X283, & X284, & X285, & X286, & X287 & \longrightarrow & ¿Y290?
\end{array}
$$

El resultado de las previsiones aparece representado en la figura 16.

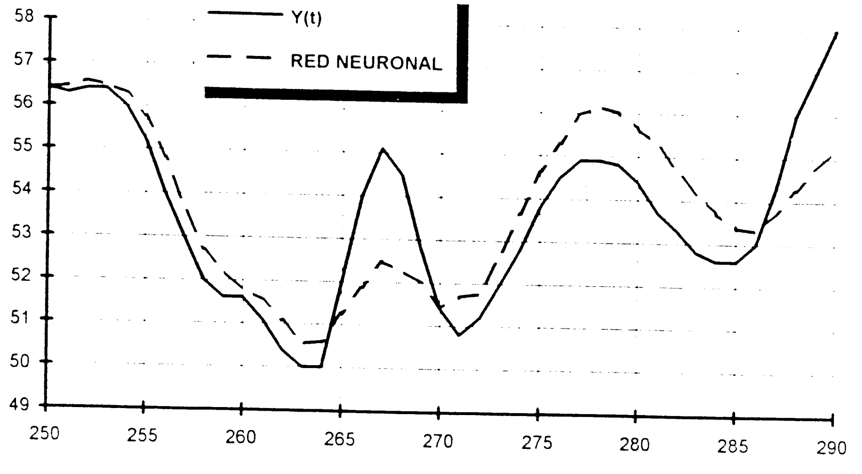

Figura 16.
Previsiones de la serie $Y(t)$ calculadas por la red53.

4. ANÁLISIS COMPARATIVO

En la figura 17, aparecen representadas las previsiones calculadas a partir de la red neuronal junto con las previsiones obtenidas utilizando la metodología de Box-Jenkins, comparadas con los valores reales de la serie UN05. Se puede comprobar que las previsiones calculadas por ambos métodos son bastante aceptables ya que parecen seguir a la serie real de una forma suficientemente correcta. En la tabla 5, están recogidos los valores del error cuadrático medio MSE, que han sido calculados comparando las previsiones con los valores reales de la serie UN05 para el modelado univariante.

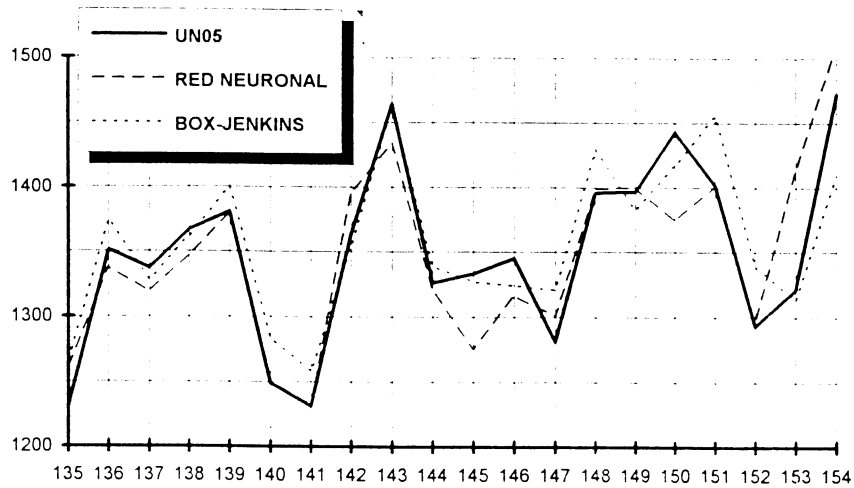

Figura 17.
Comparación de Previsiones en el modelado univariante.

Tabla 5

Valores del MSE de las previsiones.

