EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Tecnológico Nacional de México

Centro Nacional de Investigación y Desarrollo Tecnológico

Tesis de Doctorado

Identificación de Sistemas Mediante el Uso de Redes Neuronales de Orden Fraccionario

presentada por

M.C. Carlos Jesús Zúñiga Aguilar

como requisito para la obtención del grado de

Doctor en Ciencias en Ingeniería Electrónica

Director de tesis

Dr. José Francisco Gómez Aguilar

Codirector de tesis

Dr. Héctor Manuel Romero Ugalde

Cuernavaca, Morelos, México. Julio de 2020.

cenidet
Centro Nacional de Investigación y Desarrollo Tecnológico

cenidet® Centro Nacional de Investigación y Desarrollo Tecnológico	**ACEPTACIÓN DE IMPRESIÓN DEL DOCUMENTO DE TESIS DOCTORAL**	**Código: CENIDET-AC-006-D20**
		Revisión: O
	Referencia a la Norma ISO 9001:2008 7.1, 7.2.1, 7.5.1, 7.6, 8.1, 8.2.4	**Página 1 de 1**

Cuernavaca, Mor., a 29 de junio de 2020.

Dr. Gerardo Vicente Guerrero Ramírez
Subdirector Académico
P r e s e n t e

At'n: Dr. Víctor Manuel Alvarado Martínez
Presidente del Claustro Doctoral
del Departamento De Ingeniería Electrónica

Los abajo firmantes, miembros del Comité Tutorial del estudiante **Carlos Jesús Zúñiga Aguilar** manifiestan que después de haber revisado el documento de tesis titulado **"Identificación de sistemas mediante el uso de redes neuronales de orden fraccionario"**, realizado bajo la dirección del **Dr. José Francisco Gómez Aguilar** y codirección del **Héctor Manuel Romero Ugalde,** el trabajo se ACEPTA para proceder a su impresión.

A T E N T A M E N T E

Dr. José Francisco Gómez Aguilar CENIDET
CENIDET

Dr. Héctor Manuel Romero Ugalde
CEA-GRENOBLE

Dr. Víctor Manuel Alvarado Martínez
CENIDET

Dr. Carlos Daniel García Beltrán
CENIDET

Dr. Juan Reyes Reyes
CENIDET

Dr. José Alfredo Hernández Pérez
CIICAP-UAEM

Reciba un cordial saludo.

c.c.p: M.E. Guadalupe Garrido Rivera / Jefa del Departamento de Servicios Escolares.
c.c.p: Dr. Mario Ponce Silva / Jefe del Departamento de Ingeniería Electrónica.
c.c.p: Expediente.

Cuernavaca, Morelos 16/julio/2020
OFICIO No. SAC/ 220/2020
Asunto: Autorización de impresión de tesis

CARLOS JESÚS ZÚÑIGA AGUILAR
CANDIDATO AL GRADO DE DOCTOR EN CIENCIAS
EN INGENIERÍA ELECTRÓNICA
P R E S E N T E

Por este conducto tengo el agrado de comunicarle que el Comité Tutorial asignado a su trabajo de tesis titulado **"Identificación de sistemas mediante el uso de redes neuronales de orden fraccionario"**, ha informado a esta Subdirección Académica, que están de acuerdo con el trabajo presentado. Por lo anterior, se le autoriza a que proceda con la impresión definitiva de su trabajo de tesis.

Esperando que el logro del mismo sea acorde con sus aspiraciones profesionales, reciba un cordial saludo.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
"Conocimiento y tecnología al servicio de México"

CENTRO NACIONAL DE INVESTIGACIÓN Y DESARROLLO TECNOLÓGICO SUBDIRECCIÓN ACADÉMICA

DR. GERARDO VICENTE GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO

C.c.p. M.E. Guadalupe Garrido Rivera. Jefa del Departamento de Servicios Escolares
Expediente

GVGR/CHG

Mira que te mando que te esfuerces y seas valiente; no temas ni desmayes, porque Jehová tu Dios estará contigo en dondequiera que vayas.
Josué 1:9

Dedicatoria

Dios, gracias por permitirme llegar hasta aquí, por no dejarme caer a pesar de querer tirar la toalla en varias ocasiones. Me has puesto pruebas y obstáculos que nunca imaginé que podría superar. Gracias Señor por todo lo que nos das porque sin lugar a dudas siempre provees.

