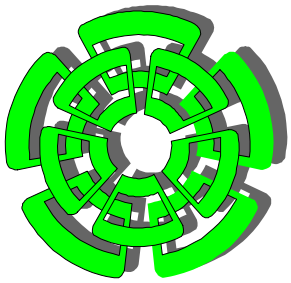

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

DEPARTAMENTO DE CONTROL AUTOMÁTICO

Modelado y control para una clase de sistemas no lineales desconocidos en tiempo discreto

TESIS QUE PRESENTA EL:

M. en C. José de Jesús Rubio Avila

Para obtener el grado de
Doctor en Ciencias en la Especialidad de Control Automático

DIRECTOR DE TESIS:
Dr. Wen Yu Liu

México, D.F., 2007

AGRADECIMIENTO

Agradezco al Consejo Nacional de Ciencia y Tecnología el apoyo brindado para cursar los estudios de Doctorado en el Centro de Investigación y de Estudios Avanzados del IPN, a través de la beca que me fue concedida durante el transcurso de los mismos.

Índice general

Índice de figuras

Capítulo 1

Introducción

El conocimiento total del modelo de una planta en general es muy difícil tenerlo, ya que al paso del tiempo, las condiciones de operación o climáticas y otros factores del entorno de la planta pueden ocasionar desgaste o degradación de sus componentes, por lo que sus parámetros pueden cambiar. De manera que si se construye un modelo de la planta antes de que suceda algún deterioro a lo largo del tiempo, seguramente se esta enfrentando a un problema en el que no se conocen completamente los parámetros del modelo, a esta falta de conocimiento sobre el modelo de la planta se le llama incertidumbre paramétrica.

A su vez, es posible que en el modelo que se este construyendo no se hayan tomado en cuenta por alguna razón ciertas dinámicas de la planta, en este caso la estructura real de la planta tendrá diferencia con respecto a la estructura del modelo matemático, a esta falta de información del modelo de la planta se le conoce como incertidumbre estructural.

Por esta razón, un problema interesante surge cuando alguna parte o toda la dinámica del sistema no es conocida. Por esto, el diseño de controladores robustos que enfrentan esta falta de información es un tema importante. Esto conduce a entender que un sistema que no ha sido hecho a la medida y este sistema siempre tiene incertidumbre estructural o paramétrica. Es decir, es mas fácil que un ingeniero de control se enfrente a controlar una planta con incertidumbre que a una que tenga un modelo exacto.

Por esta razón, una manera natural de modelar las plantas que tienen incertidumbre,

es considerando un modelo que tenga términos nominales o valores centrales conocidos mas otra parte no modelada o incierta.

Este punto de vista es utilizado como una forma de enfrentar los sistemas no lineales con incertidumbre en la presente tesis, debido a que la incertidumbre es parcial algunos autores nombran a este sistema como caja gris, en lugar de caja negra que es la situación en la que hay una incertidumbre total del sistema y sólo se tiene lectura de la salida e introducción de señales en la entrada.

Resultados recientes muestran que la técnica de redes neuronales es muy efectiva para identificar una gran cantidad de sistemas no lineales complejos cuando no se tiene información completa del modelo. Las redes neuronales se clasifican en redes neuronales con conexión hacia adelante y redes neuronales recurrentes [28]. Aunque el algoritmo de propagación hacia atrás se ha usado ampliamente como método de entrenamiento para redes neuronales, la limitación es que este algoritmo tiene una convergencia de parámetros muy lenta. La estabilidad de un algoritmo de gradiente descendiente modificado se da en [93].

El método de modos deslizantes tradicional es la aplicación de una señal de control conmutando a alta frecuencia que consigue llevar el estado del sistema a una superficie cercana al origen denominada superficie de deslizamiento o función de conmutación y una vez llegando a la superficie de deslizamiento se debe permanecer en ésta ante posibles perturbaciones externas. Dicha superficie de deslizamiento debe ser definida por el diseñador con el objeto de que el estado cumpla las especificaciones deseadas. El control por modos deslizantes en tiempo continuo genera castañeo debido a la discontinuidad en la acción de control y el castañeo es indeseable en la mayoría de las aplicaciones [11].

1.1. Estado del arte

1.1.1. Redes neuronales

En 1957, Frank Rosenblat comenzó el desarrollo del perceptrón, el perceptrón es la red neuronal más antigua. En 1959 Bernard Widrow y Marcial Hoff desarrollaron el modelo de

elementos lineales adaptables, esta fue la primera red neuronal aplicada a un problema real (filtro adaptable para eliminar ecos en las líneas telefónicas). En 1967 Stephen Grossberg realizaron una red avalancha que consistía en elementos discretos con actividad que varia con el tiempo que satisface ecuaciones diferenciales continuas para resolver actividades como reconocimiento del habla y el aprendizaje del movimiento de los brazos de un robot. En 1982 Marvin Minsky y Seymour Papert publicaron el libro llamado *Perceptrons*. En 1977 Teuvo Kohonen desarrolló un modelo de red neuronal para el reconocimiento de patrones visuales [43]. En 1982 John Hopfield presentó una red que le ha dado su nombre [30]. En 1985 se formó la *International Neural Network Society* (*INNS*) bajo la iniciativa y dirección de Grosberg en Estados Unidos, Kohonen en Finlandia y Amari en Japón. En 1987 la *IEEE* celebró la primer conferencia sobre redes neuronales. En 1988 se unieron la *INNS* y la *IEEE* para formar la llamada *International Joint Conference on Neural Networks* (*IJCNN*) [29].

1.1.2. Algoritmo elipsoidal

En 1979 L.G.Khachiyan indicó como un método elipsoidal se puede implementar para programación lineal [3]. Este resultado ha causado gran interés y ha estimulado a la investigación en este tema. La técnica elipsoidal tiene importantes ventajas en la estimación de estados con perturbaciones acotadas [21]. Hay muchas aplicaciones potenciales para problemas semejantes a la programación lineal. [89] obtuvo elipsoides los cuales eran validos para un número finito de datos. [69] presentó una programación elipsoidal tal que el nuevo elipsoide satisface una relación afín con otro elipsoide. En [12], el algoritmo elipsoidal se usa como una técnica de optimización que toma en cuenta las restricciones sobre coeficientes agrupados. [52] describió a detalle varios métodos que pueden ser usados para derivar una incertidumbre elipsoidal apropiada. En [57], se considera el problema elipsoidal con comportamiento asintótico para sistemas lineales en tiempo discreto. Hay pocas publicaciones de elipsoides en combinación con redes neuronales. En [15] se usan aprendizajes supervisado y no supervisado en forma elipsoidal para encontrar y sintonizar las reglas de funciones difusas para un proceso de identificación difuso. En [42] se proponen funciones de activación de tipo

elipsoidal para redes neuronales con conexión hacia adelante.

1.1.3. Control por modos deslizantes en tiempo discreto

En 1987 Sarpturk fue el primero que consideró el control por modos deslizantes en tiempo discreto al cual era un control de estructura variable [77]. En 1999 Sira y Ramirez trataron las generalidades de los modos deslizantes en tiempo discreto [82]. En 1999 Fang diseñó un control por modos deslizantes con redes neuronales recurrentes estable para el problema de regulación [19]. En 2000 Muñóz propuso un controlador en modos deslizantes el cual usa una superficie de deslizamiento que produce un bajo castañeo en la señal de control, este usa un termino adaptable y una red neuronal para la modelado del error de modelado y la cancelación de este, éste control satisface las condiciones necesarias pero no las suficientes que aseguran la estabilidad del sistema en lazo cerrado [55].

1.1.4. Control con redes neuronales

En 1990 Narendra fue el primero en tratar el control con redes neuronales en tiempo discreto [56]. En 1995 Chen y Khalil diseñaron un control con redes neuronales estable para sistemas afines para el problema de seguimiento de trayectoria [8]. En 1999 Cabrera y Narendra propusieron controladores por redes neuronales basados en el control inverso [5]. En 2000 Jagannathan propuso un control por redes neuronales para una clase de sistemas desconocidos para el cual garantiza su estabilidad [38]. En 2004 Ge, Zhang y Lee propusieron un controlador con redes neuronales para una clase de sistemas con múltiples entradas y múltiples salidas [24].

1.2. Motivación del tema de tesis

1.2.1. Tiempo discreto

Para el diseño de algoritmos de control en tiempo continuo se requiere usar Simulink o LabView y una tarjeta de adquisición de datos especial lo cual muchas veces no esta al alcance de cualquier persona o institución. Para el diseño de algoritmos de control en tiempo discreto se tienen ecuaciones de diferencia las cuales se pueden pasar directamente al sistema real a través de un programa en C, en Visual Basic o en Ensamblador y una tarjeta de adquisición de datos diseñada por el usuario con lo cual los costos están al alcance de prácticamente cualquier persona o institución. Esta situación ha favorecido el diseño de algoritmos de control a partir de un modelo discretizado, o más aún cuando el proceso original opera en tiempo discreto como es el caso de este documento. El trabajo a realizar en el control en tiempo discreto es extender, hasta donde sea posible, la teoría de control existente para sistemas no lineales en tiempo continuo, al caso de sistemas en tiempo discreto. En el caso de sistemas lineales invariantes en tiempo, existe un paralelismo casi perfecto entre los sistemas en tiempo continuo y en tiempo discreto. Desafortunadamente, en el caso no lineal la situación es mucho más compleja.

1.2.2. Redes neuronales

Ocurre a menudo en ingeniería química, como en muchas otras disciplinas, que la principal, y a veces única, información de que se dispone de un proceso es una serie numérica temporal. Dicha serie se corresponde con los valores de una función, de una o más variables, que contiene la dinámica interna del sistema. El problema radica entonces en la identificación de la función generadora de la serie de valores, esta tarea se ha realizado ampliamente con las redes neuronales [28], [29].

1.2.3. Algoritmo elipsoidal

Comparado al algoritmo de aprendizaje de propagación hacia atrás, el algoritmo elipsoidal acotado tiene una convergencia de parámetros más rápida, ya que este es similar en estructura al algoritmo del filtro de Kalman extendido [70], pero en el algoritmo elipsoidal no se requiere que la incertidumbre estructural se modele como un proceso Gaussiano como en el algoritmo del filtro de Kalman extendido, solo se requiere que el error de la incertidumbre estructural esté acotado.

1.2.4. Modos deslizantes

La principal ventaja del control por modos deslizantes es que aporta robustez ante perturbaciones cuando estas tienen cotas conocidas. El control por modos deslizantes en tiempo discreto se tiene la ventaja de que puede o no presentarse castañeo a diferencia del control por modos deslizantes en tiempo continuo en el que siempre se presenta castañeo [82]. Si se desconoce la planta, se pueden usar redes neuronales para modelar la planta [8], [55].

Para el siguiente sistema:

$$y(k+1) = f(z(k)) + u(k) \tag{1.1}$$

con $z(k) = y(k-1), \ldots, y(k-n), u(k-1), \ldots, u(k-m)$. Se tienen los siguientes casos para el control.

Caso 1: Usar modos deslizantes para controlar (1.1). Se propone la ley de control $u(k) = -K_1 sign(e_c(k)) + y_m(k+1)$ donde $e_c(k) = y_m(k+1) - y(k+1)$, $y_m(k+1)$ es la salida de referencia, con $K_1 \geq \overline{f}$, $|f(z(k))| < \overline{f}$,, se asegura que es estable el controlador, en este caso se genera castañeo con magnitud K_1 que puede ser grande.

Caso 2. Usar redes neuronales. Si se usan redes neuronales para aproximar $f(z(k))$ como $\widehat{f}(z(k))$ en línea con el controlador, se sabe que se tendrá un error de identificación $e_i(k) = \widehat{y}(k+1) - y(k+1)$, $\widehat{y}(k+1)$ es la salida de la planta modelada, entonces si se usa la ley de control $u(k) = -\widehat{f}(z(k)) + y_m(k+1)$ siempre persistirá el error de modelado $e_i(k)$ donde $e_i(k) \ll \overline{f}$, pero no se asegura la estabilidad del controlador.

Caso 3. Usar modos deslizantes y redes neuronales. Si se usa la ley de control $u(k) = -\widehat{f}(z(k)) - K_2 sign(ec(k)) + y_m(k+1)$ con $K_2 \geq \overline{e}$, $|e_i(k)| < \overline{e}$, si se asegura la estabilidad del controlador y además este control genera menos castañeo debido a que $K_2 \ll K_1$ ya que $e_i(k) \ll \overline{f}$.

1.3. Objetivo de la tesis

El primer objetivo de esta tesis es proponer un algoritmo de elipsoidal acotado para el entrenamiento de una red neuronal recurrente, comparando su comportamiento con el del algoritmo de propagación hacia atrás.

El segundo objetivo de esta tesis es proponer un control neuronal por modos deslizantes en tiempo discreto con ganancia variante en tiempo, comparando su comportamiento con el del control neuronal por modos deslizantes con ganancia fija y el del control neuronal por retroalimentación para controlar una clase especial de sistemas no lineales en tiempo discreto.

1.4. Contenido de la tesis

- En el capítulo 2 se presenta la teoría general para el control en tiempo discreto aplicado a una planta de naturaleza discreta o aplicado a una planta discretizada y sus métodos de discretización, después se presenta la teoría general para el control en tiempo discreto de una planta en tiempo continuo. Después se presenta un panorama general de las redes neuronales así como la descripción de las estructuras de identificación empleadas para control y el algoritmo de propagación hacia atrás. Posteriormente se presentan algunas definiciones espacios elipsoidales acotados. Finalmente, se presentan las características el control por modos deslizantes en tiempo discreto.

- En el capítulo 3 se propone un algoritmo elipsoidal acotado el cual se usa para el entrenamiento de una red neuronal recurrente para la identificación de sistemas no lineales.

Se modifica el algoritmo elipsoidal acotado tal que se garantice que se tiene un elipsoide el cual acote la intersección de dos espacios elipsoidales en cada iteración. Se actualizan capa oculta y capa de salida. Se prueba la estabilidad del error de identificación para el algoritmo elipsoidal acotado usando una técnica semejante a la técnica de Lyapunov. Desde el punto de vista de sistemas dinámicos, tal entrenamiento es útil para todas las aplicaciones de redes neuronales que requieren la actualización de los pesos en línea.

- En el capítulo 4 se presenta un control por modos deslizantes con un aproximador de funciones neuronal. El aproximador de funciones neuronal usa doble zona muerta y una modificación para asegurar la estabilidad del error de modelado y la no singularidad del controlador. Con el fin de reducir el castañeo, se modifica el control por modos deslizantes en tiempo discreto utilizando una ganancia variante en tiempo. Se da una condición necesaria para la existencia de la función de conmutación en tiempo discreto, la cual satisface la condición $|s(k)| \leq q$ donde $q > 0$. Se prueba que el sistema en lazo cerrado del control por modos deslizantes y el aproximador de funciones por redes neuronales es uniformemente estable y que el error de seguimiento de trayectoria está acotado y la cota de este error depende de la cota del error de la incertidumbre estructural.

- En el capítulo 5 se presentan las conclusiones finales de esta tesis y el trabajo a futuro.

Capítulo 2

Modelado y control con redes neuronales en tiempo discreto

Este capítulo se presenta se presenta la teoría general para el control en tiempo discreto aplicado a una planta de naturaleza discreta o aplicado a una planta discretizada y sus métodos de discretización, después se presenta la teoría general para el control en tiempo discreto de una planta en tiempo continuo. Después se presenta un panorama general de las redes neuronales así como la descripción de las estructuras de identificación empleadas para control y el algoritmo de propagación hacia atrás. Posteriormente se presentan algunas definiciones espacios elipsoidales acotados. Finalmente, se presentan las características el control por modos deslizantes en tiempo discreto.

2.1. Control discreto para plantas en tiempo continuo

2.1.1. Sistema de control por computadora

Un lazo de control de proceso digital variable sencillo con un proceso de tiempo continuo, sistema de datos muestreados una entrada una salida, tendrá la configuración que se muestra en la Figura 2.1. Los subsistemas de actuación y medición pueden considerarse incorporados

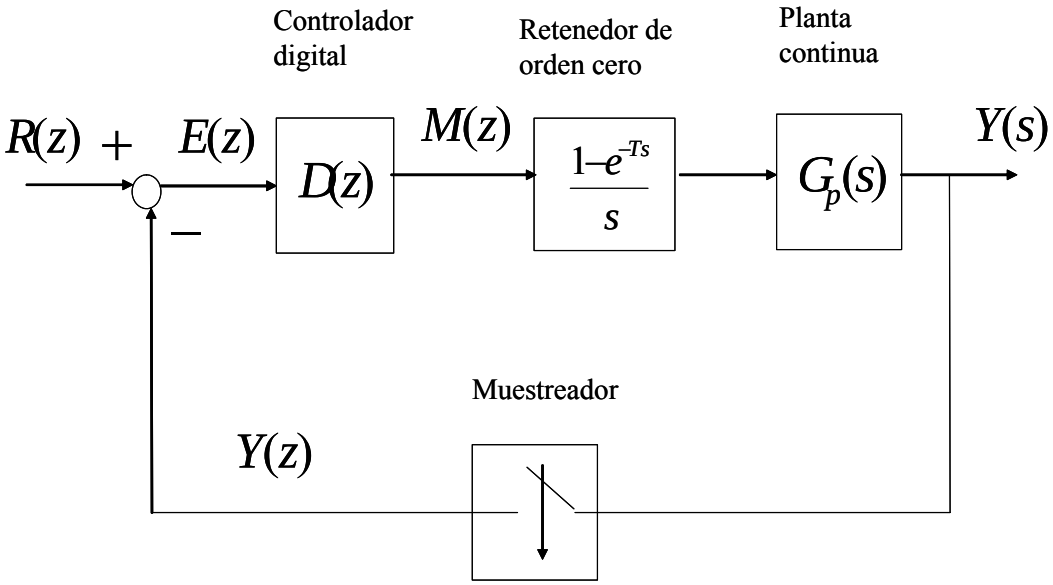

Figura 2.1: Sistema de control por computadora

en la planta, de modo que la estructura que se aprecia en la Figura es general. La salida continua de planta $y(t)$ debe convertirse en datos de tiempo discreto por muestreo en puntos discretos en el tiempo. De manera física, el muestreo se lleva normalmente usando un dispositivo convertidor analógico digital y un programa de muestreo de computadora digital, se da por hecho que el muestreo se lleva a cabo a intervalos iguales de tiempo, es decir, el periodo de muestreo T. La Figura 2.2 deja ver la relación entre la función continua $y(t)$ y la función muestreada y_k, que es una aproximación de tiempo discreto de la función de tiempo continuo. La función muestreada es también una secuencia atrasada en el tiempo de valores o números $f(kT)$, $k = 0, 1, \ldots$, y tiene una transformada z $y(z)$ [18].

Una serie de números $m_k = m(kT), k = 0, 1, 2, \ldots$, es una descripción limitada de una función continua $m(t)$. En particular, los valores en tiempos diferentes de los instantes de muestreo son indefinidos y deben aproximarse, por lo común mediante alguna técnica de interpolación. La más simple de tales reconstrucciones de $m(t)$ que utiliza la información mínima es una aproximación de primer orden o de Euler de pendiente cero ilustrada en

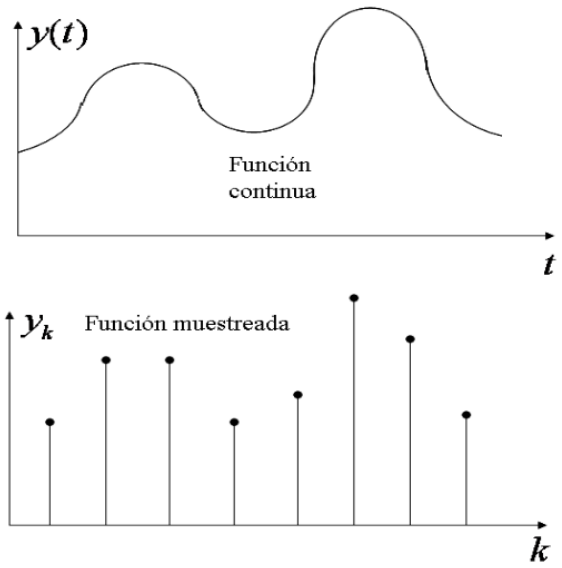

Figura 2.2: Operación del muestreador

la Figura 2.3, a esta operación se le conoce como retenedor de orden cero, es decir, una operación en la que el valor de cada señal muestreada se mantiene constante con pendiente cero sobre el periodo de muestreo de T unidades de tiempo. El resultado es una señal de patrón de escalera como se muestra en la figura 2.3. Físicamente, la conversión de los datos de tiempo discreto en una señal de tiempo continuo se suele llevar a cabo con un dispositivo convertidor digital a analógico [18].

2.2. Control en tiempo discreto

2.2.1. Control para plantas en tiempo discreto

En este documento se trata con este tipo de controles, en este sistema de control no se tiene periodo de muestreo. Ver Figura 2.4. Hay algunas plantas que se encuentran en tiempo discreto:

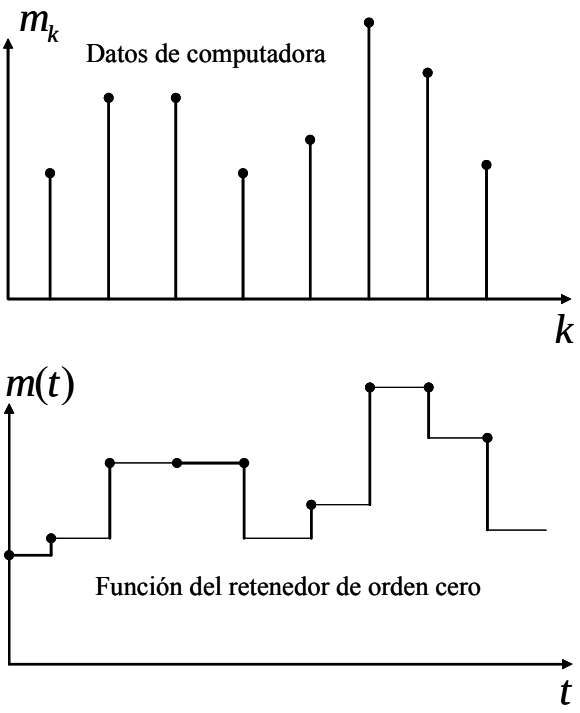

Figura 2.3: Operación del retenedor de orden cero

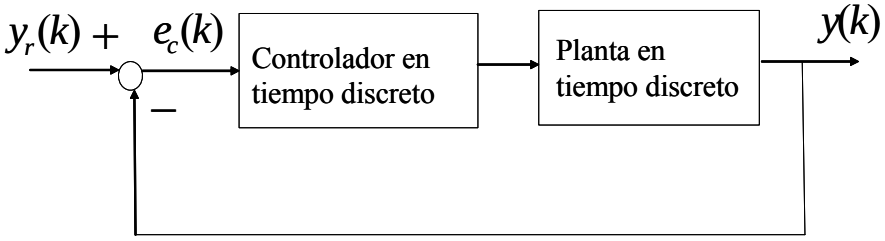

Figura 2.4: Control de una planta en tiempo discreto

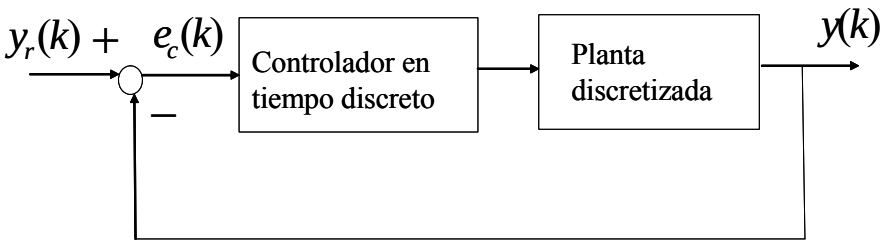

Figura 2.5: Control de una planta discretizada

1. La planta no se obtiene a través de un modelo físico ni a través de ecuaciones diferenciales sino que se obtiene a través de análisis de datos. Por ejemplo: tráfico urbano, tráfico de redes, población, etc.

2. El modelo es una caja negra solamente se pueden tomar datos de entrada y salida para el modelado. Por ejemplo, series de tiempo, sistemas financieros, procesamiento de gas, etc.

Se tratará este caso mas a detalle en las simulaciones de los capítulos 3 y 4.

2.2.2. Control para plantas discretizadas

Este caso se representa en la Figura 2.5. Algunos autores recomiendan que el periodo de muestreo de la planta sea 10 veces más pequeño que el del controlador para que se aproxime el sistema continuo y que el controlador funcione correctamente [18], [60]. Los métodos de discretización considerados en este texto son el método de Euler y el método de Runge-Kuta los cuales se analizan a continuación. Notará que se tienen una planta discretizada diferente para cada periodo de muestreo diferente.

Discretización de sistemas lineales

Primero se va a obtener la representación en tiempo discreto para la siguiente ecuación en espacio de estados lineal en tiempo continuo:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{2.1}$$

donde $x(t) \in \Re^n$ es el vector de estados, $u(t) \in \Re^m$ es el vector de entradas, $A \in \Re^{nxn}$ es una matriz constante, $B \in \Re^{nxm}$ es una matriz constante, $y(t) \in \Re^r$ es el vector de salidas, $C \in \Re^{rxn}$ es una matriz constante y $D \in \Re^{rxm}$ es una matriz constante. La ecuación (2.1) tiene la siguiente solución [60]:

$$x(t) = e^{At}x(0) + e^{At}\int_0^t e^{-A\tau}Bu(\tau)d\tau \tag{2.2}$$

donde $x(0)$ es la condición inicial de $x(t)$. La representación de la ecuación (2.1) en tiempo discreto dada en [60] es:

$$x(k+1) = G(T)x(k) + H(T)u(k) \tag{2.3}$$

Notar que las matrices $G(T)$ y $H(T)$ dependen del periodo de muestreo T, una ves que el periodo de muestreo es fijo, G y H son matrices constantes. Para determinar $G(T)$ y $H(T)$, se usa la ecuación (2.2). Se asume que la entrada $u(t)$ es muestreada y alimentada a un retenedor de orden cero así que todos los componentes de $u(t)$ son constantes en el intervalo entre cualquiera dos instantes consecutivos de muestreo, o:

$$u(t) = u(k) \qquad \text{para } k \leq t < k+1 \tag{2.4}$$

entonces se tiene $G(T)$ y $H(T)$ como sigue [60]:

$$G(T) = e^{AT} \tag{2.5}$$

$$H(T) = \int_0^T e^{Av}dvB = e^{AT}\int_0^T e^{-At}dtB \tag{2.6}$$

La salida de la ecuación (2.1) se convierte en:

$$y(k) = Cx(k) + Du(k) \tag{2.7}$$

donde las matrices C y D son constantes y no dependen del periodo de muestreo. Si la matriz A es no singular y usando el hecho de que $\frac{d}{dt}e^{-At} = -Ae^{-At} = -e^{-At}A$ se tiene para H(T) la siguiente equivalencia:

$$H(T) = e^{AT}\int_0^T e^{-At}dtB = -A^{-1}e^{AT}\left(e^{-AT} - I\right)B = A^{-1}\left(e^{AT} - I\right)B = \left(e^{AT} - I\right)A^{-1}B \tag{2.8}$$

Ejemplo 2.1 *Considere el sistema en tiempo continuo dado a continuación:*

$$\dot{x}(t) = -ax(t) + u(t)$$
$$y(t) = x(t)$$

obtener la representación del sistema en espacio de estados en tiempo discreto.

Usando ecuaciones (2.5) y (2.6) se tiene lo siguiente:

$$G(T) = e^{-aT}$$
$$H(T) = \int_0^T e^{-a\upsilon}d\upsilon = \frac{1-e^{-aT}}{a}$$

por lo tanto el sistema en tiempo discreto es:

$$x\left(k+1\right) = e^{-aT}x(k) + \frac{1-e^{-aT}}{a}u(k)$$
$$y(k) = x(k)$$

Método de Euler para discretización de sistemas no lineales

El problema que se desea resolver es el siguiente: dada la función vectorial $f(x,t)$, encuentre la función vectorial x que sea una solución aproximada a la ecuación ordinaria vectorial:

$$\frac{d}{dt}x = f(x,t)\_\_\_\_x(t_0) = x_0 \tag{2.9}$$

para el intervalo de tiempo $t_0 \leq t \leq t_f$. Considere el intervalo de tiempo $[t_0, t_f]$, dividido en N subintervalos de longitud T, de manera que:

$$t_k = t_0 + kT \tag{2.10}$$

es posible obtener los valores aproximados $x_1, x_2, \ldots, x_N$, para los valores exactos $x(t_1), x(t_2), \ldots, x(t_N)$, utilizando la aproximación directa más sencilla de la derivada en el punto (x_k, t_k) que es el cociente de diferencia:

$$f(x_k, t_k) \approx \frac{1}{T}(x_{k+1} - x_k) \tag{2.11}$$

Esta aproximación da la fórmula de recursión conocida como método de Euler [18]:

$$x_{k+1} = x_k + Tf(x_k, t_k) \_\_\_ t_k = t_0 + kT \tag{2.12}$$

La desventaja de este sencillo método, es que para una precisión razonable la longitud del paso T debe ser muy pequeño.

