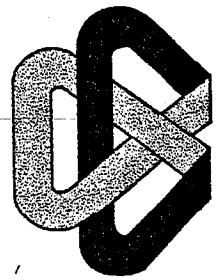


Centro de Investigación en Matemáticas, A.C.

CIMAT

Redes neuronales para predicción: Una aplicación al tipo de cambio.

TESINA

Que para acreditar la materia:

Actividades Académicas Especiales II

(Eficiencia Profesional)

Presenta:

Helga Abreu Fetter

Guanajuato, Gto. México. Junio de 2002

Comité de Evaluación:

Johan J. Van Horebeek
Rogelio Ramos Quiroga
Graciela González Farías

Redes Neuronales para Predicción: Una Aplicación al Tipo de Cambio

Helga Abreu Fetter
CIMAT

11 de junio de 2002

Resumen

Se presenta un resumen de las técnicas de redes neuronales que se utilizan para realizar predicciones. Se aplican algunas de ellas para hacer predicción del tipo de cambio peso-dólar, y se realizan comparaciones contra modelos de tipo lineal.

1 Introducción

La Hipótesis de Mercados Eficientes (HME) afirma que toda la información existente en los mercados está reflejada en los precios de los activos, lo cual se traduce en una imposibilidad de predecir los movimientos de estos precios. Es decir, los precios deben seguir una caminata aleatoria[1]. Hay varias versiones de esta hipótesis, desde la débil, que afirma que toda la información pública e histórica está incluída en los precios, hasta la fuerte que afirma que toda la información, incluyendo la interna de las empresas y gobiernos, está ya reconocida por el mercado.

Nosotros consideramos que la realidad está enmedio de estas dos versiones de la HME. Existe una gran cantidad de analistas e inversionistas analizando toda la información posible que pueda ayudarles a mejorar sus inversiones. Sin embargo, el hecho de que estos esfuerzos rindan frutos, significa que es posible encontrar algún tipo de información que ayude a tener mejores predicciones que el resto del mercado. Es claro que para encontrar este tipo de "revelaciones" se requiere de técnicas sofisticadas y novedosas, que nos den una ventaja sobre el resto de los analistas. Existe una vasta literatura donde se plantean técnicas de predicción que han dado resultados con altos rendimientos (en general teóricos, pues poca gente que publica sus resultados de hecho invierte dinero en sus técnicas), mostrando así que esto es posible.

[1] Esto es, se trata de una serie de variables aleatorias $\{y\}_t$ con la siguiente dinámica: $y_t = y_{t-1} + \epsilon_t$ donde ϵ_t es una serie con media 0, varianza σ^2 y $E(\epsilon_i, \epsilon_j) = 0$ para i diferente de j (ruido blanco).

Las técnicas más utilizadas de análisis de series financieras son el análisis técnico y el análisis fundamental. El primero es escencialmente una técnica de búsqueda de patrones en las gráficas de los precios. De la HME se deduce que esta técnica debe ser inútil, sin embargo la experiencia determina que esto no es del todo cierto. Por ejemplo, se sabe que después de que una acción sube fuértemente de precio, la psicología del inversionista llevará a una "toma de utilidades" reduciendo el precio de la acción en un porcentaje de el incremento original. La pregunta es ¿Cuándo ocurrirá este fenómeno? Esto es lo que hace que no sea del todo predecible el comportamiento de la acción. Existen varias técnicas muy conocidas de análisis técnico, como el uso de promedios móviles y de "barreras" móviles, así como de el análisis de "figuras" en el comportamiento de los precios. Estas técnicas han resultado de cierto valor y se utilizan de manera consistente en las instituciones financieras. De manera implícita, el uso de redes neuronales para la predicción de series de tiempo es un sistema de tipo análisis técnico, ya que se basa en el pasado de la serie para deducir su futuro.

El segundo, el análisis fundamental, se basa en el estudio de los pronósticos de ventas, ganancias, dividendos, etc. de las empresas, así como de las expectativas sobre las tasas de interés y otras variables macroeconómicas. Una vez más, de la HME se deduce que, en general, este análisis está destinado al fracaso, aunque si se cuenta con información de tipo confidencial o se tiene alguna manera muy especial de pensar puede dar frutos. Al utilizar como entradas de la red otras variables financieras, además de los rezagos de la serie de interés, estamos añadiendo un componente fundamental a nuestras predicciones, que de otra manera serían meramente técnicas.

En este trabajo realizamos tres aplicaciones de las redes neuronales a la predicción del tipo de cambio peso-dólar, y las comparamos con modelos equivalentes de series de tiempo (tipo Auto Regresivo o AR). En la primera aplicación trabajamos con datos diarios, de 1999 a 2002, utilizando como entradas únicamente rezagos de la misma serie. En la segunda aplicación trabajamos con datos mensuales de 1995 a 2002, nuevamente usando como entradas los mismos rezagos de la serie. En la última aplicación, utilizando los mismos datos mensuales, hacemos uso de la teoría financiera que indica que el tipo de cambio tiene una relación de equilibrio con respecto a la diferencia de inflaciones entre los dos países involucrados. Con base en esta teoría, entrenamos una red utilizando un rezago del tipo de cambio, más una variable que refleja la diferencia de inflaciones entre E.U. y México.

En la segunda sección, presentamos una introducción a las redes neuronales, enfocándonos a las que usaremos en la parte empírica de nuestro estudio. En la sección 3 presentamos un resumen de los resultados publicados sobre predicción utilizando redes neuronales, y por último, en la sección 4 mostramos los resultados de nuestras predicciones del tipo de cambio. En la sección 5 se presentan las conclusiones.

2 Redes Neuronales

El concepto de red neuronal consiste en una estructura computacional compuesta de un gran número de pequeños elementos procesadores interconectados (que asemejan de cierta manera las neuronas del cerebro), trabajando en paralelo. Esta estructura permite hacer muchas operaciones simultáneamente, en contraposición al proceso en serie tradicional. Antes de la aparación de las redes neuronales, el procedimiento estándar para el proceso de información tenía las siguientes características:

- El conocimiento se representaba explícitamente con reglas, modelos, etc.
- Se imitaba el proceso humano de razonamiento lógico para resolver los problemas.
- Se procesaba la información secuencialmente.

Sin embargo, con el rápido desarrollo en algunos campos de la inteligencia artificial, tales como el reconocimiento de patrones, aparecieron un gran número de problemas complejos donde no era conveniente una representación explícita del conocimiento y no se disponía de un razonamiento lógico para resolverlo. Las redes neuronales artificiales fueron introducidas como método alternativo de computación, con el propósito de reproducir las funciones del cerebro humano. El cerebro está compuesto de cerca de 10^{11} neuronas que reciben señales electroquímicas de otras neuronas a través de uniones sinápticas que conectan el axón de las neuronas emisoras con las dendritas de las neuronas receptoras. Basándose en la información recibida, la neurona computa y envía su propia señal. El proceso de emisión es controlado por el potencial interno asociado a una neurona. Si este potencial supera un cierto umbral, se envía un impulso eléctrico al àxón. En caso contrario no se envía ninguna señal.

2.1 Componentes de las redes neuronales

Definición 1 *Una neurona sobre un conjunto de nodos N es una tripleta (X, f, Y), donde X (nodos de entrada) es un subconjunto de N, Y (nodo de salida) es un único nodo de N y $f : \Re \to \Re$ es una función neuronal (o de activación) que calcula un valor de salida para Y basado en una combinación lineal de los valores de los componentes de X, es decir[6]*

$$Y = f(\sum_{x_i \in X} w_i x_i)$$

En caso de que se quiera tomar en cuenta un valor umbral θ para una neurona, se puede conectar una neurona auxiliar $x_0 = -1$ con un peso $w_i = \theta$.

Definición 2 *Una red neuronal artificial (RNA) es un par (N, U) donde N es un conjunto de nodos y U es un conjunto de unidades procesadoras sobre N que satisface que cada nodo $X_i \in N$ tiene que ser un nodo de entrada o de salida de al menos una unidad procesadora de U[6].*

Entonces, en el contexto de la predicción de una serie de tiempo, en el que las entradas son valores pasados de la misma serie y posiblemente de algunas otras series, y la salida es la predicción a futuro de la serie, la RNA realiza un mapeo de la forma $y_{t+j} = f(y_t, y_{t-1}, ..., y_{t-j+1}, x_1, ..., x_p)$. En este sentido, la red es equivalente a un modelo de regresión no lineal.

Las funciones de activación más populares son:
- Funciones lineales.
- Funciones escalón.
- Funciones sigmoidales.

