UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO

INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA

CENTRO DE INVESTIGACIÓN EN TECNOLOGÍAS DE INFORMACIÓN Y SISTEMAS

MAESTRÍA EN CIENCIAS EN AUTOMATIZACIÓN Y CONTROL

REDES NEURONALES DINÁMICAS PARA LA IDENTIFICACIÓN Y CONTROL ADAPTABLE PARA SISTEMAS NO LINEALES

T E S I S

QUE PARA OBTENER EL GRADO DE MAESTRO EN CIENCIAS EN AUTOMATIZACIÓN Y CONTROL

PRESENTA:

JUAN FRANCISCO HERNÁNDEZ PÉREZ

ASESORES:

DR. RAÚL VILLAFUERTE SEGURA

DR. JOEL SUÁREZ CANSINO

PACHUCA HGO., MÉXICO 26 DE ENERO DE 2018

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO
Instituto de Ciencias Básicas e Ingeniería
Institute of Basic Sciences and Engineering
Área Académica de Computación y Electrónica
Computer Science and Electronics Department

Mineral de la Reforma, Hgo., a 10 de enero de 2018
Oficio No. MCAC01_2018

Lsc. Juan Francisco Hernández Pérez
P R E S E N T E

Por medio de la presente y en mi calidad de coordinador de la Maestría en Ciencias en Automatización y Control, del Área Académica de Computación y Electrónica (AACyE) de la Universidad Autónoma del Estado de Hidalgo (UAEH), me es grato informarle que el Jurado asignado para la revisión de su trabajo de tesis titulado: "**Redes neuronales dinámicas para identificación y control adaptable para sistemas no lineales**", dirigido por el Dr. Raúl Villafuerte Segura y el Dr. Joel Suárez Cansino, que para obtener el grado de Maestro en Ciencias en Automatización y Control fue presentado por usted, ha tenido a bien en reunión de sinodales, autorizarlo para impresión. A continuación se integran las firmas de conformidad de los integrantes del Jurado:

Dr. Omar López Ortega	(Presidente)	UAEH
Dra. Liliam Rodríguez Guerrero	(Secretario)	UAEH
Dr. Raúl Villafuerte Segura	(Vocal 1)	UAEH
M. en C. Omar Samperio Vázquez	(Vocal 2)	UAEH
Dr. Joel Suárez Cansino	(Vocal 3)	UAEH

Atentamente
"Amor, Orden y Progreso"

Dr. Jesús Patricio Ordaz Oliver
Coordinador de la Maestría en Ciencias en Automatización y Control
Universidad Autónoma del Estado de Hidalgo

c.c.p. Dr. Dr. Oscar Rodolfo Suárez Castillo, Director del Instituto de Ciencias Básicas e Ingeniería
c.c.p. Dr. Hugo Romero Trejo, Jefe del Área Académica de Computación y Electrónica
c.c.p. Expediente/ apl

Ciudad del Conocimiento
Carretera Pachuca - Tulancingo km. 4.5
Colonia Carboneras
Mineral de la Reforma, Hidalgo, México, C.P. 42184
Tel. +52 771 7172000 exts. 2250 y 2251
jesus_ordaz@uaeh.edu.mx

www.uaeh.edu.mx

Agradecimientos

El presente trabajo, fruto del esfuerzo de dos años, se puedo concluir gracias al apoyo de aquellas personas que formaron parte durante mi estancia en el posgrado. Por medio de las siguientes palabras quiero expresar mi agradecimiento.

Está dedicado a mis padres Rosa y Silverio, quienes me han educado y enseñado que en esta vida lo mas importante es ser feliz y no darme por vencido. Este logro también es de ustedes.

A mi hermana Lucia y mi hermano Luis, por darme ese apoyo y empujón para lograr esta meta. Fueron muchos los momentos en los que necesite de su ayuda muchas gracias, los quiero.

Agradezco al Dr. Raúl Villafurte Segura, por permitirme trabajar con usted, por su apoyo y mucha paciencia. Por cada una de sus enseñanzas y consejos a lo largo de mi estancia en el posgrado. También por creer en mi e impulsarme a ser mejor tanto en lo académico como en el ámbito social. Gracias.

Gracias al Dr. Joel Suárez Cansino, por brindarme todo el apoyo necesario durante el desarrollo de este proyecto, sin duda alguna me llevo muchos consejos que sin duda me guiaran por un mejor camino. También quiero agradecerle por la paciencia y tolerancia brindada.

Quiero agradecer a los sinodales: Dr. Omar López Ortega, Dra. Liliam Rodríguez Guerrero y M. en C. Omar Samperio Vazquez, por todo el apoyo y tiempo brindado durante el desarrollo de este trabajo de tesis. Gracias por sus comentarios y consejos.

Mis compañeros y amigos: Andrés, Ángel, Ismael, Jovani, Leopoldo, Nicolas e Itzamara, con quienes compartí cientos de momentos difíciles, frustrantes, divertidos y sobre todo de estudio. Gracias a todos ustedes por brindarme el apoyo incondicional y la motivación necesaria para concluir una meta mas en mi vida.

Gracias al Concejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico brindado durante mi estancia en el posgrado.

Finalmente agradezco a todas aquellas personas que participaron de alguna manera en la realización de este trabajo.

Acrónimos

- **3GDL:** 3 Grados de Libertad (3DOF por sus siglas en ingles, Degrees Of Freedom).
- **ADALINE:** Adaptable a Elementos Lineales (por sus siglas en inglés, ADAptative LINear Element).
- **E-L:** Euler-Lagrange.
- **GMLP:** Perceptrón Generalizado Multi-Capa (por sus siglas en inglés, Generalized Multi-Layer Perceptron).
- **GBRF:** Función Generalizada de Base Radial (por sus siglas en inglés, Generalized Radial Basis Function).
- **LMI:** Desigualdad Matricial Lineal (por sus siglas en inglés, Linear Matrix Inequalities).
- **MISO:** Múltiples entradas - Una Salida (por sus siglas en inglés, Multiple Input - Simple Output).
- **MIMO:** Múltiples entradas - Múltiples salidas (por sus siglas en inglés, Multiple Input - Multiple Output).
- **RHONN:** Red Neuronal Recurrente de Alto Orden (por sus siglas en inglés, Recurrent High Order Neural Network.)
- **RNA:** Red Neuronal Artificial.
- **SISO:** Una entrada - Una salida (por sus siglas en inglés, Single Input - Single Output).
- **SNL:** Sistema No Lineal.
- **T-S:** Takagi-Sugeno (modelo difuso, que lleva el nombre de sus diseñadores, Kazuko Yonekawa Takagi y M. Sugeno).
- **TRMS:** Twin Rotor MIMO System (sistema basado en un helicóptero de 2 grados de libertad, diseñado por la empresa FeedBack.)
- **W-H:** Widrow-Hoff (Algoritmo de aprendizaje para RNA, propuesto por Bernard Widrow y M. E. Huff).

Índice general

Índice de figuras

Índice de tablas

Resumen

En el presente trabajo de tesis se propone una ley de control linealizante para sistemas no lineales afines con múltiples-entradas y múltiples-salidas (Multiple-Input Multiple-Output, MIMO), cuya dinámica se asume desconocida o complicada de obtener. Para estimar la dinámica del sistema se emplean redes neuronales artificiales (RNA) de dos tipo: Perceptrón Generalizado Multi-Capa (GMLP), cuyo diseño esta dado por una estructura en capas y Redes Neuronales Recurrentes de Alto Orden (RHONN), las cuales se caracterizan por tener una conectividad bidireccional entre sus unidades de entrada. Esto con la finalidad de emplear la RNA que aproxime mejor las dinámicas desconocidas del sistema MIMO. Para fines ilustrativos de los resultados teóricos obtenidos, se presenta una simulación utilizando el entorno Simulink de Matlab sobre los modelos matemáticos del Carro-Péndulo y del Twin Rotor MIMO System.

Abstract

In this work the design of feedback linearizing control law for a class of nonlinear systems with multiple-input and multiple-output (MIMO) is presented. Here it is assumed unknown the dynamics or get complicated. To estimate the unknown dynamics of the system are employed two types Artificial Neural Networks (ANN): Generalized Multi-Layer Perceptron (GMLP), whose disign is given by layered structure and Recurrent High-Order Neural Network (RHONN), which are characterized by having a bidirectional connectivity between the inputs units. The above is with the purpose of obtaining the type of RNA that can best identify the dynamics of MIMO system. For illustration purposes of the theoretical results obtained, a simulation of Simulink-Matlab is presented using the mathematical models Car-Pendulum and Twin Rotor MIMO System.

Capítulo 1

Introducción

La mayoría de los sistemas eléctricos, mecánicos y electromecánicos del mundo real son inherentemente no lineales, no obstante, es posible linealizar dichos sistemas en una vecindad sobre un punto de operación o equilibrio. En el marco de la teoría de control de sistemas no lineales el análisis para determinar su representación en forma matemática en ocasiones puede ser compleja. Además, no existen métodos generales analíticos para obtener la solución de esta clase de sistemas [3]. Otro problema latente aquí, es el diseño de leyes de control eficientes que coadyuven a su manipulación.

En los últimos años se han desarrollado técnicas y/o metodologías que permiten el análisis y la manipulación de ciertas clases de sistemas no lineales, sin la necesidad de un modelo matemático. Las redes neuronales artificiales (RNA) son una de ellas. Estas técnicas son sistemas complejos que tratan de emular ciertas características propias del cerebro humano, tal como el resolver problemas matemáticos. Es posible implementar RNA para la solución de problemas que resultan laboriosos y complicados [4]. La identificación de dinámicas desconocidas completas y/o parciales de un sistema no lineal (SNL) en el mundo real, es uno de estos tópicos donde las RNA son una herramienta valiosa.

Es claro que uno de los principales temas en el marco de la Teoría de Control, es la manipulación de un sistemas dinámico. Por tal motivo, es de interés desarrollar, diseñar e implementar una estructura o ley de control que permita llevar el estado de un sistema a una consigna deseada. Sin embargo, existen otros fenómenos en los sistemas que pueden ser de gran relevancia, como es la robustez ante variaciones en el comportamiento del sistema y la atenuación de perturbaciones externas no medibles que son complicadas de satisfacer. Una de las técnicas empleadas para compensar los fenómenos anteriores, es el control adaptable.
Los controladores adaptables pueden modificar su comportamiento en respuesta a

cambios en la dinámica del sistema y las perturbaciones [5]. Dado esto se ha desarrollado una técnica que se adapta en los casos en que la dinámica cambia a lo largo del tiempo y en particular en la presencia de perturbaciones externas no medibles. Las leyes de control adaptables se apoyan en técnicas como los sistemas de RNA, que les permiten identificar y resolver los cambios externos que afectan la dinámica de un sistema por medio de la adaptación de los parámetros que influyen directamente sobre la dinámica del sistema, cambiando en cada instante de tiempo las ganancias de la ley de control para obtener una respuesta del sistema más eficiente ante variaciones paramétricas. Los sistemas de RNA, son implementados regularmente cuando se presentan perturbaciones e incertidumbres en la dinámica del sistema, ya que han demostrado ser eficientes sobre todo en la evaluación de procesos en tiempo real [6], sin embargo el hecho de identificar un conjunto de funciones lo vuelve más complicado, además como consecuencia del incremento de procesos esto puede consumir más recursos computacionales.

Un claro ejemplo de sistemas donde se pueden implementar este tipo de técnicas son los sistemas no lineales con dinámica rápida, es decir que los cambios y perturbaciones externos afectan de forma inmediata al comportamiento del sistema, ya que este comportamiento requiere del diseño y la sintetización de leyes de control más eficientes, sobre todo cuando se carece de un modelo matemático. Además, estas leyes de control deben ser robustas para compensar las variaciones paramétricas provocadas por el entorno que lo rodea.

1.1. Antecedentes

A continuación se presenta cómo están situadas las técnicas y/o metodologías que se implementan en este tema de tesis. Se muestra una pequeña introducción de dónde se encuentra y qué se ha hecho con temas relacionados a leyes de control adaptables con RNA difusas.

1.1.1. Redes Neuronales Artificiales

1936 - Alan Turing, fue el primero en construir máquinas capaces de realizar procesos con cierto grado de inteligencia definidas como autómatas, con el sentido de que dichas máquinas realizan alguna función específica de los seres humanos. Sin embargo, los primeros teóricos que propusieron los fundamentos de la computación neuronal fueron Warren McCulloch, un neurofisiólogo, y Walter Pitts, un matemático, quienes en 1943, presentaron una teoría acerca de la forma de trabajar de las neuronas del sistema nervioso. Fueron los primeros en modelar una red neuronal mediante circuitos

eléctricos [7].

1960 - Bernard Widrow y Marcial Hoff, diseñaron una red neuronal artificial a la que llamaron el modelo Adaline (ADAptative LINear Elements) esta fue la primer red neuronal aplicada para resolver un problema real (Filtros adaptativos para eliminar ecos en las líneas telefónicas) que se ha implementado comercialmente durante varias décadas [8].

Hoy en día el uso de RNA en aplicaciones de control es muy común en varios temas de aplicación. Entre los cuales se puede mencionar el trabajo de Park et al. [9], quien utiliza tres redes neuronales para construir el modelo de un sistema *3DOF Helicopter*, donde se divide en tres subsistemas SISO independientes, y un control PD aplicado a cada subsistema. Al igual en el trabajo de Witt et al [10] es utilizada una RNA para aproximar la dinámica de la planta con una estrategia de control predictivo aproximado (APC). Demostrando en este último trabajo el buen seguimiento y rechazo de perturbaciones.

La técnicas de RNA han demostrado un buen funcionamiento en comparación con otras metodologías. Esta combinación tiende a buscar vías para mejorar las RNA, en [11] se puede apreciar la implementación de RNA sobre leyes de control a un sistema de 2 grados de libertad. Estos sistemas usan métodos de aprendizaje basados en redes neuronales para identificar y/o bien obtener parámetros de control que mejoren el desempeño.

En el trabajo [6] se utilizó un modelo neuro-difuso del tipo Mamdani y Takagi-Sugeno para la identificación de la dinámica de un sistema no lineal de 3 GDL este modelo del sistema es comparado con modelos de referencia tipo NARMA (por sus siglas en inglés, Non-linear Auto-Regressive Moving Average). Como resultado se obtuvo que el método de identificación con las RNA en sistemas no lineales es buena.

La necesidad de crear leyes de control más robustas, precisas y menos demandantes en cuanto a energía, ha llevado al desarrollo de técnicas como las RNA. Al igual que los métodos inteligentes se han implementado técnicas adaptables que permiten ajustar ciertos parámetros del sistema.

1.1.2. Control Adaptable

En 1950 se inicia la investigación en control adaptable. El control adaptable se caracteriza por cambiar sus parámetros de acuerdo al comportamiento de la dinámica del sistema. El control clásico debido a que es realimentado con una ganancia

constante, sólo funciona en un rango de operación. Como consecuencia surge así la necesidad de un control que mapee el espectro de rango de operaciones, resultando así una técnica de esquemas de ganancia por rangos.

El esquema básico del control adaptable según Landau [2], está compuesto por un bucle principal de realimentación negativa, que al igual que los sistemas convencionales actúa un regulador, además de bucle en el que se mide cierto índice de funcionamiento, el cual es comparado con el índice deseado, esto genera un error que se procesa para ajustar de acuerdo a los parámetros para optimizar directamente sobre la señal de control [5].

Este tipo de control adaptable ha sido implementado en muchos dispositivos. Uno de ellos es el *3DOF Benchmark Helicopter* que es utilizado en el trabajo [12] con un enfoque de modelo de referencia implícita y leyes de control adaptable con método de "passification" junto con un enfoque de modelo de referencia implícita. En los siguientes trabajos implementan controles adaptables sobre la plataforma *3DOF Helicopter*: en el trabajo [13] se propone un modelo experimental de referencia con un algoritmo adaptable integrador de paso hacia atrás. Para el trabajo [14] se implementa un controlador óptimo de regulación cuadrática (LQR) utilizando una aproximación linealizada del sistema.

Recientemente las leyes de control se han complementado con varias técnicas, surgiendo así controles más robustos, precisos y eficientes, algunos trabajos que implementan técnicas de control adaptable, RNA y técnicas difusas con modelos de referencia, se encuentran en [15, 16, 17, 18].

1.2. Justificación

Actualmente existen múltiples técnicas y metodologías con el propósito de diseñar leyes de control eficientes para tareas de regulación y de seguimiento de sistemas no lineales. Sin embargo, muchas de estas leyes requieren de un modelo matemático del sistema a manipular y/o no consideran incertidumbres y/o perturbaciones inherentes del mundo real.

El diseño de una ley de control que conjugue técnicas neuronales y adaptables permitirá eficientar el desempeño de una sistema en el mundo real en tareas de seguimiento sin la necesidad de requerir de un modelo matemático.

1.3. Hipótesis

En los sistemas no lineales afines en los que es complicado obtener el modelo matemático, es posible identificarlo mediante la implementación de RNA tipo GMLP tal que se pueda garantizar una eficiente manipulación de los sistemas no lineales sin la necesidad de su modelo matemático.

Como consecuencia será posible implementar una ley de control adaptable utilizando dicha identificación aplicada a tareas de seguimiento, además de compensar las perturbaciones causadas por agentes externos.

1.4. Planteamiento del problema

Considere un sistema no lineal afín, cuya dinámica puede ser representada mediante la estructura de un ecuación diferencial de la forma

$$\dot{x}(t) = F(x) + G(x)u,$$

donde u es un vector de entrada; y F, G son de dimensiones adecuadas y cada una de sus entradas son funciones suaves y desconocidas. Uno de los problemas presentes en esta clase de sistemas es sintetizar leyes de control que permitan eficientar su desempeño en tareas de seguimiento, sin la necesidad de conocer su modelo matemático y ante la presencia de perturbaciones y/o incertidumbres inherentes del sistema.

1.5. Objetivos

Se determina el objetivo general así como los objetivos específicos para realizar este trabajo de tesis. Los objetivos específicos se establecen de tal forma que al concluir todos y cada uno de ellos, se alcance el objetivo general.

1.5.1. Objetivo general

Desarrollar una ley de control adaptable para la manipulación de sistemas no lineales de la clase afín mediante la aproximación de la dinámica del sistema empleando redes neuronales artificiales tipo GMLP y RHONN.

1.5.2. Objetivos específicos

- Diseñar redes neuronales de segunda generación tales como GMLP y RHONN, utilizando algoritmos de aprendizaje dinámico para ser implementadas en tareas de aproximación de funciones no lineales.

- Identificar qué algoritmo de aprendizaje proporciona un mejor rendimiento de identificación para cada RNA mediante el error entre la variable medible y la salida de la RNA, con el fin de asegurar una mejor aproximación de la dinámica del sistema.
- Diseñar una estructura de la cual se pueda obtener las funciones F y G mediante el uso de RNA, para poder sustituir la dinámica en la ley de control.
- Evaluar las Redes Neuronales Artificiales mediante la obtención del error cuadrático medio para verificar el desempeño durante la identificación de un SNL.
- Desarrollar una ley de control adaptable utilizando la dinámica aproximada del sistema mediante las RNA para controlar el seguimiento de trayectorias.
- Realizar la simulación de la ley de control adaptable utilizando RNA sobre un SNL cuya dinámica es desconocida para determinar el comportamiento de la ley de control antes de ser implementada.

1.6. Metodología

La metodología que se utiliza para llevar a cabo la realización de los objetivos específicos de este proyecto son:

- 1.-**Recopilación de información:** Conocer los alcances, propiedades y limitaciones del sistema actuado a considerar, con el propósito de identificar su funcionamiento y comportamiento. Dicha información sera útil para el desarrollo de este trabajo.
- 2.- **Análisis y diseño de una RNA:** Se realiza una investigación sobre las diferentes estructuras y metodologías de las RNA, con el fin de seleccionar aquellas que presenten un mejor desempeño en cuanto a procesamiento de datos en línea. Además de que su aproximación presente un error mínimo respecto a su objetivo.
- 3..- **Identificación del modelo matemático del sistema mediante RNA:** Mediante Redes Neuronales Artificiales se identificará el modelo matemático de un sistema no lineal.
- 4.- **Diseño de la ley de control adaptable:** En este punto se diseña una ley de control adaptable para sistemas de tipo no lineal afín que pueda ser representado de la forma $\dot{x} = F(x) + G(x)u$ mediante la obtencion de la metodología

E−L. La ley de control adaptable permitirá al sistema realizar el seguimiento de trayectorias, además de rechazar las perturbaciones.

- 5.- **Implementación de una ley de control adaptable sobre un sistema totalmente actuado:** Se determina el comportamiento de la ley de control adaptable sobre sistemas actuados mediante el seguimiento de trayectorias. Así mismo realizar la obtención de un análisis del comportamiento con dicho control.

1.7. Organización de la tesis

El presente trabajo está organizado de la siguiente manera:

- **Capítulo 1: Introducción.** Presentación de los objetivos, alcances, estado del arte, marco de referencia en la teoría del control, entre otros. Se presenta el planteamiento del problema, su justificación e hipótesis, así como la metodología y los objetivos a seguir para llevar a cabo este trabajo de tesis.
- **Capítulo 2: Marco teórico.** Se da un panorama de los elementos, herramientas y metodologías que se utilizan para desarrollar y obtener los resultados. Se enfatiza en la descripción de RNA y su importancia en la implementación de la ley de control adaptable. Se hace referencia a los modelos matemáticos de los sistemas no lineales sólo con fines de simulación y comparación.
- **Capítulo 3: Resultados principales.** En esta sección se muestra el desarrollo y la obtención del diseño de la ley de control adaptable. Los resultados que se generaron durante la simulación son mostrados en éste apartado. Además de una investigación previa del uso de RNA y una ley de control basada en el Par o torque calculado, los cuales son el aporte de este documento.
- **Capítulo 4: Conclusiones y Trabajos futuros.** Se describe lo más relevante sobre los resultados obtenidos. Se da solución al planteamiento del problema y se verifica que la hipótesis sea correcta. Además se establecen las posibles mejoras o alternativas posibles.
- **Apéndice:** Se encuentra información complementaria como los algoritmos que se implementaron en la simulación y los análisis de rendimiento y eficiencia de las RNA.

Capítulo 2

Marco Teórico

En el presente capítulo se hace una mención de los resultados teóricos necesarios para el pleno desarrollo del presente trabajo de investigación.

2.1. Sistemas no lineales

Un sistema es la combinación de componentes que actúan en conjunto para realizar una tarea específica. Un componente es una unidad particular del funcionamiento de un sistema. Un sistema se llama dinámico si su salida en el presente depende de una entrada en el pasado; en un sistema dinámico la salida cambia con el tiempo cuando no está en su estado de equilibrio [19].