Ejemplo 2.2 *Considere la ecuación de estado lineal y homogénea de primer orden:*

$$\frac{d}{dt}x = ax \_\_\_ x(0) = x_0 \tag{2.13}$$

La solución exacta:

$$x(t) = x_0 e^{at} \tag{2.14}$$

sabemos que es estable para todo $a < 0$. Sin embargo, la solución aproximada, según el método de Euler, esta determinada por la ecuación de diferencia:

$$x_{k+1} = x_k + Tax_k \tag{2.15}$$

que se escribe como:

$$x_{k+1} = (1 + Ta)x_k = px_k \_\_\_ x_0 \text{ dado} \tag{2.16}$$

El modo para comportamiento de estabilidad es p^k o $(1+Ta)^k$. Así el sistema es inestable para todo $|1 + Ta| > 1$, Esto proporciona la estabilidad para el método de Euler como $-2 < Ta < 0$ dado que $a < 0$.

$$T < \left|\frac{2}{a}\right| \tag{2.17}$$

Método de Runge-Kuta para discretización de sistemas no lineales

En el problema de Runge-Kuta para mejorar la solución de (2.9) consiste en calcular $f(x,t)$ en varios puntos elegidos de manera estratégica cerca de la curva de solución en el intervalo $[t_k, t_k + T]$, y combinar estos valores de tal forma que se obtenga una buena precisión en el incremento calculado $x_{k+1} - x_k$. El método de Runge Kuta de cuarto orden es el siguiente [18]:

$$\begin{aligned} R_1 &= Tf(x_k, t_k) \\ R_2 &= Tf(x_k + \tfrac{1}{2}R_1, t_k + \tfrac{1}{2}T) \\ R_3 &= Tf(x_k + \tfrac{1}{2}R_2, t_k + \tfrac{1}{2}T) \\ R_4 &= Tf(x_k + R_3, t_k + T) \\ x_{k+1} &= x_k + \tfrac{1}{6}\left[R_1 + 2R_2 + 2R_3 + R_4\right] \end{aligned} \tag{2.18}$$

Esta fórmula es de cuarto orden, lo que en esencia significa que el error global de truncado en el polinomio de aproximación del orden T^4. Si bien la función f se calcula cuatro veces en cada paso de tiempo, la programación de las fórmulas de la ecuación (2.18) es relativamente sencilla.

2.3. Redes neuronales para modelado

Una variedad de arquitecturas de redes neuronales para control han sido propuestas para el control de sistemas no lineales con incertidumbre. Una de las principales ventajas de las redes neuronales en el diseño de control no lineal es precisamente la capacidad de construir mapeos no lineales complejos.

Por esto, las metodologías con redes neuronales se pueden considerar para identificar no linealidades desconocidas en un sistema y después utilizar este modelo identificado para construir una ley de control que logre los propósitos de control deseados [28].

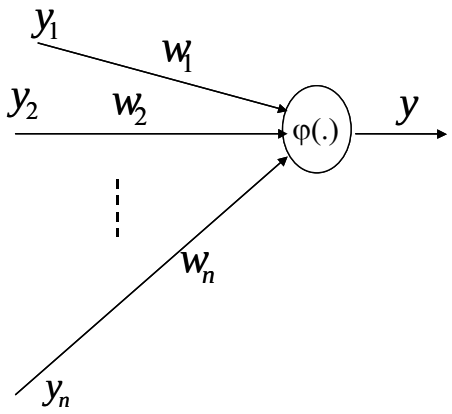

Figura 2.6: Modelo de una neurona

2.3.1. Redes neuronales

Una red neuronal artificial es un elemento capaz de procesar gran cantidad de información de forma paralela y distribuida y esta inspirada en las redes neuronales biológicas, las cuales pueden almacenar conocimiento y tenerlo disponible para su uso. La red neuronal artificial tiene similitudes con el cerebro, como lo son:

- El conocimiento es adquirido a través de un proceso de aprendizaje.
- Las conexiones entre neuronas son llamadas pesos sinápticos y son utilizados para almacenar el conocimiento.

La función del proceso de aprendizaje es modificar los pesos sinápticos de las redes neuronales con el objetivo de minimizar una función de costo. La modificación de los pesos sinápticos es el método tradicional para el diseño y el funcionamiento de las redes neuronales.

La neurona es la unidad fundamental para la operación de la red neuronal. La Figura 2.6 muestra el esquema de una neurona, el cual esta compuesto por:

1. Un conjunto de pesos sinápticos, con cada elemento caracterizado por su propio peso w_i.

2. Un sumador el cual suma los componentes de la señal de entrada, multiplicados por su respectivo peso sináptico definido en 1.

3. Una función de activación no-lineal $\varphi(\cdot)$ que transforma la salida del sumador en la salida y.

En términos matemáticos, la neurona se puede describir como:

$$v = \sum_{j=1}^{n} w_i x_i$$

$$y = \varphi(v) \tag{2.19}$$

donde:

y_i i ésima componente de la entrada.

w_i peso de la conexión entre la i-ésima componente de la entrada y la neurona.

v salida del sumador.

$\varphi(\cdot)$ función de activación no lineal.

y salida neurona.

La función de activación no-lineal denotada por $\varphi(\cdot)$, recibe como entrada v y genera el elemento de la salida y. La función de activación $\varphi(\cdot)$ tiene se clasifica en:

1. Diferenciable y no diferenciable. Las funciones de activación diferenciables (por ejemplo la función sigmoidal o la función tangente hiperbólica) son necesarias para algunos algoritmos de adaptación como es el caso del algoritmo de propagación hacia atrás, mientras que las funciones discontinuas (por ejemplo la función escalón unitario o la función signo) se emplean para generar una salida binaria y comúnmente se emplea en el perceptrón de una sola capa.

2. Positiva y de promedio cero. Para las funciones diferenciables: las funciones positivas (por ejemplo la función sigmoidal) pueden tomar cualquier valor entre 0 y 1 y las funciones de promedio cero (por ejemplo la función tangente hiperbólica) pueden tomar cualquier valor entre -1 y 1. Para las funciones discontinuas: las funciones positivas

(por ejemplo la función escalón unitario) pueden tomar los valores de 0 y 1 y las funciones de promedio cero (por ejemplo la función signo) pueden tomar los valores de -1 y 1.

La forma como se interconectan las neuronas de una red neuronal determina su estructura. Para propósitos de identificación y control, las estructuras más usadas son:

1. Red con conexión hacia adelante de una capa también llamada perceptrón de una sola capa.

2. Red con conexión hacia adelante multicapa también llamada perceptrón multicapa.

3. Red con funciones de base radial.

Las redes neuronales anteriores se les conoce como redes neuronales con conexión hacia adelante o redes neuronales estáticas.

Redes neuronales con conexión hacia adelante multicapa

La red neuronal con conexión hacia adelante multicapa también llamada perceptrón multicapa se distingue por la presencia de una o más capas ocultas, (ver Figura 2.7). Típicamente, las neuronas en cada capa tienen como señales de entrada las señales de salida de la capa precedente. Si cada neurona en cada capa es conectada con todas las neuronas de las capas adyacentes, entonces la red neuronal es llamada totalmente conectada, en el caso opuesto es llamada parcialmente conectada [28], [29].

Las ecuaciones que describen el perceptrón multicapa son:

$$y_j = \sum_{i=1}^{n} w_{ij} y_i \quad \text{para } j = 1, 2, \ldots q$$
$$y_i = \varphi_i\left(v_i\right), \quad v_i = \sum_{k=1}^{m} w_{ki} y_k \quad \text{para } i = 1, 2, \ldots n$$

El perceptrón multicapa tiene las siguientes características [28], [29]:

1. La función de activación para cada neurona debe ser diferenciable a diferencia a la función de activación discontinua la cual se puede usar en el perceptrón de una sola capa. Comúnmente, la función no-lineal empleada es la sigmoide, definida como sigue:

$$\varphi_i(v_i) = \frac{1}{1 + e^{-v_i}}$$

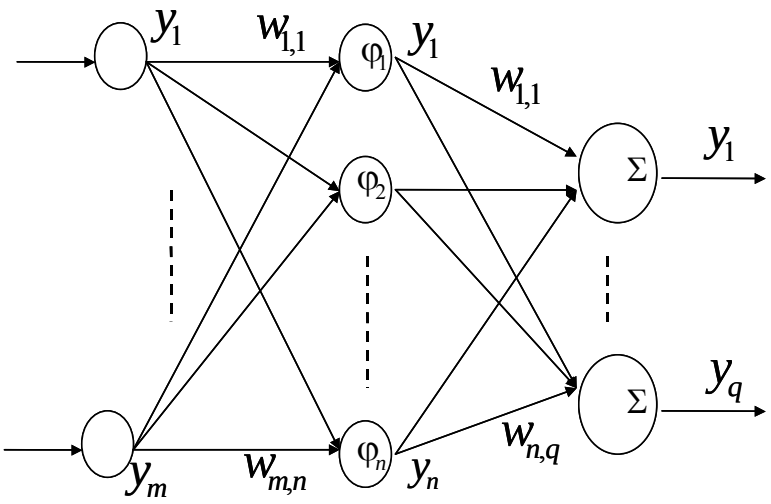

Figura 2.7: Perceptrón multicapa

2. La red esta compuesta por una o más capas ocultas.

3. La red presenta un alto grado de conectividad.

El perceptrón multicapa obtiene su poder computacional a través de la combinación de estas características y su habilidad de aprender de la experiencia. Sin embargo, la presencia de no linealidades distribuidas y la alta conectividad de la red complican su análisis teórico.

2.3.2. Algoritmo de propagación hacia atrás en tiempo discreto

El algoritmo de aprendizaje usado para ajustar los pesos del perceptrón multicapa es conocido como el algoritmo de propagación hacia atrás. En la Figura 2.8 se muestra una parte del perceptrón multicapa, donde d_j es la salida deseada, y_i es la i ésima neurona de salida, i, j y k indican neuronas, w_{ij} son los pesos entre la neurona i y la neurona j. $\varphi_i(\cdot)$ es la función de activación no lineal para la i ésima neurona.

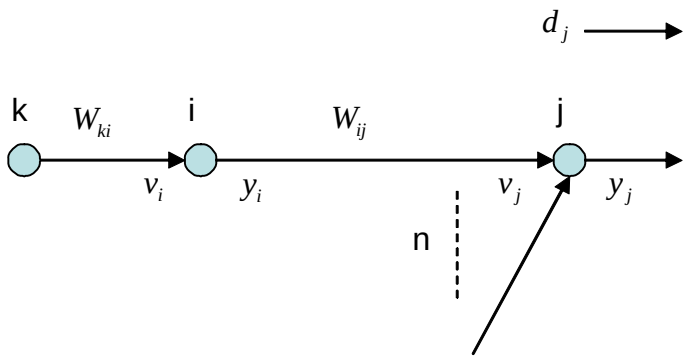

Figura 2.8: Algoritmo de propagación hacia atrás

La salida del perceptrón multicapa es:

$$y_j = \varphi_j(v_j), \quad v_j = \sum_{i=1}^{n} w_{ij} y_i$$

El error de la neurona de salida j esta dado como sigue:

$$e_j(k) = y_j(k) - d_j(k)$$

La suma instantánea de la suma de los errores de salida cuadrados es:

$$\varepsilon(k) = \frac{1}{2} \sum_{j=1}^{q} e_j^2(k)$$

donde q es el número de neuronas en la capa de salida. Usando el gradiente descendente, se actualizan los pesos que conectan las neuronas i y j como sigue:

$$w_{ij}(k+1) = w_{ij}(k) - \eta \frac{\partial \varepsilon(k)}{\partial w_{ij}(k)}$$

donde η es el coeficiente de aprendizaje. El término $\frac{\partial \varepsilon(k)}{\partial w_{ij}(k)}$ se calcula como sigue:

$$\frac{\partial \varepsilon(k)}{\partial w_{ij}(k)} = \frac{\partial \varepsilon(k)}{\partial e_j(k)} \frac{\partial e_j(k)}{\partial y_j(k)} \frac{\partial y_j(k)}{\partial v_j(k)} \frac{\partial v_j(k)}{\partial w_{ij}(k)}$$

Las derivadas parciales son $\frac{\partial \varepsilon(k)}{\partial e_j(k)} = e_j(k)$, $\frac{\partial e_j(k)}{\partial y_j(k)} = 1$, $\frac{\partial y_j(k)}{\partial v_j(k)} = \varphi_j'\left[v_j(k)\right]$, $\frac{\partial v_j(k)}{\partial w_{ji}(k)} = y_i(k)$. Comúnmente se usa una función lineal en la capa de salida, así:

$$w_{ij}(k+1) = w_{ij}(k) - \eta y_i(k) e_j(k) \tag{2.20}$$

Similarmente w_{ki} se calcula como sigue:

$$w_{ki}(k+1) = w_{ki}(k) - \eta \frac{\partial \varepsilon(k)}{\partial w_{ki}(k)}$$

donde:

$$y_i = \varphi_i(v_i), \quad v_i = \sum_{k=1}^{m} w_{ki} y_k$$

el término $\frac{\partial \varepsilon(k)}{\partial w_{ki}(k)}$ se calcula como sigue:

$$\begin{aligned}
&\frac{\partial \varepsilon(k)}{\partial w_{ki}(k)} = \frac{\partial \varepsilon(k)}{\partial e_j(k)} \frac{\partial e_j(k)}{\partial y_j(k)} \frac{\partial y_j(k)}{\partial v_j(k)} \frac{\partial v_j(k)}{\partial y_i(k)} \frac{\partial y_i(k)}{\partial v_i(k)} \frac{\partial v_i(k)}{\partial w_{ki}(k)} = \\
&\left\{ e_j(k) \varphi_j'\left[v_j(k)\right] w_{ij}(k) \right\} \varphi_i'\left[v_i(k)\right] y_k(k) = \\
&\left\{ e_j(k) w_{ij}(k) \right\} \varphi_i'\left[v_i(k)\right] y_k(k) = e_i(k) \varphi_i'\left[v_i(k)\right] y_k(k)
\end{aligned}$$

donde $e_i(k) = e_j(k) w_{ij}(k)$. Así

$$w_{ki}(k+1) = w_{ki}(k) - \eta y_k(k) e_i(k) \varphi_i'\left[v_i(k)\right] \tag{2.21}$$

donde $e_i(k) \varphi_i'\left[v_i(k)\right]$ es el error de propagación hacia atrás.

El algoritmo de propagación hacia atrás se ha convertido en el algoritmo más popular para el entrenamiento del perceptrón multicapa [74]. Este es fácil de calcular y esta habilitado para clasificar información no lineal separable. El algoritmo de propagación hacia atrás usa la técnica del gradiente, la técnica del gradiente en cada paso busca el mínimo de una función de costo del error de identificación, este podría ser un mínimo local y no se tendría optimización, así que no es posible demostrar la convergencia del error de identificación hacia un mínimo global.

La investigación de un algoritmo más rápido al algoritmo de propagación hacia atrás para entrenar lo pesos de la red neuronal considera dos categorías. La primera categoría considera

técnicas heurísticas dentro de las cuales incluye considerar un coeficiente de aprendizaje variante en tiempo, usar el momento o usar variables de escalamiento [28]. En la otra categoría de investigación se han considerado técnicas de optimización en donde la red neuronal con conexión hacia adelante se convierte en un problema de optimización.

2.3.3. Estructuras de identificación que usan las redes neuronales

A continuación se presentan 4 estructuras para la representación de plantas con una entrada y una salida, éstas se pueden generalizar para el caso multivariable. Estas estructuras se han usado en sistemas adaptables para la identificación y para el control de sistemas no lineales. Las cuatro estructuras para plantas en tiempo discreto se describen mediante las siguientes ecuaciones de diferencias:

$$\begin{aligned}
&\text{Estructura I} && y(k+1) = \sum_{i=0}^{n-1} \alpha_i y(k-i) + g\left[u(k), u(k-1), \ldots, u(k-m+1)\right] \\
&\text{Estructura II} && y(k+1) = f\left[y(k), y(k-1), \ldots, y(k-n+1)\right] + \sum_{j=0}^{m-1} \beta_j u(k-j) \\
&\text{Estructura III} && y(k+1) = f\left[y(k), y(k-1), \ldots, y(k-n+1)\right] + g\left[u(k), u(k-1), \ldots, u(k-m+1)\right] \\
&\text{Estructura IV} && y(k+1) = f\left[y(k), y(k-1), \ldots, y(k-n+1); u(k), u(k-1), \ldots, u(k-m+1)\right]
\end{aligned} \tag{2.22}$$

donde $\alpha_i \in \Re$ para $i = 0 \ldots n$ y $\beta_j \in \Re$ para $j = 0 \ldots m$ son parámetros constantes desconocidos, $m \leq n$, $u(k) \in \Re$ es la entrada de la planta y $y(k)$ es la salida de la planta, $f[\cdot]$, $g[\cdot] \in \Re$ son funciones no lineales de las cuales se desconoce su comportamiento.

Una aproximación obvia del método del sistema es seleccionar la estructura de la red neuronal que sea la misma que tiene la planta.

A continuación se muestran las estructuras de identificación para el Modelo I con $n = 1$ y $m = 1$ como se denota a continuación [56]:

$$y(k+1) = \alpha_0 y(k) + g\left[u(k)\right]$$

El argumento es similar para las otras 3 estructuras de identificación.

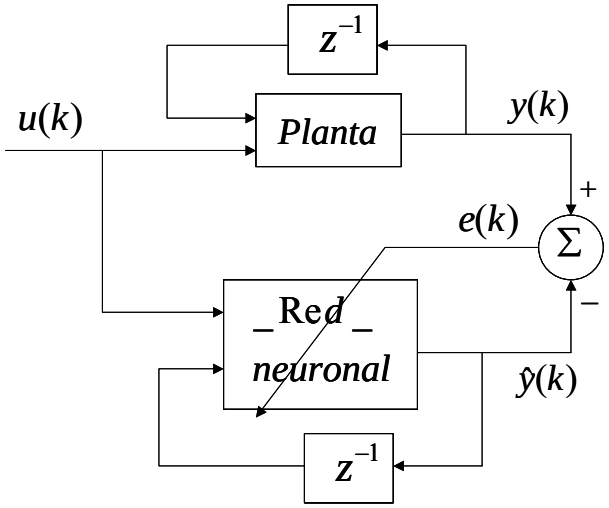

Figura 2.9: Estructura de identificación paralela

(a) Estructura de identificación paralela

En este caso se tiene la estructura siguiente:

$$\widehat{y}(k+1) = \widehat{\alpha}_0 \widehat{y}(k) + N\left[u(k)\right] \tag{2.23}$$

esta estructura de identificación se encuentra en la Figura 2.9.

(b) Estructura de identificación serie paralela

En este caso se tiene la estructura siguiente:

$$\widehat{y}(k+1) = \widehat{\alpha}_0 y(k) + N\left[u(k)\right] \tag{2.24}$$

esta estructura de identificación se encuentra en la Figura 2.10.

El proceso de identificación consiste en la estimación del parámetro α_0 y en ajustar los parámetros de la red neuronal para aproximar la función no lineal $g\left[u(k)\right]$, esto se hace comúnmente con el algoritmo de propagación hacia atrás basándose en el error $e(k)$ entre la salida de la estructura de identificación $\widehat{y}(k)$ y la salida de la planta $y(k)$. Estas estructuras no consideran perturbaciones directamente.

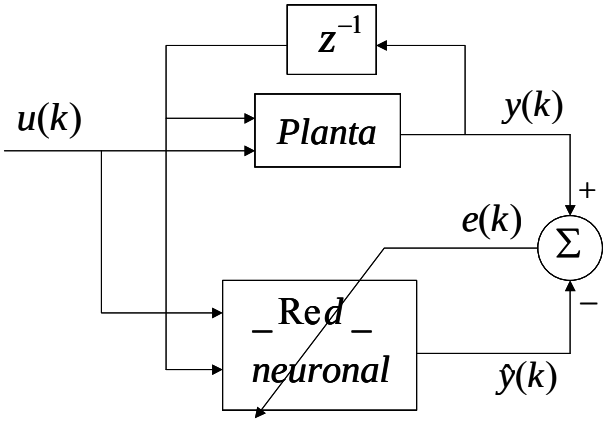

Figura 2.10: Estructura de identificación serie paralela

2.4. Espacios elipsoidales

Definición 2.1 *Un espacio elipsoidal* $E_1 = E_1(\widehat{x}, P)$ *es todos los* x *que caen dentro del siguiente espacio [69], [79]:*

$$E_1 = \left\{ x \in R^{4m} \mid (\widehat{x} - x^*)^T P^{-1} (\widehat{x} - x^*) \leq 1 \right\} \tag{2.25}$$

donde $P > 0$*, el centro de la elipsoide se determina por* $\widehat{x}$*, la orientación del elipsoide esta dado por los vectores propios de* P*,* $(u_1, \ldots, u_{4m})$*, y los ejes están dados por los valores propios de* P*,* $(\lambda_1, \ldots, \lambda_{4m})$ *como* $\frac{1}{\sqrt{\lambda_i}}$ *[69], [79], en Figura 2.11 se ve un ejemplo para un caso de dos dimensiones.*

Ejemplo 2.3 *Graficar la elipsoide que corresponde al espacio elipsoidal con* $P = \begin{bmatrix} 5 & 4 \\ 4 & 5 \end{bmatrix}$ *y* $\widehat{x} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$.

Los valores propios de P *se obtienen con* $\det(sI - P) = 0$ *con* $I \in R^{2\times 2}$ *matriz identidad y* $s \in R$ *escalar. Se tiene que* $\lambda_1 = 9$ *y* $\lambda_2 = 1$*, entonces se va a tener eje menor=*$\frac{1}{\sqrt{\lambda_1}} = \frac{1}{3}$ *y eje mayor=*$\frac{1}{\sqrt{\lambda_2}} = 1$*. Los vectores propios se obtienen de* $\lambda_i u_i = P u_i$ *para* $i = 1, 2$*, se tiene*

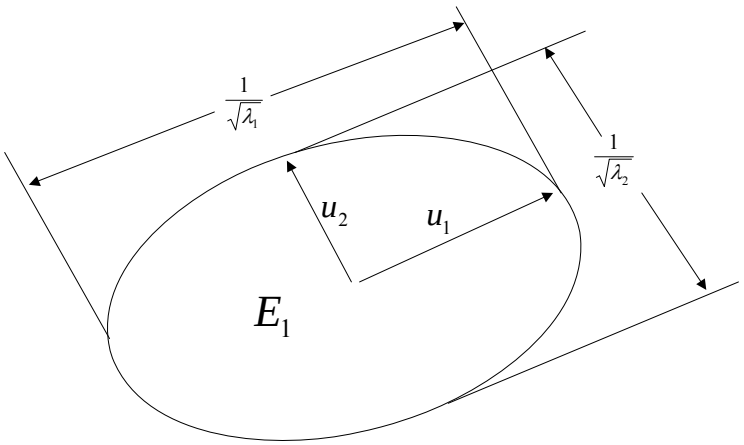

Figura 2.11: Espacio elipsoidal

$u_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ *y* $u_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ *los cuales determinan la orientación de eje menor y eje mayor, respectivamente a partir del centro* $\widehat{x}$*. El resultado de ve en la Figura 2.12.*

La intersección de dos espacios elipsoidales $E_a = \left\{ x \in R^{4m} : [x - \widehat{x}_a]^T P_a^{-1} [x - \widehat{x}_a] \leq 1 \right\}$ y $E_b = \left\{ x \in R^n : [x - \widehat{x}_b]^T P_b^{-1} [x - \widehat{x}_b] \leq 1 \right\}$ denotada como $E_a \cap E_b$ en general no es un espacio elipsoidal. Entonces se emplea un espacio elipsoidal E_c el cual contenga la intersección de los espacios elipsoidales, el cual de denota como $E_a \cap E_b \subset E_c$.

Definición 2.2 *La intersección elipsoidal denotada como* E_c *se define como sigue [41], [69], [79]:*

$$E_c = \left\{ x \in R^n : (1-\lambda) [x - \widehat{x}_a]^T P_a^{-1} [x - \widehat{x}_a] + \lambda [x - \widehat{x}_b]^T P_b^{-1} [x - \widehat{x}_b] \leq 1 \right\} \qquad (2.26)$$

donde $(1-\lambda) [x - \widehat{x}_a]^T P_a^{-1} [x - \widehat{x}_a] \leq 1 - \lambda$ *y* $\lambda [x - \widehat{x}_b]^T P_b^{-1} [x - \widehat{x}_b] \leq \lambda$ *vienen de* E_a *y* E_b*, respectivamente. Ver Figura 2.13.*

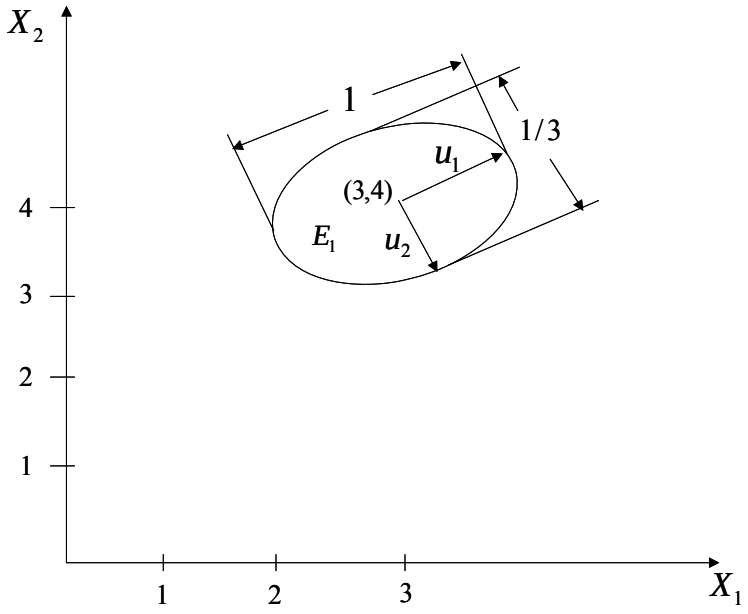

Figura 2.12: Ejemplo de un espacio elipsoidal

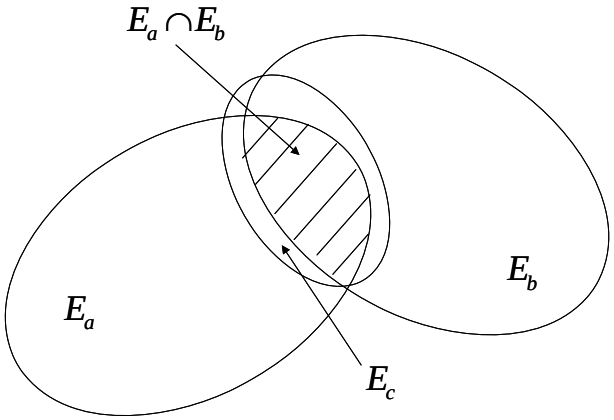

Figura 2.13: Espacio elipsoidal que contiene la intersección de dos espacios elipsoidales

2.5. Control por modos deslizantes en tiempo discreto

El método de modos deslizantes tradicional se presenta como la aplicación de una señal de control conmutando a alta frecuencia que consigue llevar el estado del sistema a una superficie cercana al origen denominada superficie de deslizamiento o función de conmutación y una vez llegando a la superficie de deslizamiento se debe permanecer en ésta ante posibles perturbaciones externas. Dicha superficie de deslizamiento es definida por el diseñador con el objeto de que el estado cumpla las especificaciones deseadas. La principal ventaja del control por modos deslizantes es que aporta robustez ante perturbaciones, tanto internas como externas, cuando estas tienen cotas conocidas.

2.5.1. Generalidades acerca de los modos deslizantes en tiempo continuo

En años recientes, se ha tenido mayor atención hacia los sistemas donde se emplea una acción de control discontinua [86]. Debido a una selección inteligente de la acción de control, las trayectorias del estado podrían cambiar tal que correspondan con las trayectorias deseadas en el sistema controlado. El problema de diseño de controlador se puede reducir a un problema de convergencia hacia una superficie especial. Cuando se satisfacen ciertas condiciones, se tiene un tipo de movimiento llamado modos deslizantes [87]. Los modos deslizantes tienen ciertas características atractivas. La idea básica en el diseño de sistemas con modos deslizantes corresponde con los siguientes dos pasos:

- Primero, se tiene el movimiento de modos deslizantes debido a la selección apropiada de las superficies discontinuas;
- Segundo, se elige la ley de control tal que la superficies de discontinuidad sea estable.

Formalmente, tales sistemas discontinuos se podrían describir mediante la siguiente ecuación

$$\dot{x} = f\left(x, t, u_t\right) \tag{2.27}$$

donde $x_t \in \Re^n$ es el vector de estados del sistema y $u_t \in \Re^m$ es una entrada de control discontinua, $t \in \Re^+$. Se define una clase de control discontinuo como sigue:

$$u_{t,i} = \begin{cases} u_{t,i}^{+}(x,t) & \text{si} \quad s_i(x) > 0 \\ u_{t,i}^{-}(x,t) & \text{si} \quad s_i(x) < 0 \end{cases} \qquad i = 1, 2 \ldots, m \tag{2.28}$$

donde $u_t = [u_{t,1}, u_{t,2}, \ldots u_{t,m}]^T$ y todas las funciones $u_{t,i}^{+}(x,t)$ y $u_{t,i}^{-}(x,t)$ son continuas. La función $s_i(x)$ es la superficie de discontinuidad definida por:

$$s_i(x) = 0, \quad s_i(x) \in \Re^1, \quad i = 1, 2 \ldots, m$$

Comúnmente se define como sigue:

$$s_i(x) \mid_{x = x_{t,i} - x_{t,i}^*} = k_i x, \quad k_i > 0$$

donde $x_{t,i}^*$ es el i-ésimo componente de la trayectoria deseada.