Las neuronas se pueden organizar en capas conectadas con conexiones hacia adelante, laterales (en la misma capa) y hacia atrás (o recurrentes). La topología de la red, que puede definirse con una matriz de pesos W, donde cada columna w_i es el vector de pesos de las conexiones de la neurona i con las demás neuronas tiene tres tipos de neuronas: la capa de entrada, la capa de salida y las capas ocultas o intermedias. Modificando el número de capas, el número de neuronas en cada capa y las conexiones entre ellas se conforman las diferentes arquitecturas de redes. Entre las arquitecturas más populares y poderosas se encuentran las de retro-propagación (back-propagation), también conocidas como perceptrones multicapa, que están formadas por una capa de entrada, una de salida y un número arbitrario de capas ocultas. Cada una de las capas ocultas o de salida recibe una entrada de las neuronas de la capa previa. Esta será la estructura que estudiaremos en este trabajo. Otra estructura popular es la de Hopfield, que consiste de una sola capa con todas las conexiones posibles entre las diferentes neuronas. Esta red sirve para dada una información incompleta o corrupta, recuperar la información original correspondiente.

2.2 Aprendizaje

Una de las principales características de una RNA es su capacidad de aprender a partir de unos datos. Una vez establecida la arquitectura de la red, los pesos de las conexiones se ajustan para codificar la información contenida en un conjunto de datos de entrenamiento. Hay dos tipos de aprendizaje:

- Aprendizaje supervisado. Los datos de entrenamiento consisten de una serie de parejas $\{(a_p, b_p), p = 1...r\}$, donde el vector a representa los datos de entrada y b el vector de salida correspondiente a a. En este caso, los pesos se obtienen minimizando una función de error que mide la diferencia entre los valores de salida deseados y los generados por la red. El punto clave de este tipo de aprendizaje es la convergencia de la función de error a un mínimo, ya que esta puede tener muchos mínimos locales.
- Aprendizaje no supervisado. Los datos se presentan a la red sin ningún tipo de información externa y la red tiene que descubrir por si misma patrones o categorías.

Una vez terminado el proceso de aprendizaje y que los pesos de la red han sido calculados, es necesario comprobar la calidad del modelo resultante. En el caso del aprendizaje supervisado, la medida de calidad es en término de los

errores entre los valores deseados y los resultados de la red. Para esto, pueden usarse medidas como la suma de cuadrados de los errores (SSE), la raíz cuadrada del error cuadrático medio (RMSE) o el error máximo.

Otra manera de validar la red es dividir los datos en dos partes, una destinada al entrenamiento de la red y otra parte para comprobar su eficacia. Cuando el error de comprobación es mucho mayor que el error de ajuste, puede haber un problema de sobre ajuste. Este es el mismo problema que ocurre en estadística cuando se trata de ajustar un conjunto de datos procedentes de un proceso con pocos grados de libertad, ya que se acaba realizando una interpolación de los datos incluyendo el ruido sin tomar en cuenta las tendencias.

2.3 Normalización de los datos

Las funciones de activación no lineales, tales como la logística o la hiperbólica tienden a aplastar las salidas de cada nodo, típicamente en (0,1) ó (-1,1). Debido a esto, es común normalizar los datos de entrada antes de empezar el proceso de aprendizaje. Hay cuatro métodos de normalización:

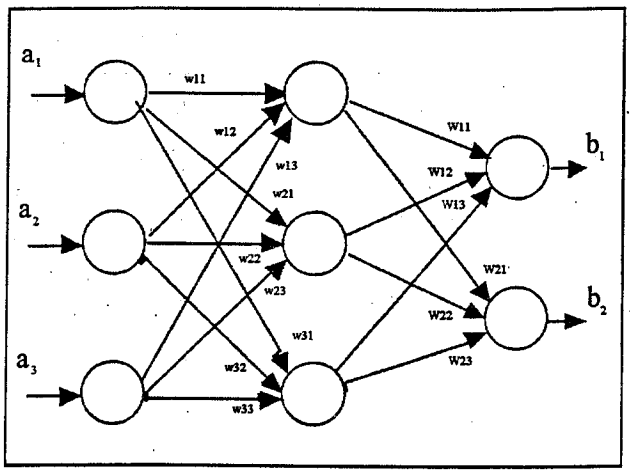
- Dentro de un canal. Se normaliza cada variable de entrada por separado.
- Através de canales. Se normalizan todos los elementos de un patrón de datos de manera conjunta.
- Mezcla de canales. Usa una combinación de las anteriores.
- Externa. Todos los datos de aprendizaje se normalizan a un rango específico.

Para cada tipo de normalización, se pueden utilizar fórmulas de transformación a [0,1], a [a,b], o una normalización de tipo estadística. Esto depende de la función de transferencia elegida, siendo el [0,1] adecuado para la logística y [-1,1] para la tangente hiperbólica. En algunos casos, se utiliza [0.1,0.9] basado en que las funciones de activación no-lineales suelen tener límites asintóticos, y que las salidas pueden ocurrir sólo en este intervalo, o inclusive en alguno más pequeño [1].

Es importante notar que como resultado de la normalización, los valores de salida de la red corresponderán al rango normalizado, y por lo tanto para interpretar los resultados de la red los datos deben regresarse a su escala original. Además, la exactitud de los resultados debe medirse en términos de la escala original.

2.4 Perceptrones multicapa

Los perceptrones multicapa (PMC) consisten de una capa de entrada, una de salida y un conjunto de capas ocultas. Los nodos de la capa de entrada alimentan la red distribuyendo hacia adelante las señales de entrada. Cada nodo en las capas ocultas y de salida recibe una entrada de los nodos de las capas previas y calcula un valor de salida para la siguiente capa. La siguiente figura presenta un ejemplo de un perceptrón multi-capa con una capa oculta.



Las capas ocultas proporcionan a este tipo de redes la flexibilidad de resolver problemas no lineales. Para cada una de las salidas b_i un PMC calcula una función $b_i = F_i(a_1, a_2, ..., a_n)$ de las entradas. Por ejemplo, la red de la figura anterior define la función:

$$b_i = f(\sum_k W_{ik} f(\sum_j w_{kj} a_j))$$

El algoritmo de aprendizaje más popular para los PMC es el de retro-propagación.

2.5 Algoritmo de retro-propagación

Este algoritmo está basado en minimizar la función de error cuadrático total, usando el método de descenso de gradiente. Supongamos que tenemos un conjunto de entradas $\{a_{p1}, ...a_{pn}\}$ y sus correspondientes salidas $\{b_{p1}, ...b_{pn}\}$, $p = 1, 2, ..., r$. Se considera la función del error cuadrático total:

$$
\begin{aligned}
E(w) &= \frac{1}{2}\sum_{p,i}(b_{pi}-\widehat{b}_{pi})^2 \\
&= \frac{1}{2}\sum_{p,i}(b_{pi}-f(\widehat{B}_{pi}))^2 \\
&= \frac{1}{2}\sum_{p,i}(b_{pi}-f(\sum W_{ik}\widehat{h}_k))^2 \\
&= \frac{1}{2}\sum_{p,i}(b_{pi}-f(\sum_k W_{ik}f(\widehat{H}_k)))^2 \\
&= \frac{1}{2}\sum_{p,i}(b_{pi}-f(\sum_k W_{ik}f(\sum_j w_{kj}a_{pj})))^2
\end{aligned}
$$

Usando la idea de descenso de gradiente, por medio del método delta, se cambian los pesos de forma que se descienda por la pendiente de la función de error:

$$\Delta W_{ik} = -\eta\frac{\partial E}{\partial W_{ik}};\quad \Delta w_{kj} = -\eta\frac{\partial E}{\partial w_{kj}};$$

donde η es el parámetro de aprendizaje.

Para ir actualizando los pesos de las neuronas ocultas y las neuronas de salida, se utiliza un algoritmo de retro-propagación de dos pasos. Primero, la entrada a_p se propaga haca adelante obteniendo el valor de las unidades ocultas $\widehat{h}_p$, y la salida $\widehat{b}_p$, y por lo tanto, el error asociado. Estos valores se usan para actualizar los pesos W_{ik} de la capa de salida. Después, estos pesos se utilizan para actualizar los pesos de la capa oculta w_{ik}, para propagar los errores hacia atrás.

Usando la regla de la cadena se tiene que

$$\Delta W_{ik} = -\eta\frac{\partial E}{\partial W_{ik}} = -\eta\frac{\partial E}{\partial \widehat{b}_{pi}}\frac{\partial \widehat{b}_{pi}}{\partial \widehat{B}_{pi}}\frac{\partial \widehat{B}_{pi}}{\partial W_{ik}}$$

donde $\widehat{b}_{pi} = f(\widehat{B}_{pi})$ es la iésima salida de la red obtenida por la propagación hacia adelante. Entonces:

$$
\begin{aligned}
\frac{\partial E}{\partial \widehat{b}_{pi}} &= -(b_{pi}-\widehat{b}_{pi}) \\
\frac{\partial \widehat{b}_{pi}}{\partial \widehat{B}_{pi}} &= f'(\widehat{B}_{pi}) \\
\frac{\partial \widehat{B}_{pi}}{\partial W_{ik}} &= \widehat{h}_k
\end{aligned}
$$

Por lo que tenemos la siguiente regla de actualización: $\Delta W_{ik} = \eta \widehat{h}_k \delta_{pi}$, donde $\delta_{pi} = (b_{pi} - \widehat{b}_{pi}) f'(\widehat{B}_{pi})$.