En este trabajo se consideran sistemas dinámicos que son modelados por un número finito de ecuaciones diferenciales ordinarias de primer orden acopladas

$$\begin{aligned} \dot{x_1} = &\quad f_1(t, x_1, \cdots, x_n, u_1, \cdots, u_p), \\ \dot{x_2} = &\quad f_2(t, x_1, \cdots, x_n, u_1, \cdots, u_p), \\ \vdots &\qquad\qquad \vdots \\ \dot{x_n} = &\quad f_n(t, x_1, \cdots, x_n, u_1, \cdots, u_p), \end{aligned}$$

donde $\dot{x_i}$ denota la derivada x_i, con respecto a la variable del tiempo t y $u_1, u_2, \cdots, u_p$, son variables de entrada y $x_1, x_2, \cdots, x_n$ son las variables de estado, que representan la información del sistema dinámico de su pasado. Usualmente se denota en forma

vectorial para describir las ecuaciones anteriores.

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{bmatrix}, u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_p \end{bmatrix}, f(t,x,u) = \begin{bmatrix} f_1(t,x,u) \\ f_2(t,x,u) \\ \vdots \\ \vdots \\ f_n(t,x,u) \end{bmatrix}.$$

Es posible escribir las n ecuaciones diferenciales de primer orden como un vector de ecuaciones diferenciales de $n-$dimensión

$$\dot{x} = f(t,x,u), \tag{2.1}$$

la ecuación (2.1) es denominada como la ecuación de estado del sistema y nos referimos a x como el estado y a u como la entrada. Es posible definir la ecuación de la siguiente forma

$$y = h(t,x,u), \tag{2.2}$$

donde podemos denotar a y como un vector de salida de $q-$dimensión que se conforma de variables de interés particular de la dinámica del sistema. Por lo tanto, llamamos ecuación de salida a (2.1) e indicamos tanto a (2.1) y (2.2) juntos como modelo de espacio estado. Los modelos matemáticos de un sistema físico de dimensión finita no siempre pueden ser representados de la forma espacio-estado. Muchas veces no es posible representar explícitamente una entrada u, es decir el estado no forzado

$$\dot{x} = f(t,x). \tag{2.3}$$

Utilizar una ecuación de estado no forzada, no significa necesariamente que la entrada al sistema es cero. Puede ser que la entrada se ha especificado como una función del tiempo $u = \gamma(t)$, o bien una función de realimentación de estado, $u = \gamma(x)$, o ambos $u = \gamma(t,x)$. La sustitución de $u = y$ en (2.1) elimina u y produce un estado forzado, la ecuación siguiente surge cuando la función f no depende explícitamente del tiempo

$$\dot{x} = f(x), \tag{2.4}$$

en cuyo caso se dice que el sistema es autónomo o invariante en el tiempo. El comportamiento de un sistema autónomo es invariante a los cambios en el tiempo inicial. Si el sistema no es autónomo, entonces se llama no autónomo o variante en el tiempo [20].

2.2. Estabilidad de sistemas no lineales

La estabilidad de puntos de equilibrio generalmente se caracteriza en el sentido de *Lyapunov*, un matemático e ingeniero ruso que estableció las bases que hoy llevan su nombre.

El punto de equilibrio es un concepto importante al tratar con la ecuación de estados. Se dice que un punto $x = x^*$ en el espacio de estados es un punto de equilibrio de la ecuación (2.3) si posee la propiedad tal que siempre que el estado del sistema comience en x^* permanecerá en x^* para todo $t \to \infty$. Para el sistema autónomo (2.4), los puntos de equilibrio son las raíces reales de la ecuación, donde

$$f(x) = 0,$$

de otra forma el punto de equilibrio es inestable. Un punto de equilibrio se dice asintóticamente estable si todas las soluciones que se inicien cerca del punto de equilibrio no sólo permanecen cerca del punto de equilibrio, sino que además tienden hacia el equilibrio a medida que $t \to \infty$ [20].

Teorema 1 (Estabilidad de Lyapunov) *Consideremos el sistema autónomo*

$$\dot{x} = f(x), \tag{2.5}$$

donde $f : D \longrightarrow \mathbb{R}^n$ es un mapa localmente Lipschitz desde un dominio $D \subset \mathbb{R}^n$ en $\mathbb{R}^n$. Supongamos que $\bar{x} \in D$ es un punto de equilibrio de (2.5), es decir $f(\bar{x}) = 0$. Vamos a caracterizar y estudiar la estabilidad de $\bar{x}$. Por conveniencia, se asume que $\bar{x} = 0$.

Definición 1 *[20]. El punto de equilibrio $x = 0$ de (2.5) es*

- *Estable, si para cada $\epsilon > 0$ existe $\delta = \delta(\epsilon)$ tal que*

$$\|x(0)\| < \delta \Rightarrow \|x(t)\| < \epsilon \qquad \forall\ t \geq 0.$$

- *Inestable si no es estable.*

- *Asintóticamente estable, es estable y δ puede elegirse tal que*

$$\|x(0)\| \Rightarrow \lim_{t\to\infty} x(t) = 0.$$

2.3. Redes neuronales Artificiales

Las Redes neuronales artificiales (RNA) se definen como sistemas de mapeos no lineales cuya estructura se basa en principios biológicos de los sistemas nerviosos de humanos y animales. Consta de un número grande de procesadores simples ligados por conexiones con pesos. Las unidades de procesamiento se denominan neuronas. Cada neurona recibe entradas de otros nodos y genera una salida escalar simple que depende de la información local disponible. Las RNA simples se caracterizan por los siguientes elementos:

1 Un conjunto de unidades de procesamiento, llamadas neuronas.

2 Un estado de activación para cada neurona, equivalente a la salida de la unidad.

3 Conexiones entre las neuronas, generalmente definidas por un peso que determina el efecto de una señal de entrada en la unidad.

4 Una regla de propagación, que determina la entrada efectiva de cada neurona a partir de las entradas externas.

5 Una función de activación que actualiza el nuevo nivel de activación basándose en la entrada efectiva y la activación anterior.

6 Una entrada externa que corresponde a un término determinado como umbral (bias) para cada unidad.

7 Un método para reunir la información, correspondiente a la regla de aprendizaje.

8 Un ambiente en el que el sistema va a operar, con señales de entrada e incluso señales de error.

En muchas redes las unidades de proceso tienen respuesta de la forma [21]:

$$y = f\left(\sum_k \omega_k x_k\right), \tag{2.6}$$

donde x_k es la señal de salida de otros nodos o entrada externa. ω_k es el peso de las ligas de conexión, $f(\cdot)$ es la función de activación [1].

2.3.1. Función de activación

Es la regla que logra establecer el efecto de la entrada total $u(t)$ en la activación de una unidad k se denomina *función de activación o de transferencia* (F_k). La función f

de la ecuación (2.6) puede tomar diferentes funciones de activación lo cual facilita las aproximaciones que se requieran hacer, empleando RNA. En muchas ocasiones esta función es de la forma no decreciente respecto a la entrada total de la unidad [21],

$$y(t+1) = F_k\left(\sum_j \omega_j(t)x_j(t) + \theta(t)\right).$$

Algunas de la funciones de activación más usadas son las siguientes:

- Función Escalón: Si la suma de las entradas es mayor igual que el umbral de la neurona, la activación es 1. Si es menor, la activación es 0.
- Función Lineal: responde a la expresión $F_k(u) = u$.
- Función Sigmoidal: Esta función hace que se puedan utilizar reglas de aprendizaje definidas para la función escalón, con la ventaja, respecto a esta función, de que la derivada está definida en todo el intervalo de 0 a 1.

2.3.2. Neurona simple (SISO)

El desarrollo de una neurona fue por Pitts y McCulloch, quienes definieron la estructura y proceso que la determinan. En sus inicios éstas sólo aceptaban valores de entrada uno o cero, formando así los primeros casos representativos de la lógica como lo es el AND y OR lógicos. Una clase de neuronas más realistas, utiliza modelos que distinguen procesos de transmisión y procesamiento de datos los cuales se realizan en dos etapas [21].

- La primera es la suma algebraica de las entradas de las sinapsis, este es el potencial lento que es alimentado a una función no lineal.
- La segunda etapa se realiza con la evaluación de la función prescrita obteniendo el valor de salida de la neurona.

La estructura representada por una neurona general está conformada por una entrada, una operación algebraica, una función de activación y una salida [1].

Normalmente el diseño de una neurona contiene más de una entrada Figura 3.1. En una neurona con R entradas, cada una de las entradas $p_1, p_2, p_3, \cdots, p_R$ son ponderadas con elementos llamados pesos, correpondientes a $W_{1,1}, W_{1,2}, W_{1,3}, \cdots, W_{1,R}$ definiendo así una matriz de pesos $\mathbb{W}$.

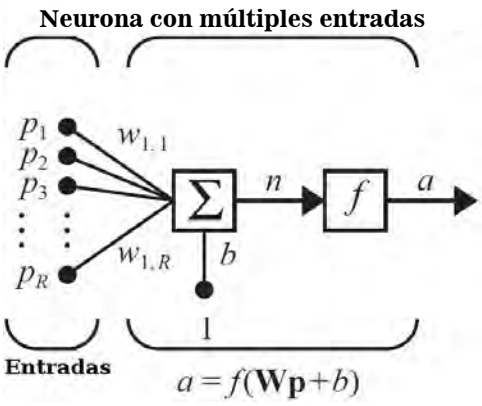


Figura 2.1: Estructura general de una neurona con múltiples entradas una salida [1].

La neurona tiene un Umbral b, el cual sumado a las entradas ponderadas con los pesos forman la entrada total n que es la sumatoria de los productos dendríticos más b, tal que :

$$n = W_{1,1}p_1 + W_{1,2}p_2 + \cdots + W_{1,R}p_R + b,$$

esta expresión puede ser escrita en forma de matrices

$$n = \mathbb{W}P + b,$$

donde la matriz $\mathbb{W}$ para el caso de una sola neurona tiene sólo una fila. Entonces podemos escribir a la salida de una neurona como

$$a = f(\mathbb{W}P + b).$$

Es posible representar a las redes neuronales como matrices, facilitando las expresiones y el manejo de información y operaciones, por lo que se usará la estructura de la Figura 2.2 para representar una neurona en este trabajo.

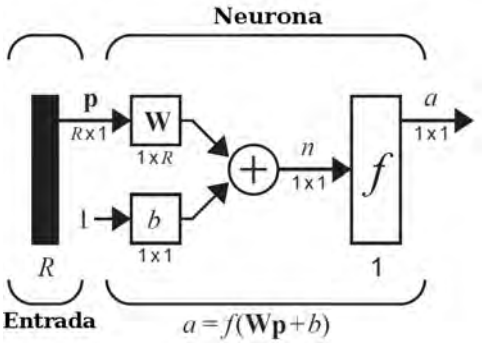


Figura 2.2: Estructura de una neurona con R entradas, con notación abreviada [1].

2.3.3. Método de aprendizaje Widrow-Hoff

El aprendizaje en una RNA se refiere a la implementación de una regla que permite la identificación de los pesos correctos, tal que se pueda resolver el problema asignado, generalmente son en reconocimiento de patrones. Este proceso es simple y automático, sin embargo no deja de ser un proceso recursivo que puede llegar a consumir más tiempo de lo deseado durante su desempeño.

El proceso de aprendizaje más utilizado en perceptrones es la **Regla Delta Generalizada** o **Backpropagation**, es una técnica para entrenar pesos de los perceptrones, para ello es necesario conocer los valores de entrada y la salida deseada de cada patrón de entrenamiento. No obstante se realiza el proceso de entrenamiento por cada capa para así determinar los pesos adecuados de cada una de ellas.

Su funcionamiento consiste en ajustar pesos y umbrales tratando de minimizar la suma del error cuadrático proveniente de la salida de la red y la salida deseada, como consecuente el resultado se acerca más a los objetivos. Este proceso se realiza de forma continua, cambiando los pesos de acuerdo a la dirección opuesta del error de la pendiente, esto es denominado como Técnica del Gradiente Descendiente. Cuando en el sistema se presentan entradas que nunca ha visto esta regla responde razonablemente durante la identificación de las salidas. Una de las propiedades de este tipo de entrenamiento y estructura es su capacidad de generalización.

El conjunto de entrenamiento está compuesto por medio del conocimiento de los valores de entrada ponderados por pesos :

$$\{p_1 w_1\}, \{p_2 w_2\}, \cdots, \{p_R w_R\},$$

el cual es representado por la sumatoria del producto de las entradas por los pesos, esto representa la salida deseada que aproxima una red de características lineales

$$\tilde{y} = \sum_{i=1}^{R} p_i \cdot w_i,$$

la salida deseada está dada por los valores y en cada $i-$ésimo instante de tiempo para cada entrada, asimismo se obtiene la salida aproximada de la RNA, por lo que es posible obtener el error definido por la salida desea menos la salida aproximada:

$$error = y - \left(\sum_{i=1}^{R} p_i \cdot w_i \right),$$

además el error permite calcular los nuevos pesos tal que sean los adecuados para obtener la salida deseada lo más cercana

$$W_{nuevos} = W_{anteriores} + \eta \cdot error \cdot P,$$

donde η es un escalar denominado tasa de aprendizaje que permite mejorar el desempeño de entrenamiento y $P \in \mathbb{R}^R$ son los valores de entrada [1].

2.4. RNA tipo Perceptrón Multi-Capa Generalizada

Una RNA de tipo Perceptrón Multi Capa Generalizada (Generalized Multi-Layer Perceptron, GMLP) y como su nombre lo indica dependiendo de su configuración puede poseer varias capas en su estructura. El diseño de una red neuronal tipo GMLP esta dado por un método de aprendizaje, una función de activación y una estructura en capas.

El método de aprendizaje está basado en la regla de gradiente extendido con la estructura de acuerdo con el algoritmo de backpropagation dado por la siguiente ecuación

$$X(t+T) = X(t) + \eta \frac{dy}{dx} s \cdot e(t) + Z(t), \tag{2.7}$$

siendo $e(t)$ el vector de error en el tiempo t compuesto por la diferencia entre y como referencia proveniente del sistema y y_s siendo la salida de la RNA, $X(t)$ es la matriz de pesos actualizada de acuerdo con la regla de gradiente extendido, η es la tasa de aprendizaje, $\frac{dy}{dx}s$ es la matriz jacobiana y T es el tiempo de muestreo.

La metodología de aprendizaje propuesta en la ecuación (2.7) se incluye $Z(t)$ que representa una contribución adicional de un error de seguimiento para mejorar el aprendizaje donde

$$Z(t) = \alpha D(t),$$

$$D(t+1) = \alpha \cdot D(t) - \eta \cdot \frac{dys}{dx} \cdot e(t),$$

la tasa de desintegración del filtro α se le llama *momento*, si $\alpha = 0$, la ley de actualización se reduce a la regla del gradiente clásico.

Como bien se sabe es indispensable utilizar la función de activación adecuada al tipo de problema a resolver por una RNA. Se determinó una función Sigmoidal

modificada, permitiendo trabajar con los valores sobre un rango específico, es decir realiza un mapeo con los datos de entrada. La función de activación que se utilizó es la siguiente

$$\bar{H}(s) = \frac{L + (U - L)}{(1 + e^{-s/P})}, \tag{2.8}$$

donde s es la entrada (escalar) a la función, L es el límite inferior, U el límite superior y P es la pendiente. La función se reduce a una sigmoidea habitual cuando $L = 0, U = 1$ y $P = 1.7$.

2.5. Redes Neuronales Recurrentes de Alto Orden (RHONN)

Los modelos de redes neuronales recurrentes (RNN) son caracterizadas por tener una conectividad bidireccional entre sus unidades de entrada. Esto es lo que las distingue de las redes neuronales feedforward. En las RNN con conexiones de primer orden, la información de cada estado de un valor o neurona es determinado por una ecuación diferencial de la siguiente forma

$$\dot{x}_i = a_i x_i + b_i \sum_j w_{ij} y_j, \tag{2.9}$$

donde x_i es el estado de la $i-$ésima neurona, a_i y b_i son constantes, w_{ij} son los pesos que se relacionan a la $j-$ésima entrada de la $i-$ésima neurona, cada y_j es una entrada externa o el estado de una neurona que pasa por una función sigmoidal, para cada $y_j = S(x_j)$ tal que

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ y_{n+1} \\ \vdots \\ y_{m+n} \end{bmatrix} = \begin{bmatrix} s(x_1) \\ \vdots \\ s(x_n) \\ s(u_1) \\ \vdots \\ s(u_m) \end{bmatrix},$$

donde

$$S(\upsilon) = \frac{1}{1 + e^{-\beta\upsilon}} + \varepsilon, \tag{2.10}$$

con β, $\varepsilon > 0$. Sea una RNN de segundo orden, la entrada no sólo es lineal, también puede ser la combinación de los componentes y_j y los productos $y_j y_k$. Además se puede proseguir de esta forma con interacciones de orden superior, esta clase de

redes neuronales forman una Red Neuronal Recurrente de Alto Orden (RHONN) [22] descrita por

$$\dot{x}_i = a_i x_i + \sum_{k=1}^{L} w_{ik} \prod_{j \in I_k} y_j^{d_j(k)},$$

es posible reescribir la ecuación anterior de forma general de tal forma que puede ser utilizada para cualquier orden recurrente, este tipo de RNA se considera la siguiente forma

$$\dot{\chi} = Ax(t) + W^T z(x(t), u(t)), \tag{2.11}$$

donde $x(t) \in \mathbb{R}^n$ es el estado de la RNA, $A = diag\left\{-a_1, \ldots, -a_n\right\}, a_i > 0$ es una matriz diagonal, $u \in \mathbb{R}^m$ es el vector de entrada, $W \in \mathbb{R}^{L \times n}$ es la matriz correspondiente a los pesos de la RNA. La función z esta definida como

$$z(x(t), u(t)) = \left[S(x_1), \ldots, S(x_n), S(u_1), \ldots, S(u_m)\right].$$

2.5.1. Ley de aprendizaje para una RHONN

Se considera el problema de aprendizaje de tal forma que pueda identificar sistemas dinámicos generales por lo que es indispensable ajustar de forma adaptativa los pesos del modelo para la RHONN.

Considerando el sistema de la ecuación (2.11), se define a la siguiente ecuación para ajustar los pesos correspondientes [23]

$$w_{ik}(t) = \upsilon_{ik}(t) + \varphi_{ik}(t)$$

donde

$$\begin{aligned}
\dot{\upsilon}_{ik} &= -\gamma \frac{e_i n_{ik}}{\zeta_{ik}} \\
\varphi_{ik}(t) &= \frac{\chi_i(t) n_{ik}}{\zeta_{ik}} + \eta_{ik}(t) \\
\dot{\eta}_{ik} &= -\chi_i \frac{d}{dt}\left(\frac{n_{ik}}{\zeta_{ik}}\right) - \sum_{t=1}^{l} \frac{w_{il}\dot{\zeta}_{il}}{\zeta_{ik}} n_{ik} \\
\dot{\zeta} &= -a_i \zeta + z.
\end{aligned}$$

con n una constante que satisface $\sum_{k=1}^{L} n_{ik} = 1$, γ es la tasa de aprendizaje, e_i es el error entre las variables $x_i - \chi_i$.

2.6. RNA en un entorno de programación y simulación

Existen dos importantes plataformas para el desarrollo de simulaciones e implementaciones de este tipo de proyectos, ambas plataformas cuentan con las herramientas necesarias para llevar acabo los objetivos de dicho proyecto.

La elección de la herramienta de software a elegir depende del sistema que se pretende controlar debido al hardware y especificaciones otorgados por el fabricante de acuerdo a cada sistema.

Simulink es un ambiente de diagramas de bloque para la simulación y el diseño basado en modelos físicos. Admite la generación automática de código y la prueba y verificación continuas de los sistemas embebidos. Simulink cuenta con un editor gráfico, bibliotecas de bloques personalizables y herramientas para modelar y simular sistemas dinámicos, se encuentra embebido con MATLAB, lo que permite incorporar algoritmos script de MATLAB en los modelos y exportar los resultados de la simulación al ambiente MATLAB para llevar a cabo el análisis de los resultados obtenidos en las simulaciones [24]. En el Apéndice A se muestran algunos ejemplos de RNA utilizando estas herramientas.

Red Neuronal con Perceptrón Multi−capa Generalizada (GMLP)

Al igual que en los ejemplos mostrados en el Apéndice A donde se realizan simulaciones de RNA utilizando módulos y funciones propias del software Matlab, con el objetivo de verificar que tipo de RNA ofrece un mejor desempeño en identificación de funciones no lineales. Del resultado de realizar las distintas pruebas de las funciones y módulos de Matlab, se muestra el siguiente ejemplo con el módulo de RNA tipo GMLP, con una entrada, una salida y además cuenta con una capa oculta de 10 neuronas a la cual se le pidió que aproximará la salida $Y = sin(t)$ donde $t \in [0, 10]$ con pasos de 0.001. Los resultados obtenidos se muestran en la la Figura 2.3 donde se realizaron en un tiempo real de $1.5s$ de $10s$ utilizando la simulación mediante la RNA.

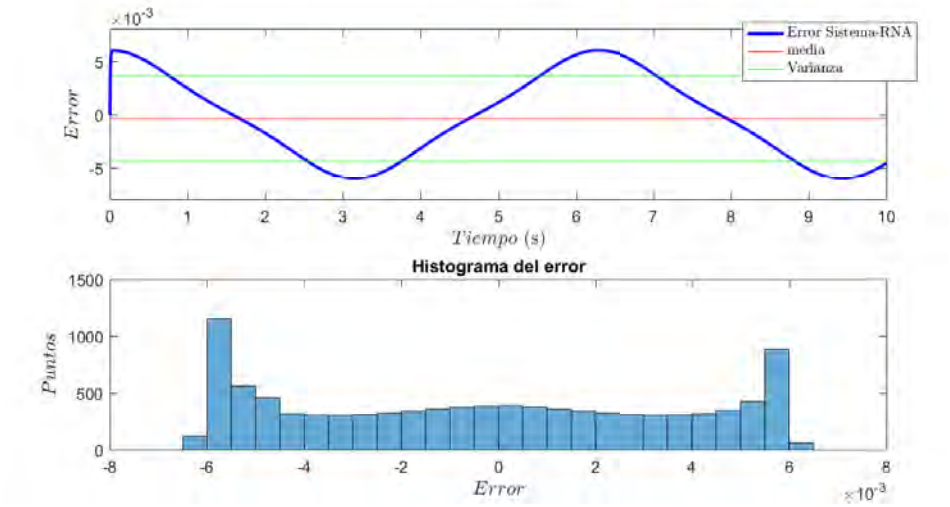


Figura 2.3: Análisis del error de la RNA GMLP de Matlab.

Como resultado del análisis del comportamiento de la RNA durante la aproximación de la función Y, podemos apreciar que la solución a los procesos es bastante rápido en comparación a todas las propuestas anteriores. Sin embargo aunque el error se encuentra en el rango $[-0.007, 0.007]$ no se puede decir que su precisión es muy exacta, no obstante se mantiene pequeño.