Algunas estrategias de conmutación de los controles continuos $u_i^{+}(x,t)$ y $u_i^{-}(x,t)$ podrían llevar a un comportamiento discontinuo en donde las trayectorias podrían ser no acotadas. Modos deslizantes existen en una superficie de discontinuidad cuando para cualquier distancia a esta superficie y sus velocidades $\dot{s}$ son de signo contrario, es decir:

$$\lim_{s \to -0} \dot{s} > 0 \text{ y } \lim_{s \to +0} \dot{s} < 0 \tag{2.29}$$

o

$$\dot{s}_i s_i < 0$$

La condición (2.29) es útil para la determinación de $u_i^{+}(x,t)$ y $u_i^{-}(x,t)$. El problema de encontrar el dominio de modos deslizantes se puede reducir al problema específico de garantizar la estabilidad de un sistemas no lineal.

2.5.2. Generalidades acerca de los modos deslizantes en tiempo discreto

Considere un sistema no lineal una entrada una salida en tiempo discreto que tiene la forma:

$$\begin{aligned} x(k+1) &= F\left(x(k), u(k)\right) \\ y(k) &= h\left(x(k)\right) \end{aligned} \tag{2.30}$$

definido $\forall x(k) \in X \subset R^n$ es el estado del sistema, $u(k) \in U \subset R$ es la entrada del sistema, $y(k) \in Y \subset R$ es la salida del sistema, se considera que las funciones $F : X \times U \rightarrow X$ y $h : X \rightarrow Y$ son analíticas.

Comentario 2.1 *En esta sección se va a considerar la superficie de deslizamiento como la salida* $s(k) = y(k)$.

Una ley de control por retroalimentación de estructura variable para el sistema (2.30) es la siguiente:

$$u = \begin{cases} u^+(x) & \text{para } h(x) > 0 \\ u^-(x) & \text{para } h(x) < 0 \\ 0 & \text{para } h(x) = 0 \end{cases} \tag{2.31}$$

con $u^+(x) > u^-(x)$ en X.

Definición 2.3 *El sistema controlado (2.30) presenta un movimiento deslizante alrededor de* $Z = \{x \in X : h\left(x(k)\right) = 0\}$ *cuando se satisface la siguiente condición [82]:*

$$y(k)\left[y(k+1) - y(k)\right] < 0 \tag{2.32}$$

Comentario 2.2 *Notar que el movimiento deslizante no se crea con una ley de control discontinua de la forma (2.31). De todas maneras es importante darse cuenta que se puede alcanzar el movimiento deslizante con una ley de control continua.*

La definición anterior es la contraparte en tiempo discreto para la condición de modos deslizantes en tiempo continuo dada como $y\frac{dy}{dt} < 0$ y se puede considerar su equivalente

en tiempo discreto como $y(k)\left[y(k+1)-y(k)\right]<0$. La condición (2.32) permite también movimientos deslizantes inestables como en el siguiente ejemplo.

Ejemplo 2.4 *Considere el siguiente sistema no lineal en tiempo discreto*

$$\begin{aligned} x_1(k+1) &= 1+x_1(k) \\ x_2(k+1) &= x_1(k)u(k) \\ y(k) &= x_2(k) \end{aligned}$$

con $x_1(1)=1$ *y* $x_2(1)\neq 0$.

Se propone la siguiente ley de control de estructura variable:

$$u(k) = -sign\left(y(k)\right)$$

En este caso se tiene $y(k+1)=x_2(k+1)=x_1(k)u(k)=-x_1(k)sign\left(y(k)\right)$, *por lo tanto se tiene:*

$$\begin{aligned} y(k)\left[y(k+1)-y(k)\right] &= y(k)\left[-x_1(k)sign\left(y(k)\right)-y(k)\right] \\ y(k)\left[y(k+1)-y(k)\right] &= -x_1(k)\left|y(k)\right|-y^2(k)<0 \end{aligned}$$

donde se satisface condición (2.32) para todos los $x_1(k)\geq 0$. *El comportamiento de la salida del sistema* $y(k)$ *con el control que satisface la condición de modos deslizantes (2.32) es claramente inestable como se ve en la Figura 2.14.*

Definición 2.4 *El sistema controlado (2.30) presenta un movimiento convergente alrededor de* $Z=\{x\in X : h\left(x(k)\right)=0\}$, *si existe una ley de control para la cual se satisface la siguiente condición [82]:*

$$\left|y(k+1)\right|<\left|y(k)\right| \tag{2.33}$$

Comentario 2.3 *Se puede pensar que el movimiento en modos deslizantes en tiempo discreto produzca castañeo, el castañeo ocurre en el movimiento en modos deslizantes en tiempo continuo. Con las condiciones (2.32) y (2.33) puede o no presentarse castañeo.*

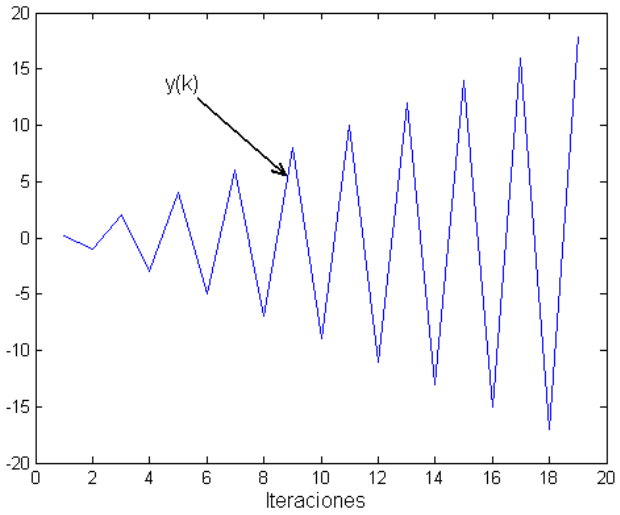

Figura 2.14: Comportamiento de la salida del sistema $y(k)$ con el control que satisface la condición de modos deslizantes (2.32)

Ejemplo 2.5 *Considere el sistema:*

$$x_1(k+1) = x_2(k)$$
$$x_2(k+1) = u(k)$$
$$y(k) = ax_1(k) + x_2(k)$$

con $|a| < 1$, $x_1(1) = 5$ *y* $x_2(1) = 5$.

Se propone la siguiente ley de control:

$$u(k) = a^2 x_1(k)$$

En este caso se tiene

$$y(k+1) = ax_1(k+1) + x_2(k+1) = ax_2(k) + u(k) =$$
$$ax_2(k) + a^2 x_1(k) = a\left[ax_1(k) + x_2(k)\right] = ay(k)$$

donde se satisface la condición (2.33) para $|a| < 1$. *Si* $0 < a < 1$ *el movimiento deslizante alcanza y=0 de manera asintótica como se ve en la Figura 2.15. Si* $-1 < a < 0$ *se tiene condiciones alrededor de* $y = 0$ *como se ve en la Figura 2.16.*

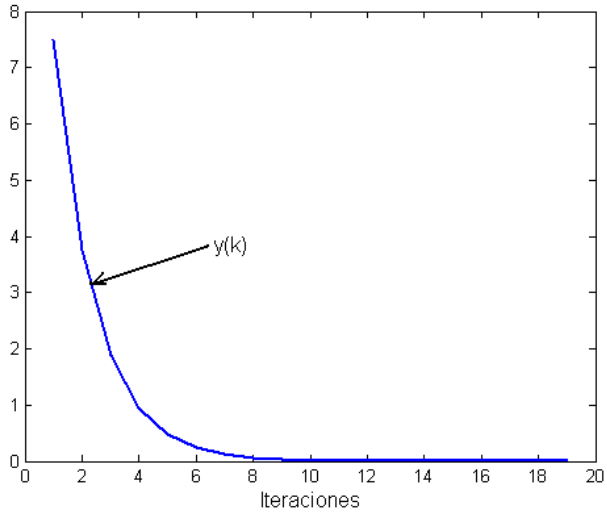

Figura 2.15: Comportamiento de la salida del sistema $y(k)$ con el control que satisface la condición de modos deslizantes (2.33) para $a = 0{,}5$

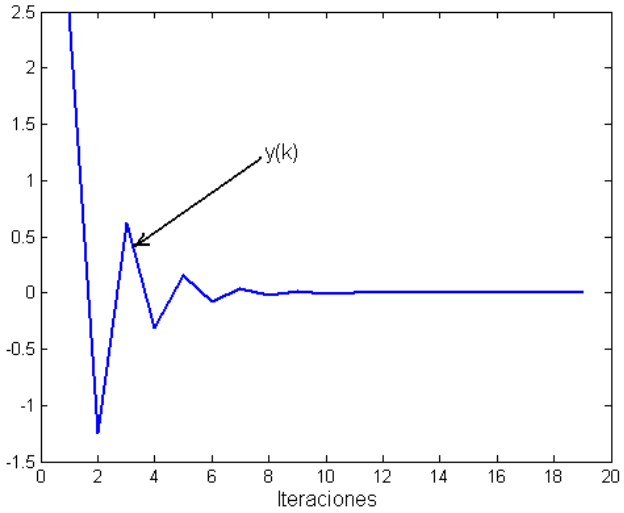

Figura 2.16: Comportamiento de la salida del sistema $y(k)$ con el control que satisface la condición de modos deslizantes (2.33) para $a = -0{,}5$

Lemma 2.1 *Una condición necesaria para la existencia de un movimiento deslizante y convergente alrededor de* $Z = \{x \in X : h(x(k)) = 0\}$*, es que se satisfaga la condición de modos deslizantes dada en (2.33), [82].*

Demostración. *Suponga que se satisface la condición (2.33), entonces se tiene*

$$|y(k+1)|\,|y(k)| < |y(k)|^2$$
$$y(k+1)y(k) < |y(k+1)y(k)| < y^2(k)$$
$$y(k)\,[y(k+1) - y(k)] < 0$$

Como se tiene (2.32) si se satisface el lema. ■

Teorema 2.1 *Se tiene un movimiento en modos deslizantes convergente si y solo si [82]:*

$$|y(k+1)y(k)| < y^2(k) \tag{2.34}$$

Demostración. *La necesidad se tiene del lema anterior. La suficiencia se tiene empleando el hecho de que* $|y(k+1)y(k)| = |y(k+1)|\,|y(k)| < y^2(k)$*, es decir se tiene* $|y(k+1)| < |y(k)|$*.* ■

Ejemplo 2.6 *Considere el siguiente sistema no lineal en tiempo discreto*

$$x_1(k+1) = 1 + x_1(k)$$
$$x_2(k+1) = x_1(k)u(k)$$
$$y(k) = x_2(k)$$

con $x_1(1) = 1$ *y* $x_2(1) \neq 0$*.*

Se propone la siguiente ley de control de estructura variable:

$$u(k) = -ax_2(k)x_1^{-1}(k)sign\,(y(k))$$

con $|a| < 1$*. En este caso se tiene* $y(k+1) = x_2(k+1) = x_1(k)u(k) = -ax_2(k)sign\,(y(k)) = -ay(k)sign\,(y(k))$*, por lo tanto se tiene:*

$$y(k)\,[y(k+1) - y(k)] = y(k)\,[-ay(k)sign\,(y(k)) - y(k)]$$
$$y(k)\,[y(k+1) - y(k)] = -y^2(k)\,[asign\,(y(k)) + 1] < 0$$

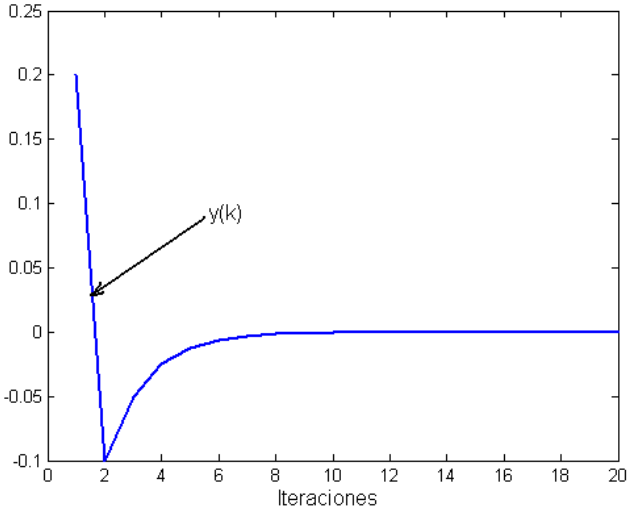

Figura 2.17: Comportamiento de la salida del sistema $y(k)$ con el control que satisface la condición de modos deslizantes (2.34) para $a = 0{,}5$

es decir existe movimiento deslizante alrededor de $y = 0$. *También se tiene*

$$|y(k+1)| = |-ay(k)sign\,(y(k))| = |a|\,|y(k)| < |y(k)|$$

es decir se tiene convergencia alrededor de $y = 0$. *El resultado se muestra en la Figura 2.17.*

La condición (2.33) puede asegurar la convergencia en la función de conmutación del error de seguimiento de trayectoria a una región cercana al origen, pero es muy dificil diseñar un controlador que satisfaga esta condición. Así [2] da otra condición de modos deslizantes en tiempo discreto, la superficie de deslizamiento $s(k) = 0$ debe satisfacer:

$$|s(k)| \leq q \tag{2.35}$$

donde el parámetro $q > 0$ es llamado ancho de banda de los modos deslizantes. Este no requiere que la trayectoria cruce la superficie de deslizamiento en cada paso de control.

2.6. Conclusión

Habiendo comprendido este capítulo, se considera que se tienen los conocimiento básicos para comprender de mejor manera lo expuesto en los capítulos siguientes.

Capítulo 3

Modelado con una red neuronal recurrente y el algoritmo elipsoidal acotado

3.1. Introducción

Resultados recientes muestran que la técnica de redes neuronales es muy efectiva para identificar una gran cantidad de sistemas no lineales complejos cuando no se tiene información completa del modelo. Las redes neuronales se clasifican en redes neuronales con conexión hacia adelante y redes neuronales recurrentes [28]. Aunque el algoritmo de propagación hacia atrás se ha usado ampliamente como método de entrenamiento para redes neuronales, la limitación es que este algoritmo tiene una convergencia de parámetros muy lenta. La estabilidad de un algoritmo de gradiente descendiente modificado se da en [93]. Para resolver este problema, se han propuesto en la identificación y la teoría de filtraje algoritmos para estimar los pesos de la red neuronal. Por ejemplo, el algoritmo del filtro de Kalman extendido aplicado al entrenamiento de redes neuronales en [36], [61], [70], [73] y [81]. La mayoría de los autores mencionados usan redes neuronales estáticas, algunas veces se considera que la capa de salida (lineal) se actualiza en cada iteración y en la capa oculta (no lineal) los

pesos se eligen aleatoriamente al principio del entrenamiento [10]. Se tiene una convergencia en parámetros más rápida con el algoritmo del filtro de Kalman extendido, por que este alcanza la convergencia de parámetros en menos iteraciones [36]. De cualquier manera, se incrementa la complejidad computacional en cada iteración, esto es, este requiere una gran cantidad de memoria. Se usa la técnica de desacoplamiento para decrementar la complejidad computacional [65], algoritmo del filtro de Kalman extendido con matriz de covarianza del error en parámetros diagonal es similar al algoritmo de propagación hacia atrás [28], pero el coeficiente de aprendizaje del algoritmo del filtro de Kalman es una matriz variante en tiempo. Una gran desventaja del algoritmo del filtro de Kalman extendido esta en el análisis del algoritmo ya que se requiere que la incertidumbre estructural de la red neuronal se modele como un proceso Gaussiano. El algoritmo elipsoidal acotado requiere solamente que la incertidumbre estructural de la red neuronal esté acotada y este tiene estructura similar al algoritmo del filtro de Kalman extendido [79]. El algoritmo elipsoidal acotado ofrece una atractiva alternativa para problemas de identificación y filtraje que consideran sistemas afines en parámetros.

En 1979 L.G.Khachiyan indicó como un método elipsoidal se puede implementar para programación lineal [3]. Este resultado ha causado gran interés y ha estimulado a la investigación en este tema. La técnica elipsoidal tiene importantes ventajas en la estimación de estados con perturbaciones acotadas [21]. Hay muchas aplicaciones potenciales para problemas semejantes a la programación lineal. [89] obtuvo elipsoides los cuales eran validos para un número finito de datos. [69] presentó una programación elipsoidal tal que el nuevo elipsoide satisface una relación afín con otro elipsoide. En [12], el algoritmo elipsoidal se usa como una técnica de optimización que toma en cuenta las restricciones sobre coeficientes agrupados. [52] describió a detalle varios métodos que pueden ser usados para derivar una incertidumbre elipsoidal apropiada. En [57], se considera el problema elipsoidal con comportamiento asintótico para sistemas lineales en tiempo discreto. Hay pocas publicaciones de elipsoides en combinación con redes neuronales. En [15] se usan aprendizajes supervisado y no supervisado en forma elipsoidal para encontrar y sintonizar las reglas de funciones difusas para un proceso de identificación difuso. En [42] se proponen funciones de activación de tipo

elipsoidal para redes neuronales con conexión hacia adelante. De la revisión de literatura se ha encontrado que hasta ahora no se ha propuesto un algoritmo elipsoidal acotado para el entrenamiento de una red neuronal recurrente.

En esta sección se propone un algoritmo elipsoidal acotado el cual se usa para el entrenamiento de los pesos de una red neuronal recurrente para la identificación de sistemas no lineales. Se actualizan capa tanto la oculta como la capa de salida. Se prueba la estabilidad del error de identificación para el algoritmo elipsoidal acotado usando una técnica semejante a la técnica de Lyapunov. Desde el punto de vista de sistemas dinámicos, tal entrenamiento es útil para todas las aplicaciones de redes neuronales que requieren la actualización de los pesos en línea. Una simulación muestra la efectividad del algoritmo propuesto.

3.2. Red neuronal recurrente para modelación de sistemas no lineales

Considere el siguiente sistema no lineal en tiempo discreto:

$$x(k+1) = f\left[x(k), u(k)\right] \tag{3.1}$$

donde $u\left(k\right) \in \Re^m$ es el vector de entrada, $\left|u(k)\right|^2 \leq \overline{u}$, $x\left(k\right) \in \Re^n$ es el vector de estados, $u\left(k\right)$ y $x\left(k\right)$ son conocidos. f es una función suave no lineal, $f \in C^\infty$. Se usa la siguiente red neuronal recurrente para identificar la planta (3.1):

$$\widehat{x}(k+1) = A\widehat{x}(k) + V_{1,k}\sigma\left[W_{1,k}x(k)\right] + V_{2,k}\phi\left[W_{2,k}x(k)\right]u(k) \tag{3.2}$$

donde $\widehat{x}\left(k\right) \in \Re^n$ representa el estado de la red neuronal. La matriz $A \in \Re^{n\times n}$ es una matriz estable. Los pesos de la capa de salida son $V_{1,k}$, $V_{2,k} \in R^{n\times m}$, los pesos de la capa oculta son $W_{1,k}$, $W_{2,k} \in R^{m\times n}$, σ es una función m dimensional, $\sigma = \left[\sigma_1 \cdots \sigma_m\right]^T$, $\phi(\cdot) \in \Re^{m\times m}$ es una

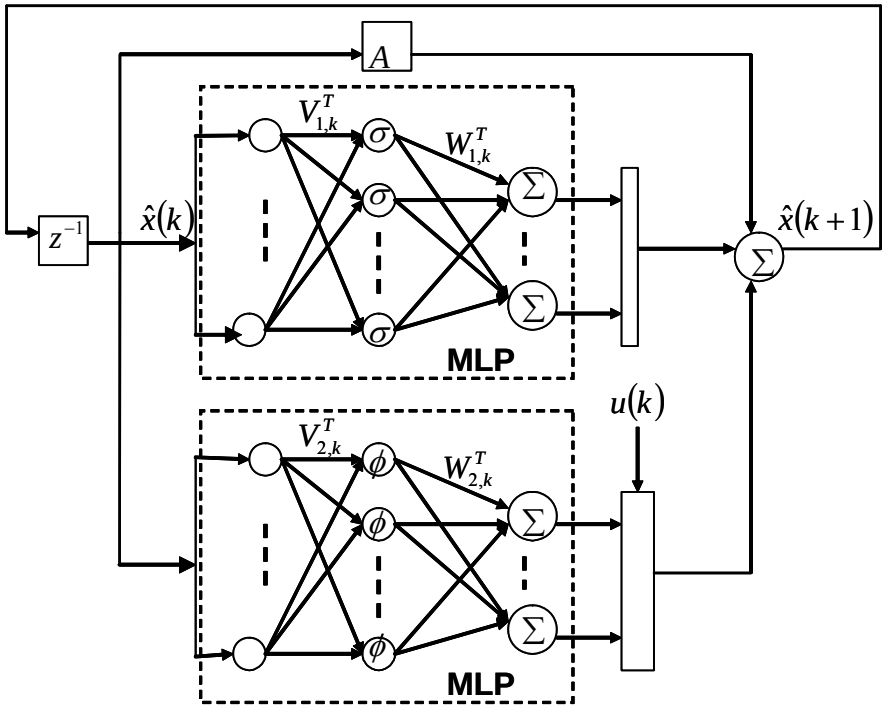

Figura 3.1: Estructura de la red neuronal recurrente

matriz diagonal.

$$\sigma\left[W_{1,k}x(k)\right] = [\sigma_1\left(\sum_{j=1}^{n} w_{1,1,j}x_j\right), \sigma_2\left(\sum_{j=1}^{n} w_{1,2,j}x_j\right), \cdots \sigma_m\left(\sum_{j=1}^{n} w_{1,m,j}x_j\right)]^T$$
$$\phi\left[W_{2,k}x(k)\right]u(k) = [\phi_1\left(\sum_{j=1}^{n} w_{2,1,j}x_j\right)u_1, \phi_2\left(\sum_{j=1}^{n} w_{2,2,j}x_j\right)u_2, \cdots \phi_m\left(\sum_{j=1}^{n} w_{2,m,j}x_j\right)u_m]^T \tag{3.3}$$

donde σ_i y ϕ_i son funciones sigmoidales o funciones tangentes hiperbólicas Ver Figura 3.1.

El sistema no lineal (3.1) se puede escribir en la siguiente forma:

$$x(k+1) = Ax(k) + V_{1,k}\sigma\left[W_{1,k}x(k)\right] + V_{2,k}\phi\left[W_{2,k}x(k)\right]u(k) - \eta(k) \tag{3.4}$$

donde $\eta(k)$ es la incertidumbre estructural. Debido a [45] se conoce que el termino $\eta(k)$ se puede hacer arbitrariamente pequeño seleccionando el numero apropiado de neuronas en la

capa oculta de (3.2), este es m. En el caso de dos variables independientes, una función suave f tiene la siguiente expansión de Taylor cerca al punto $[x_1^0, x_2^0]$,

$$f = \sum_{k=0}^{l-1} \frac{1}{k!} \left[\left(x_1 - x_1^0\right) \frac{\partial}{\partial x_1} + \left(x_2 - x_2^0\right) \frac{\partial}{\partial x_2} \right]_0^k f + \varepsilon \tag{3.5}$$

donde ε es el residuo de la serie de Taylor. Si x_1 y x_2 corresponden con $W_{1,k}x\left(k\right)$ y $V_{1,k}$, x_1^0, x_2^0 corresponden con $W_1^0 x\left(k\right)$ y V_1^0, entonces se tiene:

$$V_{1,k}\sigma\left[W_{1,k}x(k)\right] = V_1^0\sigma\left[W_1^0 x\left(k\right)\right] + \Theta_{1,k}B_{1,k} + \varepsilon_1 \tag{3.6}$$

donde V_1^0, V_2^0, W_1^0 y W_2^0 son los pesos iniciales constantes conocidos, $B_{1,k} = \left[\sigma, \sigma' V_{1,k}^T x\right]^T \in R^{2m\times 1}$, $\Theta_{1,k} = \left[V_{1,k}, W_{1,k}^T\right]^T \in R^{n\times 2m}$, σ' es la derivada de la función de activación $\sigma\left(\cdot\right)$ con respecto a $W_{1,k}x(k)$, ver la definición (3.3), $\sigma' \in \Re^{m\times m}$, similarmente se obtiene:

$$V_{2,k}\phi\left[W_{2,k}x(k)\right]u(k) = V_2^0\phi\left[W_2^0 x\left(k\right)\right]u(k) + \Theta_{2,k}B_{2,k} + \varepsilon_2 \tag{3.7}$$

donde $B_{2,k} = \left[\phi u, \phi' \mathrm{diag}(u) V_{2,k}^T x\left(k\right)\right]^T$, $\Theta_{2,k} = \left[V_{2,k}, W_{2,k}^T\right]^T$. Se define el error de la incertidumbre estructural como sigue:

$$\zeta(k) = \varepsilon_1 + \varepsilon_2 - \eta(k) \tag{3.8}$$

substituyendo (3.6) y (3.7) en (3.4) se tiene:

$$y(k) = \Theta_k B_k + \zeta(k) \tag{3.9}$$

donde el parámetro es:

$$\Theta_k = \left[\Theta_{1,k}, \Theta_{2,k}\right]^T = \left[V_{1,k}, W_{1,k}^T, V_{2,k}, W_{2,k}^T\right]^T \in R^{n\times 4m} \tag{3.10}$$

el dato es:

$$B_k = \left[B_{1,k}, B_{2,k}\right]^T = \left[\sigma, \sigma' V_{1,k}^T x, \phi u, \phi' \mathrm{diag}(u) V_{2,k}^T x\left(k\right)\right]^T \in R^{4m\times 1} \tag{3.11}$$

la salida es:

$$y(k) = x(k+1) - Ax(k) - V_1^0\sigma\left[W_1^0 x\left(k\right)\right] - V_2^0\phi\left[W_2^0 x\left(k\right)\right]u(k) \tag{3.12}$$

Se escribe (3.9) y (3.12) en forma de salidas individuales:

$$y_i(k) = B_k^T \theta_i(k) + \zeta_{i,k} \tag{3.13}$$

donde $i = 1 \cdots n$, $\theta_i(k) \in R^{4m \times 1}$, $\Theta_k = [\theta_1(k), \cdots \theta_n(k)]^T$, $y(k) = [y_1(k), \cdots y_n(k)]$, $\zeta = \left[\zeta_{1,k} \cdots \zeta_{n,k}\right]^T$, $B_k \in R^{4m \times 1}$.

$$y_i(k) = x_i(k+1) - a_i x_i(k) + V_1^0 \sigma\left[W_1^0 x_i(k)\right] + V_2^0 \phi\left[W_2^0 x_i(k)\right] u(k)$$

La salida de la red neuronal recurrente (3.1) se define como:

$$\widehat{y}_i(k) = B_k^T \theta_i(k) \tag{3.14}$$

El error del entrenamiento se define como:

$$e_i(k) = y_i(k) - \widehat{y}_i(k) \tag{3.15}$$

Entonces se define el error de identificación $\epsilon_i(k)$ entre la planta (3.1) y la red neuronal (3.2) como sigue:

$$\epsilon_i(k) = x_i(k) - \widehat{x}_i(k) \tag{3.16}$$

3.3. Algoritmo elipsoidal para el entrenamiento de los pesos de la red neuronal

Ahora se usa el algoritmo elipsoidal acotado para entrenar los pesos de la red neuronal (3.2) tal que se garantize que el error de identificación $\epsilon_i(k)$ esté acotado.

La intersección de dos espacios elipsoidales $E_a \cap E_b$ definida en (2.26) no es un espacio elipsoidal en general. El espacio elipsoidal E_c contiene la intersección normal de los espacios elipsoidales, $E_a \cap E_b \subset E_c$ y existe un elipsoide mínimo el cual corresponde a encontrar λ óptima (λ^*), ver [41], [69] y [79]. En este documento, no se trata de encontrar λ^*, se va a diseñar un algoritmo tal que el nuevo elipsoide que contiene la intersección sea cada vez más pequeño. La Figura 2.13 muestra esta idea.

Ahora se usa la definición de elipsoide dada en (2.25) para la identificación de los pesos de la red neuronal, se define el espacio elipsoidal E_k como sigue:

$$E_k = \left\{ \theta_i(k) \mid \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) \leq 1 \right\} \tag{3.17}$$

donde $\widetilde{\theta}_i(k) = \theta_i^* - \theta_i(k)$, θ_i^* es el peso óptimo que minimiza la incertidumbre estructural $\zeta_{i,k}$ en (3.13), $P_k = P_k^T > 0$.

A1 Se asume que $\left[y_i(k) - B_k^T \theta_i^* \right]$ pertenece a un espacio elipsoidal S_k,

$$S_k = \left\{ B_k^T \theta_i^* \mid \frac{1}{\overline{\varsigma}_i^2} \left\| y_i(k) - B_k^T \theta_i^* \right\|^2 \leq 1 \right\} \tag{3.18}$$

donde $\overline{\varsigma}_i > 0$ es la cota conocida del error de la incertidumbre estructural $\varsigma_{i,k}$, $|\varsigma_{i,k}| < \overline{\varsigma}_i$, $i = 1 \cdots n$. La ecuación (3.18) se puede escribir como sigue:

$$y_i(k) = B_k^T \theta_i^* + \varsigma_{i,k} \tag{3.19}$$

A2 Se asume que el error de los pesos iniciales esta dentro de una elipsoide denotada como E_1

$$E_1 = \left\{ \theta_i(1) \mid \widetilde{\theta}_i^T(1) P_1^{-1} \widetilde{\theta}_i(1) \leq 1 \right\} \tag{3.20}$$

donde $P_1 = P_1^T > 0$, $P_1 \in R^{4m \times 4m}$, $\widetilde{\theta}_i(1) = \theta_i^* - \theta_i(1)$, θ_i^* es el peso desconocido óptimo.