No hay gratitud más grande que la que siento hacia ustedes, ustedes que me dieron la vida y me han dado la oportunidad de ser su hijo. Elena y Cirino, padres míos, Dios me mandó a sus brazos para que se me hiciera posible concluir con este grado que sin duda es por y para, ustedes.

De una familia donde fuimos cinco integrantes, yo era el más pequeño y, a pesar de que los años han pasado, me sigo sintiendo protegido por ustedes hermanos, María, Jesús y América. Gracias por escucharme, por estar ahí para ayudarme, por explicarme las cosas de la manera más sencilla. Siempre los llevaré en mi corazón.

Hasta estos días sé que soy uno de los tíos más felices del mundo. Chaparritos, Rebeca, Mateo y Jesús Andrés, ustedes me dieron el título de ser su tío. Siempre los protegeré como sus papás lo han hecho conmigo a lo largo de los años.

Al parecer siempre estuve equivocado en los lugares donde buscaba entendimiento, comprensión y tranquilidad. Hasta que llegaste tú, Edna, en ti encontré la solución a todo lo que yo me preguntaba si existía. Te has convertido en todas mis razones, pensamientos y adoraciones. Sin duda y sin temor a equivocarme eres mi constante en este mundo infinito de variables.

Abuelos, me hubiera gustado que vieran el fruto del esfuerzo de sus hijos al criarme y permitirme llegar hasta aquí pero, entiendo que la vida no es infinita. Abuela Carmen, Elena, Carlos y Cirino, gracias por darme todo ese amor y alegría. Para mis tíos y primos, May, Napo, Pita, Neto, Mario, Laura, Olis, Cristy, Ester, Lulú, Vani, Apo, Beba, Patilla, Lalo, gracias por todas sus enseñanzas, paciencia y cariño.

Al nacer, Dios nos pone en una familia que somos incapaces de escoger, pero los amigos, los amigos son esa familia que el corazón escoge, Susana, Fish, Pedro, Emmanuel, Hugo, Moises, Gabs, Chuy, Vallarín, Felipe, a mis roomies Héctor y Pancho, Paty, Didhier les agradezco de corazón todos sus consejos, todo su aprecio y sobretodo, paciencia.

A todas las personas que mencioné, quiero que sepan que de aquí en adelante este trabajo es gracias a ustedes, porque sin ustedes tal vez la cuesta hubiera sido aún más inclinada.

¡MUCHAS GRACIAS!

Agradecimientos

Durante toda mi formación recibí el apoyo de diversas personas que se me nutrieron de conocimiento y sabiduría. En este espacio me gustaría agradecer a todas esas personas que transmitieron sus ideas en mí.

Al Dr. Enrique Martínez Peña quien me impulsó a realizar el posgrado cuando me encontraba haciendo mi licenciatura, que fue mi compañero de trabajo durante algunos meses pero siempre un consejero amable, honesto y humilde.

Primeramente, estaré agradecido por siempre al CENIDET que me acogió desde el primer día en el que pisé sus aulas y a su directora la Dra. Yesica Imelda Saavedra Benítez y su arduo trabajo. Gracias a CONACYT por haberme apoyado económicamente durante mi preparación doctoral por medio de la beca y beca mixta.

A mis profesores en el CENIDET de los cuales no solo aprendí cuestiones académicas sino a como congeniar con la vida: Dr. Alejandro, Dr. Carlos Astorga, Dra. Ma. Guadalupe y Dr. Adam, sus enseñanzas siempre las llevaré conmigo a donde quiera que vaya. Lore, muchas gracias por siempre atenderme y ayudarme con una sonrisa en el rostro.

Durante mi preparación, se tuvieron ciertos tropiezos y tal vez dificultades, es por ello que me gustaría agradecer a los doctores Juan Reyes, Carlos Daniel, Vicente Guerrero y José Alfredo por ayudarme a superar esos tropiezos, por aconsejarme, por reconocer mi trabajo y ayudarme con sus atentas revisiones. Doctores, no podría estar más agradecido con ustedes porque siempre me hicieron sentir en confianza a pesar de que no estaban obligados a hacerlo.