Igualmente, para modificar los pesos de la capa oculta se tiene:

$$\Delta w_{kj} = -\eta \frac{\partial E}{\partial w_{kj}} = -\eta \sum_i \frac{\partial E}{\partial \widehat{b}_{pi}} \frac{\partial \widehat{b}_{pi}}{\partial \widehat{B}_{pi}} \frac{\partial \widehat{B}_{pi}}{\partial \widehat{h}_k} \frac{\partial \widehat{h}_k}{\partial \widehat{H}_k} \frac{\partial \widehat{H}_k}{\partial w_{kj}}$$

Con

$$\begin{aligned} \frac{\partial \widehat{B}_{pi}}{\partial \widehat{h}_k} &= W_{ik} \\ \frac{\partial \widehat{h}_k}{\partial \widehat{H}_k} &= f'(\widehat{H}_k) \\ \frac{\partial \widehat{h}_k}{\partial w_{kj}} &= a_{pj} \end{aligned}$$

y por lo tanto, $\Delta w_{kj} = -\eta a_{pj} \delta_{pi} \psi_{pi}$, donde $\psi_{pi} = \sum_k \delta_{pi} W_{ki} f'(H_k)$. Con lo cual tenemos ya el algoritmo completo para ir actualizando los pesos, lo cual se repite con cada patrón de entrada.

En ocasiones, se utiliza un "término de momento" para suavizar las estimaciones de las derivadas. Entonces,

$$\Delta w_{kj}(t+1) = -\eta a_{pj} \delta_{pi} \psi_{pi} + \beta \Delta w_{kj}(t)$$

El método de Weight Decay consiste en penalizar los pesos muy grandes. A la hora de actualizar los pesos, se resta un término de tipo βw_{ij}, donde β es un parámetro de momento:

$$\Delta w_{kj} = -\eta (a_{pj} \delta_{pi} \psi_{pi} - \beta w_{kj})$$

Hay autores que proponen otro tipo de penalizaciones, tales como potencias cuartas de los pesos, o inclusive términos que penalizan los pesos pequeños en lo guar de los grandes.

2.6 Redes Neuronales vs. Modelos Estadísticos

Existe una controversia entre estadísticos y neuronistas, como puede verse del siguiente párrafo escrito por Anderson, Pellionisz y Rosenfeld [2] :

Las Redes Neuronales son Estadística para amateurs. Una red bien diseñada, mientras aprende y responde, está efectuando una buena inferencia estadística,

basado en lo que vión cuando aprendió y lo que ve cuando responde. La mayor parte de las redes ocultan la estadística al usuario[2].

Se puede generar un modelo de redes neuronales sin tener ningún tipo de teoría económica o biológica como fundamento, más que tal vez una intuición sobre qué variables pueden afectar nuestra variable de interés. Para crear un modelo econométrico es necesario tener una idea previa de cómo debe verse el modelo.

Es importante resaltar que los modelos de regresión lineal son un subconjunto de los modelos que se pueden adaptar utilizando redes neuronales. Sin embargo, las redes son como una "caja negra", no nos dan información acerca de los parámetros que conforman el modelo, sólo nos presentan con predicciones o clasificaciones sin "explicar" como lo hacen. En el caso de los modelos estadísticos podemos analizar los parámetros y por lo tanto la influencia de cada variable independiente en nuestra variable dependiente, así como hacer inferencias sobre ellos y las predicciones. Ripley [26] afirma que, aunque hay pocas comparaciones entre métodos, cuando estas se hacen cuidadosamente, los métodos estadísticos resultan mejores que las redes neuronales más sofisticadas (state of the art).

En este trabajo realizamos una comparación entre modelos de redes neuronales y modelos econométricos y de series de tiempo. Sin embargo, reconocemos que ambas técnicas podrían mejorarse respecto a lo que presentamos aquí.

3 Predicción con redes neuronales

Hay varias razones por las cuales las redes neuronales son valiosas y atractivas para hacer predicciones. Primero, al contrario de los métodos tradicionales basados en modelos, las RNA son métodos adaptables que están dirigidos por los datos, en cuanto a que hay pocos supuestos a priori. Por lo tanto, las RNA son adecuadas para problemas en que las soluciones requieren de conocimientos que son difíciles de especificar, pero para los cuales hay suficientes datos u observaciones. Esto es útil, ya que en muchos casos es más fácil tener datos que buenas teorías sobre las relaciones subyacentes que gobiernan los sistemas que generan los datos. En segundo lugar, las RNA generalizan, por lo que una vez que han aprendido de ciertos datos, pueden realizar inferencias sobre la parte desconocida de la población, aún si los datos tienen ruido. En tercer lugar, las RNA son aproximadores universales de funciones. Se ha demostrado que una red puede aproximar cualquier función contínua con la exactitud que se desee [35]. Cualquier modelo de predicción supone que hay una relación subyacente (conocida o desconocida) entre las entradas (los valores pasados) y las salidas (los valores futuros) y las RNA son una buena alternativa para encontrar esta relación. Por último, las redes neuronales son no lineales. La predicción ha sido dominada por la estadística lineal; sin embargo, no hay una razón por la cual una realización de una serie de tiempo deba ser generada por un proceso lineal,

[2] *Neural Networks are Statistics for amateurs. A properly designed network, when learning and responding, performs good statistical inference, based on what it saw when it learned and what it sees when it responds. Most networks conceal the statistics from the user.*

y por lo tanto las RNA son una buena alternativa a los métodos tradicionales de Box y Jenkins o ARIMA. En general es muy difícil formular un modelo no lineal para un conjunto de datos en particular, ya que hay una gran cantidad de parámetros posibles, y un modelo preespecificado puede no ser suficientemente general para capturar todas las características importantes de los datos.

La idea de utilizar RNA para la predicción no es nueva. La primera aplicación data de 1964 por Hu [17], que utilizó una red lineal para predecir el clima. Sin embargo, fue hast 1986 que se introdujo el algoritmo de retro-propagación [27] y desde entonces ha habido mucho desarrollo de esta área.

3.1 Aplicaciones

Zhang [35] presenta una larga lista de aplicaciones de las RNA a la predicción, entre ellas: modelación y predicción de sistemas dinámicos no-lineales y series de tiempo caóticas, con o sin ruido, actividad de las manchas solares, varias aplicaciones a la física, estudios de consumo de energía eléctrica, temperaturas ambientales, índices macroeconómicos, nivel de ozono, calificaciones promedio de estudiantes, producción industrial, transportación, etc.

En particular, existen varias aplicaciones en la literatura financiera. Entre otras, se han utilizado las RNA para predicción de quiebras, precios de acciones, de tasas de interés y de tipo de cambio, que es el problema a tratar en este trabajo.

3.2 Topología de la red

Para un problema de predicción causal, se suele usar la topología típica de perceptrones multi-capa (PMC), aunque algunos autores utilizan algunas variantes [35]. Lapedes y Farber [22] fueron los primeros en utilizar este tipo de redes para predicción en 1987.

El problema consiste en encontrar un modelo parsimonioso para un problema real, ya que el sobre-ajuste es común dentro de las RNA. Hay varios enfoques para resolver el problema de encontrar el tamaño adecuado para una red de predicción de series de tiempo reales. Weigend et al.[31] [30] [32] proponen un método de castigo introduciendo una función de costos que penaliza la complejidad en la red. De Groot y Wurtz [10] presentan una red PMC con un criterio basado en el criterio de información de Akaike para modelar y analizar datos de series de tiempo.

Lachtermacher y Fuller [21] combinan métodos de Box-Jenkins con RNA para minimizar el tamaño de la red y por lo tanto los requerimientos de datos para el entrenamiento. Utilizando la metodología de Box-Jenkins encuentran un modelo ARIMA adecuado para los datos, y luego usan una RNA con los rezagos indicados por el modelo obtenido en el primer paso. Balkin y Ord [3] utilizan este mismo enfoque. Esta es la metodología que utilizaremos en este trabajo.

Algunos autores utilizan varias redes en conjunto, ya sea en paralelo o en serie. Cheng et al. [7] utilizan un total de 32 redes para predecir la tasa del bono

del tesoro a 30 años de E.U, combinando en dos nuevas redes los resultados de las redes más exactas con las menos riesgosas para llegar a un resultado final.

Los componentes importantes que hay que elegir para tener una predicción exitosa son la arquitectura adecuada (número de capas, número de nodos en cada capa, conexiones entre los nodos), la función de activación, el algoritmo de entrenamiento, el método de normalización de los datos, los conjuntos de datos de entrenamiento y prueba, y las medidas de ajuste.

