UNIVERSIDAD POLITÉCNICA DE MADRID

FACULTAD DE INFORMÁTICA

Nuevos modelos de redes de neuronas artificiales para simulación y control de sistemas dinámicos

TESIS DOCTORAL

Autora: Inés María Galván León
Licenciada en Matemáticas

1997

DEPARTAMENTO DE INTELIGENCIA ARTIFICIAL

FACULTAD DE INFORMÁTICA

Nuevos modelos de redes de neuronas artificiales para simulación y control de sistemas dinámicos

Autora: Inés María Galván León.
Licenciada en Matemáticas

Directores: Dolores Barrios Rolanía
Doctora en Informática

Pedro Isasi Viñuela
Doctor en Informática

1997

Tribunal nombrado por el Mgfco. y Excmo. Sr. Rector de la Universidad Politécnica de Madrid, el día _____de _____ de 19__ .

Presidente D. _____ .

Vocal D. _____ .

Vocal D. _____ .

Vocal D. _____ .

Secretario D. _____.

Realizado el acto de defensa y lectura de la Tesis el día _____de _____ de 19__

en _____.

Calificación: _____.

EL PRESIDENTE LOS VOCALES

EL SECRETARIO

A mis padres, Inés y Julián,
y a mis hermanas, Mª José y Toni

Agradecimientos

A Dª. Mª Dolores Barrios Rolanía y D. Pedro Isasi Viñuela, directores de esta tesis, cuyos consejos, sugerencias y orientaciones han permitido desarrollar y dar configuración a este trabajo.

A D. Juan Ríos Carrión, cuyo interés y disponibilidad han ayudado y favorecido su presentación.

A D. Darío Maravall Gómez-Allende, por su juicio objetivo y su opinión desinteresada acerca del contenido de estas páginas.

A D. Arturo Ribagorda Garnacho, por la confianza y el apoyo que siempre me ha mostrado.

A D. José Manuel Zaldívar Comenges, por la ayuda que me proporcionó durante mi estancia en Ispra, no sólo en cuestiones profesionales, sino también personales.

A Pedro Pimenta. Él me ha orientado y guiado por el mundo de los procesos dinámicos, de su modelización y control. Las numerosas y a la par ricas conversaciones que mantuvimos, no sólo me dieron la luz necesaria para atisbar a ver las ideas aquí presentadas, sino también el apoyo para llevarlas a cabo. Desde estas páginas le expreso mi agradecimiento a su amistad y a su inestimable ayuda.

A mis compañeros de trabajo, H. Hernández, F. Strozzi, M. A. Alós, J. A. Feliu, J. Riego, H. Nieman, J. Ligthart, H. Schönherr, M. Franzetti, así como al director del grupo, D. Carlo Bassani. He de destacar a H. Hernández, ya que sus ideas sirvieron para la puesta en marcha de esta tesis. Y a H. Nieman, cuya colaboración me ha sido de gran ayuda para la realización de experimentos con el reactor químico.

De manera especial quiero expresar también mi agradecimiento a mis padres, Inés y Julián, y a mis hermanas, Mª José y Toni, por el apoyo que siempre me han prestado. A Juan Carlos, por su cariño. Y a Mª Josefa, por ese entrañable magisterio que aún vive en mi recuerdo.

Índice

Introducción

Uno de los principales problemas en el campo de los procesos dinámicos es el *control* de su comportamiento, control que es necesario para que la evolución del proceso se lleve a cabo en una dirección y con una cierta dinámica deseada. Esto puede traducirse en mantener una condición estable, seguir una trayectoria especificada o alcanzar una determinada meta. Para ello es necesario incluir un sistema de control, el cual influye en la dinámica del proceso, regulando su comportamiento.

La teoría de sistemas ha sido ampliamente desarrollada en las últimas cinco décadas, por lo que en la actualidad existe una gran variedad de técnicas de control. Dichas técnicas, en su mayoría, hacen uso de una representación matemática del proceso, también llamada *modelo* del proceso, permitiendo así la construcción de sistemas de control cuya actuación está basada en la propia naturaleza del proceso dinámico.

A pesar de que la gran mayoría de los procesos reales son no lineales, la mayor parte de las técnicas de control se han desarrollado para procesos dinámicos lineales. En los últimos años ha habido, sin embargo, un interés creciente en incorporar técnicas no lineales, tanto para tratar el problema del control de procesos dinámicos, como para afrontar el problema de su modelización.

En el desarrollo de técnicas de control no lineales, las redes de neuronas artificiales ocupan un lugar importante. Características tales como la naturaleza adaptativa y la capacidad para aproximar y aprender relaciones complejas no lineales a partir de un conjunto de ejemplos o patrones, hacen que sean buenas candidatas para tratar el problema de la modelización y el control de procesos dinámicos no lineales. Como

consecuencia, en los últimos diez años aproximadamente ha surgido una gran variedad de trabajos, en los que se utilizan diferentes arquitecturas y métodos de aprendizaje para la construcción de modelos y sistemas de control no lineales (ASMC, 92).

Los modelos estudiados en este trabajo pertenecen a la categoría de modelos NARMA, los cuales son una extensión no lineal de los modelos ARMA (Auto-Regressive Moving Average) y se caracterizan porque describen el comportamiento dinámico del proceso a partir de las variables observables o medibles. Estos modelos son generalmente más fáciles de construir y tratar que los modelos físicos o modelos construidos a partir de leyes físicas que rigen el comportamiento dinámico del proceso, los cuales suelen ser modelos complicados, ya que intervienen un gran número de ecuaciones diferenciales, su construcción es laboriosa y requiere normalmente de una gran cantidad de tiempo y experimentos.

Los modelos NARMA neuronales más populares son los llamados *modelos de identificación en serie-paralelo*, los cuales consisten en aproximar la relación no lineal utilizando el perceptron multicapa o una red de base radial, también conocidas como redes de neuronas estáticas, ya que la representación o el procesamiento de información temporal no es una propiedad intrínseca a ellas. En este caso, los patrones de entrada a la red son vectores que contienen las variables de entrada y salida del proceso, así como una historia de dichas variables, de modo que pueda ser representada la información temporal.

Un modelo apropiado de un proceso tiene que poseer la propiedad de saber actuar como simulador del proceso. Esto significa que, dado el estado inicial del proceso y la variable de entrada, el modelo tiene que aproximar o predecir la dinámica de dicho proceso durante un cierto intervalo de tiempo, utilizando únicamente dicha información.

Los modelos de identificación en serie-paralelo presentan el inconveniente de que no pueden utilizarse para simular la dinámica del proceso, ya que para predecir la salida actual del proceso es necesario conocer una historia de dicha variable, datos que no están disponibles cuando se simula el comportamiento del proceso dinámico.

Algunos autores, (NAPA, 90), han propuesto reemplazar estos valores medibles en el modelo de identificación en serie-paralelo por los valores predichos por la red en instantes anteriores de tiempo, cuando sea necesario disponer de un simulador, obteniendo los llamados *modelos de identificación en paralelo*. Sin embargo, en los

estudios realizados en este trabajo se observa que los modelos en paralelo propuestos por estos autores, no siempre proporcionan aproximaciones adecuadas y convenientes del proceso dinámico, e incluso pueden existir situaciones en las que la capacidad de representación de estos modelos quede prácticamente anulada.

Con respecto a las estrategias de control, en esta tesis se analizan dos sistemas diferentes de control no lineal, denominados *control inverso* y *control predictivo*. Ambas estrategias han sido ya estudiadas por diferentes autores, (JORD, 89), (NAPA, 90), (WIMO, 91), entre otros, aunque siguen aún presentando en la actualidad una serie de limitaciones e inconvenientes, sobre todo en lo referente a aplicaciones en tiempo real, las cuales se especifican brevemente a continuación.

Las estrategias de control inverso, básicamente, consisten en entrenar una red de neuronas para que aprenda la dinámica inversa del proceso. Cuando se plantea un esquema de control inverso se distinguen dos formas diferentes de aprendizaje del controlador, que se denominan aprendizaje generalizado y aprendizaje especializado, respectivamente. La finalidad del aprendizaje generalizado es que la red aprenda la dinámica inversa del proceso, en su totalidad, a partir de un conjunto de datos representativos de dicha dinámica. Mediante el aprendizaje especializado, sin embargo, la red aprende la inversa local del proceso en la región de interés, utilizando la diferencia entre la salida actual del proceso y la salida deseada para adaptar los pesos del controlador. En este caso no es necesario disponer de un conjunto de patrones salida-entrada del proceso, sino que los datos para el aprendizaje proceden de la evolución directa de dicho proceso.

Los esquemas de control inverso con aprendizaje generalizado son sistemas de control off-line, es decir, el aprendizaje de la red tiene que realizarse antes de su actuación como controlador del proceso dinámico real. Por tanto, el éxito del controlador depende en gran medida de los datos disponibles y de la capacidad de generalización de la red, factores que impiden, en la mayor parte de los casos, obtener un control eficiente del proceso dinámico real. Por otra parte, no es un esquema de control aplicable a cualquier proceso dinámico, ya que es necesario que la dinámica inversa de dicho proceso esté bien definida.

El esquema de control inverso con aprendizaje especializado se considera, sin embargo, un esquema de control on-line o en tiempo real, por lo que el aprendizaje de la red de neuronas se lleva a cabo mientras controla el proceso. Es, por tanto, una aproximación más conveniente y exacta, pues la red aprende a conseguir un objetivo de control, el dictado por las necesidades de cada momento. No obstante, y debido, precisamente, a que se trata de un esquema de control on-line, la inicialización de los pesos del controlador tiene una repercusión importante en los resultados de control; una inicialización no adecuada del controlador inverso cuando se dispone a realizar el aprendizaje especializado utilizando el proceso real, podría provocar situaciones de inestabilidad o incluso de “no control” en el proceso, lo cual no es admisible en aplicaciones en tiempo real.

Debido a que la señal de control en un cierto instante de tiempo no sólo influye en la respuesta del proceso en el instante siguiente, sino también en un cierto futuro, es interesante utilizar esquemas de control que hagan uso de cierta información en el futuro para calcular la acción de control en el presente, y poder así obtener controladores más eficientes. Estos sistemas de control son los llamados controladores predictivos. Las estrategias de control predictivo consisten en calcular, en cada instante de tiempo, la acción de control que minimice la diferencia entre la salida de un modelo del proceso y la salida deseada durante un cierto intervalo de tiempo en el futuro, cuya longitud viene dada por el horizonte de predicción. Los modelos más empleados en estos sistemas de control han sido los clásicos modelos ARMA lineales (WIMO, 91), ya que, debido precisamente a su estructura lineal, el problema de optimización que engloba estas estrategias es lineal. Sin embargo, la capacidad de representación de los modelos ARMA para procesos dinámicos no lineales es bastante limitada. Por otra parte, la utilización de modelos no lineales para predecir el comportamiento del proceso en el futuro involucra la resolución de un problema de optimización no lineal en cada instante de tiempo, lo cual es un serio inconveniente cuando se trata de aplicaciones de control en tiempo real, ya que normalmente requieren de un alto esfuerzo computacional. Generalmente, este hecho hace que las estrategias de control predictivo que hacen uso de un modelo no lineal sean impracticables en aplicaciones reales.

Con vistas a superar los inconvenientes anteriores, en esta tesis se plantean los siguientes objetivos fundamentales:

1º Obtener modelos NARMA neuronales que puedan proporcionar una aproximación conveniente del proceso dinámico no lineal cuando dichos modelos actúan como simulador del proceso y resolver así los inconvenientes que puedan presentar los modelos de identificación en paralelo mencionados anteriormente.

2º Encontrar métodos que proporcionen inicializaciones de los parámetros del controlador neuronal inverso, de modo que pueda obtenerse un control eficiente del proceso dinámico real cuando se aplica un esquema de control inverso con aprendizaje especializado.

3º Construir sistemas de control predictivo no lineales aplicables en tiempo real y que requieran de un menor esfuerzo computacional que los esquemas de control predictivo existentes en la actualidad.

Para cubrir los objetivos marcados, la tesis se estructura en cuatro capítulos.

En el primero de ellos, capítulo 1, se da una panorámica sobre las redes de neuronas artificiales y sobre los procesos dinámicos, presentándose conceptos que serán utilizados. En este capítulo se incluye también el estado del arte del uso de las redes de neuronas artificiales para tratar el problema de la modelización y el control de procesos dinámicos no lineales.

El capítulo 2 está dedicado al planteamiento del problema, presentándose las dificultades encontradas en los modelos de identificación en serie-paralelo para actuar como simulador del proceso y las limitaciones de las estrategias de control inverso y predictivo para controlar procesos dinámicos no lineales en tiempo real.

En el capítulo 3 se presentan las soluciones propuestas para resolver los problemas planteados en el capítulo anterior y conseguir así los objetivos marcados. Dicho capítulo contiene, en primer lugar, la descripción de la arquitectura de red parcialmente recurrente que se propone utilizar, tanto para la construcción de simuladores como

controladores de procesos dinámicos no lineales, y el algoritmo de aprendizaje para llevar a cabo su entrenamiento. En segundo lugar, se analiza la incapacidad de los modelos de identificación en paralelo mencionados anteriormente para representar adecuadamente el proceso dinámico, proponiéndose una solución para resolver los problemas que se presentan. Finalmente, se desarrollan los esquemas de control inverso y predictivo propuestos en este trabajo.

El capítulo 4 está dedicado a la validación de las soluciones propuestas en el capítulo anterior. Para ello se utiliza un proceso dinámico real que describe el comportamiento de la temperatura del fluido que circula en la camisa de un reactor químico situado en el Centro Común de Investigación de la Comunidad Europea de Ispra (Italia) y dentro del marco del proyecto FIRES (Facility for Investigating Runaway Events Safely). En este capítulo se realiza la descripción del reactor y de los circuitos de calentamiento y enfriamiento, la aplicación de las técnicas propuestas para modelizar y controlar el proceso dinámico en cuestión y finalmente un análisis de los resultados obtenidos.

Con la finalidad de comprobar la validez de las técnicas desarrolladas para afrontar la modelización y el control de procesos dinámicos no lineales, este capítulo incluye también los resultados que proporciona un modelo físico que describe el comportamiento dinámico de la temperatura del fluido de intercambio de calor (ZALD, 95), así como los resultados de control que se obtienen con el controlador actualmente incorporado en la instalación (controlador PID: proporcional-integral-derivativo).

Para finalizar, se establecen las conclusiones del trabajo desarrollado y posibles líneas futuras de investigación en este campo.

Capítulo 1:

Redes de neuronas artificiales y procesos dinámicos

El presente capítulo está organizado en dos secciones que contienen la presentación de las redes de neuronas artificiales (sección 1.1) y de los procesos dinámicos (sección 1.2).

En la sección 1.1 se realiza, en primer lugar, una breve introducción sobre algunas cuestiones históricas concernientes a las redes de neuronas artificiales, así como sus características más generales (apartado 1.1.1). Los apartados 1.1.2 y 1.1.3 están dedicados, respectivamente, a la descripción de la arquitectura del perceptron multicapa no lineal y a las redes de neuronas recurrentes, distinguiéndose las redes parcialmente recurrentes y las redes totalmente recurrentes. Los algoritmos de aprendizaje para llevar a cabo el entrenamiento de las redes de neuronas se repasan en el apartado 1.1.4, en el cual se presentan el algoritmo de retropropagación para redes estáticas y diferentes algoritmos que se disponen en la actualidad para las redes de neuronas con conexiones recurrentes o retroalimentadas. La sección finaliza con el apartado 1.1.5, en el que se muestran algunos de los resultados más importantes concernientes con las capacidades de aproximación de las redes de neuronas multicapas.

La sección 1.2, está dedicada a los procesos dinámicos, los cuales se introducen en el apartado 1.2.1. El apartado 1.2.2 contiene los dos formas existentes para representar matemáticamente un proceso dinámico: la descripción interna y la descripción externa, así como resultados referentes a la equivalencia entre ellas. El problema de la modelización y la identificación de procesos dinámicos se trata en el apartado 1.2.3, en el que se definen las fases que intervienen en la construcción de modelos y se repasan

algunas de las técnicas para afrontarlas. El apartado 1.2.4 incluye el estado de arte del uso de las redes de neuronas artificiales para afrontar el problema de la modelización de procesos dinámicos no lineales. En el apartado 1.2.5 se presenta el problema del control de procesos dinámicos y se da una visión panorámica de las técnicas clásicas y modernas de control. Finalmente, el apartado 1.2.6 contiene un repaso sobre cómo las redes de neuronas artificiales han intervenido en la construcción de sistemas de control para procesos dinámicos no lineales.

1.1 Redes de neuronas artificiales

1.1.1 Introducción

Durante varias décadas los científicos han mostrado un gran interés en intentar emular el funcionamiento del cerebro, persiguiendo la construcción de algoritmos capaces de procesar información al igual que éste.

Las redes de neuronas artificiales no son mas que una aproximación muy modesta del cerebro, y su realización está inspirada en el conocimiento científico existente sobre la estructura y forma de funcionamiento del sistema nervioso. Una red de neuronas es una interconexión de unidades, también llamadas neuronas o células, las cuales poseen una estructura similar a la de las neuronas biológicas. Cada unidad recibe múltiple información a través de sus entradas, procesan la información recibida y emiten una única salida o respuesta que se transmite a múltiples neuronas posteriores.

Los primeros estudios en el campo de las redes de neuronas fueron realizados por McCulloch y Pitts (MCPI, 43), estudios que mostraron la habilidad de un grupo de neuronas conectadas para llevar a cabo la implementación de ciertas funciones lógicas. Durante la década de los 50 hubo un considerable crecimiento es este campo de investigación, destacándose el trabajo de Rosenblatt (ROSE, 58), el cual e inspirándose en sus antecesores, introdujo la primera arquitectura de red de neuronas artificial con capacidad de aprendizaje: El Perceptron. Dicho autor mostró que, modificando los valores asociados a las conexiones, la red era capaz de aprender a responder según unas pocas funciones lógicas. Posteriormente y debido a que se demostraron las serias limitaciones de dicha red (MIPA, 69), la mayor parte de los investigadores en este área abandonaron su trabajo.

A principios de la década de los 80, sin embargo, las redes de neuronas artificiales volvieron a renacer, pues afortunadamente las limitaciones presentadas en (MIPA, 69) sólo eran aplicables a redes con una única capa de neuronas.

El resurgimiento de las redes de neuronas fue principalmente debido a Hopfield (HOPF, 82), el cual demostró que redes de neuronas totalmente conectadas tenían una gran capacidad para procesar información.

En la actualidad existe un gran número de estructuras diferentes de redes de neuronas artificiales. Esta diversidad radica principalmente en la organización de las neuronas y conexiones de la red, en las funciones de activación de dichas neuronas y en la forma de llevar a cabo el aprendizaje de la red. Entre ellas se destacan: El Perceptron (ROSE, 58), Adaline (WIHO, 60), Las Redes de Hopfield (HOPF, 82), (HOPF, 84), El Perceptron Multicapa (RUHI, 86), La Red de Jordan (JORD, 86), Las Redes de Base Radial (MODA, 88), La Red de Elman (ELMA, 88), Los Mapas de Kohonen (KOHO, 90), Teoría de la Resonancia Adaptativa, ART 1,2,3 (CAGR, 90), etc.

Para finalizar esta breve introducción, se destaca el masivo paralelismo de las estructura de las redes de neuronas artificiales, lo cual permite que gocen de propiedades como la tolerancia a fallos, capacidad de aprendizaje y aproximación a partir de ejemplos, degradación lenta del funcionamiento del sistema ante fallos de neuronas individuales, etc. Estas características han hecho que las redes de neuronas se utilicen para afrontar una gran variedad de problemas, entre los que se destacan el reconocimiento de patrones (imagen, voz, caracteres), compresión de datos, robótica, el análisis y el control de procesos, entre otros.

1.1.2 Perceptron multicapa no lineal

La red de neuronas llamada *perceptron multicapa* se caracteriza porque tiene sus neuronas agrupadas en subconjuntos disjuntos, llamados capas (figura 1.1). Cada neurona en cada capa está conectada a todas las neuronas de la siguiente capa; cada una de dichas conexiones tiene asociado un valor (número real), llamado peso de la conexión.

La primera capa recibe el nombre de capa de entrada a la red; las neuronas de esta capa se encargan únicamente de recibir las señales o patrones que proceden del exterior y propagar dichas señales a todas las neuronas de la siguiente capa. La última capa

actúa como salida de la red, proporcionando al exterior la respuesta para cada uno de los patrones de entrada; las demás capas de la red reciben el nombre de capas ocultas, pues no están directamente en contacto con el exterior.

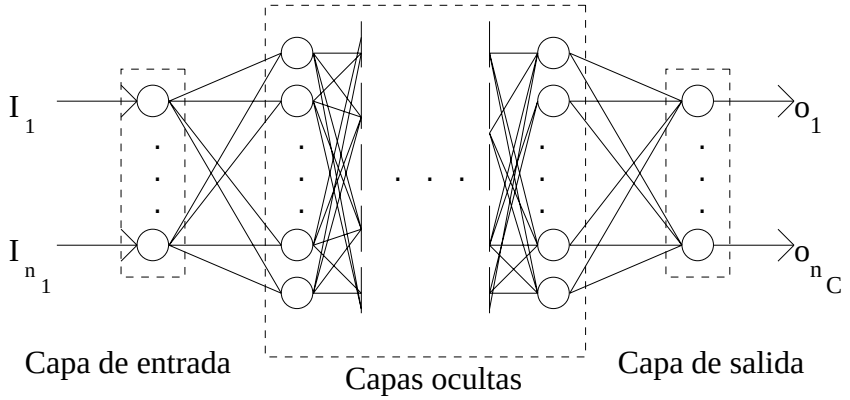


Figura 1.1. Perceptron multicapa.

Cada neurona de la red procesa la información recibida por sus entradas, produciendo una respuesta que se propaga a través de la conexión correspondiente y actúa como entrada para todas las neuronas de la siguiente capa. La relación entre la salida de la neurona y la información de entrada viene dada por la llamada función de activación. Debido a las características de la neurona biológica, las funciones de activaciones más utilizadas son la función sigmoidal y la función tangente hiperbólica (COWA, 67), denotadas respectivamente como $\sigma_1(x)$, $\sigma_2(x)$. Estas funciones poseen como imagen un rango continuo de valores dentro de los intervalos [0,1] y [-1,1], respectivamente, y están dadas por:

$$\sigma_1(x) = \frac{1}{1+e^{-x}} \qquad \sigma_2(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$$

$\sigma_1(x)$ y $\sigma_2(x)$ son funciones crecientes, aunque manteniendo dos niveles de saturación, el máximo que proporciona salida 1 y el mínimo que proporciona salida 0 o -1. Es fácil ver que ambas funciones están relacionadas de la forma $\sigma_2(x) = 2\sigma_1(2x) - 1$, por lo que la utilización de una u otra se elige en función del recorrido que interese.

Sea un perceptron multicapa con C capas y n_C neuronas en la capa c, c=1,2,...,C. El nivel de activación o salida de la neurona j, j=1,..., n_C, situada en la capa c, para el k-ésimo patrón de entrada es un número real $s_j^c(k)$ dado por:

$$s_j^1(k) = I_j(k),\ j = 1,\ldots,n_1 \tag{1.1}$$

$$s_j^c(k) = \sigma\left(\sum_{i=1}^{n_{c-1}} w_{ji}^c \cdot s_i^{c-1}(k) + \mu_j^c\right),\ j = 1,\ldots,n_c\ ,\ c = 1,\ldots,C \tag{1.2}$$

siendo:

i) $I(k) = (I_1(k),\ldots,I_{n_1}(k))$ el k-ésimo patrón de entrada a la red.

ii) w_{ji}^c el peso asociado a la conexión entre la neurona i de la capa c-1 y la neurona j de la capa c y μ_j^c es un número real que representa el umbral asociado a la neurona j de la capa c. Los pesos y los umbrales determinan el conjunto de parámetros ajustables de la red.

iii) $\sigma(\cdot)$ es la función de activación. En la mayor parte de los casos, dicha función es común a todas las neuronas de la red y es elegida por el diseñador de la topología, elección que se realiza en base a los valores de activación que se deseen que alcancen las neuronas.

Las salidas del perceptron multicapa vienen dadas por los niveles de activación de las neuronas de la última capa:

$$o_j(k) = s_j^C(k),\ j = 1,..,n_C \tag{1.3}$$

Consecuentemente, el perceptron multicapa define, a través de sus conexiones y de sus neuronas, una función continua (pues es composición de funciones continuas) del espacio $\Re^{n_1}$ (espacio de los patrones de entrada) al espacio $\Re^{n_C}$ (espacio de los patrones de salida), denotada como $\tilde{F}$:

$$O(k) = \tilde{F}(I(k), W) \tag{1.4}$$

siendo $O(k) = (o_1(k),...,o_{n_C}(k))$ el vector formado por las salidas de la red y W el conjunto de todos los pesos y umbrales de la red o conjunto de parámetros ajustables.

El número de capas y el número de neuronas en cada capa son parámetros fijos de la red, y determinan su tamaño o arquitectura.

1.1.3 Redes recurrentes

En el perceptron multicapa descrito en el apartado 1.1.2, la arquitectura o disposición de las neuronas se realiza por capas con una fuerte restricción en la conectividad, que consiste en no permitir conexiones entre neuronas creando ciclos o bucles. Esta restricción no limita la capacidad de la red en tareas donde los patrones sean estáticos, es decir, patrones en los que no interviene la variable tiempo y cuyo procesamiento no depende del orden de presentación a la red.

En este sentido, las redes cuyas conexiones no permite la creación de bucles han sido siempre consideradas como redes de neuronas estáticas, pues su arquitectura permite el procesamiento de patrones estáticos. Entre ellas se destacan el perceptron multicapa, descrito en el apartado anterior, y las redes de base radial, las cuales son redes con una única capa oculta de neuronas cuyas activaciones vienen dadas por funciones de Gauss (MODA, 88), (MODA, 89).

La fuerte restricción en la conectividad hace, sin embargo, que estas redes sean bastante limitadas en lo referente a la representación de información temporal o al tratamiento de patrones dinámicos, es decir patrones que dependen del tiempo en el sentido de que el valor actual del patrón está supeditado a sus valores anteriores. Aun así, es posible, por medio de una serie de estrategias, utilizar dichas estructuras estáticas para procesar patrones dinámicos. La idea que se aplica es muy simple, y consiste en que la entrada a la red esté compuesta, no sólo del valor del patrón en un determinado instante de tiempo, sino también de una cierta historia de dicho patrón, la cual se representa utilizando una secuencia finita temporal en el pasado. De esta forma, la información temporal puede ser procesada por la red, aunque no se puede olvidar que dicho procesamiento es debido a los patrones de entrada, y no a la propia arquitectura de la red. Esta idea ha sido ampliamente utilizada en el contexto de la modelización y el control de procesos dinámicos, (NAPA, 90), (CHBI, 92), entre otros.

Una alternativa, quizás más atractiva, para procesar patrones dinámicos, son las redes de neuronas recurrentes, las cuales se desarrollan con la finalidad de introducir elementos dinámicos y temporales en las redes de neuronas.

Las redes recurrentes son redes multicapas donde sus conexiones no están, en principio, sometidas a ninguna restricción. Esto autoriza, tanto la existencia de conexiones de una neurona a ella misma, como entre neuronas de una misma capa o entre neuronas de capas diferentes. Las arquitecturas de redes recurrentes incorporan, por tanto, bucles en las conexiones entre las neuronas, lo cual permite que la salida de una neurona de la red no sólo dependa del patrón de entrada, sino también de los estados anteriores de todas las neuronas conectadas a ella, provocando así un comportamiento dinámico en la red y facilitando el procesamiento de la información temporal. Estas redes fueron introducidas y discutidas en los trabajos de Hopfield (HOPF, 82), surgiendo así el primer modelo de red de neuronas recurrente.

Las redes de neuronas recurrentes empleadas para el tratamiento de patrones dinámicos se pueden dividir en dos grupos, *redes parcialmente recurrentes* y *redes totalmente recurrentes*. Todas ellas poseen un denominador común, y es que poseen conexiones que permiten tener en cuenta la activación de ciertas neuronas de la red en instantes anteriores de tiempo para obtener la respuesta de la red.

En las redes parcialmente recurrentes, la mayoría de las conexiones son, precisamente, no recurrentes, aunque existe un número pequeño de recurrencias cuidadosamente escogidas, que permiten a la red recordar el nivel de activación en un pasado de un cierto grupo de neuronas. Sin embargo, la presencia de recurrencias no complica en demasía el entrenamiento de la red, pues en la mayoría de los casos los pesos asociados a dichas conexiones son fijos. En estas redes aparecen una serie de neuronas especiales, llamadas neuronas de contexto, que son las receptoras de las recurrencias. Es decir, funcionan como una memoria donde se almacena el estado de una determinada capa obtenido en la iteración o instante de tiempo anterior. En algunos casos dichas neuronas de contexto actúan como neuronas de entrada. Entre las arquitecturas de redes parcialmente recurrentes se destacan la Red de Jordan (JORD, 86) y la Red de Elman (ELMA, 88). Las primeras se caracterizan porque las neuronas de contexto reciben una copia de la capa de salida y otra de sí mismas. En la red de Elman, dichas neuronas memorizan la activación de las neuronas de la capa oculta.

Las redes totalmente recurrentes son aquellas en las que no existe restricción en su conectividad. Esto implica que cada una de las neuronas de la red recibe como entrada la respuesta del resto de las neuronas en el instante precedente. En estas arquitecturas las conexiones recurrentes no se consideran fijas, sino que existen pesos asociados a ellas, produciéndose así un aumento considerable del número de parámetros ajustables de la red de neuronas. Esto significa aumentar la capacidad de representación de información de la red, aunque por otra parte complica su aprendizaje. Las redes totalmente recurrentes se pueden dividir en redes discretas y continuas. La activación de la neurona j de una red recurrente continua obedece a la siguiente ecuación diferencial:

$$\tau_j \cdot \frac{\partial s_j}{\partial t} = -s_j + \sigma\left(\sum_i w_{ji} \cdot s_i\right) \tag{1.5}$$

donde t indica la variable tiempo, τ_j es una constante de tiempo para la neurona j y el índice i varía en el conjunto de todas las neuronas conectadas a la neurona j. Para las redes recurrentes discretas el nivel de activación de sus neuronas se determina discretizando la ecuación anterior. Aproximando $\frac{\partial s_j}{\partial t}$ por $\frac{s_j(t+\tau_j) - s_j(t)}{\tau_j}$ y tomando $\tau_j = 1$, se obtiene: la siguiente expresión:

$$s_j(t+1) = \sigma\left(\sum_i w_{ji} \cdot s_i(t)\right) \tag{1.6}$$

1.1.4 Algoritmos de aprendizaje

El aprendizaje en el contexto de redes de neuronas artificiales puede definirse como el proceso mediante el cual la red modifica sus respuestas ante determinadas entradas. Dicho proceso, conocido también como entrenamiento, consiste en la modificación paulatina de los parámetros ajustables o pesos de la red, modificación que se realiza en base a un cierto criterio establecido y que permite que la red “aprenda” a dar las respuestas adecuadas en función de los patrones de entrada presentados.

Generalmente, el criterio para adaptar los parámetros de la red consiste en minimizar las diferencias entre las respuestas de la red para cada uno de los patrones de

entrada y las respuestas deseadas para estos patrones, proporcionadas a la red por el entorno. Estas respuestas deseadas actúan como guía para que la red aprenda a realizar una determinada tarea. La determinación, por tanto, de los parámetros ajustables de la red puede tratarse como un problema de optimización para minimizar la siguiente función error:

$$E(W) = \frac{1}{N_p} \cdot \sum_{k=1}^{N_p} e(k) \tag{1.7}$$

siendo N_p el número de patrones disponibles y e(k) el error asociado a cada patrón k. Suponiendo que la red tiene m neuronas de salida y que para el patrón k-ésimo genera las salidas $O(k) = (o_1(k),\ldots,o_m(k))$, siendo $t(k) = (t_1(k),\ldots,t_m(k))$ las salidas deseadas correspondientes, el error asociado a dicho patrón se define como:

$$e(k) = \frac{1}{2} \cdot \sum_{i=1}^{m} (t_i(k) - o_i(k))^2 \tag{1.8}$$

Puesto que las salidas de la red $o_i(k)$ dependen del conjunto de parámetros ajustables W, el error E definido en 1.7 es también función de dichos parámetros, pudiendo escribir E(W).

La presencia de funciones de activación no lineales hace que la respuesta de la red sea no lineal con respecto a los parámetros ajustables, por lo que el problema de determinar el conjunto de parámetros que minimizan la función error E(W) definida por las ecuaciones 1.7-1.8, es un problema de minimización no lineal y, como consecuencia, tienen que utilizarse técnicas de optimización no lineales. Dichas técnicas están generalmente basadas en una adaptación de los parámetros siguiendo una cierta dirección de búsqueda. En el contexto de redes de neuronas, la dirección de búsqueda más comúnmente usada es la dirección negativa del gradiente de la función E(W), $-\nabla E(W) = -\left(\frac{\partial E}{\partial w}\right)_{\forall w}$, pues conforme al cálculo de varias variables, ésta es la dirección en la que la función decrece. No obstante, se han desarrollado también métodos de búsqueda aleatoria para localizar el mínimo de la función E (MATY,65), (SOWE, 81), aunque no entraremos en detalle, pues no son de interés en este trabajo.

Basándonos en el método de descenso del gradiente, cada parámetro w de la red se adapta en cada iteración o ciclo de aprendizaje n de acuerdo con la siguiente ley:

$$w^{(n)} = w^{(n-1)} - \alpha \cdot \frac{\partial E}{\partial w}, \ \forall w \tag{1.9}$$

$$\text{siendo } \frac{\partial E}{\partial w} = \frac{1}{N_p} \cdot \sum_{k=1}^{N_p} \frac{\partial e(k)}{\partial w} \tag{1.10}$$

Aplicando la regla de la cadena para derivar el error e(k) respecto al parámetro w, de 1.8 se obtiene que:

$$\frac{\partial e(k)}{\partial w} = \frac{1}{2} \cdot \sum_{i=1}^{m} \frac{\partial e(k)}{\partial o_i(k)} \cdot \frac{\partial o_i(k)}{\partial w} = -\sum_{i=1}^{m} (t_i(k) - o_i(k)) \cdot \frac{\partial o_i(k)}{\partial w} \tag{1.11}$$

El proceso de entrenamiento se resume: Partiendo de un punto $W^{(0)} \in \Re^{n_w}$ (aleatorio cuando no se dispone de ninguna información adicional) siendo n_w el número de parámetros ajustables de la red, nos desplazamos en el espacio $\Re^{n_w}$ siguiendo la dirección negativa del gradiente de E en el punto $W^{(n-1)}$, alcanzando así un nuevo punto $W^{(n)}$(ecuación 1.9), que estará más próximo al mínimo de la función E que el anterior. El proceso continúa hasta que los parámetros dejen de sufrir cambios importantes, lo cual sucede cuando E alcanza un mínimo, es decir, cuando $\partial E/\partial w \approx 0$ para todo w.

El valor α, llamado también *razón* o *tasa de aprendizaje*, es el encargado de controlar cuánto se desplazan los pesos siguiendo la dirección negativa del gradiente. Este valor determina, entonces, la magnitud de dicho desplazamiento y, por lo tanto, influye en la velocidad de convergencia del algoritmo.

Desde un punto de vista teórico, el valor de la razón de aprendizaje no tendría que ser el mismo en cada iteración. De hecho, debería de depender de lo lejos o lo cerca que estemos del mínimo de la función. Un valor grande favorece a una convergencia más rápida al principio, pues los cambios que se producen en los pesos permiten avanzar rápidamente en la superficie que describe la función error E. Sin embargo, en las proximidades del mínimo existe el riesgo de saltar y oscilar alrededor de él. Por otra

parte, razones de aprendizaje pequeñas podrían evitar este problema, pero sin embargo la convergencia del algoritmo será más lenta.

Se ha mencionado que el proceso de adaptar los pesos finaliza cuando $\partial E/\partial w \approx 0$ para todo w, lo cual no garantiza que el mínimo alcanzado sea un mínimo global de la función E(W). Debido a que el método usa únicamente información local sobre la superficie para determinar la dirección en cada iteración, se corre el riesgo de que los pesos se muevan hacia un mínimo, que no tiene por qué ser un mínimo global, de manera que el proceso iterativo puede finalizar en un mínimo local. En estos puntos el método detecta que cualquier pequeño cambio en los pesos, positivo o negativo, incrementa el error y no es capaz de determinar en qué dirección debe de mover los pesos para, utilizando un incremento mayor, conseguir que el error vuelva a decrecer. En el caso de que se disponga de suficiente información como para detectar que el proceso de entrenamiento ha alcanzado un mínimo local, es posible aumentar la razón de aprendizaje con la finalidad de salir de dicho punto, y posteriormente continuar el mecanismo iterativo.

En el proceso de aprendizaje descrito anteriormente, los pesos se desplazan siguiendo la dirección negativa de la función E, y por tanto el proceso se desarrolla para encontrar su mínimo. Aunque estrictamente hablando, el aprendizaje de la red tiene que realizarse para minimizar dicha función E, el procedimiento más utilizado está basado en métodos del gradiente estocástico, los cuales consisten en una sucesiva minimización de los errores locales e(k), k=1,...,N_p, en lugar de minimizar el error total E. En este caso, los pesos de la red se adaptan para cada k-ésimo patrón de entrada presentado a la red, y no cuando el conjunto total de patrones ha sido presentado (RUHI, 86), es decir:

$$w^{(k)} = w^{(k-1)} - \alpha \cdot \frac{\partial e(k)}{\partial w}, \ \forall w, \ \forall k \qquad (1.12)$$

Algoritmos de retropropagación

Según el procedimiento descrito anteriormente, para llevar a cabo la adaptación de los parámetros ajustables de la red es necesario calcular los gradientes de los errores locales e(k). En la ecuación 1.11 se observa que en el cálculo de dichos gradientes intervienen las variaciones de las salidas de la red respecto a sus parámetros, $\frac{\partial o_i}{\partial w}$, i=1,...,m, términos que se calculan de acuerdo con la arquitectura de la red.

Para redes multicapas sin recurrencias, las derivadas parciales de las salidas de la red respecto a sus parámetros se obtienen fácilmente. Basta tener en cuenta las activaciones de las neuronas de la red (ecuaciones 1.1-1.2-1.3) y derivar dichas expresiones respecto al parámetro w. Debido a que las neuronas de la red están agrupadas en capas de distintos niveles, es posible obtener dichos términos de una forma sencilla y eficiente, resultando el conocido algoritmo de retropropagación (RUHI, 86), también referido en este trabajo como *retropropagación estático*. El término de retropropagación se utiliza debido a la forma de implementar el método del gradiente en redes multicapas, pues el error cometido en la salida de la red es propagado hacia atrás, transformándolo en un error para cada una de las neuronas de la red.

Cuando o_i es la salida de una red de neuronas recurrentes, la variación de dicha salida con respecto al parámetro w, $\frac{\partial o_i}{\partial w}$, no se puede calcular utilizando retropropagación estática. Debido a la presencia de conexiones recurrentes, los parámetros ajustables de la red no sólo aparecen de forma explícita, como en el caso del perceptron multicapa, sino también de forma implícita a través de las activaciones de las neuronas de la red en el instante de tiempo anterior, valores que intervienen en la salida de la red y que dependen, a su vez, del conjunto de parámetros ajustables de la red. Es el caso, por ejemplo, de una función f de la forma f(x(w),w) en la que la variable x depende también del parámetro w. La derivada de la función f respecto al parámetro w adopta la forma: $\frac{\partial f}{\partial x} \cdot \frac{\partial x}{\partial w} + \frac{\partial f}{\partial w}$.

Como consecuencia, el algoritmo de retropropagación estático tuvo que ser modificado y extendido para redes de neuronas recurrentes. Se han propuesto y desarrollado diferentes extensiones, pero todas ellas se basan en el concepto de derivada expuesto anteriormente.

Entre estas extensiones se encuentra el llamado algoritmo de "*Backpropagation Through Time*" (Retropropagación en el tiempo) (ROFA, 87), (WERB, 90). Consiste en desarrollar en el tiempo una red recurrente para poder analizar su evolución como una red multicapa no recurrente y aplicar el tradicional algoritmo de retropropagación para redes estáticas. Una versión de dicho algoritmo para redes recurrentes continuas pueden encontrarse en (PEAR, 89).

El algoritmo de retropropagación en el tiempo calcula el gradiente del error propagando una serie de variables hacia atrás en el tiempo y, como consecuencia, su implementación requiere de una memoria que almacene el estado de la red durante un periodo de tiempo. Existen otros métodos para llevar a cabo el aprendizaje de redes recurrentes que no requieren de dicha memoria y son aplicables en tiempo real. En estos casos, el cálculo del gradiente se lleva a cabo propagando hacia delante en el tiempo un conjunto de variables al igual que se propagan las activaciones de las neuronas de la red. Entre ellos se destaca el algoritmo propuesto en (WIZI, 89), también conocido como "*Real-Time Recurrent Learning*" (Aprendizaje recursivo en tiempo real). Los autores desarrollaron dicho algoritmo para redes recurrentes discretas, aunque su extensión a casos continuos es bastante simple.

En (NAPA, 91) se proponen aproximaciones para llevar a cabo el aprendizaje de cuatro representaciones recurrentes discretas en las que intervienen redes de neuronas multicapas, extensiones que se engloban bajo el nombre de "*Dynamic Backpropagation*" (Retropropagación dinámica).

Posteriormente, otros autores (PUFE, 94), (PACH, 94) han desarrollados métodos para llevar a cabo el aprendizaje de redes multicapas discretas con recurrencias entre las neuronas de una misma capa y recurrencias de una neurona a ella misma.

1.1.5 Aproximación mediante redes de neuronas

Durante los últimos doscientos años el Análisis Matemático ha sufrido un amplio desarrollo en lo referente a la Teoría de Aproximación, obteniéndose importantes clases de funciones, las cuales, bajo ciertas condiciones, tienen la capacidad de

aproximar cualquier función (NARE, 92). Estas familias de funciones incluyen los polinomios, las funciones trigonométricas, los splines, etc. Las redes de neuronas multicapas constituyen una nueva clase de funciones, desarrollada en el últimos años, que posee también la propiedad de ser aproximadores universales.

Cada una de estas clases de aproximadores tiene sus propias características y su propia historia. Se conocen ciertas condiciones bajo las cuales un método es preferible ante otro, pero en ningún caso se puede decir que un método sea absolutamente el mejor. De hecho, las consideraciones de cada problema son las que determinan la elección de un aproximador u otro.

Teorema de aproximación de Weierstrass.

Sea $(C(U,\Re),\|.\|_\infty)$ el espacio de las funciones continuas $f: U \subset \Re^N \to \Re$ siendo U un compacto de $\Re^N$ y $\|.\|_\infty$ la norma uniforme definida como: $\|f\|_\infty = \sup_{(x_1,\ldots,x_N)\in U} |f(x_1,\ldots,x_N)|$.

El teorema establece que el conjunto de los polinomios en las variables $x_1,\ldots,x_N$ con coeficientes reales es denso en el espacio $(C(U,\Re),\|.\|_\infty)$, es decir, dada $f \in C(U,\Re)$ y dado $\varepsilon>0$, existe un polinomio $P(x_1,\ldots,x_N)$ con coeficientes reales tal que $\|f - P\|_\infty < \varepsilon$.

El teorema de Weierstrass fue revisado en (STON, 48) con la finalidad de encontrar propiedades generales sobre la aproximación de funciones que no fueran intrínsecas a los polinomios; el resultado es el siguiente teorema.

Teorema de Stone-Weierstrass.

Sea U un compacto de $\Re^N$ y F una subálgebra del conjunto de las funciones continuas en U, $C(U,\Re)$. Si F separa los puntos de U ($\forall x, y \in U, x \neq y \Rightarrow \exists f \in F$ tal que $f(x) \neq f(y)$) y F no desvanece en los puntos de U ($\forall x \in U, \exists f \in F$ tal que $f(x) \neq 0$), entonces F es denso en $(C(U,\Re),\|.\|_\infty)$.

En este teorema el conjunto de polinomios se sustituye por un subconjunto adecuado de $C(U,\Re)$. Dada una función $f \in C(U,\Re)$, el resultado anterior muestra que puede obtenerse una aproximación a dicha función utilizando elementos de la subálgebra $F \subset C(U,\Re)$ y con el grado de exactitud deseado. El teorema de Stone-Weierstrass proporciona, entonces, un criterio simple para establecer si una

determinada clase de funciones tiene la propiedad de aproximar con la norma $\|.\|_\infty$ a cualquier función continua.

Basándose en este teorema, diferentes autores, (CYBE, 89), (HOST, 89), han obtenido independientemente que un perceptron no lineal, con al menos una capa oculta de neuronas, puede aproximar cualquier función continua sobre un compacto U de $\Re^N$ con el grado de exactitud deseado. Por otra parte, en (HAKE, 90) se demuestra que las redes de base radial poseen también la propiedad de poder aproximar cualquier función continua sobre un compacto U de $\Re^N$.

Estos resultados no proporcionan, sin embargo, un método que permita determinar cuál es el número óptimo de neuronas de la capa oculta para obtener la mejor aproximación. Incluso para el caso del perceptron multicapa, dichos resultados no garantizan que, aumentando el número de capas ocultas, las aproximaciones obtenidas sean mejores. Normalmente, el número de capas ocultas y el número de neuronas en cada capa se determinan por prueba y error en cada contexto específico.

1.2 Procesos dinámicos

1.2.1 Definición de proceso dinámico

Un sistema o proceso puede definirse como una combinación de elementos o componentes relacionados entre sí y actuando para alcanzar una determinada meta (SHKU, 90). El estudio de un proceso dinámico es el estudio de su variación a lo largo del tiempo. La teoría matemática de sistemas trata el análisis y la síntesis de dichos procesos dinámicos.

Un proceso con una variable de entrada y otra de salida puede ser esquemáticamente representado como indica la figura 1.2, donde *y* representa la variable de salida del proceso o variable medible, la cual proporciona información útil sobre el comportamiento del proceso y *u* es la variable de entrada al proceso.

La principal característica de un proceso dinámico, y lo que lo distingue de los procesos estáticos, es que la salida en un instante de tiempo no sólo depende de la

entrada en ese instante, sino también de una cierta historia en el pasado de las variables de entrada y salida.

Figura 1.2. Representación esquemática de proceso.

En este trabajo se consideran procesos dinámicos discretos, es decir, la variación de las magnitudes que se manejan se producen en intervalos discretos de tiempo. Se supone, por tanto, que el tiempo toma valores en un conjunto discreto $\{0,1,2,\ldots,n,\ldots\}$, ignorándose los valores que alcanzan las variables entre dos medidas consecutivas de tiempo.

En toda la literatura de procesos dinámicos se hace una distinción entre procesos dinámicos lineales y procesos dinámicos no lineales, pues las técnicas para su tratamiento son diferentes. Se dice que un proceso dinámico es lineal cuando la relación que liga las variables que intervienen en el comportamiento dinámico del proceso es lineal. En el caso contrario, se trata de un proceso dinámico no lineal.

La mayor parte de los procesos dinámicos en el mundo real son procesos dinámicos no lineales, y sin embargo las técnicas más desarrolladas son técnicas lineales. Debido a que existen procesos dinámicos para los que la no linealidad se interpreta como una perturbación de un caso lineal, del estudio del correspondiente caso lineal se pueden obtener conclusiones útiles. No obstante, y debido a que no siempre la no linealidad de un proceso puede interpretarse como tales perturbaciones, es interesante disponer de técnicas no lineales para el tratamiento de procesos dinámicos no lineales.

1.2.2 Representaciones de un proceso dinámico

El análisis y la síntesis de procesos dinámicos se suele realizar utilizando dos formas diferentes de representación, que se denominan descripción interna o representación del proceso en el espacio de estados y descripción externa o

representación del proceso en el espacio entrada-salida, las cuales se describen brevemente en este apartado.

Descripción interna

La representación interna de un sistema dinámico se basa en el concepto de estado del proceso. El estado de un proceso dinámico se define como el menor conjunto de variables cuyo valor, en un cierto instante, resume el pasado dinámico del sistema y es suficiente para predecir su evolución futura (OLLE, 91). Las variables que representan los estados del proceso no tienen que corresponder necesariamente con variables o magnitudes físicas medibles; pueden ser variables auxiliares que se incorporan para poder representar el comportamiento dinámico del proceso mediante un sistema de ecuaciones diferenciales de primer orden.