2.7. Modelo matemático de un sistema

Un modelo matemático se puede definir como cualquier tentativa de ecuaciones que den una predicción sobre el funcionamiento de un sistema en específico antes de que el sistema pueda diseñarse o construirse físicamente.

La dinámica de sistemas puede ser representada mediante el modelado matemático por medio del análisis de la respuesta de los sistemas dinámicos. Para la mayoría de los sistemas el modelo matemático que resulta se describe en términos de ecuaciones diferenciales.

La elaboración de modelos matemáticos se puede determinar al aplicar las leyes físicas a un sistema específico, tal que es posible desarrollar un modelo matemático que describa al sistema. En ocasiones la formulación de un modelo matemático puede resultar imposible. De ser así se puede utilizar un procedimiento de modelado experimental. En este procedimiento, se somete al sistema a un conjunto de entradas conocidas y se miden sus salidas. A partir de la relación entrada y salida se puede

obtener su modelo matemático [19].

El diseño y análisis para la elaboración de un modelo matemático es válido en la medida en que el modelo se aproxime al sistema físico. Como sabemos esto es muy complicado al determinar en los sistemas no lineales, sin embargo actualmente la rapidez con la cual una computadora puede procesar operaciones aritméticas nos permite incluir cientos de ecuaciones para describir un sistema y para determinar un modelo exacto, pero como consecuencia es muy complicado realizarlo. Si no se requiere de una exactitud extrema, es recomendable dearollar un modelo simplificado.

El modelo matemático nunca podrá representar un componente o sistema físico con precisión, sin embargo es posible realizar aproximaciones o suposiciones del modelo. El proceso para la obtención de un modelo matemático de un sistema puede resumirse como [19]:

1 Obtener un diagrama del sistema y definir las variables.

2 Mediante leyes y métodos científicos desarrollar las ecuaciones que representen a cada componente y obtener un modelo matemático.

3 Verificar la validez del modelo, tal que la predicción obtenida al resolver las ecuaciones del modelo se comparen con los resultados experimentales.

2.7.1. Modelado mediante ecuaciones de movimiento de Euler-Lagrange

Es un método estándar para la obtención de un modelo matemático que represente la dinámica de un robot manipulador de n grados de libertad formado por eslabones rígidos conectados por articulaciones. Este método está basado en las ecuaciones de movimiento de Euler- Lagrange.

Este procedimiento nos dice que la energía total ε (Hamiltoniano) del robot manipulador está dada por la suma de la energía cinética $\kappa(q, b\dot{f}q)$ y la energía potencial $\upsilon(q)$:

$$\varepsilon(q, \dot{q}) = \kappa(q, \dot{q}) + \upsilon(q),$$

donde $q, \dot{q} \in \mathbb{R}^n$ representan a los vectores de posición y velocidad articular respectivamente.

El lagrangiano $\mathcal{L}(q, \dot{q})$ de un robot se define como la diferencia entre la energía cinética y la potencial, tal que

$$\mathcal{L}(q, \dot{q}) = \kappa(q, \dot{q}) - \upsilon(q). \tag{2.12}$$

Las ecuaciones de movimiento de E-L para un robot manipulador de n gdl están dadas por

$$\frac{d}{dt}\left[\frac{\partial \mathcal{L}(q, \dot{q})}{\partial \dot{q}}\right] - \frac{\partial \mathcal{L}(q, \dot{q})}{\partial q} = \tau - \vartheta(\dot{q}, \tau),$$

donde $q = [q_1, q_2, \ldots, q_n]^T \in \mathbb{R}^n$ define al vector de posiciones articulares o coordenadas generalizadas, $\dot{q} = [\dot{q}_1, \dot{q}_2, \ldots, \dot{q}_n]^T \in \mathbb{R}^n$ es el vector de velocidades articulares, $\tau = [\tau_1, \tau_2, \ldots, \tau_n]^T$ es el vector asociado a los pares aplicados y $\vartheta(\dot{q}, \dot{f}_e) \in \mathbb{R}^n$ es el vector de fuerzas o pares de fricción. La energía cinética tiene una estructura matemática cuadrática bien definida en función de la velocidad articular

$$\kappa(q, \dot{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q},$$

en la cual $M(q) \in \mathbb{R}^{n \times n}$ es la matriz de inercias del robot, y es una matriz definida positiva. Sin embargo otro aspecto de $\upsilon(q)$ es que tiene dependencia exclusivamente del vector de posición q, ya que se considera con campos conservativos como los de la fuerza de gravedad. Las ecuaciones de movimiento E-L pueden ser reescritas de una forma compacta de acuerdo al lagrangiano, como:

$$\frac{\partial \mathcal{L}(q, \dot{q})}{\partial \dot{q}} = \frac{\partial}{\partial \dot{q}}\left[\frac{1}{2}\dot{q}^T M(q)\dot{q}\right] - \frac{\partial \upsilon(q)}{\partial \dot{q}} = M(q)\dot{q}.$$

El modelo dinámico de un robot manipulador da una descripción completa entre los pares aplicados a los servomotores y el movimiento de la estructura mecánica. Con las ecuaciones de movimiento de E-L puede ser obtenido el modelo del sistema de manera sistemática e independiente del sistema de referencia coordenado. Dejando al sistema de ecuaciones de movimiento de E-L en términos de los pares aplicados τ obtenemos la siguiente representación

$$\tau = M(q)\ddot{q} + \dot{M}(q)\dot{q} - \frac{\partial}{\partial q}\left[\frac{1}{2}\dot{q}^T M(q)\dot{q}\right] + \frac{\partial \upsilon(q)}{\partial \dot{q}} + f_f(\dot{q}, f_e),$$

y reescribiendo la ecuación anterior de forma general, obtenemos un modelo dinámico para un robot manipulador de n GDL en su forma compacta y con la notación más ampliamente utilizada en el área de robótica [25],

$$\tau = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) + f_f(\dot{q}, f_e). \tag{2.13}$$

Sistema Carro Péndulo invertido

El péndulo invertido sobre una base móvil es un sistema subactuado que consiste en una varilla que se mueve libremente de forma circular. Esta varilla se encuentra montada sobre una base móvil tipo carro que se mueve en un riel horizontal rectilíneo gracias a una fuerza externa τ, que es la ley de control de la posición del carro y la varilla.

Para proponer estructuras de control a este tipo de dispositivos es necesario usar simulaciones previamente y para ello se requiere la descripción matemática de su comportamiento. De acuerdo a la metodología de E-L que utiliza la energía total del sistema, el modelo matemático del péndulo en general se describe como:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau, \tag{2.14}$$

donde $q, \dot{q}, \ddot{q} \in \mathbb{R}^n$ son el vector de posición, velocidad y aceleración articular del péndulo respectivamente, $M(q) \in \mathbb{R}^{n\times n}$ es la matriz de Inercias, $C(q,\dot{q}) \in \mathbb{R}^{n\times n}$ es la matriz de Coriolis y fuerza Centrípeta, $G(q) \in \mathbb{R}^n$ es el par gravitacional. Aplicando las ecuaciones de Euler−Lagrange para el carro y la varilla se obtienen las siguientes matrices [26],

$$\underbrace{\begin{bmatrix} m_p + m_c & -m_p l_2 cos(q_2) \\ -m_p l_2 cos(q_2) & (m_p * (l_2^2)) + I \end{bmatrix}}_{M(q)} \begin{bmatrix} \ddot{q_1} \\ \ddot{q_2} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & \frac{m_p l_2 sin(q_2)\dot{q_2}}{2} \\ \frac{m_p l_2 sin(q_2)\dot{q_2}}{2} & 0 \end{bmatrix}}_{C(q,\dot{q})} \begin{bmatrix} \dot{q_1} \\ \dot{q_2} \end{bmatrix} +$$

$$\underbrace{\begin{bmatrix} 0 \\ -m_p l_2 sin(q_2) \end{bmatrix}}_{G} g = \begin{bmatrix} \tau \\ 0 \end{bmatrix}. \tag{2.15}$$

En el Tabla 2.1 se especifican los valores de las constantes del modelo matemático definido anteriormente .

Sistema Twin Rotor MIMO System

El sistema TRMS abreviatura por sus siglas en inglés, es un modelo basado en un helicóptero con algunas especificaciones significativas. Está unido a una torre y la velocidad y posición se controla a través de la variación de velocidad del rotor. Sin embargo las características dinámicas más importantes en un helicóptero son capturadas en este modelo. Al igual que en un helicóptero real hay un acoplamiento cruzado significativo entre dos rotores.

Parámetro	Símbolo	Valor
Masa del péndulo y del carro	$m_c + m_p$	1.462
Distancia al centro de masa	L	0.016
Momento de inercia	I	0.024
Fricción viscosa positiva	$b+$	4.200
Fricción viscosa negativa	$b-$	4.120
Fricción de Coulomb positiva	f_c+	1.260
Fricción de Coulomb negativa	f_c-	1.604
Fricción estática positiva	f_e+	0.700
Fricción estática negativa	f_e-	1.000

Tabla 2.1: Descripción de los parámetros del sistema Carro-Péndulo invertido.

El sistema TRMS cuenta con dos entradas representadas por los rotores y las salidas están dadas por los ángulos verticales y horizontales así como las velocidades angulares [27].

Las ecuaciones que representan la dinámica del sistema TRMS están dados por sus dos grados de libertad en *pitch* y *yaw* denotados por ψ y ϕ respectivamente. La dinámica completa del sistema puede ser representada de la forma Espacio-estado con las siguientes ecuaciones

$$
\begin{aligned}
&\frac{d}{dt}\psi = \dot{\psi} \\
&\frac{d}{dt}\dot{\psi} = \frac{a_1}{I_1}\tau_1^2 + \frac{b_1}{I_1}\tau_1 - \frac{M_g}{I_1}sin(\psi) + \frac{0.0326}{2I_1}sin(2\psi)\dot{\varphi}^2 - \frac{B_{1\psi}}{I_1}\dot{\psi} - \frac{k_{gy}}{I_1}cos(\psi)\dot{\varphi}(a_1\tau_1^2 + b_1\tau_1) \\
&\frac{d}{dt}\phi = \dot{\phi} \\
&\frac{d}{dt}\dot{\phi} = \frac{a_2}{I_2}\tau_2^2 + \frac{b_2}{I_2}\tau_2 - \frac{B_{1\phi}}{I_2}\dot{\phi} - \frac{k_c}{I_2}1.75(a_1\tau_1^2 + b_1\tau_1 \\
&\frac{d}{dt}\tau_1 = -\frac{T_{10}}{T_{11}}\tau_1 + \frac{k_1}{T_{11}}u_1 \\
&\frac{d}{dt}\tau_2 = -\frac{T_{20}}{T_{21}}\tau_2 + \frac{k_2}{T_{21}}u_2,
\end{aligned}
$$

en la Tabla 2.2 se especifican los parámetros y constantes de las ecuaciones anteriores. Para el sistema de ecuaciones se describen los vectores de los estados y salidas dados por

$$x = [\psi \quad \dot{\psi} \quad \varphi \quad \dot{\varphi} \quad \tau_1 \quad \tau_2]^T, \qquad y = [\psi \quad \varphi]^T.$$

Parámetro	Símbolo	Valor
Momento de inercia del rotor vertical	I_1	$6.8x10^{-2}kg-m^2$
Momento de inercia del rotor horizontal	I_2	$2x10^{-2}kg-m^2$
Parámetro de característica estática	a_1	0.0135
Parámetro de característica estática	b_1	0.0924
Parámetro de característica estática	a_2	0.02
Parámetro de característica estática	b_2	0.09
Momento de la Gravedad	M_g	$0.32N-m$
Parámetro de momento de fricción	$B_{1\psi}$	$6x10^{-3}N-m-s/rad$
Parámetro de momento de fricción	$B_{1\varphi}$	$1x10^{-1}N-m-s/rad$
Parámetro de momento del Giroscopio	K_{gv}	$0.05s/rad$
Ganancia del motor 1	k_1	1.1
Ganancia del motor 2	k_2	0.8
Parámetro de motor 1	T_{11}	1.1
Parámetro de motor 1	T_{10}	1
Parámetro de motor 2	T_{21}	1
Parámetro de motor 2	T_{20}	1
Parámetro de momento de reacción cruzada	T_p	2
Parámetro de momento de reacción cruzada	T_0	3.5
Ganancia de momento de reacción cruzada	k_c	-0.2

Tabla 2.2: Descripción de los parámetros del sistema TRMS.

2.8. Control adaptable

El término adaptable significa cambiar el comportamiento conforme a nuevas circunstancias. El control adaptable es un tipo especial de control no lineal en el que el estado del proceso puede ser separado en dos escalas de tiempo que evolucionan a diferente velocidad. La escala lenta corresponde a los cambios de los parámetros y por consiguiente a la velocidad con la cual los parámetros del regulador son modificados, y la escala rápida que corresponde a la dinámica del bucle ordinario de realimentación. El esquema básico de control adaptable [2] puede verse en la Figura 2.4, está compuesto de un bucle principal de ralimentación negativa, en el que actúa igual que en los sistemas convencionales un regulador además de otro bucle en el que se mide un cierto índice de funcionamiento, el cual es comparado con el índice deseado y se procesa el error en un mecanismo de adaptación que ajusta los parámetros del regulador y en algunos casos actúa directamente sobre la señal de control. Un sistema de control adaptable básicamente está formado por tres partes importantes: *Un controlador primario, un modelo de referencia* y la *ley de adaptación*. El controlador

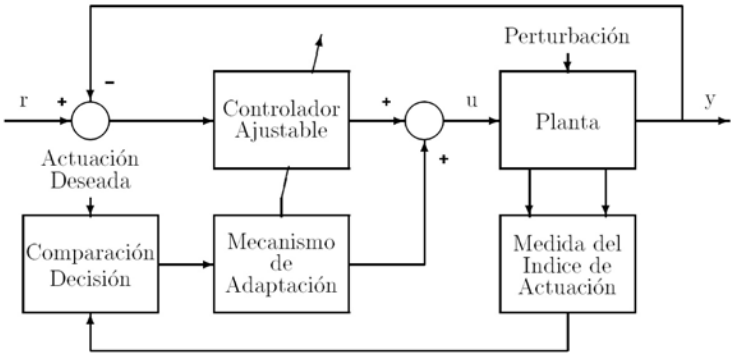


Figura 2.4: Configuración básica de control adaptativo [2].

primario puede tener en principio cualquiera de las configuraciones conocidas para el diseño de controladores lineales. Sin embargo debe cumplir la condición de que sea posible que el conjunto del proceso y el controlador puedan reproducir el modelo de referencia. El Modelo de referencia, especifica el comportamiento deseado en bucle cerrado. Se da usualmente de forma paramétrica. El modelo elegido debe ser sensible a la dinámica del proceso. Por ello la elección del modelo de referencia no es fácil, eligiéndose normalmente un modelo conservador [5].

2.8.1. Control dinámico: Realimentación de estado linealizante

Este tipo de control toma como base la dinámica de un sistema, cuya obtención contempla la energía y los torques necesarios para describir su comportamiento. La obtención de la dinámica del sistema por medio de E-L nos permite fácilmente realizar este tipo de control.

El control por Par o Torque calculado es un controlador basado en modelo general cuya representación es (2.13) para robots de N GDL. Además de ser una clase de sistemas no lineales con grado relativo r_n y de dimension p de la siguiente forma:

$$\dot{x} = \{f(x)_{r_1}, \ldots, f(x)_{r_n}\}^T + \sum_{i=1}^{p} \{g_{r_1}(x), \ldots, g_{r_n}(x)\}\, u,$$

$$y = \{h(x)_{r_1}, \ldots, h(x)_{r_n}\}^T,$$

donde podemos transformar este sistema de ecuaciones como un sistema no lineal general de la forma (2.1) con la suposición de un controlador que incluye las siguientes

dinámicas

$$\begin{aligned} \dot{x} &= F(x) + G\left\{\alpha(x) + \beta(x)u\right\}, \\ y &= H(x), \end{aligned} \tag{2.16}$$

donde se determinan los siguientes subsistemas:

$$\begin{aligned} \dot{x_r} &= f_r(x_r) + g_r\left\{\alpha_r(x) + \beta_r(x)u\right\}, \\ y_r &= h_r(x), \end{aligned}$$

para toda $r = 1, \ldots, n$. Además se asume que el par (F, G) es controlable y β es no singular para todo x. Si $\alpha(x)$ y $\beta(x)$ son conocidos exactamente, estos parámetros pueden ser cancelados por medio de una realimentación de estado linealizante de la forma

$$u = \beta^{-1}(x)\left\{\nu - \alpha(x)\right\}, \tag{2.17}$$

para el sistema generalizado (2.16). Al sustituir u obtenemos un sistema lineal

$$\begin{aligned} \dot{x} &= F(x) + G(x)\nu \\ y &= H(x). \end{aligned} \tag{2.18}$$

Este tipo de sistemas se define como *completamente linealizables* donde la ecuación de estado es linealizada. Es posible describir un controlador no lineal a partir de la dinámica obtenida mediante la realimentación del modelo dinámico de un sistema [28].

2.8.2. Diseño de un controlador basado en un error de seguimiento filtrado

Este tipo de controlador es usualmente utilizado en dispositivos que son capaces de realizar tareas de seguimiento. Para su implementación es necesario conocer el grado relativo (r) del sistema y el número de funciones que describen al sistema, ya que se deben obtener los errores de cada variable a seguir, tal que se pueda definir el error de seguimiento como:

$$\begin{aligned} e_1(t) = &\quad y_{d1}(t) - y_1(t), \\ \vdots &\qquad\qquad \vdots \\ e_p(t) = &\quad y_{dp}(t) - y_p(t), \end{aligned} \tag{2.19}$$

en las que se asume que las trayectorias deseadas y_{di}, $i = 1, \ldots, p$, son funciones en el tiempo, conocidas, acotadas y r_i veces diferenciables cuyas derivadas también son

acotadas y conocidas.

Como bien es sabido existen ciertas propiedades que se les atribuyen a las funciones: proporcional, integral y derivativa. Utilizando estas propiedades la cual nos dice que para la propiedad proporcional obtenemos una ganancia respecto al error. La propiedad integral nos genera el área de acumulación del error, y por último la derivada indica la razón de cambio respecto al error. Basándose en las propiedades anteriores podemos definir al error de seguimiento filtrado [29] como

$$\begin{array}{rcl} s_1(t) = & \left(\frac{d}{dt}+\lambda_1\right)^{(r_1-1)} e_1(t) + \alpha_1 \int_0^t e_1(\theta)d\theta + \gamma_1 \int_0^t \int_0^t e_1(\theta)d\theta, \\ \vdots & \vdots & \\ s_p(t) = & \left(\frac{d}{dt}+\lambda_p\right)^{(r_p-1)} e_p(t) + \alpha_p \int_0^t e_p(\theta)d\theta + \gamma_p \int_0^t \int_0^t e_p(\theta)d\theta, \end{array} \tag{2.20}$$

donde $\lambda_i, \alpha_i, \gamma_i > 0$ para $i = 1, \ldots, p$. Note que, si $s_i(t) \to 0$ cuando $t \to \infty$, entonces, cada término del lado derecho de la Ecuación (2.20) también converge asintóticamente a cero, es decir supongamos que $\lim_{t\to\infty} s_i(t) = 0$, esto implica que

$$\begin{array}{l} \lim_{t\to\infty}\left(\left(\frac{d}{dt}+\lambda_i\right)^{(r_i-1)} e_i(t) + \alpha_i \int_0^t e_i(\theta)d\theta + \gamma_i \int_0^t \int_0^t e_i(\theta)d\theta\right) = 0, \\ \lim_{t\to\infty}\left(\left(\frac{d}{dt}+\lambda_i\right)^{(r_i-1)} e_i(t)\right) + \lim_{t\to\infty}\left(\alpha_i \int_0^t e_i(\theta)d\theta\right) \\ \qquad + \lim_{t\to\infty}\left(\gamma_i \int_0^t \int_0^t e_i(\theta)d\theta\right) = 0, \\ \Rightarrow \lim_{t\to\infty}\left(\left(\frac{d}{dt}+\lambda_i\right)^{(r_i-1)} e_i(t)\right) = 0, \\ \lim_{t\to\infty}\left(\alpha_i \int_0^t e_i(\theta)d\theta\right) = 0, \\ \lim_{t\to\infty}\left(\gamma_i \int_0^t \int_0^t e_i(\theta)d\theta\right) = 0. \end{array}$$

Además, note que $\lim_{t\to\infty}(\dot{e}_i(t)) = 0$, si $e_i(t) \to 0$. De la misma manera $\lim_{t\to\infty}\left(\alpha_i \int_0^t e_i(\theta)d\theta\right) = 0$, si $e_i(t)$ oscila alrededor de una vecindad con centro en cero, es decir, si existe un $\epsilon > 0$ tal que $e_i(t) \in V_\epsilon(0)$. Lo mismo ocurre con el $\lim_{t\to\infty}\left(\gamma_i \int_0^t \int_0^t e_i(\theta)d\theta\right)$, y $\lim_{t\to\infty}(\lambda_i e_i(t)) = 0$, si y sólo si $e_i(t) \to 0$ cuando $t \to \infty$.

Como consecuente, se determina que, si $s_i(t) \to 0$ cuando $t \to \infty$, entonces $e_i(t)$ oscila alrededor de cero y eventualmente el $\lim_{t\to\infty}(\lambda_i e_i(t)) = 0$, tal que la ley de control force a $s_i(t) \to 0$ asintóticamente, por lo tanto $e_i(t) \to 0$ también es asintóticamente para $i = 1, \ldots, p$.

Observe que si se deriva el error de seguimiento filtrado de la ecuación (2.20) podemos sustituir al error del grado relativo del sistema, tal que éste puede ser introducido

como

$$\begin{aligned} \dot{s}_1 &= \nu_1 - f_1(x) - \textstyle\sum_{j=1}^{p} g_{1j}(x)u_j, \\ &\vdots \\ \dot{s}_p &= \nu_p - f_p(x) - \textstyle\sum_{j=1}^{p} g_{pj}(x)u_j, \end{aligned} \tag{2.21}$$

donde $\nu_1, \ldots, \nu_p$ están definidas por

$$\begin{aligned} \nu_1 &= y_{d1}^{(r_1)} + \beta_{1_{r_1-1}} e_1^{(r_1-1)} + \ldots + \beta_{1_1}\dot{e}_1 + \alpha_1 e_1 + \gamma_1 \textstyle\int_0^t e_1(\theta)d\theta, \\ &\vdots \\ \nu_p &= y_{dp}^{(r_p)} + \beta_{p_{r_p-1}} e_p^{(r_p-1)} + \ldots + \beta_{p_1}\dot{e}_p + \alpha_p e_p + \gamma_p \textstyle\int_0^t e_p(\theta)d\theta, \end{aligned}$$

con

$$\beta_{ij} = \frac{(r_i-1)!}{(r_i-j)!(j-1)!}\lambda_i^{r_i-j}, \qquad i = 1, \ldots, p, \qquad j = 1, \ldots, r_i - 1.$$

De forma general, podemos denotar las funciones del error de seguimiento filtrado como

$$\begin{aligned} s(t) &= [s_1(t), \ldots, s_p(t)]^T, \\ \nu(t) &= [\nu_1(t), \ldots, \nu_p(t)]^T, \end{aligned}$$

por lo que es posible reescribir la ecuación (2.21) de tal forma que

$$\dot{s} = \nu - F(x) - G(x)u. \tag{2.22}$$

Si asumimos conocidas a las funciones $F(x)$, $G(x)$ y ν, es posible implementar una ley de control en lazo cerrado como en la ecuación (2.17) tal que

$$u = G^{-1}(x)\left(-F(x) + \nu + K_0 s\right), \tag{2.23}$$

donde $K_0 = diag\left[k_{01}, \ldots, k_{0p}\right]$ con $k_{0i} > 0$ para $i = 1, \ldots, p$. Al sustituir la ley de control (2.23) en el sistema (2.22), obtenemos

$$\dot{s} = -K_0 s(t), \tag{2.24}$$

lo que implica que $s_i(t) \to 0$ cuando $t \to \infty$. Por lo tanto $e_i(t) \to 0$ cuando $t \to \infty$ [29].