Dr. Víctor Manuel Alvarado Martínez, gracias por atenderme sin importar qué pregunta fuese. Gracias por inmiscuirme en esto que llamamos investigación, gracias por ser mi primer mentor. Siempre lo recordaré ya que usted cimentó mis conocimientos acerca de la Identificación de Sistemas y me enseñó a quererla desde el principio de mi posgrado. Nunca olvidaré la Pepsi de dieta.

Por último, pero no menos importante, doctores, mentores, amigos José Francisco Gómez Aguilar y Héctor Manuel Romero Ugalde, estaré eternamente agradecido por su labor, por sus palabras de aliento, por todo el trabajo y tiempo que invirtieron en mí. Son de las personas más nobles, honestas y admirables que Dios me ha permitido conocer, muchas gracias por luchar y creer en mí. Nunca olvidaré lo que han hecho de y por mí, espero poder seguir contando con ustedes. Gracias por sus consejos, por sus oídos, opiniones, regaños, risas y alegrías, ¡Muchas gracias!

Resumen

El siguiente documento presenta teoría acerca del cálculo fraccionario (CF), hechos históricos que han forjado los cimientos de su disciplina, funciones especiales utilizadas y diversas definiciones de derivada fraccionaria así como las diferencias entre cada una de ellas. Del mismo modo, se presenta la definición de redes neuronales artificiales (RNAs), su arquitectura, elementos que constituyen a una RNA, capacidad de aprendizaje mediante algoritmos de optimización y la identificación de sistemas con redes neuronales. La combinación de ambas disciplinas ha dado como resultado el concepto de redes neuronales artificiales fraccionarias (RNAF), las cuales son presentadas en este trabajo.

Se presenta una aplicación que relaciona a las RNAs con el CF, la cual es basada en la búsqueda de soluciones de ecuaciones diferenciales fraccionarias con orden variable (EDF-OV). El primer caso de estudio muestra una EDF con orden constante con el fin de analizar las diferencias entre los métodos analíticos, algoritmos numéricos y la RNA. Los siguientes casos de estudio han sido incluidos con el fin de denotar las ventajas que una RNA ofrece por encima de los algoritmos numéricos con ayuda de la medición de diversos índices de desempeño y pruebas de costo computacional.

Después, se presenta una metodología para entrenar RNAs mediante el uso de algoritmos de orden fraccionario para la identificación de sistemas (IS). Lo anterior con el fin de mostrar las ventajas y diferencias entre los algoritmos de optimización convencionales y el algoritmo propuesto. El algoritmo es probado mediante la identificación de una secadora de cabello de uso industrial, un sistema mecánico con histéresis y por último un sistema médico encargado de la predicción de concentración de glucosa en la sangre (CGS).

Finalmente, se presenta una metodología de IS basada en definiciones fraccionarias-fractales (FF). Se propone un mapeo entre definiciones fraccionarias con el fin de justificar la estabilidad de la metodología mediante la teoría de estabilidad de Lyapunov. La metodología propuesta es puesta a prueba mediante la identificación de sistemas de dos tanques en cascada, la posición de un brazo de un toca discos y un motor eléctrico utilizado en automóviles.

Abstract

The following document presents theory about fractional calculus (CF), historic data that has forged the foundation of this discipline, special functions used and several definitions of fractional derivative as well as the differences between each of them. In the same way, it is presented the definition of artificial neuronal networks (RNAs), its architecture, element that constitutes a RNA, learning capacity through optimization algorithms and the definition of neuronal network systems. The combination of both disciplines has resulted in the concept of fractional artificial neural networks (RNAF), which are presented in this work.

An application is presented that relates the RNAs with the CF. Which, is based in the search of solutions of fractional differential equations with variable order (EDF-VO). The first study case shows an EDF with constant order with the purpose of analyze the differences between the analytical methods, numerical algorithms and the RNA. The following study cases have been included in order to denote the advantages that an RNA offers above the numerical algorithms with the measurement of diverse performance indexes and computational cost tests.