La descripción interna de un proceso dinámico establece, por tanto, una relación indirecta entre las variables de entrada y salida, pues en dicha descripción intervienen, como se ha dicho, las variables de estados; esta relación indirecta se hace explícita a través de un conjunto de ecuaciones en diferencias que determinan la representación interna del proceso. Para un proceso dinámico discreto con una variable de entrada y otra de salida, dicha representación se escribe de la siguiente forma:

$$\left.\begin{array}{l}\mathbf{x}(k+1) = T(\mathbf{x}(k), u(k-d)) \\ y(k) = L(\mathbf{x}(k))\end{array}\right\}, k \in N_0\ ,\ \mathbf{x}(0) = \mathbf{x}_0 \in \Re^n \qquad (1.13)$$

siendo k la variable discreta tiempo; $u(k)$ la variable de entrada al proceso, con $u(-d) = \ldots = u(-1) = u(0)$; d el retraso natural del proceso respecto a dicha variable, es decir, el tiempo que requiere el proceso para reaccionar ante un cambio en la entrada; $\mathbf{x}(k) = (x_1(k), \ldots, x_n(k)) \in \Re^n$, el vector de las variables de estado en el instante de tiempo k; $y(k)$ la salida del proceso dinámico. La función T, llamada función de transición, es una aplicación definida de $\Re^n \times \Re$ en $\Re^n$ y permite determinar el estado $\mathbf{x}(k)$ a partir del estado inicial $\mathbf{x}(0) = \mathbf{x}_0$ y de la variable de entrada definida en el intervalo discreto [0, k-d]. La aplicación L, también llamada función de salida, está definida de $\Re^n$ en $\Re$ y proporciona el valor de la salida del proceso, y(k), en cada instante de tiempo k a partir del estado del proceso, $\mathbf{x}(k)$, en ese instante.

Cuando el proceso dinámico es lineal, las funciones de transición y lectura se caracterizan porque son funciones lineales, de manera que la representación interna para un proceso dinámico discreto lineal viene dada por las siguientes ecuaciones:

$$\left.\begin{aligned}\mathbf{x}(k+1) &= A(k)\cdot\mathbf{x}(k) + B(k)\cdot u(k-d)\\ y(k) &= C(k)\cdot\mathbf{x}(k)\end{aligned}\right\}\ k\in N_0\ ,\ \mathbf{x}(0) = \mathbf{x}_0\ \in \Re^n \qquad (1.14)$$

siendo A(k), B(k) y C(k) matrices con coeficientes reales dependientes del tiempo k y de dimensión n×n, n×1 y 1×n respectivamente.

Se dice que un proceso dinámico lineal es invariante en el tiempo cuando sus coeficientes son constantes en la variable k, en cuyo caso la representación interna puede escribirse de la forma:

$$\left.\begin{aligned}\mathbf{x}(k+1) &= A\cdot\mathbf{x}(k) + B\cdot u(k-d)\\ y(k) &= C\cdot\mathbf{x}(k)\end{aligned}\right\}\ k\in N_0\ ,\ \mathbf{x}(0) = \mathbf{x}_0\ \in \Re^n \qquad (1.15)$$

Descripción externa

Obtener una descripción externa del proceso, también llamada representación del proceso en el espacio entrada-salida, significa encontrar una relación explícita directa entre las variables de entrada y salida del proceso dinámico. Esta descripción tiene una importancia fundamental en la teoría de sistemas, pues describe el comportamiento dinámico del proceso a partir de las variables observables o medibles.

La representación externa viene dada por ecuaciones en diferencias de orden mayor a uno, las cuales relacionan el valor actual de la variable de salida del proceso con una secuencia discreta finita en el pasado de la salida y entrada al proceso de la siguiente forma:

$$y(k+1) = F(y(k),\ldots,y(k-n_y),u(k-d),\ldots,u(k-d-n_u))\ ,\ k\in N_0 \qquad (1.16)$$

siendo $y(-n_y) = \ldots = y(0) = y_0 \in \Re$, $u(-d-n_u) = \ldots = u(-1) = u(0)$; n_y, n_u son números naturales que indican la longitud de las secuencias discretas de las variables de salida y entrada, respectivamente; y F(.) es un funcional que dicta la relación entre

dichas secuencias. Los números n_y, n_u tienen que ser números finitos, pues, en caso contrario, dicha descripción no podrá utilizarse para representar un proceso dinámico.

Para procesos dinámicos lineales, la descripción externa puede escribirse de la forma:

$$y(k+1) = \sum_{i=0}^{n_y} p_i \cdot y(k-i) + \sum_{j=0}^{n_u} q_j \cdot u(k-d-j) \qquad (1.17)$$

siendo p_i y q_j números reales.

La representación dada por 1.17 se conoce con el nombre de modelos ARMA (AutoRegressive-Moving-Average) (GOSI, 84). Para procesos dinámicos no lineales es más complicado obtener una forma general del funcional F(.) que represente a todos o casi todos los procesos dinámicos no lineales; para ello será necesario recurrir a la teoría matemática de aproximación, como se verá más adelante. En cualquier caso, e independientemente de la forma del funcional F, la representación dada en la ecuación 1.16 cuando F(.) es no lineal es referida en la literatura como modelos NARMA (ARMA no lineales) (LEBI, 85).

Las descripciones interna y externa de un proceso dinámico no son más que dos lenguajes o caminos diferentes para representar los procesos dinámicos y existen resultados que demuestran la equivalencia entre ellas, (OLLE, 91), (LEBI, 85).

En los siguientes apartados se supone que el proceso dinámico no posee retraso con respecto a su variable de entrada, es decir d=0.

1.2.3 Modelización e identificación de procesos dinámicos

En el apartado anterior se han descrito las distintas formas de representar un proceso dinámico discreto. Sin embargo, estas representaciones no están disponibles inicialmente ya que, en la mayor parte de los casos, la información disponible sobre el proceso viene dada por un conjunto de datos observados, los cuales requieren de una interpretación para obtener una representación matemática del proceso. El problema de determinar dicha representación, también llamada modelo del proceso, es conocido como problema de modelización e identificación de procesos dinámicos.

Para la obtención de un modelo de un proceso dinámico se distinguen dos fases, llamadas *fase de modelización* y *fase de identificación*. A continuación se definen cada una de estas fases y se repasan algunas de las técnicas y métodos existentes para afrontarlas.

Fase de modelización

La fase de modelización consiste en caracterizar la clase a la que pertenece el proceso dinámico, o lo que es lo mismo, determinar la estructura del modelo.

La estructura elegida puede pertenecer a la clase de modelos físicos, los cuales determinan la representación interna del proceso y se construyen siguiendo exclusivamente las leyes físicas que gobiernan el comportamiento de dicho proceso dinámico. La construcción de dichos modelos requiere un conocimiento sobre el proceso y el principal problema es determinar las variables de estado del proceso. Dichas variables se suelen elegir como posiciones y velocidades para procesos mecánicos, voltajes y corrientes para procesos eléctricos, niveles y flujos para sistemas hidráulicos y temperatura, presiones y densidades para procesos térmicos. La relación entre las variables de estado y las variables medibles del proceso se determinan utilizando los balances de fuerza, momento, masa y energía (ASWI, 90). La aplicación de estas leyes determinan los funcionales T y L que aparecen en la ecuación 1.13.

La principal ventaja de los modelos físicos es que los parámetros y variables que intervienen poseen una interpretación física. Sin embargo, es necesario tener presente que puede llegar a ser extremadamente difícil considerar todas las leyes físicas que intervienen en el comportamiento dinámico de un proceso real y que, aun suponiendo que esto fuera posible, el modelo resultante puede ser complicado, difícilmente tratable (pues intervienen un gran número de ecuaciones diferenciales) y su construcción puede requerir una gran cantidad de tiempo.

Otra clase estándar de modelos son los pertenecientes a la categoría de modelos empíricos, como pueden ser los modelos ARMA en el caso de procesos lineales y los modelos NARMA para procesos no lineales, correspondientes a las descripciones externas del proceso descritas anteriormente. Estas representaciones poseen un gran valor ya que son modelos simples, fáciles de construir y sobre todo fáciles de tratar con vistas a aplicaciones de control.

Si el proceso dinámico es lineal, la fase de modelización consiste en determinar la longitud de las secuencias discretas para las variables de entrada y salida, n_y y n_u, que determinan el modelo ARMA correspondiente. La fase de modelización para modelos NARMA no lineales es más compleja puesto que no sólo consiste en establecer dichos valores, sino que también es necesario determinar la estructura o forma del funcional F en la ecuación 1.16. Dicho funcional es claramente muy general y su forma puede ser muy complicada y raramente conocida a priori (CHBI, 90).

Con el objetivo de crear estructuras generales para el funcional F(.), es interesante recurrir a los resultados que proporciona la teoría matemática de aproximación de funciones. De este modo, podrá utilizarse un conjunto de funciones conocidas y con propiedades de aproximadores universales para determinar la estructura del modelo NARMA o forma del funcional F.

De la aplicación directa del teorema de Weierstrass, se deduce que es posible obtener modelos NARMA cuya estructura es un polinomio. Suponiendo que F es una función continua, el teorema garantiza la existencia de un polinomio P_F tal que $\forall \varepsilon > 0 \ \left\| F - P_F \right\|_\infty < \varepsilon$ siendo $\|\cdot\|_\infty$ la norma uniforme en el espacio $\Re^{n_y+n_u+2}$. De esta forma el modelo NARMA puede escribirse de la forma:

$$\tilde{y}(k+1) = P_F(y(k),\ldots,y(k-n_y),u(k),\ldots,u(k-n_u)) \qquad (1.18)$$

siendo $\tilde{y}(k+1)$ una aproximación de $y(k+1)$ y P_F una expresión polinómica que combina la secuencia $(y(k),\ldots,y(k-n_y),u(k),\ldots,u(k-n_u))$. La construcción de modelos NARMA utilizando polinomios se ha tratado por varios autores, entre ellos (SONT,79), (CHBI, 89), (TSBI, 92).

Los modelos construidos con polinomios tienen la ventaja de que, siendo modelos no lineales, siguen conservando la linealidad en sus parámetros, lo cual permite que puedan utilizarse técnicas lineales de optimización para la estimación de sus parámetros. Sin embargo, presentan la desventaja de que su construcción puede llegar a ser muy laboriosa ya que, incluso para secuencias discretas de longitud no excesivamente grande, el número de los monomios que forman el polinomio puede llegar a ser excesivo; en este caso, será necesario hacer una selección de los términos más importantes. Generalmente, es necesario disponer de cierta información sobre la

naturaleza de las componentes no lineales del proceso para llevar a cabo la selección de los monomios que componen el polinomio P_F. Además, los polinomios son muy inestables respecto a pequeñas perturbaciones de sus coeficientes, lo cual conduce a dificultades computacionales, fundamentalmente en la fase de identificación o estimación de sus parámetros.

Debido a la capacidad de las redes de neuronas artificiales para aproximar funciones, otra alternativa para construir modelos NARMA es recurrir a estas estructuras, aproximando el funcional F mediante una red de neuronas. Esta posibilidad será presentada en el apartado 1.2.4.

Fase de identificación

La fase de identificación consiste en obtener estimaciones de los parámetros o coeficientes desconocidos que intervienen en la estructura elegida del modelo, de forma que la determinación de dicho modelo se concluya.

Para llevar a cabo esta fase es necesario realizar un conjunto de experimentos, para los que el proceso actúa como una caja negra. Dichos experimentos proporcionan pares de medidas de las variables de entrada y salida (variables medibles del proceso) durante la evolución temporal del proceso y que denotaremos como {u(k),y(k+1)}, para k=0,...N-1.

Cuando se formula el problema de identificación, el objetivo es obtener estimaciones de los parámetros que intervienen en el modelo tales que la representación obtenida reproduzca lo más fielmente posible la evolución descrita por las medidas tomadas del proceso real. Es necesario, por tanto, introducir un criterio que proporcione una medida de fiabilidad del modelo. Para procesos dinámicos discretos, dicho criterio se expresa frecuentemente de la siguiente forma:

$$E_m = \frac{1}{N} \cdot \sum_{k=0}^{N-1} g(e_m(k+1)) \qquad (1.19)$$

siendo $e_m(k+1) = \left| y(k+1) - \tilde{y}(k+1) \right|$ con $\tilde{y}(k+1)$ la predicción proporcionada por el modelo y g una función normalmente elegida como (ASWI, 90):

$$g(z) = \frac{1}{2} \cdot z^2 \qquad (1.20)$$

La estimación de los parámetros de un modelo se formula entonces como un problema de optimización. Existen situaciones, principalmente cuando se estiman los parámetros de un modelo físico, en las que el problema de optimización viene acompañado de restricciones en los parámetros; dichas restricciones tienen que ser impuestas para que los parámetros conserven su sentido físico.

El problema de optimización puede ser resuelto utilizando diferentes técnicas, las cuales se dividen en dos grandes grupos: técnicas off-line y técnicas on-line (ASWI, 90). Los métodos on-line proporcionan recursivamente estimaciones de los parámetros a medida que se reciben las observaciones o datos experimentales del proceso real. Con un método off-line, sin embargo, los parámetros del modelo se identifican después de la obtención de los datos experimentales. Estos últimos proporcionan estimaciones más precisas, aunque existen situaciones en las que será necesario llevar a cabo una estimación on-line.

En el caso de que el modelo elegido para representar el proceso dinámico presente una dependencia lineal respecto a sus parámetros, uno de los métodos más populares y utilizados es el conocido método de los mínimos cuadrados (GOSI, 84), el cual proporciona una solución analítica del problema de optimización.

El método de los mínimos cuadrados recursivos es la versión on-line del anterior. En este caso las estimaciones para k+1 observaciones se obtienen utilizando las estimaciones conseguidas en el paso anterior para k observaciones (GOSI, 84).

Cuando el modelo es no lineal con respecto a sus parámetros desconocidos tienen que utilizarse técnicas de optimización no lineales (BICH, 89), (CHBI, 89), las cuales en su gran mayoría están basadas en un método del gradiente con o sin restricciones. Dichas técnicas proporcionan soluciones numéricas, calculadas de forma iterativa, al problema de identificación de procesos dinámicos.

Construcción de un modelo

Una vez que se ha descrito en qué consisten las fases de modelización e identificación y algunas de las técnicas utilizadas para afrontarlas, es conveniente resumir cómo tratar el problema de modelización e identificación de procesos

dinámicos, cuando se está ante un caso concreto. El procedimiento a seguir puede esquematizarse en los siguientes puntos:

1. Obtención de los datos experimentales utilizando el proceso real.
2. Basándose en la naturaleza del proceso, se elige un conjunto de modelos candidatos para representar el proceso dinámico, estableciendo la estructura de cada uno de ellos. Los parámetros o coeficientes desconocidos que forman parte de esos modelos candidatos se estiman utilizando un método específico.
3. Debido a que generalmente no es posible obtener un modelo que represente de forma exacta al proceso real, sino que se dispone de una serie de modelos que lo aproximan, es necesario establecer un criterio para seleccionar un modelo particular del conjunto de candidatos. Dichos criterios pueden ser de diferente naturaleza; criterios numéricos, los cuales miden la capacidad para representar y aproximar la dinámica del proceso; criterios de simplicidad que evalúan la simplicidad y tratabilidad del modelo para aplicaciones en tiempo real. En casos prácticos, para validar un modelo se utiliza normalmente una combinación de diferentes criterios, los cuales se eligen dependiendo, principalmente, de la finalidad del modelo del proceso dinámico.

1.2.4 Redes de neuronas artificiales en el problema de la modelización e identificación de procesos dinámicos no lineales

Debido a las capacidades de las redes de neuronas artificiales para aproximar funciones y aprender relaciones altamente no lineales y complejas, en los últimos años un gran número de autores han utilizado dichas estructuras para la construcción de modelos no lineales. Existen diferentes formas y caminos para tratar el problema, las cuales se describen brevemente en lo que sigue.

Como se mencionaba en el apartado 1.1.3, las redes de neuronas estáticas (perceptron multicapa o redes de base radial) pueden utilizarse para el tratamiento de información temporal, a pesar de su carácter estático. Introduciendo el vector $(y(k),\ldots, y(k-n_y), u(k),\ldots, u(k-n_u))$ como el patrón k-ésimo de entrada, el funcional F que determina la descripción externa de un proceso dinámico (ecuación 1.16), puede aproximarse utilizando estas estructuras, obteniendo un modelo NARMA de la forma:

$$\tilde{y}(k+1) = \tilde{F}(y(k),\ldots,y(k-n_y),u(k),\ldots,u(k-n_u),W_{\tilde{F}}) \qquad (1.21)$$

siendo $\tilde{y}(k+1)$ la salida predicha por la red de neuronas, $\tilde{F}$ una aproximación neuronal de F y $W_{\tilde{F}}$ el conjunto de pesos de la red.

Estos modelos son normalmente referidos como *modelos de identificación en serie-paralelo* (figura 1.3) y han sido tratados y utilizados en un gran número de trabajos y aplicaciones, como por ejemplo en (NAPA, 90), (BHMC, 90), (CHBI, 90), (WIMO, 91), (CHBI, 92), (TAOM, 92), (LENA, 95).

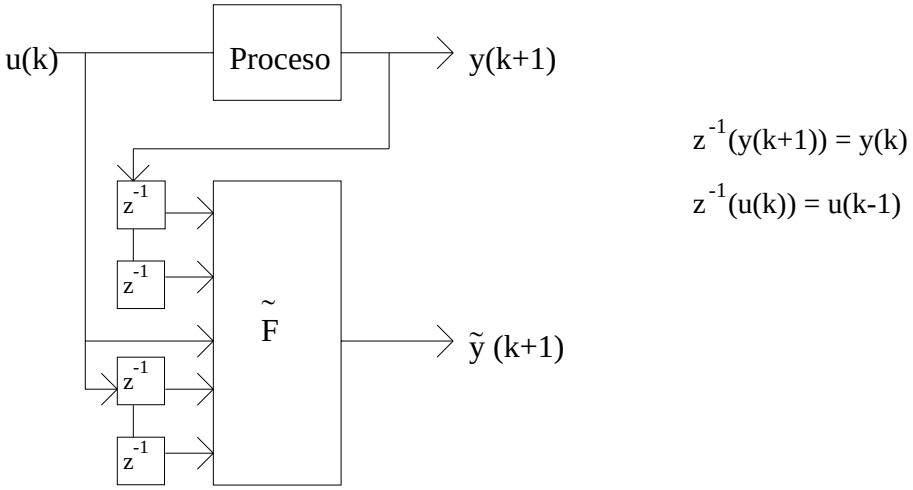


Figura 1.3. Modelos de identificación en serie-paralelo.

Es posible demostrar que la estructura de un modelo NARMA neuronal en el que la red de neuronas utilizada es un perceptron con sólo dos capas (la capa de entrada y la capa de salida) y con funciones de activaciones lineales coincide con un modelo lineal ARMA; los pesos de la red se corresponden con los parámetros del modelo lineal (BHMC, 90). Los modelos NARMA neuronales no son mas que una extensión de estos modelos lineales. Sin embargo, la incorporación de capas ocultas y funciones de activación no lineales permite obtener representaciones no lineales y por tanto aproximaciones más adecuadas a la dinámica del proceso.

Una variación de los modelos de identificación en serie-paralelo (ecuación 1.21) son los llamados *modelos de identificación en paralelo*, propuestos en (NAPA, 90).

Los autores construyen esta estructura utilizando el conjunto de parámetros o pesos, $W_{\tilde{F}}$, que resultan de entrenar el modelo de identificación en serie-paralelo dado por la ecuación 1.21 y sustituyendo los valores de la secuencia $y(k),\ldots,y(k-n_y)$ por los valores predichos por la red en instantes anteriores de tiempo (figura 1.4). El modelo NARMA se escribe de la siguiente forma:

$$\tilde{y}(k+1) = \tilde{F}(\tilde{y}(k),\ldots,\tilde{y}(k-n_y),u(k),\ldots,u(k-n_u),W_{\tilde{F}}) \qquad (1.22)$$

siendo $\tilde{y}(0) = \ldots = \tilde{y}(-n_y) = y(0)$.

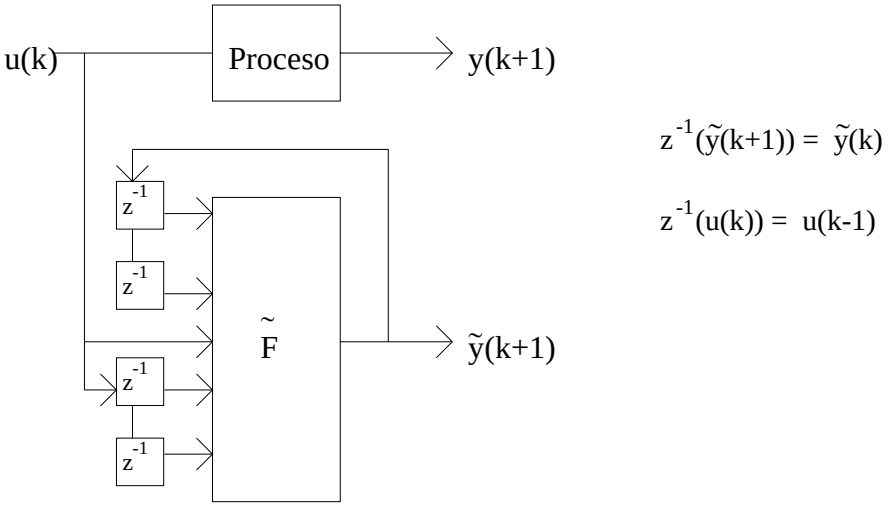


Figura 1.4. Modelos de identificación en paralelo.

Las redes de neuronas han intervenido también en la obtención de modelos de la forma dada por la ecuación 1.13. En este caso, e introduciendo respectivamente como patrones de entrada los vectores $(\mathbf{x}(k),u(k))$, $(\mathbf{x}(k))$, los funcionales T y L se aproximan utilizando una red de neuronas estática (POIO, 91), (SRPR, 94). El modelo adopta, entonces, la siguiente estructura:

$$\begin{aligned}\tilde{\mathbf{x}}(k+1) &= \tilde{T}(\mathbf{x}(k),u(k),W_{\tilde{T}})\\ \tilde{y}(k) &= \tilde{L}(\mathbf{x}(k),W_{\tilde{L}})\end{aligned} \qquad (1.23)$$

siendo $\tilde{T}$ y $\tilde{L}$ aproximaciones neuronales de T y L, respectivamente, y $W_{\tilde{T}}$ y $W_{\tilde{L}}$ los parámetros de cada una de las redes.

En la mayor parte de los casos los estados del proceso **x**(k) son variables auxiliares y por tanto, no medibles e inaccesibles, por lo que el modelo dado por la ecuación 1.23 tiene que adoptar la siguiente forma recurrente (NARE, 92):

$$\begin{aligned} \tilde{\mathbf{x}}(k+1) &= \tilde{T}(\tilde{\mathbf{x}}(k), u(k), W_{\tilde{T}}) \\ \tilde{y}(k) &= \tilde{L}(\tilde{\mathbf{x}}(k), W_{\tilde{L}}) \end{aligned} \qquad (1.24)$$

La construcción de modelos no lineales utilizando redes de neuronas recurrentes ha sido también estudiada por diferentes autores, entre ellos (YONI, 93), (PACH, 94), (KOPO, 95). Debido a la presencia de conexiones retroalimentadas en las redes recurrentes, no es necesario representar la información temporal utilizando una secuencia finita discreta en el pasado de las variables de entrada y salida del proceso para construir el modelo, como en el caso de los modelos de identificación en serie-paralelo (ecuación 1.21); basta utilizar como entradas a la red los valores actuales de las variables de entrada y salida del proceso. De este modo, en los modelos construidos con redes recurrentes, el comportamiento dinámico viene dado por la propia arquitectura de la red.

Con respecto a la fase de identificación de los modelos construidos con redes de neuronas artificiales, únicamente se hace notar que, dicha fase es equivalente a realizar el aprendizaje o entrenamiento de la red, utilizando como función error la definida por la ecuación 1.19.

1.2.5 Control de procesos dinámicos

Para conseguir que la salida de un proceso dinámico evolucione en una dirección deseada, como por ejemplo mantener una cierta condición estable o seguir una trayectoria especificada, es necesario incluir un sistema de control o controlador. Dicho sistema influirá en la dinámica del proceso, regulando el comportamiento de su salida para alcanzar así el objetivo deseado. Esta regulación se lleva a cabo manipulando la variable de entrada al proceso, también referida en este contexto como señal o acción de control.

El sistema de control es el encargado de proporcionar la acción de control que hay que aplicar al proceso para que la salida de dicho proceso tienda de manera asintótica al objetivo de control o valor deseado. Dicho objetivo puede depender del tiempo, $y^{des}(k)$, o ser un valor fijo, en cuyo caso se denota como y^{des}.

Formulación del problema de control: Determinar la acción u(k) tal que $e_c(k+1) \approx 0$, siendo $e_c(k+1) = \left| y^{des}(k+1) - y(k+1) \right|$.

En la actualidad existe una gran variedad de técnicas para afrontar el problema del control de procesos dinámicos. A continuación se hace un breve repaso sobre algunas de las técnicas de control existentes.

Las técnicas de control clásicas fueron desarrolladas en los años 40 y 50, y casi todas ellas se centran en procesos dinámicos lineales con parámetros invariantes en el tiempo y con una única variable de entrada (BROG, 85). Entre los sistemas desarrollados en esta época cabe destacar el conocido *controlador PID* (proporcional-integral-derivativo) con parámetros constantes, pues ha sido la estrategia de control dominante durante décadas; de hecho se estima que casi el 80% de los controladores usados en la industria son controladores PID o variaciones de éste. La versión más usual del controlador PID viene dada por la siguiente ecuación:

$$u(t) = K \cdot \left(e(t) + \frac{1}{T_i} \cdot \int_{t_0}^{t} e(s)ds + T_d \cdot \frac{d}{dt} e(t) \right) \qquad (1.25)$$

siendo t el tiempo, $e(t) = y(t) - y^{des}(t)$ y K, T_i y T_d parámetros que tienen que ser determinados en función de las características del proceso que se pretende controlar. En la ecuación 1.25 el tiempo se considera continuo, aunque es posible obtener discretizaciones de dicha expresión utilizando métodos de aproximación. Estos métodos consisten, por ejemplo, en aproximar el término integral utilizando la regla del rectángulo y la parte derivada junto con la proporcional usando ecuaciones en diferencias hacia atrás (ASWI, 90).

La complejidad de los procesos reales, complejidad que se traduce principalmente en la presencia de relaciones no lineales, procesos con parámetros variantes en el tiempo y procesos con múltiples entradas y salidas, llevó a nuevos desarrollos

teóricos. En la década de los 60 se introdujeron nuevas técnicas de control (BROG, 85), entre las que se destacan el control óptimo y los sistemas de control adaptativos, aunque éstos últimos adquirieron un mayor desarrollo teórico en los años 70.

El control óptimo engloba una gran variedad de métodos, pero todos ellos se caracterizan porque utilizan una modelización del proceso en la que intervienen las variables de estado (ecuación 1.13 ó 1.14), lo cual permite considerar las interacciones entre todas las variables. En las estrategias de control óptimo, la acción de control se calcula para que un cierto índice de medida alcance un valor óptimo (ATFA, 66); para ello se utilizan algoritmos que incluyen técnicas como programación dinámica, programación lineal y cálculo de variaciones.

Por otra parte, el carácter cambiante de los procesos industriales dio lugar a los sistemas de control adaptativos (GOSI, 84), (SEED, 86), los cuales tienen como objetivo ajustar en tiempo real los parámetros del controlador con el fin de evitar que el rendimiento global del sistema de control se deteriore en presencia de variaciones en la dinámica del proceso. Entre ellos se destacan los llamados *sistemas adaptativos con modelo de referencia* (LAND, 74), cuya idea básica es causar en el proceso el comportamiento de un modelo de referencia dado. Otra línea de sistemas adaptativos, son los llamados *reguladores autoajustables* (ASWI, 73), cuya estrategia de control está enmarcada en la teoría de control óptimo. Sin embargo, y para poder evitar la complejidad de ésta última, la cual era poco práctica en el contexto de sistemas adaptativos, se eligió una ley de control más simple, denominada de *mínima varianza*. Una extensión posterior, en esta misma línea de desarrollo, dio origen a los llamados *controladores autoajustables* (CLGA, 75), los cuales consideran una extensión de la ley de mínima varianza. Aunque pertenecen a la clase de reguladores autoajustables, es necesario destacar también los controladores PID autoajustables, los cuales cambian automáticamente sus constantes (K, T_i y T_d) para que el comportamiento del controlador no se deteriore ante cambios en el proceso. Se han propuesto diferentes estructuras de controladores PID autoajustables, las cuales pueden encontrarse en (WITT, 79), (CASE, 83), (HAWK, 83), entre otras.

Otra familia de controladores son los llamados *controladores predictivos*, cuya idea inicial fue propuesta en (MART, 76). Posteriormente, dichos esquemas de control fueron desarrollados en (CLMO, 87), adquiriendo una buena aceptación por parte de

la industria, (sobre todo, en la industria química). Dichos controladores se caracterizan porque hacen uso de un modelo del proceso para predecir el comportamiento de dicho proceso en el futuro.

Una estrategia de control predictivo se formula, normalmente, como un problema de optimización, en cada instante de tiempo k, de la siguiente forma:

$$\min_{A}\left(\sum_{i=1}^{H}(y^{des}(k+i)-\tilde{y}(k+i))^2\right) \qquad (1.26)$$

siendo $A=\left\{(u(k),\ldots,u(K+N_u))\ / u(k+N_u)=u(k+N_u+1)=\ldots=u(k+H)\right\}$; H y N_u números naturales llamados, respectivamente, *horizontes de predicción* y *de control*, tales que $N_u \le H$; y $\tilde{y}(k+i)$, i=1,...,H, las predicciones sobre el horizonte de predicción proporcionadas por un determinado modelo. Si la predicción realizada por el modelo es correcta, esta estrategia de control puede aplicarse para que el proceso siga una trayectoria deseada o alcance un objetivo de control.

En cada instante de tiempo k, se calcula la secuencia de acciones de control $u(k),\ldots,u(k+N_u)$ que minimiza las diferencias a lo largo del horizonte de predicción entre las salidas deseadas, $y^{des}(k+i)$, y el comportamiento de un determinado modelo del proceso, $\tilde{y}(k+i)$. Una vez calculada la secuencia $u(k),\ldots,u(k+N_u)$, sólo la acción $u(k)$ se aplica al proceso; en el siguiente instante de tiempo k+1 se calcula una nueva secuencia de acciones de control.

Los sistemas de control mencionados anteriormente están muy desarrollados para procesos dinámicos lineales, lo cual implica el uso de leyes de control y modelos lineales. A pesar de que los procesos dinámicos reales son, en su mayor parte, no lineales, dichos sistemas de control han sido ampliamente utilizados, obteniendo, en algunos casos, resultados adecuados.

Sin embargo, la necesidad de mejorar el control de los procesos dinámicos requiere de sistemas y técnicas no lineales. En los últimos años ha habido un interés creciente en desarrollar dichos sistemas de control y, como consecuencia, ha surgido una gran variedad de resultados. Algunos ejemplos pueden encontrarse en (AGSE, 87), (ISID, 89), (WAYN, 91), (MOPA, 92), (EARA, 92), (ALAR, 92), (SOKR, 94), (PRKA, 94), (TOTA, 95), entre otras.

Las redes de neuronas artificiales han intervenido, también, en la construcción de sistemas de control no lineales, dando lugar a diferentes métodos, que serán analizados en el apartado 1.2.6.

Por otra parte, y siguiendo una línea diferente de investigación, se han desarrollado sistemas de control cuya actuación está basada en un conjunto de reglas, las cuales describen, en términos lingüísticos, el comportamiento del proceso. En este campo los sistemas expertos ocupan un lugar importante. Estos no actúan, generalmente, como controlador directo del proceso, sino más bien como supervisores o reguladores de algún controlador convencional elegido para controlar el proceso. Explicaciones más detalladas sobre la construcción de sistemas expertos y su aplicación al campo de control de procesos están dadas en (ASAN, 86), (HAST, 87), (ARZE, 89) entre otras.

Muchas aplicaciones de control requieren trabajar con leyes de control cualitativas, en cuyo caso es muy útil incorporar los conceptos de conjuntos difusos y función de pertenencia introducidos por (ZADE, 65) para escribir y combinar las reglas que determinan el comportamiento del proceso. Esto ha dado lugar a los llamados *controladores difusos* (SUGE, 85), (PEDR, 93), los cuales han tenido una gran aceptación en estos últimos años.

1.2.6 Redes de neuronas artificiales en el problema del control de procesos dinámicos no lineales

Al igual que en el campo de la modelización de procesos dinámicos, las redes de neuronas artificiales ocupan también un lugar importante en el desarrollo de técnicas de control no lineales, permitiendo la construcción de leyes de control no lineales.

Cuando se habla de control de procesos utilizando redes de neuronas, en la mayor parte de los casos se entiende que es una red de neuronas la encargada de calcular la acción de control que hay que aplicar al proceso para que se alcance un determinado objetivo de control.

La diversidad de los métodos radica, principalmente, en la forma de realizar el entrenamiento de la red que actúa como controlador del proceso. A continuación se describen las técnicas más utilizados para realizar dicho aprendizaje.

Uno de los métodos más simples consiste en copiar un controlador (WISM, 64). El entrenamiento de la red se realiza para que aprenda el comportamiento de un controlador ya existente, por lo que la salida deseada para la red de neuronas será la salida del controlador que se pretende copiar. Este método puede parecer carente de sentido, puesto que el controlador ya está disponible. Sin embargo, puede ocurrir que, por razones prácticas, como un alto esfuerzo computacional en tiempo real, el controlador que ya existe no pueda implementarse, en cuyo caso es útil disponer de una estructura equivalente.

Otras de las técnicas para llevar a cabo el entrenamiento del controlador neuronal son las estrategias de control inverso, las cuales, básicamente, consisten en entrenar una red de neuronas para que aprenda la dinámica inversa del proceso.

Suponiendo que la salida de un proceso dinámico se expresa de la forma $y = S(u)$, la relación inversa es una expresión de la forma $u = S^{-1}(y)$, verificándose que $S^{-1} \circ S = I$. Por tanto, mediante una estrategia de control inverso, la red de neuronas se utiliza para obtener aproximaciones del funcional S^{-1}.

Cuando se plantea un esquema de control inverso utilizando redes de neuronas se distinguen dos formas diferentes para llevar a cabo el entrenamiento de la red, denominadas *aprendizaje generalizado* y *especializado*.

La finalidad del aprendizaje generalizado (WIMC, 78), (PSSI, 88) es que la red aprenda la dinámica inversa del proceso en su totalidad utilizando un conjunto de datos representativos de dicha dinámica. A diferencia de cuando se aproxima la dinámica del proceso (problema de modelización), en este caso, la entrada al proceso actúa como salida deseada para la red, mientras que la salida del proceso es la entrada a la red, como se observa en la figura 1.5. El aprendizaje de la red se lleva a cabo utilizando la siguiente función error:

$$E_c^g = \frac{1}{N} \cdot \sum_{k=0}^{N-1} e_c^g(k) \qquad (1.27)$$

siendo N el número de datos experimentales disponibles de pares entrada-salida del proceso y $e_c^g(k) = \frac{1}{2} \cdot (u(k) - \tilde{u}(k))^2$, con $u(k)$ la entrada real al proceso y $\tilde{u}(k)$ la salida proporcionada por la red de neuronas.

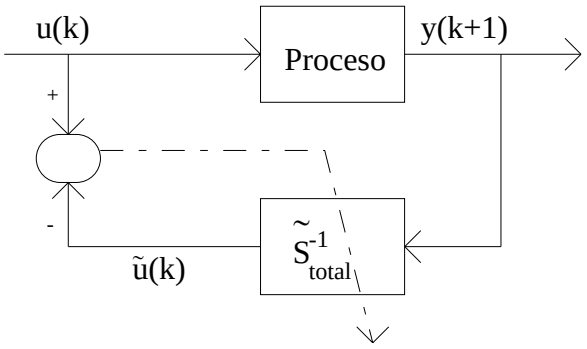


Figura 1.5. Esquema de aprendizaje generalizado.[1]

Una vez que la fase de aprendizaje ha concluido, la red de neuronas con pesos fijos se utiliza con el propósito de control, como se muestra esquemáticamente en la figura 1.6. Basta sustituir la entrada a la red, y(k+1), por el objetivo de control, $y^{des}(k+1)$.

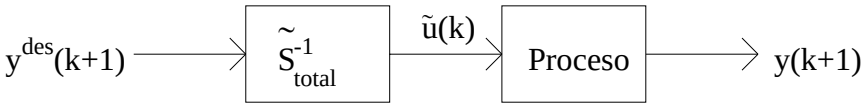


Figura 1.6. Red de neuronas actuando como controlador.

Mediante el aprendizaje especializado la red aproxima la inversa local del proceso, es decir, la inversa en una determinada región de interés y no en todo el rango de operación de la variable manipulada o acción de control. En este caso, no es necesario disponer de un conjunto de patrones salida-entrada del proceso, sino que los datos para el aprendizaje proceden de la evolución directa del proceso. La señal de control o salida de la red de neuronas se propaga a través del proceso, y la diferencia entre la salida actual del proceso y el objetivo de control se utiliza para llevar a cabo el aprendizaje de la red, como se indica en la figura 1.7. Como consecuencia, los pesos de la red de neuronas se adaptan utilizando la siguiente función error:

[1]La línea discontinua que aparece en las figuras se utiliza para indicar el error en base al cual se adaptan los pesos de la red de neuronas.

$$e_c^s(k+1) = \frac{1}{2} \cdot (y^{des}(k+1) - y(k+1))^2 \qquad (1.28)$$

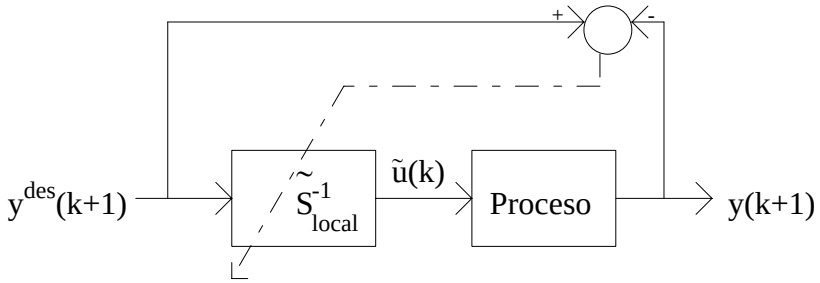


Figura 1.7. Esquema de aprendizaje especializado.

El aprendizaje especializado se refiere, también, en la literatura como métodos de control adaptativos, pues el aprendizaje de la red (o adaptación de los parámetros del controlador) se lleva a cabo en tiempo real. Existe una gran variedad de versiones y aplicaciones de este método al control de procesos dinámicos, las cuales pueden encontrarse en (WIDR, 86), (JORD, 89), (BHMC, 90), (JOJA, 90), (NAPA, 90), (POIO, 91), (TAOM, 92), (LOLE, 93), (JINI, 94), (PUFE, 94), (VEPA, 94), (NAMU, 95). En estos trabajos, tanto las redes estáticas como las redes de neuronas recurrentes se utilizan para la construcción de sistemas de control siguiendo una estrategia de control inverso local.

El aprendizaje del controlador se puede realizar también utilizando los llamados métodos de refuerzo. Estos, y al igual que los métodos adaptativos, hacen uso de una señal de error para realizar el entrenamiento del controlador, pero, en este caso, esta señal es una medida proporcionada por el entorno, llamada *medida de utilidad*, que evalúa, en un cierto sentido, el comportamiento global del controlador (MEMC, 70). A menudo, esta señal no está disponible en cada instante de tiempo, en cuyo caso es posible añadir un elemento adicional, llamado *elemento crítico adaptativo*, propuesto en (BASU, 83), cuyo propósito es predecir la señal de refuerzo cuando el entorno no la proporciona. En muchos casos, la señal de refuerzo juzga simplemente si el comportamiento del controlador es "bueno o malo", por lo que el aprendizaje puede ser muy lento; con vistas a solventar este problema en (WERB, 90) se introduce un modelo del proceso entre el elemento crítico y el controlador con la finalidad de poder

disponer de cierta información (derivadas) para acelerar el aprendizaje del controlador.

Las redes de neuronas artificiales han intervenido, también, en el desarrollo de sistemas de control predictivo no lineal, (WIMO, 91), (THGR, 91), (KAWA, 92), (DREN, 95), (BRAY, 96). En dichos esquemas, las redes actúan como modelo del proceso, proporcionando predicciones no lineales de la salida del proceso real a lo largo del horizonte de predicción.

Capítulo 2:

Planteamiento del problema

Este capítulo está dedicado a la presentación de los problemas que se resuelven en esta tesis. Dichos problemas están originados por las dificultades encontradas en determinados sistemas construidos con redes de neuronas artificiales para afrontar la modelización y el control de procesos dinámicos no lineales.

En la sección 2.1 se expone, en primer lugar, la necesidad de disponer de modelos capaces de actuar como simulador del proceso dinámico.

A continuación, se presentan los inconvenientes que poseen diferentes estructuras de modelos. Entre ellos, los modelos de identificación en serie-paralelo y los modelos de identificación en paralelo. Los primeros no poseen la propiedad de poder simular la dinámica del proceso sin su presencia. Para los segundos, se han encontrado una serie de limitaciones que impiden garantizar una aproximación adecuada de la dinámica del proceso. En esta sección, se hace entrever las causas de este hecho, aunque ellas serán analizadas y estudiadas, de forma más detallada, en el capítulo 3. En dicho capítulo se presentará, también, una solución que garantice un mejor simulador del proceso dinámico no lineal.

La sección 2.2 está dedicada a lo referente a las dificultades encontradas en los sistemas de control inverso y predictivo no lineales disponibles en la actualidad.

Se describen, en primer lugar, los factores que impiden el éxito de los diferentes métodos de control inverso neuronal, generalizado y especializado, para controlar un proceso dinámico real. Posteriormente, se analizan los inconvenientes que poseen los sistemas de control predictivo no lineal actualmente disponibles cuando tienen que

implementarse en tiempo real. Dichos problemas se resolverán en el capítulo 3, desarrollándose sistemas de control inverso y predictivo practicables en tiempo real y capaces de controlar de forma eficiente el proceso dinámico.

2.1 Simuladores de procesos dinámicos no lineales

La mayor parte de las técnicas de control desarrolladas en los últimos años se caracterizan porque están basadas en un modelo o representación matemática del proceso, y el éxito de las leyes de control depende, obviamente, de la capacidad del modelo para representar el proceso.

Sin embargo, en el contexto de control, no sólo es importante disponer de modelos no lineales que proporcionen una representación adecuada del proceso dinámico, sino que también es necesario que dichos modelos sean capaces de actuar como simulador de dicho proceso; es decir, el modelo debe predecir el comportamiento dinámico del proceso sin su presencia. Esto implica que, dado el estado inicial del proceso, y(0), y los valores de las variables de entrada al proceso en cada instante de muestreo k, u(k), k=0,...,K, el modelo debe aproximar la dinámica del proceso durante el intervalo de tiempo [0, K], utilizando únicamente dicha información.

En las aplicaciones de control aparecen, con bastante frecuencia, situaciones en las que los modelos tienen que simular la dinámica de un proceso real. En las estrategias de control predictivo en tiempo real, por ejemplo, el modelo tiene que proporcionar, en cada instante de muestreo k, predicciones de la salida del proceso en el intervalo de tiempo [k, k+H], $\tilde{y}(k+1),\ldots,\tilde{y}(k+H)$, para lo cual sólo se dispone de la información al tiempo k. En las técnicas de control que hacen uso de redes de neuronas artificiales, como las estrategias de control inverso local, generalmente, es necesario sustituir el proceso real por un simulador, con el objetivo de obtener estimaciones de los parámetros del controlador.

El primer objetivo planteado en esta tesis es la construcción de modelos no lineales que posean la propiedad de poder actuar como simulador del proceso dinámico. Para conseguir dicho objetivo, en la actualidad se disponen de diferentes estructuras, aunque ellas presentan una serie de inconvenientes, los cuales se analizan a continuación.

Los modelos físicos o modelos cuya expresión viene dada por la ecuación 1.13, son una posible alternativa para conseguir un simulador del proceso. En dicha ecuación se observa que, dado el estado inicial del proceso $\mathbf{x}_0$ y valores u(k) para k=0,...,K, el modelo puede proporcionar predicciones de la salida del proceso en dicho intervalo, $y(0),\ldots,y(K+1)$, siempre que T y L estén adecuadamente determinadas a partir de las leyes físicas que gobiernan el comportamiento del proceso. Sin embargo, y como ya se ha mencionado en el apartado 1.2.3, la construcción de dichos modelos puede ser laboriosa y, en general, son modelos complicados y difíciles de tratar, ya que interviene un gran número de ecuaciones diferenciales y algebraicas.

Ante la dificultad, en muchos casos, de obtener las funciones T y L a partir de las leyes físicas, es posible recurrir a las redes de neuronas artificiales para obtener aproximaciones de dichas funciones, como se comentaba en el apartado 1.2.4, resultando un modelo de la forma dada por la ecuación 1.23. No obstante y debido a que, en la mayor parte de los casos, las variables de estado **x**(k) no están disponibles, dicho modelo tiene que adoptar la estructura recurrente dada por la ecuación 1.24. Estos modelos pueden actuar como simulador del proceso, sin embargo, el aprendizaje de las redes que intervienen (es decir, la determinación de los conjuntos de parámetros $W_{\tilde{T}}$ y $W_{\tilde{L}}$) es normalmente complicado, lento y difícil de garantizar su convergencia hacia un mínimo global, debido, precisamente, a la estructura recurrente del modelo (LENA, 92).

Los modelos de identificación en serie-paralelo dados por la ecuación 1.21 (figura 1.3) son otra posible alternativa a la hora de construir un modelo del proceso dinámico. Dichos modelos poseen la ventaja de que son fáciles de construir y tratar. Sin embargo, presentan el inconveniente de que no pueden actuar como simulador del proceso, al menos en su presentación inicial. En la figura 1.3 se observa que para predecir la salida del proceso, $\tilde{y}(k+1)$, el modelo requiere del conocimiento de una secuencia discreta en el pasado de la variable de salida del proceso real, $y(k),\ldots,y(k-n_y)$, valores que no están disponibles cuando el modelo tiene que predecir la salida de dicho proceso sin su presencia.

Este problema podría, en principio, ser resuelto utilizando los modelos de identificación en paralelo dados por la ecuación 1.22 (figura 1.5), ya que, los valores reales, $y(k),\ldots,y(k-n_y)$, se sustituyen por los valores predichos por la red en instantes

anteriores de tiempo, $\tilde{y}(k),\ldots,\tilde{y}(k-n_y)$. Sin embargo, existen una serie de factores que impiden que dichos modelos proporcionen una buena representación del proceso dinámico.

Como se dijo en el apartado 1.2.4, los modelos de identificación en paralelo se construyen utilizando el conjunto de parámetros o pesos, $W_{\tilde{F}}$, que resultan de identificar el modelo de identificación en serie-paralelo (ecuación 1.21). Nótese que, dichos parámetros, $W_{\tilde{F}}$, han sido obtenidos al entrenar una red estática (perceptron multicapa) con patrones de entrada los vectores $I(k) = (y(k),\ldots,y(k-n_y),u(k),\ldots,u(k-n_u))$. Cuando se utiliza el modelo de identificación en paralelo (ecuación 1.22), los patrones de entrada a la red, $(\tilde{y}(k),\ldots,\tilde{y}(k-n_y),u(k),\ldots,u(k-n_u))$, son diferentes de los utilizados durante el aprendizaje. Claramente, si el perceptron multicapa de pesos $W_{\tilde{F}}$ proporciona para cada patrón I(k), aproximaciones $\tilde{y}(k+1)$ tales que $y(k+1)-\tilde{y}(k+1)\approx 0$, $\forall k$, y es capaz de filtrar los pequeños errores, el modelo dado por la ecuación 1.22 puede actuar como simulador del proceso, proporcionando una representación adecuada, o al menos aceptable, de dicho proceso. Sin embargo, factores como la falta de ejemplos en una determinada región del espacio entrada-salida del proceso, pueden provocar que una red de neuronas no siempre facilite aproximaciones indistinguibles de los valores reales para todos los patrones del conjunto de entrenamiento. En estos casos, y debido al carácter recursivo del modelo de identificación en paralelo, dicho simulador no tiene por qué proporcionar la mejor representación del proceso dinámico. Esta representación va a depender de la capacidad de la red para responder a patrones que no han sido utilizados durante el aprendizaje, como se verá en el apartado 3.2.1.

Ante esta situación, es conveniente desarrollar modelos fáciles de construir, como los modelos NARMA, y que a su vez sean capaces de actuar como simulador del proceso dinámico, proporcionando una representación adecuada de la dinámica de dicho proceso.

2.2 Control inverso y predictivo de procesos dinámicos no lineales en tiempo real

Como ya se ha mencionado, el desarrollo de técnicas de control no lineales es esencial para mejorar el control de los procesos dinámicos reales. Del mismo modo, es interesante también que dichas técnicas sean eficientes y aplicables en tiempo real.

En el apartado 1.2.6 se citaban algunos de los trabajos en los que las redes de neuronas artificiales intervienen en la construcción de sistemas de control no lineales. Esta tesis se centra en los esquemas de control inverso y predictivo. Dichos sistemas siguen presentando en la actualidad una serie de limitaciones e inconvenientes, sobre todo en lo referente a aplicaciones en tiempo real, las cuales se analizan a continuación.

Control inverso

En los sistemas de control inverso se distinguen dos caminos diferentes para realizar el entrenamiento del controlador neuronal: aprendizaje generalizado y aprendizaje especializado.