	RED NEURONAL	BOX-JENKINS
MODELADO UNIVARIANTE	1.152,8852	876,8597
FUNCIÓN DE TRANSFERENCIA	1,3712	2,7615

Un aspecto destacable de las previsiones calculadas por ambos métodos, y que se puede comprobar en la figura 17, es que, en la mayoría de las ocasiones, ambos métodos aciertan en el cálculo de los llamados "turning points" de la serie, es decir, aquellos puntos en los que la serie cambia de tendencia. Este hecho es muy interesante puesto que en muchas ocasiones, más que obtener unas previsiones en las que una medida del error (como puede ser el error cuadrático medio que hemos calculado nosotros), sea muy pequeña en términos relativos, es mucho más importante averiguar con exactitud los puntos en los que la serie cambia de tendencia, aunque luego el error que se cometa sea mayor. En cuanto a este aspecto, se observa que tanto Box-Jenkins como la Red Neuronal aciertan bastante bien, sobre todo en la primera mitad de las previsiones calculadas.

En la tabla 5, se comprueba que, aunque el método de Box-Jenkins consigue unas previsiones mejores que las de la red neuronal en cuanto al error cuadrático medio, las previsiones de la red neuronal son igualmente aceptables. Sin embargo, hay que destacar el hecho de que cuando se estudió la arquitectura óptima de la red neuronal, se utilizó información conseguida durante la etapa de modelado de Box-Jenkins, por ejemplo, durante la etapa de modelado de Box-Jenkins se comprobó que se trataba de una serie con una estacionalidad de 12 períodos. Esta información se utilizó en la etapa de diseño de la red neuronal ya que se supuso que una red en la que cada valor depende de como mínimo de 12 valores anteriores, debería dar, en principio, mejores resultados que cualquier red con menos neuronas de entrada, como así ha resultado.

Este aspecto fue aún más importante en el caso de la red neuronal que tenía que funcionar como función de transferencia en el caso del horno. No existen de momento, reglas más o menos fijas que se puedan utilizar a la hora de diseñar la arquitectura de la red neuronal, por lo que la forma de llegar a una estructura que sea óptima, no consiste en otra cosa que ir probando red tras red hasta dar con la que pueda ser buena. Tal como se puede comprobar en la figura 18 y en la tabla 5, cuando se encuentra una red aceptable, las previsiones que se obtienen, son bastante mejores que las obtenidas por Box-Jenkins, el error cuadrático medio de las previsiones de la red es prácticamente la mitad del error cuadrático de las previsiones de Box-Jenkins.

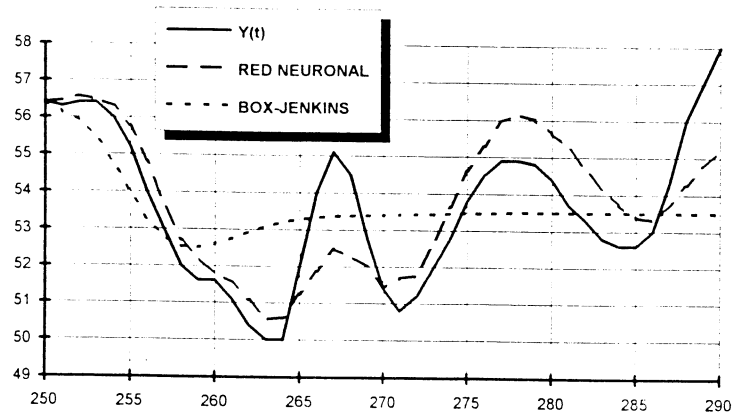

Figura 18.
Comparación de las previsiones en el caso de función de transferencia.

Sin embargo, en este segundo caso, fue todavía más difícil dar con la arquitectura óptima de la red. Gracias al estudio de las series que se ha de hacer para utilizar el método de Box-Jenkins, se pudo saber que la salida respondía a las variaciones de la entrada tres períodos más tarde. Luego, en la etapa de diseño de la red neuronal, se fueron probando redes en las que la salida aparecía retrasada cero, uno, dos, tres

y cuatro períodos con la esperanza de que la red en la que la salida estaba retrasada tres períodos fuera la que diera los mejores resultados, lo cual se pudo comprobar posteriormente con el cálculo del error cuadrático medio. Es decir, se llevó a cabo una "búsqueda dirigida" de la mejor red neuronal, o al menos de una buena red neuronal. De no haberlo hecho así, el proceso de búsqueda de la red podría llegar a ser ciertamente complicado porque podría eternizarse a no ser que, en un golpe de suerte, se diera con la red óptima.