3.2.1 La arquitectura de la red

En el modelo popular PMC todos los nodos de entrada están en la misma capa, los nodos de salida en otra, y los nodos ocultos están distribuídos en una o más capas ocultas en medio. La selección del número de capas y nodos en cada capa depende del problema específico. Existen algunos algoritmos para definir estas variables, pero en general son complejos y difíciles de implementar. Hasta ahora, no parece haber ningún método claro y eficaz para determinar estos parámetros, por lo que tiene algo de arte.

Número de capas ocultas y nodos Los nodos de las capas ocultas son los que permiten que una red detecte una característica o un patrón en los datos, y encuentre una relación funcional complicada entre las variables de entrada y salida. Sin la capa oculta un perceptrón es equivalente a un modelo de predicción lineal. Resultados teóricos [9] [16] muestran que una sola capa oculta es suficiente para que una RNA aproxime cualquier función no lineal compleja con cualquier exactitud deseada. Debido a esto, la mayoría de los autores utilizan una sola capa oculta. Sin embargo, una sola capa oculta puede requerir un número muy grande de nodos ocultos, lo cual no es deseable en cuanto a tiempo de entrenamiento. Entonces, una red con dos capas ocultas puede resultar más satisfactoria para algunos problemas [4]. Otros resultados muestran que una red puede necesitar más de dos capas ocultas para resolver ciertos problemas, incluyendo los de predicción [24] [8] [23].

Cheng et al. [7], Kimoto et al. [19], Yao y Tan [33] y Balkin y Ord [3] utilizan una sola capa de nodos ocultos para obtener sus resultados.

Ripley [26] afirma que el problema de la estructura de la red se suele tomar muy a la ligera, y dice que una "regla de dedo" es tomar como número de nodos ocultos un promedio de las entradas y las salidas. En general, las redes con menos nodos ocultos son preferibles en cuanto a que tienen más habilidad de generalización y menos problemas de sobre-ajuste. Sin embargo, una red con pocos nodos tiene menos potencia para modelar y aprender de los datos. La manera más común de determinar el número de nodos ocultos es por prueba y error. Zhang [35] dice que las redes con un número de nodos ocultos igual al número de nodos de entrada parece tener buenos resultados en predicción.

Balkin y Ord [3] proponen el siguiente algoritmo: se entrenan redes con número de nodos ocultos que van desde 1 hasta el número de entradas. Entonces se escoge la red que tenga el menor valor de GCV:

$$GCV(c) = \frac{1}{n} \times RSS/(1 - p \times \frac{c}{n})^2$$

Donde p es el número de parámetros estimados y c es un coeficiente de costo que ellos toman como 2.

Número de nodos de entrada El número de nodos de entrada corresponde al número de variables del vector de entrada que se usa para predecir los valores futuros. Para predicciones causales esto es muy fácil de elegir. En el caso de series de tiempo no es siempre claro el número de rezagos que deben incluirse. Hay quienes afirman que el número de nodos de entrada es simplemente el número de términos AR en un modelo Box-Jenkins para una serie de tiempo univariada, sin embargo, esto no es cierto, ya que para los procesos que tienen parte MA no hay parte AR (o es una serie infinita de términos) y además, los modelos de Box-Jenkins son lineales. Zhang [35] dice que no encuentra en la literatura resultados consistentes para determinar este parámetro.

Para la predicción de series de tiempo, Balkin y Ord [3] proponen el siguiente método: Los rezagos se determinan de acuerdo a la tabla siguiente:

Unidad de Tiempo	Anual	Trimestral	Mensual	Otro
Rezagos	1-4	1-6	1-15	1-6

Entonces se corren modelos de regresión lineal con las diferentes posibilidades. Dentro de los modelos se toman aquellos que tengan un estadístico F mayor a 4.0, y se escoge aquel que tenga el mayor número de rezagos. Ellos afirman que la evidencia empírica es que una RNA funciona mejor con más rezagos que un modelo AR.

Cheng et al. [7], quienes utilizan varias variables macroeconómicas en su esfuerzo para predecir la tasa de interés del bono a 30 años (de manera semanal), realizaron una encuesta entre especialistas para escoger estas variables de entrada, y posteriormente utilizaron de 20 a 26 datos de cada una de ellas, para un total de 460 a 598 nodos de entrada.

Número de nodos de salida El número de nodos de salida es muy fácil de determinar, dado que está directamente relacionado con el problema estudiado.

El único punto que no es tan claro, es cuando tenemos una predicción a más de un período. Esto puede hacerse de manera iterativa, tal como en el modelo de Box-Jenkins, utilizando las salidas de la red como entradas para la siguiente predicción, con lo cual se tiene un único nodo de salida. Otro método es el directo, dejando que la red realice directamente todas las predicciones al mismo tiempo. Zhang [34] muestra que el método directo es mejor que el iterado.

Las conexiones entre nodos Las conexiones entre los nodos de la red determinan el comportamiento de la misma. Para la mayor parte de las aplicaciones,

incluyendo la predicción, se utilizan redes totalmente conectadas, en las que todos los nodos de una capa están conectados con todos los nodos de la capa siguiente, excepto la capa de salida.

Tanto Kimoto et al. [19] como Cheng et al.[7], Yao y Tan [33] y Balkin y Ord [3] utilzan redes de tres capas totalmente conectadas.

3.2.2 La función de activación

La función de activación determina la relación entre las entradas y salidas de un nodo. En general, la función de activación introduce una no-linealidad que puede ser valiosa en muchas aplicaciones. La función más popular es la logística.

Klimasauskas [20] sugiere que las funciones de activación logística son útiles en problemas de clasificación, y sugiere el uso de funciones de tangente hiperbólica para los problemas de desviación de la media, tales como los de predicción.

Balkin y Ord [3], de acuerdo a los estudios de Faraway y Chatfield recomiendan que para la predicción de series de tiempo se utilice la función logística para los nodos de entrada y ocultos y la identidad para los nodos de salida. Kimoto et al. [19] utilizan una función sigmoidal.

En teoría, una red puede tener diferentes funciones de activación en los diferentes nodos, pero la convención es utilizar la misma.

3.2.3 Algoritmo de aprendizaje

El algoritmo más popular (y el que utilizaremos aquí), es el de back-propagation. Los "mejores" valores para el parámetro de tasa de aprendizaje η y el parámetro de momento β suelen escogerse a través de la experimentación. Tang y Fishwick [29] concluyen que estos parámteros son cruciales en el éxito de las RNA. Ellos encontraron que para cada serie de tiempo existe una RNA con los parámetros apropiados que funciona mucho mejor que las demás. Tang et. al [28] reportan que una tasa de aprendizaje alta es buena para datos menos complejos, y que una tasa baja junto con un momento alto debe usarse para series más complicadas.

Kimoto et al. [19] desarrollaron una técnica de aprendizaje basada en la de back-propagation, pero que es más rápida para aprender, y la llamaron aprendizaje suplementario (supplementary learning). En este sistema a cada nodo se le asigna un nivel de tolerancia, y sólo se actualiza su peso si el error correspondiente es mayor al nivel de tolerancia. Esto reduce el número de cálculos necesarios en el back-propagation común. También, conforme avanza el aprendizaje, los datos para los cuales se excede el nivel de tolerancia se reducen, lo cual también disminuye el tiempo de cálculo. Este método permite que la constante de aprendizaje vaya cambiando, dependiendo de la cantidad de datos de aprendizaje.

3.2.4 Normalización de los datos

En un problema de predicción de una serie de tiempo, la única forma razonable de estandarizar los datos es utilizar la normalización externa. En el caso de

predicción causal, se debe de usar el método de normalización dentro de un canal, ya que en general las variables de entrada son las variables independientes utilizadas para predecir la variable dependiente.

Balkin y Ord [3] notan que, dado que los parámetros de la RNA se estiman por mínimos cuadrados, esto sólo será eficiente si los términos de error son independientes y homoscedásticos. En el caso de tener series no estacionarias, esto no se cumplirá. Ellos proponen utilizar el logaritmo de la serie en el caso de que sea mayor la verosimilitud de la serie transformada que la de la serie original[3].

Kimoto et al. [19], para sus predicciones del índice japonés TOPIX suavizan los datos de entrada, que son otras variables financieras, utilizando la técnica de promedios móviles, para reducir el efecto de caminata aleatoria y quedarse con las tendencias. En algunos casos, utilizan los logaritmos de las series. Posteriormente normalizan los datos de entrada al intervalo [0,1].