Then, it is presented a methodology to train RNAs through the use of fractional order algorithms for the identification of systems (IS), with the purpose of showing the advantages and differences between the conventional optimization algorithms and the proposed algorithm, the algorithm is tested through the identification of an industrial hair dryer, a mechanical system with hysteresis and finally, a medical system in charge of the prediction of the glucose concentration in the blood (CGS).

Finally, it is presented an IS methodology based in fractal-fractional (FF) definitions. It is proposed a mapping between fractional definitions with the purpose of justify the stability of the methodology through Lyapunov's stability theory. The proposed methodology is tested through the system identification of two cascading tanks, the position of a turntable arm and an electrical motor used in cars.

Índice general

Índice de figuras

Índice de tablas

Lista de abreviaturas

CF	**C**álculo **F**raccionario
ED	**E**cuación **D**iferencial
EDF	**E**cuación **D**iferencial **F**raccionaria
EDFOV	**E**cuación **D**iferencial **F**raccionaria **O**rden **V**ariable
FF	**F**raccionario-**F**ractal
ML	**M**itag-**L**effler
RL	**R**iemann-**L**iouville
GL	**G**rünwald-**L**etnikov
LC	**L**iouville-**C**aputo
CFC	**C**aputo-**F**abrizio-**C**aputo
ABC	**A**tangana-**B**aleanu-**C**aputo
ABCV	**A**tangana-**B**aleanu-**C**aputo-**OV**
FFP	**F**ractal-**F**raccionario-**P**otencia
FFE	**F**ractal-**F**raccionario-**E**xponencial
FFM	**F**ractal-**F**raccionario-**ML**
RNA	**R**ed **N**euronal **A**rtificial
Hopf	**H**opfield
MLP	**M**ulti **L**ayer **P**erceptron
RNR	**R**ed **N**euronal **R**ecurrente
BRNN	**R**ed **N**euronal **R**ecurrente **B**iderccional
CNN	**R**ed **N**euronal **C**onvolucional
LST	**L**ong **S**hort **T**erm **M**emory
NFF	**R**ed **N**euronal **F**ractal **F**raccionaria
RNDF	**R**ed **N**euronal **D**dinámica **F**raccionaria
IS	**I**dentificación de **S**istemas
WH, W-H	**W**iener **H**ammerstain
HW, H-W	**H**ammerstain **W**iener
W	**W**iener
H	**H**ammerstain
LM	**L**evenberg-**M**arquardt
GD	**G**radiente **D**escendente
OE	**O**utput-**E**rror
MIMO	**M**ultiple **I**nput **M**ultiple **O**utput
OMS	**O**rganización **M**undial de la**S**alud
DT1	**D**iabetes de **T**ipo **1**
GS	**G**lucosa en la**S**angre
CGS	**C**oncentración de **G**lucosa en la**S**angre
MGS	**M**onitoreo de **G**lucosa en la**S**angre
PGS	**P**redicción de **G**lucosa en la**S**angre
PA	**P**ancreas **A**rtificial

IOB	Insulina Activa
COB	Carbohidratos Activos
ANSM	Autoridad Nacional de seguridad Francesa
PID	**P**roporcional **I**ntegral **D**erivativo
PSO	**P**article **S**warm **O**ptimization
TS	**T**iempo de **S**imulación
FIT	Término de Ajuste
FPE	**F**inal **P**rediction **E**rror
RMSE	**R**oot **M**ean **S**quare **E**rror
PRBS	**P**seudo **R**andom **B**inary **S**ignal
HP	**H**orizonte de **P**redicción
PMSM	**P**ermanent **M**agnet **S**ynchronous **M**otor