Comentario 3.1 *La suposición **A1** requiere que el término* $\left[y_i(k) - B_k^T \theta_i^* \right]$ *sea acotado por* $\overline{\varsigma}_i$. *En esta sección se considera solamente la identificación en lazo abierto, se asume que la planta (3.1) tiene entrada acotada y salida acotada, es decir,* $x(k)$ *y* $u(k)$ *en (3.1) son acotados y dado que* σ *y* ϕ *son acotados:*

$$y_i(k) - B_k^T \theta_i^* = x_i(k+1) - a_i x_i(k) + V_1^0 \sigma \left[W_1^0 x_i(k) \right] + V_2^0 \phi \left[W_2^0 x_i(k) \right] u(k) - B_k^T \theta_i^*$$

todos los datos son acotados, entonces $\left[y_i(k) - B_k^T \theta_i^* \right]$ *también es acotado. La suposición **A2** requiere que los pesos iniciales de la red neuronal sean acotados, esto se puede lograr eligiendo* P_1 *y* $\widehat{\theta}_i(1)$ *adecuados.*

Se puede ver que la parte común de los espacios elipsoidales S_1, $S_2, \cdots$ es θ_i^*, así:

$$\{\theta_i^*\} \subset \bigcap_{j=1}^{k} S_j \tag{3.21}$$

Encontrar $\{\theta_i^*\}$ utilizando esta propiedad es una tarea dificil ya que la cantidad de información en (3.21) crece linealmente con k. De cualquier manera, evaluar $\theta_i(k)$ en (3.21) es encontrar la solución de ecuaciones de $2k$ n orden. Desde la definición de E_k en (3.17), θ_i^* es el centro común de los espacios elipsoidales E_1, $E_2, \cdots$,así:

$$\{\theta_i^*\} \subset \bigcap_{j=1}^{k} E_j, \quad \{\theta_i^*\} \subset E_k \tag{3.22}$$

Entonces el problema de identificación es encontrar el espacio elipsoidal E_k el cual satisfaga (3.22). Se va a encontrar un algoritmo recursivo de identificación tal que E_{k+1} se encuentra en el espacio que corresponde a E_k y a los datos $[y_i(k), B_k]$. Debido a que los dos elipsoides satisfacen (3.17) y (3.18), se calcula la intersección de elipsoides $(1-\lambda_k)E_k + \lambda_k S_k$, ésta intersección satisface:

$$\begin{array}{l} (1-\lambda_k)\widetilde{\theta}_i^T(k)P_k^{-1}\widetilde{\theta}_i(k) \leq (1-\lambda_k) \\ \frac{1}{\overline{\varsigma}_i^2}\lambda_k \left\| y_i(k) - B_k^T\theta_i^* \right\|^2 \leq \lambda_k \\ (1-\lambda_k)\widetilde{\theta}_i^T(k)P_k^{-1}\widetilde{\theta}_i(k) + \frac{\lambda_k}{\overline{\varsigma}_i^2}\left\| y_i(k) - B_k^T\theta_i^* \right\|^2 \leq 1 \end{array} \tag{3.23}$$

El siguiente teorema muestra el proceso de propagación de los elipsoides.

Teorema 3.1 *Si E_k en (3.17) es un espacio elipsoidal, se usa el siguiente algoritmo recursivo para actualizar P_k y $\theta_i(k)$:*

$$\begin{array}{l} P_{k+1} = \frac{1}{1-\lambda_k}\left[P_k - P_k B_k \frac{\lambda_k}{(1-\lambda_k)\overline{\varsigma}_i^2 + \lambda_k B_k^T P_k B_k} B_k^T P_k\right] \\ \theta_i(k+1) = \theta_i(k) + \frac{\lambda_k}{\overline{\varsigma}_i^2} P_{k+1} B_k e_i(k) \\ \lambda_k = \begin{cases} \frac{\lambda\overline{\varsigma}_i^2}{1+B_k^T P_k B_k}, & si\ e_i^2(k) \geq \frac{\overline{\varsigma}_i^2}{1-\lambda} \\ 0, & si\ e_i^2(k) < \frac{\overline{\varsigma}_i^2}{1-\lambda} \end{cases} \end{array} \tag{3.24}$$

donde P_1 *es una matriz positiva definida,* $0 < \lambda < 1$ *y* $0 < \lambda\overline{\varsigma}_i^2 < 1$, $0 < \lambda_k < 1$ *y* $(1-\lambda_k) > 0$, *entonces* E_{k+1} *es un espacio elipsoidal y satisface:*

$$E_{k+1} = \left\{ \theta_i(k+1) \mid \widetilde{\theta}_i^T(k+1) P_{k+1}^{-1} \widetilde{\theta}_i(k+1) \leq 1 - \frac{\lambda_k}{\overline{\varsigma}_i^2}[1-\lambda] e_i^2(k) \leq 1 \right\} \tag{3.25}$$

donde $\widetilde{\theta}_i(k) = \theta_i^* - \theta_i(k)$ *y* $e_i(k)$ *están dados en (3.15).*

Demostración. *Primero se aplica el lema de inversión de matrices [26] para calcular* P_{k+1}^{-1} *en (3.24):*

$$P_{k+1}^{-1} = (1-\lambda_k)\left[P_k - P_k \frac{\lambda_k}{\overline{\varsigma}_i^2} B_k \left[B_k^T P_k \frac{\lambda_k}{\overline{\varsigma}_i^2} B_k + (1-\lambda_k) \right]^{-1} B_k^T P_k \right]^{-1}$$

el lema de inversión de matrices es:

$$(A+BCD)^{-1} = A^{-1} - A^{-1}B\left(DA^{-1}B + C^{-1}\right)^{-1} DA^{-1}$$

donde A, B, C y D son matrices de dimensiones adecuadas. La pasada ecuación se escribe como sigue:

$$\left(A^{-1} - A^{-1}B\left(DA^{-1}B + C^{-1}\right)^{-1} DA^{-1} \right)^{-1} = A + BCD$$

Específicamente usando $A \in R^{4m\times 4m}$, $B \in R^{4m\times 1}$, $C \in R^{1\times 1}$ *y* $D \in R^{1\times 4m}$, $A^{-1} = P_k$, $B = \frac{\lambda_k}{\overline{\varsigma}_i^2} B_k$, $C^{-1} = (1-\lambda_k)$ *y* $D = B_k^T$, *se tiene que* P_{k+1}^{-1} *es:*

$$\begin{aligned} P_{k+1}^{-1} &= (1-\lambda_k)\left[P_k^{-1} + \tfrac{\lambda_k}{\overline{\varsigma}_i^2} B_k \tfrac{1}{(1-\lambda_k)} B_k^T \right] \\ &= (1-\lambda_k) P_k^{-1} + \tfrac{\lambda_k}{\overline{\varsigma}_i^2} B_k B_k^T \end{aligned} \tag{3.26}$$

Ahora se calcula $\widetilde{\theta}_i(k+1)^T P_{k+1}^{-1} \widetilde{\theta}_i(k+1)$ *para* $e_i^2(k) \geq \frac{\overline{\varsigma}_i^2}{1-\lambda}$ *por (3.24) o* $\widetilde{\theta}_i(k+1) = \widetilde{\theta}_i(k) - \frac{\lambda_k}{\overline{\varsigma}_i^2} P_{k+1} B_k e_i(k)$ *se tiene:*

$$\widetilde{\theta}_i^T(k+1) P_{k+1}^{-1} \widetilde{\theta}_i(k+1) = \widetilde{\theta}_i^T(k) P_{k+1}^{-1} \widetilde{\theta}_i(k) - 2\frac{\lambda_k}{\overline{\varsigma}_i^2} \widetilde{\theta}_i^T(k) B_k e_i(k) + \frac{\lambda_k^2}{\overline{\varsigma}_i^4} B_k^T P_{k+1} B_k e_i^2(k) \tag{3.27}$$

Substituyendo (3.26) en (3.27), se tiene el siguiente resultado:

$$\begin{aligned} &\widetilde{\theta}_i^T(k+1) P_{k+1}^{-1} \widetilde{\theta}_i(k+1) \\ &= \widetilde{\theta}_i^T(k) \left[(1-\lambda_k) P_k^{-1} + \tfrac{\lambda_k}{\overline{\varsigma}_i^2} B_k B_k^T \right] \widetilde{\theta}_i(k) - 2\tfrac{\lambda_k}{\overline{\varsigma}_i^2} \widetilde{\theta}_i^T(k) B_k e_i(k) + \tfrac{\lambda_k^2}{\overline{\varsigma}_i^4} B_k^T P_{k+1} B_k e_i^2(k) \\ &= (1-\lambda_k) \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) + \tfrac{\lambda_k}{\overline{\varsigma}_i^2} \widetilde{\theta}_i^T(k) B_k B_k^T \widetilde{\theta}_i(k) - 2\tfrac{\lambda_k}{\overline{\varsigma}_i^2} \widetilde{\theta}_i^T(k) B_k e_i(k) + \tfrac{\lambda_k^2}{\overline{\varsigma}_i^4} B_k^T P_{k+1} B_k e_i^2(k) \end{aligned} \tag{3.28}$$

Debido a la propiedad de intersección de espacios elipsoidales (3.23), se tiene lo siguiente:

$$(1-\lambda_k)\,\widetilde{\theta}_i^T(k)P_k^{-1}\widetilde{\theta}_i(k) \leq 1-\frac{\lambda_k}{\overline{\varsigma}_i^2}\left\|y_i(k)-B_k^T\theta_i^*\right\|^2$$

Entonces (3.28) se convierte:

$$\begin{aligned}
&\widetilde{\theta}_i^T(k+1)P_{k+1}^{-1}\widetilde{\theta}_i(k+1)\\
&\leq 1-\tfrac{\lambda_k}{\overline{\varsigma}_i^2}\left\|y_i(k)-B_k^T\theta_i^*\right\|^2+\tfrac{\lambda_k}{\overline{\varsigma}_i^2}\widetilde{\theta}_i^T(k)B_kB_k^T\widetilde{\theta}_i(k)-2\tfrac{\lambda_k}{\overline{\varsigma}_i^2}\widetilde{\theta}_i^T(k)B_ke_i(k)+\tfrac{\lambda_k^2}{\overline{\varsigma}_i^4}B_k^TP_{k+1}B_ke_i^2(k)\\
&\leq 1+\tfrac{\lambda_k}{\overline{\varsigma}_i^2}\left[-\left\|y_i(k)-B_k^T\theta_i^*\right\|^2+\widetilde{\theta}_i^T(k)B_kB_k^T\widetilde{\theta}_i(k)-2\widetilde{\theta}_i^T(k)B_ke_i(k)\right]+\tfrac{\lambda_k^2}{\overline{\varsigma}_i^4}B_k^TP_{k+1}B_ke_i^2(k)
\end{aligned}$$

Ahora usando $\widetilde{\theta}_i(k)=\theta_i^*-\theta_i(k)$, $e_i(k)=y_i(k)-B_k^T\theta_i(k)$ *como en (3.15), el segundo término se calcula como sigue:*

$$\begin{aligned}
&-\left\|y_i(k)-B_k^T\theta_i^*\right\|^2+\widetilde{\theta}_i^T(k)B_kB_k^T\widetilde{\theta}_i(k)-2\widetilde{\theta}_i^T(k)B_ke_i(k)\\
&=-\left\|y_i(k)-B_k^T\theta_i^*\right\|^2+\left[\theta_i^*-\theta_i(k)\right]^TB_kB_k^T\left[\theta_i^*-\theta_i(k)\right]-2\left[\theta_i^*-\theta_i(k)\right]^TB_k\left[y_i(k)-B_k^T\theta_i(k)\right]\\
&=-\left[y_i^2(k)-2\theta_i^T(k)B_ky_i(k)+\theta_i^T(k)B_kB_k^T\theta_i(k)\right]=-\left[y_i(k)-B_k^T\theta_i(k)\right]^2\\
&=-e_i^2(k)
\end{aligned}$$

Así:

$$\widetilde{\theta}_i^T(k+1)P_{k+1}^{-1}\widetilde{\theta}_i(k+1)\leq 1-\frac{\lambda_k}{\overline{\varsigma}_i^2}e_i^2(k)+\frac{\lambda_k^2}{\overline{\varsigma}_i^4}B_k^TP_{k+1}B_ke_i^2(k)$$

Desde (3.26) se conoce que $P_{k+1}^{-1}=(1-\lambda_k)P_k^{-1}+\frac{\lambda_k}{\overline{\varsigma}_i^2}B_kB_k^T$. *Usando el hecho de que* $\lambda_k=\frac{\lambda\overline{\varsigma}_i^2}{1+B_k^TP_{k+1}B_k}>0$, $P_{k+1}>0$ *ya que* $P_1>0$. *Usando el algoritmo de actualización (3.24):*

$$\lambda_k^2B_k^TP_{k+1}B_ke_i^2(k)=\frac{\overline{\varsigma}_i^2\lambda_k}{\overline{\varsigma}_i^4}\lambda\frac{B_k^TP_{k+1}B_ke_i^2(k)}{1+B_k^TP_{k+1}B_k}\leq\frac{\lambda_k}{\overline{\varsigma}_i^2}\lambda e_i^2(k)$$

Así:

$$\widetilde{\theta}_i^T(k+1)P_{k+1}^{-1}\widetilde{\theta}_i(k+1)\leq 1-\frac{\lambda_k}{\overline{\varsigma}_i^2}\left[1-\lambda\right]e_i^2(k) \tag{3.29}$$

que es (3.25). Ya que $\lambda<1$ *y* $\lambda_k>0$

$$\widetilde{\theta}_i^T(k+1)P_{k+1}^{-1}\widetilde{\theta}_i(k+1)\leq 1 \tag{3.30}$$

entonces para $e_i^2(k)\geq\frac{\overline{\varsigma}_i^2}{1-\lambda}$ *el término* $\widetilde{\theta}_i^T(k+1)P_{k+1}^{-1}\widetilde{\theta}_i(k+1)$ *pertenece a un espacio elipsoidal. Ahora se considera el caso cuando* $e_i^2(k)<\frac{\overline{\varsigma}_i^2}{1-\lambda}$, *entonces* $\lambda_k=0$, *sustituyendo en (3.24) se*

tiene $P_{k+1} = \frac{1}{1-0}\left[P_k - 0\right] = P_k \rightarrow P_{k+1}^{-1} = P_k^{-1}$, $\theta_i\left(k+1\right) = \theta_i\left(k\right) + 0 = \theta_i\left(k\right) \rightarrow \widetilde{\theta}_i(k+1) = \widetilde{\theta}_i(k)$, *sustituyendo en* $\widetilde{\theta}_i^T\left(k+1\right)P_{k+1}^{-1}\widetilde{\theta}_i(k+1) \leq 1$ *se tiene:*

$$\widetilde{\theta}_i^T\left(k+1\right)P_{k+1}^{-1}\widetilde{\theta}_i(k+1) = \widetilde{\theta}_i^T\left(k\right)P_k^{-1}\widetilde{\theta}_i(k) \leq 1$$

para $e_i^2\left(k\right) < \frac{\overline{\varsigma}_i^2}{1-\lambda}$ *el término* $\widetilde{\theta}_i^T\left(k+1\right)P_{k+1}^{-1}\widetilde{\theta}_i(k+1)$ *también pertenece a un espacio elipsoidal. Entonces* E_{k+1} *es un espacio elipsoidal.* ■

Comentario 3.2 *Se va a comparar la intersección de* E_k *y* S_k *con la intersección de* E_{k+1} *y* S_{k+1}*. La intersección de* E_k *y* S_k *es (3.23), definida como sigue:*

$$\Pi_k = \left\{ z_i\left(k\right) \mid \left[z_i\left(k\right) - \theta_i^*\right]^T \Sigma^{-1}\left[z_i\left(k\right) - \theta_i^*\right] \leq 1 \right\}$$

donde $z\left(k\right)$ *es una variable desconocida. La intersección de* E_{k+1} *y* S_{k+1} *con el algoritmo (3.24) es:*

$$\begin{aligned}
&\left(1-\lambda_{k+1}\right)\widetilde{\theta}_i^T\left(k+1\right)P_{k+1}^{-1}\widetilde{\theta}_i(k+1) + \tfrac{1}{\overline{\varsigma}_i^2}\lambda_{k+1}\left\|y_i\left(k+1\right) - B_{k+1}^T\theta_i^*\right\|^2 \\
&\leq \left(1-\lambda_{k+1}\right)\left[1 - \tfrac{\lambda_k}{\overline{\varsigma}_i^2}\left[1-\lambda\right]e_i^2\left(k\right)\right] + \lambda_{k+1} \\
&= 1 - \tfrac{\lambda_k}{\overline{\varsigma}_i^2}\left(1-\lambda_{k+1}\right)\left[1-\lambda\right]e_i^2\left(k\right)
\end{aligned}$$

así:

$$\Pi_{k+1} = \left\{ z_i\left(k+1\right) \mid \left[z_i\left(k+1\right) - \theta_i^*\right]^T \Sigma^{-1}\left[z_i\left(k+1\right) - \theta_i^*\right] \leq 1 - \frac{\left(1-\lambda_{k+1}\right)\lambda_k\left(1-\lambda\right)}{\overline{\varsigma}_i^2}e_i^2\left(k\right) \right\}$$

El volumen de Π_k *es [69]:*

$$Vol(\Pi_k) = \sqrt{\det\left(\Sigma\right)}U$$

donde U *es una constante que representa de una bola unitaria en* R^n*. Debido a que se tiene* $\left(1-\lambda_{k+1}\right)\lambda_k\left(1-\lambda\right)e_i^2\left(k\right) > 0$*, el volumen de* Π_{k+1} *es menor que el volumen de* Π_k*. Desde (3.21) y (3.22) se conoce que la parte común de* Π_k *y* Π_{k+1} *es* $\left\{\theta_i^*\right\}$*. Entonces la intersección* Π_k *convergerá a* $\left\{\theta_i^*\right\}$ *cuando* $e_i\left(k\right) \neq 0$*, ver Figura 3.2.*

Comentario 3.3 *El algoritmo (3.24) es para cada subsistema. De (3.6) y (3.7) se conoce que los datos* B_k *dependen de los parámetros* $V_{1,k}^T$ *y* $V_{2,k}^T$*, esto no afecta el algoritmo de*

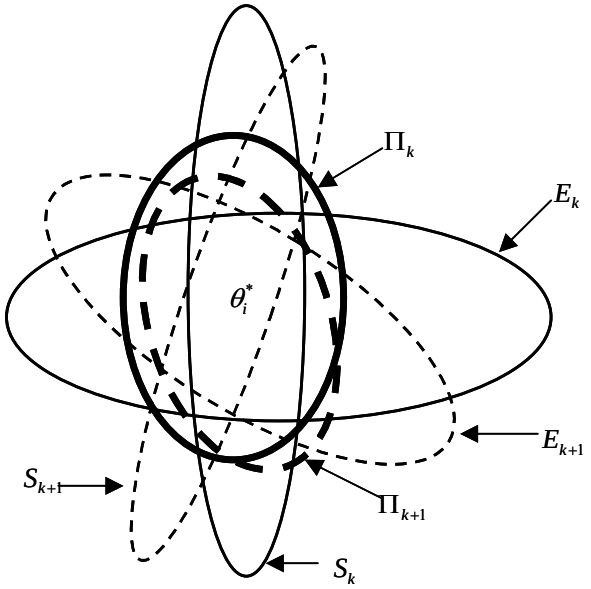

Figura 3.2: Relación entre los volúmenes de Π_k y Π_{k+1}

identificación (3.24), por el parámetro desconocido $\theta_i(k+1)$ *se calcula con los parámetros conocidos* $\theta_i(k)$ *y los datos* B_k*. Para cada elemento de* Θ_k^T *y* B_k *en (3.24) se tiene:*

$$\begin{aligned} V_{1,k+1} &= V_{1,k} + \tfrac{\lambda_k}{\bar{\varsigma}_i^2} P_{k+1}\sigma\left[W_{1,k}x(k)\right]e^T(k) \\ W_{1,k+1} &= W_{1,k} + \tfrac{\lambda_k}{\bar{\varsigma}_i^2} P_{k+1}\sigma'\left[W_{1,k}x(k)\right]V_{1,k}^T x(k)e^T(k) \end{aligned} \tag{3.31}$$

Este tiene la misma forma que el algoritmo de propagación hacia atrás [28], pero en este caso en coeficiente de aprendizaje no es una constante positiva, en este caso es una matriz positiva definida $\frac{\lambda_k}{\bar{\varsigma}_i^2} P_{k+1}$ *la cual varia con el tiempo. Esta puede ser la razón por la cual el algoritmo elipsoidal acotado es mas rápido.*

Comentario 3.4 *El algoritmo elipsoidal acotado (3.24) tiene una estructura similar a la estructura del algoritmo de filtro de Kalman extendido [71], [73], [81]. El algoritmo del filtro de Kalman extendido es:*

$$\begin{aligned} \theta_i(k+1) &= \theta_i(k) + P_k B_k\left(R_2 + B_k^T P_k B_k\right)^{-1} e_i(k) \\ P_{k+1} &= R_1 + \left[P_k - P_k B_k\left(R_2 + B_k^T P_k B_k\right)^{-1} B_k^T P_k\right] \end{aligned} \tag{3.32}$$

donde $e_i(k)$ es el mismo que se tiene en (3.15), R_1 se puede elegir como αI, donde α es una constante positiva pequeña, R_2 es la covarianza del ruido de la dinámica no modelada la cual se considera como ruido. Cuando $R_1 = 0$, este algoritmo se convierte en el algoritmo de mínimos cuadrados recursivos [26]. Si $R_1 = 0$ en (3.32), entonces $\left(R_2 + B_k^T P_k B_k\right)^{-1}$ corresponde a $\frac{\lambda_k}{(1-\lambda_k)\overline{\varsigma}_i^2 + \lambda_k B_k^T P_k B_k}$ en (3.24). Hay una gran diferencia, el algoritmo elipsoidal acotado es para el caso determinístico y el algoritmo del filtro de Kalman es para el caso estocástico.

Los siguientes pasos muestran como entrenar los pesos de la red neuronal recurrente con el algoritmo elipsoidal acotado:

1. Construir el modelo de red neuronal recurrente (3.2) para identificar el sistema no lineal (3.1). La matriz A se selecciona tal que la red neuronal es estable.

2. Se escribe la red neuronal recurrente en la siguiente forma:

$$\begin{aligned} y(k) &= B_k^T \Theta_k + \zeta(k) \\ \Theta_k &= \left[V_{1,k}, W_{1,k}^T, V_{2,k}, W_{2,k}^T\right]^T = \left[\theta_1(k), \cdots \theta_n(k)\right] \\ B_k &= \left[\sigma, \sigma' V_{1,k}^T x, \phi u, \phi' \mathrm{diag}(u(k)) V_{2,k}^T x(k)\right]^T \end{aligned}$$

3. Se entrenan los pesos de la red neuronal como sigue:

$$\begin{aligned} \theta_i(k+1) &= \theta_i(k) + \tfrac{\lambda_k}{\overline{\varsigma}_i^2} P_{k+1} B_k e_i(k) \\ B_k &= \left[\sigma, \sigma' \widehat{V}_{1,k}^T x, \phi u, \phi' \mathrm{diag}(u(k)) \widehat{V}_{2,k}^T x(k)\right]^T, \\ \lambda_k &= \begin{cases} \frac{\lambda \overline{\varsigma}_i^2}{1 + B_k^T P_k B_k}, & \text{si } e_i^2(k) \geq \frac{\overline{\varsigma}_i^2}{1-\lambda} \\ 0, & \text{si } e_i^2(k) < \frac{\overline{\varsigma}_i^2}{1-\lambda} \end{cases} \end{aligned}$$

4. P_k se actualiza con el algoritmo elipsoidal acotado:

$$P_{k+1} = \frac{1}{1-\lambda_k}\left[P_k - P_k B_k \frac{\lambda_k}{(1-\lambda_k)\overline{\varsigma}_i^2 + \lambda_k B_k^T P_k B_k} B_k^T P_k\right]$$

Con $\theta_i(1)$ que son las condiciones iniciales para los pesos y $P_1 > 0$ se inicia la identificación con la red neuronal recurrente.

3.4. Análisis de estabilidad

En esta sección se usa el análogo de la teoría de estabilidad de Lyapunov para probar que el algoritmo elipsoidal acotado propuesto (3.24) con zona muestra es estable para identificación de sistemas no lineales. Por (3.14), (3.15) y (3.19) se tiene:

$$\begin{aligned} e_i(k) &= B_k^T \widetilde{\theta}_i(k) + \varsigma_{i,k} \\ \widetilde{\theta}_i^T(k) B_k &= B_k^T \widetilde{\theta}_i(k) = e_i(k) - \varsigma_{i,k} \end{aligned} \tag{3.33}$$

donde $\widetilde{\theta}_i(k) = \theta_i^* - \theta_i(k)$, $i = 1 \cdots n$.

El teorema de la pasada sección nos dice $E_k = \left\{ \theta_i(k) \mid \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) \leq 1 \right\}$ es una elipsoide acotada siempre. Así se tiene que los pesos de la red neuronal recurrente están acotados durante el algoritmo de entrenamiento (3.24). El siguiente teorema da la cota del error de identificación.