Los esquemas de control inverso con aprendizaje generalizado se caracterizan porque son sistemas de control off-line, en el sentido de que el aprendizaje de la red tiene que realizarse antes de que dicha red actúe como controlador del proceso en tiempo real. Existen, por tanto, dos factores que pueden impedir el éxito de dichos esquemas de control: los patrones disponibles para llevar a cabo el aprendizaje de la red y la capacidad de generalización de las redes de neuronas.

Para poder garantizar un buen control del proceso, la inversa global de dicho proceso tiene que estar representada en el conjunto de patrones de entrenamiento, lo cual requiere una gran cantidad de datos experimentales, difíciles de conseguir en la mayor parte de los casos. Por otra parte, y como se describía en el apartado 1.2.6, el aprendizaje de la red se lleva a cabo utilizando como entrada a la red la salida actual del proceso (figura 1.5) y no el objetivo de control deseado, por lo que el éxito del controlador dependerá de la capacidad de generalización de la red, es decir, de la capacidad para responder correctamente a entradas que no han sido especificadas en la fase de aprendizaje.

Incluso asumiendo que es posible obtener un conjunto de datos experimentales representativos de la dinámica inversa del proceso en su totalidad, y que las redes poseen buenas capacidades de generalización, el esquema de control inverso con aprendizaje generalizado no es aplicable a cualquier proceso dinámico. Esto es debido a que es condición necesaria que la dinámica inversa global del proceso esté bien

definida. Para procesos dinámicos en los que diversas entradas producen la misma salida carece de sentido aplicar una estrategia de aprendizaje generalizado. En estos casos, tanto si se usan redes de neuronas como cualquier otra herramienta para aproximar la relación inversa, la salida aproximada será un promedio de todas las salidas deseadas. Por tanto, la red entrenada con aprendizaje generalizado no tendrá éxito cuando actúe como controlador del proceso.

El esquema de control inverso con aprendizaje especializado se considera, a diferencia del anterior, un esquema de control on-line, en el sentido de que el aprendizaje de la red de neuronas que actúa como controlador se lleva a cabo mientras controla el comportamiento del proceso (figura 1.7). Es, por tanto, una aproximación más conveniente, pues la red aprende a conseguir un objetivo de control, el dictado por las necesidades de cada momento. Este tipo de aprendizaje permite que el controlador se vaya adaptando de acuerdo con los cambios que se producen en el proceso y el entorno y, por tanto, que la red de neuronas aprenda la dinámica inversa del proceso en la región de interés, resolviendo así el problema de la no existencia de la inversa global del proceso.

Sin embargo, y debido precisamente a que se trata de un esquema de control on-line, la inicialización de los pesos del controlador tendrá una repercusión importante en los resultados de control. Si, cuando se empieza a controlar el proceso real siguiendo el esquema mostrado en la figura 1.7, los pesos del controlador son aleatorios, la acción de control calculada por la red en los primeros instantes de muestro puede ser cualquiera. Esto no es permisible en aplicaciones en tiempo real, ya que dicho comportamiento puede tener consecuencias impredecibles.

Es necesario, por tanto, disponer de métodos que permitan obtener inicializaciones de los pesos del controlador de un esquema de control inverso con aprendizaje especializado, de forma que garanticen un control eficiente del proceso dinámico real. Además, es interesante que dichas inicializaciones no dependan del objetivo de control establecido, sino que sean válidas para cualquier situación que se pretenda controlar.

Control predictivo

Como se decía en el apartado 1.2.5, los sistemas de control predictivo se caracterizan porque hacen uso del comportamiento dinámico del proceso en el futuro

para determinar la acción de control en cada instante de tiempo. Este comportamiento en el futuro viene dado por un modelo, el cual en la mayor parte de las aplicaciones es un modelo lineal. La linealidad en los modelos utilizados, hace que las estrategias de control predictivo sean fácilmente practicables, ya que, el problema de optimización dado por la ecuación 1.26 es lineal.

Sin embargo, la capacidad de representación de los modelos lineales para procesos dinámicos no lineales es bastante limitada. Las acciones de control calculadas en base a una predicción lineal no tienen por qué ser las que produzcan un comportamiento óptimo en el proceso.

No obstante, y como se mencionaba en los apartados 1.2.5 y 1.2.6, existen ya en la actualidad trabajos en los que se introducen modelos no lineales en las estrategias de control predictivo. De este modo, cabe esperar un mejor rendimiento de los sistemas de control predictivo. Sin embargo, estos sistemas poseen fuertes inconvenientes cuando se trata de una aplicación en tiempo real.

Cuando un modelo no lineal forma parte de una estrategia de control predictivo (sea un modelo neuronal o de cualquier otra naturaleza), el problema de optimización dado en la ecuación 1.26 es no lineal y, por tanto, las ecuaciones de predicción no pueden resolverse de forma explícita. Esto implica que, en cada periodo de muestreo será necesario utilizar un método de optimización no lineal para disponer de la acción de control predictivo, u(k), como se muestra en la figura 2.1. Dichos métodos requieren, generalmente, de un gran número de adaptaciones de la acción de control hasta encontrar la solución óptima, adaptaciones que deberán realizarse en cada periodo de muestreo. Esto suele ser un serio inconveniente, principalmente, cuando se trata de aplicaciones de control en tiempo real, ya que, el esfuerzo computacional que debe ser soportado en cada instante de tiempo es muy alto.

Como consecuencia, los sistemas de control predictivo no lineal que se disponen en la actualidad son impracticables, en su mayor parte, en tiempo real. Es necesario, por tanto, disponer de estrategias de control predictivo no lineales, que sean aplicables en tiempo real y que requieran de un menor esfuerzo computacional que las disponibles en la actualidad.

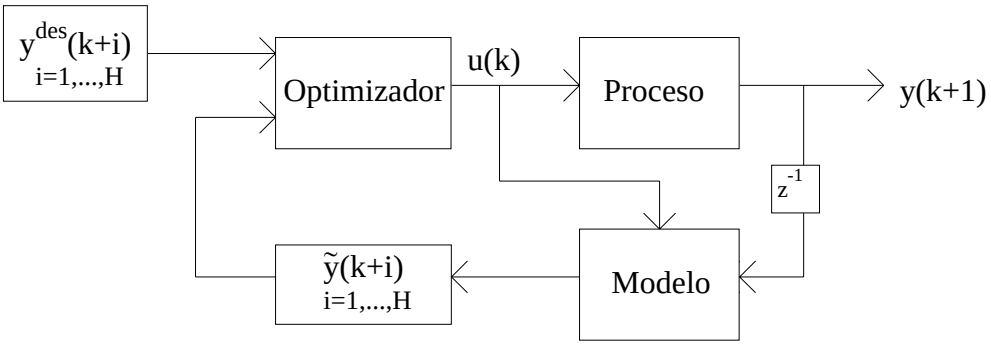


Figura 2.1. Esquema de control predictivo no lineal.

Capítulo 3:

Construcción de simuladores y controladores de procesos dinámicos no lineales

En este capítulo se proponen soluciones a los problemas planteados en el capítulo 2, tanto en lo referente a la construcción de modelos NARMA capaces de actuar como simulador del proceso, como a la construcción de sistemas de control inverso y predictivo competentes cuando se trata de una aplicación de control en tiempo real. De este modo, se consiguen los objetivos marcados al comienzo de esta tesis. El capítulo está organizado como se explica a continuación.

La sección 3.1 incluye, en primer lugar, un análisis comparativo de diferentes arquitecturas de redes de neuronas utilizadas en el contexto de la modelización y el control de procesos dinámicos, presentando ventajas e inconvenientes generales sobre cada una de ellas (apartado 3.1.1). Las arquitecturas comparadas son el perceptron multicapa, las redes de base radial y las redes recurrentes. El objetivo de este análisis es extraer ventajas y eliminar inconvenientes, con vistas a determinar una arquitectura adecuada para construir modelos y controladores no lineales.

En el apartado 3.1.2 se describe la arquitectura de red de neuronas que se propone utilizar, tanto para la construcción de simuladores como de controladores de procesos dinámicos no lineales. Dicha arquitectura es un tipo particular de red parcialmente recurrente, cuya consideración ha estado inspirada en la propia naturaleza del problema, así como de las conclusiones sacadas del estudio comparativo realizado previamente.

La sección 3.1 finaliza con el apartado 3.1.3, donde se presenta el algoritmo que se propone utilizar para llevar a cabo el entrenamiento de la red parcialmente recurrente, el cual sigue la misma línea de desarrollo que los algoritmos de retropropagación

dinámica, mencionados en el apartado 1.1.4. Puesto que la ley de aprendizaje desarrollada podría implicar un alto esfuerzo computacional, se estudia la posibilidad de utilizar una ley de aprendizaje más simple y equivalente a la dada por el algoritmo de retropropagación estático para entrenar un perceptron multicapa. Este apartado incluye también un método para acelerar el aprendizaje de la red parcialmente recurrente.

En la sección 3.2 se trata la construcción de simuladores NARMA, comenzando en el apartado 3.2.1 con una explicación detallada sobre las deficiencias que puede poseer el modelo de identificación en paralelo (ecuación 1.22), presentado en la sección 2.1, cuando se pretende simular la dinámica del proceso. Se deduce que este modelo podría proporcionar una representación no aceptable ni conveniente, ya que sus parámetros no han sido identificados con el propósito de actuar como simulador del proceso.

En el apartado 3.2.2 se propone una solución para poder garantizar una representación adecuada del proceso cuando se utilizan modelos de identificación en paralelo. Esta solución consiste, básicamente, en aproximar el funcional no lineal F que determina el modelo NARMA utilizando la red parcialmente recurrente.

Finalmente, la sección 3.3 contiene los sistemas de control inverso y control predictivo propuestos en esta tesis. El esquema de control inverso se desarrolla en el apartado 3.3.1, el cual incluye, en primer lugar, un camino para determinar la estructura de la ley de control inverso, es decir, el número de entradas a la red de neuronas que actúa como controlador y su topología. Posteriormente, se describe el algoritmo de aprendizaje del controlador, el cual se divide en tres fases diferentes: aprendizaje generalizado, aprendizaje especializado utilizando un modelo del proceso y aprendizaje especializado utilizando el proceso real. Esta división se realiza con la finalidad de obtener inicializaciones adecuadas de los pesos del controlador inverso neuronal que actúa en tiempo real.

El apartado 3.3.2 contiene el desarrollo del sistema de control predictivo propuesto. Dicho sistema se caracteriza porque la acción de control en cada instante de tiempo viene dada por la salida de una red de neuronas y no como resultado de un algoritmo de optimización no lineal, evitando así la resolución en cada periodo de muestreo del problema de optimización no lineal dado por la ecuación 1.26. En primer lugar, se justifica la presencia de dicha red. Seguidamente, se describe el algoritmo para llevar a

cabo el aprendizaje del controlador predictivo neuronal, el cual consiste en adaptar los pesos de la red para realizar una estrategia de control predictivo, es decir, siguiendo la dirección negativa del gradiente de la diferencia entre la salida del modelo y el objetivo de control durante el horizonte de predicción.

3.1 Red de neuronas parcialmente recurrente

Como se veía en los apartados 1.2.4 y 1.2.6, han sido utilizadas diferentes arquitecturas de redes de neuronas artificiales para afrontar la modelización y el control de procesos dinámicos no lineales. Antes de pasar a la descripción de la red propuesta para construir simuladores y controladores de procesos dinámicos no lineales, se ha creído conveniente analizar las características de las redes más empleadas en este contexto, ya que la construcción de la estructura de red utilizada en este trabajo no sólo ha estado inspirada en la propia naturaleza del problema, sino en las ventajas y limitaciones que poseen ciertas arquitecturas.

3.1.1 Análisis de diferentes arquitecturas de redes de neuronas artificiales

Las redes de neuronas más utilizadas en aplicaciones de modelización y control de procesos dinámicos son las llamadas redes estáticas, las cuales incluyen, como se indicaba en el apartado 1.2.3, el perceptron multicapa y las redes de base radial. La popularidad de dichas arquitecturas no sólo es debido al hecho de que poseen la propiedad de ser aproximadores universales, propiedad imprescindible cuando se trata el problema de la modelización y el control, sino que también es debido a la existencia de algoritmos de aprendizaje específicos para estas arquitecturas que permiten un entrenamiento sencillo y con éxito en la mayor parte de los casos.

El perceptron multicapa y las redes de base radial poseen sus neuronas agrupadas en capas y las conexiones están dirigidas hacia adelante (capa de entrada-capas ocultas-capa de salida). La diferencia entre ellas radica, principalmente, en la función de activación de sus neuronas. Las primeras utilizan funciones sigmoidales, las cuales se caracterizan porque poseen carácter global, mientras que las segundas emplean funciones de activación con carácter local, como son las llamadas funciones o

campanas de Gauss. Esto hace que cada una de estas arquitecturas tengan sus propias características y, por tanto, sus ventajas y limitaciones, las cuales se analizan brevemente a continuación.

Las características que se muestran en este apartado se han concluidos después de realizar previas simulaciones con las redes de base radial para modelizar y controlar procesos dinámicos no lineales, así como del estudio realizado en (GAZA, 96), donde se comparan el perceptron multicapa y las redes de base radial.

La principal desventaja del perceptron multicapa es que el aprendizaje puede ser lento, en comparación con el aprendizaje de las redes de base radial. Los factores que provocan este hecho son, en primer lugar, que los pesos del perceptron multicapa aparecen de forma no lineal con respecto a la salida, lo cual implica que el aprendizaje de la red tiene que realizarse utilizando métodos de optimización no lineal. En segundo lugar, y debido al carácter global de las funciones de activación, el cambio en un sólo peso de la red provoca cambios en la salida para todas los patrones de entrada presentados, reduciéndose así el efecto de previos ciclos de aprendizaje y retardando la convergencia del algoritmo de aprendizaje.

Para las redes de base radial el aprendizaje es generalmente más rápido. Esto es debido, por una parte, al carácter local de las neuronas de la red, el cual viene dado por el uso de funciones de activación con forma de campana de Gauss. Dichas funciones producen un nivel de activación alto para entradas cercanas a sus centros, y bajos o nulos cuando los patrones de entrada están lejanos de dichos centros. Por tanto, el cambio en un sólo peso de red afecta únicamente a un determinado grupo de patrones de entrada, los pertenecientes a la misma clase o categoría de patrones. Por otra parte, los pesos de la red aparecen de forma lineal con respecto a la salida, por lo que se puede utilizar el método de los mínimos cuadrados para determinar analíticamente dichos pesos, evitándose así problemas de mínimos locales. Ambos factores contribuyen a acelerar el aprendizaje de la red, lo cual es interesante para aplicaciones en tiempo real.

Sin embargo, un importante inconveniente que presentan las redes de base radial es la localización de los centros de las funciones de activación gaussiana de las neuronas de la red. Dicha localización se lleva a cabo usando métodos para clasificar los patrones de entrada. Estos métodos, en general, requieren de una gran cantidad de información cuando los patrones que tienen que clasificarse pertenecen a espacios de dimensión superior a tres. Este hecho puede llegar a ser una seria limitación de las redes de base

radial cuando se utilizan para modelizar o controlar procesos dinámicos no lineales ya que, como se comentaba en el apartado 1.1.3, cuando las redes de neuronas estáticas (el perceptron multicapa y las redes de base radial) se utilizan para procesar patrones dinámicos, es necesario utilizar como patrón de entrada a la red un vector que contenga las variables de entrada y salida del proceso así como una secuencia discreta de valores pasados de dichas variables. En la mayor parte de los casos, esto supone un vector de entrada de dimensión bastante superior a tres, por lo que la localización de los centros de las funciones de Gauss puede llegar a ser laborioso y no recomendable. En este sentido, se considera que el perceptron multicapa puede ser, en general, una arquitectura más conveniente que las redes de base radial para tratar la modelización y el control de procesos dinámicos, incluso si el aprendizaje es más lento.

Las redes de neuronas recurrentes están adquiriendo una gran importancia en lo referente al desarrollo de modelos y controladores. Estas redes poseen la propiedad de poder evolucionar a lo largo del tiempo y dicha evolución viene dada por la propia arquitectura de la red, y no por los patrones de entrada, como en el caso de las redes estáticas citadas anteriormente.

Las redes de neuronas recurrentes procesan la información temporal utilizando los estados internos de la red. Por tanto, para la construcción de modelos y controladores utilizando estas arquitecturas no es necesario considerar ninguna historia de las variables de entrada y salida del proceso dinámico, basta utilizar como entradas a la red los valores actuales de dichas variables. Desde este punto de vista, las estructuras recurrentes son más adecuadas que las redes estáticas para tratar la modelización y el control de procesos, ya que estas últimas requieren de un mayor número de neuronas en la capa de entrada para representar adecuadamente el problema.

Sin embargo, simulaciones previamente realizadas con las redes de neuronas totalmente recurrentes muestran que ellas son significativamente mucho más difíciles de tratar y entrenar que el perceptron multicapa, debido esencialmente a la presencia de conexiones recurrentes. Estas conexiones hacen que el aprendizaje de las redes recurrentes sea muy sensible a factores como la inicialización de sus parámetros, o de los estados de las neuronas. A menudo, encontrar inicializaciones adecuadas que garanticen la convergencia de la red hacia un mínimo global, requiere de un gran número de intentos y simulaciones. Por otra parte, la presencia de conexiones

recurrentes hace que los algoritmos de aprendizaje para estas redes requieran normalmente de un alto esfuerzo computacional, no sólo debido al hecho de que existe un mayor número de parámetros ajustables por neurona, sino también debido a que, para adaptar un peso de la red, es necesario tener en cuenta la variación de todos los estados internos de la red con respecto a dicho parámetro, lo cual implica, normalmente, un alto esfuerzo computacional cuando se implementa el algoritmo de aprendizaje.

Como consecuencia, en muchas ocasiones puede ser preferible recurrir al perceptron multicapa, a pesar de que podría llegar a requerir un gran número de neuronas en la capa de entrada. Por otra parte, es necesario tener presente que, en aplicaciones prácticas, a menudo es suficiente considerar una historia de las variables de entrada y salida del proceso de longitud no excesivamente grande para la construcción de modelos y controladores.

De lo expuesto anteriormente se podría concluir que las redes multicapas con funciones de activaciones sigmoidales (perceptron multicapa) son, en general, estructuras atractivas para el análisis y la síntesis de procesos dinámicos, lo cual no sólo es debido a la topología de dichas redes, sino también a lo relativo al aprendizaje de la red. No obstante, y como ya se ha mencionado, dichas redes son capaces de procesar información temporal debido esencialmente a los patrones de entrada, los cuales recogen una historia de las variables de entrada y salida del proceso. Esto puede suponer una limitación, y no es precisamente que el número de neuronas requerido en la capa de entrada sea grande, sino que existen situaciones en las que dicha historia no está disponible, como es el caso de cuando se pretende construir un modelo que actúe como simulador del proceso, como se indicaba en la sección 2.1.

Motivado por la complejidad de las redes totalmente recurrentes y por la necesidad de disponer de redes de neuronas capaces de procesar información temporal sin necesidad de recurrir a los patrones de entrada procedentes del exterior para representar dicha información, en esta tesis se propone utilizar una arquitectura particular de red parcialmente recurrente para el desarrollo de simuladores y controladores de procesos dinámicos no lineales. Dicha arquitectura modifica la del perceptron multicapa, mediante la introducción de un número suficiente de conexiones recurrentes, con vistas a que la capacidad de la red para representar información temporal venga dada por la propia arquitectura y no por los patrones de entrada. De este modo, será posible afrontar

problemas como la construcción de modelos capaces de actuar como simulador del proceso. Las conexiones recurrentes no llevan asociado ningún peso, por lo que el número de parámetros ajustables de la red coincide con los del perceptron multicapa de partida. Por otra parte, y a pesar de que se introducen conexiones recurrentes, se puede garantizar el éxito del aprendizaje de la red, ya que, debido a la similitud con el perceptron multicapa es posible obtener de forma sencilla inicializaciones de los parámetros de la red parcialmente recurrente que aceleren su convergencia y su aprendizaje, como se verá en el apartado 3.1.3.

3.1.2 Descripción de la arquitectura de la red parcialmente recurrente

La arquitectura de red parcialmente recurrente se construye a partir de un perceptron multicapa al que se le añaden conexiones recurrentes (figura 3.1). Estas conexiones unen la neurona de salida de la red con ciertas neuronas de la capa de entrada, también llamadas *neuronas de contexto*.

Las neuronas de la red parcialmente recurrente, al igual que las del perceptron multicapa, están agrupadas en capas: la capa de entrada, la capa de salida y las capas ocultas. Por simplicidad, y debido a que en este trabajo se tratan procesos dinámicos con una variable de salida y de entrada, se supone que la red posee una única neurona en la capa de salida, aunque es posible extender dicha arquitectura para varias neuronas de salida.

En la capa de entrada se distinguen dos grupos de neuronas. El primero de ellos actúa como la entrada a la red propiamente dicha, recibiendo los patrones de entrada $I(k) = (I_1(k),\ldots,I_{n_1}(k))$ procedentes del exterior. El segundo grupo de neuronas de la capa de entrada representan las neuronas de contexto, las cuales funcionan como una memoria en la que se almacena el nivel de activación de la salida de la red en instantes anteriores de tiempo. Puesto que el propósito es reproducir activaciones de la salida de la red en las neuronas de contexto, las conexiones recurrentes añadidas al perceptron multicapa no llevan asociado ningún peso.

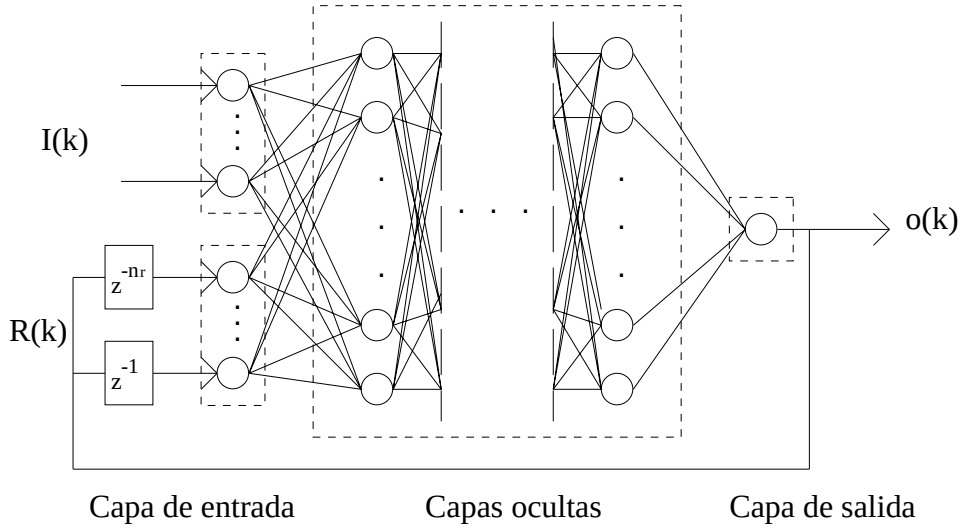


Figura 3.1. Arquitectura de red parcialmente recurrente.

El número de neuronas de contexto, n_r, está dado por la cantidad de valores anteriores de la salida de la red que necesiten ser almacenados para representar adecuadamente el problema, lo cual depende de la naturaleza del proceso dinámico que se pretende modelizar y controlar.

Introduciendo la notación $R(k) = (R_1(k),\ldots,R_{n_r}(k))$ para indicar los valores que reciben y memorizan cada neurona de contexto, las componentes del vector R(k) se calculan de la siguiente forma:

$$R_i(k) = z^{-i}(o(k)) \quad i=1,...,n_r \tag{3.1}$$

siendo o(k) la respuesta de la red para el patrón k-ésimo y z^{-i} un operador actuando sobre dicha respuesta y definido como:

> Dada $x(1), x(2),\ldots, x(n),\ldots$ una secuencia discreta, el operador z^{-i} retrasa en i unidades el término de la secuencia discreta al que se le aplica, es decir: $z^{-i}(x(n)) = x(n-i)$.

Por tanto, cuando el patrón k-ésimo se presenta a la red, cada neurona de contexto memoriza la respuesta de la red para los patrones k-1, k-2, ..., k-n_r presentados previamente.

Manteniendo la notación introducida en 1.2, las activaciones de las neuronas de la red parcialmente recurrente se expresan de la siguiente forma:

$$s_j^1(k) = I_j(k) \qquad j = 1,..,n_1 \tag{3.2}$$

$$s_{j+n_1}^1(k) = \begin{cases} t(k) & k \le n_r \\ R_j(k) & k \ge n_r + 1 \end{cases} \qquad j = 1,..,n_1 \tag{3.3}$$

con t(k) la salida deseada para el patrón de entrada I(k)

$$s_j^c(k) = \sigma\left(\sum_{i=1}^{n_{c-1}} w_{ji}^c \cdot s_i^{c-1}(k) + m_j^c \right) \qquad j = 1,...,n_c \ , \ c = 2,...,C \tag{3.4}$$

La salida de la red viene dada por el nivel de activación de la neurona en la última capa, $s^C(k)$:

$$o(k) = s^C(k) \tag{3.5}$$

La red parcialmente recurrente descrita anteriormente determina una correspondencia del espacio $\Re^{n_1}$ (espacio de los patrones de entrada a la red procedentes del exterior) al espacio $\Re$ (espacio de los patrones de salida) dada por $\tilde{F}_r$:

$$o(k) = \tilde{F}_r(I(k), R(k), W) \tag{3.6}$$

siendo W el conjunto de parámetros ajustables de la red.

Debido a que las conexiones recurrentes introducidas no llevan asociado un peso, el número de parámetros ajustables en la arquitectura propuesta coincide con el número de parámetros ajustables de un perceptron multicapa con n_1+n_r neuronas en la capa de entrada y con el mismo número de capas ocultas y neuronas en cada capa a la red recurrente. Por tanto, el grado de dificultad, en cuanto al número de parámetros que hay que adaptar para realizar el aprendizaje de la red, es el mismo para ambas estructuras.

La principal diferencia de la arquitectura presentada en este apartado con respecto a otras estructuras de redes parcialmente recurrentes radica en que la copia que reciben las neuronas de contexto no sólo se refiere a estados de la red para el patrón inmediatamente anterior, como es el caso de la red de Jordan, sino también a estados que provocan patrones presentados en un pasado más remoto. Esta consideración se basa en el hecho de que la salida de un proceso dinámico en un instante de tiempo no sólo depende de la salida en el instante anterior, sino que también depende de toda una historia de dicha variable, la cual es frecuentemente de longitud superior a 1.

3.1.3 Algoritmo de aprendizaje: retropropagación dinámica

Los pesos de la red parcialmente recurrente se adaptan para minimizar la función error de la forma dada por la ecuación 1.7 y utilizando un método del gradiente estocástico:

$$w^{(k)} = w^{(k-1)} - a \cdot \frac{\partial e(k)}{\partial w}, \ \forall w, \ \forall k \tag{3.7}$$

donde e(k) es el error local asociado al k-ésimo patrón y definido en la ecuación 1.8 para m=1. Siendo t(k) la salida deseada para el patrón k y o(k) la respuesta de la red parcialmente recurrente, los gradientes de los errores locales adoptan la siguiente forma:

$$\frac{\partial e(k)}{\partial w} = -(t(k) - o(k)) \cdot \frac{\partial o(k)}{\partial w} \tag{3.8}$$

Debido a la presencia de conexiones recurrentes en la red, el algoritmo de retropropagación estático tiene que ser modificado. La salida de la red para el k-ésimo patrón de entrada depende, no sólo de dicho patrón, sino también de las respuestas de la red para patrones previamente presentados, respuestas que a su vez dependen de los pesos. Por tanto, para calcular los términos $\frac{\partial o(k)}{\partial w}$ que intervienen en el gradiente de la

función error es necesario aplicar el concepto de derivada total presentado en el apartado 1.1.4.

El algoritmo de aprendizaje que se propone utilizar para llevar a cabo el entrenamiento de la red parcialmente recurrente sigue la misma línea de desarrollo que los citados en el apartado 1.1.4 y denominados *algoritmos de retropropagación dinámica,* por lo que podría incluirse dentro de este grupo. A continuación, y aplicando el concepto de derivada total de una función, se desarrolla el algoritmo de retropropagación dinámica para la arquitectura de red parcialmente recurrente.

Teniendo en cuenta la ecuación 3.6 y el hecho de que las componentes del vector R(k) dependen de los pesos de la red, la derivada total de la respuesta de la red o(k) respecto al parámetro w es:

$$\frac{\partial o(k)}{\partial w} = \frac{\partial \tilde{F}_r(I(k), R(k), W)}{\partial w} + \sum_{i=1}^{n_r} \frac{\partial \tilde{F}_r(I(k), R(k), W)}{\partial R_i(k)} \cdot \frac{\partial R_i(k)}{\partial w} \tag{3.9}$$

Teniendo en cuenta la definición de $R_i(k)$, se obtiene:

$$\frac{\partial R_i(k)}{\partial w} = \frac{\partial o(k-i)}{\partial w},\ i=1,...,n_r \tag{3.10}$$

Introduciendo la notación $x(k) = \frac{\partial o(k)}{\partial w}$, la variación de la salida de la red para el patrón k-ésimo puede obtenerse como la respuesta en el instante de tiempo k de un proceso dinámico discreto, cuya dinámica viene dada por la ecuación:

$$x(k) = \frac{\partial \tilde{F}_r(I(k), R(k), W)}{\partial w} + \sum_{i=1}^{n_r} \frac{\partial \tilde{F}_r(I(k), R(k), W)}{\partial R_i(k)} \cdot x(k-i) \tag{3.11}$$

y condiciones iniciales $x(0) = x(1) = ... = x(n_r) = 0$.

Los términos $\frac{\partial \tilde{F}_r(I(k), R(k), W)}{\partial w}$ y $\frac{\partial \tilde{F}_r(I(k), R(k), W)}{\partial R_i(k)}$ $i = 1, \ldots, n_r$ tienen que calcularse para cada patrón de entrada I(k). Debido a que la arquitectura interna de la

red parcialmente recurrente coincide con la de un perceptron multicapa (neuronas agrupadas en capas de distintos niveles), estos términos pueden implementarse aplicando retropropagación estática al perceptron multicapa correspondiente y con n_1+n_r neuronas de entrada.

Los pesos de la red se adaptan entonces utilizando la siguiente ley:

$$w^{(k)} = w^{(k-1)} + \alpha \cdot (t(k) - o(k)) \cdot x(k),\ \forall w,\ \forall k \qquad (3.12)$$

siendo x(k) la salida al instante de tiempo discreto k del proceso dinámico dado por la ecuación 3.11.

Comentarios sobre la ley de adaptación dada por la ecuación 3.12

La ley obtenida para llevar a cabo el aprendizaje de la red parcialmente recurrente puede llegar a requerir un alto esfuerzo computacional, principalmente cuando el número de neuronas de contexto es elevado, es decir cuando es necesario considerar un alto número de activaciones anteriores de la salida de la red para que la función que se pretende aproximar esté bien representada. Nótese que, para implementar la ley de aprendizaje dada por la ecuación 3.12, deberán calcularse n_r+1 derivadas parciales $\left(\frac{\partial \tilde{F}_r}{\partial w}, \frac{\partial \tilde{F}_r}{\partial R_1(k)}, \ldots, \frac{\partial \tilde{F}_r}{\partial R_{n_r}(k)}\right)$ (ecuación 3.11) para cada patrón k.

Por tanto, si se compara la ley de adaptación dada por la ecuación 3.12 con la ley para adaptar los pesos de un perceptron multicapa, se observa que el número de cálculos involucrados en la primera pueden llegar a ser altamente superior y no recomendable, sobre todo desde un punto de vista práctico.

Ante esta situación es importante disponer de métodos para adaptar los pesos de la red parcialmente recurrente que requieran un menor esfuerzo computacional.

Es posible, por ejemplo, aproximar la variación total de la salida de la red parcialmente recurrente, $\frac{\partial o(k)}{\partial w}$ (ver ecuación 3.9), por el término $\frac{\partial \tilde{F}_r}{\partial w}$, en cuyo caso se obtiene la ley de aprendizaje utilizada para adaptar los pesos de un perceptron multicapa:

$$w^{(k)} = w^{(k-1)} + \alpha \cdot (t(k) - o(k)) \cdot \frac{\partial \tilde{F}_r}{\partial w}, \ \forall w, \ \forall k \qquad (3.13)$$

Si el término $\frac{\partial \tilde{F}_r}{\partial w}$ en la ecuación 3.9 fuese dominante respecto al término $\sum_{i=1}^{n_r} \frac{\partial \tilde{F}_r}{\partial R_i(k)} \cdot \frac{\partial R_i(k)}{\partial w}$, la ley dada por la ecuación 3.13 podría actuar de forma satisfactoria. El esfuerzo computacional requerido por esta ley es, sin embargo, evidentemente menor, ya que, se cancelan los términos que recogen las variaciones de la salida de la red respecto a sus entradas. De hecho, los cálculos que implican la ley 3.13 son exactamente los mismos que los requeridos para adaptar los parámetros de un perceptron multicapa.

Desde un punto de vista teórico, al utilizar la ley dada por la ecuación 3.13, los pesos de la red parcialmente recurrente no se adaptan en base al verdadero gradiente de la función error, sino que la adaptación se realiza en base a una aproximación de dicho gradiente. Por tanto, la validez de esta ley para conseguir un aprendizaje eficiente de la red parcialmente recurrente depende de la bondad de la aproximación, es decir, de cuánto de despreciable sea el término $\sum_{i=1}^{n_r} \frac{\partial \tilde{F}_r}{\partial R_i(k)} \cdot \frac{\partial R_i(k)}{\partial w}$ con respecto al término $\frac{\partial \tilde{F}_r}{\partial w}$, lo cual va a variar en cada contexto específico.

No obstante, desde un punto de vista práctico puede ser interesante disponer de leyes que aunque no realicen un aprendizaje óptimo de la red, si requieran de un menor esfuerzo computacional.

En las simulaciones realizadas en este trabajo, se ha observado que las ventajas que ofrece la ley de adaptación dada por la ecuación 3.12 con respecto a la ley simplificada (ecuación 3.13) consisten, normalmente, en conseguir un mínimo en un menor número de iteraciones o ciclos de aprendizaje. Sin embargo, este hecho puede no ser importante ni dominante cuando el esfuerzo computacional que involucra es grande.

Generalmente, las condiciones específicas de cada problema son las que determinan si es necesario utilizar el verdadero gradiente para llevar a cabo un aprendizaje adecuado de la red parcialmente recurrente (ecuación 3.12) o basta utilizar una

aproximación (ecuación 3.13). La elección de una u otra ley va a depender, en la mayor parte de los casos, de la magnitud de n_r. Para un número pequeño de neuronas de contexto, es conveniente utilizar la ley dada por la ecuación 3.12 ya que no será necesario soportar un gran esfuerzo computacional; cuando el número de neuronas de contexto sea elevado es preferible recurrir a la ley dada por la ecuación 3.13 puesto que las ventajas que proporciona el uso del verdadero gradiente de la función error para adaptar los pesos de la red (ley dada por la ecuación 3.12) pueden no ser importantes con respecto al esfuerzo computacional que involucra su cálculo.

Acelerando la convergencia del algoritmo

Como ya se ha mencionado, las redes de neuronas recurrentes presentan, generalmente, el inconveniente de que el aprendizaje puede ser laborioso, en el sentido de que puede requerir un gran número de iteraciones para alcanzar la convergencia. Uno de los factores más influyentes en esta convergencia es el conjunto de pesos con el que se inicializa la red. Debido a la presencia de conexiones recurrentes, una inicialización con pesos aleatorios puede dificultar el aprendizaje, por lo que, en la mayor parte de los casos, es interesante disponer de inicializaciones que aceleren la convergencia, o incluso que eviten situaciones de no convergencia.

Esta dificultad en el aprendizaje podría también suceder cuando se entrena la red parcialmente recurrente considerada en este trabajo, e independientemente de la ley utilizada para adaptar los pesos (ecuaciones 3.12 o 3.13), ya que se incluyen conexiones recurrentes. Por esta razón, se cree conveniente disponer de inicializaciones para los pesos de la red que posean información sobre la relación que se pretende aproximar, de modo que pueda acelerarse el aprendizaje de la red.

Debido a la equivalencia entre la arquitectura considerada y el perceptron multicapa es posible, siempre que sea necesario, obtener de forma sencilla un conjunto de pesos para inicializar la estructura recurrente. A continuación se describe el procedimiento a seguir.

Puesto que la fase de aprendizaje de la red se lleva a cabo de forma supervisada, está disponible la respuesta deseada para cada patrón de entrada, t(k), y por tanto también sus valores anteriores $t(k-1), t(k-2), \ldots, t(k-n_r)$. Los pesos obtenidos después de entrenar un perceptron multicapa con patrones de entradas, los vectores

$(I_1(k),\ldots,I_{n_1}(k),t(k-1),t(k-2),\ldots,t(k-n_r))$, y con el mismo número de capas ocultas y neuronas en cada capa que la red parcialmente recurrente, pueden utilizarse para inicializar la estructura parcialmente recurrente, ya que, tanto el número de parámetros ajustables como la organización de dichos parámetros, coinciden en ambas estructuras de redes.

Debido a que la red parcialmente recurrente consiste simplemente en sustituir los valores reales $t(k-1),t(k-2),\ldots,t(k-n_r)$ por los valores predichos por la red en instantes anteriores, $o(k-1),o(k-2),\ldots,o(k-n_r)$, los pesos obtenidos al entrenar el perceptron multicapa equivalente poseen ya información sobre la función que se pretende aproximar con la red parcialmente recurrente, de manera que dichos pesos podrán ayudar a acelerar la convergencia de la red.

Las simulaciones realizadas muestran que al inicializar los pesos de la red parcialmente recurrente con el método descrito anteriormente, la convergencia del algoritmo de aprendizaje queda garantizada.

Una vez inicializados los pesos, se procede al aprendizaje de la red parcialmente recurrente utilizando una de las leyes presentadas anteriormente (ecuaciones 3.12 o 3.13).

3.2 Simuladores NARMA utilizando redes de neuronas

En esta sección se trata la construcción de modelos NARMA neuronales capaces de simular adecuadamente la dinámica del proceso sin su presencia. Como se veía en el apartado 1.2.2, los modelos NARMA se caracterizan porque describen el comportamiento dinámico del proceso a partir de las variables medibles, de modo que para determinarlos no es necesario recurrir a las variables de estado del proceso, las cuales normalmente complican la construcción de modelos.

Se considera un proceso dinámico discreto no lineal con una variable de entrada y otra de salida. Se supone que la salida de dicho proceso posee un retraso natural d con respecto a la entrada, es decir, que el proceso requiere de d instantes de tiempo para reaccionar ante un cambio en su variable de entrada. Suponiendo que el proceso dinámico es observable, es decir, que existe una secuencia discreta finita formada por

valores en instantes diferentes de tiempo de las variables de entrada y salida, la salida de dicho proceso al instante actual k+1 puede expresarse como:

$$y(k+1) = F(y(k),\ldots,y(k-n_y),u(k-d),\ldots,u(k-d-n_u)),\ k\in N_0 \qquad (3.14)$$

con $y(-n_y) =\ldots= y(0) = y_0 \in \Re,\ u(-d - n_u) =\ldots= u(-1) = u(0)$.

Como se veía en el apartado 1.2.4, el perceptron multicapa puede utilizarse para aproximar el funcional F, obteniendo los modelos de identificación en serie-paralelo:

$$\tilde{y}(k+1) = \tilde{F}(y(k),\ldots,y(k-n_y),u(k-d),\ldots,u(k-d-n_u),W_{\tilde{F}}),\ k\in N_0 \qquad (3.15)$$

Existen situaciones en las que el modelo, una vez que ha sido identificado, tiene que actuar como simulador del proceso, es decir, situaciones en las que la secuencia de valores $y(k),\ldots,y(k-n_y)$ no está disponible. En estos casos, los modelos de identificación en serie-paralelo no pueden utilizarse, sino que, como se indicaba en la sección 2.1, es necesario recurrir a los modelos de identificación en paralelo, sustituyendo los valores reales $y(k),\ldots,y(k-n_y)$ por los valores predichos por la red en instantes anteriores de tiempo. Introduciendo la notación $\tilde{y}_p(k+1)$ para indicar la salida en el instante de tiempo k+1 del modelo de identificación en paralelo construido con el conjunto de pesos $W_{\tilde{F}}$, dicho modelo se escribe de la forma:

$$\tilde{y}_p(k+1) = \tilde{F}(\tilde{y}_p(k),\ldots,\tilde{y}_p(k-n_y),u(k-d),\ldots,u(k-d-n_u),W_{\tilde{F}}),\ k\in N_0 \qquad (3.16)$$

con $\tilde{y}_p(-n_y) =\ldots= \tilde{y}_p(0) = y(0) = y_0 \in \Re,\ u(-d - n_u) =\ldots= u(-1) = u(0)$.

En la sección 2.1 se comentaban brevemente las dificultades que puede presentar el modelo en paralelo (ecuación 3.16) para simular adecuadamente la dinámica del proceso. A continuación se analizan dichas dificultades, con vistas a proponer una solución que garantice una aproximación adecuada del proceso dinámico cuando se utilizan modelos de identificación con estructura en paralelo, es decir, cuando se pretende simular la dinámica del proceso con modelos NARMA.

3.2.1 Incapacidad del modelo dado por la ecuación 3.16 para representar adecuadamente un proceso dinámico

Como se indicaba en el apartado 1.2.4, el conjunto de pesos $W_{\tilde{F}}$ se obtiene al aproximar el funcional F que determina el modelo NARMA por un perceptron multicapa, de modo que dichos parámetros se calculan para minimizar la siguiente función error, también llamada en este contexto *error de identificación* del modelo:

$$E_m^I = \frac{1}{2N} \cdot \sum_{k=0}^{N-1} (y(k+1) - \tilde{y}(k+1))^2 \qquad (3.17)$$

siendo $y(k+1)$ la salida real del proceso al instante de tiempo discreto k+1, $\tilde{y}(k+1)$ la salida predicha por el modelo de identificación en serie-paralelo dado por la ecuación 3.15 en dicho instante de tiempo y N el número de datos experimentales de entrada-salida del proceso dinámico $\{u(k),y(k+1)\}$.

Las aproximaciones $\tilde{y}(k+1)$ serán más o menos exactas, dependiendo de los patrones disponibles para el aprendizaje y de la capacidad de la red para aproximar el funcional F. No obstante, en cualquier caso sí serán las mejores aproximaciones obtenidas con un modelo de identificación en serie-paralelo, ya que los parámetros o pesos que las determinan se han calculado para que el error de identificación definido en 3.17 sea mínimo.

Debido a que para el modelo en paralelo (ecuación 3.16) los patrones de entrada a la red son diferentes de los que utiliza el modelo en serie-paralelo (ecuación 3.15), y diferentes, por tanto, a los utilizados para obtener el conjunto de pesos $W_{\tilde{F}}$, la cuestión que se plantea es si las aproximaciones $\tilde{y}_p(k+1)$, $k = 0,\ldots,N-1$, obtenidas con el modelo de identificación en paralelo dado por la ecuación 3.16 proporcionan una representación adecuada del proceso dinámico. En lo que sigue, se ilustra la validez de estas aproximaciones.

Por simplicidad de las expresiones que aparecen a continuación, se supone que n_y=0 y n_u=0, aunque en el caso general el análisis es similar. Los modelos de identificación en serie-paralelo y paralelo se escriben, entonces, de la siguiente forma:

$$\tilde{y}(k+1) = \tilde{F}(y(k), u(k-d), W_{\tilde{F}}) \tag{3.18}$$

$$\tilde{y}_p(k+1) = \tilde{F}(\tilde{y}_p(k), u(k-d), W_{\tilde{F}}), \tag{3.19}$$

con $\tilde{y}_p(0) = y(0) = y_0 \in \Re$, $u(-d) = \ldots = u(-1) = u(0)$.

Las aproximaciones $\tilde{y}(k+1)$ proporcionadas por el modelo de identificación en serie-paralelo dado por la ecuación 3.18 se pueden expresar de la forma:

$$\tilde{y}(k+1) = y(k+1) + \varepsilon_{k+1},\ k = 0, \ldots, N-1 \tag{3.20}$$

siendo ε_{k+1} un número real que indica el error ocasionado por el modelo de identificación en serie-paralelo para el patrón de entrada $I(k) = (y(k), u(k-d))$.

> Dado $I(k) = (y(k), u(k-d))$ un patrón de entrada, $\tilde{y}(k+1)$ la respuesta del perceptron multicapa de pesos $W_{\tilde{F}}$ y ε un número real, la respuesta de dicha red para una perturbación del patrón I(k) de la forma $I_\varepsilon(k) = (y(k) + \varepsilon, u(k-d))$, se denota como $\tilde{y}(k+1) + \delta(\varepsilon)$, donde $\delta(\varepsilon)$ es un número real que evalúa la capacidad de la red de pesos $W_{\tilde{F}}$ para responder a dicha perturbación $I_\varepsilon(k)$.

Utilizando estas cantidades $\delta(\cdot)$, las aproximaciones $\tilde{y}_p(k+1)$ proporcionadas por el modelo en paralelo dado por la ecuación 3.19 pueden expresarse en función de los errores cometidos por el modelo de identificación en serie-paralelo 3.18, ε_{k+1}, de la siguiente forma:

$$\tilde{y}_p(1) = \tilde{y}(1) = y(1) + \varepsilon_1 \tag{3.21}$$

$$\begin{aligned}\tilde{y}_p(k+1) &= \tilde{y}(k+1) + \delta\left[\varepsilon_k + \delta(\varepsilon_{k-1} + \delta(\varepsilon_{k-2} + \ldots + \delta(\varepsilon_1)\ldots))\right] = \\ &= y(k+1) + \varepsilon_{k+1} + \delta\left[\varepsilon_k + \delta(\varepsilon_{k-1} + \delta(\varepsilon_{k-2} + \ldots + \delta(\varepsilon_1)\ldots))\right]\end{aligned} \tag{3.22}$$

para $k = 1, \ldots, N-1$

siendo $\delta(\varepsilon_1)$, $\delta(\varepsilon_2 + \delta(\varepsilon_1)), \ldots, \delta\left[\varepsilon_k + \delta(\varepsilon_{k-1} + \delta(\varepsilon_{k-2} + \ldots + \delta(\varepsilon_1)\ldots))\right]$ números reales que miden la capacidad del perceptron multicapa de pesos $W_{\tilde{F}}$ para responder a perturbaciones de los patrones de entrada $I(1)$, $I(2), \ldots, I(k)$, respectivamente.

En efecto, la igualdad 3.21 es inmediata puesto que $\tilde{y}_p(1) = \tilde{y}(1) = y(1) + \varepsilon_1$.

Para demostrar la igualdad 3.22 se procede por inducción. Veamos que es cierta para k=1.

$$\tilde{y}(2) = \tilde{F}(y(1), u(0), W_{\tilde{F}}) = y(2) + \varepsilon_2$$

$$\begin{aligned}\tilde{y}_p(2) &= \tilde{F}(\tilde{y}_p(1), u(0), W_{\tilde{F}}) = \tilde{F}(y(1) + \varepsilon_1, u(0), W_{\tilde{F}}) = \\ &= \tilde{y}(2) + \delta(\varepsilon_1) = y(2) + \varepsilon_2 + \delta(\varepsilon_1)\end{aligned}$$

Suponiendo que es cierta para k, se obtiene que:

$$\tilde{y}(k+1) = \tilde{F}(y(k), u(k-d), W_{\tilde{F}}) = y(k) + \varepsilon_k$$

$$\begin{aligned}\tilde{y}_p(k+1) &= \tilde{F}(\tilde{y}_p(k), u(k-d), W_{\tilde{F}}) = \\ &= \tilde{F}(y(k) + \varepsilon_k + \delta(\varepsilon_{k-1} + \delta(\varepsilon_{k-2} + \ldots + \delta(\varepsilon_1)\ldots)), u(k-d), W_{\tilde{F}}) = \\ &= \tilde{y}(k+1) + \delta\left[\varepsilon_k + \delta(\varepsilon_{k-1} + \delta(\varepsilon_{k-2} + \ldots + \delta(\varepsilon_1)\ldots))\right] = \\ &= y(k+1) + \varepsilon_{k+1} + \delta\left[\varepsilon_k + \delta(\varepsilon_{k-1} + \delta(\varepsilon_{k-2} + \ldots + \delta(\varepsilon_1)\ldots))\right]\end{aligned}$$

es decir, es cierta para k+1. Por tanto, es posible afirmar que 3.22 es cierta para $k = 1, \ldots, N-1$.

En la ecuación 3.22 se observa que la validez de cada aproximación $\tilde{y}_p(k+1)$ depende no sólo del error cometido por el modelo de identificación en serie-paralelo dado por la ecuación 3.18 para el patrón I(k), ε_{k+1}, sino que también depende de los errores para todos los patrones presentados previamente, $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k$ y de la capacidad de la red para responder adecuadamente a patrones perturbados, es decir del valor que tome $\delta(\cdot)$ para errores que se van acumulando.