3.2.5 Muestra de entrenamiento y muestra de prueba

Aunque no hay una solución general al problema de cómo separar los datos entre entrenamiento y prueba, hay varios factores, tales como las características del problema, el tipo de datos y la cantidad de datos disponible que deben considerarse. Es muy importante que ambos conjuntos sean representativos de la población. La mayoría de los autores escogen con una regla tipo 90% vs 10% ó 80% vs. 20%. Algunos autores como Gorr et al. [12] han utilizado métodos de bootstrap para elegir los subconjuntos. Granger [13] sugiere que para problemas de predicción no lineal al menos se utilicen el 20% de los datos para prueba. Nam y Schaefer [25] prueban el efecto del tamaño de la muestra de entrenamiento y muestran que mientras esta se incrementa, mejora el desempeño de la red. Kimoto et al. [19] utilizan 2/3 de los datos para entrenamiento y 1/3 para probar su sistema de predicción.

3.2.6 Medidas de desempeño

La medida más importante del desempeño de un RNA para predicción es, finalmente, qué tan bien predice. Sin embargo, no hay una medida de esto que sea universalmente aceptada por todos los autores. Una medida de exactitud normalmente se define en términos del error de predicción.

Yao y Tan [33] afirman que en el caso de predicción de series financieras lo que importa no es tanto minimizar los errores en una forma tradicional, sino maximizar las ganancias en una estrategia de inversión basada en las predicciones. Además, ellos afirman que los modelos que le dan igual peso a todos los datos son menos eficientes que aquellos que dan más importancia a datos

[3] Si definimos $w^2 = \frac{1}{n}\sum(x_i - \overline{x})^2$ y $s^2 = \frac{1}{n}\sum_i(\log(x_i) - \frac{1}{n}\sum_j \log(x_j))^2$ entonces la verosimilitud será mayor para la serie transformada cuando $\log(s^2) > (\log(w^2) + \frac{2}{n}\sum_i \log(x_i))$.

más recientes, y por lo tanto proponen una medida de desempeño que toma en cuenta las ganancias y "descuenta" la importancia de los datos en la red.

3.2.7 Desempeño relativo de las RNA

Hay muchos reportes inconsistentes en la literatura respecto al desempeño de la predicción con RNA contra los métodos tradicionales.

Algunos estudios empíricos muestran que las RNA son mejores para pronosticar datos mensuales y trimestrales que para datos anuales [18] [14] [15]. Esto puede deberse a que los datos más seguidos presentan mayores irregularidades, y las RNA son buenas para detectar los patrones subyacentes que pueden estar enmascarados por el ruido en los datos.

Tang et al. [28] y Tang y Fishwick [29] encontraron que las RNA funcionan mejor si se incrementa el plazo de predicción, y para series de memoria corta. También encontraron que las redes con más nodos de entrada funcionan mejor.

Denton [11] encontró que bajo condiciones ideales, donde se cumplen todos los supuestos de regresión, hay poca diferencia entre los resultados de una RNA y un modelo de regresión. Sin embargo, en situaciones menos ideales, tales como problemas de outliers, multicolinealidad o mala especificación del modelo, funcionan mejor las RNA.

La evidencia empírica que presentan Yao y Tan [33] muestra que las estrategias de inversión que utilizan el sistema de rendimientos y dependencia en el tiempo generan rendimientos excesivos hasta en un 27.6% anual y de un mínimo de 2.8%.

Kimoto et al. [19] en su predicción del TOPIX obtienen un rendimiento en exceso de 31% en 30 meses, utilizando un sistema de ventanas móviles en sus datos de aprendizaje.

Cheng et al. [7] lograron un rendimiento sobre su inversión de 17.27% contra 13.88% del índice de T-Bonds de Lehmannn Brothers en el mismo período. También resaltan que su técnica presenta un sistema de poco riesgo. Ellos detectan que su sistema de predicción tiene vida limitada, ya que funcionó bien durante cuatro años, pero el quinto año su exactitud decayó.

4 Resultados Empíricos

En esta sección presentamos los resultados de tres experimentos diferentes para predecir el tipo de cambio peso-dólar[4]. En el primero, se trabajó con datos diarios y se utilizó una red méramente "técnica", es decir, utilizando únicamente los datos de la misma serie. En el segundo, se utilizaron datos mensuales, y también se utilizó únicamente la serie como entrada de la red. En el último, se utilizaron los datos mensuales, y se añadieron los datos de inflación de Estados Unidos y México como entradas de la red.

Debido al gran número de parámetros y funciones con los cuales podemos experimentar para encontrar la red que mejor pronostique nuestros datos, de-

[4]Se utilizó el software Thinks Pro, de Logical Designs, elaborado por el Dr. Robert Trippi.

cidimos dejar la gran mayoría fijos, de acuerdo a las recomendaciones de Balkin y Ord [3] y algunos criterios personales, y básicamente dedicarnos a experimentar con el número de nodos de entrada y el número de nodos en la capa oculta. Dentro de cada experimento, encontraremos la red que mejor ajusta los datos de prueba, es decir, el mejor predictor, según el criterio de el error cuadrático medio. Posteriormente, usando métodos de econometría y series de tiempo, encontraremos el mejor modelo lineal. Posteriormente, compararemos ambos modelos en base a tres medidas en cuanto a su ajuste a los datos de prueba. La primera medida es simplemente la suma de cuadrados del error (SSE). La segunda medida es la raíz del error cuadrático medio (RECM), que es la medida que toma el software, y que es equivalente (en cuanto a jerarquía) a la de la SSE. La tercera se trata de una medida de "eficacia", que se refiere al número de veces que el modelo predice correctamente un incremento (o un decremento) en el tipo de cambio. Esta medida es importante porque finalmente la dirección de los cambios es lo que decide la estrategia de inversión o cobertura de un inversionista.

4.1 Experimento con datos diarios

Para este experimento se tomaron 795 datos diarios, desde el 8 de febrero de 1999 hasta el 29 de abril de 2002, del tipo de cambio peso-dólar. Se utilizaron 735 datos para el entrenamiento (o ajuste del modelo) y 60 para probar los resultados.

Como preparación de los datos de entrada, se utilizó el criterio del cociente de verosimilitud para decidir si era necesario sacar logaritmo de los datos, lo cual resultó no ser necesario. Se realizó un análisis de raíz unitaria (utilizando una prueba de Dickey-Fuller aumentada) y se llegó a la conclusión de que esta serie no tiene raíz unitaria.

Para decidir cuántos datos utilizar de entrada a la red, de acuerdo con la técnica propuesta por Balkin y Ord [3], se analizaron los modelos de series de tiempo tipo AR, con rezagos que van desde 1 a 15. Se realizó una prueba F para cada uno de estos modelos:

Rezagos	1	2	3	4	5	6	7	
Prueba F	5.76	6.26	3.26	2.42	1.94	2.12	1.87	
Rezagos	8	9	10	11	12	13	14	15
Prueba F	1.6	1.49	1.35	1.32	1.35	1.26	1.28	1.22

De aquí observamos que sólo están por arriba de 2.0[5] los modelos con 6 ó menos rezagos (excepto el de 5). En base a esto, realizamos corridas con redes neuronales con 1 hasta 6 neuronas de entrada. El criterio de información de Akaike, que es el que utilizan De Groot y Wurz [10] da como resultado un modelo con 2 rezagos. De acuerdo a Balkin y Ord [3], se entrenaron redes con una sola capa oculta y con el número de nodos ocultos que van desde 1 hasta el número de entradas.

[5] En realidad, los autores recomiendan usar los modelos cuya F sea mayor a 4.0, pero dado que en este caso casi todos tienen una F menor, se tomó como límite el 2.0.

Se utilizó el preproceso de min-max para normalizar los datos, y como función de transferencia para los nodos ocultos se utilizó una función sigmoidal y para los de salida una lineal, que es lo que recomiendan Balkin y Ord [3]. Se utilizó una η de 0.01 y una β de 0, que son los defaults del programa. Cada red se dejó correr 5,000 iteraciones, ya que se observó que la convergencia se vuelve muy lenta después de este número de iteraciones.

Se obtuvieron los siguientes resultados:

Entradas/Nodos	1	2	3	4	5	6
1	.047					ETRN
	.028					ETST
2	.046	.048				
	.029	.028				
3	.048	.047	.048			
	.029	.029	.029			
4	.046	.047	.049	.049		
	.030	.029	.029	.030		
5	.048	.047	.048	.049	.049	
	.028	.029	.029	.030	.030	
6	.048	.047	.048	.048	.048	.050
	.028	.029	.028	.030	.029	.028

La tabla presenta en la parte de arriba la raíz del error cuadrático medio[6] para los datos de entrenamiento y en la parte inferior para los datos de prueba.

Debido a la gran cantidad de datos con que contamos, consideramos que no tenemos un problema de grados de libertad, y por consiguiente analizamos el ajuste de las redes únicamente en función de los errores, y no utilizamos ningún criterio del estilo del GCV.

Puede observarse que no hay demasiadas diferencias entre las diferentes redes. Para motivos de comparación, utilizaremos la red de 6 entradas y un nodo oculto, que es una de las que mejor predice. Observando los pesos de esta red, podemos ver que el último rezago tiene un peso del 85%, y el umbral (que equivale a un intercepto) tiene casi todo el restante 15%, por lo que este modelo es casi como un AR(1).