Teorema 3.2 *Si se usa el algoritmo elipsoidal (3.24) se puede asegurar que el error de identificación $\epsilon_i(k)$ es uniformemente estable y que el error $e_i(k)$ converge a:*

$$\lim_{k \to \infty} \sup e_i^2(k) \leq \frac{\overline{\varsigma}_i^2}{1 - \lambda} \tag{3.34}$$

Demostración. Se usa la siguiente función de energía:

$$V(k) = \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) \tag{3.35}$$

Evaluando $\Delta V(k)$ como:

$$\Delta V(k) = \widetilde{\theta}_i^T(k+1) P_{k+1}^{-1} \widetilde{\theta}_i(k+1) - \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) \tag{3.36}$$

Primero, se considera cuando $\lim_{k \to \infty} \sup e_i^2(k) \geq \frac{\overline{\varsigma}_i^2}{1-\lambda}$. Así $\lambda_k = \frac{\lambda \overline{\varsigma}_i^2}{1 + B_k^T P_{k+1} B_k}$. Desde (3.15), (3.33) y (3.24), se tiene lo siguiente:

$$\widetilde{\theta}_i(k+1) = \widetilde{\theta}_i(k) - \frac{\lambda_k}{\overline{\varsigma}_i^2} P_{k+1} B_k e_i(k) \tag{3.37}$$

usando (3.37) y (3.33) se tiene:

$$\begin{aligned}\Delta V(k) &= \left[\widetilde{\theta}_i(k) - \tfrac{\lambda_k}{\overline{\varsigma}_i^2} P_{k+1} B_k e_i(k)\right]^T P_{k+1}^{-1} \left[\widetilde{\theta}_i(k) - \tfrac{\lambda_k}{\overline{\varsigma}_i^2} P_{k+1} B_k e_i(k)\right] - \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) \\ &= \widetilde{\theta}_i^T(k)\left[P_{k+1}^{-1} - P_k^{-1}\right]\widetilde{\theta}_i(k) - 2\tfrac{\lambda_k}{\overline{\varsigma}_i^2}\widetilde{\theta}_i^T(k) B_k e_i(k) + \tfrac{\lambda_k^2}{\overline{\varsigma}_i^4} B_k^T P_{k+1} B_k e_i^2(k)\end{aligned}$$

desde (3.26) se tiene $P_{k+1}^{-1} = (1-\lambda_k) P_k^{-1} + \frac{\lambda_k}{\overline{\varsigma}_i^2} B_k B_k^T$, entonces:

$$P_{k+1}^{-1} - P_k^{-1} = -\lambda_k P_k^{-1} + \frac{\lambda_k}{\overline{\varsigma}_i^2} B_k B_k^T \tag{3.38}$$

substituyendo (3.38) en $\Delta V(k)$ da:

$$\begin{aligned}\Delta V(k) &= \widetilde{\theta}_i^T(k)\left[-\lambda_k P_k^{-1} + \tfrac{\lambda_k}{\overline{\varsigma}_i^2} B_k B_k^T\right]\widetilde{\theta}_i(k) - 2\lambda_k \widetilde{\theta}_i^T(k) B_k e_i(k) + \tfrac{\lambda_k^2}{\overline{\varsigma}_i^4} B_k^T P_{k+1} B_k e_i^2(k) \\ &= -\lambda_k \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) + \tfrac{\lambda_k}{\overline{\varsigma}_i^2} \widetilde{\theta}_i^T(k) B_k B_k^T \widetilde{\theta}_i(k) - 2\tfrac{\lambda_k}{\overline{\varsigma}_i^2} \widetilde{\theta}_i^T(k) B_k e_i(k) + \tfrac{\lambda_k^2}{\overline{\varsigma}_i^4} B_k^T P_{k+1} B_k e_i^2(k)\end{aligned}$$

y substituyendo (3.33) en $\Delta V(k)$ y usando $\lambda_k = \frac{\lambda \overline{\varsigma}_i^2}{1 + B_k^T P_{k+1} B_k}$ da lo siguiente:

$$\begin{aligned}\Delta V(k) &= -\tfrac{\lambda_k}{\overline{\varsigma}_i^2}\widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) + \tfrac{\lambda_k}{\overline{\varsigma}_i^2}\left[e_i(k) - \varsigma_{i,k}\right]^2 - 2\tfrac{\lambda_k}{\overline{\varsigma}_i^2}\left[e_i(k) - \varsigma_{i,k}\right] e_i(k) + \tfrac{\lambda_k^2}{\overline{\varsigma}_i^4} B_k^T P_{k+1} B_k e_i^2(k) = \\ &-\lambda_k \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) + \tfrac{\lambda_k}{\overline{\varsigma}_i^2} e_i^2(k) - 2\tfrac{\lambda_k}{\overline{\varsigma}_i^2} e_i(k)\varsigma_{i,k} + \tfrac{\lambda_k}{\overline{\varsigma}_i^2}\varsigma_{i,k}^2 - 2\tfrac{\lambda_k}{\overline{\varsigma}_i^2} e_i^2(k) + 2\tfrac{\lambda_k}{\overline{\varsigma}_i^2} e_i(k)\varsigma_{i,k} + \tfrac{\lambda_k^2}{\overline{\varsigma}_i^4} B_k^T P_{k+1} B_k e_i^2(k) \\ &= -\lambda_k \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) - \tfrac{\lambda_k}{\overline{\varsigma}_i^2} e_i^2(k) + \tfrac{\lambda_k^2}{\overline{\varsigma}_i^4} B_k^T P_{k+1} B_k e_i^2(k) + \tfrac{\lambda_k}{\overline{\varsigma}_i^2}\varsigma_{i,k}^2 \\ &= -\lambda_k \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) - \tfrac{\lambda_k}{\overline{\varsigma}_i^2} e_i^2(k) + \tfrac{\lambda_k \overline{\varsigma}_i^2}{\overline{\varsigma}_i^4} \tfrac{B_k^T P_{k+1} B_k}{1 + B_k^T P_{k+1} B_k} \lambda e_i^2(k) + \tfrac{\lambda_k}{\overline{\varsigma}_i^2}\varsigma_{i,k}^2 \\ &\leq -\tfrac{\lambda_k}{\overline{\varsigma}_i^2} e_i^2(k) + \tfrac{\lambda_k}{\overline{\varsigma}_i^2}\lambda e_i^2(k) + \lambda_k\end{aligned}$$

entonces se tiene:

$$\Delta V(k) \leq -\frac{\lambda_k}{\overline{\varsigma}_i^2}\left[(1-\lambda) e_i^2(k) - \overline{\varsigma}_i^2\right] \tag{3.39}$$

Dada la zona muerta se tiene que $e_i^2(k) \geq \frac{\overline{\varsigma}_i^2}{1-\lambda}$ y $\lambda_k > 0$, $\Delta V(k) \leq 0$, por lo tanto $V(k)$ es acotado. Si $e_i^2(k) < \frac{\overline{\varsigma}_i^2}{1-\lambda}$ se conoce de (3.24) que $\lambda_k = 0$ y ninguno de los pesos cambia, entonces los pesos permanecen acotados, así $V(k)$ es acotado. Cuando $e_i^2(k) \geq \frac{\overline{\varsigma}_i^2}{1-\lambda}$, sumando (3.39) desde 1 a T:

$$\sum_{k=1}^{T} \frac{\lambda_k}{\overline{\varsigma}_i^2}\left[(1-\lambda) e_i^2(k) - \overline{\varsigma}_i^2\right] \leq V(1) - V(T+1) \tag{3.40}$$

ya que $V(T+1)$ es acotado:

$$\lim\sup_{k\to\infty}\sum_{k=1}^{T}\frac{\lambda\left[(1-\lambda)\,e_i^2\,(k)-\overline{\varsigma}_i^2\right]}{\overline{\varsigma}_i^2\,(1+B_k^T P_k B_k)}<\infty \tag{3.41}$$

sabiendo que $e_i^2\,(k)\geq\frac{\overline{\varsigma}_i^2}{1-\lambda}$

$$\frac{\lambda\left[(1-\lambda)\,e_i^2\,(k)-\overline{\varsigma}_i^2\right]}{\overline{\varsigma}_i^2\,(1+B_k^T P_k B_k)}\geq 0$$

así

$$\lim\sup_{k\to\infty}\frac{\lambda}{\overline{\varsigma}_i^2\,(1+B_k^T P_k B_k)}\left[(1-\lambda)\,e_i^2\,(k)-\overline{\varsigma}_i^2\right]=0$$

sabiendo que $V(k)$ está acotado, entonces $B_k^T P_{k+1} B_k<\infty$

$$\lim\sup_{k\to\infty}(1-\lambda)\,e_i^2\,(k)\leq\overline{\varsigma}_i^2 \tag{3.42}$$

Esto es (3.34). El error de entrenamiento $e_i\,(k)$ no es el mismo que el error de identificación $\epsilon_i\,(k)=x_i(k)-\widehat{x}_i(k)$, pero ambos se minimizan al mismo tiempo. Desde (3.2), (3.4), (3.13) y (3.15), se tiene:

$$\epsilon_i\,(k+1)=a_i\epsilon_i\,(k)+e_i\,(k) \tag{3.43}$$

donde $A=diag\,\{a_i\}$. Usando la relación:

$$\begin{aligned}
&\epsilon_i\,(2)=a_i\epsilon_i\,(1)+e_i\,(1)\\
&\epsilon_i\,(3)=a_i\epsilon_i\,(2)+e_i\,(2)\\
&=a_i^2\epsilon_i\,(1)+a_i e_i\,(1)+e_i\,(2)\\
&\vdots\\
&\epsilon_i\,(k+1)=a_i^k\epsilon_i\,(1)+\sum_{j=1}^{k}a_i^{k-j}e_i\,(j)
\end{aligned}$$

y que $|a_i|<1$

$$|\epsilon_i\,(k+1)|\leq|\epsilon_i\,(1)|+\sum_{j=1}^{k}|e_i\,(j)|$$

ya que $\epsilon_i\,(1)$ es una constante, la minimización del error de entrenamiento $e_i\,(j)$ es la minimización de la cota superior del error de identificación $\epsilon_i\,(k+1)$. Cuando $e_i\,(j)$ es acotado, ϵ_i también está acotado. ■

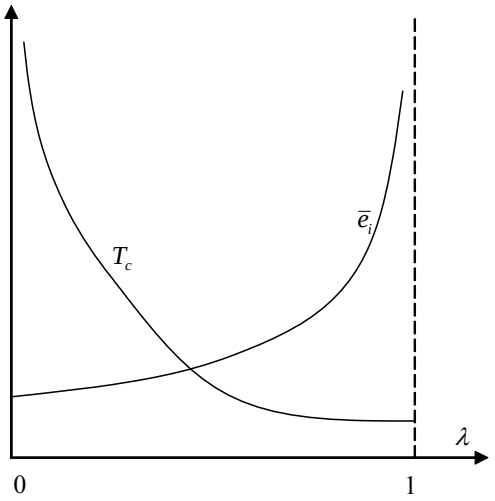

Figura 3.3: Caracteristica de convergencia del error de identificación

Comentario 3.5 *Aunque los parámetros convergen a sus valores óptimos con el algoritmo de entrenamiento (3.24), desde (3.4) se conoce que siempre existe error de la incertidumbre estructural (error en la estructura). Así que no se puede alcanzar que* $e_i(k) \to 0$.

Comentario 3.6 λ *es el parámetro llamado coeficiente de aprendizaje y determina la velocidad de aprendizaje de la red neuronal, cuando* λ *es grande se tiene una cota grande para la incertidumbre estructural* $\overline{\varsigma}_i^2$ *dada en (3.34) y la red neuronal aprende rápidamente el comportamiento de la función no lineal* f*, cuando* λ *es pequeño se tiene una cota pequeña para la incertidumbre estructural* $\overline{\varsigma}_i^2$ *dada en (3.34) y la red neuronal aprende lentamente el comportamiento de la función no lineal* f*. Se tienen las siguientes condiciones extremas: cuando* $\lambda = 1$*, el tiempo de convergencia del error de identificación* T_c *es mínimo, pero la cota superior del error de identificación es infinitamente grande,esto es,* $\overline{e_i} = \frac{\overline{\varsigma}_i^2}{1-\lambda} \to \infty$*; cuando* $\lambda = 0$*, la cota superior del error de identificación* $\overline{e_i} = \frac{\overline{\varsigma}_i^2}{1-\lambda}$ *es mínima, pero el tiempo de convergencia del error de identificación* T_c *es infinitamente grande,esto es,* $T_c \to \infty$*, ver Figura 3.3.*

3.5. Simulaciones

A continuación se tratan dos modelos para comprobar el desempeño del identificador propuesto.

3.5.1. Identificador propuesto aplicado al modelo de tráfico

Una avenida emplea el método de densidad de bloque híbrido como su modelo de flujo de tráfico. El cálculo de flujo de éste método se basa en el método de densidad de bloque [9], el cual trata el flujo de tráfico de manera continua. En el método de densidad de bloque, cada vía se divide en varios bloques de los cuales su longitud es igual a la distancia a la que el vehículo corre a velocidad de flujo libre en la vía durante las iteraciones consideradas en cada intervalo de tiempo. El flujo de entrada y salida y la densidad de cada bloque se revisan en cada bloque en cada intervalo de tiempo basándose en la ley de conservación de flujo y la relación $f-\rho$ mostradas en la Figura 3.4. Este método puede razonablemente reproducir las condiciones de tráfico de movimiento libre y de embotellamiento incluyendo la propagación de los carros en los bloques [31], [32]. La Figura 3.5 muestra las variables usadas en el cálculo de flujo entre dos bloques vecinos con sufijos i e $i+1$. En cada iteración, el flujo permitido de salida y entrada a cada bloque se calculan acorde a la relación $f-\rho$, entonces el flujo entre dos bloques vecinos se determina al tomar el mínimo del flujo permitido a la salida del bloque i y el flujo permitido a la entrada del bloque $i+1$. Las densidades en la siguiente iteración se calculan usando el flujo de entrada y de salida de cada bloque. Las siguientes ecuaciones muestran el cálculo de flujo de cada bloque con sufijo i [31], [32].

$$\begin{gathered} A_i^{out}(k) = \text{mín}(\rho_{ci}, \rho_i(k))\frac{\Delta L}{\Delta k} \\ A_i^{in}(k) = \begin{cases} \text{mín}(\rho_{ci}, \rho_{Ji} - \rho_i(k))\frac{\Delta L}{\Delta k} & \text{si } \rho_i(k) \leq \rho_{ci} \\ \frac{\rho_{ci}(\rho_{Ji}-\rho_i(k))}{\rho_{Ji}-\rho_{ci}}\frac{\Delta L}{\Delta k} & \text{si } \rho_i(k) > \rho_{ci} \end{cases} \\ f_{i,i+1}(k) = \text{mín}\left(A_{i+1}^{in}(k), A_i^{out}(k)\right) \\ \rho_i(k+1) = \rho_i(k) - f_{i,i+1}(k)\frac{\Delta k}{\Delta L} + f_{i-1,i}(k)\frac{\Delta k}{\Delta L} \end{gathered} \tag{3.44}$$

donde $A_i^{out}(k)$ es el flujo permitido a la salida del bloque i, $A_i^{in}(k)$ es el flujo permitido a la entrada del bloque i, $\rho_i(k)$ es la densidad del bloque i, $f_{i,i+1}(k)$ es el flujo desde el

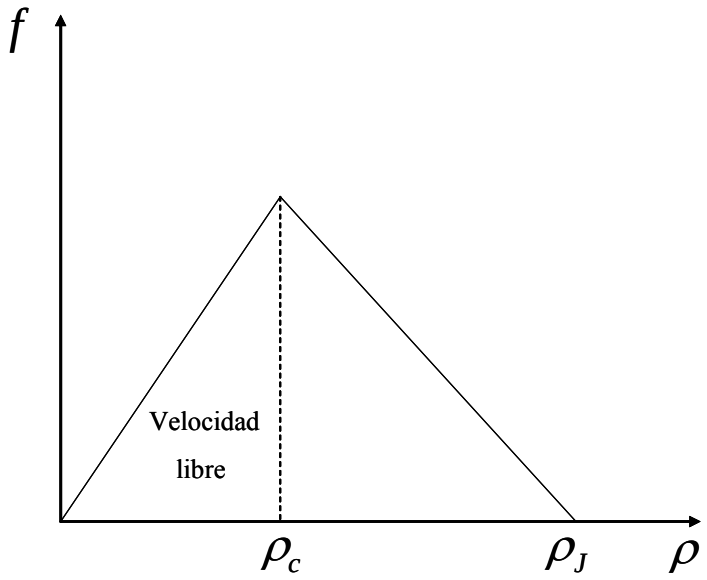

Figura 3.4: Relación densidad-flujo utilizada en el método de densidad de bloque

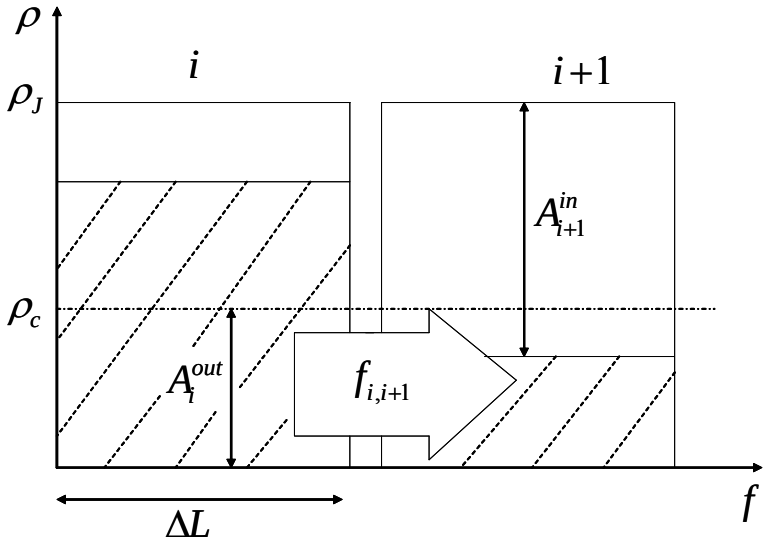

Figura 3.5: Cálculo del flujo entre bloques vecinos

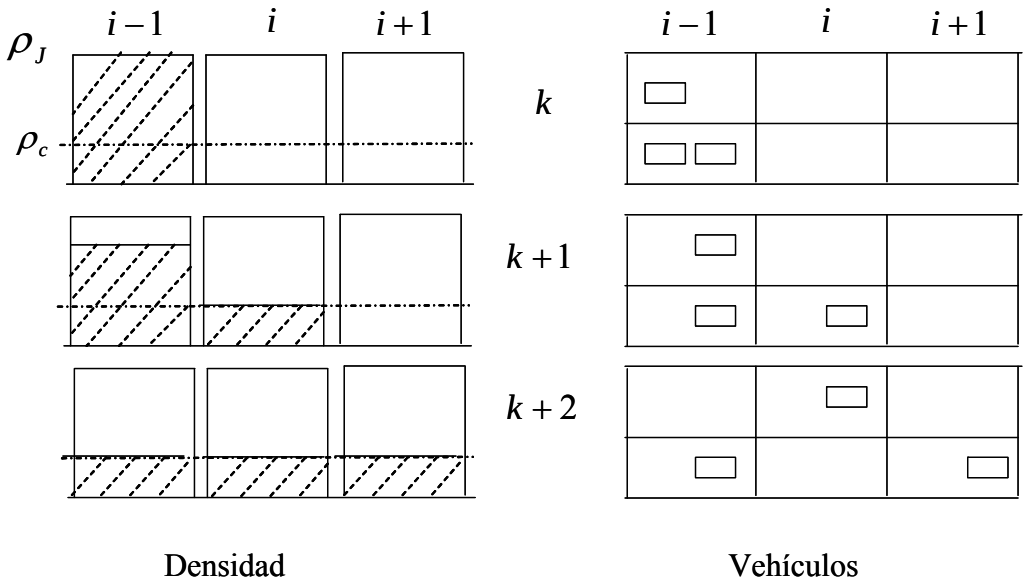

Figura 3.6: Movimiento de los vehiculos en tiempo discreto

bloque i hacia el bloque $i+1$, ρ_{ci} es la densidad critica del bloque i, ρ_{Ji} es la densidad de embotellamiento, ΔL es la longitud del bloque, Δk es el intervalo de tiempo para cada iteración. Ya que la aproximación de flujo de tráfico tiene algunas dificultades en el manejo de vehículos individuales tal como es su bloque destino [9], [20], el método de densidad de bloque fue extendido hacia el método de densidad de bloque híbrido, es decir, los bloques no solamente contienen densidad de tráfico continuo, también tienen imágenes discretas de vehículos que contienen diferentes atributos como son el tipo de vehículos, el destino y el criterio de elección de la ruta para la conveniencia del modelado del comportamiento del vehículo. La Figura 3.6 ilustra los movimientos de los vehículos en tiempo discreto en este método. El número de vehículos que se mueven desde el bloque i hacia el bloque $i+1$, es proporcional a el flujo que se tiene desde el bloque i al bloque $i+1$. El numero de vehículos que permanece entre los bloques i e $i+1$ en la iteración $k+1$ será considerado como k para la siguiente iteración.

El sistema no lineal (3.44) tiene la forma (3.1), donde $i = 1, 2, \ldots, 100$, $\rho_{ci} = 6\ \forall i$, $\rho_{Ji} =$

$10\ \forall i$, $\Delta k = 20$ segundos y $\Delta L = 40$ metros, esto es, cada bloque tiene 40 metros de largo, se toman datos cada 20 segundos, cuando se tiene menos de 6 vehículos en el bloque de 40 metros se tiene movimiento libre, pero cuando se tiene más de 6 vehículos en el bloque de 40 metros se tiene un embotellamiento, cada $x(k) = [x_1(k), x_2(k), \ldots, x_{100}(k)]^T = [\rho_1(k), \rho_2(k), \ldots, \rho_{100}(k)]^T$, Se usa la red neuronal dada en (3.2) para identificar este sistema no lineal con $\widehat{x}(k) \in R^{100}$, $A \in \Re^{100\times100}$ es una matriz diagonal estable definida como $A = diag(0{,}1)$. En esta sección, con el fin de examinar la efectividad del algoritmo elipsoidal acotado para el entrenamiento, se usan 20 nodos en la capa oculta. Los pesos en la capa de salida son $V_{1,k} \in \Re^{100\times20}$, los pesos en la capa oculta son $W_{1,k} \in \Re^{20\times100}$, $\sigma = [\sigma_1, \sigma_2, \ldots, \sigma_{100}]^T$. Los elementos de los pesos iniciales $W_{1,0}$, y $V_{1,0}$, se eligen como números aleatorios entre $(0, 1x10^{-3})$ y $(0, 1x10^{-1})$ respectivamente. La entrada es $f_{i-1,i}(1) = 2$, $f_{i-1,i}(7) = 4$, $f_{i-1,i}(14) = 6$, $f_{i-1,i}(20) = 4$, $f_{i-1,i}(26) = 6$, $f_{i-1,i}(32) = 4$; $f_{i-1,i}(39) = 6$, $f_{i-1,i}(45) = 4$ y $f_{i-1,i}(52) = 2$. Se selecciona $P = diag(100) \in \Re^{40\times40}$, $\lambda = 0{,}5$, $\overline{\varsigma}_i^2 = 1$ tal que $\lambda\overline{\varsigma}_i^2 < 1$. Se define el error promedio cuadrático para un tiempo finito como $J(N) = \frac{1}{2N}\sum_{k=1}^{N} e^2(k)$.

Ahora se compara el algoritmo elipsoidal acotado propuesto dado en (3.24) con el algoritmo de propagación hacia atrás [28] con coeficiente de aprendizaje de 0,01 para entrenar la red neuronal dada en (3.2). El resultado de identificación para $x_{25}(k)$ se muestra en la Figura 3.7. El resultado de identificación para $J(N)$ se muestra en la Figura 3.8. Se aprecia que el algoritmo elipsoidal acotado tiene un mejor desempeño que el algoritmo de propagación hacia atrás.

3.5.2. Identificador propuesto aplicado al proceso de combustión de gas

Los datos del proceso de gas de Box y Jenkin se usan frecuentemente para evaluar los métodos de identificación [46], el proceso de muestreo consiste de 200 datos de entrada-salida tomados con un periodo de muestreo de 9 segundos. El proceso de combustión de gas tiene una variable de entrada que es el flujo de gas, $u(k)$, y una variable de salida que es la concentración de CO_2 , $y(k)$. El salida se puede considerar influenciada por las variables

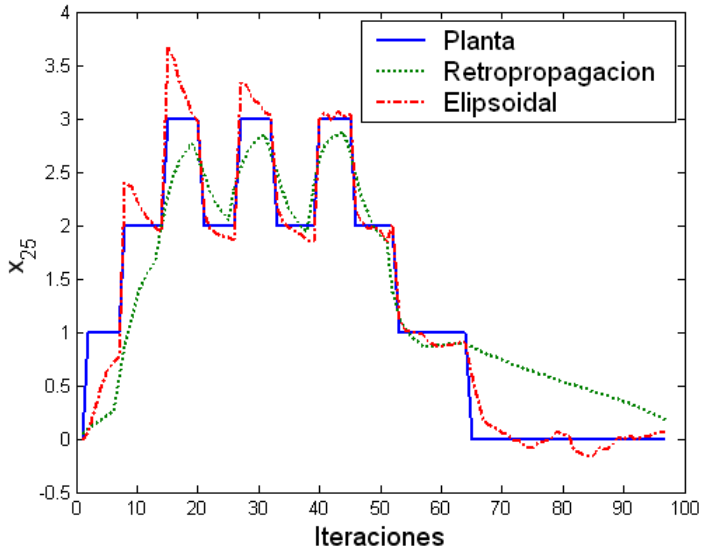

Figura 3.7: Identificación del estado $x_{25}(k)$ para el ejemplo 1

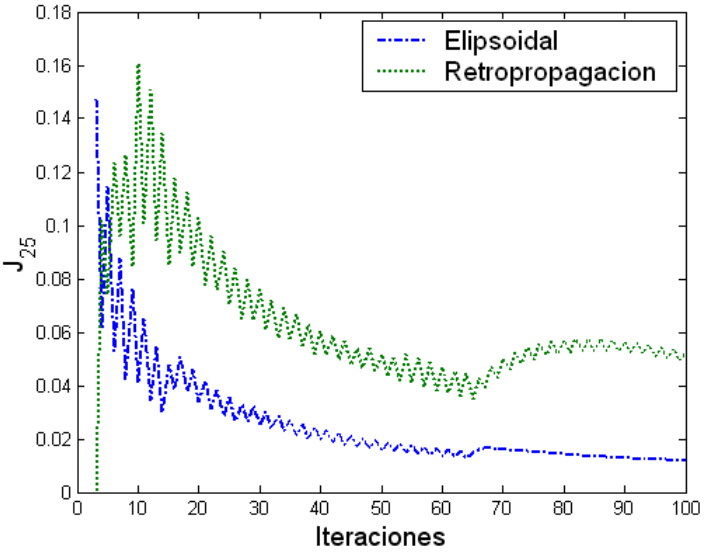

Figura 3.8: Error promedio cuadratico para el identificador del ejemplo 1

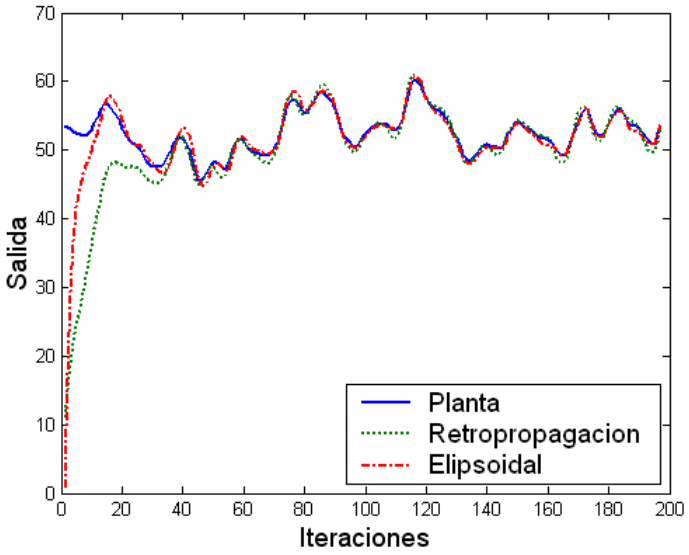

Figura 3.9: Identificación para $y(k)$ del ejemplo 2

$y(k-1), y(k-2), \ldots, y(k-4), u(k-1), u(k-2), \ldots, u(k-4)$. Se usa la red neuronal dada en (3.1) con la estructura entrada-salidas para identificar este sistema no lineal, se usan 16 nodos en la capa oculta, *i.e.,* $V_k \in \Re^{8\times 16}$, $W_{1,k} \in \Re^{16\times 8}$, $\sigma = [\sigma_1, \sigma_2, \ldots, \sigma_{16}]^T$. Los pesos iniciales W_1 y V_1 se eligen como aleatorios entre $(0,1)$ y $(0,0{,}2)$ respectivamente. Se selecciona $P(1) = diag(100) \in \Re^{32\times 32}$, $\lambda = 0{,}5$, $\overline{\varsigma}^2 = 1$ tal que $\lambda\overline{\varsigma}^2 < 1$. Se compara el algoritmo elipsoidal (3.24) con la estructura entradas salidas con el algoritmo de propagación hacia atrás [28] con coeficiente de apredizaje de 0,5. Se define el error promedio cuadrático para un tiempo finito como $J(N) = \frac{1}{2N}\sum_{k=1}^{N} e^2(k)$. El resultado de identificación para la salida $y(k)$ se muestra en la Figura 3.9. El resultado de identificación para $J(N)$ se muestra en la Figura 3.10.

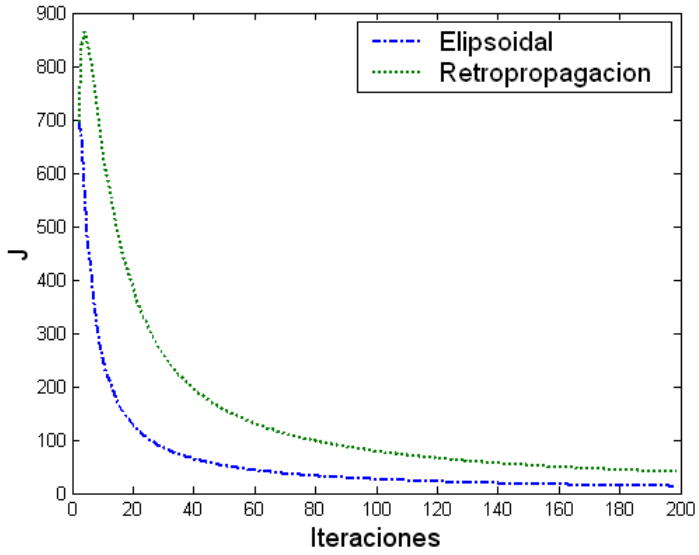

Figura 3.10: Error promedio cuadratico para el identificador del ejemplo 2

3.6. Conclusión

En esta sección se propone un método de identificación para redes neuronales. Se modifica el algoritmo elipsoidal acotado con un coeficiente de aprendizaje variante en tiempo para entrenar una red neuronal. Se actualizan tanto capa oculta como capa de salida de una red neuronal recurrente en espacio de estados. Se prueba que el error de identificación converge a una zona la cual depende del error de la incertidumbre estructural.

Capítulo 4

Control por modos deslizantes con redes neuronales

4.1. Introducción

El método de modos deslizantes tradicional es la aplicación de una señal de control conmutando a alta frecuencia que consigue llevar el estado del sistema a una superficie cercana al origen denominada superficie de deslizamiento o función de conmutación y una vez llegando a la superficie de deslizamiento se debe permanecer en ésta ante posibles perturbaciones externas. Dicha superficie de deslizamiento debe ser definida por el diseñador con el objeto de que el estado cumpla las especificaciones deseadas. La principal ventaja del control por modos deslizantes es que aporta robustez ante perturbaciones cuando estas tienen cotas conocidas. En este documento se trata con un control por modos deslizantes en tiempo discreto el cual no tiene el mismo comportamiento que el control por modos deslizantes en tiempo continuo. El control por modos deslizantes en tiempo discreto tiene la ventaja de que puede generar o no generar castañeo a diferencia del control por modos deslizantes en tiempo continuo que siempre genera castañeo [82]. El castañeo se genera debido a la discontinuidad en la acción de control y el castañeo es indeseable en la mayoría de las aplicaciones [11]. En

tiempo continuo, la condición en modos deslizantes es:

$$s\dot{s} < 0 \tag{4.1}$$

donde s es la función de conmutación, la función de conmutación conmuta a cada instante cuando las trayectorias del error de seguimiento de trayectoria cruzan al origen.