Lista de Símbolos

w_i	Peso sináptico.
Ω	Matriz de pesos sinápticos.
Γ	Función Gamma.
n_a	Número de regresos en la señal de salida.
n_b	Número de regresos en la señal de entrada.
n_k	Tiempo muerto o regresor natural en la entrada.
$\hat{y}$	Salida estimada o salida de la red neuronal
$y(k)$	Señal de salida real o medición.
$\Upsilon(k)$	Compensación neuronal.
$e(k)$	Error.
k	Instante de tiempo o iteración.
N	Número total.
R	Parámetro de dirección de búsqueda.
η, μ	Tamaño de paso o coeficiente de aprendizaje.
$\mathcal{E}$	Función objetivo.
η_0	Valor inicial de η.
μ_0	Valor inicial de μ.
s	Compensación Fraccionaria.
α, γ	Orden fraccionario.
β	Orden fractal.
A, B, L	Matriz de parámetros de RNDF.
Ψ_x	Variable auxiliar para los estados de salida.
Ψ_u	Variable auxiliar para los estados de entrada.
δ_x	Incertidumbre de la salida.
δ_u	Incertidumbre de la entrada.
φ	Función de Activación.
$J(\cdot)$	Matriz Jacobiana.
nn	Número de neuronas.
J_u	Matriz de regresión de la entrada.
$J_{\hat{y}}$	Matriz de regresión de la salida.
W	Parámetro de peso sináptico.
n_p	Número de parámetros.

CAPÍTULO 1

Introducción

1.1. Introducción

Se vive en un mundo en constante evolución, complejo, caótico y lleno de ruido. Sin embargo, la inteligencia humana tratado de inmiscuirse en el caos con la búsqueda de patrones que caracterice los fenómenos del universo. El desarrollo de la especie humana se ha debido especialmente gracias a la facultad de definir estos patrones y poderlo utilizar a su favor. La ciencia ha permitido explotar la capacidad del ser humano de explorar el mundo de una manera simplificada convirtiendo todo el ruido y caos en conocimiento puro, es decir, reconstruyendo la realidad a través de modelos.

Un modelo es una construcción conceptual simplificada de una realidad más compleja. En la vida cotidiana se convive con diversos tipos de modelos, por ejemplo un mapa, un mapa es un modelo ya que puede representar de manera simplificada en un plano bidimensional el mundo tridimensional en el que se vive. Es decir, matemáticamente representa una función que es capaz de mapear de una dimensión a otra mediante una expresión matemática como $\mathbf{M} : \mathbb{R}^2 \to \mathbb{R}^3,\ (x, y) \mapsto \mathbf{M}(x, y)$. Una partitura (documento que indica cómo debe interpretarse una composición musical) como otro ejemplo, es una simple representación de cómo diferentes instrumentos deben combinarse, sincronizarse y tocarse para producir siempre la misma canción. Desde el punto de vista ingenieril, una partitura puede analizarse mediante el espectro de frecuencia producido por el sonido de cada instrumento pero, estos espectros no es la información que un músico necesita para interpretar una melodía. Por lo tanto, un modelo siempre busca el equilibrio entre aproximarse a representar correctamente la realidad y ser simple para poder utilizarlo.

Desde otro punto de vista, un modelo puede ser una ecuación física donde se utilizan las relaciones matemáticas entre diferentes variables para así poder aproximar el comportamiento físico de la realidad. A este último concepto se conoce también como **modelo matemático**. Un modelo matemático puede ser tan simple como una constante (como la separación entre las vías del tren) y tan complicado como las ecuaciones de Maxwell. Pero, como se ha mencionado con anterioridad, siempre se busca un equilibrio entre la realidad y la simpleza. La manera eficaz de garantizar dicho equilibrio, es comparando el comportamiento del modelo propuesto con datos del fenómeno que se desea estudiar. Si el error entre estos dos **sistemas** es mínimo o nulo, el modelo puede ser ajustado para

encontrar nuevos escenarios, analizar su pasado o inclusive, predecir su futuro.

Existen dos maneras de obtener un modelo matemático, el primero se basa en el análisis de la física que rodea al sistema por ejemplo, un sistema masa-resorte-amortiguador es representado mediante las leyes de Newton, Hook, viscosidad de Newton, entre otras. Se pueden hacer ciertas suposiciones como que la temperatura es constante al igual que la fricción del material viscoso, qué fuerza de movimiento es puntual y la dirección, etc. La precisión del modelo dependerá unicamente de la persona que desea explicar el fenómeno o el comportamiento del sistema y está altamente relacionado con las suposiciones de modelado que se realicen. Por otro lado, el segundo método se basa en la construcción de modelos matemáticos mediante la estimación de parámetros utilizando señales de entrada excitadoras y las señales de salida que estas en consecuencia producen. Este último método es conocido como **Identificación de Sistemas (IS)**.