Ajustando el mejor modelo tipo AR[7] para los datos de entrenamiento encontramos que se trata de un AR(2)[8]. Comparando el "mejor" predictor de redes neuronales con el "mejor" modelo lineal, tenemos la siguiente tabla:

[6] $\sqrt{\frac{\sum(y_i-\hat{y}_i)^2}{n}}$

[7] De hecho, el modelo que mejor ajusta los datos es un AR(2) con GARCH(2,1), pero no utilizamos técnicas GARCH en este trabajo.

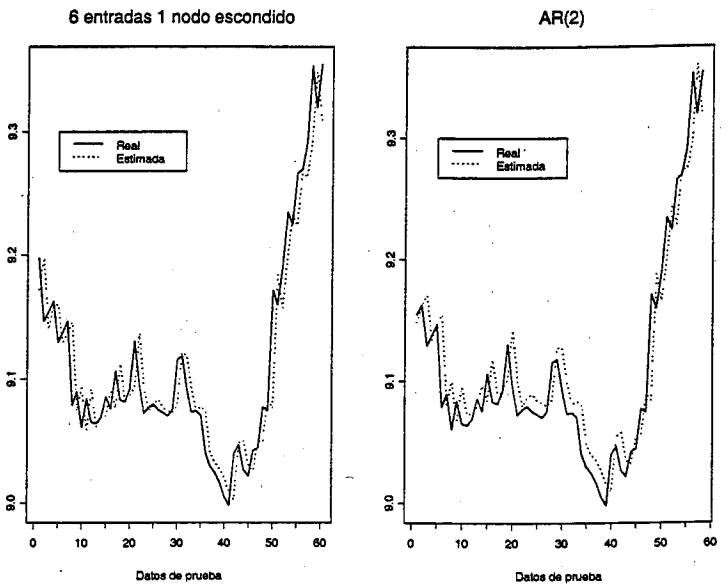
[8] El modelo es:

$tc_t = 0.283799 + 1.072743 * tc_{t-1} - 0.102976 * tc_{t-2}$

Modelo	SSE	RECM	Eficacia
Redes 6 entradas, 1 nodo ocultos	.047	.028	47%
Modelo lineal AR(2)	.045	.027	51%

Podemos observar que, aunque el modelo lineal es ligeramente mejor en ambas medidas, esto no es concluyente. De hecho, ambos modelos tienen una eficacia muy baja, y no son útiles para decidir en una estrategia de inversión.

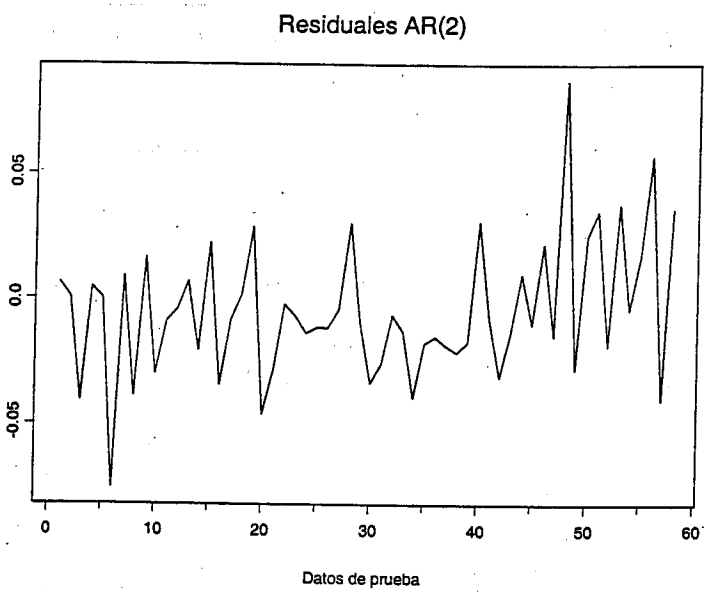
En las siguientes gráficas podemos comparar visualmente el ajuste de ambos modelos:



Se puede observar que en ambos casos la predicción va un poco retrasada con respecto a los datos reales, es decir, los "persigue". En conclusión, no parece haber una diferencia significativa entre ambas técnicas para este caso, siendo tal vez ligeramente mejor el modelo de series de tiempo.

4.1.1 Análisis de residuales del modelo AR(2)

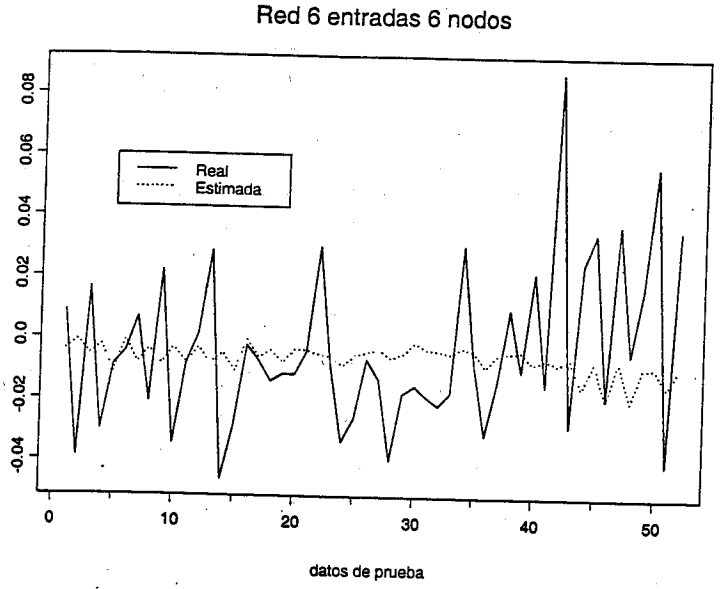
Los residuales del modelo AR(2) presentado arriba se pueden observar en la siguiente gráfica:



Dado que los residuales del modelo AR(2) podrían presentar aún una estructura, intentamos ajustar una red para estos datos, obteniendo los siguientes resultados:

2 entradas 1 nodo	6 entradas 6 nodos
.045	.046
.027	.027

Como puede verse, aumentar el número de nodos de entrada y el número de nodos en la capa oculta no produce un mejor ajuste. En la gráfica de resultados podemos observar que la red capta muy poca de la estructura "escarpada" de los datos:



Posteriormente, corrimos la red con 6 entradas y 12 nodos ocultos, obteniendo los mismos resultados en cuanto ajuste del ECM, y una gráfica muy parecida a la anterior, es decir, no se nota ningún incremento en la flexibilidad de la red. Posteriormente volvimos a correr esta red, pero con una α de .001, de acuerdo a los resultados de Tang et. al [28], que proponen que se utilice una α pequeña para datos complicados. Nuevamente, los datos de ECM son los mismos, y la gráfica no muestra un mejor ajuste a los datos.

Debido a que los resultados de intentar ajustar una red a los residuales no son muy prometedores, no lo intentamos en los siguientes experimentos, ya que sabemos que la estructura será muy parecida.

4.2 Experimento con datos mensuales

Para este experimento, se utilizaron 87 datos mensuales del tipo de cambio peso-dólar desde enero de 1995 hasta marzo de 2002. No se utilizaron datos anteriores a 1995 debido a que antes de esta fecha existía una banda para el tipo de cambio fijada por el gobierno Mexicano, y consideramos que esto altera la estructura de los datos. Se usan los primeros 66 para entrenamiento y los último 21 para probar los resultados.

Al igual que con los datos diarios, se realizó una prueba de verosimilitud para decidir si es necesario transformar los datos a logaritmos, llegando a que no es así. Sin embargo la prueba de Dickey Fuller aumentada para estos datos indica que poseen una raíz unitaria, por lo que los modelos AR con los que se trabajó se estimaron usando las primeras diferencias de la serie. Se realizaron pruebas F para modelos AR con 1 hasta 15 rezagos, con los siguientes resultados:

Rezagos	1	2	3	4	5	6	7
Prueba F	0.15	4.02	2.39	1.51	1.44	1.93	1.79

Rezagos	8	9	10	11	12	13	14	15
Prueba F	2.05	1.069	0.94	0.76	0.68	0.71	0.68	0.71

Ya que estos valores son bajos, se decidió tomar desde 2 hasta 6 rezagos (6 meses) de información para las predicciones. El criterio de información de Akaike, vuelve a dar como resultado un modelo con 2 rezagos. Se entrenaron redes con una sola capa oculta y con el número de nodos ocultos que van desde 1 hasta el número de entradas y se utilizó el preproceso de min-max para normalizar los datos, y como función de transferencia para los nodos ocultos se utilizó una función sigmoidal y para los de salida una lineal Como función de error se utiliza el ECM. Se utilizó una η de 0.01 y una β de 0. Cada red se dejó correr 5,000 iteraciones, y luego hasta 10,000 iteraciones.