2.9. Conclusión

El marco teórico presentado anteriormente contiene los conceptos básicos para el desarrollo de este proyecto, los cuales contemplan conceptos de sistemas no lineales y su dinámica. En el primer apartado se muestra un breve conocimiento sobre el mundo de las RNA donde, se hace mención de algunas configuraciones y algoritmos de aprendizaje que permitan identificar funciones matemáticas de forma rápida y precisa. Como segunda instancia se muestran las herramientas para comprender el uso de la ley de control propuesta en el Capítulo III, así como las variables y funciones para la simulación e implementación de dicho control.

Capítulo 3

Resultados principales

En este capítulo se proporciona la metodología para el diseño y análisis de la ley de control adaptable la cual puede aplicarse a cierto tipo de sistemas no lineales afín, además de mencionar la metodología propuesta para el diseño y estructura de redes neuronales artificiales para la identificación de la dinámica del sistema.

Esta unidad se divide en dos secciones. En la primera parte se desarrolla una ley de control adaptable utilizando RNA tipo GMLP, la cual asume que el sistema puede ser considerado de la forma (2.13) y su configuración se adapta a este tipo de sistema. En la segunda parte se desarrolla la misma ley de control adaptable, pero la identificación del sistema se realiza mediante RNA tipo RHONN, las cuales consideran al sistema de la forma (2.1) y debido que es un sistema de forma general, que puede ser tratado como (2.13).

3.1. Primera propuesta de control para sistemas no lineales

Se desarrolla la implementación de una ley de control adaptable utilizando un error de seguimiento filtrado y la aproximación de la dinámica del sistema mediante RNA tipo Perceptrón Muti-Capa Generalizada.

3.1.1. RNA tipo GMLP para Carro-Péndulo

Es importante destacar que el conocimiento de la estructura del modelo matemático del sistema es sólo para determinar algunos parámetros para la ley de control y simulación. Sin embargo se consideran desconocidas las funciones no lineales que lo componen.

De acuerdo a la dinámica de un sistema modelado mediante ecuaciones de Euler-Lagrange en el cual se describe detalladamente la relación entre la fuerza y el movimiento es posible modelar cualquier sistema no lineal. Como bien sabemos las ecuaciones E-L para un sistema mecánico están descritas por (2.13). A continuación se describe la dinámica de un sistema de cuarto orden con dos grados de libertad.

Para el diseño de nuestro controlador es necesario llevar nuestro sistema no lineal (2.13) a la forma $\dot{x} = f(x, u)$. Despejando a $\ddot{q}$ de la ecuación (2.13) tenemos

$$\ddot{q} = M^{-1}(q)(\tau - C(q, \dot{q})\dot{q} - G(q)), \tag{3.1}$$

donde

$$\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} = \begin{bmatrix} H_{11}^{-1} & H_{12}^{-1} \\ H_{21}^{-1} & H_{22}^{-1} \end{bmatrix} \left\{ \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} - \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} - \begin{bmatrix} G_{11} \\ G_{21} \end{bmatrix} \right\},$$

se definen las variables de estado $x_1 = q_1$, $x_2 = \dot{x}_1 = \dot{q}_1$, $x_3 = q_2$, $x_4 = \dot{x}_3 = \dot{q}_2$, además de reescribir las ecuaciones de tal forma que el sistema sea afín, por lo tanto el sistema reescrito es el siguiente

$$\begin{bmatrix} \ddot{x}_2 \\ \ddot{x}_4 \end{bmatrix} = \underbrace{- \begin{bmatrix} H_{11}^{-1} & H_{12}^{-1} \\ H_{21}^{-1} & H_{22}^{-1} \end{bmatrix} \left\{ \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} G_{11} \\ G_{21} \end{bmatrix} \right\}}_{F(x)} + \underbrace{\begin{bmatrix} H_{11}^{-1} & H_{12}^{-1} \\ H_{21}^{-1} & H_{22}^{-1} \end{bmatrix}}_{G(x)} \underbrace{\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}}_{u}, \tag{3.2}$$

la representación del sistema es de acuerdo al modelo matemático del carro péndulo, donde los valores de H_{ij}, C_{ij} y G_{ij} están definidos en (2.15). Generalizando la estructura de la ecuación (3.2) tenemos que

$$\dot{x} = F(x) + G(x)u. \tag{3.3}$$

Una vez se tiene la forma (3.3) es posible realizar un cambio de variable de acuerdo al número de estados disponibles que se encuentran en el sistema, donde p es el número de funciones no lineales suaves. Además del orden relativo del sistema, el cual es considerado de segundo orden y está denotado por $r = 2$.

$$\begin{aligned} y_1^{r_1} &= f_1(x) + \sum_{j=1}^{p} g_{1j}(x)u_j, \\ &\vdots \\ y_p^{r_p} &= f_p(x) + \sum_{j=1}^{p} g_{pj}(x)u_j, \end{aligned}$$

donde de manera generalizada puede ser expresada cada función y definiendo $y_1 = \psi$ y $y_2 = \varphi$. Así se tiene que

$$y = \begin{bmatrix} \psi \\ \dot{\psi} \\ \varphi \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} y_1^1 \\ y_1^2 \\ y_2^1 \\ y_2^2 \end{bmatrix},$$

como resultado del cambio de variable se tiene la siguiente ecuación que representa la dinámica del sistema (3.3) tal que

$$y = F(x) + G(x)u, \tag{3.4}$$

el diseño de la ley de control se determina con base en el error de seguimiento filtrado, para ello se genera un error para cada variable del sistema definido por la diferencia entre la referencia desea y_d y la salida del sistema y tal que para el sistema (3.3) existen los siguientes errores

$$\begin{aligned} e_1(t) &= y_{d1}(t) - y_1(t), \qquad e_2(t) = y_{d2}(t) - y_2(t), \\ \dot{e}_1(t) &= \dot{y}_{d1}(t) - \dot{y}_1(t), \qquad \dot{e}_2(t) = \dot{y}_{d2}(t) - \dot{y}_2(t), \\ \ddot{e}_1(t) &= \ddot{y}_{d1}(t) - \ddot{y}_1(t), \qquad \ddot{e}_2(t) = \ddot{y}_{d2}(t) - \ddot{y}_2(t), \end{aligned}$$

la ley de control se desempeña con base en ciertas propiedades, las cuales se describen en el Capítulo 2. El desarrollo del error de seguimiento para el sistema (3.3) se describe a continuación.

Observe que en este caso, $y_1 = \psi$ y $y_2 = \varphi$, utilizando al error y la propiedad natural de la derivada, la cual nos indica el cambio de velocidad respecto al error, además de usar la propiedad de la integral, la cual disminuye el error en estado estacionario. Utilizando estas propiedades se propone las siguientes ecuaciones donde $i = 1, 2$:

$$s_1(t) = \left(\frac{d}{dt} + \lambda_1\right)^{(1)} e_1(t) + \alpha_1 \int_0^t e_1(\theta)d\theta + \gamma_1 \int_0^t \int_0^\varphi e_1(\theta)d\theta d\varphi,$$

$$s_2(t) = \left(\frac{d}{dt} + \lambda_2\right)^{(1)} e_2(t) + \alpha_2 \int_0^t e_2(\theta)d\theta + \gamma_1 \int_0^t \int_0^\varphi e_2(\theta)d\theta d\varphi,$$

debido a que en la ecuación anterior se define a la posición s_i del error de seguimiento filtrado y se requiere conocer la velocidad de acuerdo a los criterios de estabilidad, es necesario obtener la derivada respecto al tiempo de s_1 y s_2, de tal forma que

$$\dot{s}_1(t) = \lambda_1 \dot{e}_1(t) + \alpha_1[e_1(t) - e_1(0)] + \gamma_1 \int_0^t e_1(\theta)d\theta + \ddot{e}_1(t),$$

$$\dot{s}_2(t) = \lambda_2 \dot{e}_2(t) + \alpha_2[e_2(t) - e_2(0)] + \gamma_2 \int_0^t e_2(\theta)d\theta + \ddot{e}_2(t),$$

la propuesta de este error de seguimiento filtrado y la obtención de su derivada como se muestra en la ecuación anterior, note que queda un término igual al grado relativo del sistema, por lo que es posible tomar la dinámica del sistema y sustituirla en la ecuación $\dot{S}_i$ de tal manera que para cada derivada del error de seguimiento, obtenemos que $\ddot{e}_1 = \ddot{y}_{d1} - \ddot{y}_1$ y $\ddot{e}_2 = \ddot{y}_{d2} - \ddot{y}_2$, nuestra ecuación quedaría como

$$\dot{s}_1(t) = \lambda_1 \dot{e}_1(t) + \alpha_1[e_1(t) - e_1(0)] + \gamma_1 \int_0^t e_1(\theta)d\theta + \ddot{y}_{d1} - (f_1(x) + g_1 x u),$$

$$\dot{s}_2(t) = \lambda_2 \dot{e}_2(t) + \alpha_2[e_2(t) - e_2(0)] + \gamma_2 \int_0^t e_2(\theta)d\theta + \ddot{y}_{d2} - (f_2(x) + g_2 x u),$$

de forma general se agrupan los términos que contienen las dinámicas respecto al error de seguimiento en el vector ν de tal forma que el sistema del error de seguimiento filtrado quede en términos del sistema real, las ecuaciones del error de seguimiento filtrado son las siguiente

$$\dot{s}_1(t) = \nu_1 - f_1(x) - g_1 x u,$$

$$\dot{s}_2(t) = \nu_2 - f_2(x) - g_2 x u,$$

donde

$$\nu_1 = \lambda_1 \dot{e}_1(t) + \alpha_1[e_1(t) - e_1(0)] + \gamma_1 \int_0^t e_1(\theta)d\theta + \ddot{y}_{d1}.$$

$$\nu_2 = \lambda_2 \dot{e}_2(t) + \alpha_2[e_2(t) - e_2(0)] + \gamma_2 \int_0^t e_2(\theta)d\theta + \ddot{y}_{d2}.$$

reescribiendo las ecuaciones obtenemos

$$\dot{S}(t) = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} - \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} - \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} u,$$

$$\dot{S}(t) = \nu - F(x) - G(x)u. \tag{3.5}$$

Se propone la ley de control linealizante, donde $\tau = u$ con

$$u = G^{-1}(x)\left[-F(x) + \nu + Ks\right], \tag{3.6}$$

donde G^{-1} del sistema existe, $K \in \mathbb{R}^{n\times n}$ es una matriz definida positiva y s es el vector proveniente del error de seguimiento filtrado.

3.1.2. Identificación del sistema usando RNA tipo GMLP

Para el diseño de una RNA se debe tener en cuenta, la simplicidad y rapidez con la que realice las tareas asignadas. Para ello existen diferentes tipos de metodologías de aprendizaje y estructuras de redes que se especializan en ciertas tareas de acuerdo a las necesidades requeridas. Es posible con la combinación de estos criterios conseguir la propiedades y necesidades adecuadas.

El número de neuronas que se propuso fue de acuerdo a la experiencia, ya que se determinó después de varias pruebas con base en el error mínimo durante la identificación de una función no lineal. Asimismo se determinó que se utilizaría solo una capa oculta ya que es suficiente para resolver problemas de funciones no lineales.

De acuerdo a las ecuaciones mostradas en el Capítulo II, donde se describe el comportamiento de cada neurona y la forma en que aprende, como consecuente, se definió una estructura para la realización en tareas de seguimiento de funciones no lineales. Se determino a través del conocimiento práctico y empírico que una RNA de al menos una capa oculta resolviera la identificación de una función del sistema no lineal (3.4), por lo que si definimos como una red neuronal artificial a *Rna* tenemos que

$$y \approx \underbrace{\begin{bmatrix} Rna_{f_1} \\ Rna_{f_2} \end{bmatrix}}_{\hat{F}(x)} + \underbrace{\begin{bmatrix} Rna_{g_{11}} & Rna_{g_{12}} \\ Rna_{g_{21}} & Rna_{g_{22}} \end{bmatrix}}_{\hat{G}(x)} u,$$

$$\dot{y} \approx \hat{F}(x) + \hat{G}(x)u, \tag{3.7}$$

donde

$$\begin{array}{ll} & Rna_{g_{11}} \approx g_{11}(x), \\ Rna_{f_1} \approx f_1(x), & Rna_{g_{12}} \approx g_{12}(x), \\ Rna_{f_2} \approx f_2(x), & Rna_{g_{21}} \approx g_{21}(x), \\ & Rna_{g_{22}} \approx g_{22}(x), \end{array}$$

de acuerdo a lo anterior se propone la siguiente estructura de redes neuronales artificiales, tal que puedan aproximar la dinámica deseada. Para cada *Rna* se determinan las entradas N_i y salidas N_o previamente. El tipo de red neuronal que se utilizó para la investigación es GMLP y cuenta en su estructura con una capa oculta en la que de acuerdo al proceso a realizar o bien el usuario pueden determinar el número de neuronas N_h en esta capa, tal que

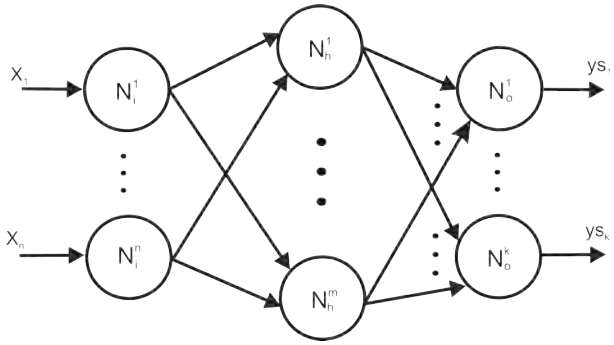


Figura 3.1: RNA con una capa oculta y múltiples entradas - múltiples salidas.

$$\begin{matrix} N_i^1 & N_h^1 & N_o^1 \\ \vdots & \vdots & \vdots \\ N_i^n & N_h^m & N_o^k \end{matrix}$$

donde la salida de la red neuronal artificial está dada por y_s, y el entrenamiento de cada neurona se determina por el error e_r dado por la diferencia entre y con y_s. Se define a cada $R_n a$ como una composición de funciones:

$$y_s = H_3\left(H_2\left(H_1\left(N_i\right)\right)\right),$$

donde H_i es la función de activación correspondiente a cada capa de la RNA la cual está definida por (2.8), además, de que para cada H_i se determinan los pesos adecuados obtenidos mediante (2.7). De acuerdo al diseño de la RNA anterior y conociendo el sistema (2.15) se propone la siguiente arquitectura para determinar la dinámica del sistema (3.3),

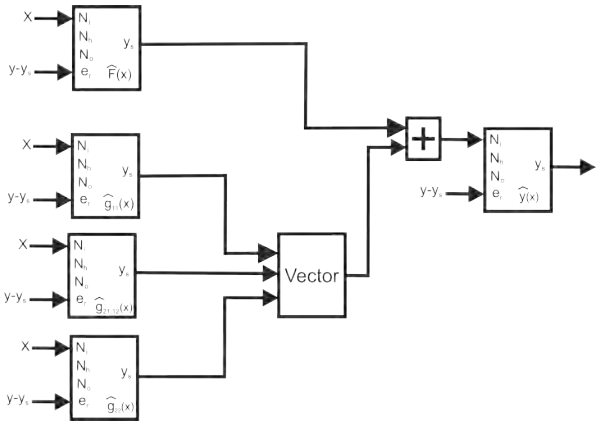


Figura 3.2: Estructura general de la RNA para determinar el sistema (3.3).

Se debe tomar en cuenta que la estructura de una RNA contenga sólo lo necesario para resolver el problema deseado. Para ello se contempla evitar el sobreajuste, el cual se presenta al incluir más procesos de los necesarios o bien muy pocos procesos como para lograr obtener el resultado esperado. De acuerdo a esto se realizó un análisis para determinar el número de neuronas que utilizaría la RNA en su capa oculta. Para llegar al objetivo deseado se realizaron varias pruebas de las cuales sólo se muestran tres ya que se aprecia considerablemente el resultado,

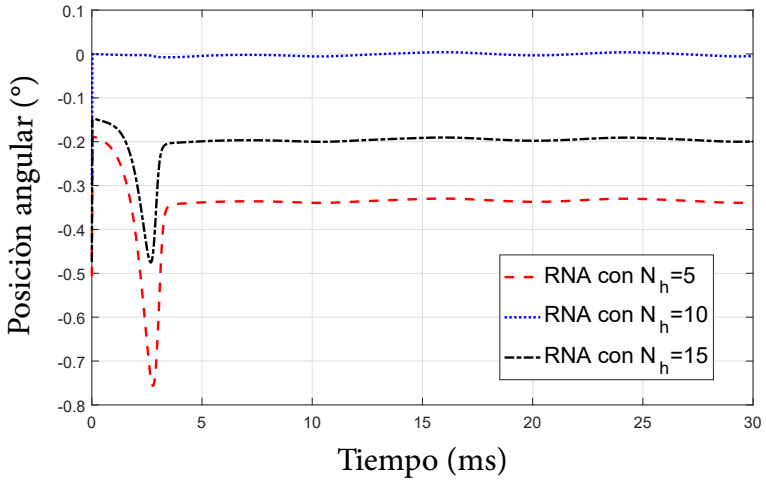
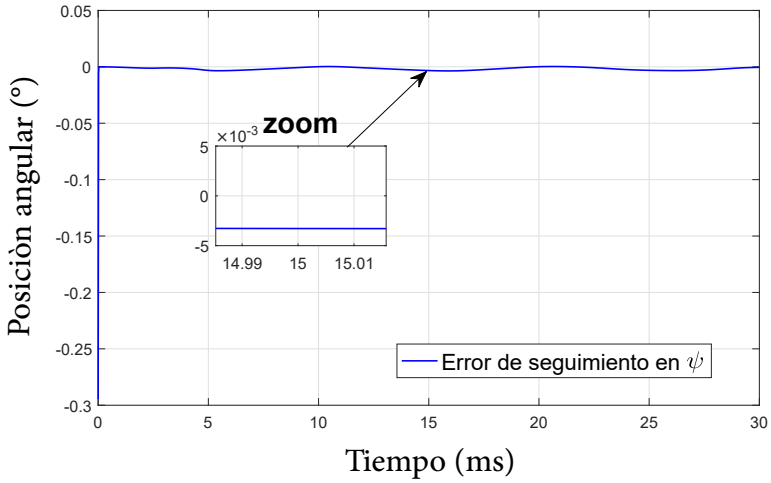


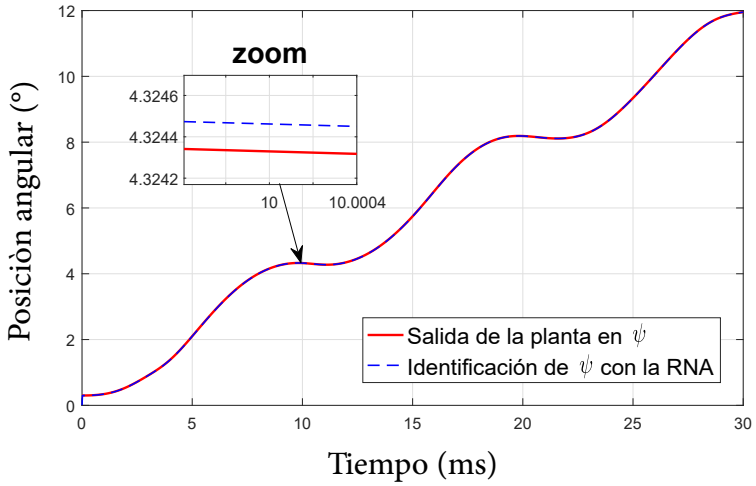
Figura 3.3: Respuesta del análisis del número de neuronas en la capa oculta.

de acuerdo a la gráfica anterior, la RNA mostró un mejor desempeño de identificación de funciones no lineales con una configuración de 10 neuronas en la capa oculta. Debido a que con un menor número de neuronas presenta una falta de información para obtener el resultado deseado, por otra parte tener más neuronas de lo necesario puede inducir a un conflicto en la obtención de la información. Por tal motivo se busco un equilibrio de procesos dado por 10 neuronas determinado empíricamente.

Sin pérdida de generalidad, se realizará la aproximación de las funciones F y G de un sistema Carro-Péndulo. La estructura de la RNA propuesta se ilustra en la Figura 3.2. La entrada X se determinó como aquella señal que perturba al sistema internamente definida por $u = [0.1\text{sin}(0.6t), 0.2cos(0.4t)]$ a lo largo de $t \in [0, 30]$ segundos, la identificación se realizó en una referencia igual a cero y como resultado se obtuvo lo siguiente:

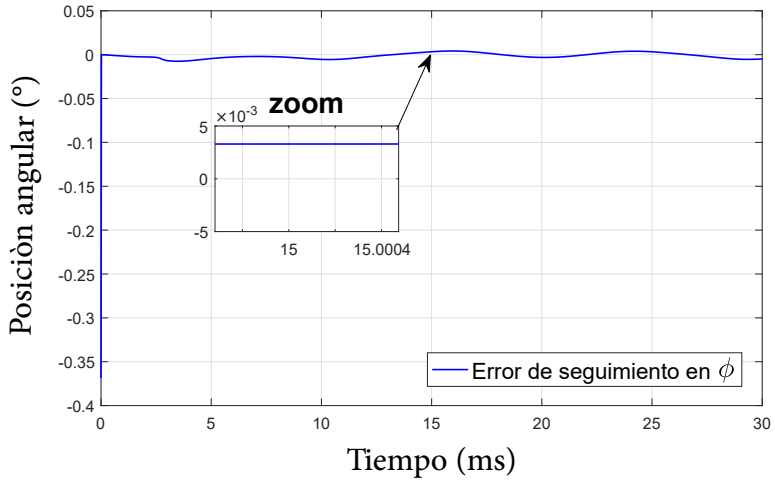


(a) Error en posición ψ

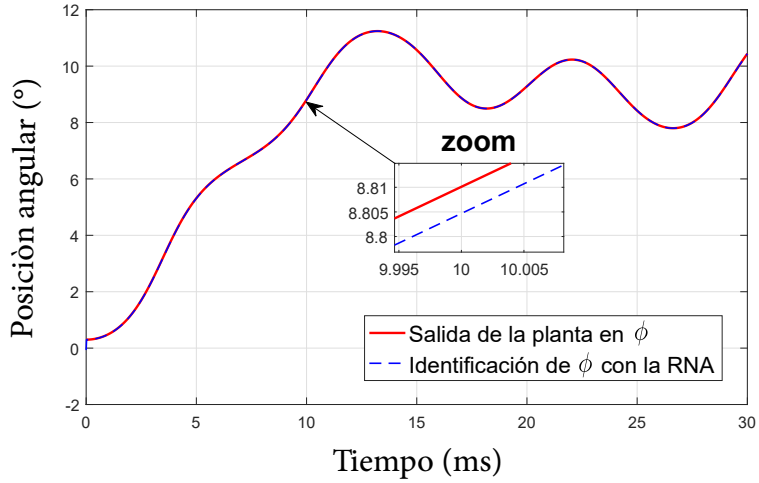


(b) Identificación en ψ

Figura 3.4: Error de identificación de la dinámica del sistema en ψ.