De acuerdo con la definición presentada anteriormente, si $\varepsilon = 0$ entonces $\delta(\varepsilon) = 0$. Para $\varepsilon \neq 0$, $\delta(\varepsilon)$ será un número real distinto de cero y su magnitud dependerá de la magnitud de ε y de la capacidad de la red para filtrar dicho error. Generalmente, la red produce buenos resultados para errores pequeños en sus patrones de entrada, es decir $\delta(\varepsilon)$ toma valores cercanos a cero cuando ε es pequeño. A medida que los errores en las entradas crecen, estos valores también crecen, y por tanto, la capacidad de la red para responder adecuadamente desaparece.

Este comportamiento puede observarse en los resultados experimentales que se muestran en la figura 3.2, en la que se representan las relaciones entre ε y $\delta(\varepsilon)$ para dos modelos de identificación en serie-paralelo y para diferentes patrones, elegidos de forma aleatoria de un conjunto de datos de entrenamiento. Dichos modelos se construyeron con la finalidad de aproximar la dinámica de la temperatura del fluido de intercambio de calor que circula en la camisa de un reactor químico y poseen estructura similar a los modelos de indentificación en serie-paralelo presentados en el apartado 4.2.1. La secuencia discreta para la variable de entrada al proceso, u(k), es de longitud 20 (n_u=20) en ambos modelos. Con respecto a la variable de salida del proceso, y(k), en el primer modelo n_y toma el valor 0, mientras que en el segundo modelo tiene el valor 1, lo cual implica que son dos valores, $y(k), y(k-1)$, los que influyen en la respuesta actual, $\tilde{y}(k+1)$.

Las gráficas que se muestran en la figura 3.2 se han obtenido presentado a los respectivos modelos, ya entrenados, perturbaciones de los diferentes patrones de entrada elegidos y midiendo el error que provoca dicha perturbación en el salida del modelo, $\delta(\varepsilon)$. Estas perturbaciones consisten en añadir errores crecientes a las variables de entrada de la red correspondientes a los valores pasados de la salida del proceso. En el

segundo modelo, dichas perturbaciones se han añadido separadamente a las variables y(k), y(k-1).

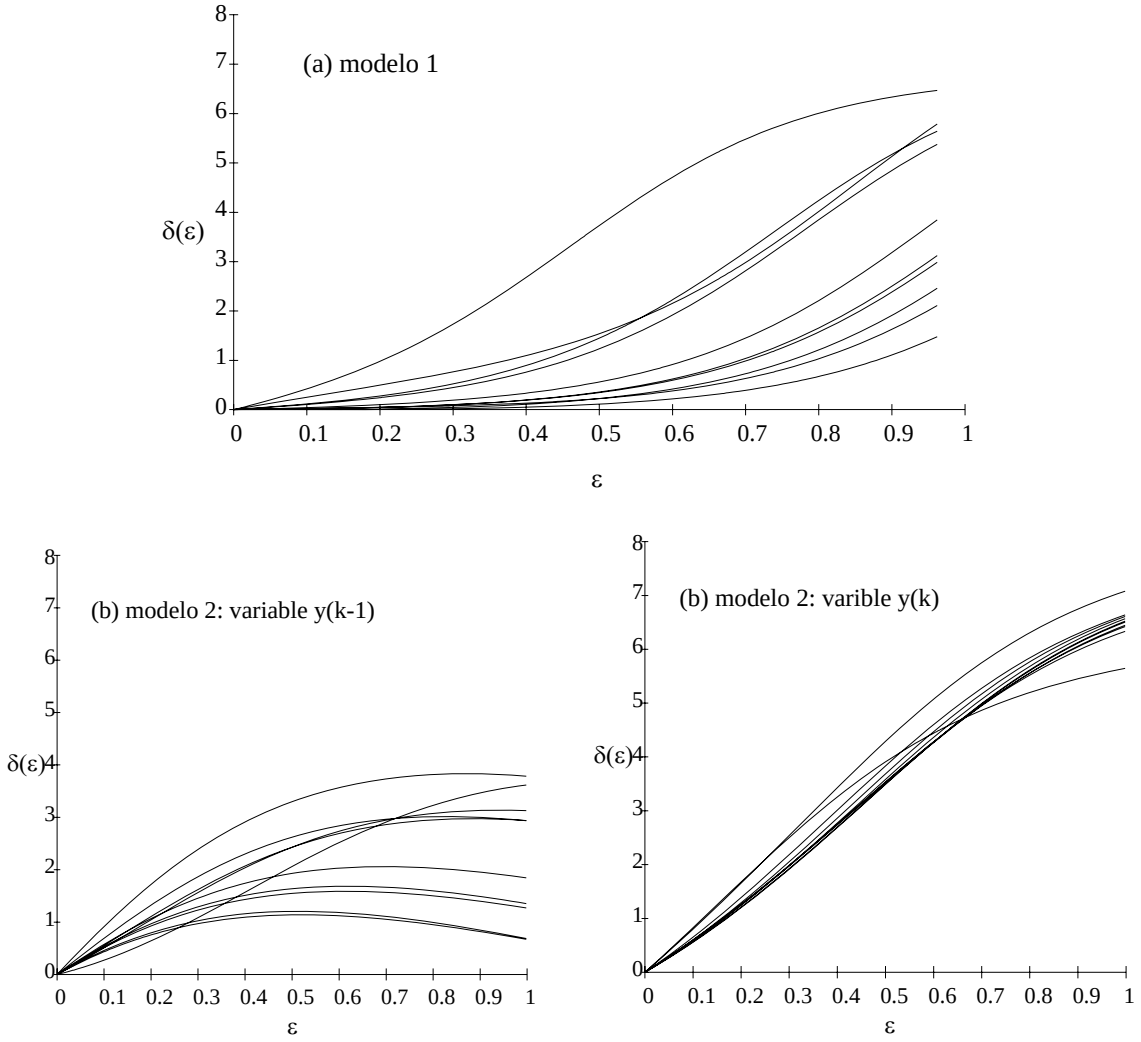


Figura 3.2. Relación entre ε y $\delta(\varepsilon)$ para diferentes modelos. (a) Modelo1 (b) Modelo2.

Se observa que el error en la salida del modelo crece a medida que los errores provocados en las entradas de la red se incrementan, por lo que la red va perdiendo la capacidad para producir respuestas adecuadas a los patrones de entrada perturbados con errores grandes, aunque no siempre se puede decir que la velocidad y la forma de crecimiento sean las mismas, depende de la estructura considerada.

Como consecuencia de estos resultados experimentales y de la igualdad 3.22, las siguientes conclusiones son inmediatas:

Caso 1º

Si los errores ε_k son nulos o cercanos a cero $\forall\, k = 1,\ldots,N$ o la red de pesos $W_{\tilde{F}}$ responde adecuadamente a las respectivas perturbaciones $I_{\varepsilon_1}(1), I_{\varepsilon_2+\delta(\varepsilon_1)}(2),\ldots,$ $I_{\varepsilon_N+\delta(\varepsilon_{N-1}+\delta(\varepsilon_{N-2}+\ldots+\delta(\varepsilon_1)\ldots))}(N)$ de los patrones de entrada $I(1), I(2),\ldots, I(N)$, entonces $\delta(\varepsilon_1), \delta(\varepsilon_2+\delta(\varepsilon_1)),\ldots, \delta\left[\varepsilon_N+\delta(\varepsilon_{N-1}+\delta(\varepsilon_{N-2}+\ldots+\delta(\varepsilon_1)\ldots))\right]$ serán valores cercanos a cero. De este modo y según la ecuación 3.22, las aproximaciones $\tilde{y}_p(k+1),\ k = 0,\ldots,N-1$, obtenidas por el modelo en paralelo, podrán considerarse predicciones apropiadas de la salida real del proceso dinámico.

Caso 2º

Si existe un número natural n tal que el error ε_n ocasionado por el modelo de identificación en serie-paralelo toma un valor lejano de cero, entonces el número real $\delta\left[\varepsilon_n+\delta(\varepsilon_{n-1}+\delta(\varepsilon_{n-2}+\ldots+\delta(\varepsilon_1)\ldots))\right]$ toma un valor grande y según la ecuación 3.22 la aproximación $\tilde{y}_p(n+1)$ difiere del valor real de la salida del proceso, $y(n+1)$, en el instante de tiempo n+1. Sin embargo, esto no supondría un problema si dicho error se produjese únicamente en dicho instante de tiempo, pues de sobra es sabido que las redes de neuronas no siempre proporcionan aproximaciones adecuadas para todos los patrones de entrada presentados. El problema es, principalmente, debido a que dicho error se propaga en los siguientes patrones de entrada y la red tendrá que filtrar errores cada vez mayores. En este caso, no se puede esperar una buena aproximación por parte de la red, como se observa en los resultados experimentales mostrados en la figura 3.2. De este modo, es posible que la capacidad del modelo en paralelo para producir aproximaciones $\tilde{y}_p(n+i)$ tales que $\tilde{y}_p(n+i) \approx y(n+i)$ para $i = 2,\ldots,N-1$ se destruya.

Por tanto, la presencia de un sólo patrón no adecuadamente aproximado por el modelo de identificación en serie-paralelo, la incapacidad de la red para responder correctamente ante patrones de entrada con errores, es decir patrones que no han sido utilizados durante el aprendizaje y el carácter recurrente del modelo de identificación en

paralelo dado por la ecuación 3.16, puede provocar un mal comportamiento de dicho modelo cuando simula la dinámica real del proceso.

Ejemplo:

Si se supone que la relación entre ε y $\delta(\varepsilon)$ viene dada por una ecuación de la forma $\delta(\varepsilon) = a \cdot \varepsilon^2$, con $a \approx 5$, como es el caso de los datos experimentales obtenidos con el primer modelo (figura 3.2a), según la ecuación 3.22 las aproximaciones de la salida del proceso dinámico proporcionadas por el correspondiente modelo en paralelo, adoptarían la forma:

$$\tilde{y}_p(k+1) = y(k+1) + \varepsilon_{k+1} + a^{n_1-1} \cdot \varepsilon_k^{n_1} + a^{n_2-1} \cdot \varepsilon_{k-1}^{n_2} + \ldots + a^{n_k-1} \cdot \varepsilon_1^{n_k} + P$$

siendo $n_i = 2^i$ y P una expresión polinómica que puede escribirse de la forma:

$$P = P\left(\varepsilon_k, \varepsilon_{k-1}, \varepsilon_{k-1}^{n_1}, \varepsilon_{k-2}, \varepsilon_{k-2}^{n_1}, \varepsilon_{k-2}^{n_2}, \ldots, \varepsilon_2, \varepsilon_2^{n_1}, \varepsilon_2^{n_2}, \ldots, \varepsilon_2^{n_{k-2}}, \varepsilon_1^{n_1}, \varepsilon_1^{n_2}, \ldots, \varepsilon_1^{n_{k-1}}\right)$$

donde $P(x_1, \ldots x_n) = \sum_{i,j=1,\ldots n} c_{ij} x_i x_j$.

Si $\varepsilon_k \approx 0 \ \forall k$, entonces $\tilde{y}_p(k+1) \approx y(k+1)$. Sin embargo, si existe un n tal que ε_n es lejano a cero, entonces $\tilde{y}_p(k+1)$ no es una aproximación adecuada de $y(k+1)$, $\forall k \geq n$. Además, a medida que k se aleja de n, $\tilde{y}_p(k+1)$ se aleja cada vez más de $y(k+1)$, ya que los errores cometidos en instantes anteriores se amplifican en potencias crecientes de magnitud 2^k.

En la discusión presentada anteriormente se observa que el motivo principal por el que el modelo en paralelo podría proporcionar una representación no aceptable del proceso dinámico es que el conjunto de patrones de entrada, para dicho modelo, difiere del conjunto de patrones con el que se obtienen los parámetros $W_{\tilde{F}}$.

Para poder garantizar un buen funcionamiento del modelo en paralelo (ecuación 3.16) sería necesario calcular sus pesos o parámetros, no para minimizar el error de identificación dado por la ecuación 3.17, sino para minimizar la función error descrita

por las aproximaciones $\tilde{y}_p(k+1)$, $k = 0,\ldots,N-1$, también llamada en este contexto *error de predicción*:

$$E_m^P = \frac{1}{2N} \cdot \sum_{k=0}^{N-1} (y(k+1) - \tilde{y}_p(k+1))^2 \tag{3.23}$$

siendo $\tilde{y}_p(k+1)$ la respuesta del modelo de identificación en paralelo dado por la ecuación 3.16.

El conjunto de pesos $W_{\tilde{F}}$ ciertamente minimiza el error de identificación E_m^I, puesto que la estimación de dichos pesos se lleva a cabo siguiendo la dirección negativa del gradiente de dicha función. Sin embargo, esto no garantiza que dicho conjunto de pesos sea también un mínimo para la superficie definida por E_m^P, superficie generalmente diferente a la definida por E_m^I.

Para n_y=0 y n_u=0, la ecuación 3.22 permite obtener una relación entre ambas superficies:

$$2N \cdot E_m^P = \sum_{k=0}^{N-1} (y(k+1) - \tilde{y}(k+1))^2 + \sum_{k=1}^{N-1} \left(\delta\left[\varepsilon_k + \delta(\varepsilon_{k-1} + \delta(\varepsilon_{k-2} + \ldots + \delta(\varepsilon_1)\ldots)) \right] \right)^2 -$$

$$-2 \sum_{k=1}^{N-1} (y(k+1) - \tilde{y}(k+1)) \cdot \delta\left[\varepsilon_k + \delta(\varepsilon_{k-1} + \delta(\varepsilon_{k-2} + \ldots + \delta(\varepsilon_1)\ldots)) \right]$$

Por tanto y teniendo en cuenta las ecuaciones 3.17 y 3.20, se obtiene que:

$$E_m^P = E_m^I + \frac{1}{2N} \cdot \sum_{k=1}^{N-1} \left(\delta\left[\varepsilon_k + \delta(\varepsilon_{k-1} + \delta(\varepsilon_{k-2} + \ldots + \delta(\varepsilon_1)\ldots)) \right] \right)^2 -$$

$$-\frac{1}{N} \cdot \sum_{k=1}^{N-1} \varepsilon_{k+1} \cdot \left(\delta\left[\varepsilon_k + \delta(\varepsilon_{k-1} + \delta(\varepsilon_{k-2} + \ldots + \delta(\varepsilon_1)\ldots)) \right] \right)^2$$

Se observa que los errores ε_k, $k = 1,\ldots,N$, y la capacidad de la red para responder a patrones perturbados y que no han sido utilizados durante el entrenamiento (capacidad evaluada por $\delta(\cdot)$), son los que determinan cuánto difiere una superficie de otra.

3.2.2 Modelo NARMA neuronal propuesto

Como se ha concluido en el apartado anterior, el modelo de identificación en paralelo dado por la ecuación 3.16 no garantiza una aproximación adecuada del proceso real, debido precisamente al conjunto de pesos $W_{\tilde{F}}$ utilizado para construir el modelo.

Si el objetivo es construir un simulador del proceso de la forma dada por la ecuación 3.16 y que proporcione una predicción adecuada de la dinámica del proceso real sin su presencia, el conjunto de parámetros que determinan el modelo deberán calcularse con esta finalidad.

La solución propuesta en este trabajo consiste en utilizar la red parcialmente recurrente descrita en el apartado 3.1.2 para aproximar el funcional F que determina el modelo NARMA. Considerando el vector $I(k) = (u(k-d),\ldots,u(k-d-n_u))$ como entrada a la red y n_y+1 neuronas de contexto, dicha arquitectura permite obtener un modelo en paralelo que puede escribirse como:

$$\tilde{y}_r(k+1) = \tilde{F}_r(I(k),R(k),W_{\tilde{F}_r}) = \tilde{F}_r(u(k-d),\ldots,u(k-d-n_u),R_1(k),\ldots,R_{n_y+1}(k),W_{\tilde{F}_r}) \qquad (3.24)$$

siendo $\tilde{y}_r(k+1)$ la respuesta de la red parcialmente recurrente para el patrón de entrada $I(k)$, $R_i(k) = \tilde{y}_r(k+1-i)$ para $i = 1,\ldots,n_y+1$ y $W_{\tilde{F}_r}$ el conjunto de pesos de dicha red.

Al utilizar esta arquitectura de red recurrente para aproximar el funcional F, los parámetros del modelo de identificación en paralelo se adaptan para que dicho modelo aprenda a actuar sin la presencia del proceso dinámico. Debido a la propia arquitectura de la red, la función error utilizada para llevar a cabo su aprendizaje, es decir, para calcular el conjunto de pesos $W_{\tilde{F}_r}$, coincide exactamente con el error de predicción definido por la ecuación 3.23 (en este caso, la salida del modelo en paralelo en 3.23, $\tilde{y}_p(k+1)$, es la salida de la red parcialmente recurrente, denotada como $\tilde{y}_r(k+1)$). De este modo, cuando se realiza el aprendizaje de la red, el conjunto de parámetros $W_{\tilde{F}_r}$ se determinan con el propósito de simular la dinámica del proceso. Por tanto, el modelo en paralelo dado por la ecuación 3.24 podrá proporcionar una mejor representación del proceso dinámico que la obtenida con el modelo en paralelo dado por la ecuación 3.16, para el cual los parámetros no eran determinados con dicho propósito.

En el apartado anterior, se ha mencionado que podrían darse situaciones en las que el modelo en paralelo dado por la ecuación 3.16 proporcionara una representación aceptable del proceso dinámico (caso 1º). En las simulaciones realizadas en este trabajo se ha observado que, incluso en estos casos, es aconsejable utilizar el modelo en paralelo dado por la ecuación 3.24. La razón principal es que el número de ciclos de aprendizaje requerido por este último modelo para obtener una representación adecuada del proceso cuando se simula su dinámica, es considerablemente inferior a si el funcional F se aproxima utilizando un perceptron multicapa. Esto podría ser debido a que, al utilizar un perceptron multicapa, los parámetros del modelo se adaptan siguiendo la dirección negativa del gradiente del error de identificación (ecuación 3.17), error que, como se veía en el apartado anterior, difiere del error de predicción que se pretende minimizar (ecuación 3.23). Sin embargo, cuando se utiliza la red parcialmente recurrente, los parámetros del modelo se determinan en base al gradiente del error de predicción, lo cual facilita la localización de un mínimo para dicho error.

Fase de identificación del modelo propuesto

La fase de identificación del modelo o, lo que es lo mismo, el aprendizaje de la red, se realiza utilizando el algoritmo descrito en el apartado 3.1.3, y obteniendo previamente, siempre que se considere necesario, una inicialización de los pesos de la red parcialmente recurrente, como se indicaba también en dicho apartado, y con el propósito de acelerar la convergencia. Durante el aprendizaje los patrones de entrada a la red se presentan de forma ordenada siguiendo la evolución del tiempo discreto k.

En el contexto de estimación de los parámetros de un modelo, es necesario distinguir entre las técnicas de identificación off-line y las técnicas on-line o en tiempo real, como se mencionaba en el apartado 1.2.3. Si el conjunto de datos experimentales del que se dispone para estimar los parámetros del modelo es representativo de la dinámica del proceso en su totalidad, dichos parámetros podrán identificarse utilizando un método off-line, es decir previamente a la utilización del modelo. El modelo construido con dichos parámetros representará al proceso dinámico en cualquier situación. Sin embargo, no siempre será posible obtener tal conjunto de datos experimentales; de hecho, es habitual disponer de observaciones que representan la dinámica del proceso para ciertas condiciones iniciales. En estos casos, el modelo construido con los parámetros obtenidos al realizar una estimación off-line, no

aproximará de forma adecuada al proceso en regiones que no han sido representadas. Como consecuencia, será necesario utilizar técnicas on-line que identifiquen los parámetros del modelo mientras que dicho modelo esté actuando, en cuyo caso los datos experimentales proceden de la evolución directa del proceso real.

En este trabajo, y con el propósito de control, se utilizan ambas técnicas de identificación. La diferencia entre ellas radica únicamente en la función error utilizada para estimar los parámetros o pesos de la red.

Mediante las técnicas de identificación off-line los pesos de la red se adaptan para minimizar el error global sobre todo el conjunto de datos experimentales (ecuación 3.23). Debido a que la estimación de los parámetros se lleva a cabo previamente a la utilización del modelo, dichas técnicas permiten la realización de tantos ciclos de aprendizaje como para conseguir que el error de predicción global sea menor que un cierto valor establecido a priori.

Durante la identificación on-line, sin embargo, no es posible utilizar una medida de error global, ya que el modelo actúa en paralelo con el proceso real. En este caso, los pesos de la red se adaptan en cada instante de tiempo k+1 utilizando la siguiente función error:

$$e_m^P(k+1) = \frac{1}{2} \cdot \sum_{i=0}^{T} (y(k+1-i) - \tilde{y}_r(k+1-i))^2 \qquad (3.25)$$

siendo T la longitud de un cierto intervalo discreto de tiempo, [k+1-T, k+1], elegido a priori. Para T=0, la adaptación de los parámetros del modelo se realiza en base al error instantáneo $\frac{1}{2} \cdot (y(k+1) - \tilde{y}_r(k+1))^2$.

Cuando se lleva a cabo una identificación on-line de los parámetros del modelo (figura 3.3), en cada periodo de muestreo (periodo de tiempo transcurrido hasta que se recibe una nueva medida) se realiza el entrenamiento del modelo utilizando como patrones para la red los pares {I(k-T),y(k-T+1)},..., {I(k),y(k+1)}, por lo que el valor T elegido dependerá del tiempo de muestreo.

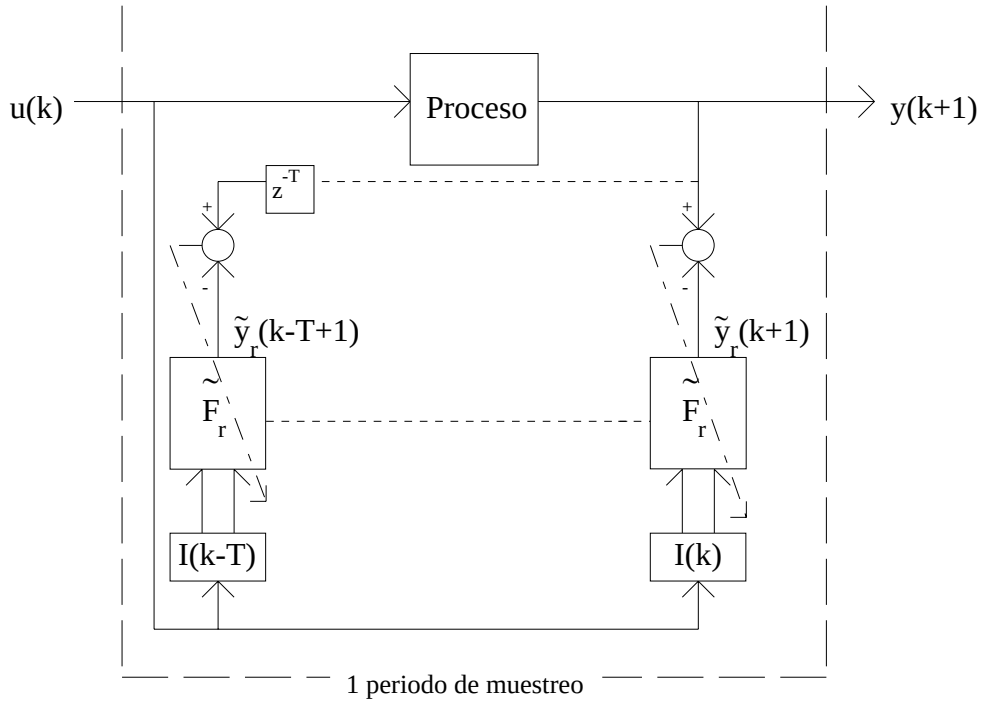


Figura 3.3. Identificación on-line de los parámetros del modelo.

Una cuestión importante cuando se lleva a cabo una identificación on-line de los parámetros del modelo es el tiempo de aprendizaje, es decir, el número de periodos de muestreo que tienen que transcurrir para que la red proporcione una buena respuesta. Nótese que, en este caso, cada periodo de muestreo corresponde con T ciclos de aprendizaje. Una buena aproximación, pero demasiado tarde, puede provocar resultados no satisfactorios, principalmente cuando el control del proceso se basa en dicha aproximación. Interesa, por tanto, conseguir que el modelo aprenda en el menor número posible de periodos de muestreo.

Como ya se ha comentado, partir de una inicialización adecuada para los pesos de la red acelera su aprendizaje. Sin embargo, en el contexto de identificación en tiempo real puede ocurrir que dicha inicialización no acelere lo suficiente el aprendizaje de la red como para que produzca una respuesta adecuada en cada instante de tiempo k+1.

Cuando el periodo de muestreo de las variables medibles del proceso es más grande que el tiempo que requiere la red de neuronas para calcular su salida y adaptar sus parámetros, es posible presentar varias veces a la red los patrones {I(k-T),y(k-T+1)},..., {I(k),y(k+1)}, tantas como permita el tiempo del que se dispone hasta recibir la nueva medida del proceso. De este modo, se puede favorecer al aprendizaje de la red,

reduciéndose el error dado por la ecuación 3.25 y consiguiéndose predicciones más precisas de la salida del proceso dinámico.

3.3 Controladores no lineales utilizando redes de neuronas

En esta sección se desarrollan dos esquemas diferentes de control no lineal utilizando las redes de neuronas artificiales: control inverso no lineal y control predictivo no lineal.

Como ya se ha mencionado en el apartado 1.2.6, diferentes autores han desarrollado y estudiado estos esquemas de control. Sin embargo, ambos sistemas siguen presentando en la actualidad una serie de inconvenientes y limitaciones, sobre todo en lo referente a aplicaciones en tiempo real, como se veía en la sección 2.2.

A continuación se presentan las soluciones propuestas para resolver dichas limitaciones, de forma que se pueda obtener un control eficiente del proceso dinámico real cuando se aplican las estrategias de control inverso y de control predictivo.

3.3.1 Sistema de control inverso propuesto

Como se decía en el apartado 1.2.6, las estrategias de control inverso consisten en determinar la dinámica inversa del proceso. Las redes de neuronas artificiales pueden utilizarse para aproximar dicha relación no lineal, permitiendo así la obtención de leyes de control inverso para procesos dinámicos no lineales.

No obstante, y según se veía en la sección 2.2, los sistemas de control inverso utilizando redes de neuronas, tanto en su versión global (aprendizaje generalizado) como en su versión local (aprendizaje especializado) poseen una serie de inconvenientes. El esquema que resulta de aplicar el aprendizaje generalizado es un sistema de control off-line, por lo que requiere de una gran cantidad de datos experimentales, obtenidos a priori, para poder garantizar el éxito del controlador. El esquema de control inverso con aprendizaje especializado no requiere de dicho conjunto, ya que se trata de un sistema de control on-line y los datos para el aprendizaje proceden de la evolución directa del proceso dinámico. Sin embargo, este último

requiere de una inicialización adecuada de los pesos del controlador para poder garantizar el éxito de aplicaciones en tiempo real.

En este trabajo se desarrolla un esquema de control inverso en el que una red de neuronas es la encargada de aproximar la inversa del proceso y, por tanto, de calcular la acción de control que hay que aplicar al proceso en cada instante de tiempo para que la salida de dicho proceso tienda asintóticamente al objetivo de control o salida deseada. El entrenamiento del controlador neuronal consiste en utilizar el aprendizaje generalizado y especializado.

En primer lugar, se lleva a cabo un aprendizaje generalizado del controlador con los ejemplos disponibles; dicho aprendizaje no se realiza con el objetivo de construir el controlador definitivo del proceso, sino con vistas a obtener una inicialización de los pesos de la red de neuronas que actúa como controlador del proceso real. Los pesos obtenidos al realizar este aprendizaje no tienen por qué generar una ley de control eficiente, todo dependerá de los datos con los que se lleve a cabo el aprendizaje. Sin embargo, esta inicialización ayuda a evitar situaciones de “no control” o inestabilidad cuando se realice el aprendizaje especializado, ya que dichos pesos poseen información sobre la relación inversa del proceso.

Inicializando, entonces, el controlador con dichos pesos se procede a realizar el aprendizaje especializado, con la finalidad de que la red de neuronas aprenda a conseguir el objetivo de control deseado en cada momento. Con el objetivo de obtener un control eficiente del proceso real, el aprendizaje especializado del controlador neuronal se realiza, en primer lugar, utilizando un modelo del proceso y, posteriormente, con el proceso real. Esta última fase permite que errores en el control provocados por la utilización de un modelo no exacto, se corrijan en tiempo real.

Antes de pasar al desarrollo detallado del método de aprendizaje, existe la necesidad de determinar la estructura del controlador, es decir la expresión de la ley de control inverso no lineal. En el contexto de redes de neuronas, esto es equivalente a determinar la arquitectura de la red.

Estructura del controlador

Cuando se afronta el problema de aproximar la relación entrada-salida de un proceso dinámico con redes de neuronas, la determinación de las variables de entrada a la red es crucial y tiene una repercusión importante en la calidad del resultado final. Lo

mismo sucede cuando se trata de aproximar la dinámica inversa; deben especificarse todas las variables que influyan en el cálculo de la señal de control para producir la salida deseada en el proceso.

Se considera un proceso dinámico discreto no lineal con una única variable de entrada y salida, representado por el siguiente modelo NARMA:

$$y(k+1) = F(y(k),\ldots,y(k-n_y),u(k-d),\ldots,u(k-d-n_u)) \tag{3.26}$$

Se supone que existe un funcional G tal que la entrada manipulada se puede expresar en función de la salida actual del proceso y de las secuencias de entrada y salida de la siguiente forma:

$$u(k-d) = G(y(k+1),y(k),\ldots,y(k-n_y),u(k-d-1),\ldots,u(k-d-n_u)) \tag{3.27}$$

Actualizando esta ecuación al tiempo k, se obtiene:

$$u(k) = G(y(k+1+d),y(k+d),\ldots,y(k+d-n_y),u(k-1),\ldots,u(k-n_u)) \tag{3.28}$$

Asumiendo que la función G es conocida, la expresión dada en la ecuación 3.28 permite calcular la acción de control que hay que aplicar al proceso en cada instante de tiempo k, u(k), para que dicho proceso alcance el valor y(k+d+1) al instante de tiempo k+d+1, ya que G define la "relación inversa" del modelo NARMA dado en la ecuación 3.26 con respecto a la variable u(k-d). Nótese que, debido al retraso del proceso, la acción de control al tiempo k tiene una influencia en la salida del proceso en el instante k+d+1.

Si el propósito es que el proceso responda con un valor deseado constante, y^{des}, la acción de control que hay que aplicar al proceso en cada instante de tiempo sería:

$$u(k) = G(y^{des},y(k+d),\ldots,y(k+d-n_y),u(k-1),\ldots,u(k-n_u)) \tag{3.29}$$

En la ecuación 3.29 se observa que, para poder aplicar esta ley de control, en cada instante de tiempo k es necesario conocer las medidas de la salida del proceso en el futuro $y(k+d),\ldots,y(k+1)$, lo cual es imposible cuando se realiza el control en tiempo real. Por tanto, es necesario transformar la ecuación 3.29 en una expresión que permita

calcular la acción de control en función de la información disponible en el instante de tiempo k.

Utilizando la ecuación 3.26, los valores $y(k+d), y(k+d-1), \ldots, y(k+1)$ se pueden expresar de la forma:

$$y(k+d) = F(y(k+d-1), \ldots, y(k-(n_y+1-d)), u(k-1), \ldots, u(k-(n_u+1))) \quad (3.30)$$

.
.
.

$$y(k+3) = F(y(k+2), \ldots, y(k-(n_y-2)), u(k-(d-2)), \ldots, u(k-(d+n_u-2))) \quad (3.31)$$

$$y(k+2) = F(y(k+1), \ldots, y(k-(n_y-1)), u(k-(d-1)), \ldots, u(k-(d+n_u-1))) \quad (3.32)$$

$$y(k+1) = F(y(k), \ldots, y(k-n_y), u(k-d), \ldots, u(k-(d+n_u))) \quad (3.33)$$

Sustituyendo el término $y(k+1)$ que aparece en el segundo miembro de 3.32 por la expresión dada en la ecuación 3.33, se obtiene que $y(k+2)$ puede expresarse en función de los términos $y(k), \ldots, y(k-(n_y-1)), y(k-n_y), u(k-(d-1)),\ u(k-d), \ldots, u(k-(d+n_u))$.

Sustituyendo el término $y(k+2)$ que aparece en el segundo miembro de 3.31 por la expresión obtenida anteriormente y utilizando la ecuación 3.33 para expresar $y(k+1)$, se obtiene que $y(k+3)$ se puede expresar en función de la secuencia discreta $y(k), \ldots, y(k-(n_y-2)), y(k-(n_y-1)), y(k-n_y), u(k-(d-2)), \ldots, u(k-(d+n_u))$.

Realizando este mismo razonamiento hasta llegar al término $y(k+d)$, se obtiene que dicho valor es una función de $y(k), \ldots, y(k-n_y), u(k-1), \ldots, u(k-(d+n_u))$.

Reemplazando entonces los valores $y(k+d), y(k+d-1), \ldots, y(k+1)$ en la ecuación 3.29 por las nuevas expresiones y manteniendo la notación G, es posible escribir dicha ley de control de la siguiente forma:

$$u(k) = G(y^{des}, y(k), \ldots, y(k-n_y), u(k-1), \ldots, u(k-n_u), \ldots, u(k-d-n_u)),\ k \in N_0 \quad (3.34)$$

siendo $y(-n_y) = \ldots = y(0),\ u(-d-n_u) = \ldots = u(-1) = u_{-1} \in \Re$.

En las ecuaciones 3.29 y 3.34 la longitud de la secuencia discreta para la variable de entrada u es diferente. Para la ley de control original (ecuación 3.29) dicha secuencia

es de longitud n_u, mientras que en la ley transformada (ecuación 3.34) la longitud de la secuencia ha aumentado en d términos. Como se puede observar en el desarrollo anterior, este aumento viene provocado por la presencia de un retraso natural en el proceso. Para procesos dinámicos sin retraso sería suficiente considerar una secuencia de longitud n_u para determinar la relación inversa.

Por tanto, partiendo de un modelo del proceso, es posible establecer las variables influyentes (número de valores anteriores de las variables de entrada y salida) en la dinámica inversa. Una vez determinadas dichas variables, el funcional G en 3.34 puede aproximarse utilizando una red de neuronas.

Sería posible pensar, al igual que cuando se trataba la modelización de procesos dinámicos, que, considerando como patrón de entrada a la red el vector $I(k) = (y^{des}, y(k),\ldots, y(k-n_y), u(k-1),.., u(k-d-n_u))$, el funcional G podría aproximarse con una red estática. Sin embargo, en el contexto de control de procesos dinámicos en tiempo real esto no es válido, ya que, cuando se afronta el problema del control, la variable de entrada al proceso es la que se pretende calcular y, por tanto, la secuencia de valores reales $u(k-1),.., u(k-d-n_u)$ es desconocida; los únicos valores disponibles de u son los calculados por la red en instantes anteriores de tiempo.

Considerando, entonces, como patrón de entrada el vector $I(k) = (y^{des}, y(k), \ldots, y(k-n_y))$ y $d+n_u$ neuronas de contexto, la red parcialmente recurrente descrita en el apartado 3.1.2 puede utilizarse para aproximar el funcional G en 3.34, obteniendo una ley de control inverso no lineal de la forma:

$$\tilde{u}_r(k) = \tilde{G}_r(I(k), R(k), W_{\tilde{G}_r}) = \tilde{G}_r(y^{des}, y(k),\ldots, y(k-n_y), R_1(k),\ldots, R_{d+n_u}(k), W_{\tilde{G}_r}) \qquad (3.35)$$

donde $\tilde{u}_r(k)$ es la salida de la red de neuronas parcialmente recurrente, $R_i(k) = \tilde{u}_r(k-i)$ para $i = 1,\ldots, d+n_u$ y $W_{\tilde{G}_r}$ el conjunto de parámetros ajustables de la red.

Nótese que en este caso, y a diferencia de cuando se trata la modelización de procesos dinámicos, el vector de entrada a la red parcialmente recurrente está formado por la respuesta deseada para el proceso y por la historia de la variable de salida del

proceso, mientras que las neuronas de contexto memorizan la historia de la variable de entrada al proceso.

Algoritmo de aprendizaje

El aprendizaje del controlador consiste en determinar el conjunto de parámetros $W_{\tilde{G}_r}$ de la ley dada en la ecuación 3.35. Como ya se ha mencionado al comienzo de este apartado, dicho aprendizaje se divide en tres fases diferentes:

1ª Aprendizaje generalizado con los datos disponibles.
2ª Aprendizaje especializado utilizando un modelo del proceso.
3ª Aprendizaje especializado utilizando el proceso real.

Las dos primeras se llevan a cabo con la finalidad de obtener una inicialización de los pesos o parámetros del controlador neuronal por lo que se consideran fases de aprendizaje off-line, mientras que el último paso se realiza en tiempo real y con la finalidad de obtener el controlador definitivo del proceso dinámico. Como se verá a continuación la diferencia entre ellas radica principalmente en la función error utilizada para adaptar los pesos de la red parcialmente recurrente.

1ª fase: Aprendizaje generalizado con los datos disponibles

Con esta fase se pretende obtener un conjunto de parámetros $W_{\tilde{G}_r}$ que posea información sobre el comportamiento inverso del proceso dinámico, de modo que dicha información pueda utilizarse posteriormente para controlar el proceso.

Si se conociera la dinámica inversa del proceso en su totalidad, dicha relación podría proporcionar un control adecuado del proceso. El problema es, como ya se ha comentado, que la relación inversa es raramente conocida y no siempre se disponen de datos representativos, de modo que pueda obtenerse una aproximación más o menos exacta de dicha inversa global.

El objetivo en esta tesis es llevar a cabo un aprendizaje generalizado con los datos disponibles de forma que, aunque el controlador neuronal no aproxime la inversa en su totalidad, sí conozca las características más generales de dicho comportamiento inverso.

Los patrones para el entrenamiento en esta fase de aprendizaje no son mas que pares entrada-salida del proceso dinámico {u(k),y(k+1)}, k=0,...,N-1, obtenidos de

forma experimental. En este caso, y a diferencia de cuando se afrontaba la construcción de modelos NARMA, la entrada al proceso es la que actúa como salida deseada para la red (figura 3.4). El conjunto de pesos $W_{\tilde{G}_r}$ se determina, entonces, para minimizar la siguiente función error:

$$E_c^g = \frac{1}{2N} \cdot \sum_{k=0}^{N-d-1} (u(k) - \tilde{u}_r(k))^2 \qquad (3.36)$$

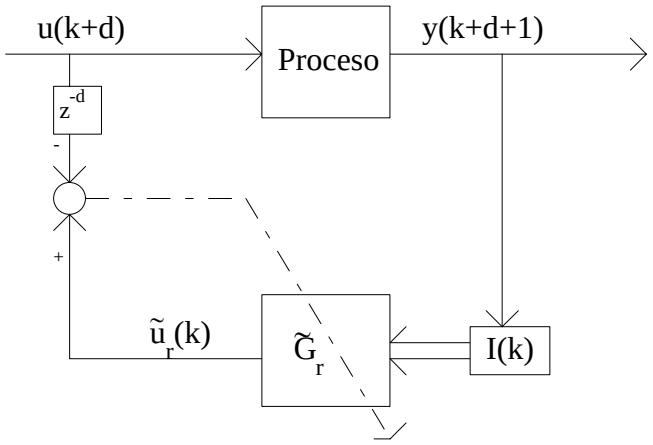


Figura 3.4. Esquema del aprendizaje generalizado.

Según la ecuación 3.35 el vector de entrada a la red parcialmente recurrente debería ser $I(k) = (y^{des}, y(k), \ldots, y(k - n_y))$. Sin embargo, cuando el controlador es sometido a un aprendizaje generalizado ningún objetivo de control ha sido todavía definido, pues la red está aprendiendo la dinámica inversa y no está aún controlando el proceso. Por tanto, el valor y^{des} tiene que ser sustituido por la respuesta real del proceso y(k+d+1), k=0,...,N-d-1, resultando como patrón de entrada el vector $I(k) = (y(k + d + 1), y(k), \ldots, y(k - n_y))$.

Los pesos o parámetros del controlador neuronal se adaptan utilizando el algoritmo descrito en el apartado 3.1.3 para minimizar la función error definida en la ecuación 3.36. Puesto que en este contexto la acción de control u(k) está disponible, es posible acelerar el aprendizaje de la red partiendo de una inicialización de sus pesos, la cual se obtiene entrenando el perceptron multicapa equivalente a la red parcialmente recurrente,

con patrones de entrada $I(k) = (y(k+d+1), y(k), \ldots, y(k-n_y), u(k-1), \ldots u(k-d-n_u))$, como se indicaba en el apartado 3.1.3.

2ª fase: Aprendizaje especializado utilizando un modelo del proceso

Dado un modelo del proceso y un objetivo de control y^{des}, la finalidad de esta fase de aprendizaje es que el conjunto de pesos $W_{\tilde{G}_r}$ obtenidos en la fase anterior se especialice para que la ley de control dada por la ecuación 3.35 haga que el modelo elegido alcance el objetivo de control definido previamente.

En este caso, los datos para el aprendizaje proceden de la evolución directa del modelo. La función error para adaptar los pesos del controlador neuronal se obtiene propagando a través del modelo la acción de control calculada por la red y evaluando la diferencia entre la respuesta del modelo y el objetivo de control (figura 3.5). Definiendo un intervalo discreto de tiempo [0,n] (siendo n un número natural mayor o igual al tiempo necesario para que el proceso dinámico alcance el objetivo de control y^{des}, el cual se determina de forma experimental), los pesos del controlador se adaptan para minimizar la función error definida como:

$$E_c^s = \frac{1}{n} \cdot \sum_{k=0}^{n-1} e_c^s(k+1) \tag{3.37}$$

$$\text{siendo } e_c^s(k+1) = \frac{1}{2} \cdot (y^{des} - \tilde{y}(k+1)^2 \tag{3.38}$$

con $\tilde{y}(k+1)$ la respuesta del modelo.

Dado $w_{\tilde{G}_r}$ un peso o parámetro del controlador neuronal, la variación de la función error definida en la ecuación 3.38 con respecto a dicho parámetro es:

$$\frac{\partial e_c^s(k+1)}{\partial w_{\tilde{G}_r}} = -(y^{des} - \tilde{y}(k+1)) \cdot \frac{\partial \tilde{y}(k+1)}{\partial \tilde{u}_r(k-d)} \cdot \frac{\partial \tilde{u}_r(k-d)}{\partial w_{\tilde{G}_r}}, \; k = 0, \ldots, n-1 \tag{3.39}$$

con $\tilde{u}_r(-d) = \ldots = \tilde{u}_r(-1) = \tilde{u}_r(0)$.

Nótese que, debido a que se tratan procesos dinámicos y modelos para los que la salida presenta un retraso natural d con respecto a la entrada, la acción de control que provoca un cambio en la respuesta $\tilde{y}(k+1)$ no es $\tilde{u}_r(k)$, sino $\tilde{u}_r(k-d)$.

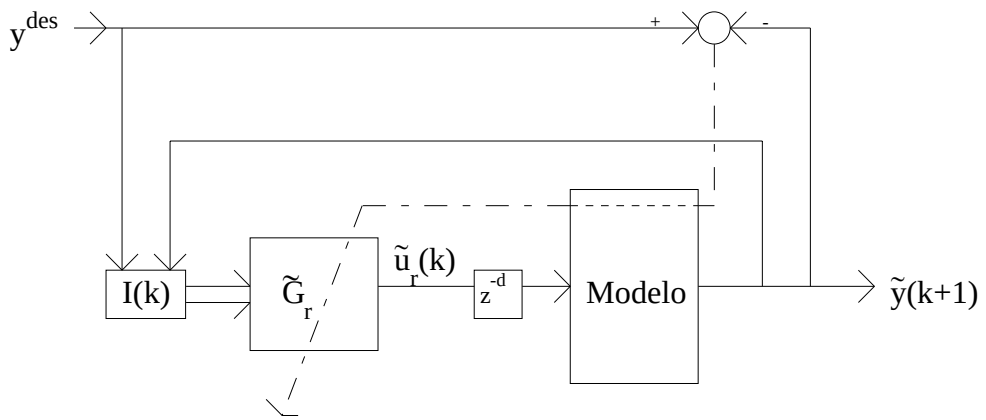


Figura 3.5. Esquema del aprendizaje especializado utilizando un modelo del proceso.

Los términos $\frac{\partial \tilde{y}(k+1)}{\partial \tilde{u}_r(k-d)}$, $k=0,\ldots,n-1$, representan la variación de la salida del modelo respecto a su entrada, es decir, el Jacobiano del modelo. Dichos términos aparecen en el cálculo del gradiente de la función error debido a que el modelo se coloca entre la red neuronal que actúa como controlador y dicha función (figura 3.5). Estos términos permiten conocer la relación entre el error medido en la salida del modelo y el error en la neurona de salida del controlador neuronal, la cual viene dada por $(y^{des}-\tilde{y}(k+1))\cdot\frac{\partial \tilde{y}(k+1)}{\partial \tilde{u}_r(k-d)}$. En esta fase de aprendizaje el modelo no sólo se encarga de proporcionar las predicciones $\tilde{y}(k+1)$, $k=0,\ldots,n-1$, sino también de propagar hacia la salida del controlador el error medido en la salida del modelo, como se indica en la figura 3.5, y poder así adaptar los pesos del controlador.

Cada derivada $\frac{\partial \tilde{u}_r(k-d)}{\partial w_{\tilde{G}_r}}$, $k=0,\ldots,n-1$, en la ecuación 3.39 representa la variación total de la salida de la red parcialmente recurrente con respecto al parámetro $w_{\tilde{G}_r}$ y se calcula utilizando retropropagación dinámica (ecuación 3.11).

Como consecuencia, en esta fase de aprendizaje los pesos del controlador neuronal se adaptan en cada periodo de muestreo de acuerdo con la siguiente ley:

$$w_{\tilde{G}_r}^{(k+1)} = w_{\tilde{G}_r}^{(k)} + \alpha \cdot (y^{des} - \tilde{y}(k+1)) \cdot \frac{\partial \tilde{y}(k+1)}{\partial \tilde{u}_r(k-d)} \cdot \frac{\partial \tilde{u}_r(k-d)}{\partial w_{\tilde{G}_r}},\ k = 0,\ldots,n-1 \qquad (3.40)$$

siendo $\tilde{u}_r(-d) = \ldots = \tilde{u}_r(-1) = \tilde{u}_r(0)$.

La consideración de un modelo del proceso para llevar a cabo el aprendizaje especializado del controlador en lugar del propio proceso, permite que el conjunto de patrones entrada-salida, $\left\{\left(I(k), y^{des}\right): I(k) = (y^{des}, \tilde{y}(k), \ldots, \tilde{y}(k-n_y)), k = 0,\ldots,n-1\right\}$, pueda presentarse varias veces a la red neuronal y, por tanto, que ésta pueda aprender a calcular las acciones de control $u(0),\ldots,u(n-1-d)$ que provocan en el modelo el comportamiento deseado.

El modelo elegido para llevar a cabo esta fase de aprendizaje tiene que poseer la propiedad de saber actuar como simulador del proceso. Por otra parte, es conveniente utilizar modelos cuya estructura facilite el cálculo del Jacobiano.

Los simuladores NARMA desarrollados en el apartado 3.2.2 son una alternativa atractiva, no sólo por el hecho de que pueden actuar como simulador del proceso proporcionado una representación adecuada, sino porque, debido a su estructura neuronal, el Jacobiano puede calcularse aplicando retropropagación estática.

3ª fase: Aprendizaje especializado utilizando el proceso real

Esta última fase de aprendizaje se lleva a cabo con la finalidad de que la ley de control dada por la ecuación 3.35 haga que el proceso dinámico real alcance el objetivo de control deseado y^{des}.

Al inicializar esta ley de control con el conjunto de pesos $W_{\tilde{G}_r}$ obtenidos en la fase anterior, no suelen producirse situaciones de no control del proceso o inestabilidad, puesto que dichos parámetros ya poseen información suficiente como para evitarlas. No obstante, el grado de aceptabilidad de esta ley para controlar el proceso real depende, obviamente, de la capacidad del modelo elegido para representar y simular dicho

proceso. Obsérvese que el controlador neuronal ha aprendido a controlar un modelo del proceso y no el proceso real.

Si el modelo utilizado en la fase anterior fuese un modelo exacto, la ley de control construida proporcionaría un control adecuado del proceso real y sus parámetros podrían permanecer fijos, sin necesidad de someterlos a ninguna otra adaptación.

Sin embargo, en la mayor parte de los casos los modelos no son exactos, sino que proporcionan aproximaciones de la dinámica del proceso. Este hecho, y dependiendo de la validez de dichas aproximaciones, puede conducir a que la ley de control inverso con los parámetros obtenidos en la fase anterior no provoque los mismos resultados de control en el proceso real que en el modelo. Será necesario, por tanto, modificar los parámetros que determinan dicha ley y poder así controlar de forma eficiente el proceso dinámico real.

Esta modificación consiste en llevar a cabo un aprendizaje especializado del controlador, pero utilizando el proceso real y no un modelo del proceso, por lo que en este caso la adaptación de los pesos del controlador se realiza en tiempo real.