Se obtuvieron los siguientes resultados hasta 5,000 iteraciones:

Entradas/Nodos	1	2	3	4	5	6
2	.20862	.20831				ETRN
	.17328	.17188				ETST
3	.20622	.20595	.20308			
	.17364	.17344	.17845			
4	.20216	.19762	.19062	.19048		
	.17644	.18408	.19288	.19207		
5	.20289	.19661	.19237	.18014	.18048	
	.17425	.18948	.1866	.19476	.19679	
6	.20268	.18659	.18429	.18589	.18067	.18323
	.17412	.19656	.1915	.20063	.20305	.19743

Para 10,000 iteraciones se obtuvieron los siguientes resultados:

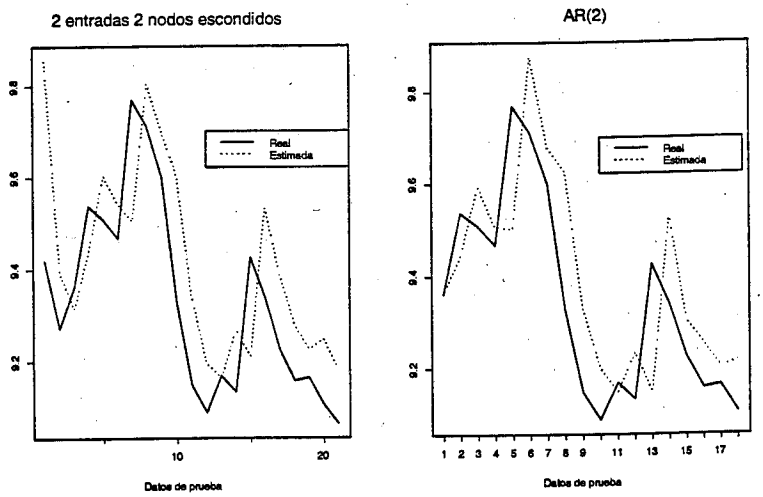
Entradas/Nodos	1	2	3	4	5	6
2	.20848	.20826				ETRN
	.17289	.17235				ETST
3	.20587	.20572	.20073			
	.17358	.17356	.18185			
4	.20201	.19476	.18858	.18781		
	.17769	.18982	.19577	.19507		
5	.20203	.19021	.18245	.17575	.17624	
	.17772	.19578	.19469	.20018	.20113	
6	.20193	.18225	.17512	.18269	.1749	.17786
	.17796	.20564	.20933	.2097	.20949	.20497

De estas tablas podemos concluir lo siguiente: con un mayor número de entradas y un mayor número de nodos, mejora el ajuste de la red a los datos de entrenamiento, pero al mismo tiempo disminuye el ajuste a los datos de prueba,

lo cual es una indicación de que se está presentando un sobre entrenamiento. Así mismo, al incrementar el número de iteraciones de cada red, mejora el ajuste a los datos de entrenamiento, pero empeora la predicción, mostrando también síntomas de sobre entrenamiento.

En ambos cuadros podemos observar que el mejor predictor es la red con 2 entradas y 2 nodos ocultos, mientras que la red que mejor ajusta cambia, siendo la de 5 entradas con 4 nodos ocultos la mejor para 5,000 iteraciones y la de 6 entradas con 5 nodos ocultos para 10,000 iteraciones. Como se mencionó en el párrafo anterior, es mejor el ajuste a los datos con 10,000 iteraciones, pero es mejor la predicción usando solamente 5,000 iteraciones. El mejor modelo AR para los mismos datos, que se trata de un AR(2) para los datos en diferencias, o un ARIMA(2,1,0) [9], tiene una SSE de 0.46, cuando se trata de ajustar los datos en niveles. La SSE es de 0.41 al ajustar los datos en diferencias.

Modelo	SSE	RECM	Eficacia
Red 2 entradas, 2 nodos ocultos	0.62	.176	47%
Modelo lineal AR(2)	0.46	.143	47%



Nuevamente puede observarse en las gráficas que la predicción "persigue" a los datos reales. Aunque visualmente parece que la red ajusta un poco mejor a los datos, analizando la SSE y la eficacia, esto no es concluyente.

4.3 Predicción con inflación

En esta sección nuevamente utilizamos datos mensuales de 1995 a la fecha. Para este experimento se utilizó la relación de equilibrio que debe de existir entre la

[9] El modelo es: $dtc_t = 0.037 + 0.206 * dtc_{t-1} - 0.191 * dtc_{t-2}$

inflación de dos países y el tipo de cambio entre sus monedas[5]:

$$\frac{E(1+i_{usa})}{E(1+i_{mex})} = \frac{E(tc)}{tc}$$

Por simplificación, en lugar del valor esperado de la inflación, tomamos el dato del mes anterior. Cabe resaltar que esto puede ser causa de que el modelo pierda validez o sea menos eficaz.

Utilizando como entrada el cociente de las inflaciones rezagadas más uno y un rezago del tipo de cambio, y utilizando los mismos parámetros y funciones de activación y de error de las secciones anteriores, entrenamos la red con uno y dos nodos en la capa oculta.

Los resultados son los siguientes:

Nodos/Iteraciones	5,000	10,000	15,000	20,000	25,000
1	.24312	.24313	.24313		ETRN
	.21847	.21843	.21843		ETST
2	.24129	.23155	.22928	.22819	.22789
	.1805	.2744	.15266	.18566	.22476

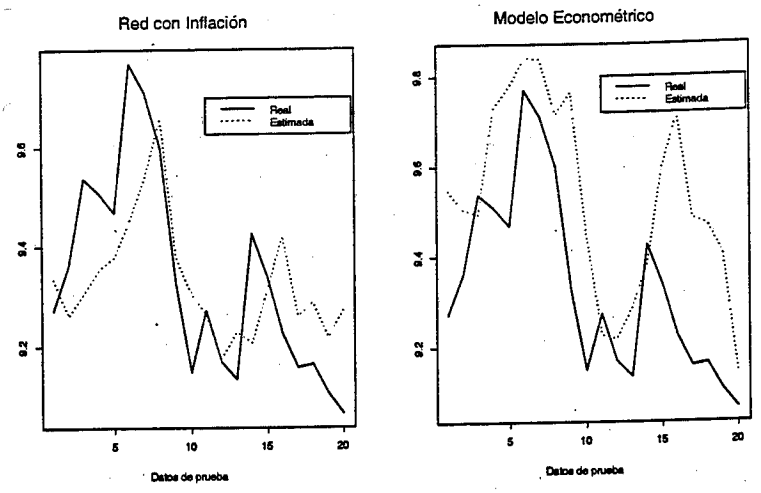
Nuevamente el número superior muestra el error de de entrenamiento y el inferior el error de prueba.

En este caso podemos observar que cuando tenemos un solo nodo en la capa oculta, la red se estabiliza después de las 10,000 iteraciones. En el caso de la red con dos neuronas ocultas continuamos teniendo disminución del error de entrenamiento hasta las 25,000 iteraciones, sin embargo el error de predicción comienza a incrementarse después de las 15,000 iteraciones, mostrando sobre entrenamiento. Entonces, el mejor modelo de predicción para estos datos se trata del de dos nodos en la capa oculta, entrenado con 15,000 iteraciones. El modelo econométrico equivalente[10] tiene una SSE de 1.1863.

Modelo	SSE	RECM	Eficacia
Red 2 nodos ocultos, 15,000 iteraciones	0.43	.147	42%
Modelo lineal	1.19	.243	47%

Las siguientes gráficas muestran los resultados de la predicción con ambos métodos, comparando contra los datos reales:

[10] $tc_t = 0.35 + 0.23 * \frac{1+i_{usa}}{1+i_{mex}} + 0.95 * tc_{t-1}$



En este caso el modelo de redes neuronales es un mejor predictor para los datos desde el punto de vista de la SSE. Esto no es extraño, ya que la forma misma de la relación de equilibrio es no lineal, por lo que es de esperarse que funcione mejor un modelo no lineal. Sin embargo, la eficacia de ambos modelos es muy baja, mostrando que ninguno de los dos modelos es realmente útil.

4.4 Otros intentos

Además de los resultados presentado anteriormente, intentamos también ajustar las siguientes redes:

- una red con 6 entradas y con un parámetro de weight decay de 0.9 para los datos diarios. Esta red resultó en una SSE de 0.45, y una eficacia de 51%, lo cual es una ligera mejoría sobre la red presentada anteriormente. Sin embargo, aún presenta el efecto de "perseguir" los datos.
- una red para predecir las diferencias del tipo de cambio (diario), pero, al igual que con los residuales del modelo AR(2) no logramos captar la estructura de esta serie. En este caso pudimos observar que a mayor número de nodos en la capa oculta mejora el ajuste a los datos de entrenamiento, pero no la predicción.
- una red para predecir subidas (1) y bajadas (-1) para el tipo de cambio, pero nunca logramos que convergiera, parecía ir moviendo los pesos de manera aleatoria.
- Se cambió el parámetro de aprendizaje η a 0.1 para los datos diarios, teniendo como resultado que empeoró el ajuste.