(a) Error en posición ϕ.



(b) Identificación en ϕ.

Figura 3.5: Error de identificación de la dinámica del sistema en ϕ.

Como se puede apreciar en las Figuras 3.4 y 3.5, donde las variables del sistema ψ y ϕ respectivamente, son identificadas mediante la RNA tipo GMLP en los cuales sus parámetros están determinados para un $\alpha = 0.5$, $N_i = 2$ y $N_h = 10$, las salidas de la RNA cambian dependiendo la función que identifiquen. No obstante, los resultados obtenidos muestran que dicha identificación produce un error entre las variables reales y las obtenidas dentro de una vecindad que oscila entre ± 0.005 centímetros. Por lo que, la RNA tipo GMLP se considera adecuada.

3.1.3. Implementación de resultados utilizando RNA tipo GMLP

A continuación se implementa la metodología utilizada en el diseño de la ley de control adaptable y configuración de una RNA aplicada a sistema no lineales mediante el uso de redes tipo GMLP.

Identificación del sistema usando RNA tipo GMLP

Tomando en cuenta las ecuaciones (2.14) de la dinámica del péndulo obtenida mediante E-L y bajo la ley de control de la ecuación (3.6) se determinó la identificación de las funciones $F(x)$ y $G(x)$ del sistema mediante la RNA.

El uso de una RNA para identificar las funciones no lineales no asegura que sean exactamente iguales, sin embargo, es posible aproximarlas. Para ello se utiliza una RNA para identificar cada función no lineal. La aproximación depende del método de aprendizaje propuesto en la ecuación (2.7) en la que se determinan ciertos valores para disminuir el error de aproximación,

$$e_r = y - y_s,$$

donde y es la señal a identificar (Planta) y esta debe ser aproximada a la señal y_s que se obtiene de la RNA, si el error e_r es lo más cercano a un ϵ muy pequeño se puede decir que la identificación es satisfactoria. Además, y_s esta definida por:

$$y_s = \hat{F}(x) + \hat{G}(x)u, \tag{3.8}$$

tal que $\hat{F}(x)$ y $\hat{G}(x)$ son las aproximaciones obtenidas la RNA.

Se propuso la siguiente configuración de una RNA tipo GMLP para obtener la identificación de la dinámica del sistema.

Para obtener la aproximación de $\hat{F}(x)$ y $\hat{G}(x)$ se deben conocer las entradas a la GMLP las cuales se determina de acuerdo al número de variables en el error e_r, las salidas de la red son igual al número de funciones en $F(x)$ y $G(x)$. La estructura fue propuesta de tal forma que se asemejara a la ecuación que representa la dinámica del sistema en la que:

la primer RNA tipo GMLP aproxima las funciones de $F(x)$

$$\hat{F}(x) = \begin{bmatrix} \hat{f}_1 \\ \hat{f}_2 \end{bmatrix}. \tag{3.9}$$

La segunda RNA tipo GMLP aproxima las funciones de $G(x)$

$$\hat{G}(x) = \begin{bmatrix} \hat{g}_{11} & \hat{g}_{12} \\ \hat{g}_{21} & \hat{g}_{22} \end{bmatrix}, \tag{3.10}$$

como bien se sabe en este tipo de sistema se tiene disponible el estado de posición tanto del carro y la varilla mediante sensores. Debido a que la identificación del sistema se hace respecto a esta variable y la dinámica del sistema obtenida es respecto a $\ddot{q}$ quien es la aceleración de las posiciones, entonces se definió una RNA más como en la Figura 3.2 para aproximar la función integral, tal que la salida final y_s de la RNA es la posición aproximada del sistema real.

Utilizando librerías de la herramienta de Simulink es posible implementar dichas ecuaciones y estructuras de la RNA. De acuerdo a lo anterior se establece la siguiente configuración:

Parámetro	RNA_1	RNA_2	RNA_3
N_i (neuronas de entrada)	2	2	4
N_h (neuronas en la capa oculta)	10	10	10
N_o (neuronas de salida)	2	2	2
η (tasa de aprendizaje)	0.1	0.3	0.3
α (momentum)	0.1	0.5	0.5
Condición inicial	1	1	1
Muestra de tiempo	0.001	0.001	0.001

Tabla 3.1: Configuración de los parámetros de la RNA tipo GMLP.

Para poder decidir si la configuración anterior de la RNA es ideal se le dio la tarea de identificar al sistema (3.1) el cual fue excitado con la ley de control (3.6). Se proporciona una referencia a seguir donde $Ref_\psi = 0$ y $Ref_\phi = sin(t)$. Las condiciones iniciales del sistema se definen en $\psi = 0.3$ y $\phi = 0.3$ con la finalidad de ver la reacción en un diferente punto de operación. Como resultado se obtuvierón las siguientes gráficas:

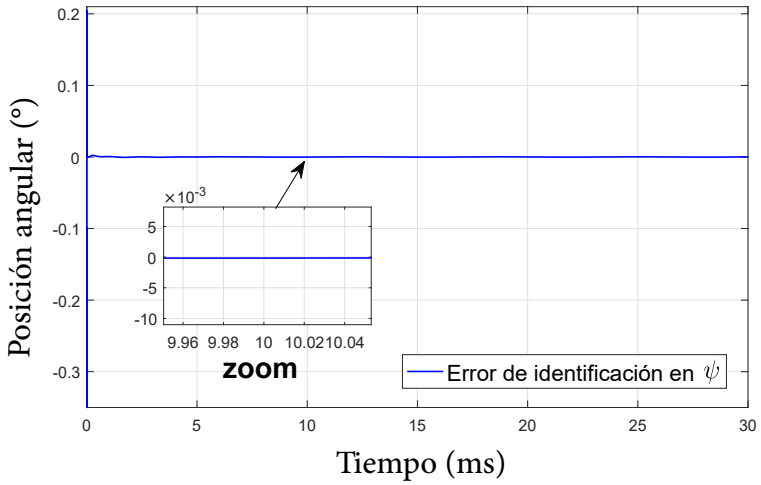


Figura 3.6: Error de la posición en ψ.

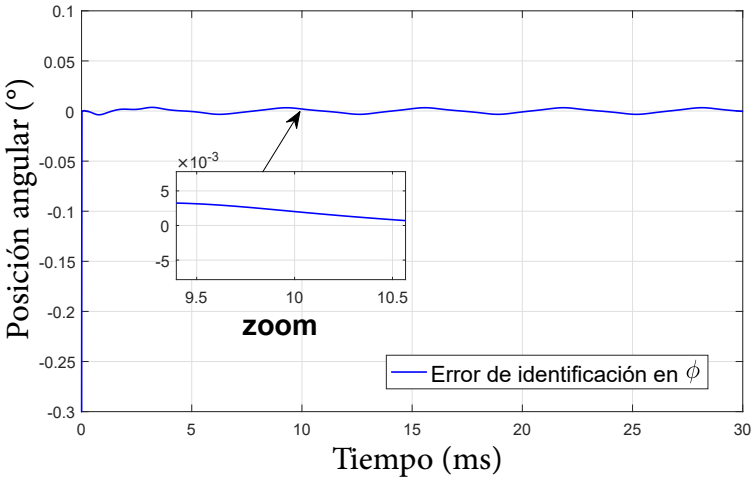


Figura 3.7: Error de la posición en ϕ.

En la Figura 3.6 se observa la diferencia que existe entre la señal de posición de la varilla que se obtiene de la planta y la señal que aproxima la RNA. Se puede decir que la identificación es muy buena ya que el error esta dentro de un rango ± 0.002. Al igual en la Figura 3.7 se observa la diferencia que existe entre la señal de la posición del carro que se obtiene de la planta y la señal que aproxima la RNA. Se puede decir que la identificación es muy buena ya que el error esta dentro de un rango ± 0.004.

En la Tabla 3.2 se realiza una comparación de las funciones de F respecto a $\hat{F}$, posteriormente en la Tabla 3.3 se realizá la comparación de las funciones de G

respecto a $\hat{G}$. Para ambos casos se seleccionaron diez muestras de los valores que se obtuvieron durante la identificación y control del sistema. En la Figura 3.8 y la Figura

$t(seg)$	F		$\hat{F}$	
	f_1	f_2	$\hat{f}_1$	$\hat{f}_2$
0.001	0.0206	0.1690	-0.2543	0.2575
0.1	0.0199	0.1652	0.2161	0.0323
0.5	0.0091	0.2246	0.2959	0.3314
1	0.0098	0.4403	0.3425	0.3169
5	-0.0288	-0.4733	0.2683	0.2403
10	-0.0081	-0.3064	0.2787	0.2526
15	0.0114	0.3465	0.3193	0.3037
20	0.0264	0.4586	0.3265	0.3105
25	-0.0008	-0.0703	0.2938	0.2725
30	-0.0313	-0.4807	0.2676	0.2395

Tabla 3.2: Resultado de la identificación de las funciones F del sistema (2.14).

3.9 se muestran los datos completos que se obtuvieron durante la identificación de las funciones de f_1 y f_2 respectivamente.

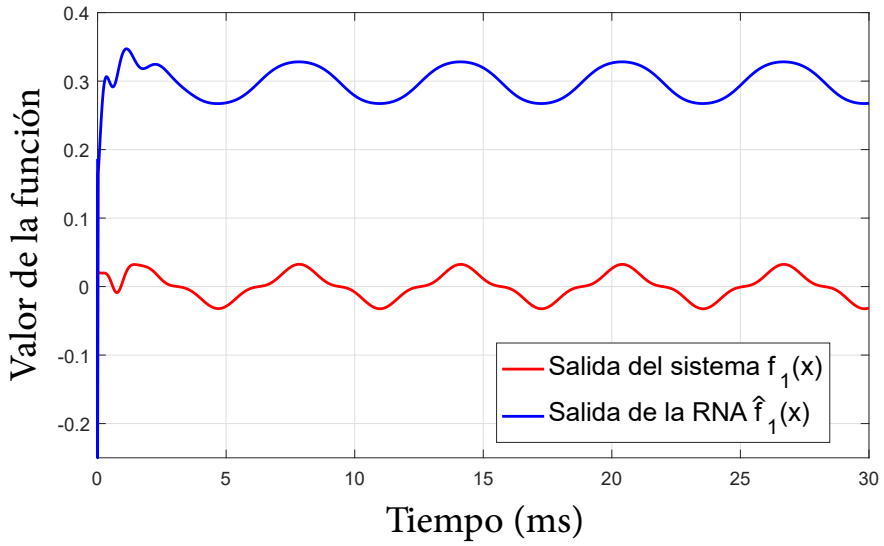


Figura 3.8: Indetificación de la función f_1.

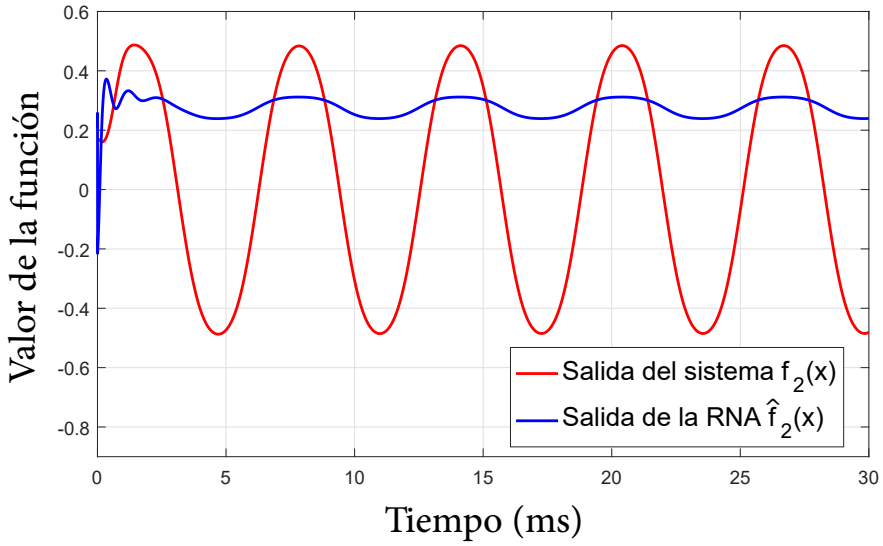


Figura 3.9: Indetificación de la función f_2.

$t(seg)$	G				$\hat{G}$			
	g_{11}	g_{12}	g_{21}	g_{22}	$\hat{g}_{11}$	$\hat{g}_{12}$	$\hat{g}_{21}$	$\hat{g}_{22}$
0.001	2.1421	0.1185	0.1185	0.9718	0.2430	0.0225	0.0225	0.1710
0.100	2.1422	0.1187	0.1187	0.9718	0.4020	0.1680	0.1680	1.1358
0.500	2.1410	0.1140	0.1140	0.9713	0.4020	0.1680	0.1680	1.1358
1	2.1340	0.0782	0.0782	0.9681	0.4020	0.1680	0.1680	1.1358
5	2.1325	0.0686	0.0686	0.9675	0.4020	0.1680	0.1680	1.1358
10	2.1389	0.1044	0.1044	0.9703	0.4020	0.1680	0.1680	1.1358
15	2.1376	0.0982	0.0982	0.9698	0.4020	0.1680	0.1680	1.1358
20	2.1332	0.0732	0.0732	0.9678	0.4020	0.1680	0.1680	1.1358
25	2.1433	0.1232	0.1232	0.9723	0.4020	0.1680	0.1680	1.1358
30	2.1322	0.0662	0.0662	0.9673	0.4020	0.1680	0.1680	1.1358

Tabla 3.3: Resultado de la identificación de las funciones G del sistema (2.14).

En las siguientes gráficas se muestran los datos completos que se obtuvieron durante la identificación de las funciones de G mediante la RNA tipo GMLP.

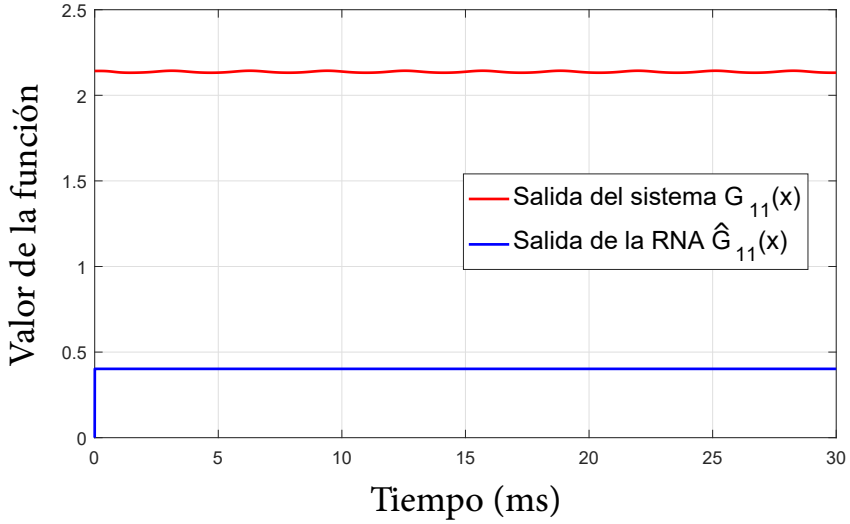


Figura 3.10: Identificación de la función g_{11}.

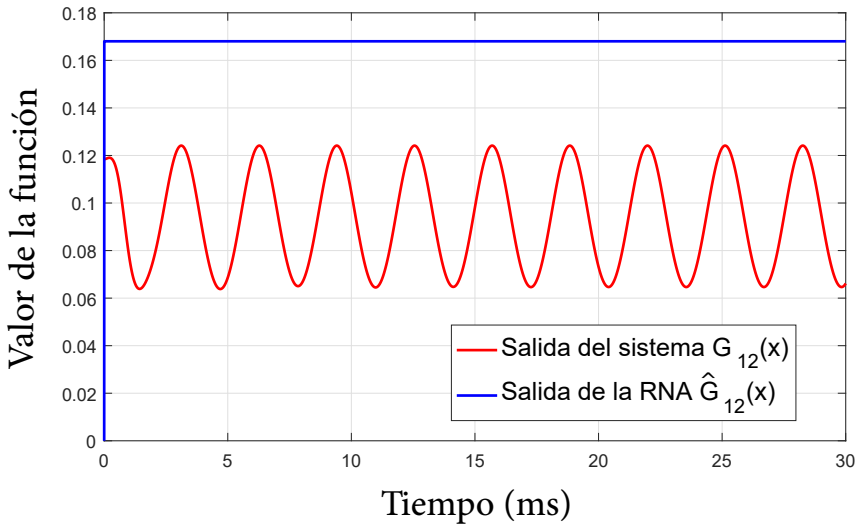


Figura 3.11: Identificación de la función g_{12}.

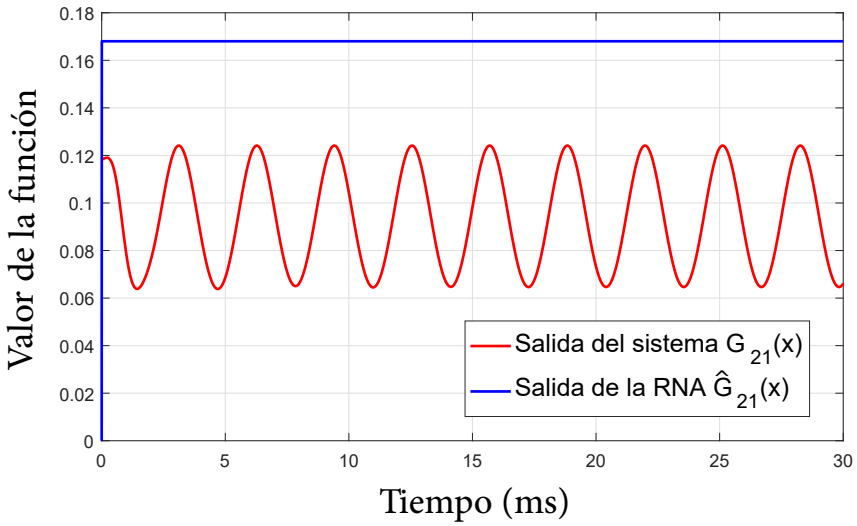


Figura 3.12: Identificación de la función g_{21}.

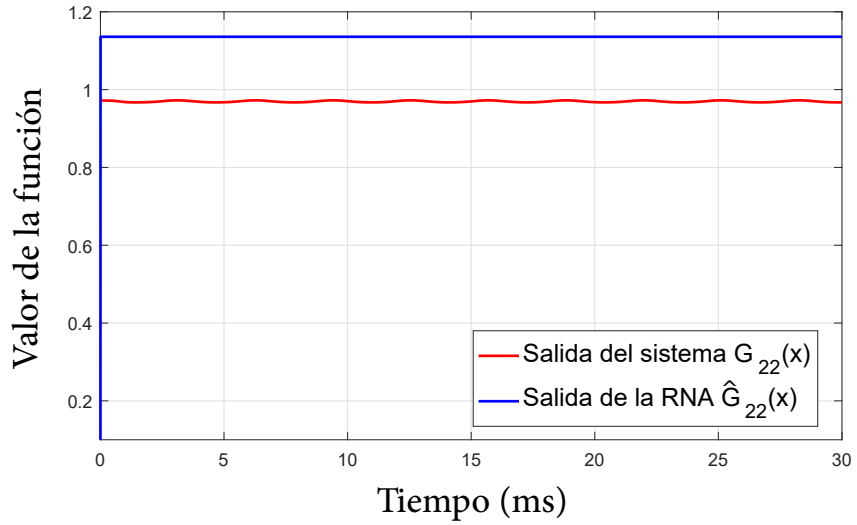


Figura 3.13: Identificación de la función g_{22}.

De acuerdo a los resultados de las tablas anteriores, se puede observar que las funciones F, G no son cercanas a las funciones $\hat{F}$, $\hat{G}$. Sin embargo, no es necesario que las funciones sean similares, ya que es posible obtener el mismo resultado con diferentes valores. Por tal motivo en la siguiente tabla se observan en los mismos instantes de tiempo, la salida del sistema y la RNA.

$t(seg)$	Y		Y_s	
	ψ	ϕ	$\hat{\psi}$	$\hat{\phi}$
0.001	0.3000	0.3000	-0.6208	0.5076
0.1	0.2428	0.2949	0.2475	0.2823
0.5	-0.0680	0.4050	-0.0674	0.4051
1	-0.0109	0.8854	-0.0109	0.8840
5	-0.0088	-0.9816	-0.0088	-0.9822
10	0.0024	-0.5689	0.0026	-0.5669
15	0.0032	0.6548	0.0033	0.6568
20	-0.0006	0.9367	-0.0007	0.9359
25	-0.0044	-0.1240	-0.0045	-0.1273
30	-0.0026	-1.0054	-0.0026	-1.0057

Tabla 3.4: Identificación de las variables del sistema (2.14)usando GMLP.