En cada instante de tiempo k y utilizando $I(k) = (y^{des}, y(k), \ldots, y(k-n_y))$ como vector de entrada a la red parcialmente recurrente que actúa como controlador, la salida calculada por dicha red, $\tilde{u}_r(k)$, se presenta al proceso y sus pesos se adaptan siguiendo la dirección negativa del gradiente del error instantáneo medido en la salida del proceso, como se indica en la figura 3.6:

$$e_c^s(k+1) = \frac{1}{2} \cdot (y^{des} - y(k+1))^2 \tag{3.41}$$

siendo y(k+1) la respuesta del proceso dinámico para la acción de control $\tilde{u}_r(k)$.

Realizando el mismo razonamiento que en la fase anterior, la correspondencia entre el error medido en la salida del proceso y el error en la salida del controlador viene dada, en este caso, por el Jacobiano del proceso $\frac{\partial y(k+1)}{\partial \tilde{u}_r(k-d)}$. Los pesos del controlador se adaptan, entonces, en cada instante de tiempo k+1 de acuerdo con la siguiente ley:

$$w_{\tilde{G}_r}^{(k+1)} = w_{\tilde{G}_r}^{(k)} + \alpha \cdot (y^{des} - y(k+1)) \cdot \frac{\partial y(k+1)}{\partial \tilde{u}_r(k-d)} \cdot \frac{\partial \tilde{u}_r(k-d)}{\partial w_{\tilde{G}_r}}, \; k \in N_0 \tag{3.42}$$

con $\tilde{u}_r(-d) = \ldots = \tilde{u}_r(-1) = \tilde{u}_r(0)$.

Sin embargo, para la mayor parte de los procesos reales su Jacobiano es desconocido y difícil de calcular a partir de los datos experimentales. Una posible solución a este problema es sustituir el Jacobiano del proceso por una aproximación.

Esta aproximación puede obtenerse utilizando ecuaciones en diferencias para aproximar derivadas parciales, como por ejemplo: $\frac{\partial y}{\partial u} \approx \frac{y(u+\delta u) - y(u)}{\delta u}$, siendo δu suficientemente pequeño. Dicha aproximación se determina realizando cambios pequeños en la entrada del proceso y midiendo el cambio producido en la salida, o bien, comparando la variación de la salida del proceso con respecto a iteraciones anteriores. No obstante, es necesario prestar mucha atención cuando el Jacobiano del proceso en la ecuación 3.42 se sustituye por una aproximación de este estilo. Diversos factores, como el ruido en la medida de la salida del sistema, pueden hacer que la aproximación obtenida con dicha expresión tenga un error considerable, el cual puede provocar cambios inadecuados en los pesos y, por tanto, resultados no satisfactorios durante el aprendizaje. Esto se traduce, generalmente, en problemas de inestabilidad en el controlador debido a oscilaciones en la señal de control calculada por la red.

Una solución más fiable y conveniente es aproximar el Jacobiano del proceso utilizando un modelo de dicho proceso, en cuyo caso la ley de adaptación adopta la siguiente forma:

$$w_{\tilde{G}_r}^{(k+1)} = w_{\tilde{G}_r}^{(k)} + \alpha \cdot (y^{des} - y(k+1)) \cdot \frac{\partial \tilde{y}(k+1)}{\partial \tilde{u}_r(k-d)} \cdot \frac{\partial \tilde{u}_r(k-d)}{\partial w_{\tilde{G}_r}}, \; k \in N_0 \qquad (3.43)$$

siendo $\frac{\partial \tilde{y}(k+1)}{\partial \tilde{u}_r(k-d)}$ el Jacobiano del modelo elegido.

Al igual que en la fase anterior, este esquema de aprendizaje requiere también de un modelo del proceso, aunque en este caso, dicho modelo se utiliza únicamente para aproximar el Jacobiano, es decir para propagar el error medido en la salida del proceso hacia la salida del controlador como se muestra en la figura 3.6 y no como simulador del proceso. Nótese que el modelo actúa en paralelo con el proceso real, por lo que los modelos de identificación en serie-paralelo pueden utilizarse en esta aplicación. En cualquier caso, los modelos construidos con redes de neuronas son de interés puesto

que, su estructura permite calcular de forma sencilla la aproximación del Jacobiano del proceso.

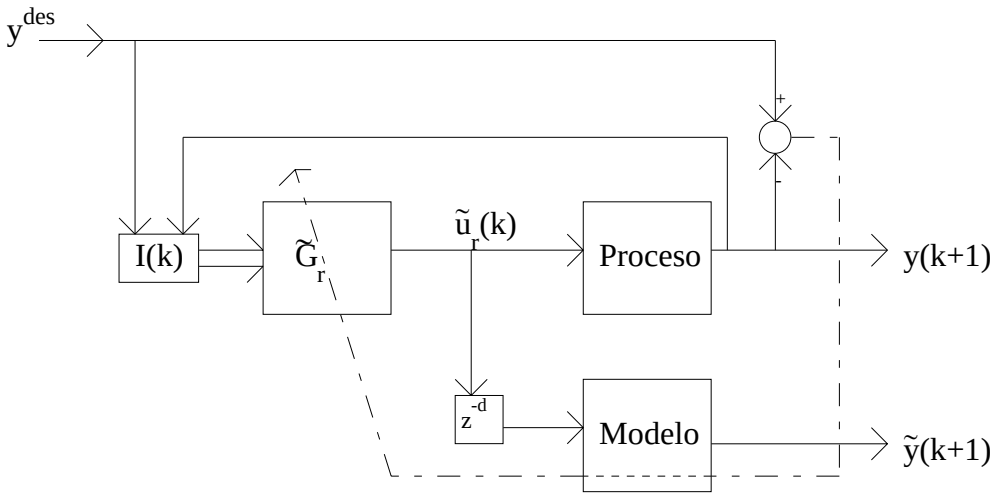


Figura 3.6. Esquema de aprendizaje especializado utilizando el proceso real.

Anteriormente se comentaba que una aproximación no adecuada del Jacobiano del proceso puede tener una repercusión negativa en los resultados de control. Puesto que en este esquema dicha aproximación se obtiene calculando el Jacobiano de un modelo del proceso, $\frac{\partial \tilde{y}(k+1)}{\partial \tilde{u}_r(k-d)}$, su exactitud depende, obviamente, de la predicción $\tilde{y}(k+1)$ proporcionada por el modelo, es decir, de la capacidad del modelo para aproximar la dinámica del proceso.

Generalmente, los parámetros del modelo se identifican con los datos disponibles y previamente a su utilización en esta aplicación. Sin embargo, y para evitar problemas de poca exactitud del modelo en ciertas regiones, es aconsejable y conveniente realizar una identificación on-line de los parámetros del modelo como se indicaba en el apartado 3.2.2. Por tanto, en esta fase de aprendizaje, tanto los pesos del controlador como los pesos del modelo se adaptan en tiempo real.

3.3.2 Sistema de control predictivo propuesto

En el esquema de control inverso desarrollado anteriormente se observa que los parámetros del controlador se adaptan en base al error instantáneo medido en la salida

del proceso o del modelo (ecuaciones 3.37 y 3.41), por lo que el cálculo de la acción de control en cada instante de tiempo k está basado en la información actual y pasada sobre las variables que intervienen en el comportamiento del proceso. Sin embargo, y debido a la naturaleza dinámica de los procesos, la acción de control en un determinado instante de tiempo no sólo influye en la salida del proceso en el instante inmediatamente posterior, sino que también influye en las respuestas futuras de dicho proceso. Por tanto, y con el objetivo de obtener mejores controladores, se considera oportuno utilizar una información más completa sobre el comportamiento del proceso para determinar la acción de control.

Como ya se ha mencionado en el apartado 1.2.5, las estrategias de control predictivo consideran la evolución del proceso dinámico en el futuro para calcular la acción de control en cada instante de tiempo, consideración que les permite poseer una mayor información sobre el comportamiento del proceso. No obstante, estas estrategias de control presentan limitaciones, principalmente en lo referente a la utilización de modelos no lineales para predecir el comportamiento del proceso en el futuro. Como se comentaba en la sección 2.2, una estrategia de control predictivo utilizando un modelo no lineal puede ser impracticable, sobre todo en aplicaciones en tiempo real, ya que en cada instante de tiempo es necesario encontrar la solución del problema de optimización no lineal dado por la ecuación 1.26, lo cual supone un gran número de adaptaciones y, por tanto, un alto esfuerzo computacional. Por otra parte, la utilización de modelos lineales para evitar este problema no es precisamente la mejor solución, puesto que la capacidad de representación de dichos modelos es muy limitada y el éxito de las estrategias de control predictivo dependen, en gran medida, del modelo utilizado.

El esquema de control predictivo desarrollado en este trabajo se caracteriza, en primer lugar, porque hace uso de un modelo no lineal para predecir el comportamiento del proceso dinámico en el futuro. En segundo lugar, y con la finalidad de no tener que resolver un problema de optimización no lineal en cada periodo de muestreo para disponer de la acción de control, se introduce una red de neuronas que actuará como controlador del proceso, al igual que cuando se plantea una estrategia de control inverso. De este modo, la acción de control en cada instante de tiempo k vendrá dada por la salida de dicha red y no como resultado de un algoritmo de optimización no lineal.

La adaptación de los pesos de la red se realiza en tiempo real y para llevar a cabo una estrategia de control predictivo, es decir, los pesos de la red se adaptan siguiendo la dirección negativa del gradiente de una función error que evalúa la diferencia entre el comportamiento del modelo en el futuro y el comportamiento deseado:

$$e_c^p(k+1) = \frac{1}{2} \cdot \sum_{i=1}^{H} (y^{des}(k+d+i) - \tilde{y}(k+d+i))^2 \,,\ k \in N_0 \qquad (3.44)$$

siendo H y N_u números naturales que representan el horizonte de predicción y control respectivamente; $\tilde{y}(k+d+i)$, i=1,...,H, las predicciones sobre el horizonte de predicción proporcionadas por un determinado modelo; $y^{des}(k+d+i)$ i=1,...,H, las salidas deseadas a lo largo de dicho horizonte.

Evidentemente, para adaptar los pesos de la red es necesario utilizar un algoritmo de optimización no lineal. Sin embargo, la capacidad de la red para almacenar información en sus pesos, hace que en cada periodo de muestreo baste realizar una única adaptación, lo cual implica un menor esfuerzo computacional que el requerido por las técnicas de control predictivo disponibles en la actualidad. La consideración de esta red permite la construcción de esquemas de control predictivo no lineales practicables en tiempo real.

Justificación de la red de neuronas utilizada en la estrategia de control predictivo. Estructura del controlador

En el esquema de control predictivo desarrollado en este trabajo la acción de control aplicada al proceso en cada instante de tiempo es, como se ha dicho, la salida de una red de neuronas. La utilización de esta red de neuronas está motivada por el hecho de que, cuando se realiza una estrategia de control predictivo, existe una relación entre la acción de control y el resto de las variables que intervienen en el proceso, al igual que cuando se lleva a cabo una estrategia de control inverso. A continuación se muestra dicha relación.

Se supone, por simplicidad, que el horizonte de control es 0 (N_u=0), en cuyo caso la estrategia de control predictivo consiste en calcular, en cada instante de tiempo k, la acción de control u(k) (secuencia de longitud 1) que minimiza la función error definida

en la ecuación 3.44, bajo la restricción de que la acción de control durante el horizonte de predicción es constante e igual a $u(k)$, es decir $u(k) = \ldots = u(k+H)$.

Al igual que en el apartado anterior, se supone que el proceso dinámico puede representarse por un modelo NARMA de la siguiente forma:

$$y(k+1) = F(y(k),\ldots,y(k-n_y),u(k-d),\ldots,u(k-d-n_u)) \tag{3.45}$$

Suponiendo que las aproximaciones $\tilde{y}(k+d+i)$, $i = 1,\ldots,H$, proporcionadas por un modelo aproximado del proceso coinciden con las salidas reales del proceso, $y(k+d+i)$, $i = 1,\ldots,H$, y utilizando la ecuación 3.45, las ecuaciones de predicción se pueden escribir de la forma:

$$\tilde{y}(k+d+1) = F(y(k+d),\ldots,y(k+d-n_y),u(k),\ldots,u(k-n_u)) \tag{3.46}$$

$$\tilde{y}(k+d+2) = F(y(k+d+1),\ldots,y(k+d+1-n_y),u(k+1),\ldots,u(k+1-n_u)) \tag{3.47}$$

.
.
.

$$\tilde{y}(k+d+H) = F(y(k+d+H-1),\ldots,y(k+d+H-1-n_y),u(k+H-1),\ldots,u(k+H-1-n_u)) \tag{3.48}$$

Realizando el mismo razonamiento que en el apartado anterior para deducir la estructura del controlador inverso e imponiendo, en este caso, que la acción de control permanezca constante en el horizonte de predicción, $u(k) = u(k+1) = \ldots = u(k+H-1) = u(k+H)$, se obtiene que los términos $y(k+1),\ldots,y(k+d),\ldots,y(k+d+H-1)$ que aparecen en los segundos miembros de las igualdades 3.46-4.48 se pueden expresar en función de la información disponible al instante de tiempo k, $y(k),\ldots,y(k-n_y),u(k),\ldots,u(k-d),\ldots,u(k-d-n_u)$, y, por tanto, las predicciones $\tilde{y}(k+d+i)$, $i = 1,\ldots,H$, son también función de dicha secuencia.

Teniendo en cuenta las nuevas expresiones para las ecuaciones de predicción 3.46-3.48, la acción de control $u(k)$ que minimiza la función error definida en 3.44 puede expresarse en términos de la secuencia $y^{des}(k+d+1),\ldots,y^{des}(k+d+H),y(k),\ldots,y(k-n_y),\ u(k-1),\ldots,u(k-d-n_u)$. De este modo, la solución del problema de optimización puede escribirse de la siguiente forma:

$$u(k) = P(y^{des}(k+d+1),\ldots,y^{des}(k+d+H),y(k),\ldots,y(k-n_y),u(k-1),\ldots,u(k-d-n_u)),\ k \in N_0 \tag{3.49}$$

siendo $y(-n_y) = \ldots = y(0)$, $u(-d - n_u) = \ldots = u(-1) = u_{-1} \in \Re$.

Suponiendo que el objetivo de control es constante durante el horizonte de predicción, es decir, $y^{des}(k+d+1) = \ldots = y^{des}(k+d+H) = y^{des}$, se obtiene:

$$u(k) = P(y^{des}, y(k), \ldots, y(k-n_y), u(k-1), \ldots, u(k-d-n_u)) \qquad (3.50)$$

En el caso de que la función P fuera conocida, la expresión dada por la ecuación 3.50 proporcionaría la acción de control que hay que aplicar al proceso, en cada instante de tiempo, para que alcance el objetivo de control, y^{des}, cuando se plantea una estrategia de control predictivo, sin necesidad, por tanto, de tener que resolver un problema de optimización no lineal. Sin embargo, el conocimiento de esta función puede llegar a ser complicado, puesto que el funcional F en la ecuación 3.45 es no lineal.

No obstante, y a pesar de que la expresión obtenida para la acción de control (ecuación 3.50) no es aplicable (debido al desconocimiento de la función P), es posible afirmar que, cuando se plantea una estrategia de control predictivo, existe también una relación entre la acción de control en un determinado instante de tiempo k y las variables que intervienen en el comportamiento dinámico del proceso, al igual que cuando se realizaba el control inverso. Además, la expresión 3.50 permite conocer el número de variables que determinan dicha relación.

En base a este hecho, se decide utilizar una red de neuronas para aproximar el funcional P y, por tanto, para calcular la acción de control. Introduciendo el vector de entrada a la red $I(k) = (y^{des}, y(k), \ldots, y(k-n_y))$ y considerando $d+n_u$ neuronas de contexto, la acción de control aplicada al proceso cuando se lleva a cabo una estrategia de control predictivo es la salida de la red parcialmente recurrente descrita en el apartado 3.1.2:

$$\tilde{u}_r(k) = \tilde{P}_r(I(k), R(k), W_{\tilde{P}_r}) = \tilde{P}_r(y^{des}, y(k), \ldots, y(k-n_y), R_1(k), \ldots, R_{d+n_u}(k), W_{\tilde{P}_r}) \qquad (3.51)$$

siendo $R_i(k) = \tilde{u}_r(k-i)$, $i = 1, \ldots, d+n_u$, y $W_{\tilde{P}_r}$ el conjunto de parámetros ajustables de la red de neuronas.

Cuando el objetivo de control permanece constante durante el horizonte de predicción, aparentemente la ley de control predictivo (ecuación 3.51) adopta la misma

forma que la ley de control inverso (ecuación 3.35). Sin embargo, existe una diferencia fundamental entre ellas y es, precisamente, la función error utilizada para llevar a cabo el aprendizaje de la red. En este caso, el conjunto de parámetros $W_{\tilde{P}_r}$ de la red parcialmente recurrente se adaptan para realizar una estrategia de control predictivo, es decir, utilizando el gradiente de la función error dada en 3.44, la cual evalúa el comportamiento del modelo a lo largo del horizonte de predicción. En las estrategias de control inverso en tiempo real, sin embargo, los parámetros de la red se adaptan en base al gradiente del error instantáneo (ecuación 3.41).

Algoritmo de aprendizaje

El entrenamiento del controlador predictivo se lleva a cabo en tiempo real y los pesos de la red se adaptan siguiendo la dirección negativa del gradiente de la función error dada en 3.44. A continuación se deduce la ley de adaptación.

Dado $w_{\tilde{P}_r}$ un peso del controlador neuronal, la variación de la función error definida en la ecuación 3.44 con respecto a este parámetro es:

$$\frac{\partial e_c^p(k+1)}{\partial w_{\tilde{P}_r}} = -\sum_{i=1}^{H}(y^{des} - \tilde{y}(k+d+i)) \cdot \frac{\partial \tilde{y}(k+d+i)}{\partial w_{\tilde{P}_r}},\ k \in N_0 \tag{3.52}$$

Al igual que cuando se realiza el aprendizaje especializado del controlador inverso, es necesario propagar los errores evaluados en la salida del modelo durante el horizonte de predicción hacia la salida del controlador, propagación que se basa también en el Jacobiano de dicho modelo. Puesto que el horizonte de control se considera igual a 0, la acción de control aplicada al modelo permanece constante durante el horizonte de predicción $\tilde{u}_r(k) = \ldots = \tilde{u}_r(k+H)$, por lo que al aplicar la regla de la cadena se obtiene que la variación de la salida del modelo con respecto al parámetro $w_{\tilde{P}_r}$ es:

$$\frac{\partial \tilde{y}(k+d+i)}{\partial w_{\tilde{P}_r}} = \frac{\partial \tilde{y}(k+d+i)}{\partial \tilde{u}_r(k)} \cdot \frac{\partial \tilde{u}_r(k)}{\partial w_{\tilde{P}_r}},\ i = 1,\ldots,H,\ k \in N_0 \tag{3.53}$$

Los términos $\frac{\partial \tilde{y}(k+d+i)}{\partial \tilde{u}_r(k)}$, $i = 1,\ldots,H$, representan el Jacobiano del modelo a lo largo del horizonte de predicción y determinan la correspondencia entre el error medido

en la salida del modelo y el error en la salida del controlador neuronal predictivo. El término $\frac{\partial \tilde{u}_r(k)}{\partial w_{\tilde{P}_r}}$ representa la variación total de la salida de la red parcialmente recurrente en el instante de tiempo k con respecto al parámetro $w_{\tilde{P}_r}$ y se calcula utilizando retropropagación dinámica (ecuación 3.11).

Los pesos del controlador predictivo se adaptan entonces en cada instante de tiempo k+1 de acuerdo con la siguiente ley:

$$w_{\tilde{P}_r}^{(k+1)} = w_{\tilde{P}_r}^{(k)} + \alpha \cdot \frac{\partial \tilde{u}_r(k)}{\partial w_{\tilde{P}_r}} \cdot \sum_{i=1}^{H} (y^{des} - \tilde{y}(k+d+i)) \cdot \frac{\partial \tilde{y}(k+d+i)}{\partial \tilde{u}_r(k)}, \; k \in N_0 \qquad (3.54)$$

Obsérvese que, debido a que el horizonte de control es considerado igual a 0, el término $\frac{\partial \tilde{u}_r(k)}{\partial w_{\tilde{P}_r}}$ debe calcularse una única vez en cada periodo de muestreo.

La actuación del controlador predictivo junto con su aprendizaje se resume entonces en los siguientes pasos: en cada instante de tiempo k se presenta el proceso la salida de la red parcialmente recurrente, $\tilde{u}_r(k)$, para el vector de entrada $I(k) = (y^{des}, y(k), \ldots, y(k-n_y))$; dicho valor se presenta también al modelo elegido para predecir el comportamiento del proceso en el futuro como se muestra en la figura 3.7. Una vez evaluados los errores $(y^{des} - \tilde{y}(k+d+i))$ y los Jacobianos $\frac{\partial \tilde{y}(k+d+i)}{\partial \tilde{u}_r(k)}$ a lo largo del horizonte de predicción, se espera hasta el próximo instante de tiempo en el que los pesos de la red se adaptan utilizando la ley dada en la ecuación 3.54.

Como ya se ha mencionado, en los esquemas de control predictivo los modelos tienen que predecir el comportamiento del proceso dinámico a lo largo del horizonte de predicción, por lo que tienen que utilizarse modelos capaces de actuar como simulador del proceso en el intervalo de tiempo $[k+d, k+d+H]$. Por otra parte, y como se muestra en la figura 3.7, el modelo elegido es también el encargado de propagar hacia el controlador neuronal predictivo el error medido en la salida de dicho modelo en el intervalo de tiempo $[k+d, k+d+H]$. Como consecuencia, los simuladores NARMA

desarrollados en el apartado 3.2.2 son una posibilidad interesante para construir sistemas de control predictivo no lineal.

Es aconsejable corregir en tiempo real los errores existentes en el modelo, ya que el éxito de la estrategia de control predictivo depende, obviamente, de la precisión del modelo elegido para predecir el comportamiento del proceso. Para ello, al igual que en el esquema de control inverso con aprendizaje especializado utilizando el proceso real, se realiza una identificación on-line de los parámetros del modelo como se indicaba en el apartado 3.2.2.

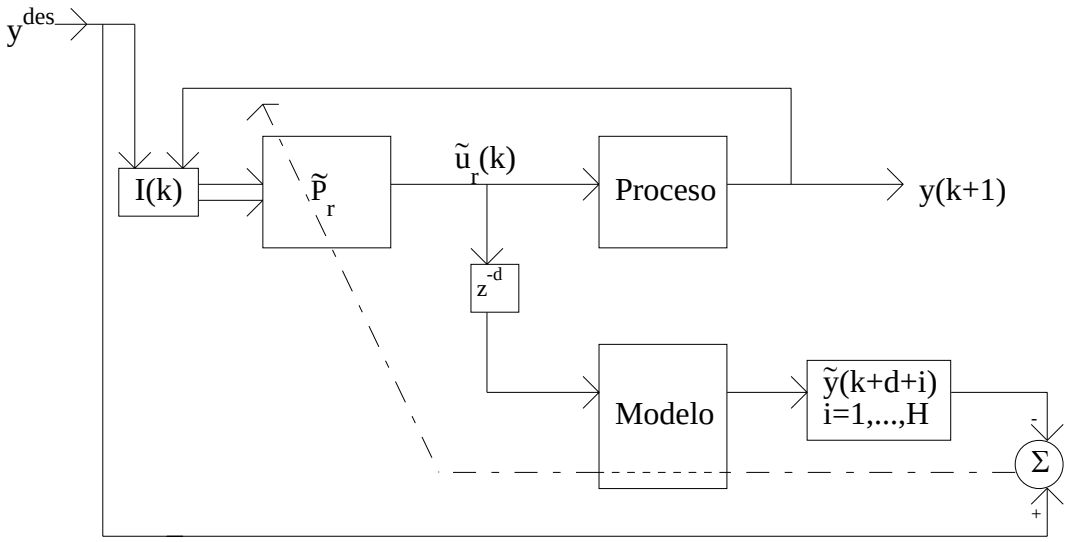


Figura 3.7. Esquema de aprendizaje del controlador predictivo.

Inicialización de los pesos del controlador predictivo

Cuando se plantea un esquema de control on-line en el que intervienen las redes de neuronas, los resultados de control dependen de los pesos con los que se inicialice el controlador neuronal. Como consecuencia, es conveniente inicializar el controlador predictivo con pesos que ya posean cierta información sobre el comportamiento dinámico del proceso, y poder así evitar situaciones de no control o inestabilidad.

Es posible, al igual que cuando se plantea una estrategia de control inverso, realizar un aprendizaje previo del controlador predictivo utilizando el mismo esquema descrito anteriormente, pero sustituyendo el proceso real por un modelo del proceso. De este modo, el objetivo de control deseado se presenta varias veces al controlador y éste

aprende así a realizar una estrategia de control predictivo para que el proceso alcance dicho objetivo.

Como se observaba anteriormente, cuando el objetivo de control permanece constante durante el horizonte de predicción, la estructura de la ley de control predictivo (ecuación 3.51) coincide con la ley de control inverso (ecuación 3.35). Como consecuencia, otro camino alternativo para inicializar los pesos del controlador predictivo es inicializar sus parámetros con los obtenidos después de haber finalizado la fase de aprendizaje generalizado para que la red aprenda la dinámica inversa del proceso. Claramente, al inicializar la ley dada en la ecuación 3.51 con estos parámetros, dicha ley no realizará una estrategia de control predictivo, pero sí posee información suficiente como para evitar situaciones de no control del proceso en los primeros pasos de actuación.

Capítulo 4:

Aplicación de las técnicas desarrolladas. Modelización y control de la temperatura del fluido que circula en la camisa de un reactor químico

En este capítulo se aplican las técnicas desarrolladas en el capítulo 3. Para ello, se utiliza un proceso dinámico real, concretamente el proceso que gobierna la evolución de la temperatura del fluido para el intercambio de calor que circula en la camisa de un reactor químico.

La sección 4.1 contiene una descripción de dicho reactor químico y de los circuitos de calentamiento y enfriamiento, así como de la instalación. En esta misma sección se describen también los mecanismos para llevar a cabo el control, tanto de la temperatura del reactor como de la temperatura del fluido de el intercambio de calor.

En la sección 4.2 se trata el problema de la modelización e identificación del proceso dinámico en cuestión, lo cual es equivalente a modelizar los circuitos de calentamiento/enfriamiento del reactor. Los modelos NARMA neuronales se desarrollan en el apartado 4.2.1. En primer lugar, se realiza la fase de modelización mediante la cual se determina el número de variables que deben tenerse en cuenta para la construcción de estos modelos, así como la longitud de las secuencias discretas de las variables de entrada y salida del proceso. Posteriormente, se describe la fase de identificación o entrenamiento de los modelos NARMA neuronales. Con la finalidad de poder validar experimentalmente los resultados presentados en el apartado 3.2.1, el funcional no lineal que determina el modelo NARMA se aproxima utilizando tanto el perceptron multicapa como la red parcialmente recurrente propuesta en este trabajo. Esta sección finaliza con el apartado 4.2.2, donde se presentan las aproximaciones de la temperatura del fluido de intercambio de calor que proporciona un modelo físico

disponible para explicar el comportamiento de dicha temperatura. Dicho modelo se utiliza para poder verificar las capacidades de representación de los modelos NARMA neuronales en paralelo propuestos.

En la sección 4.3 se lleva a cabo el control de la temperatura del fluido de intercambio de calor utilizando el controlador PID (proporcional-integral-derivativo) y los sistemas de control inverso y predictivo descritos en la sección 3.3 del capítulo anterior. El apartado 4.3.1 se dedica a las estrategias de control inverso; en primer lugar, se especifica la estructura del controlador y a continuación se realizan las tres fases de aprendizaje, presentándose los resultados de control obtenidos al finalizar cada una de estas fases. Las estrategias de control predictivo se aplican en el apartado 4.3.2.

Por último, en la sección 4.4 se realiza un análisis de los resultados obtenidos, tanto en lo referente a la modelización de la temperatura del fluido de intercambio de calor que circula en la camisa del reactor químico, como lo que se refiere al control de dicha temperatura. Este análisis incluye también un estudio comparativo de los esquemas de control inverso y predictivo propuestos.

4.1 Descripción del reactor y del circuito de recirculación

En el Centro Común de Investigación de la Comunidad Europea situado en Ispra (Italia) y dentro del proyecto FIRES (Facility for Investigating Runaway Events Safely), se llevan a cabo investigaciones sobre la dinámica de reactores químicos en condiciones cercanas a la pérdida de control, con el consiguiente riesgo de explosión térmica. Para ello se dispone de un reactor de 100 litros, el cual está equipado con una camisa en la que circula el fluido de intercambio de calor.

Los circuitos de enfriamiento/calentamiento (figura 4.1) del reactor están formados por dos bucles, en los que circula una mezcla 50/50% de etilenglycol-agua con un flujo de 12 m^3/h. El circuito principal está conectado a la camisa del reactor y contiene una fuente de calor: 20 m del tubo de metal se calientan aplicando una corriente directa, pudiéndose proporcionar hasta 45kW de potencia. El circuito secundario proporciona líquido frío al bucle principal y posee un tanque de 2m^3 de capacidad, el cual está

conectado a la unidad de refrigeración. Este depósito de líquido frío se mantiene a una temperatura de -20^{o}C y puede utilizarse para proporcionar un enfriamiento rápido en situaciones de emergencia, utilizando las válvulas de emergencia (figura 4.1).

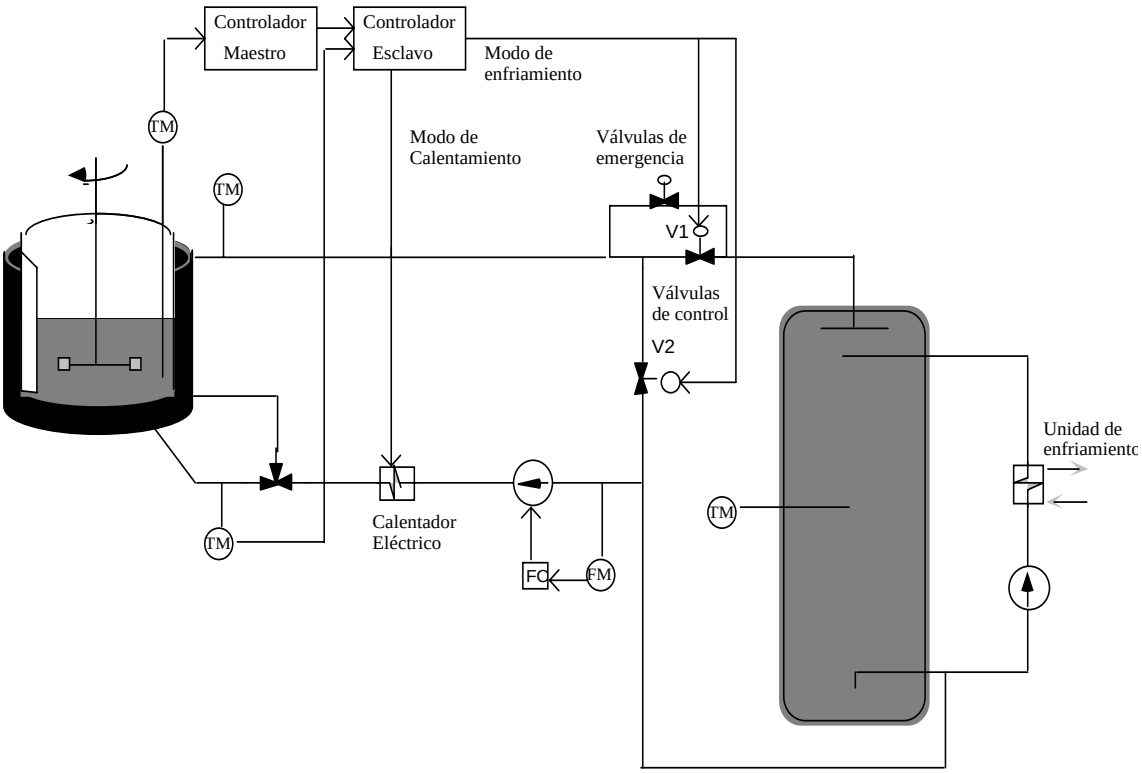


Figura 4.1 Esquema del circuito de calentamiento y enfriamiento.

Estos circuitos incorporan instrumentos que proporcionan medidas en tiempo real de las distintas variables del proceso. Se ha prestado especial atención a la exactitud y a la sensibilidad de estos instrumentos de medida (representados como TM, FM y FC en la figura 4.1), de forma que se pueda garantizar la seguridad de operación del reactor y la apropiada supervisión de los parámetros principales que intervienen en la reacción. Concretamente, los instrumentos que se utilizan para medir la temperatura del fluido para la transferencia de calor a la entrada y a la salida de la camisa del reactor, junto con su flujo, son de una elevada precisión.

La instalación está organizada de forma jerárquica, y pueden distinguirse tres niveles de funcionalidad como se muestra en la figura 4.2. La parte superior de esta jerarquía (nivel 3) está formada por una estación de trabajo que actúa como supervisor. El nivel 2 incluye los mecanismos de adquisición de datos, los convertidores

analógicos/digitales y digitales/analógicos, y los sistemas de actuación, los cuales envían las señales eléctricas a las bombas y válvulas. Por otra parte, en este nivel se incluyen también algunos sistemas de control, como por ejemplo el controlador PID del que dispone actualmente el proceso para el control de las temperaturas. Por último, el nivel 1 que contiene los elementos del proceso, como válvulas bombas, agitador, sensores, etc,...

El nivel 2 está conectado directamente al proceso y a otros mecanismos de salida, mientras que el nivel 3 actúa en el proceso a través del nivel 2, el cual se encarga de ejecutar las especificaciones dadas por el nivel 3.

La instalación posee una segunda estación de trabajo, desde la cual es posible leer las variables medidas y enviar órdenes al proceso pasando por los niveles 3 y 2. Todos los experimentos realizados en esta aplicación se han llevado a cabo desde esta última estación de trabajo.

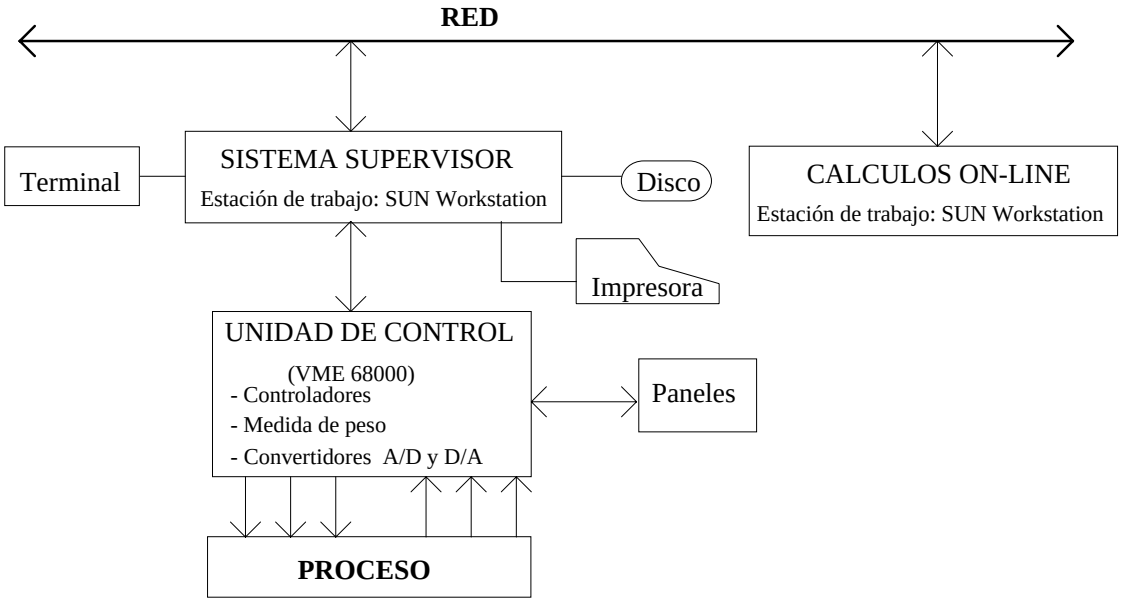


Figura 4.2. Sistema de adquisición de datos y control de la instalación de FIRES.

Durante la realización de los experimentos, los valores de las variables que intervienen en el proceso, los estados de los diferentes dispositivos y las acciones de control se presentan en una pantalla y se graban en un disco magnético.

La operación del proceso puede realizarse de varias formas:

a) Manualmente.

b) A través de un panel de control situado en el nivel 2.

c) A través de la consola situada en el nivel 3.

d) De forma totalmente automática, siguiendo los pasos de un algoritmo programado por el operador, los cuales se pueden ubicar en una de las estaciones de trabajo de la instalación.

Mecanismo de control

El control de la temperatura del reactor, Tm, se lleva a cabo a través del ajuste o del control de la temperatura del fluido de intercambio de calor que circula en la camisa del reactor, Te. Se dispone, por tanto, de dos sistemas de control, como se indica en la figura 4.3. El primero de ellos, llamado *controlador maestro*, es el que dicta el valor que debe tomar la temperatura del fluido de intercambio de calor, Te^{des}, para que la temperatura del reactor alcance su valor deseado, Tm^{des}. El segundo controlador, también llamado *controlador esclavo*, es el que se encarga de controlar directamente la temperatura del fluido de intercambio de calor, con el objetivo de que dicha temperatura tome el valor deseado proporcionado por el controlador maestro.

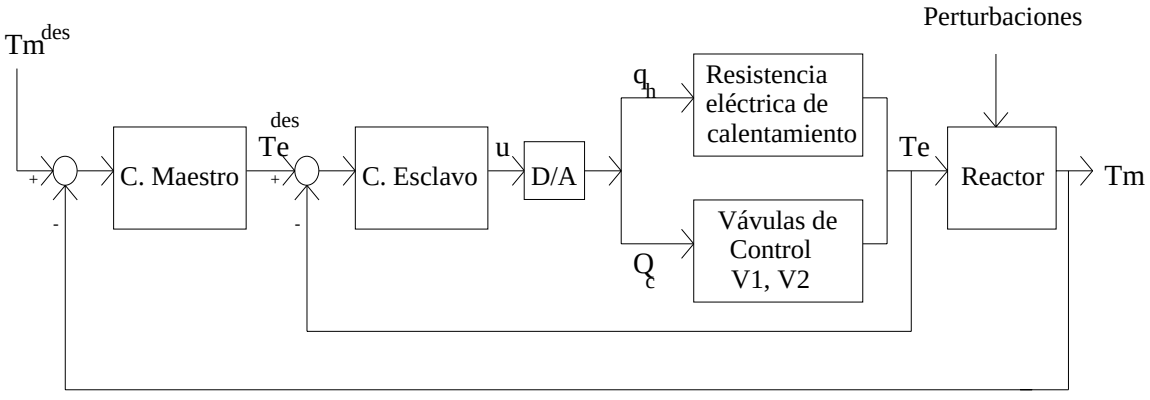


Figura 4.3. Esquema del control de la temperatura del reactor químico.

La salida del controlador esclavo, u, es una señal que actúa para enfriar o calentar la temperatura del fluido de intercambio de calor, dependiendo del objetivo de control deseado Te^{des}. La forma de actuación de esta señal es la siguiente. Una salida del controlador esclavo entre 11 % y 100 % (equivalente a una señal digital entre 450 y

4095) significa que se producirá un enfriamiento en el fluido que circula en la camisa del reactor, siendo el 100 % el que corresponde al máximo enfriamiento. En este caso, la válvula V1 (figura 4.1) se abre con el porcentaje correspondiente y la válvula V2 (figura 4.1) se cierra en paralelo con el mismo porcentaje. Al abrir la válvula V1, el líquido frío (-20^{o}C) almacenado en el depósito entra en el circuito principal con un flujo Q_c, enfriando el fluido de intercambio de calor que se encuentra en la camisa del reactor. En la figura 4.4 se muestra la correlación entre dicho flujo y la señal digital enviada por el controlador esclavo.

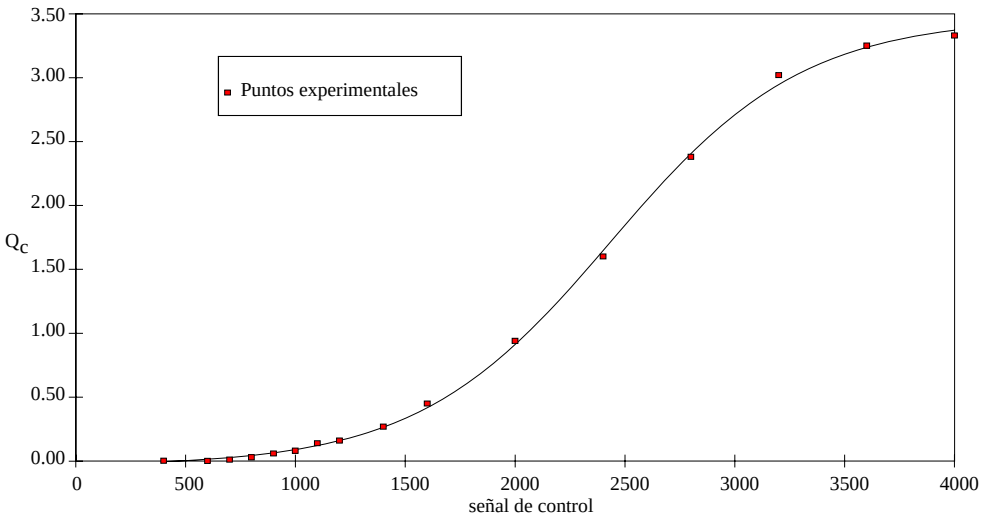


Figura 4.4. Flujo del circuito de enfriamiento en función de la señal de control.

Para salidas del controlador esclavo entre 0 % y 10 % (equivalente a una señal digital entre 0 y 450) se produce el modo de calentamiento, correspondiendo el 0 % al máximo calentamiento. El fluido de intercambio de calor se calienta debido a la aplicación directa de una potencia q_h generada por una resistencia eléctrica (figura 4.1). La correlación entre dicha potencia y la señal digital enviada por el controlador esclavo se muestra en la figura 4.5. Como puede observarse en las figuras 4.4. y 4.5, las relaciones entre la acción de control y las magnitudes Q_c, q_h son no lineales.

Los controladores maestro y esclavo que posee actualmente la instalación para su funcionamiento son sistemas de control PID (apartado 1.2.5).

El objetivo perseguido en esta aplicación es controlar la temperatura del fluido que circula en la camisa del reactor, por lo que los sistemas de control neuronales desarrollados ocupan la posición del controlador esclavo (figura 4.3), sustituyendo al correspondiente controlador PID.

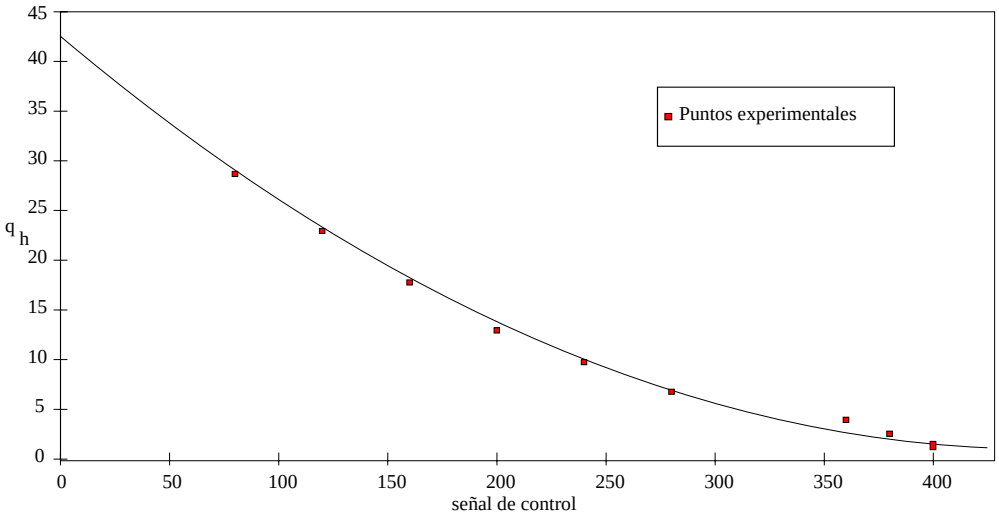


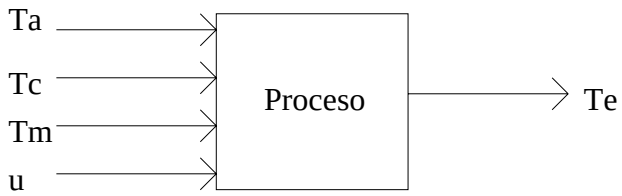
Figura 4.5. Potencia de calentamiento en función de la señal de control.

4.2 Modelización de los circuitos de calentamiento y enfriamiento

Para poder controlar la temperatura del fluido de intercambio de calor utilizando los controladores neuronales desarrollados, es necesario, en primer lugar, reproducir el comportamiento dinámico de dicha temperatura, lo cual es equivalente a construir un modelo de los circuitos de calentamiento y enfriamiento del reactor, descritos en la sección 4.1.

El proceso dinámico que gobierna la evolución de la temperatura de dicho fluido puede representarse de forma esquemática como se indica en la figura 4.6. De todas las variables de entrada al proceso, sólo la variable u puede manipularse, pues el resto son las salidas de otros procesos dinámicos o simplemente variables del entorno.

En esta sección se construyen modelos NARMA neuronales y se presentan los resultados que proporciona un modelo físico disponible.



Ta : Temperatura ambiente
Tc : Temperatura del depósito de frío
Tm : Temperatura del contenido del reactor
u : Señal digital enviada por el controlador esclavo
Te : Temperatura del fluido de intercambio de calor de la camisa

Figura 4.6. Representación esquemática del proceso dinámico

Adquisición de datos

Con la finalidad de obtener estimaciones de los parámetros que intervienen en los modelos que se van a construir, y con la finalidad de comprobar las capacidades de dichos modelos para representar al proceso en situaciones diferentes a las utilizadas para determinar sus parámetros (capacidad de generalización), se han obtenido diferentes conjuntos de datos experimentales, cuyas condiciones se especifican en la tabla 4.1.

Para la obtención de estos conjuntos de datos experimentales, la señal de control u se manipula directamente desde la consola de la primera estación de trabajo (nivel 3); esta señal junto con las medidas de las variables que intervienen en el proceso (Ta, Tc, Tm y Te) se almacenan en ficheros (segunda estación de trabajo), de modo que dichos datos puedan utilizarse para realizar una identificación off-line de los parámetros de los modelos.

En todos los experimentos la variable de control toma valores que cubren su rango normal de operación y dichos valores se mantienen constantes durante un cierto intervalo de tiempo. El periodo de muestreo es aproximadamente de 10 segundos.

	Contenido del reactor	Velocidad de agitación	Te (inicial)
Dato 1	vacío	0 r.p.m.	20º C
Dato 2	≈ 80 l. de agua	199.11 r.p.m.	20º C

Dato 3	≈ 80 l. de agua	300.99 r.p.m.	60º C

Tabla 4.1. Condiciones bajo las cuales se han obtenido los conjuntos de datos experimentales.

4.2.1 Modelos NARMA neuronales

De acuerdo con la representación esquemática del proceso mostrada en la figura 4.6, y bajo ciertas condiciones de observabilidad, el proceso dinámico que describe el comportamiento de la temperatura del fluido que circula en la camisa del reactor se puede representar por un modelo NARMA de la siguiente forma:

$$Te(k+1) = F(Te(k),\ldots,Te(k-n_e),u(k-d),\ldots,u(k-d-n_u),Tm(k),\ldots,Tm(k-n_m),$$
$$Tc(k),\ldots,Tc(k-n_c),Ta(k),\ldots,Ta(k-n_a)) \qquad (4.1)$$

siendo n_e, n_u, n_m, n_c y n_a números naturales que indican la longitud de las secuencias discretas utilizadas para representar la historia de las variables que intervienen en el proceso, Te, u, Tm, Tc y Ta respectivamente; d es el retraso natural de la variable de salida del proceso Te respecto a la variable de entrada u; para el resto de las variables de entrada al proceso, el retraso es prácticamente despreciable, por lo que suponemos que no existe.

En base a la experiencia sobre el proceso dinámico que se está tratando, así como en previas simulaciones realizadas, se considera conveniente dividir el modelo NARMA dado en la ecuación 4.1, en dos partes: una parte lineal, formada únicamente por Te(k), y una parte no lineal, que incluye el resto de las variables que intervienen en el modelo NARMA. De este modo, en lugar de identificar y modelizar la temperatura del fluido que circula en la camisa del reactor, se decide modelizar e identificar el incremento de dicha temperatura $\Delta Te(k+1) = Te(k+1) - Te(k)$. Manteniendo la notación F para indicar la relación no lineal, el modelo NARMA resultante adopta, entonces, la siguiente forma:

$$Te(k+1) = Te(k) + F(Te(k-1),\ldots,Te(k-n_e),u(k-d),\ldots,u(k-d-n_u),$$
$$Tm(k),\ldots,Tm(k-n_m),Tc(k),\ldots,Tc(k-n_c),Ta(k),\ldots,Ta(k-n_a)) \qquad (4.2)$$

Fase de modelización

Antes de aproximar el funcional F utilizando una red de neuronas, es necesario establecer la longitud de las secuencias discretas (n_e, n_u, n_m, n_c y n_a) y el retraso natural del proceso.