- También para los datos diarios, se corrió una red con parámetro de momento $\beta = .5$, obteniendo un resultado totalmente equivalente a cuando no se utiliza este parámetro.

5 Conclusiones

En base a los experimentos realizados, que se basaron principalmente en las recomendaciones de Balkin y Ord[3], los resultados no son del todo satisfactorios. En ninguno de los casos podemos decir de manera contundente que las redes presentan una mejor predicción que los modelos lineales, excepto tal vez en el tercer caso si sólo nos preocupamos por la SSE.

En los casos en que se utilizan únicamente rezagos de las mismas series, los predictores de los modelos van ligeramente retrasados con respecto a los datos reales, lo cual, aunque puede presentar errores pequeños, no es útil para los inversionistas o empresarios que requieren esta información, ya que lo importante para ellos es saber si el tipo de cambio subirá o bajará para poder tomar medidas al respecto y estos modelos tienden a predecir al revés los incrementos y decrementos, ya que por fenómenos técnicos las subidas tienden a ser seguidas por una bajada y viceversa. El último modelo no presenta este fenómeno de retraso, pero tampoco muestra una mayor eficacia en las predicciones.

Podemos observar que el mejor predictor (desde el punto de vista de minimizar la SSE) para los datos mensuales se trata del modelo de red que utiliza las inflaciones de E.U. y México como entradas, lo cual indica que estos datos sí presentan una información valiosa para el pronóstico del tipo de cambio. Sin embargo ningún modelo es capaz de ganarle a un volado desde el punto de vista de la eficacia.

Aparentemente la estructura de estos datos, sobre todo de los datos diarios, es demasiado compleja para el tipo de redes que utilizamos, y posiblemente fuera más efectivo suavizar los datos de alguna manera antes de entrenar la red. Cabe resaltar que ninguno de los artículos que analizamos trabaja con datos diarios, y casi siempre trabajan con datos trimestrales o con menos periodicidad.

Es claro que, así como no utilizamos los modelos óptimos de series de tiempo (probablemente en estos casos deberíamos haber utilizado modelos tipo GARCH), las redes utilizadas pueden tener muchas mejorías. Algunos caminos a seguir para intentar mejorar los resultados aquí obtenidos serían: experimentar con los valores de η y β; en el tercer modelo, utilizar previamente dos redes que hagan una predicción de las inflaciones en E.U. y México para entonces sí utilizar el valor esperado de los datos, y no los datos en sí; utilizar otras variables financieras como entradas, tal como las tasas de interés; experimentar con otras variables de la red, tal como las funciones de transferencia o la función de error, en particular sería interesante utilizar una función de error basada en las pérdidas y ganacias de una estrategia de inversión derivada del modelo. También, se podría experimentar más con la técnica de "weight decay", que nos resultó en pequeñas mejorías.

Estos resultados parecen ser un punto a favor de la HME, ya que no logramos

pronosticar el tipo de cambio de manera consistente con los datos que tenemos.

Referencias

[1] Azoff, E.M., 1994. Neural Network Time Series Forecasting of Financial Markets. John Wiley and Sons.

[2] Anderson, J.A., Pellionisz, A. and Rosenfeld, E. (Eds.) 1990, Neurocomputing 2: Directions for Research. MIT Press.

[3] Balkin, S.D., Ord, J.K., 2000. Automatic neural network modeling for univariate time series. International Journal of Forecasting 16, 509-515.

[4] Barron, A.R., 1994. A comment on "Neural Networks: a review from a statistical perspective". Statistical Science 9 (1), 33-35.

[5] Brealey, R.A., Myers, S.C.1991. Principles of Corporate Finance. Fourth Edition. McGraw Hill.

[6] Castillo, E., Cobo, A. Gutiérres, J.M., Pruneda, R.E., 1998. Introducción a las redes funcionales con aplicaciones. Un nuevo paradigma neuronal. Ed. Paraninfo.

[7] Cheng, W., Wagner, L., Lin, C.H., 1996. Forecasting the 30 year U.S. treasury bond with a system of neural networks. Neuro Ve$t journal.

[8] Cybenko, G., 1988. Continuos Valued Neural Networks with Two Hidden Layers are Sufficient. Technical Report. Tuft University.

[9] Cybenko, G., 1989. Approximation by superposition of a sigmoidal function. Mathematical Control Signals Systems 2. 303-314.

[10] De Groot, C., Wurz, D., 1991. Anlysis of univariate time series with connectionist nets: a case study of two classical examples. Neurocomputing 3, 177-192.

[11] Denton, J.W., 1995. How good are neural networks for causal forecasting? The Journal of Business Forecasting 14 (2). Summer, 17-20.

[12] Gorr, W.L., Nagin, D., Szcypula, J., 1994. Comparative study of artificial neural networks and statistical models for predicting student grade point averages. International Journal of Forecasting 10. 17-34.

[13] Granger, C.W.J., Terasvirta, T., 1993. Modelling nonlinear economic relationships. Oxford University Press, Oxford.

[14] Hill, T., Marquez, L., O'Connor, M., Remus, W., 1994. Artificial neural networks for forecasting and decision making. International Journal of Forecasting 10, 5-15.

[15] Hill, T., Marquez, L., O'Connor, M., Remus, W., 1996. Neural network models for time series forecasts. Management Sciences 42 (7), 1082-1092.

[16] Hornik, K. Stinchcombe, M., White, H., 1989. Multilayer feed forward networks are universal approximators. Neural Networks 2, 359-366.

[17] Hu, M.:J.C.. 1964. Application of the adaline system to weather forecasting. Master Thesis. Technical Report 6775-1. Stanford Electronic Laboratories, Stanford, CA., June.

[18] Kang, S. 1991. An investigation of the use of feedforward neural networks for forecasting. Ph. D. Thesis. Kent State University.

[19] Kimoto, T., Asakawa, K., Yoda, M., Takeoka, M., 1990. Stock market prediction system with modular neural networks. Proceedings of the IEEE international joint conference on neural networks. San Diego CA, 2. pp 11-16.

[20] Klimasauskas, C.C., 1991. Applying neural networks. Part 3: training a neural network. PC-AI. May/June, 20-24.

[21] Lachtermacher, G., Fuller, J.D., 1995. Backpropagation in time series forecasting. Journal of Forecasting 14. 381-393.

[22] Lapedes, A., Farber, R.,1987. Nonlinear signal processing using neural networks: prediction and system modelling. Technical Report LA-UR-87-2662. Los Alamos National Laboratory. Los Alamos, NM.

[23] Lapedes, A., Farber, R., 1988. How neural nets work. In Anderson, D.., (Ed), Neural information processing systems. American Institute of Physics, New York, pp 442-456.

[24] Lippmann, R.P., 1987. An introduction to computing with neural nets, IEEE ASSP Magaine, April, 4-22.

[25] Nam, K., Schaefer, T., 1995. Forecasting international airline passenger traffic using neural networks. Logistics and Transportation 31 (3), 239-251.

[26] Ripley, B.D. Statistical aspects of neural networks. Chapter 2 of Networks and Chaos - Statistical and Probabilistic Aspects. Barndorff-Nielsen, O.E., Jensen, J.L., Kendall, W.S. (Eds.) Chapman Hall.1993.

[27] Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by backpropagation errors. Nature 323 (6188) 533-536.

[28] Tang, Z., Almeida, C., Fishwick, P.A., 1991. Time series forecasting using neural networks vs Box-Jenkins methodology. Simulation 57 (5). 303-310.

[29] Tang, Z., Fishwick, P.A., 1993. Feedforward neural nets as models for time series forecasting. ORSA Journal of Computing 5 (4). 374-385.

[30] Weigend, A.S., Huberman, B.A., Rumelhart, D.E., 1990. Predicting the future: A coneccionist approach. International Journal of Neural Systems 1, 193-209.

[31] Weigend, A.S., Huberman, B.A., Rumelhart, D.E., 1992. Predicting sunspots and exchange rates with connecciontst networks. In: Casdagli, M., Eubank, S. (Eds.) Nonlinear modelling and forecasting. Addison Wesley, Redwood city, CA. pp 395-432.

[32] Weigend, A.S., Huberman, B.A., Rumelhart, D.E., 1991. Generalization by weight elimination with application to forecasting. Advances in Neural Information Processing Systems 3, 875-882.

[33] Yao, J., Tan, C.L. 2000. Time dependent directional profit model for financial time series forecasting. IEEE.

[34] Zhang, X., 1994. Time series analysis and prediction by neural networks. Optimization Methods and Software 4, 151-170.

[35] Zhang, G., Patuwo, B.E., Hu, M.Y., 1998. Forecasting with artificial neural networks: the state of the art. International Journal of Forecasting 14, 35-62.