Ahora, hablando un poco del funcionamiento del sistema nervioso, el cerebro humano está constituido por un gran número de elementos (10^{11} aproximadamente) altamente interconectados (aproximadamente 10^4 conexiones por elemento) llamados neuronas. Una neurona biológica consta de tres partes esenciales, las dendritas, el cuerpo de la célula o soma y el axón. Las dendritas son ramificaciones nerviosas que cargan de señales eléctricas el cuerpo de la célula y se encuentran alrededor de la célula y del axón. El axón es una fibra nerviosa más larga que las dendritas encargado de llevar la señal desde el cuerpo de la célula hacia otras neuronas. El cuerpo de la célula o soma es el encargado de realizar la suma de todas las señales entrantes, también se encarga de estimular a la célula de tal manera que esta produzca una señal hacia otra neurona. A la "unión" entre el axón de una célula con una dendrita de otra célula se le llama sinápsis. Aprovechando las ventajas de las neuronas biológicas, las redes neuronales artificiales (RNA) se definen como sistemas de mapeos no lineales cuya estructura se basa en la distribución de información como en el sistema nervioso de humanos y animales. Constan de un número grande de procesadores simples ligados por conexiones con parámetros ponderados llamados pesos sinápticos. Las unidades de procesamiento se denominan neuronas. Cada unidad recibe entradas de otros nodos y juntos, generan una salida simple ya sea un escalar o un vector que depende de la información recibida. Las neuronas artificiales también pueden nombrarse como nodos, neuronodos, celda, unidad o elemento de procesamiento (PE) y fueron propuestas por [2] a inicios de los años 40's.

Por otro lado y un poco ajeno a lo antes mencionado pero igual de importante en esta tesis. El Cálculo Fraccionario (CF) es una rama del análisis matemático que estudia los operadores de derivación e integración de ordenes no enteros, es decir, $\mathcal{D}^\sigma$ donde $\mathcal{D}$ representa el operador de derivación y σ es un número real que representa el orden de derivación. Si $\sigma < 0$ se estaría representando una integral fraccionaria, si $\sigma > 0$ se representa una derivación fraccionaria, si $\sigma = 1$ es el caso clásico y si $\sigma = 0$ se tiene la misma función. Aunque en la actualidad se encuentran diversas definiciones de derivada fraccionaria, el principal objetivo del CF es generalizar las derivadas de orden entero, debido a que dentro del dominio de σ se encuentran casos particulares donde su valor es entero y la derivada puede ser calculada de manera convencional. Por lo tanto, con el fin de generalizar aún más la arbitrariedad que el orden de derivación es capaz de tener, el dominio de σ puede estar entre los números complejos. El CF ha destacado en la comunidad científica gracias a sus vastas aportaciones matemáticas en teorías probabilísticas, procesos estocásticos, transformaciones integrales, entre otras. A pesar de que la teoría matemática que órbita al CF es compleja y rigurosa, se han encontrado diversas explicaciones de fenómenos del mundo real que el cálculo convencional es incapaz de describir y ha provocado diversos

cuestionamientos de cómo interpretar lo obtenido por la derivada fraccionaria. La interpretación física mas aceptada del CF involucra fenómenos de disipación en escalas de tiempo diferentes a la ordinaria definidas como *tiempos cósmicos* [3].

Aprovechando las ventajas de las disciplinas antes mencionadas, se utilizarán en conjunto a las RNAs y el CF para realizar la IS de diversos fenómenos dinámicos no lineales. A continuación se presentan algunos conceptos generales para el desarrollo de este trabajo.

1.1.1. Sistema

Un sistema es una porción del universo donde variables de diversa naturaleza interactúan entre sí para completar un fin. La Figura 1.1 muestra las variables observables representadas por $y(t)$, señales o variables manipuladas $u(t)$ y las variables que modifican a las variables de salida pero, que no son medibles conocidas como perturbaciones $v(t)$.