5. CONCLUSIONES

Se ha hecho un estudio comparativo del cálculo de previsiones mediante Box-Jenkins y Redes Neuronales Artificiales. Aunque debería de comprobarse los resultados con muchos más casos, podemos intuir que se obtienen resultados aceptables con las dos metodologías para previsiones univariantes de series temporales.

En cambio, parece ser mejor el cálculo de previsiones de funciones de transferencia por Redes Neuronales, pero la dificultad de encontrar la red adecuada para cada caso es una limitación importante de esta aproximación. Hemos encontrado que este problema puede ser resuelto, al menos en parte, gracias a la información que se obtiene de la aplicación de la sistemática metodología de Box-Jenkins, que resulta ciertamente útil a la hora de hacer una "búsqueda dirigida" de la arquitectura adecuada de la red, y evita el tener que entrar en procesos de tipo "prueba y error" de forma indiscriminada.

REFERENCIAS

[1] **Alonso, G.** y **Becerril, J.L.** (1993). *Introducción a la Inteligencia Artificial.* Multimedia Ediciones S.A.

[2] **Box, G.E.P.** y **Jenkins, G.M.** (1976). *Time Series Analysis: Forecasting and Control.* 2nd. Ed. San Francisco: Holden Day.

[3] **Chapman, A.J.** (1994). "Stock market trading systems through neural networks: developing a model". *Int. Journal of Applied Expert Systems.* **12, 2.**

[4] **Eberhart, R.C.** y **Dobbins, R.W.** (1990). *Neural Network PC Tools.* Academic Press.

[5] **Farmer, J.D.** y **Sidorowich, J.J.** (1987). "Predicting chaotic time series". *Physical Review Letters*, **59**, 845–848.

[6] **Granger, C.W.J.** y **Newbold, P.** (1986). *Forecasting economic time series.* (2nd. ed.). Orlando, FL: Academic Press.

[7] **Kimoto, T., Asakawa, K., Yoda, M.** y **Takeoka, M.** (1990). "Stock market prediction system with modular neural networks". *IJCNN Int. Joint Conference on Neural Networks.* San Diego, CA.

[8] **Lapedes, A.** y **Farber, R.** (1987). "NonLinear signal processing using neural networks". *IEEE Conference on Neural Information Processing System - Natural and Synthetic.*

[9] **Lippmann, R.P.** (1987). "An Introduction to Computing with Neural Nets". *IEEE ASSP Magazine.* April 1987.

[10] **Neuroshell "Neural Network Shell Program".** (1989). *Ward Systems Group Inc.*

[11] **Priestley, M.B.** (1988). *Non-Linear and non-stationary time series analysis.* San Diego, CA: Academic Press.

[12] **Reilly, D.P.** (1980). "Experiences with an Automatic Box-Jenkins Modeling Algorithm". In *Time Series Analysis.* Ed. O.D. Anderson. (Amsterdam: North Holland), pp. 493–508.

[13] **Richeson, L.** y **Zimmermann, R.A.** (1994). "Predicting consumer credit performance: can neural networks outperform traditional statistical methods?". *Int. Journal of Applied Expert Systems.* **2, 2**, February, 1994.

[14] **Salchenberger, L.M., Cinar, E.M.** y **Lash, N.A.** (1992). "Neural Networks: a new tool for predicting thrift failures". *Decision Sciences.* July-August, 1992.

[15] **Tiao, G.C.** y **Box, G.E.P.** (1983). *An Introduction to Applied Multiple Time Series Analysis.* Illinois, USA: Scientific Computing Associated.

[16] **Tiao, G.C.** y **Tsay, R.S.** (1989). "Model specification in multivariate time series". *Journal of the Royal Statistical Society.* **B 51**, 157–213.

[17] **Tong, H.** (1983). "Threshold models in non-linear time series analysis". *Lecture Notes in Statisti.* **21**. New York: Springer-Verlag.

[18] **Tong, H.** (1990). *Non-Linear time series: A dynamical system approach.* Oxford: Oxford University Press.

[19] **Zaremba, T.** (1990). *Technology in Search of a Buck.* Neural Network PC Tools. Academic Press Inc.