Implementación de la ley de control sobre el sistema péndulo sobre el carro utilizando la dinámica aproximada de la RNA

Ahora que la RNA identificó los estados del sistema real, es posible introducir los valores que representan a las funciones del sistema tal que

$$\hat{F}(x) \approx F(x), \qquad \hat{G}(x) \approx G(x).$$

Sustituyendo las aproximaciones de las funciones $\hat{F}(x)$ y $\hat{G}(x)$ en la ley de control (3.6), obtenemos la siguiente propuesta de control

$$u = \hat{G}^{-1}(x)\left[-\hat{F}(x) + \nu + Ks\right],$$

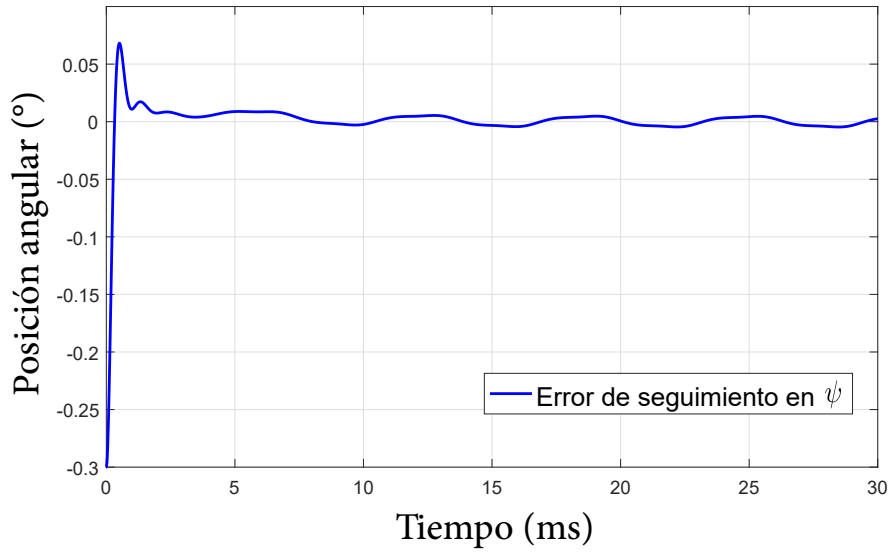
no obstante $\hat{G}(x)$ al ser una aproximación obtenida por la RNA se desconoce si posee inversa, por lo tanto se propone el uso de una inversa regularizada, dando así a la siguiente ley de control

$$u = \hat{G}^{T}(x)\left(I_e + \hat{G}(x)\hat{G}^{T}(x)\right)^{-1}\left[-\hat{F}(x) + \nu + Ks\right], \tag{3.11}$$

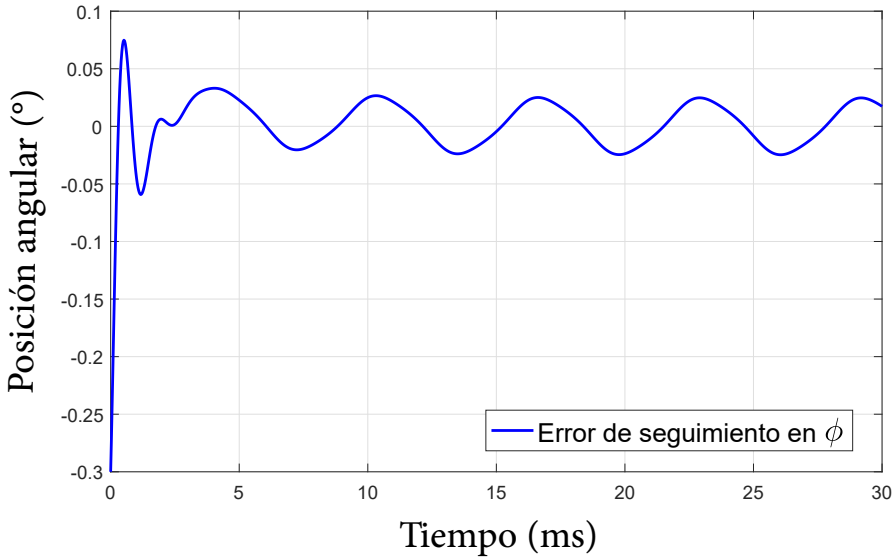
donde

$$I_e = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}, \qquad K = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Aplicando la ley de control (3.11) al sistema (3.1) y aproximando la dinámica en cada instante de tiempo además de considerar las condiciones iniciales $\psi = 0.3$ y $\phi = 0.3$. Se propone una trayectoria a seguir por la planta donde $Ref_\psi = 0$ y $Ref_\phi = sin(t)$, como resultado se obtuvieron las siguientes gráficas



(a) Error de seguimiento en ψ.



(b) Error de seguimiento en ϕ.

Figura 3.14: Error del seguimiento de una trayectoria deseada.

En la Figura 3.14 se muestra el error respecto a la salida de la planta y la referencia de ambas variables. Este error oscila entre ± 0.05 y es considerado aceptable lo cual implica que la ley de control esta actuando correctamente.

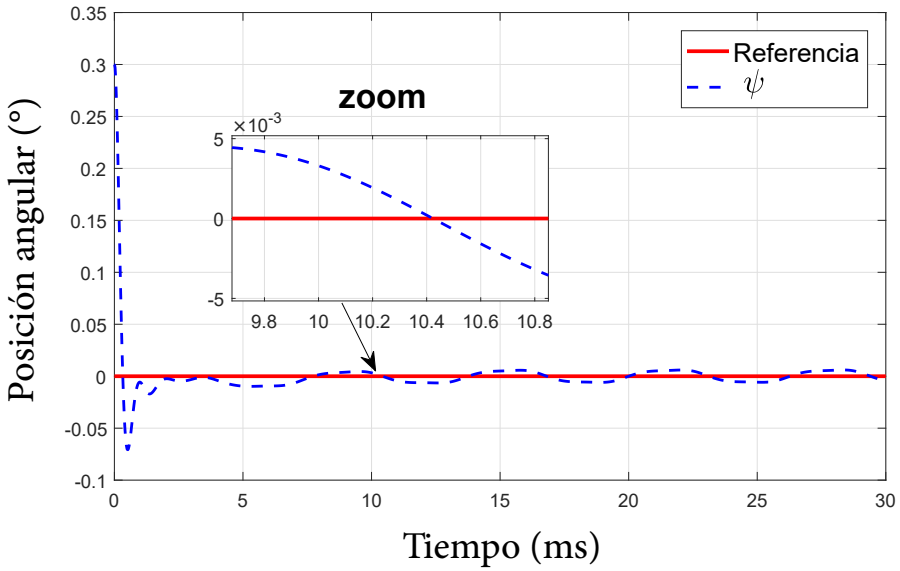


Figura 3.15: Seguimiento en ψ.

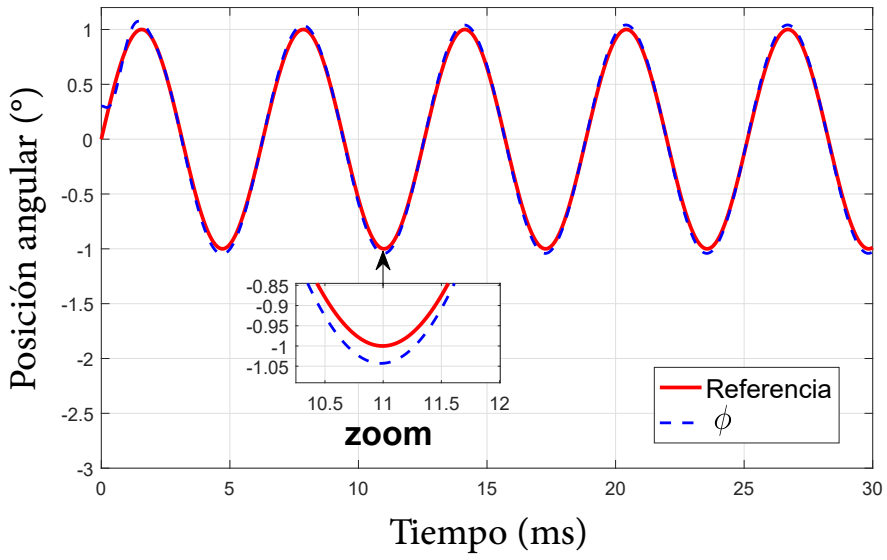


Figura 3.16: Seguimiento en ϕ.

En las figuras 3.15 y 3.16 se muestra el comportamiento de la salida del sistema ante una trayectoria deseada, en la cual se utilizó la ley de control (3.11) donde se contempla la dinámica aproximada por la RNA. El resultado muestra un seguimiento satisfactorio.

El error de seguimiento filtrado de la ley de control se muestra en las siguientes gráficas. Se observa que tanto S como $\dot{S}$ permanecen cercano a 0 en ambas variables, por lo que podemos considerar que si el error esta dentro de una vecindad en 0, entonces, el sistema está realizando el seguimiento de la trayectoria.

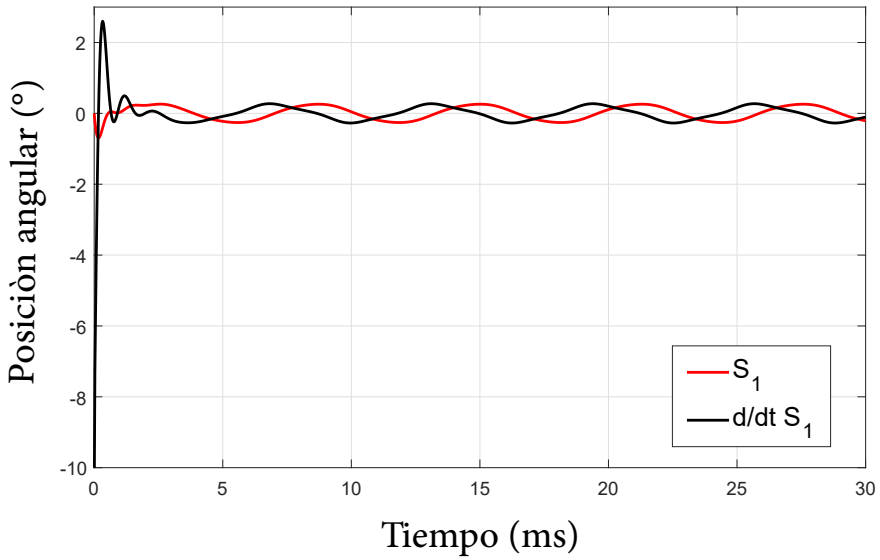


Figura 3.17: Error de seguimiento filtrado en ψ.

3.2. Segunda propuesta de control para sistemas no lineales

Se desarrolla la implementación de una ley de control adaptable utilizando una realimentación de estados los cuales son obtenidos mediante Redes Neuronales Recurrentes de Alto Orden (RHONN).

3.2.1. Aproximación de la dinámica del sistema Twin-Rotor mediante RNA RHONN

Al igual que el diseño del controlador (3.6) de la Sección anterior, se desarrolla de igual manera pero para este caso se utiliza el sistema (2.7.1) quien representa un

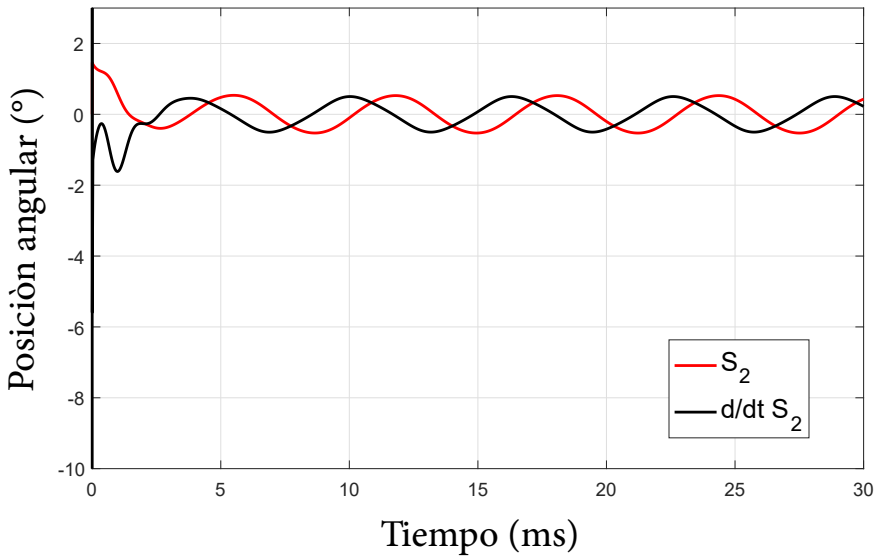


Figura 3.18: Error de seguimiento filtrado ϕ.

sistema no lineal denominado Twin Rotor MIMO Systems (TRMS).

$$\dot{x} = A(x) + B(x)u, \tag{3.12}$$

debido a las no linealidedes que presenta este sistema, se propone una linealizacion mediante Series de Taylor, de tal forma que el sistema (2.7.1) es representado como un sistema lineal de la forma:

$$\begin{aligned}
\dot{\chi}_1(t) &= 0.8333\chi_1(t) + 0.9166u_1(t) \\
\dot{\chi}_2(t) &= \chi_3(t) \\
\dot{\chi}_3(t) &= E_1\chi_1(t) + E_2\chi_2(t) - (B_1/I_1)\chi_3(t) + E_3\chi_6(t) \\
\dot{\chi}_4(t) &= -\chi_4(t) + 0.8u_2(t) \\
\dot{\chi}_5(t) &= \chi_6(t) \\
\dot{\chi}_6(t) &= E_4\chi_4(t) - (B_2/I_2)\chi_6(t) - (1/I_2)\chi_7(t) \\
\dot{\chi}_7(t) &= E_5\chi_7(t) - (1/I_2)\chi_7(t) + E_6u_1(t)
\end{aligned} \tag{3.13}$$

con $I_1 = 0.068$, $I_2 = 0.02$, $B_1 = 0.006$, $B_2 = 0.1$,$E_1 = (b_1 + a_1\chi_1)\left(\frac{1-K_{gy}\chi_6 cos(\chi_2)}{I_1}\right)$, $a_1 = 0.0135$, $b_1 = 0.0924$, $K_{gy} = 0.05$, $E_2 = \frac{-M_{fg}sin(\chi_2)}{I_2\chi_2}$, $M_{fg} = 0.32$, $E_3 = \frac{0.0163\chi_6 sin(2\chi_2)}{I_2}$,

$E_4 = \frac{1}{I_2}(b_2 + a_2\chi_4)$, $a_2 = 0.02$, $b_2 = 0.09$, $E_6 = 0.9166A_{cte}(0.5b_1 + a_1\chi_1)$, $A_{cte} = -0.7$, $E_5 = 0.5B_{cte}(b_1 + a_1\chi_1) - A_{cte}(0.5b_1 + a_1\chi_1)$ y $B_{cte} = -0.2$.

El sistema (3.13)es posible llevarlo a su forma de representacion Espacio-Estado tal que sea posible interpretarse como (2.11),donde, se asume que es posible sustituir ecuación del sistema por la ecuación de la RNA tipo RHONN.

Usando técnicas de control lineal, se desarrolla una ley de control mediante una retro de estado sintonizada por una asignación de polos, donde:

$$u = Kx, \tag{3.14}$$

en la que el vector de ganancias K es una matriz que contiene las ganancias obtenidas mediante LMI que optimizan al sistema.

3.2.2. Identificación del sistema usando RNA tipo RHONN

Para la identificación del sistema TRMS, se utiliza el modelo linealizado descrito en la ecuación (3.13) debido a la estructura requerida por la RHONN la cual esta descrita en la ecuación (2.11).

El resultado de la identificación del sistema lineal de 7 variables de estado mediante una RNA tipo RHONN de la forma (2.11), se muestra en las siguientes gráficas, donde sólo se eligieron los estados X_2 (ángulo de elevación) y X_5 (ángulo de traslación), debido a que son las variables del sistema medibles físicamente. Como se

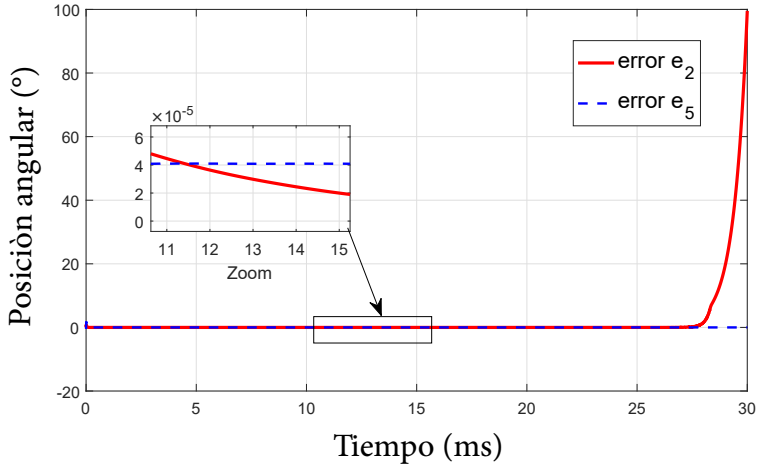


Figura 3.19: Error de identificación de las variables x_2 y x_5 mediante una RHONN

puedo observar en la Figura 3.19 se aprecia el error de identificación de las funciones correspondientes a las variables X_2 y X_5, cuyo error esta dado por $e_2 = \chi_2 - x_2$ y $e_5 = \chi_5 - x_5$ respectivamente. El resultado que se muestra es factible para el trabajo propuesto ya que la vecindad en que oscila el error es de ± 0.006 grados. Cabe mencionar que durante el proceso se utilizó una señal $u = [0.001cos(60t), 0.05cos(60t)]$ la cual no garantiza ninguna estabilidad a lo largo de $t = [0 : 30]$ segundos. Esta señal es utilizada para fines de simulación de tal forma que modifique la dinámica del sistema. Se asume que $u = Kx$ por tal motivo el sistema no presenta estabilidad a lo largo del tiempo durante la identificación, sin embargo, podemos decir que

$$Kx = [u_1, u_2]\,. \tag{3.15}$$

En la Figura 3.20 y Figura 3.21 se muestra el comportamiento que obtuvo la RHONN

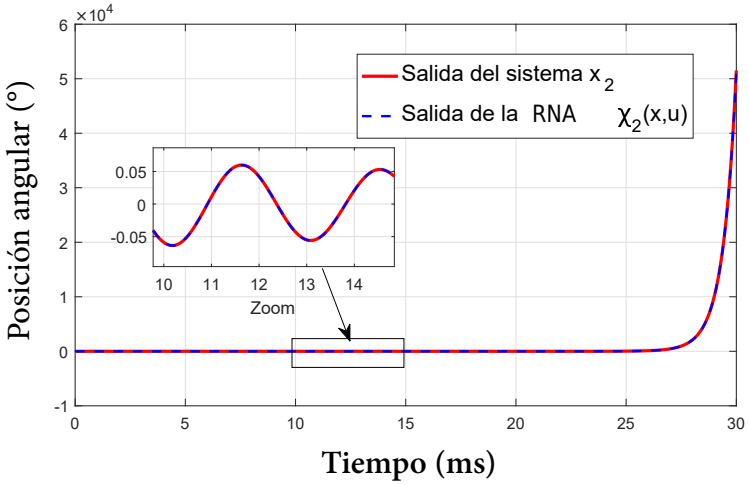


Figura 3.20: Identificación de x_2 utilizando una RHONN para el estado χ_2.

al identificar las variables x_2 y x_5 respectivamente, ante una entrada u,descrita en (3.15), y la salida del sistema a identificar. Como se puede observar la identificación de la variable x_2 se encuentra dentro de una vecindad de ± 0.06, y respecto a x_5 ésta se encuentra dentro de la vecindad ± 0.02, para ambas identificaciones se puede considerar como factibles para implementar una ley de control que contemple la dinámica del sistema.

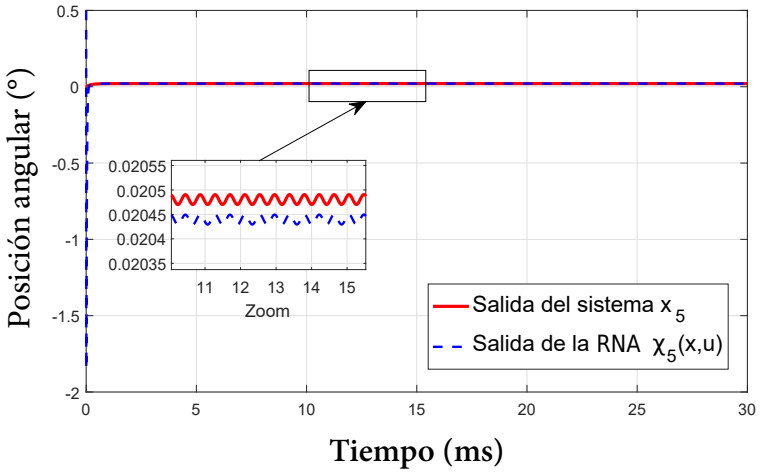


Figura 3.21: Identificación de x_5 utilizando una RHONN para el estado χ_5.

3.2.3. Implementación de resultados utilizando RNA tipo RHONN

Con la identificación del sistema TRMS mediante la RHONN, podemos decir que la ecuación $\dot{x} = Ax + Bu$ es aproximada a

$$\dot{\chi} = \hat{A}x + +Wz(x(t), u(t)) \tag{3.16}$$

donde $\dot{\chi}$ es el vector de los estados aproximados por la RHONN, $\hat{A} = diag\{0.1, 0.2, 0.1, 0.2, 0.1, 0.2, 0.1\}$ es un vector de parámetros cuya suma debe ser 1 y son definidos por el usuario, $z \in \mathbb{R}^{n+m}$ es el vector de la funciones definidas en la ecuación (2.10), $W \in \mathbb{R}^{n \times n+m}$ es la matriz de pesos obtenidos por la RHONN durante la identificación del sistema, cuyos valores fueron determinados previamente, tal que

$$W = \begin{bmatrix} 0.5380 & 0.2955 & 0.1687 & 0.1393 & 0.3968 & -0.5545 & 0.0395 & -0.6240 & -0.3993 \\ 0.0207 & 0.0731 & -0.0754 & 0.2484 & -1.0171 & 0.3788 & 0.2020 & 0.3433 & -0.1738 \\ -0.5820 & 0.2086 & 0.6845 & -0.2324 & -0.5294 & -0.2321 & 0.3871 & 0.2078 & 0.0878 \\ -0.2105 & -0.0664 & -0.1449 & 0.5007 & -0.3110 & 0.6700 & -0.5393 & 0.4235 & -0.3221 \\ 0.5265 & -0.2654 & 0.4964 & 0.3377 & 0.3346 & -0.5836 & -0.2363 & -0.4418 & -0.1682 \\ -0.5773 & 0.3265 & 0.3329 & 0.6305 & -0.1172 & -0.4875 & 0.0948 & 0.2233 & -0.4260 \\ 0.4984 & 0.4481 & 0.0588 & 0.1593 & -0.1350 & -0.5209 & 0.1743 & 0.1099 & -0.7928 \end{bmatrix},$$

$$z = \begin{bmatrix} S(x_1) & S(x_2) & S(x_3) & S(x_4) & S(x_5) & S(x_6) & S(x_7) & S(u_1) & S(u_2) \end{bmatrix},$$

utilizando el sistema (3.16) se obtiene la matriz de parámetros $\hat{K}$ mediante el uso de LMI y la herramienta de Matlab, cuya desigualdad matricial esta definida por $AP + BY + PA^T + Y^T B^T + 2P\alpha > 0$ y $P > 0$. Recordando que el nuevo sistema es obtenido mediante la identificación de la RHONN, a partir de ello, se obtuvieron los

siguientes valores para las ganancias del control

$$\hat{K} = \begin{bmatrix} 7.2990 & 36.3304 & 162.8073 & -127.0278 & -41.1055 & -117.8086 & 145.4670 \\ 8.9963 & 35.0407 & 108.0715 & -56.7109 & -23.1062 & -67.4659 & 72.3724 \end{bmatrix},$$

al sustituir $\hat{K}$ en la ley de control propuesta en (3.15), tenemos que

$$u = \hat{K}x, \tag{3.17}$$

esta ley de control se generó a partir de los valores de las funciones identificadas por una RHONN del sistema TRMS. La señal u es sustituida en la ecuación (3.13) tal que estabilice al sistema TRMS. Se realizó la simulación de esta ley de control en un lapso de $30s$, respecto a un punto de referencia en 0 para todas sus variables disponibles. Los resultados obtenidos se muestran a continuación.