La condición de modos deslizantes más simple es substituir la derivada de (4.1) por una diferencia, así la condición de modos deslizantes es [23]:

$$s(k)\left[s(k+1) - s(k)\right] < 0 \tag{4.2}$$

Esta condición es necesaria pero no suficiente para la existencia de un movimiento en modos deslizantes [22], [82]. Además, ésta condición no puede asegurar la convergencia del error de seguimiento a una región cercana al origen, y esto podría incrementar la amplitud del castañeo [85]. Una condición de modos deslizantes necesaria y suficiente en tiempo discreto es la propuesta en [77] y [34] es:

$$|s(k+1)| < |s(k)| \tag{4.3}$$

Esta condición puede asegurar la convergencia en la función de conmutación del error de seguimiento de trayectoria a una región cercana al origen, pero es muy dificil diseñar un controlador que satisfaga esta condición. Así [2] da otra condición de modos deslizantes en tiempo discreto, la superficie de deslizamiento $s(k) = 0$ debe satisfacer:

$$|s(k)| \leq q \tag{4.4}$$

donde el parámetro $q > 0$ es llamado ancho de banda de los modos deslizantes. Este no requiere que la trayectoria cruce la superficie de deslizamiento en cada paso de control.

El diseño de los controladores por modos deslizantes mencionados requiere de información parcial para los controles equivalentes [75], es decir, cuando el controlador equivalente se diseña cuidadosamente, la función de conmutación satisface (4.2), (4.3) y (4.4). Si se desconoce la planta, se pueden usar redes neuronales para estimar la planta. [19] propone un control por modos deslizantes con una red neuronal en tiempo discreto para el caso de regulación, se

prueba la estabilidad de la regulación usando la técnica de Lyapunov. [55] da un controlador por modos deslizantes en tiempo discreto con una función de conmutación modificada con el fin de mantener un castañeo pequeño y se adiciona un término adaptable para mejorar su desempeño. Para mejorar la robustez del controlador por modos deslizantes en tiempo discreto, [51] usa redes neuronales difusas para compensar las incertidumbres. [85] considera la regulación de sistemas lineales en espacio de estados con control por modos deslizantes. [22] presenta un controlador por modos deslizantes en tiempo discreto estable el cual no es sensible al intervalo de muestreo. Los autores mencionados no consideran la estabilidad del modelo neuronal y el problema de que se puedan presentar singularidades en la ley de control, además consideran que la ganancia del control por modos deslizantes es constante.

En este capitulo se presenta un control por modos deslizantes con un modelador por redes neuronales. El modelador por redes neuronales usa doble zona muerta y una modificación para asegurar la estabilidad del error de modelado y la no singularidad del controlador. Con el fin de reducir el castañeo, se modifica el control por modos deslizantes en tiempo discreto con una ganancia variante en tiempo. Se da una condición necesaria para la existencia de la función de conmutación en tiempo discreto, la cual satisface la condición (4.4). Se prueba que el sistema en lazo cerrado por modos deslizantes y el modelador por redes neuronales es uniformemente estable, se obtiene el resultado de que el error de seguimiento de trayectoria está acotado y esta cota depende de la incertidumbre estructural.

4.2. Sistema no lineal en tiempo discreto

Se considera el sistema con una entrada y una salida descrito como sigue:

$$y(k+1) = f\left[z(k)\right] + g\left[z(k)\right]\psi\left(u(k)\right) \tag{4.5}$$

donde $z(k) = \left[y(k), y(k-1)\ldots y(k-n), u(k-1), u(k-2)\ldots u(k-m)\right]^T$, n es el orden de $y(k)$ y m es el orden de $u(k)$, $m < n$, y es la salida, u es la entrada, $f(\cdot)$ y $g(\cdot)$ son funciones

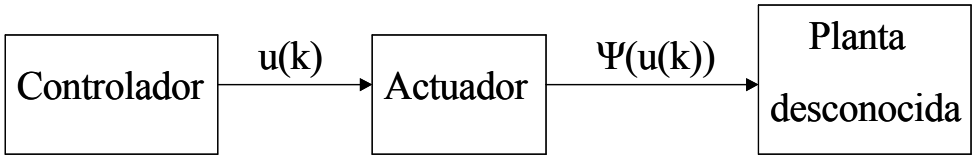

Figura 4.1: Comportamiento del actuador en el sistema de control

reales desconocidas. $\psi(\cdot)$ es una función descrita como sigue:

$$\psi(u(k)) = \begin{cases} u(k) & \text{si } |u(k)| \leq b \\ b & \text{si } |u(k)| > b \end{cases} \tag{4.6}$$

donde b es un parámetro dado por las restricciones del actuador. Ver la Figura 4.1. El sistema no lineal se asume que es controlable y observable. A (4.5) se le llama sistema afín. Se asume que $f[z(k)]$ y $g[z(k)]$ son funciones continuas y diferenciables, $g[z(k)]$ es acotada y diferente de cero:

$$0 < \underline{g} \leq |g(z(k))| \leq \overline{g} < \infty \tag{4.7}$$

(4.5) se puede escribir en espacio de estados [38]. Sea $z_i(k) = y(k-n+i)$ para $(1 \leq i \leq n)$ y $z_i(k) = u(k-n-m-1+i)$ para $(n+1 \leq i \leq n+m)$.

$$\begin{aligned} &z_1(k+1) = z_2(k) && z_{n+1}(k+1) = z_{n+2}(k) \\ &\vdots && \vdots \\ &z_n(k+1) = f[z(k)] + g[z(k)]\psi(u(k)) && z_{n+m}(k+1) = \psi(u(k)) \\ &y(k) = z_n(k) \end{aligned} \tag{4.8}$$

(4.8) se puede escribir como un sistema en espacio de estados:

$$z(k+1) = f_u[z(k), u(k)], \quad y(k) = h[z(k)] \tag{4.9}$$

donde $z(k) \in \Re^{n+m}$ es el vector de estados. f_u y h son funciones no lineales suaves $f_u, h \in C^\infty$.

Desde (4.9) se tiene:

$$\begin{aligned} &y(k) = h[z(k)] = F_1[z(k)], \\ &y(k+1) = h[f_u[z(k), u(k)]] = F_2[z(k), u(k)] \\ &y(k+n-1) = F_n[z(k), u(k), u(k+1) \cdots u(k+n-2)] \end{aligned} \tag{4.10}$$

Denotando $Y(k) = [y(k), y(k+1), \cdots y(k+n-1)]^T$, $U(k) = [u(k), u(k+1), \cdots u(k+n-2)]^T$, así $Y(k) = F[x(k), U(k)]$, $F = [F_1 \cdots F_n]^T$. Si $\frac{\partial Y}{\partial x}$ es no singular en $x = 0$, $U = 0$, (4.10) se puede expresar como $x(k+1) = g[Y(k+1), U(k+1)]$. Así (4.9) se puede transformar en un modelo NARMAX [95]

$$y(k) = h[z(k)] = \Phi[X(k)] \tag{4.11}$$

donde $X(k) = [y(k), y(k-1) \ldots y(k-n), u(k-1), u(k-2) \ldots u(k-m)]^T$. $\Phi(\cdot)$ es una ecuación de diferencias no lineal que representa la dinámica de la planta, $u(k)$ y $y(k)$ son entrada y salida escalares medibles, d es un retardo en tiempo. [16] muestra que un sistema no lineal general (4.11) se puede transformar en un sistema afín (4.5).

4.3. Modelado de funciones no lineales usando redes neuronales

4.3.1. Algoritmo con zona muerta y su modificación

Si las funciones no lineales f y g en (4.5) son desconocidas, se pueden usar redes neuronales multicapa para aproximarlas. Se definen $\widehat{f}$ y $\widehat{g}$ como las estimaciones de f y g, respectivamente, estas se pueden expresar como:

$$\begin{aligned} \widehat{f} &= \sum_{j=1}^{q} v_j^f(k)\phi_j^f\left(\sum_{i=1}^{n+m} w_{ji}^f(k) z_i(k)\right) \\ \widehat{g} &= \sum_{j=1}^{q} v_j^g(k)\phi_j^g\left(\sum_{i=1}^{n+m} w_{ji}^g(k) z_i(k)\right) \end{aligned} \tag{4.12}$$

donde $v_j^f(k)$ y $w_{ji}^f(k)$ son lo pesos de la capa de salida y oculta, respectivamente, q es el número de capas ocultas. Las funciones de activación de las redes neuronales se eligen como sigue: $\phi_j^f(\cdot)$ es una función tangente hiperbólica, esta puede ser positiva y negativa. $\phi_j^g(\cdot)$ es una función sigmoidal, esta es positiva, así $\phi_j^g(\cdot) > 0$. Las funciones no lineales f y g se

pueden modelar como sigue:

$$\begin{aligned} f &= \sum_{j=1}^{q} v_j^{f*} \phi_j^f \left(\sum_{i=1}^{n+m} w_{ji}^{f*} z_i(k) \right) + \epsilon_f \\ g &= \sum_{j=1}^{q} v_j^{g*} \phi_j^g \left(\sum_{i=1}^{n+m} w_{ji}^{g*} z_i(k) \right) + \epsilon_g \end{aligned} \tag{4.13}$$

donde v_j^{f*} y w_{ji}^{f*} son los pesos óptimos los cuales minimizan el error de la incertidumbre estructural ϵ_f, w_{ji}^{g*} y v_j^{g*} son los pesos óptimos los cuales minimizan el error de la incertidumbre estructural ϵ_g.

Ya que las funciones no lineales f y g son aproximadas por $\widehat{f}$ y $\widehat{g}$, respectivamente, el sistema no lineal (4.5) se puede modelar con una red neuronal como sigue:

$$\beta \widehat{y}(k+1) = \widehat{f}\left[z(k)\right] + \widehat{g}\left[z(k)\right] \psi\left(u(k)\right) \tag{4.14}$$

donde β es una constante positiva $\beta > 0$ la cual es un parámetro de diseño. El sistema no lineal (4.5) se puede expresar con la siguiente red neuronal:

$$\beta y(k+1) = \left[\sum_{j=1}^{q} v_j^{f*} \phi_j \left(\sum_{i=1}^{n+m} w_{ji}^{f*} z_i(k) \right) + \epsilon_f \right] + \left[\sum_{j=1}^{q} v_j^{g*} \phi_j \left(\sum_{i=1}^{n+m} w_{ji}^{g*} z_i(k) \right) + \epsilon_g \right] \psi\left(u(k)\right) \tag{4.15}$$

Para la función derivable $\widehat{f}$, se aplica la serie de Taylor a (4.12) cercano a los puntos $w_{ji}^f(k) z_i(k)$ y v_j^f:

$$\begin{aligned} \widehat{f} = \sum_{j=1}^{q} v_j^{f*} \phi_j \left(\sum_{i=1}^{n+m} w_{ji}^{f*} z_i(k) \right) &+ \left[v_j^f(k) - v_j^{f*} \right] \frac{\partial \widehat{f}}{\partial \left[w_j^f(k) \right]} \\ &+ \left[w_{ji}^f(k) z_i(k) - w_{ji}^{f*} z_i(k) \right] \frac{\partial \widehat{f}}{\partial \left[w_{ji}^f(k) z_i(k) \right]} + R_f \end{aligned} \tag{4.16}$$

donde R_f es el residuo de la formula de Taylor,

$$\begin{aligned} \frac{\partial \widehat{f}}{\partial \left[w_j^f(k) \right]} &= \phi_j^f \left(\sum_{i=1}^{n+m} w_{ji}^f(k) z_i(k) \right) = \phi_j^f \left[v_j^f(k) \right] \\ \frac{\partial \widehat{f}}{\partial \left[w_{ji}^f(k) z_i(k) \right]} &= v_j^f(k) \frac{\partial \phi_j^f \left(\sum_{i=1}^{n+m} w_{ji}^f z_i(k) \right)}{\partial \left[w_{ji}^f(k) z_i(k) \right]} = w_j^f(k) \dot{\phi}_j^f \end{aligned}$$

Se define $\widetilde{f} = \widehat{f} - f$, desde (4.12), (4.13) y (4.16) se tiene:

$$\widetilde{f} = v_j^f(k)\widetilde{w}_{ji}^f(k)z_i(k)\overset{\cdot f}{\phi}_j + \widetilde{v}_j^f(k)\phi_j^f + \zeta_f \tag{4.17}$$

donde $\widetilde{w}_{ji}^f(k) = w_{ji}^f(k) - w_{ji}^{f*}$, $\widetilde{v}_j^f(k) = v_j^f(k) - v_j^{f*}$, $\zeta_f = R_f - \epsilon_f$. Similarmente para $\widetilde{g} = \widehat{g} - g$ se tiene:

$$\widetilde{g} = v_j^g(k)\widetilde{w}_{ji}^g(k)z_i(k)\overset{\cdot g}{\phi}_j + \widetilde{v}_j^g(k)\phi_j^g + \zeta_g \tag{4.18}$$

donde $\widetilde{w}_{ji}^g(k) = w_{ji}^g(k) - w_{ji}^{g*}$, $\widetilde{v}_j^g(k) = v_j^g(k) - v_j^{g*}$, $\zeta_g = R_g - \epsilon_g$.

Se define el error de modelado del sistema como:

$$e_i(k+1) = \widehat{y}(k+1) - y(k+1) \tag{4.19}$$

La dinámica del error de puede obtener desde (4.14) y (4.15):

$$\begin{aligned}\beta e_i\left(k+1\right) = \widetilde{v}_j^f(k)\phi_j^f + v_j^f(k)\widetilde{w}_{ji}^f(k)z_i(k)\overset{\cdot f}{\phi}_j + \widetilde{v}_j^g(k)\phi_j^g\psi\left(u(k)\right) \\ + v_j^g(k)\widetilde{w}_{ji}^g(k)z_i(k)\overset{\cdot g}{\phi}_j\psi\left(u(k)\right) + \zeta\left(k\right)\end{aligned} \tag{4.20}$$

donde $\zeta\left(k\right) = \zeta_f + \zeta_g\psi\left(u(k)\right)$ es el error de la incertidumbre estructural el cual está acotado dada (4.6).

Con el fin de asegurar la estabilidad del modelado y la no singularidad en el controlador ($\widehat{g} \neq 0$), se usa la siguiente ley de aprendizaje para actualizar los pesos del modelador por redes neuronales (4.12):

$$\begin{aligned}v_j^f(k+1) &= v_j^f(k) - \eta(k)\phi_j^f e_i\left(k\right) \\ w_{ji}^f(k+1) &= w_{ji}^f(k) - \eta(k)v_j^f(k)\overset{\cdot f}{\phi}_j z_i(k)e_i\left(k\right) \\ w_{ji}^g(k+1) &= w_{ji}^g(k) - \eta(k)v_j^g(k)\overset{\cdot g}{\phi}_j z_i(k)\psi\left(u(k)\right)e_i\left(k\right) \\ \overline{v}_j^g(k+1) &= v_j^g(k) - \eta(k)\phi_j^g\psi\left(u(k)\right)e_i\left(k\right)\end{aligned} \tag{4.21}$$

donde $j = 1 \ldots q$, $i = 1 \ldots n+m$. Se aplica la doble zona muerta a $\eta(k)$ como:

$$\eta(k) = \begin{cases} \dfrac{\eta}{1+a(k)} & \text{si } e_i^2\left(k\right) \geq \frac{\overline{\zeta}^2}{1-\eta} \text{ y } \beta e_i\left(k\right)e_i\left(k+1\right) \geq e_i^2\left(k\right) \\ 0 & \text{si } e_i^2\left(k\right) < \frac{\overline{\zeta}^2}{1-\eta} \text{ o } \beta e_i\left(k\right)e_i\left(k+1\right) < e_i^2\left(k\right) \end{cases} \tag{4.22}$$

donde $a(k) = \phi_j^{f2} + \left|\dot{\phi}_j^{f} v_j^f(k) z_i(k)\right|^2 + \left|\psi\left(u(k)\right)\phi_j^g\right|^2 + \left|\psi\left(u(k)\right)\dot{\phi}_j^{g} v_j^g(k) z_i(k)\right|^2$, $0 < \eta < 1$. $\overline{\zeta}$ es la cota superior de $\zeta\left(k\right)$, $\left|\zeta\left(k\right)\right| \leq \overline{\zeta}$.

El algoritmo de proyección se aplica a $v_j^g(k+1)$.

$$v_j^g(k+1) = \begin{cases} \overline{v}_j^g(k+1) & \text{si } \left|\overline{v}_j^g(k+1) - v_j^g(1)\right| \leq M \\ \frac{\overline{v}_j^g(k+1) - v_j^g(1)}{\left|\overline{v}_j^g(k+1) - v_j^g(1)\right|} M + v_j^g(1) & \text{si } \left|\overline{v}_j^g(k+1) - v_j^g(1)\right| > M \end{cases} \tag{4.23}$$

donde $M > 0$. Desde la definición de (4.12) se conoce que $\phi_j^g\left(\cdot\right) > 0$. Con el fin de obtener $\widehat{g}\left[z(k)\right] > 0$, se necesita asegurar $v_j^g(k) > 0$. La proyección (4.23) asegura esta condición.

4.3.2. Análisis de estabilidad de la identificación

Comparado a otros algoritmos de entrenamiento convencionales, tales como el algoritmo de propagación hacia atrás [74], la ley de actualización de los pesos (4.21) es más compleja. En este documento se usan dos técnicas juntas: zona muerta [8] y una modificación [88]. En [8] la zona muerta se usa en la ley de control, en este documento de usa doble zona muerta para la actualización de los pesos y asegura que el error de modelado está acotado. Con el fin de asegurar que $\widehat{g}$ es diferente de cero, se usa la modificación en la doble zona muerta. Se puede ver que la doble zona muerta y la modificación se aplican cuando el proceso de aprendizaje alcanza algunas condiciones extremas. El aprendizaje normal (4.21) tiene la misma forma que el algoritmo de propagación hacia atrás [74], la única diferencia es que se usa coeficiente de aprendizaje variante en tiempo $\eta(k)$, [38], [56], [74] usan coeficiente de aprendizaje fijo. El coeficiente de aprendizaje variante en tiempo asegura la estabilidad del modelado. El parámetro η que esta en el coeficiente de aprendizaje variante en tiempo es fácil de conseguir, no se requiere ninguna información previa, por ejemplo se puede seleccionar $\eta = 0{,}5$. El siguiente teorema da la estabilidad del modelador por redes neuronales y la no singularidad en el controlador con el algoritmo (4.21).

Teorema 4.1 *Si se usa el modelo de red neuronal (4.14) para modelar el sistema no lineal (4.5), la ley de aprendizaje (4.21) con zona muerta (4.22) y su modificación (4.23) hace*

el modelado estable y no se tiene singularidad en el controlador, es decir, 1) el error de modelado $e_i(k)$ *está acotado y* $\widehat{g}[z(k)] \neq 0$, *2) El error de modelado* $e_i(k)$ *satisface:*

$$\lim_{k\to\infty} \sup e_i^2(k) \leq \frac{\overline{\zeta}^2}{1-\eta} \tag{4.24}$$

donde $\overline{\zeta}$ *es la cota superior de* $\zeta(k)$.

Demostración. Se selecciona la siguiente función de energía $V_i(k)$ como:

$$V_i(k) = \widetilde{v}_j^f(k)^2 + \widetilde{w}_{ji}^f(k)^2 + \widetilde{v}_j^g(k)^2 + \widetilde{w}_{ji}^g(k)^2 \tag{4.25}$$

1) Primero se considera $\left|\overline{v}_j^g(k+1) - v_j^g(1)\right| \leq M$. Desde (4.23), se sabe que $v_j^g(k+1) = \overline{v}_j^g(k+1)$, así $\left|v_j^g(k+1) - v_j^g(1)\right| \leq M$, $v_j^g(k+1)$ esta en la bola con centro $v_j^g(1)$ y radio M. Por la actualización (4.21), se tiene:

$$\begin{aligned}
\widetilde{v}_j^f(k+1) &= \widetilde{v}_j^f(k) - \eta(k)\phi_j^f e_i(k) \\
\widetilde{w}_{ji}^f(k+1) &= \widetilde{w}_{ji}^f(k) - \eta(k)v_j^f(k)\overset{\cdot}{\phi}{}_j^f z_i(k) e_i(k) \\
\widetilde{v}_j^g(k+1) &= \widetilde{v}_j^g(k) - \eta(k)\phi_j^g \psi(u(k))\, e_i(k) \\
\widetilde{w}_{ji}^g(k+1) &= \widetilde{w}_{ji}^g(k) - \eta(k)v_j^g(k)\overset{\cdot}{\phi}{}_j^g z_i(k)\psi(u(k))\, e_i(k)
\end{aligned}$$

Ahora se calcula $\Delta V_i(k)$:

$$\begin{aligned}
&\Delta V_i(k) = \left[\widetilde{v}_j^f(k) - \eta(k)\phi_j^f e_i(k)\right]^2 - \widetilde{w}_j^f(k)^2 + \left[\widetilde{w}_{ji}^f(k) - \eta(k)v_j^f(k)\overset{\cdot}{\phi}{}_j^f z_i(k)e_i(k)\right]^2 - \widetilde{w}_{ji}^f(k)^2 + \\
&\left[\widetilde{v}_j^g(k) - \eta(k)\phi_j^g\psi(u(k))\, e_i(k)\right]^2 - \widetilde{v}_j^g(k)^2 + \left[\widetilde{w}_{ji}^g(k) - \eta(k)v_j^g(k)\overset{\cdot}{\phi}{}_j^g z_i(k)\psi(u(k))\, e_i(k)\right]^2 - \widetilde{w}_{ji}^g(k)^2 \\
&= \eta^2(k)\left\{\phi_j^{f2} + \left[\overset{\cdot}{\phi}{}_j^f v_j^f(k) z_i(k)\right] + \left[\phi_j^g\psi(u(k))\right]^2 + \left[\overset{\cdot}{\phi}{}_j^g v_j^g(k) z_i(k)\psi(u(k))\right]^2\right\} e_i^2(k) - \\
&2\eta(k)e_i(k)\left[\phi_j^f\widetilde{v}_j^f(k) + \overset{\cdot}{\phi}{}_j^f v_j^f(k)z_i(k)\widetilde{w}_{ji}^f(k) + \phi_j^g\widetilde{v}_j^g(k)\psi(u(k)) + \overset{\cdot}{\phi}{}_j^g v_j^g(k)z_i(k)\widetilde{w}_{ji}^g(k)\psi(u(k))\right]
\end{aligned} \tag{4.26}$$

Por (4.20), se sabe que el ultimo término de (4.26) es:

$$\begin{aligned}
&-2\eta(k)e_i(k)\left[\phi_j^f\widetilde{v}_j^f(k) + \overset{\cdot}{\phi}{}_j^f v_j^f(k)z_i(k)\widetilde{w}_{ji}^f(k) + \phi_j^g\widetilde{v}_j^g(k)\psi(u(k)) + \overset{\cdot}{\phi}{}_j^g v_j^g(k)z_i(k)\widetilde{w}_{ji}^g(k)\psi(u(k))\right] \\
&= -2\eta(k)e_i(k)\left[\beta e_i(k+1) - \zeta(k)\right] \\
&= -2\eta(k)\beta e_i(k)\, e_i(k+1) + 2\eta(k)\zeta(k)\, e_i(k)
\end{aligned}$$

si $e_i^2(k) \geq \frac{\overline{\zeta}^2}{1-\eta}$ y $\beta e_i(k) e_i(k+1) \geq e_i^2(k)$, usando (4.20) y $\eta(k) \geq 0$

$$\begin{aligned}
&-2\eta(k)\beta e_i(k) e_i(k+1) + 2\eta(k)\zeta(k) e_i(k) \\
&\leq -2\eta(k) e_i^2(k) + 2\eta(k)\zeta(k) e_i(k) \\
&\leq -2\eta(k) e_i^2(k) + \eta(k) e_i^2(k) + \eta(k)\zeta^2(k)
\end{aligned}$$

Por las definiciones $a(k) = \phi_j^{f2} + \left|\overset{\cdot}{\phi}_j^f v_j^f(k) z_i(k)\right|^2 + \left|\psi(u(k))\phi_j^g\right|^2 + \left|\psi(u(k))\overset{\cdot}{\phi}_j^g v_j^g(k) z_i(k)\right|^2$ y $\eta(k) = \dfrac{\eta}{1+a(k)}$,

$$\begin{aligned}
&\Delta V_i(k) \leq \eta^2(k) a(k) e_i^2(k) - \eta(k) e_i^2(k) + \eta(k)\zeta^2(k) \\
&\leq \eta(k)\frac{\eta}{1+a(k)} a(k) e_i^2(k) - \eta(k) e_i^2(k) + \eta(k)\zeta^2(k) \\
&\leq \eta(k)\eta e_i^2(k) - \eta(k) e_i^2(k) + \eta(k)\zeta^2(k)
\end{aligned}$$

Usando $\zeta^2(k) \leq \overline{\zeta}^2$ se tiene:

$$\Delta V_i(k) \leq -\eta(k)\left[(1-\eta) e_i^2(k) - \overline{\zeta}^2\right] \tag{4.27}$$

Ya que la zona muerta, $e_i^2(k) \geq \frac{\overline{\zeta}^2}{1-\eta}$ y $\eta(k) > 0$, $\Delta V_i(k) \leq 0$. $V_i(k)$ es acotado. Si $e_i^2(k) < \frac{\overline{\zeta}^2}{1-\eta}$ o $\beta e_i(k) e_i(k+1) < e_i^2(k)$, desde (4.21) se conoce que $\eta(k) = 0$, ninguno de los pesos cambia, por tanto el error de pesos está acotado, así $V_i(k)$ está acotado.

Ahora, se considera $\left|\overline{v}_j^g(k+1) - v_j^g(1)\right| > M$, desde (4.23) se tiene:

$$\begin{aligned}
&v_j^g(k+1) = v_j^g(1) + \frac{\overline{v}_j^g(k+1) - v_j^g(1)}{\left|\overline{v}_j^g(k+1) - v_j^g(1)\right|} M \\
&\left|v_j^g(k+1) - v_j^g(1)\right| = M
\end{aligned}$$

Así $v_j^g(k+1)$ esta en la bola con centro $v_j^g(1)$ y radio M. Ya que $v_j^g(k+1) - v_j^g(1) = M$ entonces $v_j^g(k+1)$ está acotado. Desde la prueba anterior para el caso $\left|\overline{v}_j^g(k+1) - v_j^g(1)\right| \leq M$, se conoce que cuando $v_j^g(k+1)$ esta en la bola, $V_i(k)$ está acotado. Así para cualquier caso, $V_i(k)$ está acotado.

Con esto se forza que $v_j^g(k+1)$ esté en la bola con centro $v_j^g(1)$ y radio M. Si $v_j^g(k+1)$ esta dentro de la bola ($\overline{v}_j^g(k+1) - v_j^g(1) \leq M$), se usa el algoritmo de gradiente normal.

Cuando $\overline{v}_j^g(k+1)$ esta fuera de la bola, $\left|v_j^g(k+1) - v_j^g(1)\right| = M$, así $v_j^g(k+1)$ esta dentro de la bola, es decir, $v_j^g(k+1)$ nunca abandonará la bola. Si se elige $\left\|v_j^g(1)\right\| > M$, $v_j^g(k+1) \neq 0$. Ya que $\phi_j^g(\cdot)$ se selecciona como $\phi_j^g(\cdot) \neq 0$, $\widehat{g}$ en (4.12) es diferente de cero.

2) Si $\left|\overline{v}_j^g(k+1) - v_j^g(1)\right| \leq M$, desde (4.23) se conoce que $\left|v_j^g(k+1) - v_j^g(1)\right| \leq M$, $v_j^g(k+1)$ ya esta dentro de la bola con centro $v_j^g(1)$ y radio M. Cuando $e_i^2(k) \geq \frac{\overline{\zeta}^2}{1-\eta}$, sumando (4.27) desde 1 a T se tiene:

$$\sum_{k=1}^{T} \eta(k) \left[(1-\eta)\, e_i^2(k) - \overline{\zeta}^2\right] \leq V_i(1) - V_i(T+1) \tag{4.28}$$

Usando que $V(T+1)$ está acotado:

$$\limsup_{T\to\infty} \sum_{k=1}^{T} \frac{\eta\left[(1-\eta)\, e_i^2(k) - \overline{\zeta}^2\right]}{1+a(k)} < \infty \tag{4.29}$$

Ya que $e_i^2(k) \geq \frac{\overline{\zeta}^2}{1-\eta}$, $\dfrac{\eta\left[(1-\eta)\, e_i^2(k) - \overline{\zeta}^2\right]}{1+a(k)} \geq 0$, así:

$$\limsup_{k\to\infty} \left(\frac{\eta}{1+a(k)}\right) \left[(1-\eta)\, e_i^2(k) - \overline{\zeta}^2\right] = 0 \tag{4.30}$$

Ya que $V_i(k)$ está acotado, así $a(k) < \infty$ implica que:

$$\limsup_{k\to\infty} (1-\eta)\, e_i^2(k) \leq \overline{\zeta}^2 \tag{4.31}$$

Esto es (4.24). Cuando $e_i^2(k) < \frac{\overline{\zeta}^2}{[1-\eta]}$, ya esta en esta zona. Así cuando $w_j^g(k+1)$ esta dentro o fuera de la bola con centro $v_j^g(1)$ y radio M, (4.31) es correcto.