Es necesario realizar un estudio que permita establecer las variables verdaderamente influyentes en la dinámica del proceso y la longitud óptima para representar la historia de dichas variables, de forma que el modelo resultante pueda representar adecuadamente al proceso real. Este estudio se basa, normalmente, en analizar las características del proceso y las leyes físicas que gobiernan su comportamiento, así como en la propia experiencia del experto con dicho proceso. Esto permite disponer de cierta información, que denominamos, en este contexto, conocimiento empírico del proceso.

El conocimiento empírico disponible sobre el proceso que se está tratando informa, en primer lugar, que la temperatura ambiente Ta y la temperatura del depósito de frío Tc permanecen prácticamente constantes durante la realización de un experimento, por lo que no tiene sentido considerar una secuencia discreta de longitud mayor de uno para representar su historia. Como consecuencia, se elige $n_a=n_c=0$ en el modelo NARMA dado por la ecuación 4.2. Con respecto a los números naturales d, n_e, n_u y n_m, el conocimiento empírico proporciona, únicamente, valores aproximados: $3 \le d \le 6$, $0 \le n_u \le 20$, $1 \le n_e \le 2$, $0 \le n_m \le 1$. Eligiendo, entonces, d=3, n_u=20, n_e=2, n_m=1 se puede construir un modelo NARMA que explique la dinámica del proceso casi en su totalidad, el cual viene dado por la siguiente expresión (modelo1):

$$Te(k+1) = Te(k) + F(Te(k-1), Te(k-2), u(k-3), \ldots, u(k-23), Tm(k), Tm(k-1), Tc(k), Ta(k)) \quad (4.3)$$

Se observa que el conocimiento empírico sobre el proceso no proporciona las cantidades óptimas para las longitudes de las secuencias discretas, sino más bien rangos o aproximaciones en los que dichas cantidades tienen que ser elegidas. El elegir el rango superior, como se ha hecho para la construcción del modelo dado por la ecuación 4.3, garantiza la obtención de un modelo que pueda representar adecuadamente al proceso dinámico, aunque también es necesario señalar que, en dicho modelo, interviene un gran número de variables. Para poder simplificar el modelo dado por la

ecuación 4.3, se decide eliminar las variables menos influyentes, es decir las variables que no aporten información significativa al modelo.

Si se recurre al conocimiento empírico para realizar esta simplificación, sería necesario considerar una serie de modelos candidatos, los cuales se determinarían eligiendo, dentro de sus rangos respectivos, diferentes valores para d, n_e, n_u y n_m. Una vez realizada la fase de identificación de cada uno de los modelos candidatos, se procedería a la selección del más simple, pero que a su vez proporcione la misma representación del proceso que la proporcionada por el modelo 4.3. Sin embargo, esto puede ser un trabajo muy laborioso, pues habría que considerar un gran número de modelos candidatos.

Para evitar este trabajo, es interesante disponer de métodos que proporcionen información sobre la influencia que tienen en la salida del proceso cada una de las variables consideradas para la construcción del modelo.

En esta aplicación dicha información se obtiene del siguiente modo: en primer lugar, se entrena un perceptron multicapa con un número de variables de entrada que se establece de acuerdo con la información que se pretende obtener. Posteriormente, y para cada entrada I_i del perceptron multicapa, se evalúa la media en valor absoluto de los pesos asociados a las conexiones que unen la neurona de entrada I_i con todas las neuronas de la segunda capa, w^1_{ij}, $j=1,...,n_2$. Por tanto, a cada entrada del perceptron multicapa I_i se le asocia el siguiente valor:

$$m_i = \frac{1}{n_2} \cdot \sum_{j=1}^{n_2} \left| w^1_{ij} \right|, \; i=1,...,n_1 \qquad (4.4)$$

Para el proceso que se está tratando, la cantidad definida anteriormente proporciona información sobre la influencia en $\Delta Te(k+1)$ de cada una de las variables consideradas para construir el modelo NARMA dado por la ecuación 4.3, como se verá a continuación. Dichas cantidades serán, por tanto, útiles para obtener un modelo simplificado, ya que permitirán determinar cuáles son las variables menos significativas y que, por tanto, pueden ser eliminadas de dicho modelo.

Simplificación del modelo dado por la ecuación 4.3

La simplificación del modelo dado por 4.3 se realiza en dos partes. La primera se lleva a cabo para determinar cuáles de las variables de entrada al proceso tienen que ser consideradas para la construcción del modelo NARMA. La segunda, para reducir la longitud de la secuencia discreta de la variable u.

1) Con la finalidad de determinar qué variables de entrada al proceso son las más influyentes en su comportamiento dinámico, se entrena un perceptron multicapa con entradas u, Tr, Te, Tc y Ta y con el objetivo de aproximar ΔTe. Las cantidades m_i obtenidas para cada una de las entradas se muestran en la figura 4.7. Interpretando que la magnitud de m_i mide la influencia de la variable I_i en la salida ΔTe, en el sentido de que a mayor magnitud mayor influencia, las cantidades que se muestran en la figura 4.7 reflejan que u y Te son las variables más importantes, destacándose la dominancia de u.

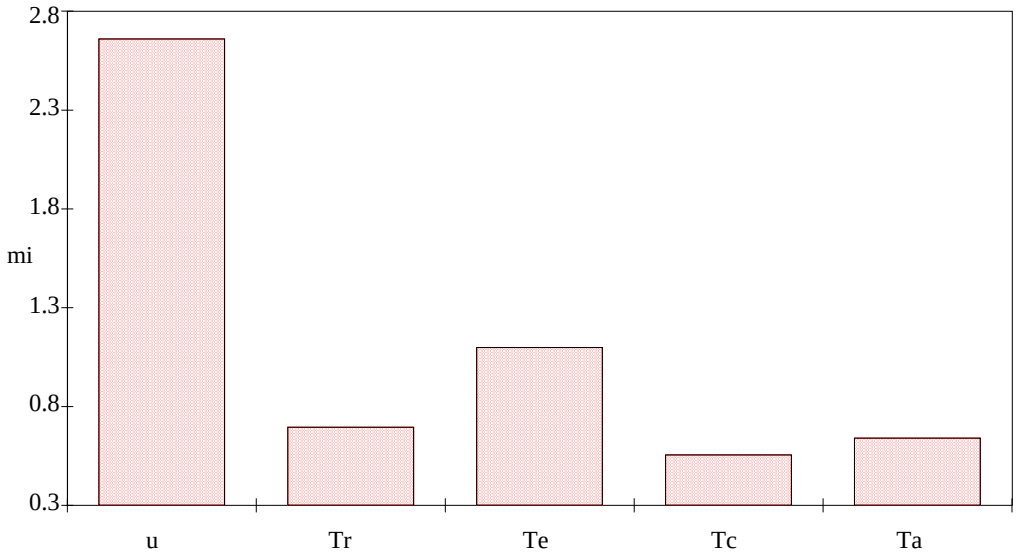


Figura 4.7. Cantidades m_i para las variables u, Tr, Te, Tc, Ta.

Estas conclusiones coinciden con el conocimiento a priori del proceso, para el cual la variable de control u es de vital importancia en la aproximación de ΔTe. No obstante, y con el objetivo de confirmar esta interpretación, se entrena un perceptron multicapa suprimiendo la variable u. Como puede observarse en la figura 4.8,

efectivamente, el perceptron con entradas Tr, Te, Tc y Ta presenta problemas para aproximar ΔTe.

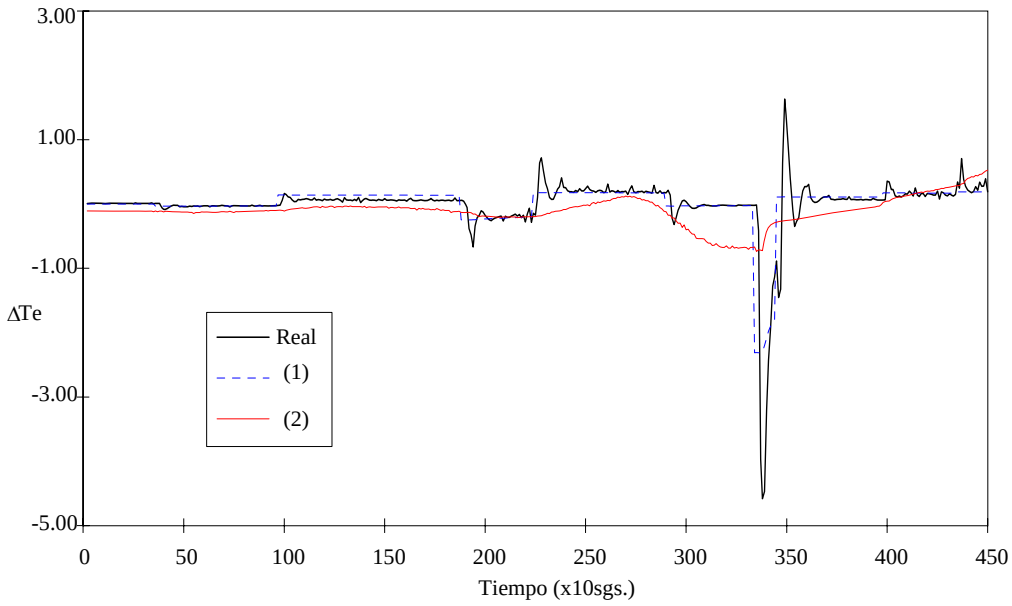


Figura 4.8. Aproximación de ΔTe obtenida con:
(1) Perceptron multicapa con entradas u, Tr, Te, Tc, Ta.
(2) Perceptron multicapa con entradas Tr, Te, Tc, Ta.

Por otra parte, en la figura 4.7 se observa también que Ta, Tc, Tm no poseen una influencia significativa en la dinámica de Te, pudiéndose entonces suprimir en el modelo NARMA dado por la ecuación 4.3. De hecho, al eliminar estas variables como entradas del perceptron multicapa, la aproximación de ΔTe es prácticamente indistinguible a la obtenida con el perceptron multicapa que las considera como entradas.

2) Con la finalidad de reducir la longitud de la secuencia discreta para la variable de entrada u en el modelo dado por la ecuación 4.3, se decide entrenar un perceptron multicapa para aproximar ΔTe que posea como entradas la secuencia u(k-23),...,u(k-3). Las cantidades m_i obtenidas se muestran en la figura 4.9, observándose la dominancia de la variable u(k-6), la cual corresponde exactamente con el verdadero retraso del proceso, es decir con la variable más importante de toda la secuencia discreta u(k-23),...,u(k-3).

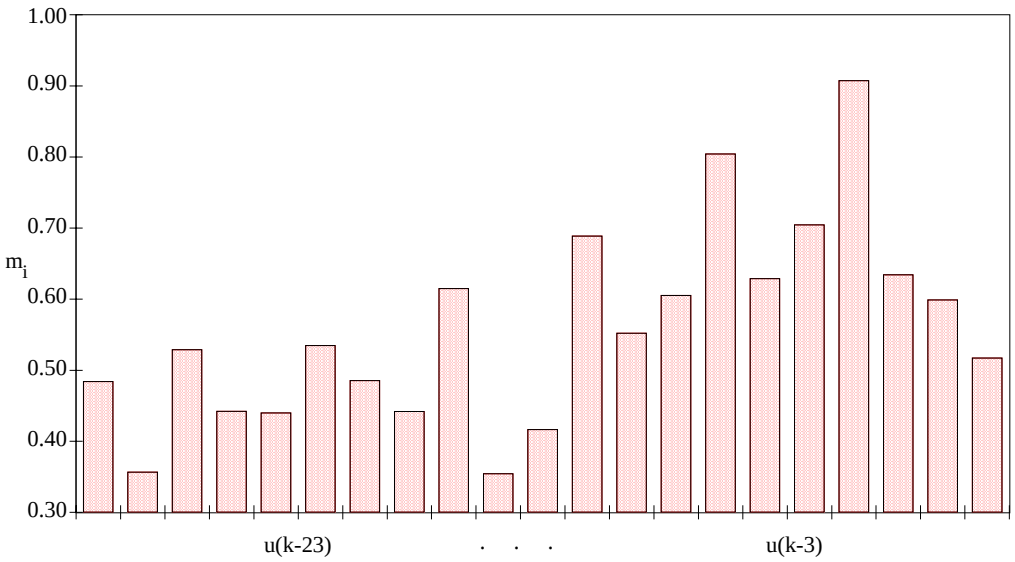


Figura 4.9. Cantidades m_i para la secuencia u(k-23),...,u(k-3)

Extrapolando los resultados obtenidos en el punto anterior, se pueden suprimir los términos de la secuencia u(k-23),...,u(k-3) cuyos valores asociados m_i corresponden a los valores más bajos.

La información obtenida en este último punto (figura 4.9) no proporciona una idea clara y exacta sobre los términos de la secuencia discreta u(k-23),...,u(k-3) que deben ser eliminados, aunque sí es posible deducir cierta información, como, por ejemplo, que generalmente los términos u(k-12),...,u(k-5) son más importantes que los términos u(k-23),...,u(k-13). En la figura 4.9, se observa también que existen términos de esta última secuencia, u(k-23),...,u(k-13), que predominan; ellos son u(k-23), u(k-21), u(k-18), u(k-15).

De acuerdo con esta información y con la información obtenida en la primera parte, se elige el siguiente modelo simplificado (modelo2):

$$\mathrm{Te}(k+1) = \mathrm{Te}(k) + F(\mathrm{Te}(k-1) - \mathrm{Tm}(k-1), \mathrm{Te}(k-2) - \mathrm{Tm}(k-2),$$
$$u(k-5), \ldots, u(k-12), u(k-15), u(k-18), u(k-21), u(k-23)) \qquad (4.5)$$

Nótese que, en el modelo dado por la ecuación 4.5, se han considerado las diferencias Te-Tm en lugar de Te y Tm. Como se verá más adelante, esto no se ha introducido con el objetivo de reducir más aún el número de variables, sino con vistas,

principalmente, a obtener modelos neuronales con buenas propiedades de generalización y extrapolación, es decir, modelos que respondan de forma adecuada en rangos de temperatura diferentes a los que se utilizan para llevar a cabo la estimación de sus parámetros.

Posteriormente, cuando se realice la fase de identificación de los modelos dados por la ecuaciones 4.3 y 4.5, se comprobará que ambos modelos pueden proporcionar representaciones similares del proceso dinámico. De este modo, se confirma la veracidad de la información que se ha obtenido al evaluar las cantidades m_i definidas por la ecuación 4.4, con el propósito de reducir el número de variables del modelo dado por la ecuación 4.3.

Fase de identificación

Una vez definida la estructura de los modelos, se realiza la fase de identificación o fase de aprendizaje; dicha identificación se lleva a cabo de forma off-line y utilizando el conjunto de datos experimentales *Dato1* dado en la tabla 4.1. *Dato2* y *Dato3* se utilizarán para observar las capacidades de generalización de los modelos considerados.

Las redes de neuronas utilizadas poseen funciones de activación sigmoidales cuya imagen es el intervalo [0,1] y están formadas por una única capa oculta; el número de neuronas en esta capa se determina por prueba y error. Para realizar la fase de identificación o aprendizaje, todas las variables de entrada a la red se normalizan en el intervalo [0,1] y los patrones se presentan de forma ordenada, siguiendo la evolución del tiempo.

Aproximación de F utilizando un perceptron multicapa

Con la finalidad de validar los resultados presentados en el apartado 3.2.1, el funcional F que determina los modelos NARMA dados por las ecuaciones 4.3 y 4.5 se aproxima, en primer lugar, utilizando el perceptron multicapa, obteniendo los modelos NARMA de identificación en serie-paralelo:

$$\tilde{Te}(k+1) = Te(k) + \tilde{F}(Te(k-1), Te(k-2), u(k-3), \ldots, u(k-23), Tm(k), Tm(k-1), Tc(k), Ta(k), W_{\tilde{F}}) \quad (4.6)$$

$$\tilde{Te}(k+1) = Te(k) + \tilde{F}(Te(k-1) - Tm(k-1), Te(k-2) - Tm(k-2),$$
$$u(k-5), \ldots, u(k-12), u(k-15), u(k-18), u(k-21), u(k-23), W_{\tilde{F}}) \qquad (4.7)$$

Con los respectivos conjuntos de parámetros, $W_{\tilde{F}}$, se construyen los correspondientes modelos de identificación en paralelo de la forma dada por la ecuación 3.16:

$$\tilde{Te}(k+1) = \tilde{Te}(k) + \tilde{F}(\tilde{Te}(k-1), \tilde{Te}(k-2), u(k-3), \ldots, u(k-23),$$
$$Tm(k), Tm(k-1), Tc(k), Ta(k), W_{\tilde{F}}) \qquad (4.8)$$

$$\tilde{Te}(k+1) = \tilde{Te}(k) + \tilde{F}(\tilde{Te}(k-1) - Tm(k-1), \tilde{Te}(k-2) - Tm(k-2),$$
$$u(k-5), \ldots, u(k-12), u(k-15), u(k-18), u(k-21), u(k-23), W_{\tilde{F}}) \qquad (4.9)$$

Los modelos de identificación en serie-paralelo (ecuaciones 4.6 y 4.7) se entrenan utilizando el algoritmo de retropropagación estático y con razón de aprendizaje variando de 0.1 a 0.0001. En la tabla 4.2 se indican los errores de identificación $E_m^I(W_{\tilde{F}})$ (ecuación 3.17) obtenidos para cada uno de los modelos después de 6000 ciclos de aprendizaje, aproximadamente. La tabla incluye también las arquitecturas de las redes, el número de parámetros ajustables de cada una de ellas y el error de predicción $E_m^p(W_{\tilde{F}})$ (ecuación 3.23) que ocasionan los modelos de identificación en paralelo construidos con los pesos $W_{\tilde{F}}$, (ecuaciones 4.8 y 4.9).

En la figura 4.10 se muestran los perfiles de la temperatura del fluido de intercambio de calor predichos por los respectivos modelos en paralelo (ecuaciones 4.8 y 4.9), es decir utilizando como única información la temperatura inicial del fluido de intercambio de calor y la variable de entrada al proceso. Se observa que las predicciones proporcionadas por los modelos en paralelo no son aproximaciones adecuadas. En el caso del modelo2, esta predicción podría considerarse aceptable, aunque nuestro objetivo es obtener mejores aproximaciones, ya que el control del proceso va a depender de estas representaciones.

	Arquitectura	Nº de parámetros	$E_m^I(W_{\tilde{F}})$	$E_m^p(W_{\tilde{F}})$
modelo1	27 - 28 - 1	812	$1.9\cdot10^{-2}$	359.84
modelo2	14 - 20 - 1	320	$1.5\cdot10^{-2}$	9.81

Tabla 4.2. Aproximación de F utilizando un perceptron multicapa: *Dato1*.

Aproximación de F utilizando la arquitectura de red parcialmente recurrente

Introduciendo, respectivamente, $I(k) = (u(k-3),\ldots,u(k-23),Tm(k),Tm(k-1),Tc(k),Ta(k))$ y $I(k) = (u(k-5),\ldots,u(k-12),u(k-15),u(k-18),u(k-21),u(k-23))$ como patrones de entrada, y considerando 2 neuronas de contexto, la red parcialmente recurrente descrita en el apartado 3.1.2 puede utilizarse para aproximar los funcionales F en los modelos NARMA dados por las ecuaciones 4.3 y 4.5, obteniendo los siguiente modelos:

$$\tilde{Te}_r(k+1) = \tilde{Te}_r(k) + \tilde{\tilde{F}}_r(u(k-3),\ldots,u(k-23),Tm(k),Tm(k-1),$$
$$Tc(k),Ta(k),R_1(k),R_2(k),W_{\tilde{\tilde{F}}_r}) \qquad (4.10)$$

$$\tilde{Te}_r(k+1) = \tilde{Te}_r(k) + \tilde{\tilde{F}}_r(u(k-5),\ldots,u(k-12),u(k-15),u(k-18),u(k-21),$$
$$u(k-23),R_1(k),R_2(k),W_{\tilde{\tilde{F}}_r}) \qquad (4.11)$$

siendo $R_1(k) = \tilde{Te}_r(k-1)$, $R_2(k) = \tilde{Te}_r(k-2)$ para el modelo dado por la ecuación 4.10 y $R_1(k) = \tilde{Te}_r(k-1) - Tm(k-1)$, $R_2(k) = \tilde{Te}_r(k-2) - Tm(k-2)$ para el modelo dado por la ecuación 4.11.

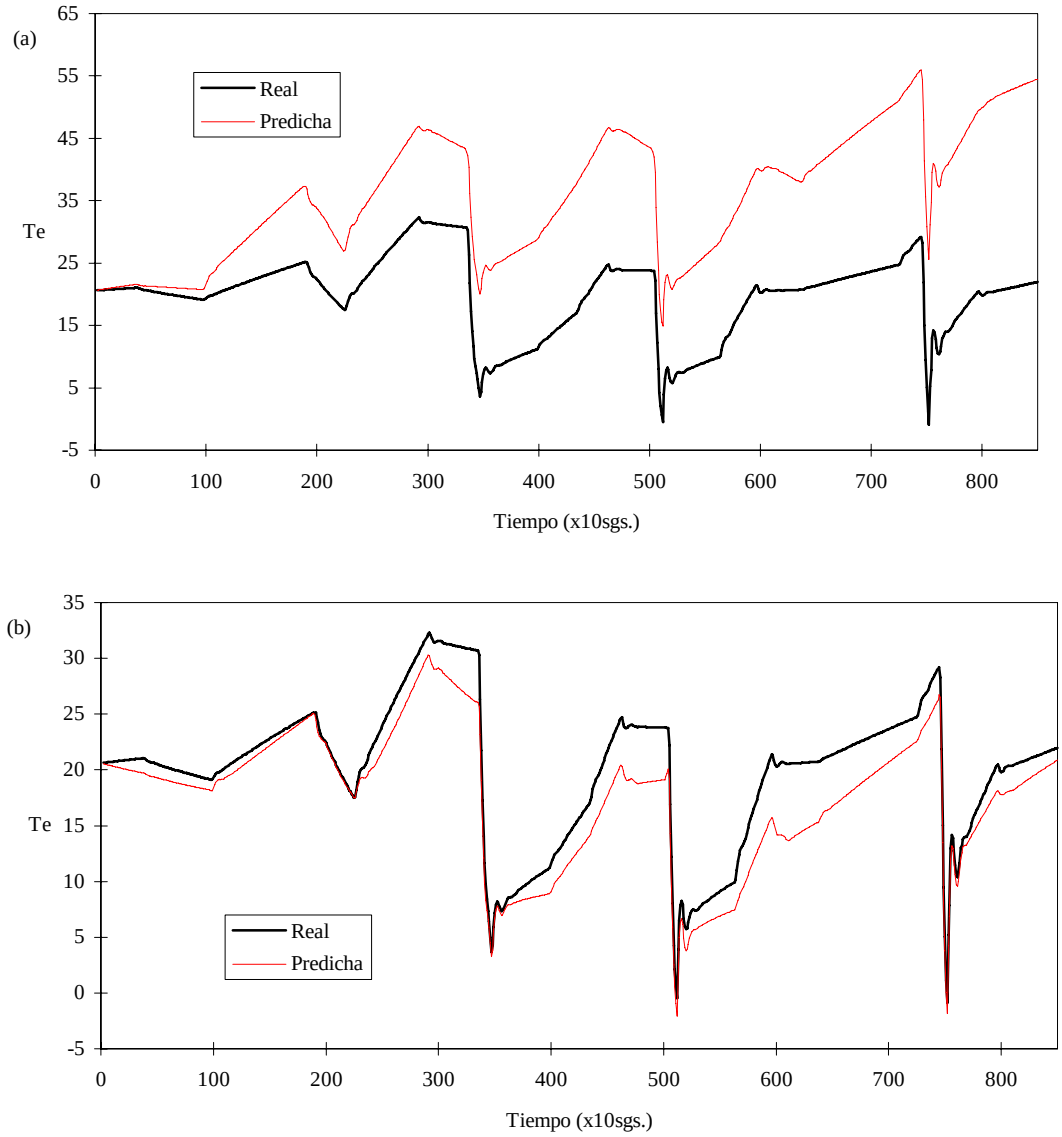


Figura 4.10. Perfil de la temperatura Te obtenida con el modelo en paralelo de pesos $W_{\tilde{F}}$ para *Dato1*. (a) modelo1 (b) modelo2.

Al realizar el aprendizaje de estas redes parcialmente recurrentes, se ha observado que es difícil conseguir la convergencia hacia un mínimo cuando los pesos se inicializan aleatoriamente. Se ha creído, por tanto, conveniente comenzar el entrenamiento con pesos que posean cierta información sobre los funcionales que se pretenden aproximar, de forma que se garantice el éxito del aprendizaje.

Las redes parcialmente recurrentes se inicializan, entonces, con los parámetros que se obtienen después de realizar 2000 ciclos de aprendizaje con los respectivos modelos

de identificación en serie-paralelo (ecuaciones 4.6 y 4.7). A continuación, se entrenan las redes parcialmente recurrentes utilizando el algoritmo de retropropagación dinámica descrito en el apartado 3.1.3 (ley de aprendizaje dada por la ecuación 3.12) y con razón de aprendizaje 0.00001. Nótese que, debido a que esta ley de adaptación incluye nuevos términos, la razón de aprendizaje elegida debe ser de magnitud inferior a cuando se utiliza una ley del tipo dada por la ecuación 3.13, de manera que puedan evitarse problemas de inestabilidad en el algoritmo de aprendizaje. En este caso, para realizar el entrenamiento no se ha recurrido a leyes de adaptación más simples (ecuación 3.13), ya que es suficiente considerar dos neuronas de contexto para que el problema esté bien representado y por tanto, la diferencia de esfuerzo computacional entre una ley y otra es casi indistinguible.

El número de ciclos de aprendizaje que se requiere para conseguir la convergencia de las redes parcialmente recurrentes y, por tanto, para alcanzar un mínimo del error de predicción (ecuación 3.23) es aproximadamente de 1500.

En la tabla 4.3 se indican los errores de predicción $E_m^p(W_{\tilde{F}_r})$ cometidos por los modelos en paralelo de pesos $W_{\tilde{F}_r}$ (ecuaciones 4.10 y 4.11); en la figura 4.11 se muestran los perfiles de la temperatura del fluido de intercambio de calor obtenidos con dichos modelos. Se observa que, cuando los funcionales F de los modelos NARMA dados por las ecuaciones 4.3 y 4.5 se aproximan utilizando la red parcialmente recurrente, es posible obtener una representación adecuada del proceso dinámico. Esto, y como ya se ha mencionado, es debido a que los parámetros del modelo se estiman con el propósito de simular la dinámica de dicho proceso.

	Nº de Parámetros	$E_m^p(W_{\tilde{F}_r})$
modelo1	812	3.25
modelo2	320	2.48

Tabla 4.3. Aproximación de F utilizando la red parcialmente recurrente: *Dato1*.

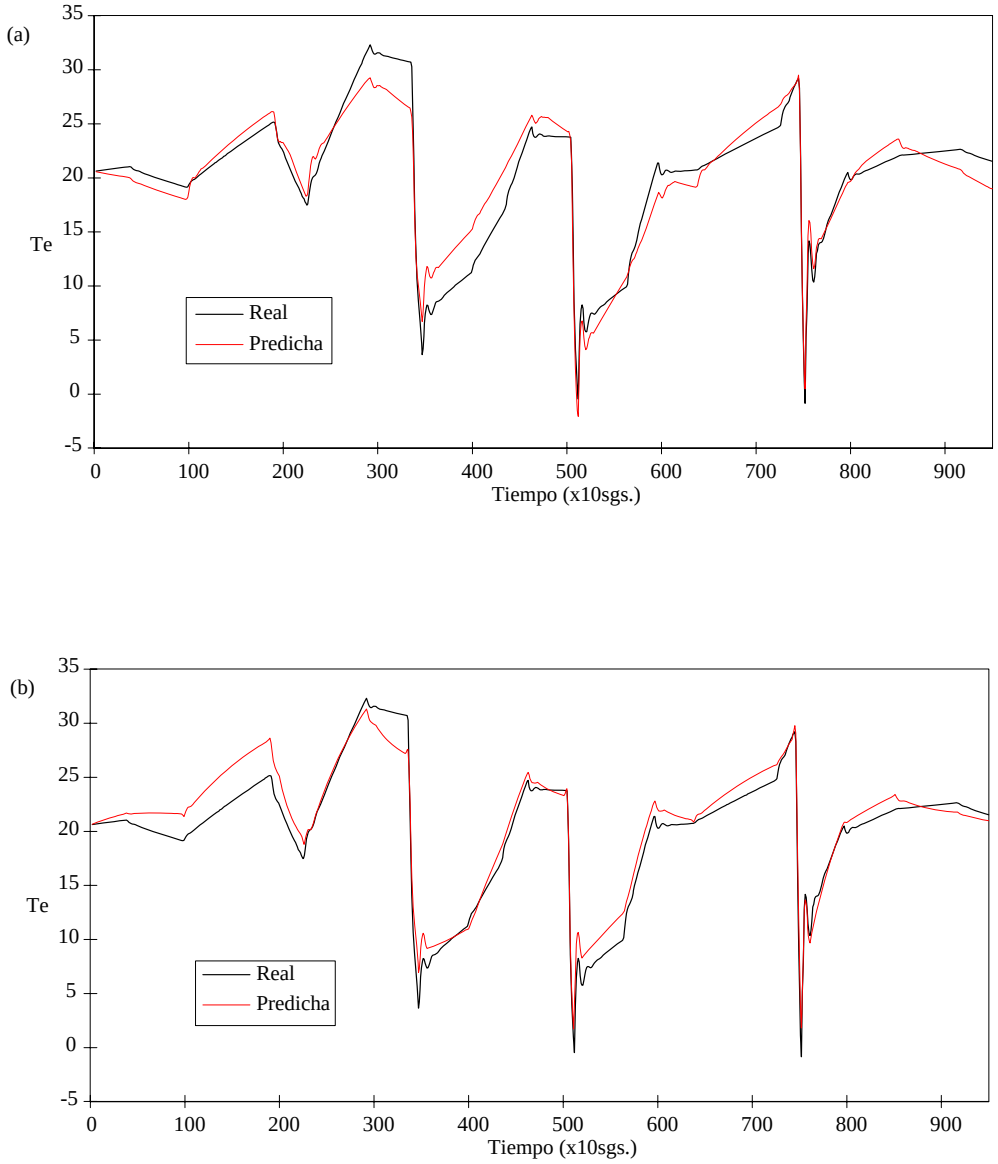


Figura 4.11. Perfil de la temperatura Te obtenida con el modelo en paralelo de pesos $W_{\tilde{F}_r}$ para *Dato1*. (a) modelo1 (b) modelo2.

Para poder analizar y observar las capacidades de generalización de los modelos NARMA en paralelo construidos con la red parcialmente recurrente (ecuaciones 4.10 y 4.11), se evalúa el error de predicción (ecuación 3.23) cometido por cada uno de ellos para los conjuntos *Dato2* y *Dato3,* datos que no han intervenido en la obtención de los parámetros de los modelos. Los errores obtenidos se indican en la tabla 4.4 y los perfiles de temperatura se muestran en las figuras 4.12 y 4.13, respectivamente.

	$E_m^p(W_{\tilde{F}_r})$: *Dato2*	$E_m^p(W_{\tilde{F}_r})$: *Dato3*
modelo1	2.86	685.4
modelo2	0.77	20.3

Tabla 4.4. Evaluación de la capacidad de generalización de los modelos NARMA neuronales en paralelo.

Los resultados obtenidos muestran que los modelos NARMA dados por las ecuaciones 4.10 y 4.11 proporcionan representaciones similares de la dinámica de la temperatura del fluido de intercambio de calor (tabla 4.3), por lo que se confirma que la información que proporcionan las cantidades m_i (ecuación 4.4) puede utilizarse para simplificar el modelo dado por la ecuación 4.3.

Es necesario señalar que siempre será más conveniente utilizar el modelo dado por la ecuación 4.11 (modelo2), ya que se trata de un modelo con un número de parámetros ajustables, lo cual implica un menor esfuerzo computacional. Esto es interesante, sobre todo, en aplicaciones en las que los parámetros del modelo tengan que adaptarse en tiempo real.

Por otra parte, como puede observarse en la tabla 4.4, las propiedades de generalización del modelo2 son significativamente mejores, aunque esto no es debido a que se trate de un modelo simplificado, sino a que dicho modelo utiliza las diferencias Te-Tm en lugar de Te y Tm, como lo hace el modelo1.

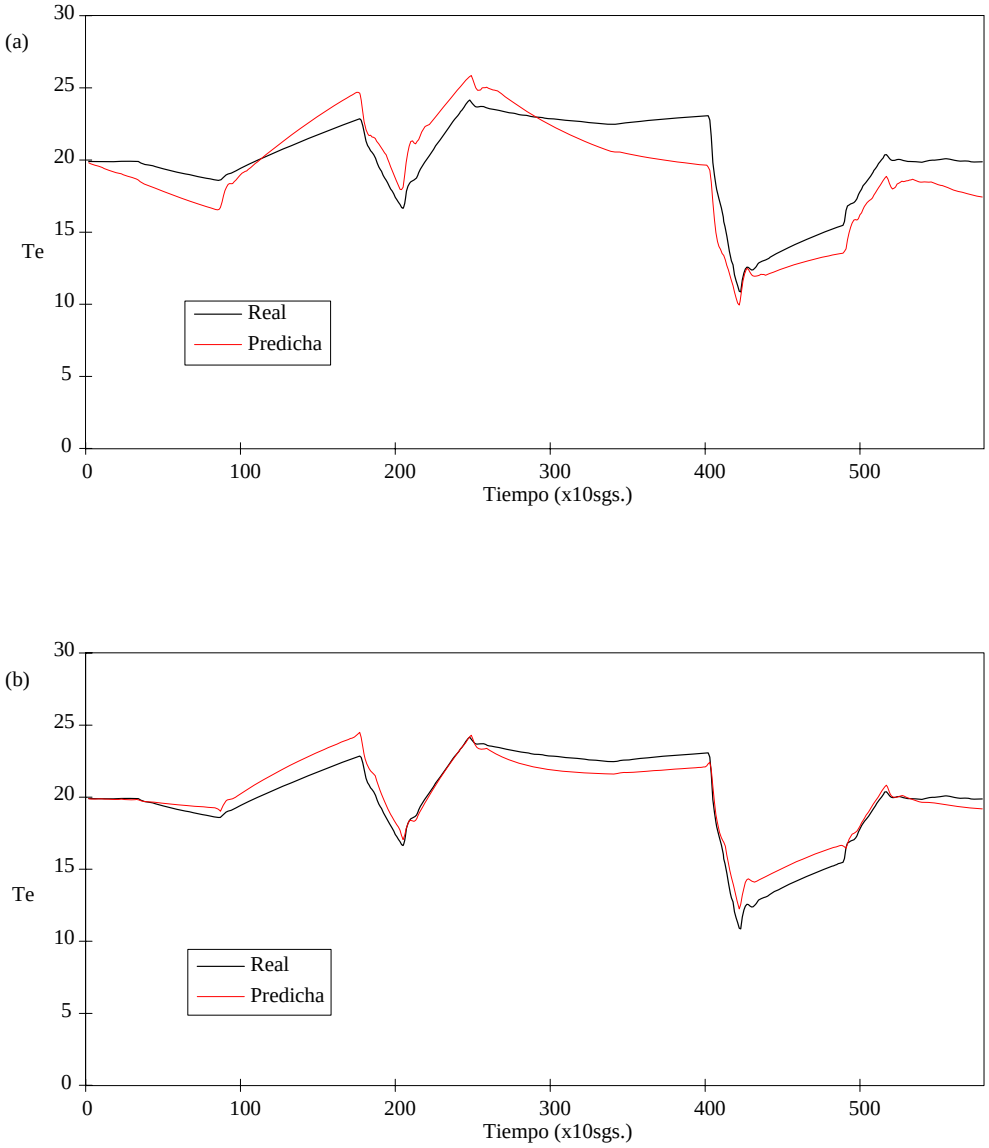


Figura 4.12. Perfil de la temperatura Te obtenida con el modelo en paralelo de pesos $W_{\tilde{F}_r}$ para *Dato2*. (a) modelo1 (b) modelo2.

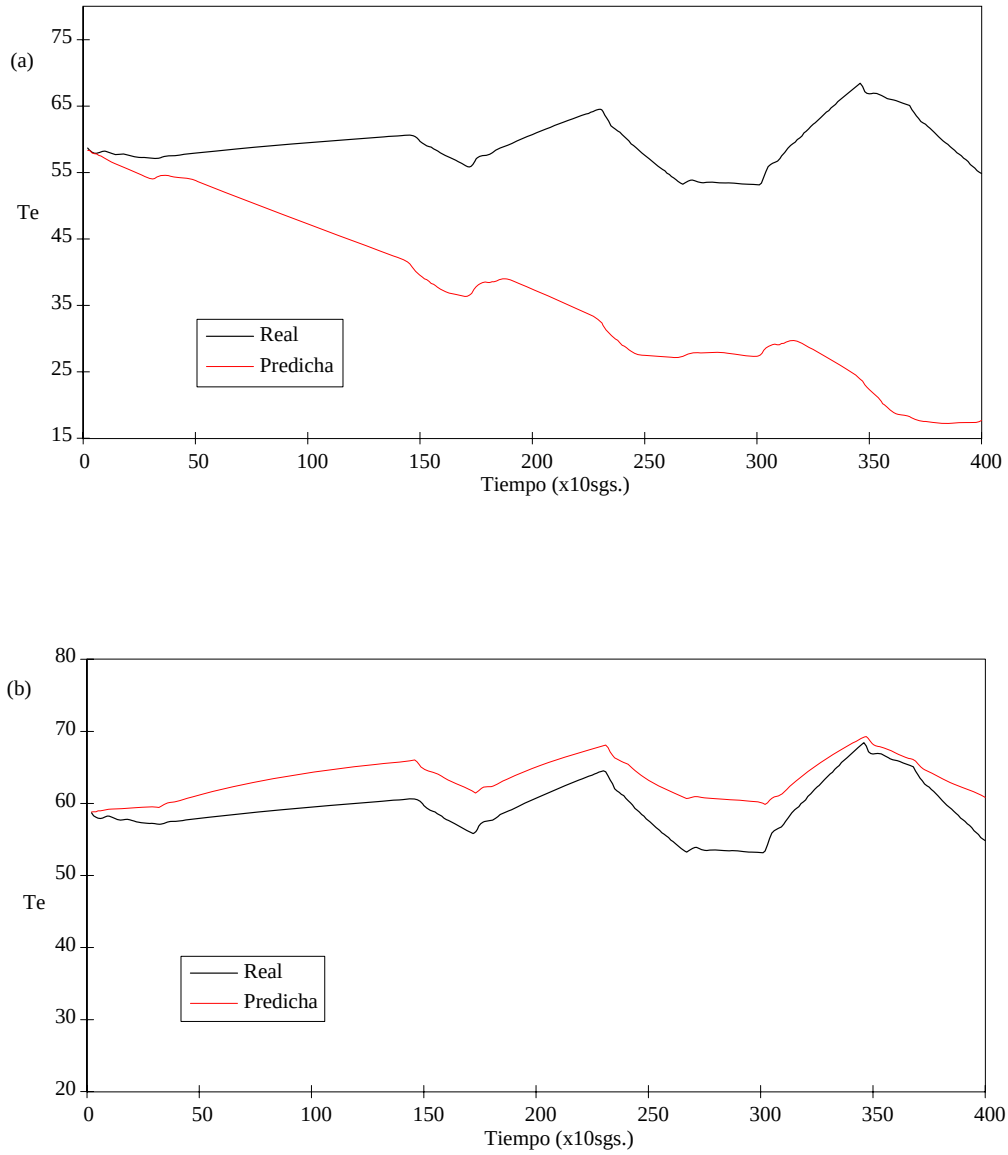


Figura 4.13. Perfil de la temperatura Te obtenida con el modelo en paralelo de pesos $W_{\tilde{F}_r}$ para *Dato3*. (a) modelo1 (b) modelo2.

4.2.2 Modelo basado en el balance de calor

En el apartado anterior se han mostrado los resultados obtenidos con los modelos NARMA en paralelo propuestos en esta tesis. Como se decía en la introducción del presente trabajo, para poder validar y comprobar la bondad de dichos resultados, se recurre a un modelo físico o modelo basado en el balance de calor que describe el comportamiento dinámico de los circuitos de calentamiento y enfriamiento (figura 4.1). Dicho modelo fue construido por expertos en Ingeniería Química y estaba disponible en el Centro de investigación donde se han llevado a cabo estos experimentos.

No es de interés en esta aplicación conocer los detalles de la construcción del modelo físico utilizado, ya que únicamente interesan conocer los perfiles de la temperatura del fluido de intercambio de calor que proporciona dicho modelo. Sin embargo, se cree conveniente escribir las ecuaciones diferenciales que constituyen el modelo, así como los parámetros que intervienen, de modo que queden expuestas las características más generales.[1]

Para la obtención del modelo físico, los circuitos de enfriamiento/calentamiento fueron divididos en diferentes subsistemas: el bucle principal, Te, el aislamiento de dicho bucle, Tp, y el depósito de frío, Tc. Este último subsistema fue dividido en 6 partes de idéntica capacidad térmica. La aplicación de los balances de energía para cada uno de los subsistemas produce el siguiente conjunto de un conjunto de ecuaciones diferenciales:

$$\frac{dTe}{dt} = \frac{1}{\Gamma_e}\left[q_{rxn} + q_h + q_p - Q_c \cdot Cp_e (Te - Tc_0)\right] + \frac{Tp - Te}{\tau_{e_1}} \tag{4.12}$$

$$\frac{dTp}{dt} = \frac{Te - Tp}{\tau_{e_2}} - \frac{q_L}{\tau_k} \tag{4.13}$$

$$\frac{dTc_i}{dt} = \frac{6}{\Gamma_c}\left[Q_c \cdot Cp_e (Te - Tc_i) + Q_k \cdot Cp_c (Ta - Tc_i)\right] \tag{4.14}$$

$$\frac{dTc_1}{dt} = \frac{6}{\Gamma_c} Q_T \cdot Cp_c (Tc_i - Tc_1) \tag{4.15}$$

$$\frac{dTc_2}{dt} = \frac{6}{\Gamma_c} Q_T \cdot Cp_c (Tc_1 - Tc_2) \tag{4.16}$$

[1] Una información más detallada puede encontrarse en (ZALD, 95).

$$\frac{dTc_3}{dt} = \frac{6}{\Gamma_c} Q_T \cdot Cp_c (Tc_2 - Tc_3) \tag{4.17}$$

$$\frac{dTc_4}{dt} = \frac{6}{\Gamma_c} Q_T \cdot Cp_c (Tc_3 - Tc_4) \tag{4.18}$$

$$\frac{dTc_0}{dt} = \frac{6}{\Gamma_c} Q_T \cdot Cp_c (Tc_4 - Tc_0) \tag{4.19}$$

$$\frac{dTm}{dt} = \frac{1}{\Gamma_m} \cdot q_{rxn} \tag{4.20}$$

siendo:

- $q_p(kW) = 1.38 \exp\left(\frac{352.6}{Te}\right)$
- $Q_e = 197\, kg/s.$
- $Q_k = 3.3\, kg/s.$
- $Q_T = Q_c + Q_k$
- $q_L(kW) = (0.455 - 9.56 \cdot \exp(-4\,Te)) \cdot (Te - Ta)$
- Q_c y q_h se calculan de acuerdo con la respuesta del sistema de control como se indica en las figuras 4.4 y 4.5.
- Los parámetros Γ y τ se optimizan utilizando los datos experimentales de *Dato1*. Dicha optimización está basada en un método del gradiente sujeto a restricciones, pues las constantes no pueden tomar valores negativos, en cuyo caso no tendrían sentido físico. Los valores obtenidos se indican en la tabla 4.5.

Parámetros	Valores
τ_{e1}	1156 (sgs.)
τ_{e2}	2504 (sgs.)
τ_k	500 (sgs.)
Γ_e	605.35 + 1.28T_e (KJ/K)
Γ_c	9191 (KJ/K)

Tabla 4.5. Valores de los parámetros Γ y τ del modelo físico.

En la figura 4.14 se muestran las temperaturas (Te, Tm y Tc) predichas por el modelo físico, para los conjuntos de datos experimentales *Dato1*, *Dato2*, *Dato3*. Se observa que el modelo no puede explicar las fluctuaciones existentes en la dinámica de

la

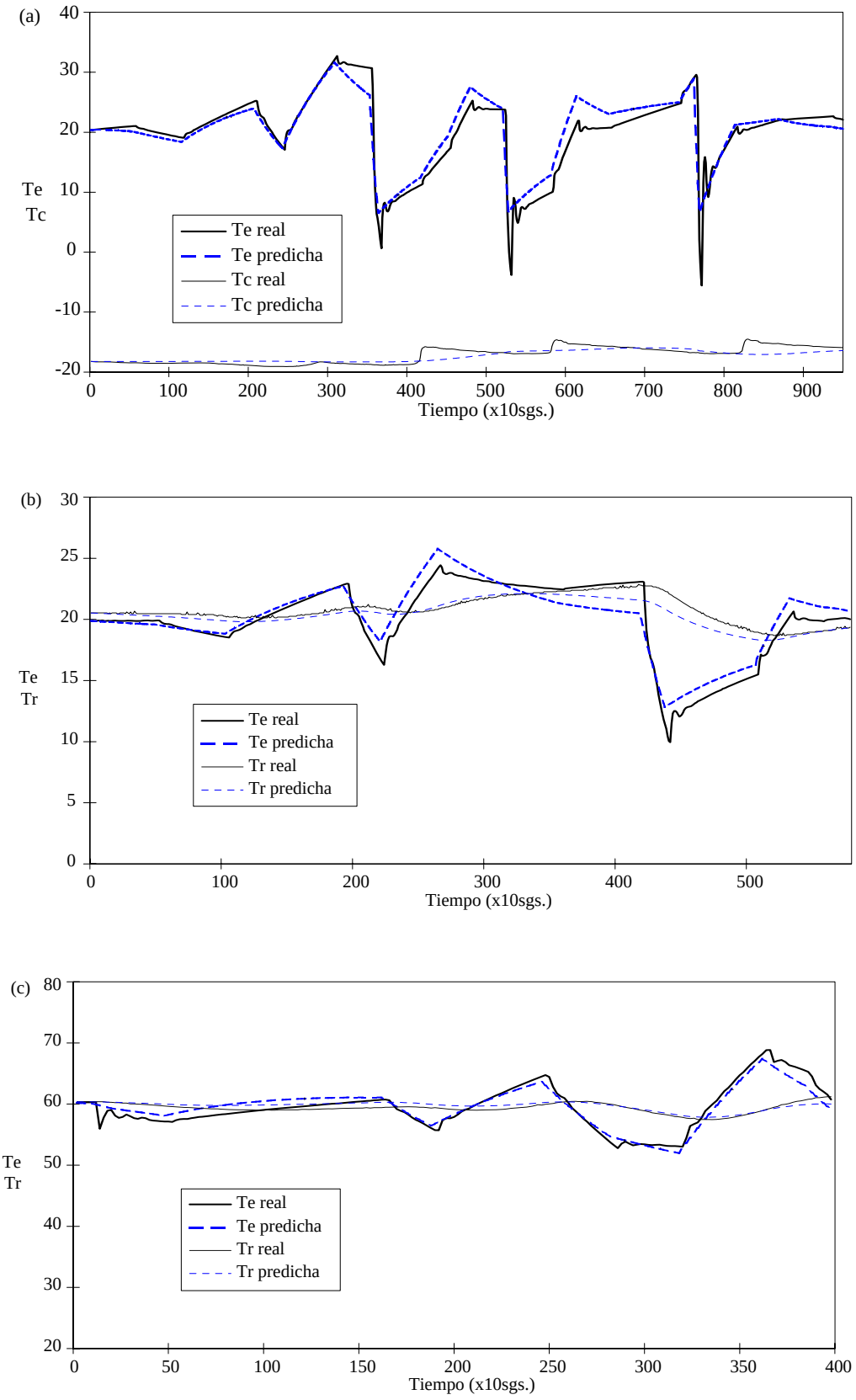


Figura 4.14. Perfiles de las temperaturas Te, Tm, Tc obtenidos con el modelo basado en el balance de calor. (a) *Dato1* (b) *Dato2* (c) *Dato3*.

temperatura fluido de intercambio de calor; ello es debido a que el flujo que circula en el bucle principal se ha considerado constante, Qe=197 kg/sg.