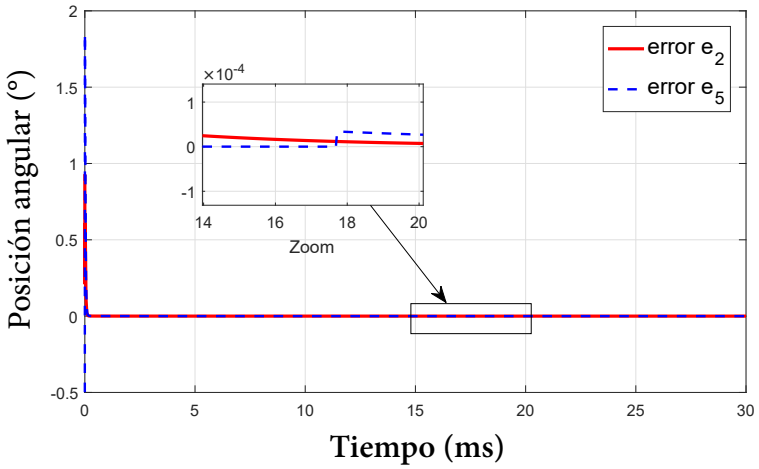


Figura 3.22: Errores de estabilización de la x_2 y x_5 mediante una RHONN.

En la Figura 3.22 se aprecia el error que existe entre las variables x_2, x_5 y χ_2, χ_5 respectivamente. Sabiendo que la ley de control fue construida a partir de un sistema cuya identificación fue realizada por una RHONN, el resultado obtenido muestra que la ley de control es factible. La vecindad de ambos errores $e_2 = \chi_2 - 0$ y $e_5 = \chi_5 - 0$) se encuentran entre ± 0.0001 considerándose valores adecuados a la referencia propuesta.

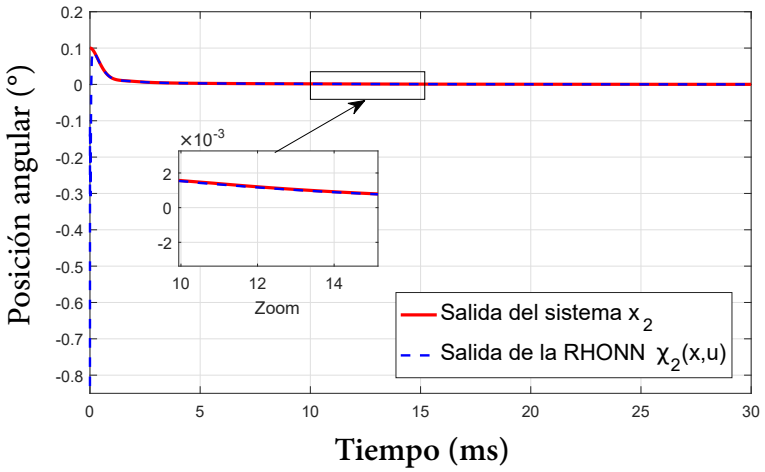


Figura 3.23: Estabilización de la variable x_2 usando la dinámica de la RHONN.

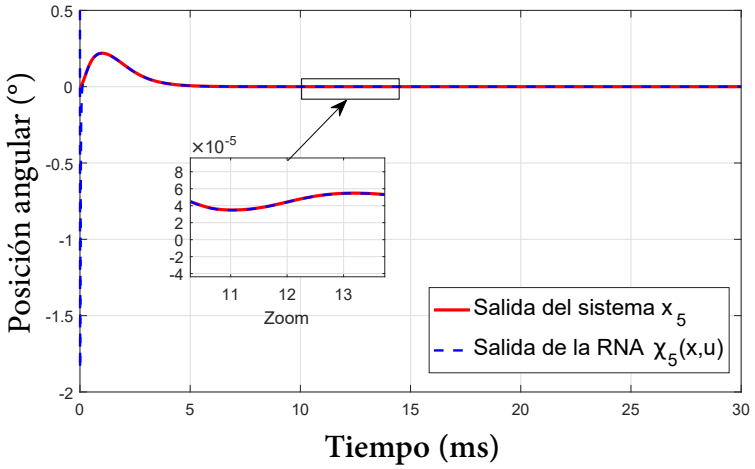


Figura 3.24: Estabilización de la variable x_5 usando la dinámica de la RHONN.

En la Figura 3.23 y la Figura 3.24 se muestra la estabilización de la variable x_2 y la variable x_5 respecto a la referencia 0 donde, la señal se encuntra dentro de una vecindad de ± 0.006 grados. Además se muestra como la RHONN nuevamente identifica al sistema.

En la Figura 3.25 se muestran todas las variables del sistema TRMS, las cuales fueron controladas mediante la señal de control de la ecuación (3.17) donde, se observa que el sistema se estabiliza a partir del los $5s$, y así lo hace a lo largo del tiempo.

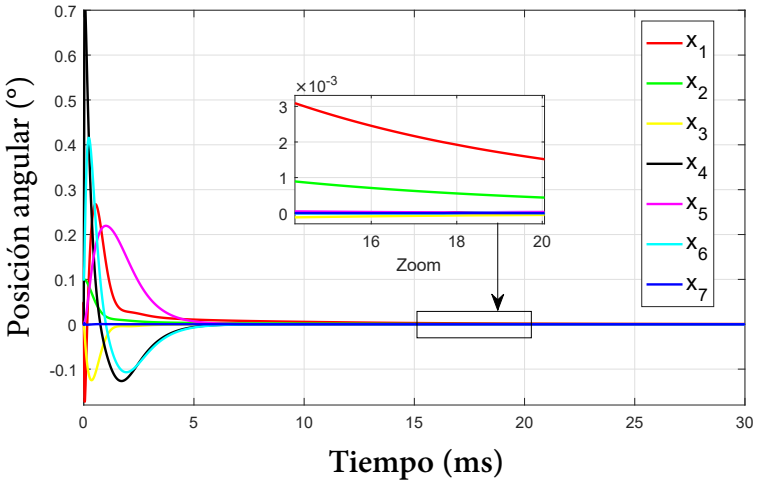


Figura 3.25: Estabilización del sistema mediante la dinámica de la RHONN.

3.3. Discusión

Uno de los principales objetivos de esta investigación es la identificación de un sistema dinámico mediante el uso de RNA. Durante el desarrollo se propusieron múltiples tipos de RNA, de las cuales se eligieron dos, debido a las características que se necesitaban para atacar el problema de identificación. La primer RNA es del tipo GMLP y la segunda es del tipo RHONN. Ambas propuestas se presentaron en simulación con acción de identificar de la mejor manera un sistema dinámico
En la Figura 3.26 y la Figura 3.28 se evaluaron ambas RNA con el objetivo de visualizar en cual se obtiene un mejor rendimiento respecto una de otra. Se utilizó el mismo sistema de ecuaciones que representan la dinámica, para identificar en ambas RNA, el cual esta descrito por el sistema de ecuación (3.13), se perturbo al sistema mediante una serie de entradas u: para $t \in [0, 11]s$, $u = (K_pX)sin(0.0001t)$, para $t \in (11, 20]s$, $u = (K_pX)0.7$ y para $t \in (20, 30]$, $u = (K_pX)cos(0.0001t)$, donde X son los estados del sistema y K_p esta definida por la siguiente matriz

$$K_p = \begin{bmatrix} 6.1814 & -17.3559 & 6.9634 & 0.7353 & 0.8039 & -0.0684 & -9.7639 \\ -1.9721 & 6.9414 & 0.1320 & 0.7781 & 2.3904 & 1.5385 & -18.7058 \end{bmatrix}. \quad (3.18)$$

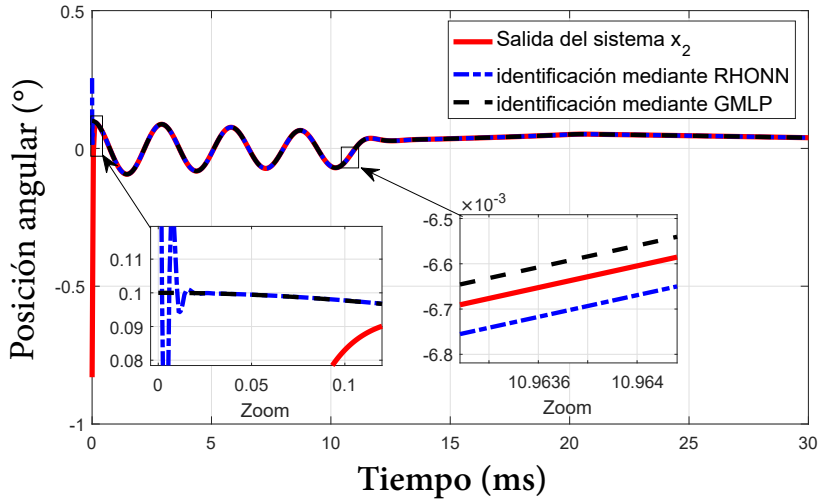


Figura 3.26: Identificación de la variable x_2 mediante una RHONN y una GMLP.

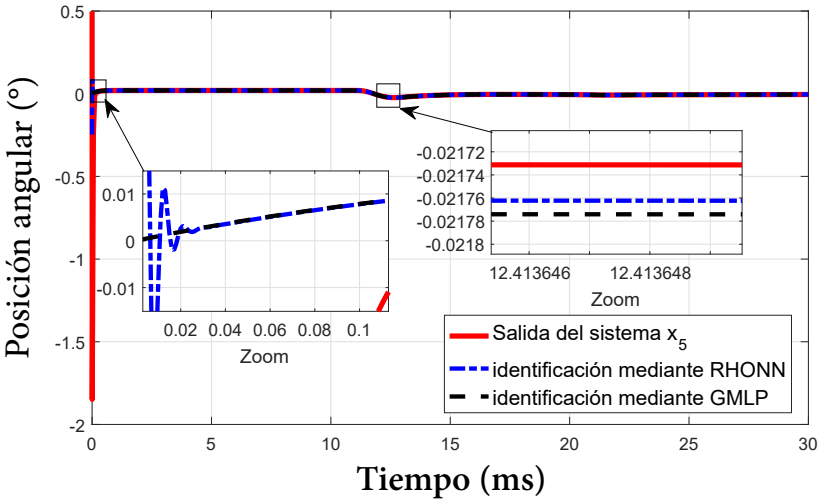


Figura 3.27: Identificación de la variable x_5 mediante una RHONN y una GMLP.

Como resultado se obtuvo que ambas RNA identifican adecuadamente un sistema, en el zoom de las gráficas se aprecia que las funciones se encuentran muy cerca de la referencia deseada, sin embargo, se hace un acercamiento al comienzo de la identificación en el cual se observa que la RHONN tarda un poco para adaptarse a la función deseada en comparación a la RNA. Por lo que supongamos que se intenta identificar un sistema no lineal de la forma (3.3) con dinámica rápida y perturbaciones externas,

la RHONN tendría dificultades al adaptarse rápidamente.
Por otra parte se obtuvo el error cuadrático medio (ECM) respecto a la variable de salida del sistema TRMS y la salida de la RNA GMLP y RHONN, cuyos valores se reflejan en la siguiente tabla

RNA/Variables	ECM en x_2	ECM en x_5
GMLP	8.4462e-07	6.1181e-06
RHONN	5.0301e-04	0.0021

Tabla 3.5: Error cuadrático medio del sistema TRMS.

no obstante se propone un segundo caso el cual la función a identificar se caracteriza por ser un poco impredecible su comportamiento. Cabe mencionar que para poder identificar con dichas RNA es necesario que la función propuesta sea continua y diferenciable, de lo contrario el sistema puede indeterminarse. La señal propuesta esta definida por $sin(xt) + 5exp(\frac{t}{10})sin(t)$ considerando a x como una variable. En la siguiente imagen se muestra el resultado obtenido

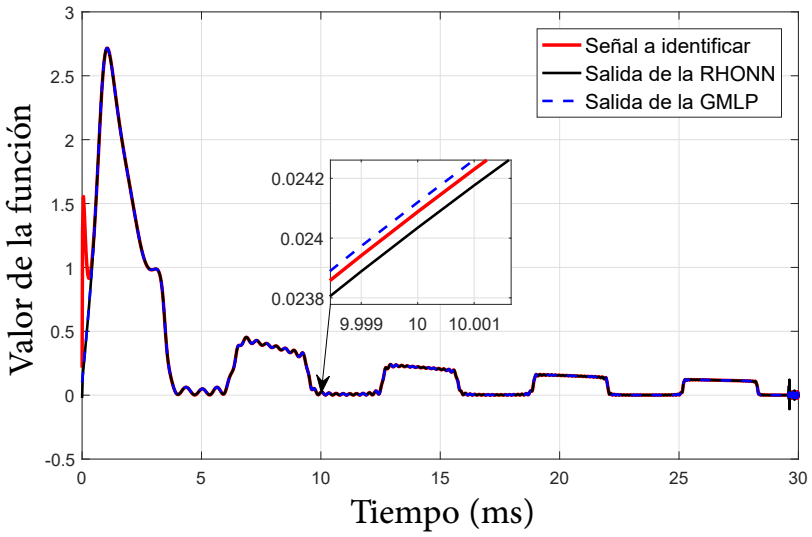


Figura 3.28: Identificación de una funcion mediante una RHONN y una GMLP.

RNA/Variables	ECM
GMLP	2.1313e-05
RHONN	0.0074

Tabla 3.6: Error cuadrático medio de una Función no lineal.

Se determinó que el error cuadrático medio es mucho menor en la RNA GMLP para ambos casos. Razón por la que se decidió utilizar una RNA tipo este tipo.

3.4. Conclusión

En la obtención de los resultados teóricos obtenidos, se concluyen satisfactorios. El análisis que se realizo para determinar que la ley de control llevaría la variable del sistema por una trayectoria deseada, se ha comprobado. Mediante el uso de RNA se logro aproximar la dinámica de un SNL, para elló se realizaron varias pruebas de tal forma que se obtuvieran los mejores resultados, esto fue gracias a las múltiples simulaciones realizadas, obteniendo los parámetros que dieron como resultado un desempeño adecuado durante la identificación del sistema. Consecuencia de estas pruebas se dieron lugar a dos propuestas de sistemas y leyes de control que utilizan la dinámica aproximada por una RNA de un sistema para determinar dicha ley de control.

Sea la primer propuesta una RNA tipo GMLP, en la cual se estableció la identificación del sistema *Péndulo−Invertido*, así como la utilización de una ley de control basada en un error de seguimiento filtrado. La identificación obtenida por este tipo de RNA demostró ser muy parecida a la real, por tal motivo fue posible implementarla de acuerdo a los criterios matemáticos establecidos, obteniendo como resultado el seguimiento de una trayectoria deseada utilizando la aproximación de la dinámica. Se puede observar en la Figura 3.15 y la Figura 3.16.

Sea la segunda propuesta una RHONN, en la cual se estableció la identificación del sistema *TRMS*, así como la implementación de una ley de control basada en una retro de estado, cuyas ganancias son obtenidas mediante una LMI. La aproximación de la dinámica obtenida por la RHONN demostró, al igual que en la primer propuesta, una identificación factible para ser utilizada en la ley de control. Como resultado se obtuvo que, la ley de control con ganancias obtenidas mediante la sustitución del sistema por la aproximación de la RHONN y resolviendo la LMI definida, estabiliza al sistema. Se puede observar en la Figura 3.25.

El uso de las RNA en conjunto con la ley de control basada en el error de seguimiento filtrado, proporcionaron un buen seguimiento de trayectorias. Como resultado se obtuvo que si la dinámica del sistema cambia a causa de una perturbación externa o debido al entorno que lo rodea, las RNA son capaces de identificar la nueva dinámica y modificar los parámetros del sistema y para corregir el error de seguimiento

filtrado, haciendo que el controlador se adapte.

La identificación de las funciones o en este caso del sistema mediante una RNA tipo GMLP es obtenida en muy poco tiempo, además presenta un mejor seguimiento de la variable deseada en muy poco tiempo en consideración de una RHONN.

Capítulo 4

Conclusiones y Trabajos Futuros

4.1. Trabajos futuros

Los alcances y limites de este trabajo se han concluido, sin embargo es posible trabajar con nuevas técnicas y metodologías, que pudiesen mejorar algunos aspectos de este trabajo. El diseño de la ley de control permite adaptarse a cambios, los cuales se pretende investigar para determinar qué tan factibles son estas metodologías para el control e identificación de sistemas no lineales. Como consecuente se determinan los siguientes puntos a investigar en un futuro.

- El uso de RNA para la identificación de un sistema no lineal se ha determinado por funciones de activación tipo sigmoidales. Este tipo de funciones son rígidas y hallar el valor deseado puede llevar un tiempo considerable. Por tal motivo se pretende usar funciones de activación tipo difusas para encontrar el valor de dichas funciones no lineales, se pretende que este tipo de funciones pueda reducir la incertidumbre durante el proceso de obtención del valor deseado tal que pueda mejorar el desempeño de la RNA.

- Se sabe que el uso de las RNA generan un costo alto en procesos computacionales, por lo que la implementación de las RNA durante procesos en tiempo real conlleva el uso de hardware especializado. Por tal motivo se pretende implementar la información obtenida en este proyecto de forma física, para determinar el comportamiento de las RNA implementadas en un sistema real.

4.2. Estancia

Durante el periodo escolar del posgrado, se realizó una estancia de investigación en el Instituto Tecnológico de Sonora (ITSON) el 8 de marzo al 13 de abril de 2017,

bajo la tutela del Dr. Miguel A. Bernal miembro del Departamento de Ingeniería en Electricidad y Electrónica del ITSON. Como producto de dicha estancia se obtuvo la publicación de un artículo en congreso internacional:

C. Armenta, M. Bernal, J. Hernández y R. Villafuete. *Identification-Based Linear Control of a Twin Rotor MIMO System via Dynamical Neural Networks*, 14*th* International Conference on Electronical Engineering, Computing Science and Automatic Control (CCE). Octubre 2017, Ciudad de México.

Bibliografía

[1] Howard B Demuth, Mark H Beale, Orlando De Jess, and Martin T Hagan. *Neural network design*. Martin Hagan, 2014.

[2] Yoan D Landau. Adaptive control: The model reference approach. *IEEE Transactions on Systems, Man, and Cybernetics*, (1):169–170, 1984.

[3] Benjamin C Kuo. *Sistemas de control automático*. Pearson Educación, 1996.

[4] Damián Jorge Matich. Redes neuronales: Conceptos básicos y aplicaciones. *Cátedra de Informática Aplicada a la Ingeniería de Procesos–Orientación I*, 2001.

[5] Francisco Rodríguez Rubio and Manuel Jesús López Sánchez. *Control adaptativo y robusto*, volume 9. Universidad de Sevilla, 1996.

[6] Toxqui Toxqui Rigoberto. Redes neuronales difusas dinámicas para identificación y control adaptable., 2003.

[7] Warren McCulloch & Walter Pitts. Un Cálculo Lógico de la Inminente Idea de la Actividad Nerviosa. 1943.

[8] Félix de Moya Anegón, Víctor Herrero Solana, and Vicente Guerrero Bote. La aplicación de redes neuronales artificiales (rna): a la recuperación de la información. *Bibliodoc: anuari de biblioteconomia, documentació i informació*, pages 147–164, 1998.

[9] Duckgee Park, Moon-Soo Park, and Suk-Kyo Hong. A study on the 3-dof attitude control of free-flying vehicle. 2:1260–1265, 2001.

[10] Witt Jonas , Boonto Sudchai & Werner Herbert. Approximate Model Predictive Control of a 3-DOF Helicopter. 2007.

[11] Alireza Jalalian Khakshour and Mojtaba Ahmadieh Khanesar. Model reference fractional order control using type-2 fuzzy neural networks structure: Implementation on a 2-dof helicopter. *Neurocomputing*, 193:268–279, 2016.

[12] Boris Andrievsky, Dimitri Peaucelle, and Alexander L Fradkov. Adaptive control of 3dof motion for laas helicopter benchmark: design and experiments. In *American Control Conference, 2007. ACC'07*, pages 3312–3317. IEEE, 2007.

[13] Zheng Fang, Weinan Gao, and Lei Zhang. Robust adaptive integral backstepping control of a 3-dof helicopter. *International Journal of Advanced Robotic Systems*, 9(3):79, 2012.

[14] Hao Liu, Geng Lu, and Yisheng Zhong. Robust lqr attitude control of a 3-dof laboratory helicopter for aggressive maneuvers. *IEEE Transactions on Industrial Electronics*, 60(10):4627–4636, 2013.

[15] Salim Labiod, Mohamed Seghir Boucherit, and Thierry Marie Guerra. Adaptive fuzzy control of a class of mimo nonlinear systems. *Fuzzy sets and systems*, 151(1):59–77, 2005.

[16] Chang-Woo Park and Young-Wan Cho. Ts model based indirect adaptive fuzzy control using online parameter estimation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 34(6):2293–2302, 2004.

[17] T. Shaocheng , C. Bin & W. Yongfu. Fuzzy adaptive output feedback control for mimo nonlinear systems. Fuzzy Sets Systems. 2005.

[18] T. Shaocheng, T. Wang & J.T. Tang. Tuning of Proportional Retarded Controllers: Theory and Experiments. 2000.

[19] Katsuhiko Ogata and Guillermo Lopez Portillo Sanchez. *Dinámica de sistemas*. Prentice-Hall Hispanoamericana, 1987.

[20] Hassan K Khalil. Noninear systems. *Prentice-Hall, New Jersey*, 2(5):5–1, 1996.

[21] Pedro Ponce Cruz and Alejandro Herrera. *Inteligencia artificial con aplicaciones a la ingeniería*. Marcombo, 2011.

[22] Elias B Kosmatopoulos, Marios M Polycarpou, Manolis A Christodoulou, and Petros A Ioannou. High-order neural network structures for identification of dynamical systems. *IEEE transactions on Neural Networks*, 6(2):422–431, 1995.

[23] Elias B Kosmatopoulos, Manolis A Christodoulou, and Petros A Ioannou. Dynamical neural networks that ensure exponential identification error convergence. *Neural Networks*, 10(2):299–314, 1997.