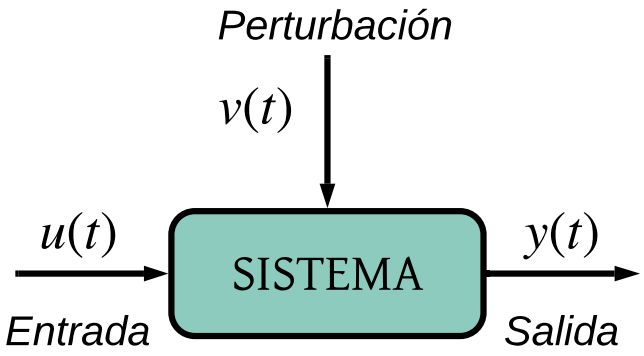


Figura 1.1: Sistema dinámico con entrada $u(t)$, perturbación $v(t)$ y salida $y(t)$,

1.1.2. Identificación de Sistemas

En IS se busca proponer modelos matemáticos de sistemas dinámicos basados en datos que son observables, es decir, que son medibles. En el entorno, abundan diversos sistemas dinámicos que pueden ser modelados mediante las leyes físicas que caracterizan el comportamiento del sistema pero, aprovechando la principal ventaja que la IS ofrece, estos sistemas pueden ser modelados sin utilizar las leyes físicas que rigen a los sistemas por lo que, en consecuencia, las aplicaciones de la IS son abundantes. La Figura 1.2 muestra un esquema del concepto de la identificación de sistemas.

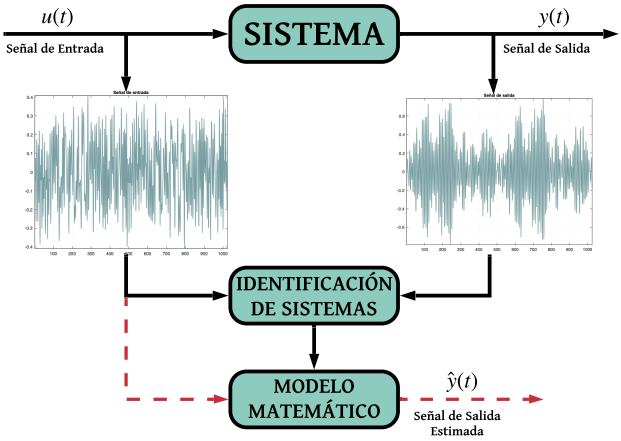


Figura 1.2: Identificación de Sistemas.

1.1.3. Proceso de Identificación

En [4] se propone una metodología para generalizar el proceso de identificación. El algoritmo propuesto se muestra en la Figura 1.3 y es descrito en los siguientes puntos:

1. *Obtención de datos de entrada y salida:* Para ello se debe excitar o alimentar al sistema mediante la aplicación de una señal de entrada que en consecuencia produce una señal de salida. Ambas señales son registradas durante un intervalo de tiempo.

2. *Tratamiento previo de los datos registrados:* Los datos registrados están generalmente acompañados de valores numéricos indeseables que pueden ser necesariamente corregidos antes de realizar la identificación del modelo, esto permitirá facilitar y mejorar el proceso de identificación.

3. *Elección de la estructura del modelo:* Se debe seleccionar la estructura de modelo que se utilizará para caracterizar el comportamiento del sistema dinámico. Este punto tiende a facilitarse en gran medida si se tiene cierto conocimiento sobre las leyes físicas que rigen al proceso.

4. *Obtención de los parámetros del modelo:* Se procede a la estimación de los parámetros que conforman a la estructura del modelo seleccionado en el punto anterior del tal manera que este se ajuste mejor a la respuesta del modelo real.

5. *Validación del modelo:* El último paso consiste en verificar si el modelo propuesto después de haber realizado la estimación cumple con los estándares deseados. Estos estándares se basan en mediciones estadísticas de la comparativa entre las señales del sistema real (señal de salida medida) y la salida del modelo propuesto, número de parámetros, distancia entre las dos señales, etc. Si se llega a la conclusión de que el modelo no es válido, se debe revisar las posibles causas:

    a) El conjunto de datos de entrada y salida no proporciona suficiente información sobre la dinámica del sistema real.

    b) La estructura escogida no es capaz de proporcionar una buena descripción del modelo.

    c) El criterio de ajuste de parámetros seleccionado no es el adecuado.