Si $\left|\overline{v}_j^g(k+1) - v_j^g(1)\right| > M$, desde (4.23) se conoce $\left|v_j^g(k+1) - v_j^g(1)\right| = M$, $v_j^g(k+1)$ esta sobre la bola con centro $v_j^g(1)$ y radio M. Esta es una condición para (4.31). ■

Comentario 4.1 *La doble zona muerta usada en (4.21) es para resolver el problema de robustez causado por las dinámica no modelada $\zeta(k)$. La modificación usada en (4.21) es para garantizar que los parámetros permanecen dentro de una región restringida y no alteran*

las propiedades de la ley adaptable normal. Si $e_i(k)$ *y* $e_i(k+1)$ *tienen el mismo signo,* β *se selecciona suficientemente grande, una de las zonas muertas se hace pequeña, la condición* $\beta e_i(k)\, e_i(k+1) \geq e_i^2(k)$ *no altera mucho la ley de aprendizaje.*

Comentario 4.2 *η es el parámetro llamado coeficiente de aprendizaje y determina la velocidad de aprendizaje de la red neuronal, cuando η es grande se tiene una cota grande para la dinámica no modelada $\overline{\zeta}^2$ dada en (4.24) y la red neuronal aprende rápidamente el comportamiento de las funciones no lineales f y g, cuando η es pequeño se tiene una cota pequeña para la dinámica no modelada $\overline{\zeta}^2$ dada en (4.24) y la red neuronal aprende lentamente el comportamiento de las funciones no lineales f y g. Se tienen las siguientes condiciones extremas: cuando $\eta = 1$, el tiempo de convergencia del error de identificación T_c es mínimo, pero la cota superior del error de identificación es infinitamente grande,esto es, $\overline{e_i} = \frac{\overline{\zeta}^2}{1-\eta} \to \infty$; cuando $\eta = 0$, la cota superior del error de identificación $\overline{e_i} = \frac{\overline{\zeta}^2}{1-\eta}$ es mínima, pero el tiempo de convergencia del error de identificación T_c es infinitamente grande,esto es, $T_c \to \infty$, ver Figura 3.3 con la consideración de que $\eta = \lambda$.*

4.4. Control por modos deslizantes en tiempo discreto

4.4.1. Estructura del controlador

En esta sección se diseña el control por modos deslizantes con modelado por redes neuronales para forzar al sistema no lineal (4.5) a que siga la trayectoria generada por el modelo de referencia estable que tiene la siguiente forma:

$$y_m(k+1) = a_0 y_m(k) + \ldots + a_n y_m(k-n) + b_0 v(k) + \ldots + b_m v(k-m) \tag{4.32}$$

donde $v(k)$ es la entrada de referencia. Se define el error de seguimiento de trayectoria como sigue:

$$e_c(k+1) = y_m(k+1) - y(k+1) \tag{4.33}$$

Ahora se define el vector

$$e(k) = [e_c(k) \cdots e_c(k-n+1)]^T$$

En este documento se propone un controlador por modos deslizantes como sigue:

$$\psi\left(u(k)\right) = \begin{cases} u(k) = \frac{1}{\widehat{g}}\{-\widehat{f} + y_m(k+1) + K^T e(k) - \varepsilon(k) sign\left[s(k)\right]\} & \text{si } |u(k)| \leq b \\ u(k) = b & \text{si } |u(k)| > b \end{cases} \tag{4.34}$$

donde $\widehat{f}$ y $\widehat{g}$ se tienen del modelador por redes neuronales, considerando el algoritmo de modificación dado en (4.23) se tiene que $\widehat{g} > 0$, también se tiene:

$$K = \left[k_1 \cdots k_n\right]^T \in R^{nx1}$$

el cual se selecciona tal que $s^n + \sqrt{2}k_1 s^{n-1} + \cdots + 2^{\frac{n}{2}} k_n$ es estable, $\varepsilon(k+1)$ es una ganancia variante en tiempo la cual satisface:

$$\varepsilon(k) = -e_i(k) sign\left[s(k)\right] + \alpha e^{-ck} \tag{4.35}$$

donde $e_i(k)$ es el error de modelado el cual esta dado en (4.5), (4.14) y (4.19), $c > 0$ y $\alpha > 0$ y se define la función signo como sigue:

$$sign\left[s(k)\right] = \begin{cases} 1 & \text{si } s(k) > 0 \\ 0 & \text{si } s(k) = 0 \\ -1 & \text{si } s(k) < 0 \end{cases} \tag{4.36}$$

$s(k)$ es la función de conmutación la cual se define como sigue:

$$s(k) = e_c(k) + K^T e(k-1) = e_c(k) + k_1 e_c(k-1) + \ldots + k_n e_c(k-n) \tag{4.37}$$

Comentario 4.3 *El control por modos deslizantes propuesto en esta sección es similar al control propuesto en [55], la diferencia es que en [55] la ganancia es constante, aquí la ganancia $\varepsilon(k)$ varia con el tiempo con el error debido a la variación del error modelado $e_i(k)$ y a un termino exponencial decreciente αe^{-ck}. El castañeo de $u(k)$ depende de dos hechos: del error de modelado y de las constantes c y α. Una c más pequeña causa menos castañeo y mas grande error de seguimiento de trayectoria. Un mejor c se puede elegir por simulaciones. La amplitud del castañeo en $u(k)$ esta determinado por la ganancia $\varepsilon(k)$, esta ganancia decrece*

al decrecer el error de modelado $e_i(k)$. Desde el Teorema de la sección pasada se conoce que $e_i(k)$ converge a una bola con radio $\frac{\overline{\zeta}^2}{1-\eta}$, la amplitud del castañeo se reduce conforme el error de modelado $e_i(k)$ converge. El controlador por modos deslizantes normal requiere de una ganancia constante mayor que la cota de la incertidumbre para garantizar la robustez [55], [87]. La robustez del controlador (4.34) se puede explicar como sigue: cuando la incertidumbre estructural es mas grande ($\overline{\zeta}$ es más grande), la zona muerta (4.22) y el radio $\frac{\overline{\zeta}^2}{1-\eta}$ son mas grandes. Así $e_i(k)$ es más grande y el castañeo se incrementa.

La estructura del control por modos deslizantes con modelador por redes neuronales esta dada en la Figura 4.2. Desde la desigualdad (4.24) obtenida del Teorema de la sección pasada se conoce que el error de modelado $e_i(k)$ está acotado, es decir:

$$\lim_{k\to\infty} \sup e_i^2(k) \leq \frac{\overline{\zeta}^2}{1-\eta} \rightarrow |e_i(k+1)| \leq H \quad |e_i(k)| \leq H \tag{4.38}$$

donde H es una constante positiva. El siguiente lema da la cota de la función de conmutación.

Lemma 4.1 *La función de conmutación (4.37) en el controlador por modos deslizantes en tiempo discreto (4.34) satisface la condición (4.4), es decir:*

$$|s(k+1)| \leq \alpha + (1+\beta) H \tag{4.39}$$

donde β y H están dadas en (4.14) y (4.38) .

Demostración. Desde (4.14) y (4.15) el error de modelado satisface lo siguiente:

$$\beta e_i(k+1) = \widetilde{f} + \widetilde{g}\psi(u(k)) \tag{4.40}$$

donde $\widetilde{f} = \widehat{f} - f$ y $\widetilde{g} = \widehat{g} - g$. Sustituyendo el control (4.34) en la planta (4.5), el sistema en lazo cerrado es:

$$\begin{aligned} y(k+1) &= \widehat{f} - \widetilde{f} + \tfrac{\widehat{g}-\widetilde{g}}{\widehat{g}}[-\widehat{f} + y_m(k+1) + K^T e(k) - \varepsilon(k) sign\,[s(k)]] \\ &= -\widetilde{f}(k) + y_m(k+1) + K^T e(k) - \varepsilon(k) sign\,[s(k)] - \widetilde{g}(k)\psi(u(k)) \end{aligned}$$

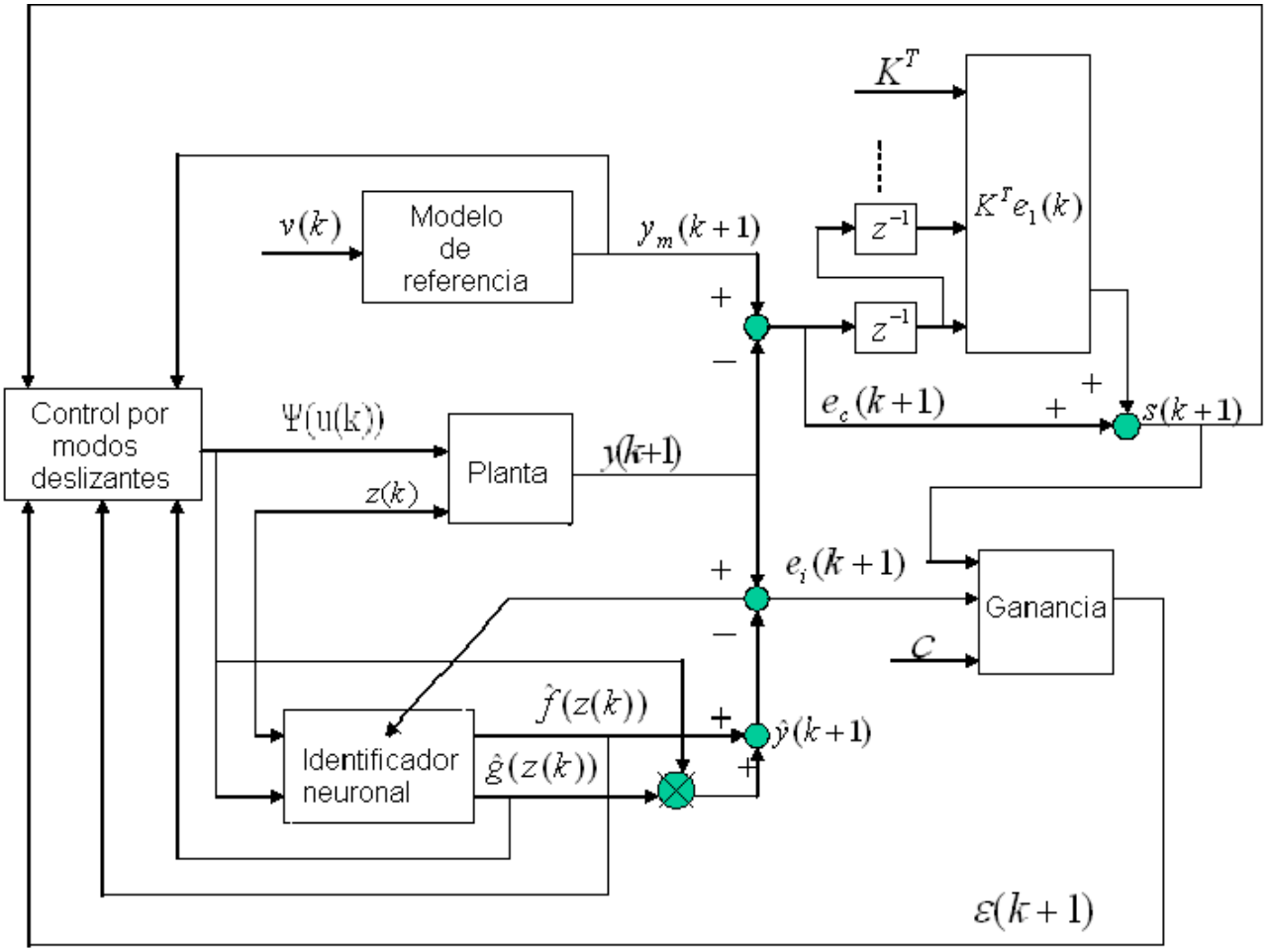

Figura 4.2: Control neuronal utilizando modos deslizantes

Desde (4.33):

$$e_c(k+1) + K^T e(k) = \varepsilon(k) sign\left[s(k)\right] + \widetilde{f}(k) + \widetilde{g}(k)\psi\left(u(k)\right) \tag{4.41}$$

La función de conmutación (4.37) es:

$$s(k+1) = e_c(k+1) + K^T e(k) \tag{4.42}$$

Usando (4.42) y (4.40), (4.41) se convierte en:

$$\begin{aligned} s(k+1) &= \varepsilon(k) sign\left[s(k)\right] + \beta e_i(k+1) \\ &= -e_i(k) sign\left[s(k)\right]^2 + sign\left[s(k)\right]\alpha e^{-ck} + \beta e_i(k+1) \end{aligned} \tag{4.43}$$

Se calcula $s(k+1)$ con (4.43). Si $s(k) \neq 0$, $sign^2\left[s(k)\right] = 1$, (4.43) se convierte en:

$$s(k+1) = \beta e_i(k+1) - e_i(k) + sign\left[s(k)\right]\alpha e^{-ck} \tag{4.44}$$

Desde (4.38) se conoce que:

$$|s(k+1)| \leq (1+\beta)\, H + |sign\left[s(k)\right]|\, \alpha e^{-ck} \leq (1+\beta)\, H + \alpha \tag{4.45}$$

si $s(k) = 0$, $sign^2\left[s(k)\right] = 0$ se tiene:

$$s(k+1) = \beta e_i(k+1) \tag{4.46}$$

entonces, usando (4.38), (4.46) se convierte en:

$$|s(k+1)| \leq \beta H < (1+\beta)\, H + \alpha \tag{4.47}$$

desde (4.45) y (4.47), (4.39) es cierto. ■

Comentario 4.4 *Si se desconoce el comportamiento de las funciones no lineales $f()$ y $g()$, se propone que se aproximen estas con redes neuronales, para esto solamente se cuenta con el error de identificación $e_i(k)$, como el aproximador de funciones con redes neuronales es lento y trabaja en línea con el controlador, si se considera el controlador con redes neuronales sin modos deslizantes se tiene la siguiente dinámica del error $s(k+1) = \beta e_i(k+1)$ que*

difiere a (4.44), en este caso se tiene que siempre va a permanecer el error de identificación $\beta e_i(k+1)$ *debido a que las redes neuronales no proporcionan una buena identificación. Los modos deslizantes no pueden compensar a la función no lineal* $g()$ *dado que esta multiplica al control como indica en (4.5), se tiene que utilizar el aproximador por lo menos para estimar* $g()$*, si se considera esto el controlador con redes neuronales para aproximar* $g()$ *y modos deslizantes con ganancia fija para compensar* $f()$ *se tiene la siguiente dinámica del error* $s(k+1) = \beta e_i(k+1) - Ke_i(k)$ *con lo cual dado que* $\beta e_i(k+1)$ *es grande debido a lo ya explicado, se requiere de una ganancia de modos deslizantes* K *grande para compensar este error, lo cual produce castañeo. En el controlador con aproximador de funciones neuronal de tiene la siguiente dinámica del error* $s(k+1) = \beta e_i(k+1) - e_i(k) + sign\left[s(k)\right]\alpha e^{-ck}$ *donde el término* $-e_i(k) + sign\left[s(k)\right]\alpha e^{-ck}$ *compensa a* $\beta e_i(k+1)$*. Si se conociera* $e_i(k+1)$ *se tendría un controlador exponencialmente estable.*

4.4.2. Análisis de estabilidad para el control en lazo cerrado

Ahora se discute la estabilidad del sistema en lazo cerrado. (4.5) se puede escribir en espacio de estados como (4.8) [38]. El subsistema de control de (4.8) es:

$$\begin{cases} z_{n+1}(k+1) = z_{n+2}(k) \\ \vdots \\ z_{n+m}(k+1) = \psi\left(u(k)\right) \end{cases}$$

este se puede expresar como sigue:

$$\omega(k+1) = \begin{bmatrix} 0 & 1 & 0 & \cdots \\ \vdots & & \ddots & \\ 0 & \cdots & \cdots & 1 \\ 0 & \cdots & \cdots & 0 \end{bmatrix} \omega(k) + b_u \psi\left(u(k)\right) \tag{4.48}$$

donde $\omega(k) = [z_{n+1}(k) \cdots z_{n+m}(k)]^T$, $b_u = [0 \cdots 1]^T$. Ya que $\begin{bmatrix} 0 & 1 & 0 & \cdots \\ \vdots & & \ddots & \\ 0 & \cdots & \cdots & 1 \\ 0 & \cdots & \cdots & 0 \end{bmatrix}$ es estable, el sistema (4.48) es estable. La función de conmutación (4.42) se escribe como sigue:

$$e_c(k+1) = -K^T e(k) + s(k+1) \tag{4.49}$$

Por la definición $e(k) = [e_c(k-n+1) \cdots e_c(k)]^T$, (4.49) se escribe como sigue:

$$e(k+1) = Ae(k) + bs(k+1) \tag{4.50}$$

donde $A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & \cdots & \cdots & 0 & 1 \\ -k_n & \cdots & \cdots & \cdots & -k_1 \end{bmatrix} \in R^{nxn}$, $b = [0, \cdots 0, 1]^T \in R^{nx1}$. Ya que $\det(sI - \alpha A) = \alpha^n k_n + \alpha^{n-1} k_{n-1} s + \cdots + \alpha k_1 s^{n-1} + s^n$ [33], se selecciona $K = [k_1 \cdots k_n]^T$ tal que $\sqrt{2}A$ es estable ($\alpha = \sqrt{2}$). Un valor estable de $\sqrt{2}A$ puede hacer que la siguiente función tenga soluciones positivas definidas para P y Q:

$$2A^T PA - P = -Q \tag{4.51}$$

donde $P = P^T > 0$, $Q = Q^T > 0$.

Teorema 4.2 *El sistema en lazo cerrado con control por modos deslizantes (4.34) y modelador por redes neuronales (4.12) y (4.21), es uniformemente estable y la cota superior del error de seguimiento de trayectoria es:*

$$\lim_{k\to\infty} \sup \frac{1}{T} \sum_{k=1}^{T} e^T(k) e(k) \leq \frac{2p_{nn}}{\lambda_{\text{mín}}(Q)} r^2 \tag{4.52}$$

donde $r = (1+\beta) H + \alpha$, β *y* H *están dadas en (4.14) y (4.38),* P *y* Q *están dados en (4.51),* $b = [0, \cdots 0, 1]^T$, $b^T P b = p_{nn}$.

Demostración. *Se define la siguiente función de energía:*

$$V(k) = e^T(k)Pe(k) \tag{4.53}$$

donde P es una solución de (4.51). Usando (4.50) se calcula $\Delta V(k)$:

$$\begin{aligned} \Delta V(k) &= e^T(k+1)Pe(k+1) - e^T(k)Pe(k) \\ &= e^T(k)\left(A^TPA - P\right)e(k) + 2e^T(k)A^TPbs(k+1) + b^TPbs^2(k+1) \end{aligned} \tag{4.54}$$

Usando la siguiente desigualdad:

$$2e^T(k)A^TPbs(k+1) \leq e^T(k)A^TPAe(k) + b^TPbs^2(k+1) \tag{4.55}$$

Sea:

$$2b^TPbs^2(k+1) \leq cs^2(k+1) \tag{4.56}$$

donde $c = 2b^TPb \geq 0$. Por (4.51) y desde (4.39) se tiene $|s(k+1)| \leq r = (1+\beta)H + \alpha$, (4.54) es:

$$\Delta V(k) \leq -e^T(k)\left(2A^TPA - P\right)e(k) + cr^2$$

usando (4.51) se tiene lo siguiente:

$$\Delta V(k) \leq -e^T(k)Qe(k) + cr^2 \tag{4.57}$$

multiplicando por $P^{-1}P$ se tiene lo siguiente:

$$\begin{aligned} \Delta V(k) &\leq -e^T(k)QP^{-1}Pe(k) + cr^2 \\ \Delta V(k) &\leq -\lambda_{\text{mín}}(Q)\,\lambda_{\text{mín}}(P^{-1})\,e^T(k)Pe(k) + cr^2 \\ \Delta V(k) &\leq -\gamma V(k) + cr^2 \end{aligned}$$

donde $\gamma = \lambda_{\text{mín}}(Q)\,\lambda_{\text{mín}}(P^{-1}) > 0$, desde [37] se tiene que $V(k)$ está acotado, así $e(k)$ está acotado. Sumando (4.57) desde 1 a T y usando sabiendo que $V(T) > 0$ y que $V(1)$ es constante:

$$\begin{aligned} V(T) - V(1) &\leq \sum_{k=1}^{T} -\lambda_{\text{mín}}(Q)\,e^T(k)e(k) + 2b^TPbr^2 \\ \lim_{T\to\infty}\sup\tfrac{1}{T}\sum_{k=1}^{T}\lambda_{\text{mín}}(Q)\,e^T(k)e(k) &\leq V(1)\lim_{T\to\infty}\sup\tfrac{1}{T} + 2b^TPbr^2 \end{aligned}$$

entonces (4.52) si se satisface. Por la definición $e(k) = [e_c(k) \cdots e_c(k-n+1)]^T$ *se tiene que el error de seguimiento de trayectoria* $e_c(k)$ *también está acotado.* ■

Comentario 4.5 *P y Q debe satisfacer la ecuación (4.51), entonces ninguno de estos parámetros se puede usar para hacer mas pequeño el error de seguimiento de trayectoria en (4.52).*

4.5. Simulaciones

En esta sección, se dan dos ejemplos para ilustrar como se aplican los resultados teórico propuestos en esta sección. El primer ejemplo es el modelo de trafico tomado de [31] y [32] el cual se explica a detalle en el capítulo anterior. El segundo ejemplo es más complejo y con mayor dimensión, este también es usado por varios autores como son [48] y [8].

4.5.1. Control propuesto aplicado al modelo de tráfico

El modelo se explica a detalle en el capítulo anterior. Se va a consideras solamente el control para $i = 1$, tomando esto se cuenta a partir de (3.44) se tiene el siguiente modelo:

$$\begin{gathered} A_1^{out}(k) = \text{mín}(\rho_{c1}, \rho_1(k))\frac{\Delta L}{\Delta k} \\ A_2^{in}(k) = \begin{cases} \text{mín}(\rho_{c2}, \rho_{J2} - \rho_2(k))\frac{\Delta L}{\Delta k} & \text{si } \rho_2(k) \leq \rho_{c2} \\ \frac{\rho_{c2}(\rho_{J2}-\rho_2(k))}{\rho_{J2}-\rho_{c2}}\frac{\Delta L}{\Delta k} & \text{si } \rho_2(k) > \rho_{c2} \end{cases} \\ f_{1,2}(k) = \text{mín}\left(A_2^{in}(k), A_1^{out}(k)\right) \\ \rho_1(k+1) = \rho_1(k) - f_{1,2}(k)\frac{\Delta k}{\Delta L} + f_{0,1}(k)\frac{\Delta k}{\Delta L} \end{gathered} \tag{4.58}$$

donde $f_{0,1}(k) = \psi\left(u(k)\right)$ es la variable de control, $\rho_2(k) = 8$, $y(k) = \rho_1(1) = 8$, $\rho_{c1} = 6$, $\rho_{J1} = 10$, $\Delta k = 20$ segundos y $\Delta L = 40$ metros, esto es, cada bloque tiene 40 metros de largo, se toman datos cada 20 segundos, donde, cuando se tiene menos de 6 vehículos en 40 metros se tiene movimiento libre, pero cuando se tiene más de 6 vehículos en 40 metros se tiene un embotellamiento. Notar que se tiene en el bloque 2 la condición de embotellamiento y la condición inicial del bloque 1 es también de embotellamiento, el sistema (4.58) tiene la misma

estructura que (4.5) con $g(1) = \frac{\Delta k}{\Delta L} > 0$ que satisface (4.7). Se va a diseñar un controlador para forzar que (4.58) siga a un modelo de referencia. Se selecciona el modelo de referencia $\rho_r(k+1) = y_r(k+1)$ como:

$$\begin{aligned} \rho_r(k+1) &= 4 \quad \text{si } 0 \leq k \leq 50 \\ \rho_r(k+1) &= 2 \quad \text{si } 50 < k \leq 100 \\ \rho_r(k+1) &= 0 \quad \text{si } 100 < k \leq 150 \\ \rho_r(k+1) &= 2 \quad \text{si } 150 < k \leq 200 \end{aligned} \tag{4.59}$$

notar que se tiene una señal de referencia tal que obligue a que se tenga la condición de vehículos viajando con movimiento libre. Se usa usa una red neuronal para modelar $f\left[z(k)\right] = \rho_1(k) - f_{1,2}(k)\frac{\Delta k}{\Delta L}$ y $g\left(1\right) = \frac{\Delta k}{\Delta L}$, el numero de estados es $n+m=1$, así el número de neuronas en la capa oculta es $q = 1$, $\beta = 1$. Los pesos iniciales son $v_j^f(1)$, $w_{ji}^f(1)$ y $w_{ji}^g(1)$ se eligen como números aleatorios entre $(0,1)$, $v_j^g(1) = 0{,}95$ y $M = 0{,}05$. La ley de actualización es (4.21) con $\eta = 0{,}01$.

Usando el modelador neuronal propuesto, el control por modos deslizantes empleado es (4.34), se seleccionan los parámetros del controlador como $k_1 = -0{,}01$ y $c = 100$, $\alpha = 0{,}1$. Ahora se compara el comportamiento del controlador por modos deslizantes con ganancia variante en el tiempo contra el control por retroalimentación con modelador por redes neuronales [8] y contra el controlador por modos deslizantes con ganancia fija [55], la ganancia fija se elige como $\varepsilon = 0{,}05$. Si $\varepsilon > 0{,}05$, el control por modos deslizantes con ganancia fija [55] general mucho castañeo. En [8] y [55] la modelado por redes neuronales es fuera de línea, es decir, el control se aplica después que se ha finalizado la etapa de modelado. En este documento se consideran modelador y controlador en lineal (on-line). Los resultados del control en modos deslizantes con ganancia variante en tiempo se muestran en Figura 4.3.

El error de seguimiento de trayectoria de los 3 controladores se muestra en la Figura 4.4. Se puede ver que aunque el error de modelado no es muy pequeño, el error de seguimiento de trayectoria si es muy pequeño por la compensación de los modos deslizantes. También la función de conmutación es decreciente, pero esta no converge a cero. El control por modos deslizantes en tiempo discreto con redes neuronales propuesto en este documento tiene un

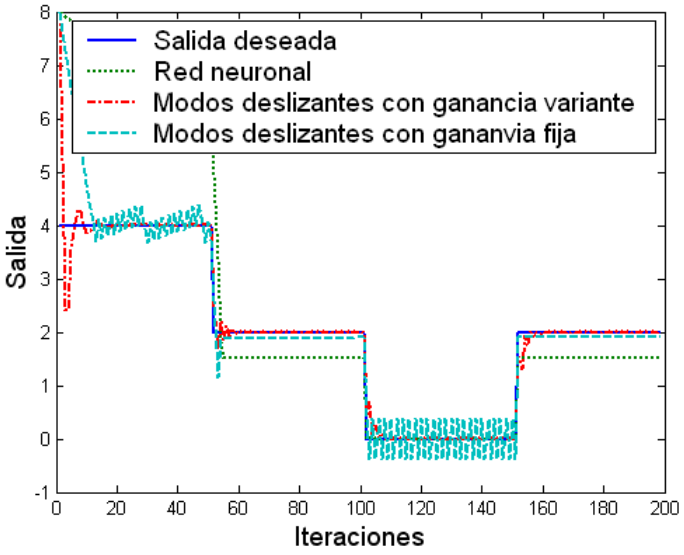

Figura 4.3: Desempeño del controlador por modos deslizantes con ganancia variante

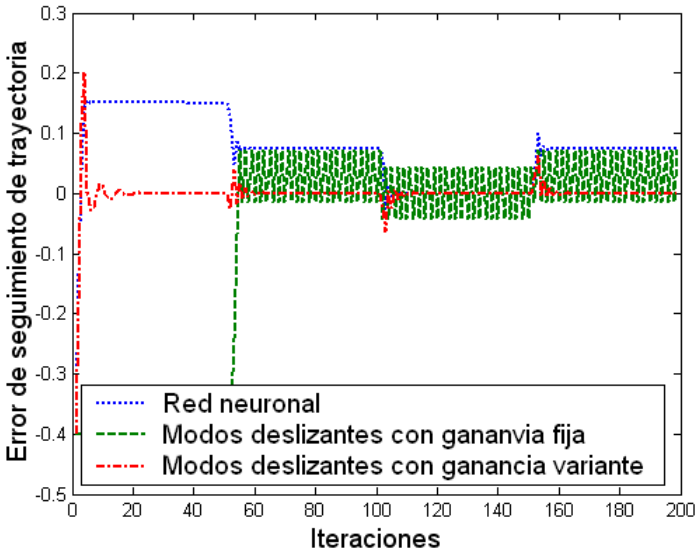

Figura 4.4: Errores de seguimiento de trayectoria

mejor seguimiento de trayectoria aún cuando el modelado no es muy bueno. Para este sistema no lineal simple el control neuronal por modos deslizantes con ganancia variante en el tiempo tiene mejor desempeño que el control neuronal por modos deslizantes con ganancia fija y que el control neuronal.