Con la finalidad de poder disponer de un criterio numérico para evaluar el concepto de "mejor representación del proceso" se calcula el error cuadrático definido por la ecuación 3.23 cometido por el modelo basado en el balance de calor. En este caso, $y(k+1)$ es la temperatura medida del fluido de intercambio de calor al instante de tiempo k+1 y $\tilde{y}_p(k+1)$ es el valor de dicha temperatura que resulta de integrar el modelo dado por las ecuaciones 4.12-4.20 utilizando un método de Runge-Kutta. Estos errores toman los siguientes valores:

Dato1: E_m^p=4.99 *Dato2*: E_m^p=1.08 *Dato3*: E_m^p=0.42

4.3 Control de la temperatura del fluido que circula en la camisa del reactor químico

El proceso dinámico que gobierna la evolución de la temperatura del fluido que circula en la camisa del reactor, es un proceso dinámico no lineal y por tanto se considera conveniente introducir técnicas de control no lineales con el objetivo de mejorar los resultados de control que proporciona el controlador PID actualmente incorporado en la planta.

En esta sección se presentan los resultados obtenidos con los sistemas de control inverso y predictivo descritos en la sección 3.3, los cuales ocupan la posición del controlador esclavo en la figura 4.3, regulando la variable de entrada u.

Los tests de control se realizan desde la segunda estación de trabajo de la instalación (figura 4.2), en la que se encuentran los programas que contienen los algoritmos de control neuronal. Cada programa está organizado para que envíe la acción o señal de control calculada por el controlador neuronal, u, y en el siguiente periodo de muestreo (10 segundos más tarde) reciba la respuesta del proceso (medidas reales de las variables que intervienen en dicho proceso).

Todos los experimentos de control se han realizado utilizando el proceso real y, por motivos de seguridad, dichos experimentos se han llevado a cabo con agua en el reactor, aproximadamente 70 u 80 Kg., y velocidad de agitación de 300 rpm.

Con la finalidad de observar el funcionamiento de los controladores neuronales, tanto para el calentamiento del fluido que circula en la camisa del reactor como para el enfriamiento, se consideran dos objetivos de control: partiendo de una temperatura del fluido de 15^{o}C, el primer objetivo es alcanzar la temperatura de 23^{o}C y estabilizarse cuando dicha temperatura se consiga; de este modo el objetivo de control se escribe como $Te^{des} = 23^{o}$ C. El segundo objetivo es alcanzar (y posteriormente estabilizarse) la temperatura de 15^{o}C cuando la temperatura inicial de fluido es de 23^{o}C, es decir $Te^{des} = 15^{o}$ C.

Los resultados que proporciona el controlador PID actualmente instalado para estos objetivos de control se muestran en la figura 4.15.

4.3.1 Control inverso neuronal

Al plantear un esquema de control inverso para controlar la temperatura del fluido que circula en la camisa del reactor, el primer objetivo era aproximar la inversa del proceso en la región de interés, de forma que el controlador aprendiera a realizar el objetivo de control dado por las necesidades de cada momento. Por tanto, el esquema de control que debe utilizarse es un esquema de control inverso con aprendizaje especializado (figura 1.7). Sin embargo, y con la finalidad de obtener un controlador eficiente del proceso en tiempo real, es necesario partir de una buena inicialización de los parámetros o pesos del controlador neuronal inverso. Para ello, se realizan las tres fases de aprendizaje descritas en el apartado 3.3.1.

A continuación se describe, en primer lugar, la estructura o ley de control inverso. Seguidamente, se llevan cabo las diferentes fases de aprendizaje, mostrándose los resultados de control obtenidos al finalizar cada una de ellas.

Estructura del controlador

Para determinar el número de entradas a la red de neuronas que actuará como controlador inverso del proceso, se siguen los pasos descritos en el apartado 3.3.1. Se supone que el proceso que gobierna el comportamiento de la temperatura del fluido de intercambio de calor, puede representarse por un modelo NARMA con la siguiente estructura:

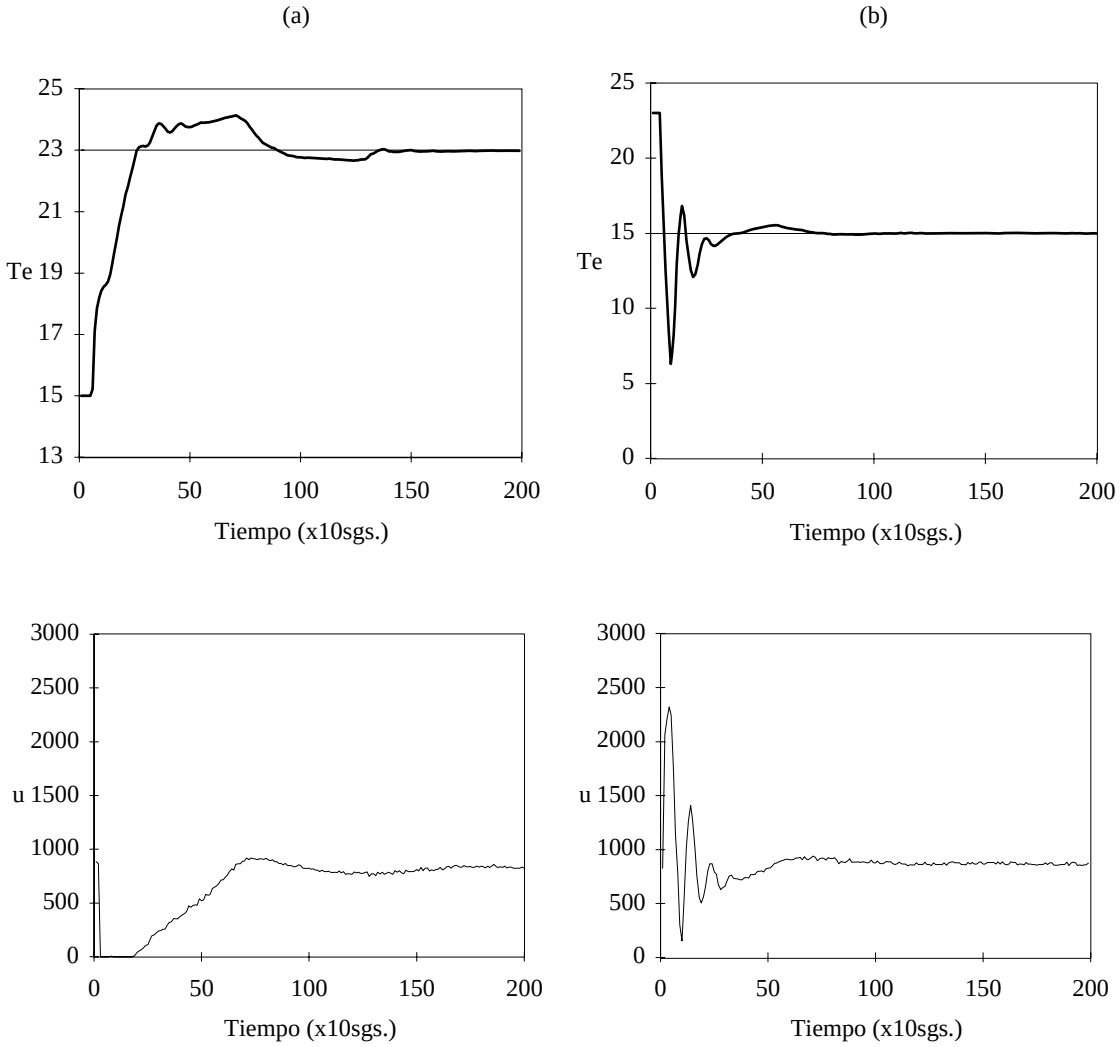


Figura 4.15. Controlador PID instalado en la planta.

Constantes: K=13%, T_i=1' 41" y T_d=0.

Parte superior: evolución de la temperatura Te en función del tiempo.

Parte inferior: evolución de la acción de control u en función del tiempo.

(a) $Te^{des} = 23^o$ C (b) $Te^{des} = 15^o$ C.

$$Te(k+1) = Te(k) + F(u(k-4), Te(k-1) - Tm(k-1)) \tag{4.21}$$

Este modelo no representa la dinámica del proceso en su totalidad, pero sí explica gran parte de dicha dinámica, ya que, según se vio en la sección anterior, las variables u y Te son las que más influyen en el comportamiento dinámico de la temperatura del fluido de intercambio de calor (figura 4.7). Nótese que se ha creído conveniente partir de un modelo simplificado, con vistas, principalmente, a obtener un controlador neuronal que no posea un gran número variables de entrada.

Realizando, entonces, un desarrollo similar al que se exponía en el apartado 3.3.1, y suponiendo que la temperatura Tm permanece constante en el intervalo de tiempo [k+3,k], es decir, $Tm(k+3) \approx Tm(k+2) \approx Tm(k+1) \approx Tm(k)$ (suposición que es cierta ya que dicha temperatura posee una dinámica muy lenta), se obtiene que la ley de control inverso adopta la siguiente forma:

$$u(k) = G(Te^{des} - Te(k), Te(k) - Tm(k), u(k-1), \ldots, u(k-4)) \tag{4.22}$$

siendo Te^{des} el objetivo de control o valor deseado para Te.

Introduciendo el vector $I(k) = (Te^{des} - Te(k), Te(k) - Tm(k))$ como patrón de entrada a la red y considerando 4 neuronas de contexto, la red parcialmente recurrente puede utilizarse para aproximar el funcional G, obteniendo una ley de control de la forma:

$$\tilde{u}_r(k) = \tilde{G}_r(Te^{des} - Te(k), Te(k) - Tm(k), R_1(k), \ldots, R_4(k), W_{\tilde{G}_r}) \tag{4.23}$$

siendo $R_i(k) = \tilde{u}_r(k-i)$ para $i = 1,2,3,4$ y $W_{\tilde{G}_r}$ los parámetros de la red.

Las funciones de activaciones de dicha red son, al igual que las utilizadas para modelizar, funciones sigmoidales cuya imagen está en el intervalo [0,1] y la topología, elegida después de varios intentos, es una red formada por una única capa oculta de 10 neuronas.

Aprendizaje del controlador inverso

1ª fase: Aprendizaje generalizado

Puesto que el propósito al realizar el aprendizaje generalizado no es aproximar la inversa global del proceso dinámico, sino obtener una inicialización adecuada de los parámetros $W_{\tilde{G}_r}$ que determinan la ley de control inverso (ecuación 4.23), se decide realizar el aprendizaje generalizado del controlador utilizando los datos disponibles en *Dato2* (tabla 4.1). Las acciones de control representadas en dicho conjunto cubren sólo determinadas regiones del espacio de entrada, por lo que, probablemente, el controlador obtenido después de haber realizado el aprendizaje generalizado con dicho conjunto de datos, no será un controlador eficiente y universal (en el sentido de que pueda controlar de manera eficiente todas las situaciones). Sin embargo, sí poseerá la suficiente información como para evitar situaciones de inestabilidad y "no control" del proceso.

Cuando se lleva a cabo el aprendizaje generalizado del controlador (figura 3.4.), no está definido aún ningún objetivo de control. Como consecuencia, el valor Te^{des} que aparece en la ecuación 4.23 se sustituye por $Te(k+5)$, de manera que, en esta fase de aprendizaje la acción de control se aproxima utilizando como patrón de entrada a la red el vector $I(k) = (Te(k+5) - Te(k), Te(k) - Tm(k))$

Los patrones de entrada a la red y las salidas deseadas se normalizan en el intervalo [0,1] y se presentan de forma ordenada, siguiendo la evolución del tiempo.

Al igual que ocurría cuando se trataba la modelización del proceso dinámico utilizando la red parcialmente recurrente (apartado 4.2.1), el entrenamiento de la red puede llegar a ser laborioso cuando sus pesos se inicializan aleatoriamente. Conseguir la convergencia hacia un mínimo de la función error dada por 3.36, requiere normalmente de un gran número de iteraciones, sin poder garantizar el éxito en el aprendizaje. Se cree, por tanto, conveniente partir de pesos que poseen ya cierta información. Para ello, se entrena primeramente el perceptron multicapa equivalente a la red parcialmente recurrente (arquitectura 6-10-1) y cuyos patrones de entrada son los vectores $I(k) = (Te(k+5) - - Te(k), Te(k) - Tm(k), u(k-1), \ldots, u(k-4))$.

El aprendizaje del preceptron se lleva a cabo durante 10000 ciclos de aprendizaje utilizando el algoritmo de retropropagación estático y con razón de aprendizaje variando de 0.1 a 0.01. Al inicializar la red parcialmente recurrente con los pesos obtenidos anteriormente, el error cuadrático definido por la ecuación 3.40 toma el valor

$E_c^g = 1.7 \cdot 10^{-3}$ (nótese que, las acciones de control reales u(k) y las aproximadas $\tilde{u}_r(k)$, están normalizadas en el intervalo [0,1]); la aproximación de la señal u se muestra en la figura 4.16.

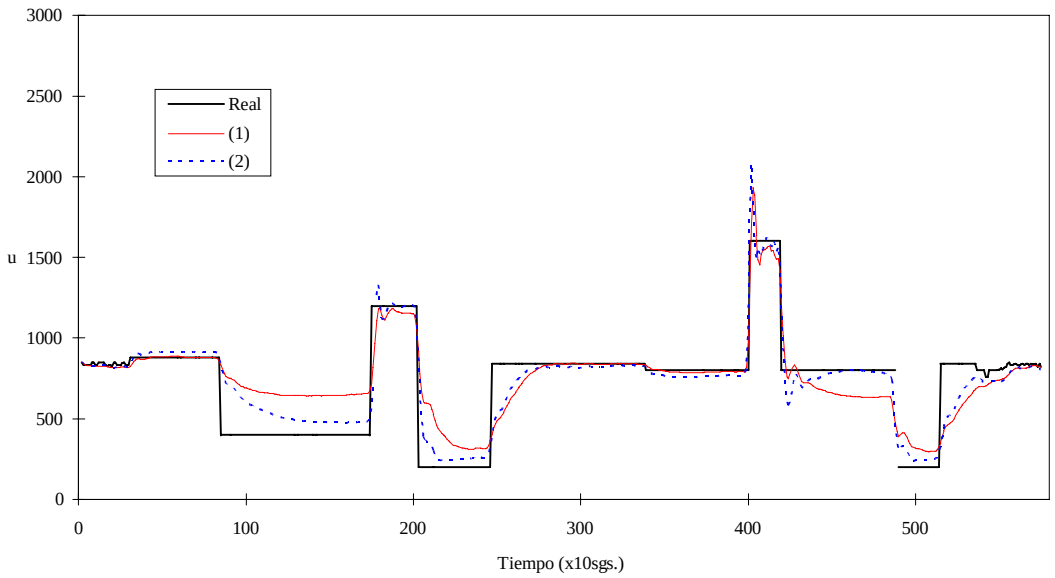


Figura 4.16. Aproximación de la dinámica inversa para *Dato2*.
(1) Predicción utilizando el perceptron multicapa (6-10-1)
(2) Predicción utilizando la red parcialmente recurrente (6-10-1, $n_1 = 2, n_r = 4$)

Una vez inicializada la red parcialmente recurrente, se procede a su entrenamiento utilizando la ley de adaptación dada por la ecuación 3.13 y razón de aprendizaje 0.001. En este caso, y a diferencia de cuando se trataba la modelización, es conveniente recurrir a una ley de adaptación más simple (ecuación 3.13) que aquella dada por el algoritmo de retropropagación dinámica (ecuación 3.12), ya que, son necesarias 4 neuronas de contexto para aproximar la dinámica inversa, lo cual incrementa de forma significativa el esfuerzo computacional. El error cuadrático obtenido después de 3000 ciclos de aprendizaje toma el valor $E_c^g = 8.8 \cdot 10^{-4}$ y la aproximación de la acción de control se muestra en la figura 4.16.

Comparando los resultados obtenidos, se observa que cuando se aproxima la inversa del proceso, al igual que cuando se aproximaba la dinámica del proceso, es conveniente realizar también la fase de aprendizaje utilizando la red parcialmente

recurrente, ya que se obtiene una mejor aproximación de dicha inversa y, como consecuencia, un mejor controlador.

A pesar de que el aprendizaje generalizado se lleva a cabo con la finalidad de obtener pesos iniciales para el controlador inverso definitivo del proceso real, se decide observar el comportamiento de la ley de control dada por la ecuación 4.23 con parámetros $W_{\tilde{G}_r}$ fijos, los obtenidos en esta fase de aprendizaje, parámetros que denotaremos en este contexto como $W_{\tilde{G}_r}$ (generalizado). En la figura 4.17 se muestra la evolución de la temperatura del fluido de intercambio de calor y la evolución de la acción de control calculada por el controlador neuronal, para cada uno de los objetivos de control considerados.

Cuando la red de neuronas ya entrenada realiza el control del proceso real, la variable $Te(k+5)$ del patrón de entrada se sustituye por el objetivo de control deseado, Te^{des}. Puesto que este valor permanece constante, los patrones de entrada al controlador mientras actúa en tiempo real difieren de aquellos con los que se realizó el aprendizaje generalizado, ya que la dinámica natural del proceso no permite alcanzar el valor Te^{des} en un sólo periodo de tiempo. Por tanto, una buena respuesta por parte del controlador neuronal no puede garantizarse. Si en lugar de considerar un objetivo de control fijo, se considera variable en el tiempo, es posible que el controlador neuronal proporcione un mejor control del proceso ya que, en este caso, los patrones de entrada del controlador no serían tan diferentes de aquellos con los que se realizó el aprendizaje.

En la figura 4.18 se muestran los resultados obtenidos con el mismo controlador de parámetros fijos, $W_{\tilde{G}_r}$ (generalizado), pero considerando un objetivo de control variable en el tiempo de la siguiente forma:

$$Te^{des}(k) = Te(k) + \frac{1}{2} \cdot (Te^{des} - Te(k)) \qquad (4.24)$$

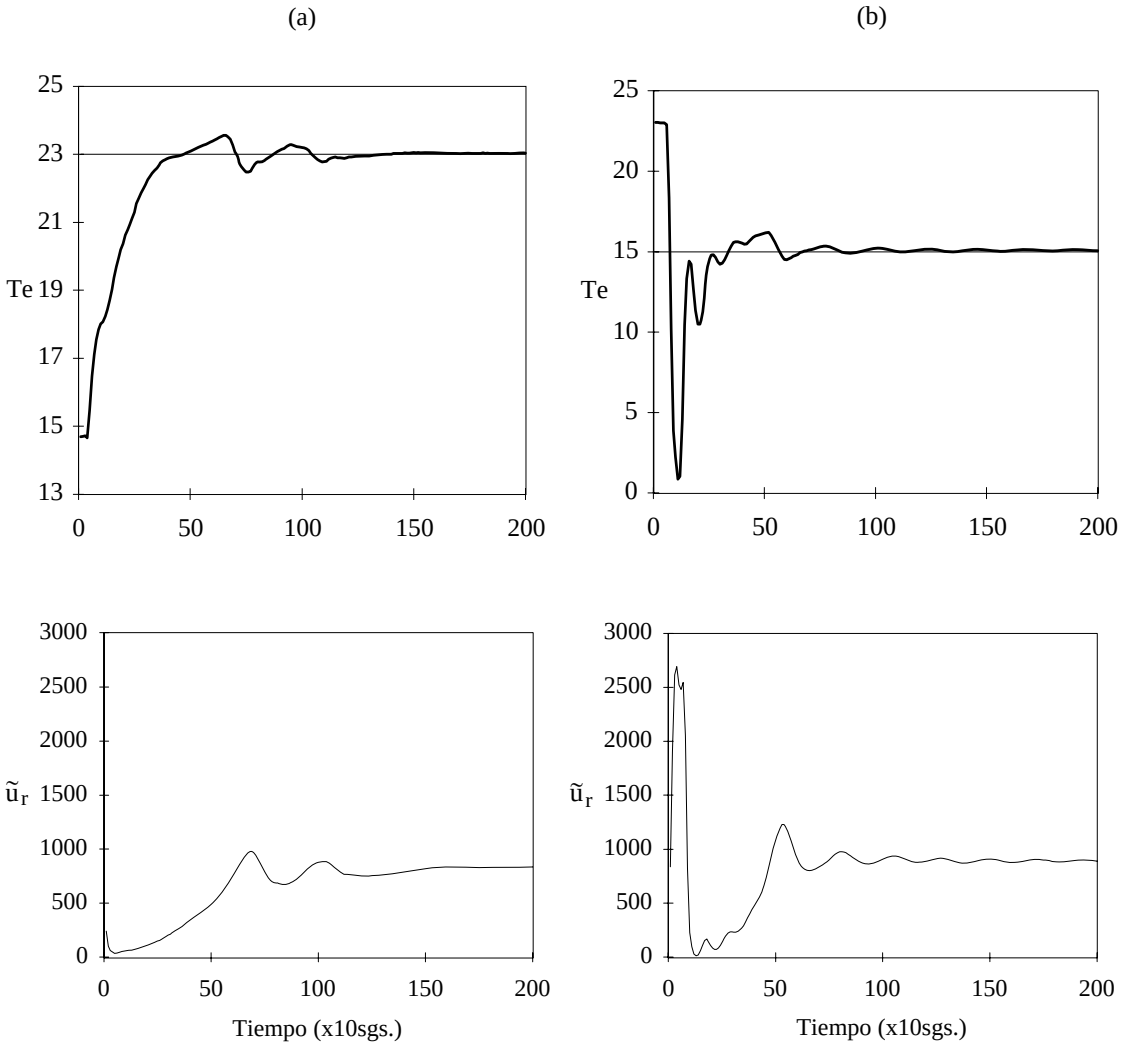


Figura 4.17. Controlador neuronal inverso con pesos $W_{\tilde{G}_r}$ (generalizado) fijos.

Parte superior: evolución de la temperatura Te en función del tiempo

Parte inferior: evolución de la acción de control $\tilde{u}_r$ en función del tiempo

(a) $Te^{des} = 23^o$ C (b) $Te^{des} = 15^o$ C.

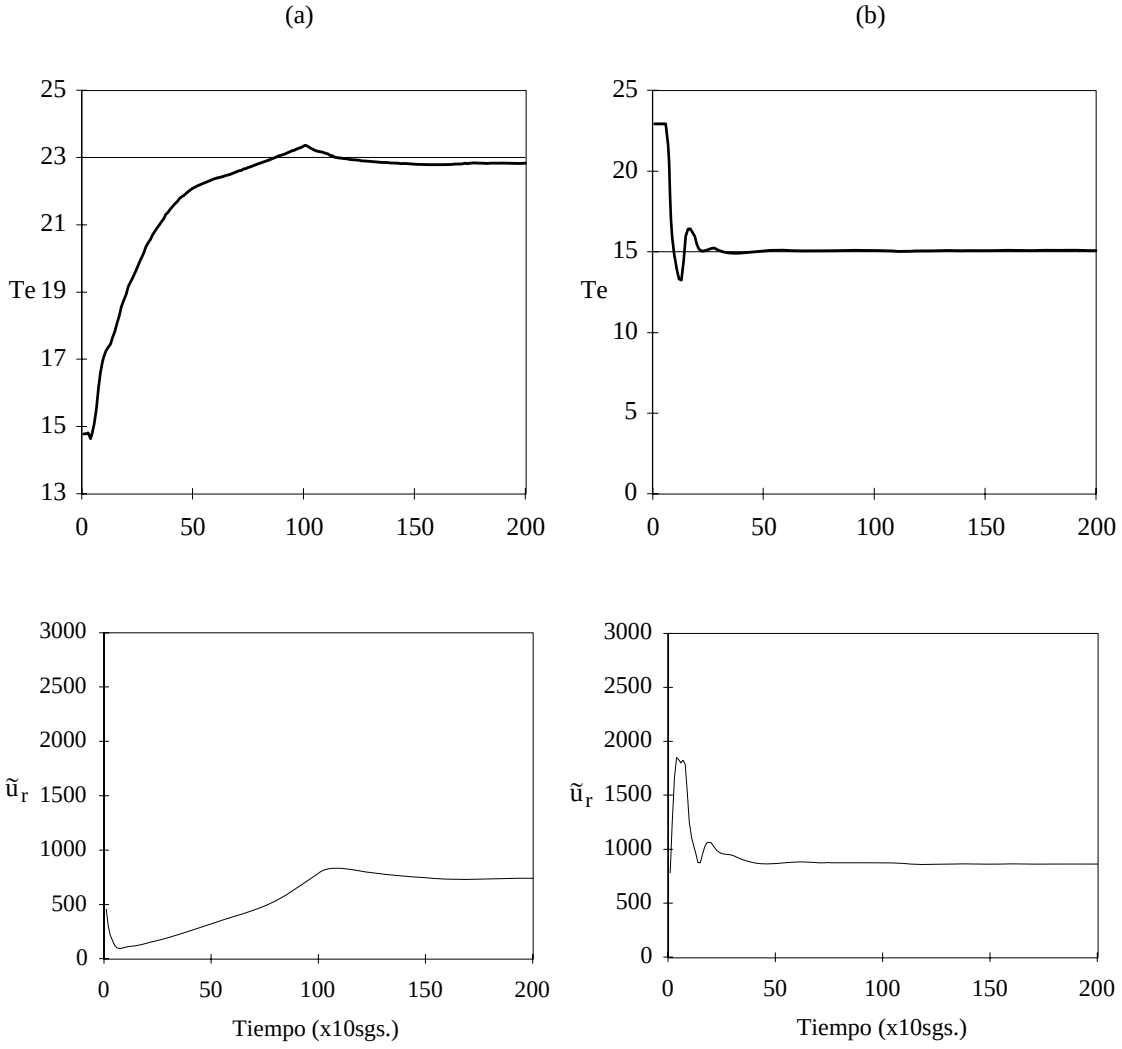


Figura 4.18. Controlador neuronal inverso con pesos $W_{\tilde{G}_r}$ (generalizado) fijos.

Objetivo de control variable.

Parte superior: evolución de la temperatura Te en función del tiempo.

Parte inferior: evolución de la acción de control $\tilde{u}_r$ en función del tiempo.

(a) $Te^{des} = 23^o$ C (b) $Te^{des} = 15^o$ C.

Se observa que los resultados de control son mejores, lo cual es debido a que los patrones de entrada a la red que actúa como controlador están mejor representados en el conjunto de datos con los que se ha realizado el aprendizaje. Aunque es necesario señalar también, que utilizar objetivos de control variables hace que el control del proceso sea generalmente más lento, en el sentido de que se necesita más tiempo para que el proceso alcance el valor deseado.

2ª fase: Aprendizaje especializado utilizando un modelo del proceso

Para poder mejorar los resultados obtenidos en la fase anterior para el control de la temperatura del fluido que circula en la camisa del reactor cuando el objetivo de control permanece fijo (figura 4.17), es necesario realizar un aprendizaje especializado del controlador, de manera que dicho controlador aprenda la inversa en la región de interés y dada por las necesidades de cada momento. Para ello, se realiza el aprendizaje especializado del controlador utilizando un modelo del proceso real (figura 3.5).

El modelo utilizado en esta fase de aprendizaje es un modelo NARMA neuronal en paralelo, concretamente el modelo dado por la ecuación 4.11.

El aprendizaje especializado utilizando un modelo del proceso se lleva a cabo para alcanzar el objetivo de control $Te^{des} = 23^{o}\,C$ en el intervalo de tiempo [0,3000] y el objetivo $Te^{des} = 15^{o}\,C$ en el intervalo [3001, 6000] (tiempo en segundos).

Inicializando la ley de control inverso dada por le ecuación 4.23 con el conjunto de pesos obtenido en la fase anterior, $W_{\tilde{G}_r}$ (generalizado), se procede al aprendizaje especializado, mediante el cual los parámetros del controlador se adaptan en cada instante de tiempo k+1 utilizando la ley dada por la ecuación 3.40. Después de realizar 20 ciclos de aprendizaje sobre el intervalo de tiempo [0,6000] y con razón de aprendizaje 0.01, los pesos obtenidos, que nombramos como $W_{\tilde{G}_r}$ (especializado_ m), se almacenan y se utilizan directamente para controlar el proceso real. Los resultados que se obtienen se muestran en la figura 4.19. Se observa que el comportamiento del controlador ha mejorado significativamente.

3ª fase: Aprendizaje especializado utilizando el proceso real

Finalmente y con la finalidad de corregir los errores en los resultados de control que han sido provocados por el uso de un modelo no exacto en la fase anterior, se realiza la

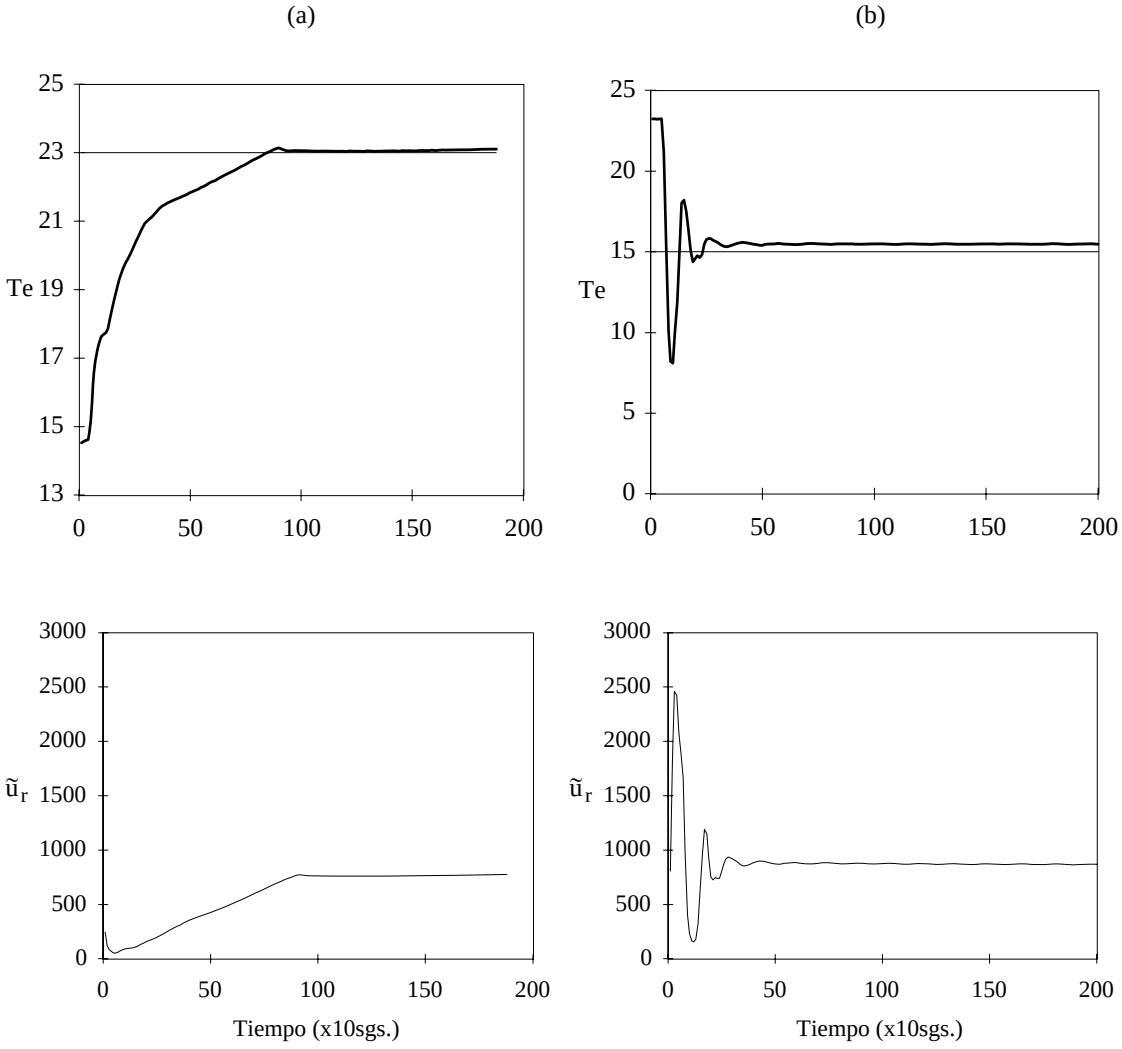


Figura 4.19. Controlador neuronal inverso con pesos $W_{\tilde{G}_r}$ (especializado_ m) fijos.

Parte superior: evolución de la temperatura Te en función del tiempo.

Parte inferior: evolución de la acción de control $\tilde{u}_r$ en función del tiempo.

(a) $Te^{des} = 23^{o}$ C (b) $Te^{des} = 15^{o}$ C.

tercera fase de aprendizaje (figura 3.6), en la que los pesos del controlador neuronal se adaptan en tiempo real.

Inicializando la ley de control inverso dada por la ecuación 4.23 con los parámetros obtenidos en la fase anterior, $W_{\tilde{G}_r}(\text{especializado\_m})$, se empieza a controlar el proceso real, pero, en este caso, dichos parámetros no se mantienen fijos, como para los resultados mostrados en la figura 4.19, sino que en cada periodo de muestro se ajustan utilizando la ley de adaptación dada por la ecuación 3.43 y con razón de aprendizaje 0.01. La evolución de la temperatura del fluido de intercambio de calor y la evolución de la acción de control se muestran en la figura 4.20.

El modelo que se utiliza para llevar a cabo esta fase de aprendizaje es el modelo NARMA en paralelo dado por la ecuación 4.11. En los experimentos realizados se ha observado que siempre es conveniente adaptar los parámetros de dicho modelo en tiempo real (identificación on-line), con vistas a obtener una aproximación más exacta del Jacobiano del proceso real en la región de interés.

4.3.2 Control predictivo neuronal

Inspirados en la idea de que una acción de control en un determinado instante, influye no sólo en la respuesta del proceso en el próximo instante de tiempo, sino también en las respuestas futuras de dicho proceso, se decide plantear una estrategia de control predictivo para controlar la temperatura del fluido que circula en la camisa del reactor, con la finalidad de mejorar los resultados obtenidos con los sistemas de control inverso.

A continuación se muestran los resultados de control que se obtienen cuando se utiliza la estrategia de control predictivo descrita en el apartado 3.3.2.

Estructura del controlador

Realizando un desarrollo similar al que se presenta en el apartado 3.3.2 y suponiendo que el proceso dinámico puede representare por el modelo NARMA dado por la ecuación 4.21, la acción de control predictivo se puede expresar de la forma:

$$u(k) = P(Te^{des} - Te(k), Te(k) - Tm(k), u(k-1), \ldots, u(k-4)) \qquad (4.25)$$

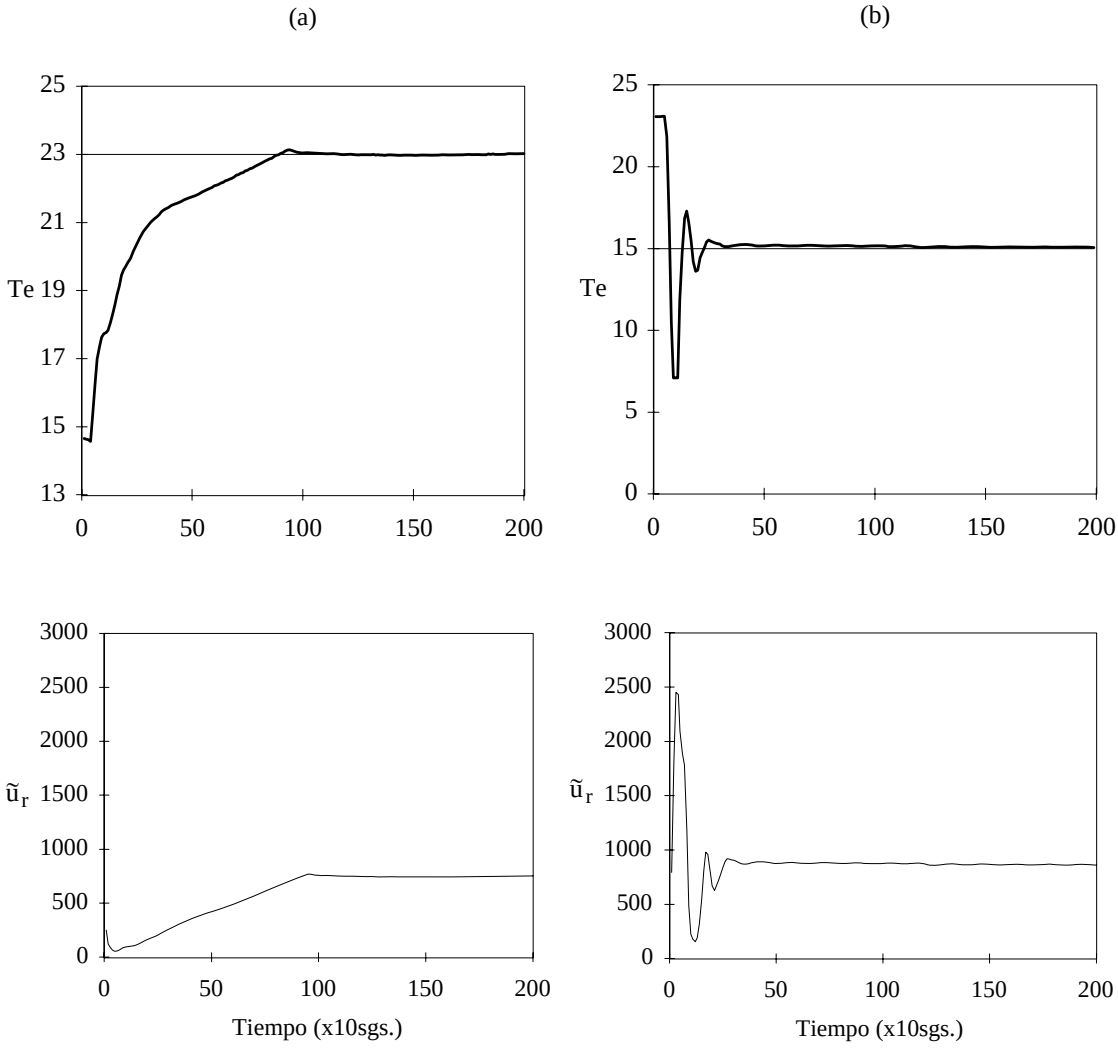


Figura 4.20. Controlador neuronal inverso con pesos adaptándose en tiempo real e inicializados con $W_{\tilde{G}_r}$ (especializado_ m).

Parte superior: evolución de la temperatura Te en función del tiempo.

Parte inferior: evolución de la acción de control $\tilde{u}_r$ en función del tiempo.

(a) $Te^{des} = 23^o$ C (b) $Te^{des} = 15^o$ C.

Al igual que en el apartado anterior, la red parcialmente recurrente puede utilizarse para aproximar el funcional P, basta introducir $I(k) = (Te^{des} - Te(k), Te(k) - Tm(k))$ como patrón de entrada a la red y considerar 4 neuronas de contexto:

$$\tilde{u}_r(k) = \tilde{P}_r(Te^{des} - Te(k), Te(k) - Tm(k), R_1(k), \ldots, R_4(k), W_{\tilde{P}_r}) \quad (4.26)$$

siendo $R_i(k) = \tilde{u}_r(k-i)$ para $i = 1,2,3,4$ y $W_{\tilde{P}_r}$ el conjunto de parámetros de la red.

Aprendizaje del controlador predictivo

En este caso, y puesto que el objetivo es controlar el proceso real, se ha creído también conveniente inicializar los parámetros de la ley de control dada por la ecuación 4.26. Esta ley de control posee la misma estructura que la ley de control inverso (ecuación 4.23), por lo que se decide inicializar sus parámetros utilizando el conjunto de pesos $W_{\tilde{G}_r}$ (generalizado), obtenidos en el apartado anterior. Estos parámetros no provocarán una estrategia de control predictivo, pero sí se sabe que evitarán situaciones de inestabilidad o de "no del control" del proceso.

Debido a la inicialización elegida, el controlador neuronal predictivo tiene que poseer la misma estructura que el controlador inverso, es decir una única capa oculta de 10 neuronas; las funciones de activación de las neuronas de la red siguen siendo las funciones sigmoidales con imagen en el intervalo [0,1].

Después de realizar algunas pruebas utilizando un modelo del proceso e incluso el proceso real, se observa que se puede elegir un horizonte de predicción de 40 periodos de muestreo (equivalente aproximadamente a 400 segundos) para llevar a cabo un control predictivo de la temperatura del fluido que circula en la camisa del reactor, en el modo de calentamiento.

Inicializando los parámetros de la ley de control dada por la ecuación 4.26 con el conjunto $W_{\tilde{G}_r}$ (generalizado), se empieza a controlar el proceso real para el objetivo de control $Te^{des} = 23^o C$ y utilizando una estrategia de control predictivo (figura 3.7). Se siguen los pasos descritos en el apartado 3.3.2; en cada periodo de muestreo y después de conocer el comportamiento del proceso a lo largo del horizonte de predicción (H=40), comportamiento que lo proporciona el modelo NARMA neuronal en paralelo

dado por la ecuación 4.11, los parámetros del controlador se adaptan de acuerdo con la ley dada por la ecuación 3.54 y razón de aprendizaje 0.0005. Los resultados de control obtenidos se muestran en la figura 4.21(a).

El horizonte de predicción elegido, H=40, no resulta tan adecuado cuando se realiza el enfriamiento del fluido de intercambio de calor, es decir, cuando el objetivo de control es $Te^{des} = 15^{o}\,C$. En este caso, se ha visto la necesidad de realizar un previo aprendizaje del controlador neuronal predictivo sustituyendo el proceso real por un modelo del proceso, con la finalidad de presentar varias veces el objetivo de control al controlador predictivo. Dicho aprendizaje se ha llevado a cabo durante 10 iteraciones (ciclos de aprendizaje) sobre el intervalo de tiempo [0,3000]. Con los pesos obtenidos se inicializa la ley de control dada por la ecuación 4.26 y se procede al control del proceso en tiempo real, utilizando la ley de adaptación dada por la ecuación 3.54 y razón de aprendizaje 0.0005. En la figura 4.21(b) se muestra la evolución de la temperatura y de la señal de control calculada por el controlador predictivo neuronal.

Con el objetivo de obtener un controlador que realice el enfriamiento del fluido de intercambio de calor, sin que la temperatura sufra bruscas oscilaciones, como sucede para el calentamiento, se decide tomar un horizonte de predicción superior. Para un horizonte de predicción de 70 periodos de muestreo (equivalente a 700 segundos), se obtiene el resultado que se muestra en la figura 4.22. Nótese que, en este caso, no ha sido necesario realizar un previo aprendizaje del controlador, sino que sus pesos se han inicializado con el conjunto $W_{\tilde{G}_r}$ (generalizado), al igual que para el resultado que se muestra en la figura 4.21(a).

4.4 Análisis de los resultados

- En primer lugar, es necesario señalar que los modelos NARMA neuronales son una posible alternativa a los modelos físicos, como se deduce de los resultados presentados en la sección 4.2. En las figuras 4.11 y 4.14(a) se observa que los modelos neuronales, siempre que sus parámetros sean estimados adecuadamente, pueden llegar a proporcionar aproximaciones de la temperatura del fluido de intercambio de calor

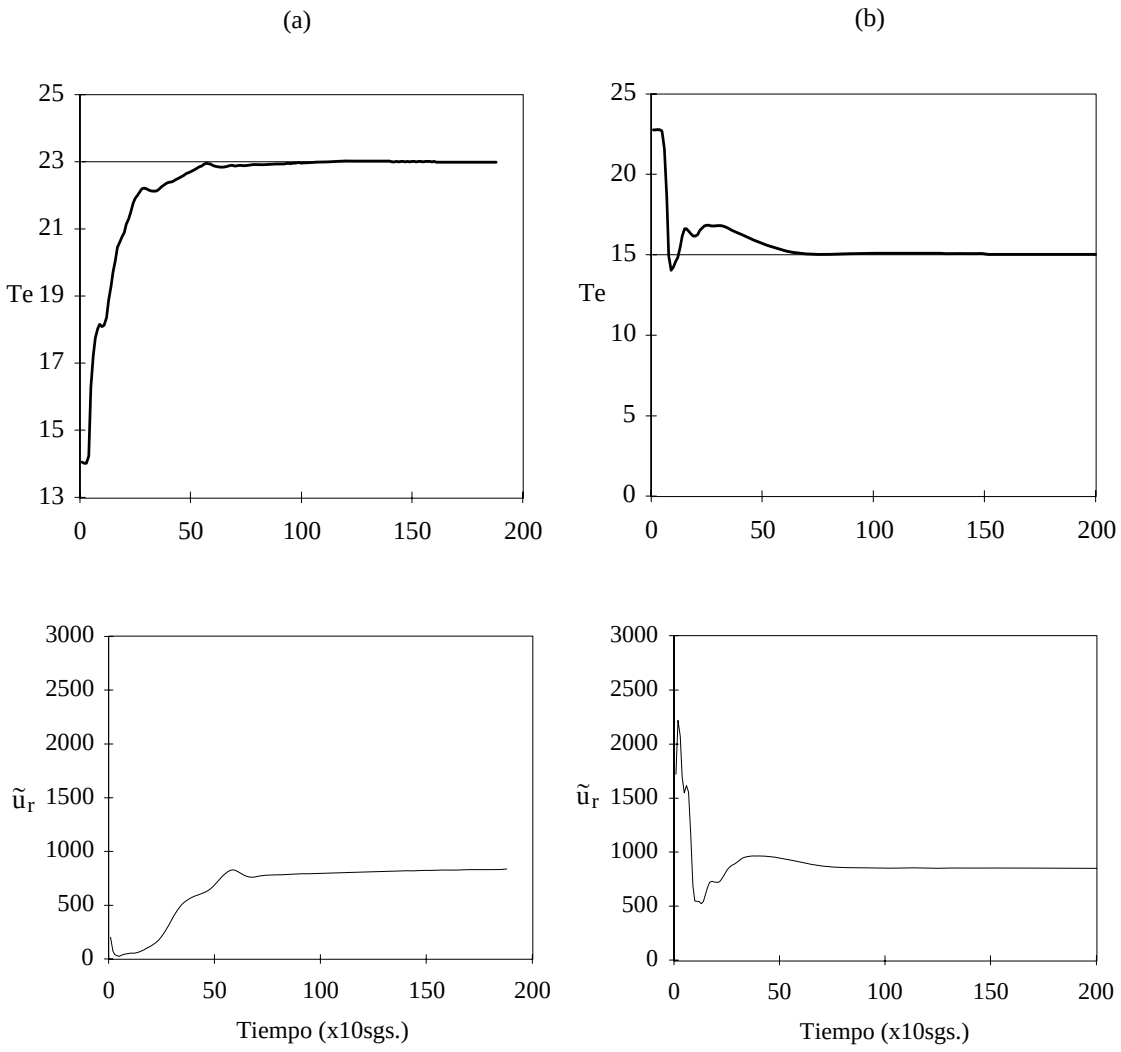


Figura 4.21. Controlador neuronal predictivo con N_u=0, H=40 y pesos iniciales $W_{\tilde{G}_r}$ (generalizado).

Parte superior: evolución de la temperatura Te en función del tiempo.

Parte inferior: evolución de la acción de control $\tilde{u}_r$ en función del tiempo.

(a) $Te^{des} = 23^o$ C (b) $Te^{des} = 15^o$ C.

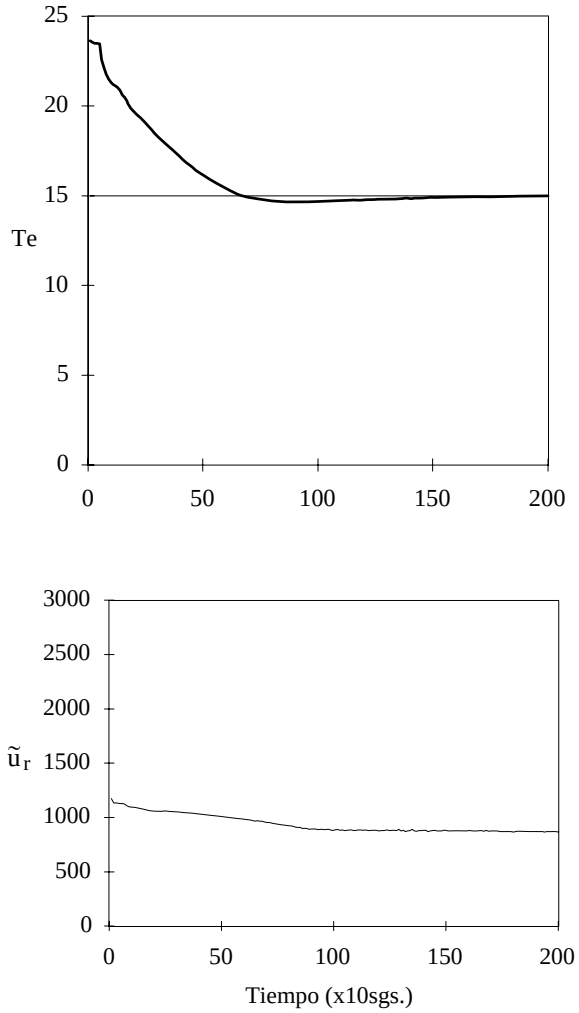


Figura 4.22. Controlador neuronal predictivo con N_u=0, H=70 y pesos iniciales $W_{\tilde{G}_r}$ (generalizado).

Parte superior: evolución de la temperatura Te en función del tiempo.

Parte inferior: evolución de la acción de control $\tilde{u}_r$ en función del tiempo.