[20] **Zurada, J.** (1991). *Using Nworks: An Extended Tutorial for Neural Works Proffessional II/Plus and NeuralWorks Explorer.* Pittsburg.

ENGLISH SUMMARY:

COMPARATIVE ANALYSIS OF UNIVARIATE AND TRANSFER-FUNCTION FORECASTING USING NEURAL NETWORKS AND BOX-JENKINS TECHNIQUES

David de la Fuente García and Raúl Pino Díez

Time series forecasting is a very important statistical tool in studying time dependent data behaviour and in predicting future values.

A time series is a set of data measured in discrete or continous time units. A multivariate time series consists of a set of data from several time dependent variables. A special case is when the measured variables are significantly correlated, for instance, when similar attributes are being measured at distinct geographical locations. When forecasting a new value for each variable, the best prediction is achieved by taking into account the variations of the other variables.

Traditionally speaking, and especially during the 1980's, there were many techniques available for time series analysis, which assumed linear relationships between variables (Box- Jenkins, 1976). But in real life situations, data do not have such simple relationships, and it is difficult to make good forecasts. This is the reason why it seems necessary to build non-linear models to analize time series data in real cases.

Linear time series models also present certain disadvantages, such as not explaining sudden changes over a very wide range during irregular time intervals (Tong, 1983).

Further studies (Tiao and Tsay, 1989) analyzed other problems involved in multivariable modelling. In order to solve them, non-linear statistical models were used, such as the "threshold" and "bilinear" models suggested by Tong (1990). Alternatively, Granger and Newbold (1986) suggested using non-linear transformations of the original data before traditional linear modelling. Farmer and Sidorowich (1987) significantly improved chaotic time series predictions by using local approximations methods.

Although in the past decade major advances have been made, non-linear modelling is, unfortunately, an extremely hard task, due to the simplifications made in the modelling step, such as omitting parameters which are unknown or which are expected not to have a direct effect on the data.

In order to achieve good forecasts, we have hence looked to artificial Neural Networks as an alternative for calculating quantitative predictions.

In this work, we present a overview of the Box-Jenkins methodology for univariate time series modelling and transfer functions with one input and one output. Subsequently, we describe the Neural Networks approach and then go on finally to compare both methods with a number of examples. From these examples, it can be seen that the forecasts from both methods are quite acceptable, because they follow the actual series in a sufficiently correct way.

A key aspect of forecasts obtained from both methods is that in most cases both of them succeed in calculating the series turning points, i.e., those observations at which the series trend changes. This is very important since it is often better to try to find out the exact observations at which the series trend changes, rather than to obtain a good forecast in terms of a relatively small error. With regards to this objective, it can be seen that both Box-Jenkins and Neural Networks techniques are fairly accurate, above all in the first half of the forecasts.

When the optimal architecture for the net was studied, information obtained from Box- Jenkins modelling step was used. For example, during the Box-Jenkins modelling step, it was proven that the series had an order 12 seasonality. This information was used in the design step of the nets because it was assumed that a 12-period related net would have better performance than any other net.

This aspect is even more important in the case of the net which has to work as a transfer function. Presently, there are no rules for designing the architecture of Neural Networks, so the best way to obtain an optimum structure is by testing net by net until we find the one which may be good. Thanks to the series analysis that has to be done in order to use the Box-Jenkins method, we were able to ascertain that the output followed the input variations with a 3-period delay. Later, in the design step of the Neural Net, we tested nets in which the output was delayed in zero, one, two, three and four periods. This was in the hope that the one with a 3-period delay was the best of all. We then proved this by observing the MSE. Thus, it was a "guided search" of the best net or, at least, of a good one.

In conclusion, we can state that, although the results have to be checked against many more experiences, good performances are obtained with both methods for univariate time series forecasting. On the other hand, Neural Networks seem to be better in transfer function forecasting, although a handicap for this method is the difficulty of finding the adequate net for each case. We found a partial solution to this problem in the application of the systematic Box- Jenkins method which is very useful when carrying out a "guided search" of the appropiate architecture for the net, thus avoiding indiscriminate "trial and error" processes.