[24] Steven T. Carris. *Introduction to simulink with engineering applications*. Tercera edition, 2011.

[25] Fernando Reyes. *Robótica-Control de robots manipuladores*. Alfaomega Grupo Editor, 2011.

[26] Gabriel Romero-Rodríguez, Pablo Sánchez-Sánchez, Fernando Reyes-Cortés, Antonio Michua-Camarillo, Benjamín Calderón-Flores, Jaime Cid-Monjaraz, Juan Carlos Torres-Monsivais, and Gabriela Morales-Timal. Modelado, control y simulación de un sistema péndulo invertido sobre base móvil.

[27] Feedback instruments Ltd, East Sussex, U.K. *TRMS 33-949S User Manual: Twin Rotor MIMO System Control Experiments*, 1998.

[28] Alberto Isidori. *Nonlinear control systems*. Springer Science & Business Media, 2013.

[29] Villafuerte R., Domínguez O. A. , González O. y Hoyos M. A. A Simple Implementation of an Intelligent Adaptive Control Systems for Human-Robot Interaction. 2016.

Apéndice A

Identificación de funciones no lineales usando RNA

Perceptrón simple con Aprendizaje Widrow−Hoff

Llevando la metodología de la Regla Delta Generalizada a un entorno de simulación, podemos analizar y estudiar la rápidez con la que desempeña una tarea asignada, así como determinar que tan precisa es su aproximación. Para ello se utiliza la herramienta de Matlab, donde se realizan los siguientes pasos del algoritmo:

- 1.- Se determina el vector de entrada y salida
- 2.- Se realiza el producto dendrítico de las entradas por los pesos.
- 3.- Se realiza la activación de la Neurona por medio de una función de transferencia.
- 4.- Se obtiene el error de la salida deseada respecto a la salida obtenida con la neurona.
- 5.- Con el error se calculan los nuevos pesos multiplicándolos por la tasa de aprendizaje.
- 6.- Se determina el error cuadrático, mientras sea mayor a un ε se regresará al paso 2.

Se implemento el algoritmo en código script de Matlab en la función *Perceptron(Xs, Y)* que se muestra en el apéndice B de este documento. Se llevo acabo la implementación de este código con las siguientes entradas y salidas definidas como $Xs = cos(t) + sin(t)$ y $Y = sin(t)$ a lo largo de un tiempo de $10s$ con pasos de 0.001

.

En la Figura A.1 se muestran algunos resultados donde se analiza el comportamiento del error durante el seguimiento de la RNA respecto a la entrada Y.

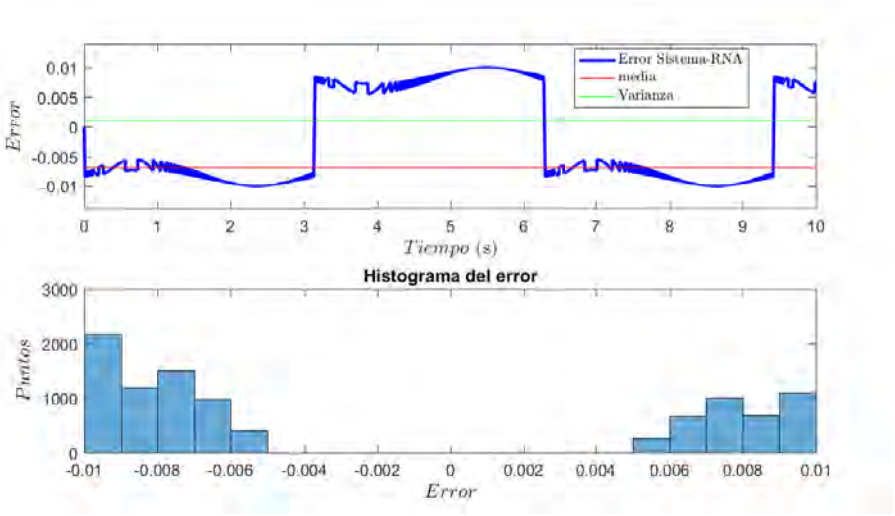


Figura A.1: Análisis del error con la Regle Delta Generalizada

Además de obtener el comportamiento de la RNA respecto al error, es importante contemplar el tiempo en que tarda en realizar las tareas, para ello se utiliza un cronometro que determina el tiempo en que realiza la aproximación, para la simulación anterior se obtuvo un tiempo de $17s$ en tiempo real, y en simulación represento a $10s$. Por lo que la respuesta obtenida respecto a la entrada y salida no es aceptable para este trabajo.

Sin embargo no es que la RNA no se desempeñe correctamente, sino que hay muy poca información que está adquiriendo para determinar el objetivo, por tal motivo se proponen RNA tipo MISO y MIMO.

Perceptrón MISO con Aprendizaje Widrow−Hoff

RNA con múltiples entradas una salida (MISO): Utilizando el mismo algoritmo de la regla de aprendizaje de la simulación anterior se anexan 6 entradas y se conserva una salida. La implementación del código en Matlab así como los parámetros de salida son los mismo, sin embargo para esta prueba ahora se determinan las siguientes entradas:

$$X1 = cos(t) + sin(t) \qquad X4 = -(cos(t) + sin(t))$$
$$X2 = cos(t) \qquad X5 = -sin(t)$$
$$X3 = sin(t) \qquad X6 = -cos(t)$$

Se eligieron estas funciones aleatoriamente, con un tiempo de $10s$ con pasos de 0.001, para el entrenamiento se utilizó W−H con una RNA de R entradas una salida, este algoritmo esta definido como la función *PerceptronMiSO($\bar{X}s, Y$)* en el apéndice B de este documento, así como el código de su implementación.

Al llevar la propuesta del la RNA a Matlab se obtuvierón los siguientes resultados donde en la Figura A.2 se muestra el comportamiento del error respecto a la salida Y.

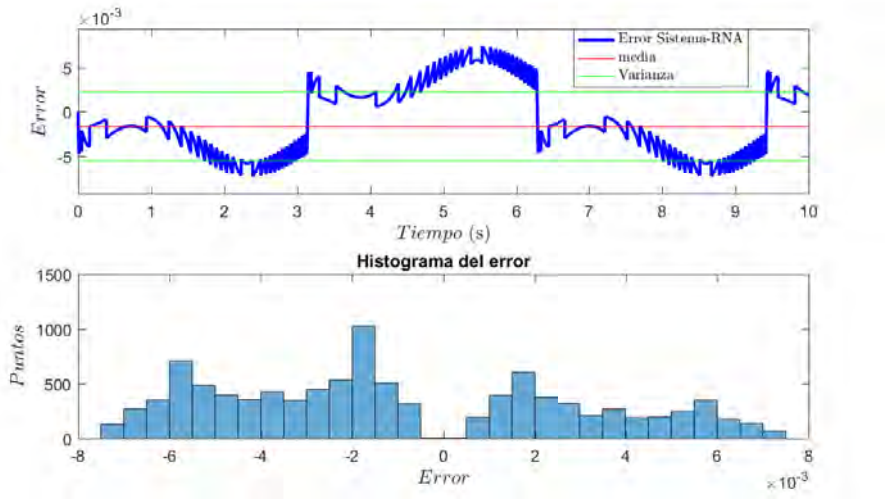


Figura A.2: Análisis del error con multiples entradas

Esta propuesta mejoro mucho el desempeño en el tiempo. Para la simulación anterior con 6 entradas y una salida se obtuvo un tiempo de $1.5s$ en tiempo real, y en simulación represento a $10s$. Por lo que la respuesta obtenida respecto a la entrada y salida es sumamente aceptable.

Pero como se puede observar en la Figura A.2, el objetivo se alejo un poco en comparación al primer caso de una entrada una salida, en la Figura A.1 el error permanece dentro de un rango de $[-0.01, 0.01]$ y en cuanto a la opción de introducir 6 entradas a la RNA el error se encuentra en un rango de $[-0.08, 0.08]$, lo que nos indica que no es una buena aproximación de la salida.

Perceptrón con Filtro adaptable

La segunda opción que se propone es utilizar n retardos de una sola señal para formar una entrada de R valores, este método es conocido como **filtro adaptable**.

Ahora se introduce al código una sentencia que genere un vector de n retardos de la señal y los envíen como entrada a la RNA. Al igual que los casos anteriores es posible entrenar la RNA con la regla delta generalizada para obtener los pesos adecuados.

La implementación del algoritmo es similar al anterior, pero ahora definimos D como el número de retardos que se desean, para este caso $D = 6$, así como las condiciones iniciales de la entrada, ya que si se definen n entradas donde $n = D + 1$ se necesitan D valores en el instante de tiempo $t = 0$, por lo que no es posible si solo se cuenta con una señal, por lo tanto se define un $\bar{0}_n$ en el instante $t = 0$, las configuraciones anteriores se encuntran en la función *PerceptronFA(Xs,Y,*$\bar{X}_{ant}$*,D)* que s enuentra en el apéndice B de este documento.

Los resultados que se obtuvieron de la función *PerceptronFA(Xs,Y,*$\bar{X}_{ant}$*,D)* se muestran a continuación donde se completo en un tiempo de $23.1s$ en tiempo real de una simulación definida para $10s$, Al igual que en el primer caso de una entrada una salida el tiempo no es aceptable para desempeñar tareas de seguimiento en linea. Sin embargo la aproximación que realizó la RNA del objetivo deseado es aceptable, pues la media es cercana a 0 como se muestra en la Figura A.3 en comparación a las anteriores.

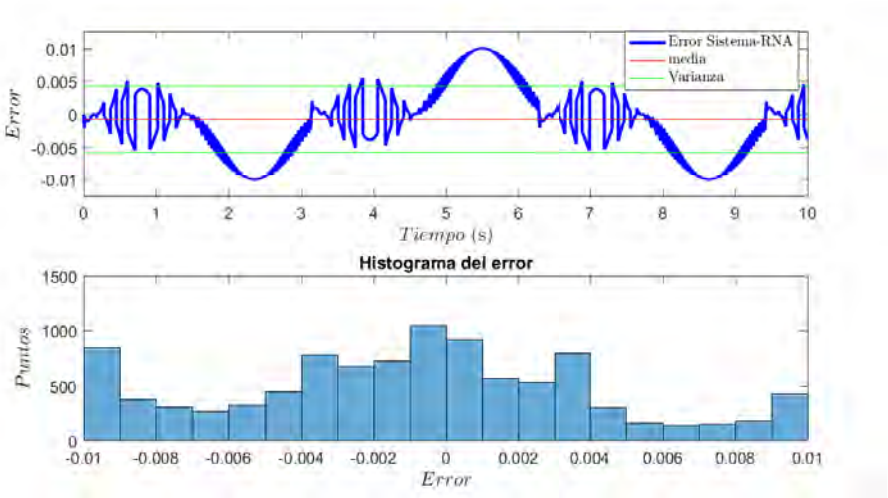


Figura A.3: Análisis del error con filtro adaptable

A.0.1. Herramientas y funciones de RNA

Se ha visto que las RNA con estructura de filtro adaptable permiten una mejor aproximación del objetivo deseado, a lo que se pretende trabajar con este tipo de RNA, pretendiendo mejorar el tiempo en que realiza la RNA sus procesos. Por lo que a continuación se muestran variantes de este tipo de RNA Adaline con filtro adaptable.

La herramienta de matlab, cuenta con varios diseños de redes predefinidas las cuales solo hace falta configurar de acuerdo a las necesidades.

Newlin

NEWLIN: Es una función de matlab que crea una RNA lineal la cual es usada como un filtro adaptable para el procesamiento y predicción de señales, su estructura esta definida por el comando newlin

$$net = newlin(P, S, ID, LR)$$

donde $P \in^{RxQ}$ es una matriz, con Q es el numero de entradas. S el numero de elementos del vector de salida. ID es el vector de entrada de retardos, por default es [0]. LR es la tasa de aprendizaje, por default es 0.01.

Utilizando como valores de entrada y salida a $X = cos(t) + sin(t)$ y $Y = sin(t)$ con $t \in [0, 10]$ se implemento la función newlin. Además al igual que en los casos anteriores se manejo un total de 6 entradas para así poder realizar la comparativa del desempeño en cuanto a tiempo y aproximación del objetivo. La información anterior es implementada en una función para Matlab, con el nombre de *PerceptronNewlin( )* el cual se muestra en el apéndice B de este documento.

A continuación se muestran los resultados obtenidos al implementar la función anterior. El tiempo que tardo la simulación fue de $6.3s$ en tiempo real respecto a $10s$ que simulo la RNA, por lo que se aprecia que es rápida en comparación al menos de 2 de las RNA propuestas anteriormente. Además en la Figura A.4 se muestra el comportamiento del error, lo cual nos dice que se concentra la mayor parte del tiempo en 0, sin embargo tiene muy pequeñas desviaciones que bien podrían ser despreciadas.

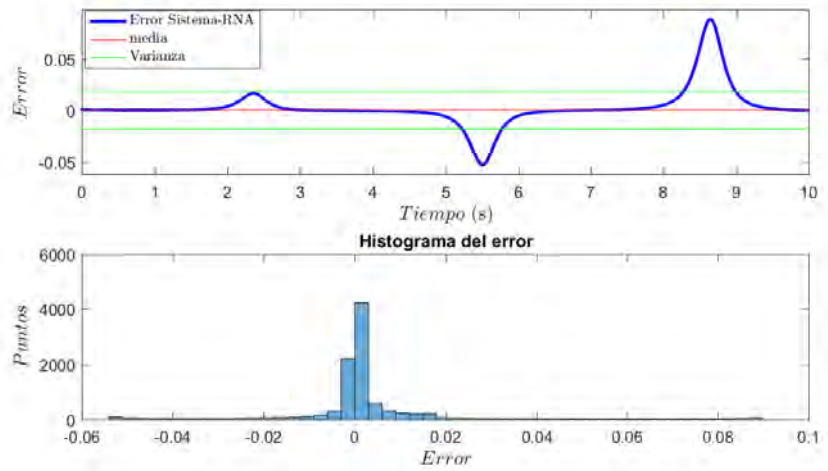


Figura A.4: Análisis del error con newlin de Matlab

Timedelaynet

Es una RNA con retardos en el tiempo la cual puede aprender a predecir valores en series de tiempo, la función *Timedelaynet* toma un vector fila para definir la entrada con retardos, un vector fila de tamaño N para declar el numero de capas ocultas y un método de entrenamiento, la estructura de la función esta dada por

$$timedelaynet(inputDelays, hiddenSizes, trainFcn)$$

Las entras y salidas de la RNA por default se colocan en 0, ya que estos tamaños los configura automáticamente para que coincidan con datos particulares, también es posible configurar las entradas y salidas por el usuario. El método de entrenamiento que utiliza por default es la función *trainlm* la cual realiza un entrenamiento donde actualiza los pesos y umbral según la optimización de Levenberg−Marquardt, a menudo este entrenamiento es más rápido que el *backpropagation*. Se implemento la función *PerceptronTimeDelay()* que se muestra en el apéndice B de este documento, en el se encuentra el algoritmo con los valores de entrada y salida así como la función propia del entorno Matlab timedelaynet con 6 retardos de la señal y una sola neurona.

Como resultado se obtuvo lo siguiente, donde se muestra en la Figura A.5 con un tiempo real de $28.3s$ respecto a $10s$ definidos en la simulación, se puede apreciar que tanto en tiempo como aproximación del objetivo no son favorables ya que lo realizó en más del doble de tiempo y el error no se concentra en 0.

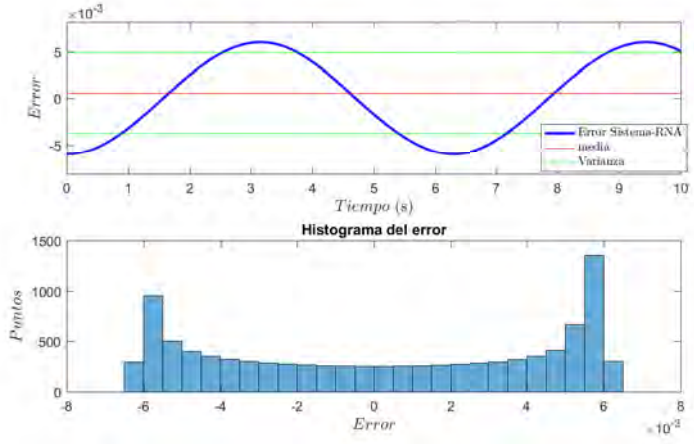


Figura A.5: Análisis del error de la RNA timedelay de Matlab

Apéndice B

Algoritmos

Función *Perceptron(Xs,Y)*

```
function y_tilde = fcn(V_xx,V_y)
[N ent]=size(V_xx); %col numero de neuronas, fil entradas)
X=[V_x]; Y=[V_y]; w0=zeros(1,ent); pesos=zeros(N,ent); lf
   =0.13;
for i=1:N
Wold=w0;      e2=1;
 while e2>0.0001  %rango de error
  eaux2 =0;
   d=Y(i,:); %salidas
   x=X(i,:); %entradas por neurona
   %ep=d'-purelin(Wold*x');
   ep=d'-(Wold*x');   %funcion de transferencia purelin
   Wnew=Wold+lf*ep*x;
   Wold=Wnew;
   eaux2=eaux2+ep'*ep;  %rango de error al cuadrado
  e2=eaux2;
 end
 pesos(i,:)=Wold ;
end
 y_tilde=pesos*V_xx   %(pesos * entrada) para aproximar la
    salida
```

Código para llamar la función *Perceptron(Xs,Y)*

```
t=0:0.001:10;
Xs=cos(t)+sin(t);
Y=sin(t);
```

```
y_deseada=zeros(1,length(t));
size(y_deseada)
%simula el tiempo real
for i=1:length(t)
  y_deseada(i)=y_tilde(Xs(i),Y(i));
end
error=y_deseada-Y;
plot(t,Y,'g',t,y_deseada,'b',t,error,'r');
legend({'Objetivo', 'Aproximacion RNA', 'Error'});
```

Función *PerceptronMISO($\bar{X}s, Y$)*

```
function y_tilde = y_tildeMISO(V_xx,V_y)
[N ent]=size(V_xx);  %entradas (ent) y neuronas (N))
X=[V_xx];  Y=[V_y];  w0=zeros(1,ent);
pes=zeros(N,ent);  lf=0.13;
for i=1:N
Wold=w0;
e2=1;
 while e2>0.0001  %rango de error
  eaux2 =0;
   d=Y(i); %salidas
   x=X(i,:); %entradas por neurona
   ep=d'-(Wold*x');   %funcion de transferencia purelin
   Wnew=Wold+lf*ep*x;
   Wold=Wnew;
   eaux2=eaux2+ep'*ep;  %rango de error al cuadrado
  e2=eaux2;
 end
 pes(i,:)=Wold ;
end
 y_tilde=pes*V_xx'   %pesos multiplicados por la entrada para
    aproximar la salida
```

Código para llamar la función *PerceptronMISO($\bar{X}s, Y$)*

```
t=0:0.001:10;
X1=(cos(t)+sin(t)); X2=cos(t);
X3=sin(t);          X4=-(cos(t)+sin(t));
X5=-sin(t);         X6=cos(t);
Xs=[X1',X2',X3',X4',X5',X6'];
Y=sin(t);
y_deseada=zeros(1,length(t));
```

```
%simula el tiempo real
for i=1:length(t)
  y_deseada(i)=y_tildeMISO(Xs(i,:),Y(i));
end
error=y_deseada-Y;
plot(t,Y,'g',t,y_deseada,'b',t,error,'r');
legend({'Objetivo', 'Aproximacion RNA', 'Error'});
```

Función *PerceptronFA(Xs,Y,X_a^-nt,D)*

```
t=0:0.001:10;
Xs=(cos(t)+sin(t));
Y=sin(t);
y_deseada=zeros(1,length(t));
d=6;    X0=zeros(1,d);    %condiciones iniciales
%simula el tiempo real
for i=1:length(t)
  [y_deseada(i),X_anterior]=y_tildeMISO_delay(Xs(i),Y(i),X0,d)
     ;
  X0=X_anterior;
end
error=y_deseada-Y;
plot(t,Y,'g',t,y_deseada,'b',t,error,'r');
legend({'Objetivo', 'Aproximacion RNA', 'Error'});
xlabel('tiempo(seg)')
```

Código para llamar la función *PerceptronFA(Xs,Y,X_a^-nt,D)*

```
function [y_tilde X_ant]= y_tildeMISO_delay(V_xx,V_y,V_ant,D)
%Genera un vector de D retardos, que representa la entrada
V_nuevo=ones(1,D);
for j=D:-1:1
     if j==1
       V_nuevo(1,j)=V_xx;
     else
       V_nuevo(1,j)=V_ant(1,j-1);
     end
end
 X_ant=V_nuevo;
 V_x=V_nuevo;


[N ent]=size(V_x);  %entradas (ent) y neuronas (N))
```

```
X=[V_x];  Y=[V_y];  w0=zeros(1,ent);
pes=zeros(N,ent);  lf=0.13;
for i=1:N
Wold=w0;
e2=1;
 while e2>0.0001  %rango de error
  eaux2 =0;
   d=Y(i); %salidas
   x=X(i,:); %entradas por neurona
   ep=d'-(Wold*x');   %funcion de transferencia purelin
   Wnew=Wold+lf*ep*x;
   Wold=Wnew;
   eaux2=eaux2+ep'*ep;  %rango de error al cuadrado
  e2=eaux2;
 end
 pes(i,:)=Wold ;
end
 y_tilde=pes*V_x' ;  %pesos multiplicados por la entrada para
    aproximar la salida
```

Función *PerceptronNewlin( )*

```
t = 0 : 0.001 : 10;
X = cos(t)+sin(t);
Y=sin(t);
P = con2seq(X); %es transformado a secuencias de tiempo
Target = con2seq(Y);
%Se define la RNA Adaline con entradas en un dominio de [-2
   2], con
%una sola neurona, y 6 retardos respecto a la senial de
   entrada
%0.5 es la constante de aprendizaje
net = newlin([-2 2],1,[0 1 2 3 4 5 6],0.13);
%entrenamiento de la RED
[net,Y_aprox,E,Pf] = adapt(net,P,Target);
aux=cell2mat(Y_aprox)
error=Y-aux;
plot(t,Y,'r',t,aux,'b',t,error,'g');
```

Función *PerceptronTimeDelay( )*

```
t = 0 : 0.001 : 10;
X = cos(t)+sin(t);
```

```
Y=sin(t);
P = con2seq(X); %es transformado a secuencias de tiempo
T = con2seq(Y);
     net = timedelaynet(1:6,1);
     [Xs,Xi,Ai,Ts] = preparets(net,P,T);
     net.trainParam.showWindow=false;
     net = train(net,Xs,Ts,Xi);
Y_aprox = sim(net,Xs,Xi,Ai)

aux=cell2mat(Y_aprox)
error=Y-aux;
plot(t,Y,'r',t,aux,'b',t,error,'g');
```