4.5.2. Control propuesto aplicado a un sistema no lineal académico complejo

Considerando el sistema dado en [8], [48] y [56]:

$$\begin{aligned} &y(k+1) = 0{,}2y^2(k) + 0{,}2y(k-1) \\ &+0{,}4\sin\left[0{,}5\left(y(k)+y(k+1)\right)\right]\cos\left[0{,}5\left(y(k)+y(k+1)\right)\right] + 1{,}2\psi\left(u(k)\right) \end{aligned} \tag{4.60}$$

donde $\psi\left(u(k)\right)$ es el flujo de entrada (entrada de control), la cual puede ser positivo o negativo. Se desea forzar la salida del sistema (4.60) que siga el siguiente modelo de referencia lineal:

$$\begin{aligned} y_r(k+1) &= 0{,}6 \quad \text{si } 0 \leq k \leq 50 \\ y_r(k+1) &= 0{,}4 \quad \text{si } 50 < k \leq 100 \\ y_r(k+1) &= 0{,}2 \quad \text{si } 100 < k \leq 150 \\ y_r(k+1) &= 0{,}4 \quad \text{si } 150 < k \leq 200 \end{aligned} \tag{4.61}$$

Se usa una red neuronal para modelar $f\left(z(k)\right) = 0{,}2y^2(k)+0{,}2y(k-1)+0{,}4\sin\left[0{,}5\left(y(k)+y(k+1)\right)\right]\cos[$ y $g\left(1\right) = 1{,}2$, el número de estados es $n+m=2$, el número de neuronas en la capa oculta se selecciona como $q=2$, los pesos iniciales son $v_j^f(1)$, $w_{ji}^f(1)$, y $w_{ji}^g(1)$ se seleccionan como números aleatorios entre $(0,1)$, $v_j^g(1) = 0{,}95$ y $M = 0{,}05$. La ley de actualización es (4.21) con $\eta = 0{,}01$. Se seleccionan los parámetros del controlador como $k_1 = -0{,}01$, $k_2 = -0{,}001$ y $c = 100$ y $\alpha = 0{,}1$. Para (4.50), $A = \begin{bmatrix} 0 & 1 \\ 0{,}001 & 0{,}01 \end{bmatrix}$, los valores propios de A son $\lambda_1\left(2A\right) = -0{,}054$ y $\lambda_2\left(2A\right) = 0{,}074$, así A es estable. Una posible combinación de P y Q es $P = \begin{bmatrix} 1 & 0 \\ 0 & 5{,}0021 \end{bmatrix} > 0$ y $Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} > 0$. Se compara el algoritmo propuesto en este documento contra el control por retroalimentación con modelador por redes neuronales [8] y

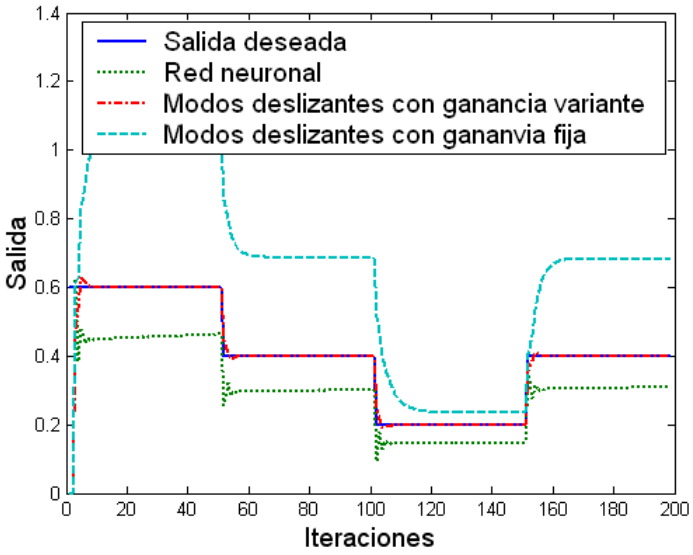

Figura 4.5: Desempeño del controlador por modos deslizantes con ganancia variante

contra el controlador por modos deslizantes con ganancia fija [55] con ganancia $\varepsilon = 0{,}1$. El desempeño del control por modos deslizantes de ganancia fija variante en tiempo contra el de ganancia fija se muestran en la Figura 4.5.

La ganancia variante en tiempo $\varepsilon(k)$ en (4.34) reduce el castañeo comparado al de ganancia fija. El error de seguimiento de trayectoria de los tres controladores se muestra en la Figura 4.6.

Si c es pequeña, el castañeo es menor pero el error de seguimiento es mas grande. Para este sistema no lineal complejo el control neuronal por modos deslizantes con ganancia variante en el tiempo tiene mejor desempeño que el control neuronal por modos deslizantes con ganancia fija y que el control neuronal.

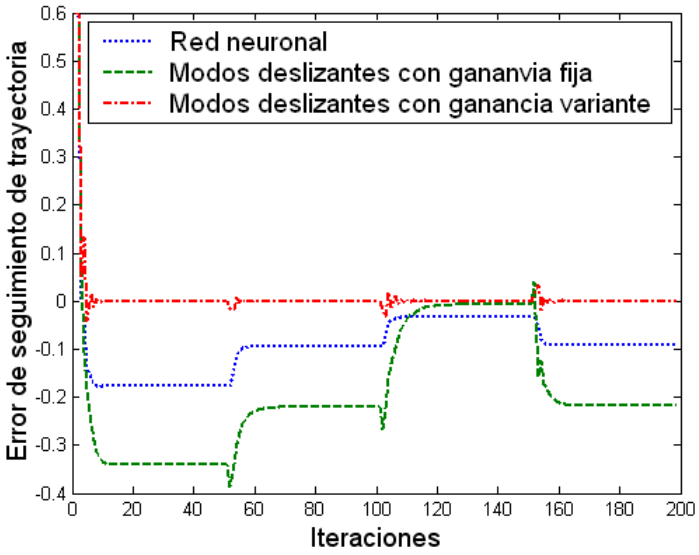

Figura 4.6: Errores de seguimiento de trayectoria

4.6. Conclusión

En esta sección se propone un nuevo controlador por modos deslizantes para una clase de sistemas no lineales en tiempo discreto. Las principales contribuciones son: 1) Se propone un nuevo algoritmo de redes neuronales para la aproximación de funciones no lineales, el cual usa las técnicas de zona muerta y proyección. La zona muerta asegura que el error de modelado es acotado, la proyección evita cualquier singularidad en la ley de control. 2) La ganancia variante en tiempo en el controlador por modos deslizantes asegura menor castañeo comparado al controlador por modos deslizantes con ganancia fija. Se prueba la estabilidad del sistema en lazo cerrado con el controlador por modos deslizantes y el modelador por redes neuronales. El modelador y el controlador trabajan en línea. Desde el punto de vista de los sistemas dinámicos, este control puede ser útil para aplicaciones de control con redes neuronales que requieren actualización de los pesos en línea.

Capítulo 5

Conclusiones y trabajo futuro

Se propone un método de identificación para redes neuronales recurrentes. Para aplicar el algoritmo de elipsoidal acotado se requiere que el error de incertidumbre estructural sea acotado, además este algoritmo tiene una convergencia en parámetros más rápida que el algoritmo de propagación hacia atrás. Se modifica el algoritmo elipsoidal con un coeficiente de aprendizaje variante en tiempo para entrenar una red neuronal. Se actualizan tanto capa oculta como capa de salida de una red neuronal recurrente en espacio de estados. Se prueba que el error de identificación converge a una zona la cual depende del error de la incertidumbre estructural.

Se propone un nuevo controlador por modos deslizantes para una clase de sistemas no lineales en tiempo discreto. Las principales contribuciones son: 1) Se propone un nuevo algoritmo de redes neuronales para la aproximación de funciones no lineales, el cual usa las técnicas de zona muerta y una modificación. La zona muerta asegura que el error de modelado es acotado, la su modificación evita cualquier singularidad en la ley de control. 2) La ganancia variante en tiempo en el controlador por modos deslizantes asegura menor castañeo comparado al controlador por modos deslizantes con ganancia fija. Se prueba la estabilidad del sistema en lazo cerrado con el controlador por modos deslizantes y el modelador por redes neuronales. El modelador y el controlador trabajan en línea. Desde el punto de vista de los sistemas dinámicos, este control puede ser útil para aplicaciones de control con redes

neuronales que requieren actualización de los pesos en línea.

Los trabajos futuros son los siguientes:

- Se pretende encontrar otra ganancia adaptable para el control por modos deslizantes la cual elimine el castañeo.
- En la actualidad, se utiliza una variante del algoritmo de propagación hacia atrás para el modelado de funciones en el control por modos deslizantes, se puede cambiar el algoritmo de propagación hacia atrás por una variante del algoritmo elipsoidal acotado para el modelado de funciones en el control por modos deslizantes.
- Se pretende trabajar con sistemas no lineales con modelo en espacio de estados en tiempo discreto.

Capítulo 6

Apéndice: publicaciones

1. José de Jesús Rubio and Wen Yu, A new discrete-time sliding-mode control with time-varying gain and neural identification, *International Journal of Control*, Vol. 79, No. 4, 338-348, 2006

2. José de Jesús Rubio and Wen Yu, Dead-zone Kalman filter algorithm for recurrent neural networks, *44rd IEEE Conference on Decision and Control, CDC'05*, Seville, Spain, 2562-2567, 2005.

3. José de Jesús Rubio, Wen Yu, Andrés Ferreyra, A new discrete-time sliding-mode control with neural identification, *25th American Control Conferences, ACC'06*, Minneapolis, Minnesota, USA, 5413-5418, 2006

4. José de Jesús Rubio, Wen Yu, Discrete-Time Sliding-Mode Control Based on Neural Networks, *Advances in Neural Networks -ISNN 2006*, Srpinger-Verlgag, Lecture Notes in Computer Science, LNCS 3972, 956-961, 2006

5. Wen Yu, José de Jesús Rubi, Xiaoou Li, Recurrent Neural Networks Training with Stable Risk-Sensitive Kalman Filter Algorithm, *Internal Joint Conference on Neural Networks*, IJCNN'05, Montreal, Canada, 700-705, 2005

6. José de Jesús Rubio, Wen Yu, A new discrete-time sliding-mode control using neural networks, *Congreso anual de la Asociación de México de Control Automático 2005*, Cuernavaca, Mexico, 213-218, 2005

7. José de Jesús Rubio, Nonlinear system identification with recurrent neural networks and dead-zone Kalman filter algorithm, *Neurocomputing*, Vol. 70, No. 13, 2460-2466, 2007.

8. José de Jesús Rubio, Wen Yu, Recurrent neural networks training with optimal bounded ellipsoid algorithm, *26th American Control Conferences, ACC'07*, New York, USA, 2007. Aceptado para publicación.

Bibliografía

[1] A.Alessandri, MCuneo, S.Pagnan, M.Sanguineti, On the convergence of EKF-based parameters optimization for neural networks, *42nd IEEE Conference on Decision and Control,* 6181 - 6186, 2003

[2] A. Bartoszewicz, Discrete-Time Quasi-Sliding Mode Control Strategies, *IEEE Transactions on Industrial Electronics*, Vol.45, No.4, 633-637, Nobember 1998.

[3] R.G.Bland, D.Goldfarb, and M.J.Todd, The ellipsoid method: A survey, *Operation Research*, vol. 29, pp. 1039–1091, 1981.

[4] R.K. Boel, M.R.James and I.R.Petersen, Robustness and Risk-Sensitive Filtering, *IEEE Trans. Automatic Control*, Vol.47, No.3, 451-462, 2002.

[5] J. B. D. Cabrera and K. S. Narendra, Issues in the Application of Newral Networks for Tracking Based on Inverse Control, *IEEE Transactions on Automatic Control*, Vol.44, No.11, 2007-2027, Nobember 1999.

[6] F. Charbonnier, H. Alla and R. David, The Supervised Control of Discrete-Event Dynamic Systems, *IEEE Transactions on Control Systems Technology*, Vol.7, No.2, 175-187, March 1999.

[7] S. Chen, S. A. Billings and P. M. Grant, A recursive hybrid algorithm for nonlinear system identification using radial basis function networks, *Int. J. Contr.*, Vol.55, No.5, 1051-1070, 1992.

[8] F.-C. Chen and H. K. Khalil, Adaptive Control of a Class of Nonlinear Discrete Time Systems Using Neural Networks, *IEEE Transactions on Automatic Control*, Vol.40, No.5, 791-801, May 1995.

[9] B. Chopard P. O. Luthi, P-A Queloz, Cellular automata model of car traffic in a two-dimensional street network, *IJ. Phys. A: Math. Gen. 29*, 2325-2336, 1996.

[10] F.N.Chowdhury, A new approach to real-time training of dynamic neural networks, *International Journal of Adaptive Control and Signal Processing*, 509-521, 2003.

[11] M. J. Corless and G. Leitman, Continuous State Feedback Guaranteeing Uniform Ultimate Buondnesss for Uncertain Dynamic Systems, *IEEE Transactions on Automatic Control*, Vol.26, 1139-1144, 1981.

[12] M.V.Correa, L.A.Aguirre, and R.R.Saldanha, Using Steady-State Prior Knowledge to Constrain Parameter Estimates in Nonlinear System Identification, *IEEE Transactions on Circuits and Systems, Part I*, Vol.49, No.9, 1376-1381, 2002.

[13] G.Cybenko, Approximation by Superposition of Sigmoidal Activation Function, *Math.Control, Sig Syst*, Vol.2, 303-314, 1989.

[14] S.Dey and J.B. Moore, Finite-Dimensional Risk-Sensitive Filters and Smoothers for Discrete-Time Nonlinear Systems, *IEEE Trans. Automatic Control*, Vol.44, No.6, 1234-1239, 1999.

[15] J.A. Dickerson, B.Kosko, Fuzzy function approximation with ellipsoidal rules, *IEEE Transactions on Systems, Man and Cybernetics*, Vol.26, 542-560, Aug 1996.

[16] M. Dogruel, Input Linearization of Nonlinear Systems via Pulse-With control, *IEEE Transactions on Automatic Control*, Vol.48, No.4, 635-638, April 2003.

[17] El Ghaoui, Robust filtering for discrete-time systems with bounded noise and parametric uncertainty, *IEEE Trans. Automatic Control,* Vol.46 , No.7, 1084 - 1089, 2001

[18] E. Umes-Eronini, Dinámica de sistemas y control, *International Thomsom Editores*, 2001.

[19] Y. Fang, T. W. S. Chow and X. D. Li, Use of a Recurrent Neural Network in Discrete Sliding-Mode Control, *IEE Proceeding-Control Theory Applications*, Vol.146, No.1, 84-90, 1999.

[20] A. D. Febraro, G. Giglio, N. Sacco, Urban Traffic Control Structure Based on Hybrid Petri Nets, *IEEE Transactions on intelligent transportation systems,* Vol. 5,No.4, 224-236, 2004.

[21] E. Fogel and Y. F. Huang, On the value of information in system identification: Bounded noise case, *Automatica*, vol. 18, no.2, pp. 229–238, 1982.

[22] K. Furuta, Sliding mode control of a discrete system, *Systems and Control Letters*, No.14, 145-152, 1990.

[23] W. Gao, Y. Wang, and A. Homaifa, Discrete-time variable structure control systems, *IEEE Trans. Ind. Electron.*, Vol.42, 117–122, 1995.

[24] S. S. Ge, J. Zhang and T. H. Lee, Adaptive Neural Network Control for a Class of MIMO Nonlinear Systems with Disturbances in Discrete-Time, *IEEE Transactions on Systems, Man and Cybernetics*, Vol.34, No.4, 1630-1645, August 2004.

[25] S. Gollamudi, S.Nagaraj, S.Kapoor and Y.F.Huang, Set Membership State Estimation with Optimal Bounding Ellipsoids, *Proc. Int. Symp. on Information Theory and its App.,* pp. 262-265, Victoria, B.C., Canada, September, 1996.

[26] G. C. Goodwin and K. Sang Sin, *Adaptive filtering prediction and control*, Prentice Hall, Englewood Cliffis, NJ07632, 1984.

[27] L.Guo, Estimating time-varying parameters by the Kalman filter based algorithm: stability and convergence, *IEEE Trans. Automatic Control*, 141-147, 1990.

[28] S.Haykin, *Neural Networks- A Comprehensive Foundation*, Macmillan College Publ. Co., New York, 1994.

[29] José Ramón Hilera Gonzáles y Víctor José Martínez Hernando, *Redes Neuronales Artificiales*, Adison-Wesley Iberoamericana, E.U.A., 45-180, 1995.

[30] J. Hopfield. Networks with graded response have collective computational properties like those of two-state networks, *Procedings of the National Acaddemic of Sciences*, 3088-3092, 1982.

[31] R. Horiguchi, M. Katakura, H. Akahane, M. Kuwahara, A Development of a Traffic Simulator for Urban Road Networks: Avenue, *Vehicle Navigation and Information Systems Conference Proceedings*, 245-250, 1994.

[32] R. Horiguchi, M. Katakura, I. Nishikawa, The Model Validation of Traffic System for Urban Road Networks: Avenue, *Proceedings of The Second Worl Congress on Intelligent Transport Systems*, 1977-1982, 1995.

[33] R.A.Horn, C.R.Johnson, *Matrix Analysis*, Cambridge University Press, 1985.

[34] S. Hui and H. Kak, On discrete-time variable structure sliding mode control, *Systems and Control Letters*, No.38, 283-288, 1999.

[35] C.-L. Hwang, and C.-H. Li, A Discrete-Time Multivariable Neuro-Adaptive Control for Nonlinear Unknown Dynamic Systems, *IEEE Transactions on Systems, Man and Cybernetics*, Vol.30, No.6, 865-877, December 2000.

[36] Y. Iiguni, H. Sakai, Member, IEEE and Higekatsu Tokumaru, A Real-Time Learning Algorithm for a Multilayered Neural Network Based on the Extended Kalman Filter, *IEEE Transactions on Signal Processing*, Vol.40, No.4, 959-966, April 1992.

[37] P.A.Ioannou and J.Sun, *Robust Adaptive Control*, Prentice-Hall, Inc, Upper Saddle River: NJ, 1996

[38] S. Jagannathan, Control of a Class of Nonlinear Discrete-Time Systems Using Multilayer Neural Networks, *IEEE Transactions on Neural Networks*, Vol.12, No.5, 1113-1120, September 2001.

[39] M. Jayakumar and R.N. Banavar, Risk-Sensitive Filters for Recursive Estimation of Motion From Images, I*EEE Trans. Pattern Analysis and Machine Intelligence*, Vol.20, No.6, 659-667, 1998.

[40] D.Joachim and J.R.Deller, Multiweight Optimization in Optimal Bounding Ellipsoid Algoritms, *ITransactions on Signal Processing*, August 2004.

[41] S.Kapoor, S.Gollamudi, S. Nagaraj and Y. F.Huang, Tracking of Time-Varing Parameters using Optimal Bounding Ellipsoid Algorithms, *Proc. of 34th Allerton Conference on Comm., Contr. and Computing.*, Monticello, Oct. 1996.

[42] N.S. Kayuri,V.Vienkatasubramanian, Representing bounded fault classes using neural networks with ellipsoidal activation functions, *Computers & Chemical Engineering*, Vol. 17, no. 2, pp. 139-163. 1993

[43] T. Kohonen, *Associative Memory. Asystem theoretical approach*, Springer Verlag, 1977.

[44] S. Kollias and D. Anastassiou, An adaptive least square algorithm for efficient training of artificial neural networks, *IEEE Transactions on Circuits Systems*, Vol.36, No.5, 1092-1101, 1989.

[45] E.B. Kosmatopoulos, M.M. Poly carpou, M.A. Christodoulou and P.A. Ioannou, High-Order Neural Network Structures for Identification of Dynamical Systems, *IEEE Transactions on Neural Networks*, Vol.6, No.2, 422-431, 1995.

[46] D. Kukolj and E. Levi, Identification of Conplex Systems Based on Neural Takagi-Sugeno Fuzzy Model, *IEEE Transactions on Systems, Man and Cybernetics*, Vol.34, No.1, 272-1282, 2004.

[47] L.Jin, P. N. Nikiforunk, M.M. Gupta, Adaptive Model Reference Control of Discrete-Time Nonlinear Systems Using Neural Networks, *Control-Theory and Advanced Technolnogy*, Vol.10, No.4, Part.3, 1373-1399, September 1999.

[48] C.-H. Lee and C.-C. Teng, Identification and Control of Dynamic Systems Using Recurrent Fuzzy Neural Networks, *IEEE Transactions on Fuzzy Systems*, Vol.8, No.4, 349-366, August 2000.

[49] F. L. Lewis, *Control of Robot Manipulators*, NJ10022, 1993.

[50] S. Limanond and J. Si, Neural- Network-Based Control Design: An LMI Approach, *IEEE Transactions on Neural Networks*, Vol.9, No.6, 1402-1429, November 1998.

[51] F.-J.Lin, R.-F. Fung and R.-J. Wai, Conparison of Sliding-Mode and Fuzzy Neural Network Control for Motor-Toggle Servomechanism, *IEEE Transactions on Mechatronics*, Vol.3, No.4, 302-318, December 1998.

[52] R.G.Lorenz, and S.P.Boyd, Robust Minimum Variance Beamforming, *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, VOL. 53, NO. 5, 1684-1696, 2005.

[53] J.T.Lo and T.Wanner, Existence and Uniqueness of Risk-Sensitive Estimates, *IEEE Trans. Automatic Control*, Vol.47, No.11, 1945-1

[54] D. Q. Mayne and H. Michalska, Receding Horizont Control for Nonlinear Systems, *IEEE Transactions on Automatic Control*, Vol.35, No.7, 814-824, November 1990.

[55] D. Muñoz and D. Sbarbaro, An Adaptive Sliding-Mode Controller for Discrete Nonlinear Systems, *IEEE Transactions on Industrial Electronics*, Vol.47, No.3, 574-581, June 2000.

[56] K.S.Narendra and K.Parthasarathy, Identification and Control of Dynamical Systems Using Neural Networks, *IEEE Trans. Neural Networks*, Vol.1, No.1, 4-27, 1990.

[57] S.A.Nazin and B.T.Polyak, Limiting Behavior of Bounding Ellipsoids for State Estimation, –.

[58] K.Nishiyama, K.Suzuki, H_∞-learning of layered neural networks, Neural Networks, *IEEE Trans.Neural Networks*, Vol.12, No. 6, 1265 - 1277, 2001

[59] H. N. Nounou and K. M. Passino, Stable Auto-Tuning of Adaptive Fuzzy /Neural Controllers for Nonlinear Discrete-Time Systems, *IEEE Transactions on Fuzzy Systems*, Vol.12, No.1, 70-83, February 2004.

[60] K. Ogata, *Discrete-time control systems*, Prentice Hall, Englewood Cliffis, NJ07458, 1995.

[61] A. G. Parlos, S. K. Menon and A. F. Atiya, An Algorithm Approach to Adaptive State Filtering Using Recurrent Neural Network, *IEEE Transactions on Neural Networks*, Vol.12, No.6, 1411-1432, 2001.

[62] A. G. Parlos, S. Parthasarath and A. F. Atiya, Neuro-Predictive Process Control Using On-Line Controller Adaptation, *IEEE Transactions on Control Systems*, Vol.9, No.5, 741-755, September 2001.

[63] S. Piché, B. S.-R., D. Jonson and Mark Gerules, Nonlinear Model Predictive Control Using Neural Networks, *IEEE Control Systems Magazine*, 53-62, June 2000.

[64] K. M. Passino and S. Yurkovich, *Fuzzy Control*. Menlo Park, CA: Addison-Wesley Longman, 1998.

[65] G. V. Puskorius and L. A. Feldkamp, Neurocontrol of Nonlinear Dynamical Systems with Kalman Filter Trained Recurrent Networks, *IEEE Transactions on Neural Networks*, Vol.5, No.2, 279-297, March 1994.

[66] K. Reif and R. Unbehauen, The Extended Kalman Filter as an Exponential Observer for Nonlinear Systems, *IEEE Transactions on Signal Processing*, Vol.47, No.8, 2324-2328, August 1999.

[67] K.Reif, S.Gunther, E.Yaz, R.Unbehauen, Stochastic stability of the continuous-time extended Kalman filter, *IEE Proceedings-Control Theory and Applications*, 1350-2379, 2000

[68] I. Rivals and L. Personnanz, Nonlinear Internal Model Control Using Neural Networks: Application to Processes with Delay and Design Issues, *IEEE Transactions on Neural Networks*, Vol.11, No.1, 80-90, January 2000.

[69] L.Ros, A.Sabater and F.Thomas, An Ellipsoidal Alculus Based on Propagation and Fusion, *IEEE Transactions on Systems, man, and Cybernetics*, Vol.32, No.4, 430-442, 2002.

[70] José de Jesús Rubio and Wen Yu, Nonlinear system identification with recurrent neural networks and dead-zone Kalman filter algorithm, *Neurocomputing*, 2007.

[71] José de Jesús Rubio and Wen Yu, Dead-zone Kalman filter algorithm for recurrent neural networks,*44rd IEEE Conference on Decision and Control, CDC'05*, Seville, Spain, 2562-2567, 2005.

[72] Wen Yu, José de Jesús Rubi, Xiaoou Li, Recurrent Neural Networks Training with Stable Risk-Sensitive Kalman Filter Algorithm, *2005 International Joint Conference on Neural Networks, IJCNN'05*, Montreal, Canada, 700-705, 2005.

[73] D. W. Ruck, S. K. Rogers, M. Kabrisky, P. S. Maybeck and M. E. Oxley, Comparative Analysis of propagación hacia atrás and the Extended Kalman Filter for Training Multilayer Perceptrons, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.14, No.6, 686-691, June 1992.

[74] D.E.Rumelhart, G.E.Hinton and R.J.Williams, Learning representations by back-propagating errors, *Nature*, Vol.323, 533-536, 1986

[75] M. C. Saaj, B. Bandyopadhyay and H. Unbehauen, A new Algorithm for Discrete-Time Sliding-Mode Control Using Fast output Sampling Feedback, *IEEE Transactions on Industrial Electronics*, Vol.49, No.3, 518-523, June 2002.

[76] M.G. Safonov, M. Athans, Robustness and computational aspects of nonlinear stochastic estimators and regulators, IEEE Trans. on Automatic Control, vol. 23, no. 4, 717-725, 1978.

[77] S. Z. Sarpturk, Y. Istefanopolos and O. Kaynak, On the Stability of Discrete-Time Sliding Mode Control Systems, *IEEE Transactions on Automatic Control*, Vol.32, No.10, 930-932, October 1987.

[78] R. S. Scalero and N. Tepelenlioglu, A Fast New Algorithm for Training Feedforward Neural Network, *IEEE Transactions on Signal Processing*, Vol.40, No.1, 202-210, January 1992.

[79] F.C.Schueppe, *Uncertain dynamic systems*, Englewood cliffs, Prentice-Hall, 1973

[80] S. Shah F. Palmeieri and M. Datum, Optimal filtering algorithms for fast learning in feed forward neural networks, *Neural Networks*, Vol.5, 779-787, 1992.

[81] S. Singhal and L. Wu, Training multilayer perceptrons with the extended Kalman algorithm, *Advances in Neural inform. Processing Syst. I*, 133-140, 1989.

[82] H. Sira-Ramirez, Nonlinear discrete variable strusture systems in quasi-sliding mode, *Int. J. Control*, No.5, 1171-1187, 1999.

[83] J. E. Slotine, W. Li, *Applied Nonlinear Control*, Macmillan Pub.Co., Englewood Cliffs, NJ 07632, 1991.

[84] J.Sum, C.L., G. H. Young and W. Kan, On the Kalman Filtering Method in Neural-Network Training and Pruning, *IEEE Transactions on Neural Networks*, Vol.10, No.1, 161-166, 1999.

[85] C.-H. Tsai and H.-Y. Chung, Neuro Sliding Mode Control With Its Appications to Seesaw Systems, *IEEE Transactions on Neural Networks*, Vol.15, No.1, 124-134, January 2004.

[86] V.I.Utkin, *Sliding Modes and Their Application in Variable Structure Systems*, MIR Publishers, Moscow, Soviet Union, 1978.

[87] V.I.Utkin, *Sliding Modes in Optimization and Control*, Springer-Verlag, 1992.

[88] L. X. Wang, *A Course in Fuzzy Systems and Control*, Prentice Hall, Englewood Cliffs, NJ 07458, 1997.

[89] E.Weyer, M.C. Campi, Non-asymptotic confidence ellipsoids for the least squares estimate, *39rd IEEE Conference on Decision and Control*, Sydney, Australia, 2688-2693, 2000.

[90] P. Whittle, *Risk-sensitive Optimal Control*, John Wiley & Sons, NY 10158-0012, USA, 1990.

[91] K. W. Wong, C. S. Leung and S. J. Chang., Use of periodic and monotonic activation functions in multilayer feedforward neural networks trained by extended Kalman filter algorithm, *IEE Image Signal Process*, Vol.149, No.4, 217-224, August 2002.

[92] S. Yamada, M. Nakashima and S. Shiono, Reinforcement Learning to Training a Cooperative Network with Both Discrete and Continuous Output Neurons, *IEEE Transactions on Neural Networks*, Vol.9, No.6, 1502-1508, November 1998.

[93] W.Yu, Nonlinear system identification using discrete-time recurrent neural networks with stable learning algorithms, *Information Sciences*, Vol.158, No.1, 131-147, 2004.949, 2002.

[94] X.Yu and G.Chen, Discretization Behaviors of Equivalent Control Based Sliding-Mode Control Systems, *IEEE Transactions on Automatic Control*, Vol.48, No.9, 1641-1646, 2003.

[95] Q. Zhu and L. Guo, Stable Adaptive Neurocontrol for Nonlinear Discrete- Time Systems, *IEEE Transactions on Neural Networks*, Vol.15, No.3, 653-662, May 2004.