$Te^{des} = 15^o C$.

similares o equivalentes a las que resultan de integrar el modelo físico dado por las ecuaciones 4.12-4.20. Esto es interesante ya que, debido a la presencia de ecuaciones diferenciales, los modelos físicos suelen ser difíciles de tratar, sobre todo en aplicaciones en tiempo real, pues el tiempo de integración puede ser superior al tiempo de muestreo. Por otra parte, la estimación de los parámetros que intervienen en dichos modelos puede ser muy laboriosa y requiere, generalmente, de un tiempo de cálculo demasiado grande, lo cual es también un inconveniente cuando dicha estimación tiene que realizarse en tiempo real. Por ejemplo, las constantes de tiempo del modelo físico presentado en el apartado 4.2.2 fueron determinadas después de varios días de cálculo en una SUN Workstation.

- Como se ha dicho anteriormente, los modelos NARMA neuronales pueden remplazar a los modelos físicos, siempre que sus parámetros sean estimados de manera adecuada.

  Los resultados presentados en el apartado 4.2.1 muestran que, si el objetivo es construir un modelo NARMA capaz de actuar como simulador del proceso que gobierna la dinámica de la temperatura del fluido de intercambio de calor que circula en la camisa del reactor químico, es necesario considerar los modelos NARMA de identificación en paralelo construidos con la red parcialmente recurrente propuesta en este trabajo, de modo que sus parámetros sean identificados con el propósito de simular la dinámica de dicho proceso. Como se observa en la figura 4.10(a) o en la tabla 4.2, el modelo NARMA en paralelo dado por la ecuación 4.8 no simula de forma eficiente el comportamiento dinámico del proceso. Sin embargo, la capacidad del modelo neuronal para simular esta dinámica se recupera cuando dicho modelo se construye y se entrena utilizando la red parcialmente recurrente (ecuación 4.10), como se observa en la figura 4.11(a) o tabla 4.3, consiguiéndose un error de predicción considerablemente inferior.

  Para el segundo modelo NARMA considerado (ecuación 4.5), la estructura en paralelo construida con pesos $W_{\tilde{F}}$ (ecuación 4.9) proporciona una representación que podría considerarse aceptable (ver figura 4.10(b) y tabla 4.2), dependiendo, evidentemente, de las exigencias. Sin embargo, es importante señalar que, incluso en este caso, el modelo en paralelo construido con la red parcialmente recurrente

(ecuación 4.11) tiene sus ventajas, no sólo desde el punto de vista que proporciona una mejor representación, como puede observarse en la figura 4.11(b) y tabla 4.3, sino también debido a que requiere de un menor número de iteraciones para conseguir un mínimo de la superficie descrita por el error de predicción. Esto es una ventaja fundamental, sobre todo en aplicaciones en las que la fase de identificación del modelo tiene que realizarse en tiempo real; en estas situaciones es de crucial importancia el tiempo que requiere el modelo para proporcionar una respuesta adecuada.

- Es interesante analizar también las capacidades de generalización de los modelos NARMA neuronales. Generalmente, para obtener buenas capacidades de generalización por parte de modelos construidos con redes de neuronas, es necesario llevar a cabo el aprendizaje de la red con un conjunto de ejemplos o patrones que cubran todo el rango de actuación de las variables que intervienen en el proceso.

  En los resultados presentados en la sección 4.2, se observa que para el conjunto *Dato2*, los perfiles de temperatura simulados por los modelos en paralelo propuestos en esta tesis, no sólo son satisfactorios (ver figura 4.12 y tabla 4.4), sino que incluso son equivalentes a los obtenidos con el modelo físico (figura 4.14(b)). No es el caso, sin embargo, para el conjunto de datos experimentales *Dato3*, como puede observarse en la figura 4.13 y en la tabla 4.4. Para dicho conjunto, los modelos neuronales no proporcionan aproximaciones adecuadas; esto es debido a que las temperaturas pertenecen a un rango diferente del que se ha utilizado para llevar a cabo la fase de aprendizaje de la red o fase de identificación del modelo, hecho que no sucede cuando se utiliza *Dato2*. Cuando las neuronas de contexto de la red parcialmente recurrente memorizan las diferencias entre Te y Tr (modelo dado por la ecuación 4.11), se obtienen mejores capacidades de extrapolación, como se observa en la figura 4.13(b), aunque esto no es mas que una estrategia para obtener buenas aproximaciones. Para garantizar resultados equivalentes a los obtenidos por el modelo físico (figura 4.14(c)), sería necesario llevar a cabo el aprendizaje de la red en dicho rango.

Sin embargo, esto no supone una limitación de los modelos NARMA neuronales, ya que estos modelos pueden adaptarse en tiempo real cuando sea necesario disponer de un modelo en dicho rango de temperaturas.

- Con respecto a los resultados de control, es necesario señalar que la introducción de técnicas no lineales y el uso de una modelización adecuada del proceso mejora, en general, el control de la temperatura del fluido que circula en la camisa del reactor, como puede deducirse de los resultados en la sección 4.3. Tanto las estrategias de control inverso como las estrategias de control predictivo desarrolladas producen un control adecuado y eficiente del proceso dinámico en tiempo real. La calidad del control obtenido con las técnicas neuronales (figuras 4.20, 4.21, 4.22) es claramente superior al control que proporciona el controlador PID actualmente incorporado en la instalación (figura 4.15).

- Sobre los resultados obtenidos al implementar el esquema de control inverso propuesto para controlar la temperatura del fluido de intercambio de calor, en primer lugar, debe señalarse que los pesos conseguidos al finalizar la fase de aprendizaje generalizado del controlador (fase 1), no producen un control eficiente del proceso dinámico real (figura 4.17), ya que el conjunto de datos utilizado para determinar dichos parámetros no representa la dinámica inversa del proceso en su totalidad (sólo unas pocas acciones de control han sido representadas, figura 4.16). Sin embargo, sí poseen información suficiente como para evitar situaciones de “no control” del proceso dinámico.

  Para obtener un mejor control de la temperatura del fluido de intercambio de calor, es necesario llevar a cabo la fase de aprendizaje especializado del controlador inverso, mediante la cual los pesos se especializan en el objetivo de control deseado y, como consecuencia, adquieren una información más completa sobre la situación a controlar, pudiéndose así mejorar el control de dicha temperatura.

  Una vez finalizada la fase de aprendizaje especializado del controlador inverso utilizando un modelo del proceso (fase 2), el controlador de pesos fijos $W_{\tilde{G}_r}(\text{especializado\_ m})$ produce un mejor control del proceso, como se observa en la figura 4.19. Esta mejora se aprecia, principalmente, en que la temperatura no sufre

bruscas oscilaciones para alcanzar el objetivo de control deseado, tanto en el modo de calentamiento como en el modo de enfriamiento.

Sin embargo, se observa también que para el objetivo de control correspondiente al enfriamiento del fluido (figura 4.19(b)), el controlador de pesos fijos $W_{\tilde{G}_r}(\text{especializado\_m})$ hace que la salida del proceso se estabilice a 15.4 ^{o}C en lugar de hacerlo a 15 ^{o}C. Esto es debido a que al realizar la 2ª fase de aprendizaje, los pesos o parámetros del controlador se especializan en controlar un modelo del proceso dinámico real que no es exacto. Dichos errores se corrigen llevando a cabo el aprendizaje especializado del controlador inverso en tiempo real (fase 3), como se observa en la figura 4.20(b).

Por tanto, y con el objetivo de garantizar un buen control del proceso real cuando se lleva a cabo una estrategia de control inverso, será necesario realizar cada una de las fases de aprendizaje que se indican en el apartado 3.3.1.

- Como se decía en la sección 2.2 del capítulo 2, el éxito de una estrategia de control inverso con aprendizaje especializado depende, en gran medida, de los pesos o parámetros utilizados para inicializar el controlador neuronal. Esto puede observase en los resultados obtenidos en el apartado 4.3.1.

  Cuando se realiza el aprendizaje especializado del controlador utilizando directamente el proceso real (figura 4.20), el comportamiento de dicho controlador es prácticamente idéntico al comportamiento del controlador de pesos fijos $W_{\tilde{G}_r}(\text{especializado\_m})$ (figura 4.19), únicamente se corrigen pequeños errores. Esto implica que no hubiera sido posible obtener un buen control del proceso si se hubiese realizado únicamente el aprendizaje especializado del controlador directamente con el proceso real, sino que es necesario partir de inicializaciones de los pesos que posean ya bastante información sobre la situación que se pretende controlar.

  Además, es importante también la forma o el método utilizado para obtener esta inicialización. Al analizar los resultados se observa que el aprendizaje generalizado del controlador, es decir, aproximar previamente la dinámica inversa del proceso, favorece, en gran medida, a la obtención de un control adecuado del proceso real. En la figura 4.20(b), por ejemplo, se puede ver que, incluso después de realizar el aprendizaje especializado del controlador utilizando un modelo del proceso (fase 2),

dicho controlador sigue provocando oscilaciones en la temperatura. En las simulaciones realizadas se ha observado que no es posible corregir dicho comportamiento ni siquiera realizando más ciclos de aprendizaje utilizando un modelo del proceso. Para corregirlo hubiera sido necesario partir de pesos que tuvieran más información sobre el comportamiento de la dinámica inversa del proceso en el modo de enfriamiento, es decir, llevar a cabo un aprendizaje generalizado con más datos correspondientes al enfriamiento del fluido. Nótese que el conjunto de datos utilizado para llevar a cabo el aprendizaje generalizado del controlador (*Dato2*, figura 4.16) posee pocos ejemplos o patrones representando el modo de enfriamiento (acciones de control superiores a 900).

Por tanto, partir de pesos que posean información sobre la dinámica inversa del proceso, no sólo ayuda a evitar situaciones de no control cuando se plantea una estrategia de control inverso, sino que también favorece a obtener un mejor control del proceso dinámico real.

Análisis comparativo de los sistemas de control inverso y predictivo

Debido a que se desarrollan dos esquemas de control neuronal, existe la necesidad de realizar un estudio comparativo de dichos esquemas. En principio son dos estrategias diferentes de control y, como tales, cada una tiene sus ventajas e inconvenientes. El estudio se limita, simplemente, a analizar estas características y a establecer factores que pueden influir en la decisión de realizar una u otra estrategia de control.

Como se ha comentado, las estrategias de control inverso presentan una gran dependencia con respecto a los parámetros que se utilizan para inicializar el controlador que actuará en tiempo real. Puesto que el esquema de control predictivo propuesto en este trabajo se trata de un sistema de control on-line en el que la acción de control viene dada por la salida de una red de neuronas, la inicialización de los pesos de dicha red tiene también su importancia en el resultado final. Es necesario destacar, sin embargo, que esta dependencia es mayor en el caso del sistema de control inverso. En la figura 4.22 se observa que, partiendo de los pesos obtenidos al finalizar el aprendizaje generalizado del controlador, es posible obtener un control adecuado de la temperatura del fluido de intercambio de calor cuando se aplica una estrategia de control predictivo, hecho que no es posible al realizar la estrategia de control inverso. Esto muestra que, para los sistemas de control predictivo, el resultado de control no está tan ligado al

conjunto de parámetros que se utiliza para inicializar el controlador. El motivo principal es la función error que se utiliza para adaptar los pesos del controlador predictivo (ecuación 3.44). Dicha función contiene una información más completa sobre el comportamiento dinámico del proceso, ya que evalúa cómo actuará el proceso a lo largo del horizonte de predicción, en lugar de utilizar únicamente el error medido en la salida del proceso (ecuación 3.41), como es el caso de las estrategias de control inverso. Esta información en el futuro permite al controlador predictivo corregir con una mayor coherencia acciones de control no adecuadas y provocadas por una mala inicialización de los pesos de la red.

Como consecuencia, las posibilidades de obtener inicializaciones de los parámetros del controlador influirá en la decisión de utilizar una u otra estrategia de control.

Desde un punto de vista lógico y teórico, una estrategia de control predictivo provocará siempre el mejor control del proceso, puesto que la información utilizada para calcular la acción de control en cada instante de tiempo permite al controlador anticiparse a lo que sucederá en el futuro. De hecho, si se piensa un poco en los problemas de control que resuelve con éxito el ser humano en su vida cotidiana, como el simple hecho de conducir un coche, las estrategias de control que se utilizan son estrategias de control predictivo. Desde un punto de vista práctico, sin embargo, no siempre una estrategia de control predictivo va a provocar el mejor control del proceso. Los principales factores que influyen en este hecho son, por una parte, la elección de los horizontes de control y predicción y por otra, el modelo disponible sobre el proceso dinámico.

Los horizontes de control y predicción elegidos para realizar la estrategia de control predictivo influyen en el funcionamiento del controlador; éstos tienen que ser elegidos adecuadamente y conforme a la naturaleza del proceso que se está controlando, de modo que pueda obtenerse un control adecuado del proceso dinámico real. Sin embargo, esto no es una limitación seria, ya que puede elegirse por prueba y error utilizando un modelo del proceso.

Por otra parte, e incluso asumiendo que los horizontes de control y predicción son apropiados, la capacidad del modelo para representar al proceso dinámico real en el horizonte de predicción tiene una repercusión importante en el comportamiento del controlador predictivo. Una predicción con errores puede provocar controladores

inestables y en general no apropiados, ya que las acciones de control se estarían calculando en base a una información errónea. Por tanto, para garantizar el éxito con una estrategia de control predictivo, es necesario disponer de un modelo capaz de explicar la dinámica del proceso adecuadamente.

El esquema de control inverso necesita también de la presencia de un modelo del proceso, tanto en la segunda como en la tercera fase de aprendizaje, aunque, en general, estas estrategias de control son menos sensibles a dicho modelo, ya que los errores existentes en el controlador debido a la utilización de un modelo del proceso pueden corregirse en tiempo real.

Por tanto, la capacidad de representación del modelo disponible sobre el proceso dinámico que se pretende controlar dictará, en cierta medida, si es conveniente utilizar una estrategia de control inverso o una estrategia de control predictivo.

Como consecuencia, no se puede establecer cuál de los sistemas de control es preferible usar, sino que cada uno tiene sus propias características. En el caso del proceso dinámico que gobierna el comportamiento de la temperatura del fluido de intercambio de calor, las estrategias de control predictivo producen, generalmente, un mejor control del proceso. Téngase en cuenta que se disponen de modelos que, aunque no son exactos, sí son capaces de explicar casi la totalidad de la dinámica del proceso. En el caso del calentamiento, la estrategia de control predictivo es capaz de controlar más rápidamente dicha temperatura, existiendo una diferencia de aproximadamente 4000 sgs. (figuras 4.20(a), 4.21(a)). En el caso del enfriamiento, sin embargo, son las estrategias de control inverso las que producen un control más rápido, aunque es necesario señalar que el esquema de control predictivo ofrece un control sin oscilaciones en la temperatura (figura 4.22).

Conclusiones y futuras líneas de investigación

Conclusiones

Un modelo apropiado de un proceso dinámico tiene que poseer la propiedad de saber actuar de forma eficiente como simulador del proceso. Los modelos NARMA neuronales de identificación en serie-paralelo no poseen dicha propiedad, ya que requieren de una secuencia discreta de valores medidos en el pasado de la variable de salida del proceso para predecir o aproximar la salida actual de dicha proceso, valores que no están disponibles cuando se simula el comportamiento dinámico del proceso. Los modelos NARMA con estructura en paralelo pueden, sin embargo, simular la dinámica del proceso, puesto que los valores pasados de la salida real del proceso se sustituyen por los valores predichos por la red de neuronas en instantes anteriores de tiempo. El resultado presentado en el apartado 3.2.1 demuestra que el modelo en paralelo construido con los parámetros que resultan de identificar el correspondiente modelo de identificación en serie-paralelo no tiene por qué proporcionar una predicción adecuada de la salida del proceso dinámico, ya que estos parámetros han sido determinados para minimizar el error de identificación (ecuación 3.17) y no con el objetivo de construir un simulador del proceso.

Para poder garantizar un buen funcionamiento de los modelos NARMA con estructura en paralelo, es necesario que sus parámetros sean estimados con el propósito de simular el comportamiento dinámico del proceso, lo cual es equivalente a minimizar el error de predicción (ecuación 3.23). Esto implica que la arquitectura de red parcialmente recurrente descrita en el apartado 3.1.2, tiene que utilizarse, no sólo para construir el modelo, sino también para estimar sus parámetros. De este modo, si el

objetivo es construir un modelo NARMA capaz de actuar como simulador del proceso proporcionando una aproximación adecuada de la dinámica de dicho proceso, la red parcialmente recurrente podrá utilizarse para aproximar el funcional F que determina el modelo NARMA.

Es importante señalar también que, incluso en las situaciones en las que los modelos en paralelo construidos con los pesos que resultan de identificar el modelo en serie-paralelo proporcionen una representación aceptable del proceso dinámico, es conveniente y aconsejable utilizar los modelos en paralelo propuestos en esta tesis, ya que requieren de un menor número de iteraciones para conseguir un mínimo del error de predicción.

El éxito de los sistemas de control inverso construidos con redes de neuronas artificiales depende, podría decirse que casi exclusivamente, de los parámetros utilizados para inicializar la red que actúa como controlador inverso del proceso dinámico en tiempo real. Inicializar dicha red con pesos aleatorios para controlar el proceso dinámico real puede producir situaciones peligrosas e incontrolables, principalmente cuando se tratan procesos dinámicos industriales. Es necesario partir de inicializaciones que posean información suficiente sobre la situación que se pretende controlar. Ciertamente, se puede hablar de un aprendizaje del controlador neuronal en tiempo real, pero siempre con el objetivo de mejorar el control que produce las inicializaciones disponibles.

En este contexto, es importante también la forma o el método que se utiliza para conseguir dicha inicialización. El realizar el aprendizaje especializado del controlador utilizando un modelo del proceso permite que el controlador inverso pueda especializarse en el objetivo de control deseado y, de este modo, conocer la situación que se pretende controlar. Sin embargo, en muchos casos, las inicializaciones que resultan de realizar únicamente esta fase de aprendizaje no proporcionan el mejor control del proceso en tiempo real. Generalmente, el aprendizaje generalizado del controlador, es decir, la obtención de pesos que posean información sobre la dinámica inversa del proceso en su totalidad, ayuda a conseguir un mejor control del proceso dinámico, ya que el controlador puede conocer así tendencias generales del comportamiento dinámico del proceso.

Ambos aprendizajes, generalizado y especializado, ayudan, cada uno en su medida, a obtener inicializaciones para que el controlador inverso que actúa en tiempo real pueda producir un buen control del proceso dinámico.

El esquema de control predictivo propuesto en este trabajo se caracteriza porque es aplicable en tiempo real, reduciéndose el esfuerzo computacional requerido por las estrategias de control predictivo no lineales disponibles en la actualidad. La naturaleza adaptativa y la habilidad de las redes de neuronas artificiales para capturar relaciones desconocidas y almacenar información en sus pesos favorece a que la red aprenda a realizar una estrategia de control predictivo a medida que se avanza en el tiempo. Estas propiedades hacen que los cálculos realizados en cada instante de tiempo para adaptar los pesos de la red sea menores que si se tuviera que resolver un problema de optimización no lineal. Nótese que la resolución de un problema de optimización no lineal en cada instante de tiempo para calcular la acción de control implicaría adaptar dicha señal hasta conseguir que la función error definida por la ecuación 3.44 sea cercana a cero. Esto puede suponer la realización de un gran número de iteraciones o adaptaciones en cada periodo de muestreo, mientras que la consideración de una red de neuronas para calcular la acción de control predictivo supone una única adaptación en cada periodo de muestreo.

Por otra parte, el poder establecer una relación entre la acción de control predictivo y las variables que intervienen en el comportamiento dinámico del proceso permite que pueda realizarse una estimación off-line de los parámetros del controlador predictivo, lo cual facilita el aprendizaje de la red en tiempo real y favorece a un mejor control del proceso.

Finalmente, mencionar que los sistemas de control inverso y predictivo estudiados en esta tesis tienen sus propias características y serán las condiciones específicas de cada proceso dinámico las que determinarán cuál de ellos producirá un control más eficaz. Dependiendo de la naturaleza del proceso real, de la información y datos disponibles, una estrategia de control puede ser preferible a la otra.

No obstante, cada uno de ellos poseen ciertas ventajas e inconvenientes. De forma resumida se puede concluir que el comportamiento de ambos controladores depende de los pesos utilizados para inicializar la red de neuronas que actúa como controlador del

proceso en tiempo real, aunque esta dependencia toma una menor importancia en los sistemas de control predictivo. En este caso, los parámetros del controlador se adaptan en base al error medido a lo largo del horizonte de predicción (ecuación 3.44), mientras que el esquema de control inverso utiliza una información más pobre, el error instantáneo medido en la salida del proceso (ecuación 3.41). Por otra parte, sin embargo, son las estrategias de control predictivo las que dependen en una mayor medida de la capacidad del modelo disponible para aproximar la dinámica del proceso, dependencia que se reduce para los esquemas de control inverso.

Futuras líneas de investigación

En el contexto de la modelización y el control de procesos dinámicos utilizando redes de neuronas artificiales existe una serie de temas y cuestiones que no han sido tratadas en su totalidad en esta tesis y que, sin embargo, se consideran importantes. Se cree, por tanto, conveniente reflexionar sobre ellas y considerarlas como futuras líneas de investigación. A continuación se exponen estas cuestiones y se presentan posibles líneas a seguir para su posterior investigación.

Debido a que los modelos NARMA no lineales, ya sean con estructura en serie-paralelo como con estructura en paralelo, requieren de secuencias discretas de las variable de entrada y salida del proceso para aproximar la salida actual del proceso, sería interesante disponer de métodos generales que permitan establecer la longitud mínima de dichas secuencias, así como las variables de entrada del proceso verdaderamente influyentes en su comportamiento dinámico. De este modo, se evitaría el tener que considerar variables que no son importantes, pero que, sin embargo, su consideración aumenta de forma significativa el número de parámetros ajustables de la red y por tanto complican su aprendizaje.

La idea utilizada en este trabajo para eliminar las variables no significativas y fijar la longitud mínima de las secuencias discretas cuando se construyen modelos NARMA consiste en entrenar un perceptron multicapa con todas las posibles variables de entrada y evaluar la importancia de dichas variables en la salida, a partir de los pesos de la red ya entrenada (apartado 4.2.1). Los valores definidos en la ecuación 4.4 permiten

determinar, aunque de forma experimental, las variables que deben considerarse para la construcción de los modelos NARMA capaces de explicar el comportamiento dinámico del proceso que gobierna la evolución de la temperatura del fluido de intercambio de calor que circula en la camisa del reactor químico. Sin embargo, no ha sido posible obtener un resultado teórico que relacione dichas cantidades con las variaciones o derivadas parciales de la salida de la red respecto a las entradas, resultado que hubiera permitido estudiar y establecer el número mínimo de variables que tienen que considerarse cuando se construyen modelos NARMA, para cualquiera que sea el proceso dinámico que se esté tratando.

Una futura línea de investigación en este contexto será insistir en estas cuestiones y encontrar relaciones teóricas entre la variación de la salida de la red respecto a cada una de las entradas y alguna medida en función de los pesos de la red, de forma que, si esta medida es próxima a cero, pueda deducirse que la derivada de la salida de la red respecto a la entrada correspondiente es también cercana a cero, y que, por tanto, dicha variable puede ser eliminada del vector de entrada. Al mismo tiempo, sería necesario también determinar, a partir de estas medidas, la importancia de unas variables de entrada con respecto a otras y poder así decidir el menor número de variables que tienen que considerarse para construir modelos NARMA capaces de explicar el comportamiento de cada proceso dinámico.

En el contexto de modelización y control de procesos dinámicos, la estabilidad de modelos y controladores ocupa un lugar importante. De forma simplificada, se puede decir que el estudio de la estabilidad de un sistema consiste en analizar el comportamiento de las trayectorias que describe dicho sistema. Para los sistemas descritos mediante una relación entrada-salida, el estudio de la estabilidad consiste en poder establecer que para entradas acotadas, las salidas están también acotadas.

Cuando se construye un modelo de un proceso dinámico, cabe preguntarse si dicho modelo es o no estable, ya que problemas de estabilidad por parte del modelo puede conducir a situaciones indeseadas, principalmente en aplicaciones de control, como, por ejemplo, en las aplicaciones de control predictivo. La utilización de un modelo que sea sensible a pequeños cambios en las entradas, puede producir un control predictivo inestable y no robusto.

Los resultados obtenidos en el apartado 4.2.1 y las numerosas simulaciones realizadas permiten intuir que los modelos NARMA neuronales construidos en esta tesis son estables. Sería interesante, sin embargo, poder demostrar su estabilidad y, de este modo, tener la seguridad de que son modelos fiables que pueden utilizarse para implementar las estrategias de control predictivo.

Una posible línea de investigación a seguir para demostrar la estabilidad de los modelos NARMA de identificación en paralelo construidos con la red parcialmente recurrente podría estar basada en resultados de la teoría de estabilidad de entrada-salida y de la teoría de estabilidad de Lyapunov. Para ello se haría uso del método de linearización de Lyaponov, el cual consiste en transformar el modelo neuronal ya entrenado en un sistema lineal dado por la matriz Jacobiana de la relación entre el vector de entrada y la salida. Debido a que existen resultados que demuestran que el sistema lineal que se obtiene poseen las mismas propiedades de estabilidad que el sistema inicial, bastaría estudiar la estabilidad de este último sistema lineal. A este punto es posible realizar un estudio de los exponentes de Lyapunov de la matriz Jacobiana y analizar si es posible deducir a partir de dichos exponentes, propiedades de estabilidad de los modelos NARMA de identificación en paralelo propuestos en este trabajo.

Al igual que es importante conocer las propiedades de estabilidad de los modelos neuronales, también lo es conocer dichas propiedades para el bloque cerrado formado por el controlador neuronal y el proceso, de modo que pueda garantizarse la estabilidad de los controladores construidos. Para ello, podrían utilizarse las mismas ideas expuestas anteriormente, aunque en este caso sería necesario calcular la matriz Jacobiana del bloque.

Bibliografía

[AGSE, 87] Agarwal M. y D.E. Seborg:
*“Self-Tunning Controllers for Nonlinear Systems”. Automatica 23, 209-214, 1987.
*“Multivariable Nonlinear Self-Tunning Controller”. AIChE Journal 33,1379-386, 1987.

[ALAR, 92] Álvarez J. y E. Aranda: “Nonlinear Adaptive Control of a Chemical Reactor using Singular Perturbation Techniques”. Dynamics and Control of Chemical Reactors DYCORD+'92, Maryland, USA, 1992.

[ARZE, 89] Årzen K.-E.: “An architecture for expert system based feedback control”. Automatica 25:(6), 813-827, 1989.

[ASAN, 86] Åström K.J., J.J. Anton y K.-E Årzén: “Expert Control”. Automatica 22:(3), 277-286, 1986.

[ASMC, 92] Åström K.J. y T. J. McAvoy: “Intelligent Control: An Overview and Evaluation”. Capítulo 1 de "Handbook of Intelligent Control", ed. D.A. White and D.A. Sofge, Van Nostrand Reinhold, New York, 1992.

[ASTR, 96] Åström K.J.: “Adaptive Control Around 1960”. IEEE Control Systems, Vol 16, 3, 41-49, 1996.

[ASWI, 73] Åström K.J. y B. Wittenmark: “On Self-Tunnig Regulators”. Automatica 9, 185-199, 1973.

[ASWI, 90] Åström K.J. y B. Wittenmark: “Computer-Controlled Systems. Theory and design”. Prentice-Hall International Editions, 2ª Edición, 1990.

[ATFA, 66] Athanas M. y P.L. Faib: “Optimal Control: An Introduction to the Theory its Applications”. MC Graw-Hill, 1966.

[BAGA, 97] D. Barrios, I.M. Galván, P. Isasi and J. Ríos: “Improving inverse control on real time by means of neural networks”. Aceptado en Proc.

ICONIP/ANZIIS/ANNES'97. University of Otago, New Zeland, Noviembre, 1997.

[BAGA, 97] D. Barrios, I.M. Galván, P. Isasi, J. Ríos and J.M. Zaldívar: "A new neural network approach to allow predictive control on real time". Submited to 9th International Symposium on System Modelling Control. Institute of Computer Science of the Technical Institute of Lodz, Poland, 1997.

[BAGA, 97] D. Barrios, I.M. Galván, P. Isasi and J. Ríos: "NARMA recurrent neural modelling for controlling chemical bath reactors". Technical Report. Submited to Department of Computer Science. University of Carlos III, 1997.

[BASU, 83] Barto A.G., R.S. Sutton y C.W. Anderson: "Neuronlike Adaptive Elements that can be Solve Difficult Learning Control Problems". IEEE Transactions on Systems, Man and Cybernetics SMC-13, 834-846, 1983.

[BICH, 89] Billings S.A. y S. Chen: "Identificaion of nonlinear rational systems using a prediction-error estimation algortihm". Int J. of Systems Science, Vol 29, 467-494, 1989.

[BHMC, 90] Bhat N. y T.J. McAvoy: "Use of Neural Nets for Dynamic Modelling and Control of Chemical Process Systems". Computers and Chemical Engineering 14 Nº 4/5, 573-583, 1990.

[BRAY, 96] Braake H.A.B., M. Ayala y H.J.L. van Can: "Neural Model Based Predictive Control". Journal on Automatic Control, Vol 37, Nº3, 1996.

[BROG, 85] Brogan W. L.: "Modern Control Theory". 2ª ed, Prentice-Hall, 1985.

[CAGR, 90] Carpenter G.A. y S. Grossberg: "Adaptive Resonance Theory: Neural Network Architectures For Self Organising Pattern Recognition". Parallel Processing in Neural Systems and Computers, 383-389, Elsevier Science Publ. (North-Holland), 1990.

[CASE, 83] Cameron F. y D.E. Seborg: “A Self-Tunning Controller with a PID Structure”. International Journal of Control, 38, 401, 1983.

[CHBI, 89] Chen S. y S.A Billings: “Recursive prediction error estimator for nonlinear models”. Int J. of Control, Vol 49, 569-594, 1989

[CHBI, 89] Chen S., S.A Billings y W. Luo: “Orthogonal least squares methods and their application to non-linear system identification”. Int J. of Control, Vol 50, Nº 5, 1873-1896, 1989.

[CHBI, 90] Chen S., S.A. Billings, C.F.N. Cowan y P.M. Grant: “Practical identification of NARMAX models using radial basis functions”. Int. J. of Control, Vol 52, Nº 6, 1327-1350, 1990.

[CHBI, 92] Chen S. y S.A. Billings: “Neural networks for nonlinear dynamic systems modelling and identification”. Int. J. of Control, Vol 56, Nº 2, 319-346, 1992.

[CHKH, 95] Fu-Chuang Chen y Hassan K. Khalil: “Adaptive Control of a Class of Nonlinear Discrete-Time Systems Using Neural Networks”. IEEE Transactions on Automatic Control, Vol 40, 5, 791-801, 1995.

[CHVA, 96] Choi J.Y., H.F. Vanlandingham y S. Bingulac: “A Constructive Approach for Nonlinear System Identification Using Multilayer Perceptrons”. IEEE Transactions on Systems, Man and Cybernetic, Vol 26, 2, 307-312, 1996.

[CLGA, 75] Clarke D.W. y P.S. Gawthrop: “On Self-Tunning Controllers”. Proc. IEE, 122, 929-934, 1975.

[CLMO, 87] Clarke D.W., C. Mohtadi y P.S. Tuffs:
*”Generalized Predictive Control- Part I. The Basic Algoritnm”. Automatica 23, Nº 2, 137-148, 1987.

*"Generalized Predictive Control- Part II. Extensions and Interpretations". Automatica 23, Nº 2, 149-160, 1987.

[COSA, 97] Cohen B., D. Saad y E. Maram: "Efficient Training of Recurrent Neural Network with Time Delays". Neural Networks, 10, 1, 51-59, 1997.

[COWA, 67] Cowan. J.D.: "A mathematical theory of central nervous activity". PhD Thesis, Univ. London, U.K, 1967.

[CYBE, 89] Cybenko G.: "Approximation by superposition of a sigmoidal function". Mathematics of Control, Signals, and Systems, 2, 303-314, 1989.

[DREN, 95] Draeger A., S. Engell and H. Ranke: "Model Predictive Control Using Neural Network". IEEE Control Systems, Vol 15, Nº 5, 61-66, 1995.

[EARA, 92] Eaton J.W. y J.B. Rawlings: "Model-Predictive Control of Chemical Processes". Chemical Engineering Science, Vol 47, Nº 4, 705-720, 1992.

[ELMA, 88] Elman J. L.: "Finding Structure in Time". CRL Technical Report 8801. Center for Research in Language, University of California, San Diego, 1988.

[FAKA, 96] Fabri S. y V. Kadirkamanathan.: "Dynamic Structure Neural Networks for Stable Adaptive Control of Nonlinear Systems". IEEE Transactions on Neural Networks, Vol 7, 5, 1151-1167, 1996.

[GAZA, 96] Galván I.M., J.M: Zaldívar, H. Hernández y E. Molga: "The use of neural networks for fitting complex kintic data". Computers and Chemical Engineering, Vol 20, 12, 1451-1465, 1996.

[GOPA, 77] Goodwin G.C. y R.L. Payne: "Dynamic System Identification: Experiment Design and Data Analysis". New York: Academic Press, 1977.

[GOSI, 84] Goodwin G.C. y K. S. Sin: "Adaptive Filtering Prediction and Control". Prentice-Hall Information Systems, Englewood Cliffs, New Jersey, 1984.

[HAKE, 90] Hartman E.J., J.D. Keeler y J.M. Kowalski: “Layered Neural Networks with Gaussian Hidden Units as Universal Approximations”. Neural Computation 2, 210-215, 1990.

[HAST, 87] Handelman D.A. y R.F. Stengel: “An architecture for real time rule-based control”. In Proceedings of the American Control Conference, 1636-1642, Minneapolis, MN, 1987.

[HAWK, 83] Hawk W. M.: “A Self-Tunning, Self-Contained PID Controller”. Proc. 1983 American Control Conference, San Francisco, 838, 1983.

[HOPF, 82] Hopfield J.J.: “Neural Networks and Physical Systems with Emergent Collective Computational Abilities”. Proc. Nat. Acad. Sci., U. S., vol 79, 2554-2558, 1982.

[HOPF, 84] Hopfield J.J.: “Neurons With Graded Response Have Collective Computational Properties Like Those of Two-State”. Proc. Natl. Acad. Sci. USA, 81, 3088-3092, 1984.

[HOST, 89] Hornik K., M. Stinchcombe y H. White: “Multilayer feedforward networks are universal approximators”. Neural Networks 2, 359-366, 1989.

[ISID, 89] Isidori A.: “Nonlinear Control Systems: An introduction”. 2ª ed. Springer-Verlag, Berlín 1989.

[JINI, 94] Jin L., P.N. Nikiforuk y M.M. Gupta: “Adaptive Control of Discrete-Time Nonlinear Systems using Recurrent Neural Networks”: IEE Proc-Control Theory Appl. Vol 141, Nº 3, May 1994.

[JOHA, 93] Rolf Johansson.: “System Modeling Identification”. Prentice Hall, Englewood Cliffs, NJ, 1993.

[JORD, 86] Jordan M.I.: “Serial Order: A Parallel Distributed Processing Approach”. ICS Report 8604. Institute for Cognitive Science, University of California, San Diego, 1986.

[JORD, 89] Jordan M.I.: “Generic Constraints on Underspecified Target Trajectories”. IJCNN Proceedings, IEE, New York, June, 1989.

[JORD, 90] Jordan M.I. y R.A. Jacobs: “Learning to Control Unstable Systems with Forward Modelling”. Advances in Neural Information Processing Systems 2, D.S, Touretzky ed, Morgan Kaufmann Publishers, 1990.

[KAWA, 92] Kambhampati C. Y K. Warwick: “Use of Stochastic Neural Networks for Process Control”. IFAC Dynamics and Control of Chemical Reactors (DYCORD+´92), Maryland, USA, 1992.

[KOHO, 90] Kohonen T.: “The Self-Organizing Map”. Proc of the IEEE, 78, 1464-1480, 1990.

[KOPO, 95] Kosmatopoulos E.B., M.M. Polycarpou, M.A. Chistodoulou y P.A. Ioannou: “High-Order Neural Network Structures for Identification of Dynamical Systems”. IEEE Transactions on Neural Networks, Vol 6, Nº 2, 422-431, March, 1995.

[LAND, 74] Landau I.D.: “A Survey of Model Reference Adaptive Techniques”. Theory and Applications, Automatica 10, 353-379, 1974.

[LEBI, 85] Leontaritis I.J. y S.A. Billings: “Input-ouput parametric models for nonlinear systems. Part I: Deterministic nonlinear systems”. International Journal of Control 41(2), 303-328, 1985.

[LENA, 96] Levin A. y Narendra K.: “Control of nonlinear dynamical systems using neural networks. Part II: Observability, Identification and Control”. IEEE Transactions on Neural Networks, Vol 7, 1, 30-42, 1996.

[LENA, 95] Levin A. U. y K.S. Narendra: “Identification Using Feedforward Networks”. Neural Computation 7, 349-357, 1995.

[LEYE, 96] Lewis F. L., A. Yesildirek y K. Lin Narendra K.: “Multilayer Neural-Net Robot Controller with Garantueed Tracking Performance”. IEEE Transactions on Neural Networks, Vol 7, 2, 388-399, 1996.

[LIIR, 97] Lightbody G. y G.W. Irwin: “Nonlinear Control Structures Based on Embedded Neural System Models”. IEEE Transactions on Neural Networks, Vol 8, 3, 553-567, 1997.

[LOLE, 93] Low Teck-Seng, Tong_Heng Lee y Hock-Koon Lim: “A Methodology for Neural Network Training for Control of Drives with Nonlinearities”. IEEE Transactions on Industrial Electronics, Vol 39, Nº 2, April, 1993.

[MART, 76] Martín Sánchez J.M.: “Adaptive Predictive Control Systems”. Patente en EEUU, Nº 4, 197, 576. Fecha de prioridad, 4 de agosto de 1976.

[MATY, 65] Matyas, J.: “Random Optimizaction”. Automation and Remote Control, 26, 244-253, 1965.

[MCPI, 43] McCulloch W.S. y W. Pitts: “A logical Calculus of the Ideas Immanent in Nervous Activity”. Bull. Math. Biophys, 5, 115-133, 1943.

[MEMC, 70] Mendel J.M y R.W. MClaren: “Reinforcement Learning Control and Pattern Recognition Systems”. J.M. Mendel and K.S. Fu eds, “Adaptive learning and pattern recognition systems: Theory and applications”. 287-318. New York: Academics Press, 1970.

[MIPA, 69] Minsky M. y S. Papert: “Perceptrons”. The MIT Press, 1969.

[MISU, 91] Miller W.T., R.S. Sutton y P.J. Werbos (eds): “Neural Networks for Control”, MIT Press, 1991.

[MODA, 88] Moody J. y C. J. Darken: “Learning with localized receptive fields”. Proceedings of the 1988 Connectionist Models Summer School, eds. Touret-zky, Hinton and Sejnowski. Morgan-kaufmann, Publishers, 1988.

[MODA, 89] Moody J. y C. J. Darken: “Fast Learning in Networks of Locally-Tuned Processing Units”. Neural Computation 1, 281-294, 1989.

[MOPA, 92] Morningred J.D., B. E. Paden, D. E. Seborg y D. A Mellichamp: “An Adaptive Nonlinear Predictive Controller”. Chemical Engineering Science 47 Nº 4, 755-762, 1992.

[NAMU, 94] Narendra K.S. y S. Mukhopadhyay: “Adaptive Control of Nonlinear Multivariable Systems Using Neural Networks”. Neural Networks, Vol 7, Nº 5, 737-752, 1994.

[NAMU, 97] Narendra K.S. y S. Mukhopadhyay: “Adaptive Control Using Neural Networks and Approximative Models”. IEEE Transactions on Neural Networks, Vol 8, Nº 3, 475-485, 1997.

[NAPA, 90] Narendra K.S. y K. Pathasarthy: “Identification and Control of Dynamical Systems Using Neural Networks”. IEEE Transactions on Neural Networks Vol 1, Nº 1, 4-27, March 1990.

[NAPA, 91] Narendra K.S. y K. Parthasarathy: “Gradient Methods for the Optimization of Dynamical Systems Containing Neural Networks”. IEEE Transactions on Neural Networks Vol 2, Nº 2, March 1991.

[NARE, 92] Narendra K. S.: “Adaptive control of dynamical systems using neural networks”. Capítulo 5 de “Handbook of Intelligent Control”, ed. D.A. White and D.A. Sofge, Van Nostrand Reinhold, New York, 1992.

[NAWA, 91] Narendra K.S. y K. Wakatsuki: “A comparative study of two neural networks architectures for the identification and control of nonlinear dynamical systems”. Technical Report, Center for Systems Science, Yale University, New Haven, CT, 1991.

[OLLE, 91] Ollero A.: “Control por Computador, descripción interna y diseño óptimo”. Marcombo Boixareu Editores, Barcelona-Mexico, 1991.

[PACH, 94] Parlos, A.G., K.T. Chong y A.F. Atiya: “Application of Recurrent Multilayer Perceptron in Modeling Complex process Dynamics”. IEEE Transactions on Neural Networks, Vol 5, Nº 2, March, 1994.

[PEAR, 89] Pearlmutter B:A: “Learning State Space Trajectories in Recurrent Neural Networks”. Neural Computation 1, 263-269, 1989.

[PEDR, 93] Pedrycz W.: “Fuzzy Control and Fuzzy Systems”, 2ª edición. Research Studies Press Ltd., England, 1993.

[PEYO, 94] Percy P.C. Yip y Yoh-Han Pao: “A Recurrent Neural Net Approach to One-Step Ahead Control Problems”. IEEE Transactions on Systems, Man and Cybernetics, Vol 24, 4, 678-682, 1994.

[POIO, 91] Polycarpou M.M. y P.A. Ioannou: “Identification and Control of Nonlinear Systems Using Neural Network Models: Design and Stability Analysis”. Report 91-09-01, September 1991.

[POLY, 96] Polycarpou M.M.: “Stable Adaptive Neural Control Scheme for Nonlinear Systems”. IEEE Transactions on Automatic Control, Vol 41, 3, 1996.

[PRKA, 94] Pröll T. Y M N. Karim: “Model-Predictive pH Control Using Real-time NARX Approach”. AIChE Journal, Vol. 40, Nº 2, February 1994.

[PSSI, 88] Psaltis D., A. Sideris y A.A. Yamamura: “A multi-layered neural network controller”. IEEE Control Systems Mag.,17-20, April, 1988.

[PSUN, 92] Psichogios D.C. y L. H. Ungar: “A Hybrid Neural Network-First Principles Approach to Process Modelling”. AIChE Journal, Vol 38, Nº 10, 1992.

[PUFE, 94] Puskoris G. V. y L. Feldkamp: “Neurocontrol of Nonlinear Dynamical Systems with Filter Trained Recurrent Networks”. IEEE Trans. on Neural Networks, vol 5, Nº 2, 1994.

[ROFA, 87] Robison A.J. y F. Fallside: “The utility driven dynamic error propagation network”. Technical Report CUED/F-INFENG/TR.1, Cambridge University Engineering Department, Cambridge, England, 1987.

[ROSE, 58] Rosenblatt F.: “The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain”. Psycholog. Rev., 65, 386-108, 1958.

[RUHI, 86] Rumelhart D., G. Hinton y R.J. Williams: “Learning Internal Representations by Error Propagation”. Parallel Distributed Processing, MIT Press, 1986.

[SEED, 86] Serborg D.E., T.F. Edgar y S.L. Shah: “Adaptive Control Strategies for Process Control: A Survey”. AIChE Journal, 32, 6, 881-913, 1986.

[SHKU, 90] Shearer J. L. y B.T. Kulakowski: “Dynamic Modelling and Control of Engineering Systems”. Macmillan Publishing Company, NY, 1990.

[SOKR, 94] Soroush M. y C. Kravaris: “Nonlinear Control of a Polymeritazion CSTR with Singular Characteristic Matrix”. AIChE Journal, Vol 40, Nº 6, 1994.

[SONT, 79] Sontang E.D.: “Polynomial response maps. Lecture Notes in Control and Information Sciences”. Springer_Verlag, Berlín 1979.

[SOWE, 81] Solis, F.J. y J.B. Wets: “Minimization by random search techniques”. Mathematics of Operations Research, 6, 19-30, 1981.

[SRPR, 94] Srinivasan B., U.R. Prasad y N.J. Rao: “Back Propagation Through Adjoints for the Identification of Nonlinear Dynamic Systems Using Recurrent Neural Models”. IEEE Transactions on Neural Networks, Vol 5 Nº 2, 1994.

[STON, 48] Stone M.H.: “The generalized Weierstrass approximation theorem”. Mathematics Magazine, 21:167-184, 237-254, 1948.

[SUGE, 85] Sugeno M.: “An introduction survey of fuzzy control”. Information Sciences 36, 59-83, 1985.

[SUGE, 85] Sugeno M. (ed): “Industrial Applications of Fuzzy Control”. Elsevier Science Publishers BV, The Netherlands, 1985.

[SUYU, 94] Sunil Elanayar V.T. y Yung C. Shin: “Radial Basis Function Neural Network for Approximation and Estimation of Nonlinear Stochastic Dynamic Systems”. IEEE Transactons on Neural Networks, Vol 5, 4, 594-603, 1994.

[TAOM, 92] Tanomaru, J. y S. Omatu: “Process Control by On-line Trained Neural Controllers”. IEEE Transactions on Industrial Electronics, Vol 39, Nº 6, December, 1992.

[THGR, 91] Thibault J. y B.P.A. Grandjean: “Neural Networks in Process Control-A Survey”. IFAC Advanced Control of Chemical processes, Toulouse, France, 1991.

[TOTA, 95] To L.C., M.O. Tadé y G.P. Page: “Nonlinear Control of a simulated industrial evaporation process”. J. Proc. Cont. Vol 5, Nº 3, 173-182, 1995.

[TSBI, 92] Tsang K.M. y S.A. Billings: “Identification of multi-class and nonlinear systems”. Int. J. of Control, Vol 23, Nº 10, 1603-1636, 1992.

[VEPA, 94] Venugopal K.P., A.S. Pandya y R. Sudakar: “A Recurrent Neural Network Controller and Learning Algorithm for the On-Line Learning Control of Autonomous Underwater Vehicles”. Neural Networks Vol 7, Nº 5, 833-846, 1994.

[WABE, 96] Wan E. A. y F. Beaufays.: “Diagrammatic Derivation of Gradient Algorithms for Neural Networks”. Neural Computation, 8, 1, 182-201, 1996.

[WAYN, 91] Wayne Bequete B.: “Nonlinear Control of Chemical Process: A Review”. Ind. Eng. Chem. Res. 30, 1391-1413, 1991.

[WERB, 90] Werbos P.J.: “Backpropagation Through Time: What it does and how to do it”. Proceedings of the IEEE, vol. 78, nº10, October, 1990.

[WERB, 90] Werbos P. J.: “A menu of designs for reinforcement learning over time”. “Neural Networks for Control”, W.T. Miller III, R.S. Sutton and P.J. Werbos (eds). Cambridge, MA: The MIT press, 67-95, 1990.

[WIDR, 86] Widrow B.: “Adaptive Inverse Control”. Proceedings of the Second IFAC Workshop on Adaptive Systems in Control and Signal Processing, 1-5, Lund, Sweden: Lund Institute of Technology, 1986.

[WIHO, 60] Widrow B. y M. Hoff: “Adaptive Switching Circuits”. IRE WESCON Conv. Record, Part 4, 96-104, 1960.

[WIMC, 78] Widrow B., J. McCool y B. Medoff: “Adaptive control by inverse Modelling”. Twelfth Asimolar Conference on Circuits, Systems and Computers, 1978.

[WIMO, 91] Willis M.J., G.A. Montague, C. Massimo, M.T. Tham y A.J. Morris: “Artificial Neural Network Based Predictive Control”. IFAC Advanced Control of Chemical Processes, Toulouse France, 1991.

[WISM, 64] Widrow B. y F.W. Smith: “Pattern-recognizing control systems”. Computer and Information Sciences (COINS) Proceedings, Washington, D.C.: Spartan, Washington, 1964.

[WITT, 79] Wittenmark B.: “Self-Tuning PID controllers Based on Pole Placement”. Report TFRT-7179, Dept. Auto. Control, Lund Inst. Technol., Lund, Sweden, 1979.

[WIZI, 89] Willians R. J. y D. Zipser: “A Learning Algorithm for Continually Running Fully Recurrent Neural Networks”. Neural Computation 1, 270-280, 1989.

[YONI, 93] Yong You y M. Nikolaou: “Dynamic Process Modeling with Recurrent Neural Networks”. AIChE Journal, Vol 39, Nº10, 1993.

[YUFE, 96] Yuan F., Lee A. Feldkamp y G.V. Puskoris: “Neural Networks in Automatic Control: Series Hybrid Electric Vehicle”. Journal on Automatic Control, Vol 37, Nº3, 1996.

[ZADE, 65] Zadeh L.A.: “Fuzzy sets”. Information and Control 8, 330-353, 1965.

[ZALD, 95] Zaldívar J.M.: “Mathematical Modelling and Numerical Simulation of Aromatic Nitrations by Mixed Acid in Discontinuous Reactors”. Joint Research Centre, European Commission, 1995.