Dependiendo la causa, deberá repetirse el proceso de identificación desde el punto que corresponde. Por ende, puede decirse que el proceso de identificación es iterativo.

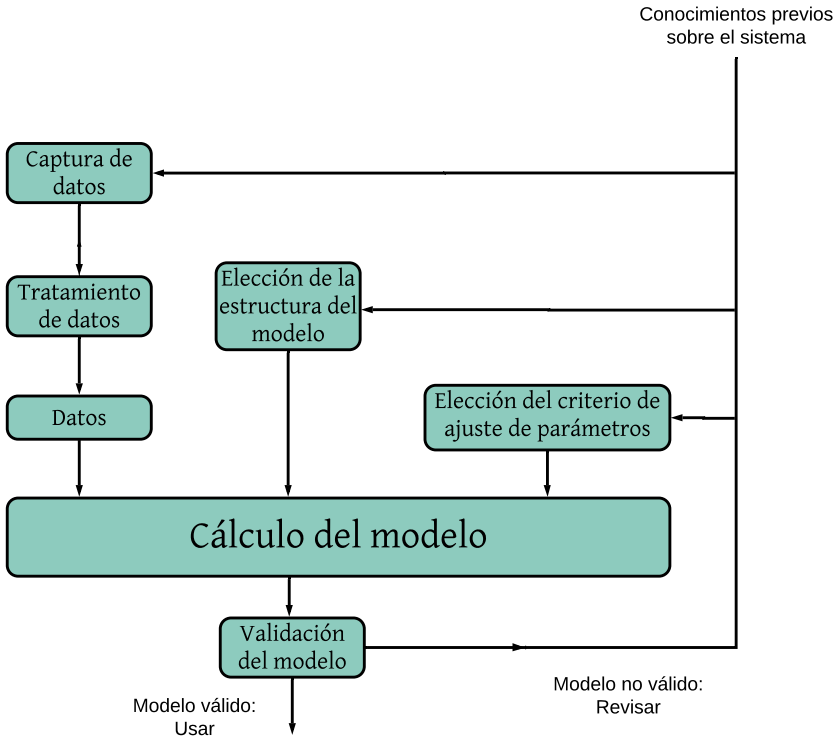


Figura 1.3: Proceso de Identificación de Sistemas.

1.2. Estado del Arte

1.2.1. Antecedentes del CENIDET

En el Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET) se han desarrollado trabajos relacionados con el CF y otros correspondientes a la IS mediante el uso de RNAs.

En [5] se realizó la IS para sistemas no lineales mediante la implementación de modelos W-H utilizando las ventajas que ofrece la transformada de Laplace con frecuencias fraccionarias (TLF) como lo es la descripción de sistemas de orden superior mediante la función de transferencia (FT) cuyo polinomio característico está compuesto por un grado inferior. Del mismo modo que en [5], en [6] se propuso una metodología para la identificación de sistemas mediante estructuras a bloques H-W utilizando derivadas Fractales. También se han desarrollado trabajos referentes al modelado de sistemas, tal como se realizó en [7], donde se comprobó que un modelo fraccionario con un número de parámetros reducido es capaz de describir y generalizar la dinámica de un regenerador de energía.

Referente a la IS con RNAs, en [8] se propuso un método capaz de obtener la misma precisión que ofrece una RNA con un número de parámetros alto utilizando la misma estructura de RNA pero con un número de parámetros bajo. En [9] se propuso una estructura NARX en forma de perceptrón y una estructura HW para realizar la IS para sistemas no lineales utilizando el algoritmo de Levenberg-Marquardt (LM). En [10] se propuso

utilizar una RNA similar a lo propuesto en [8] implementando un estimador robusto de máxima verosimilitud utilizando el algoritmo de LM para la estimación de los pesos sinápticos.

Concluyendo con lo anterior, en CENIDET no se ha realizado ningún trabajo que relacione la disciplina de las RNA y del CF para realizar la IS.

1.2.2. Identificación de Sistemas con Redes Neuronales Artificiales

A pesar de que en la literatura es posible encontrar diversas estructuras para realizar la IS, las RNAs han sido utilizadas gracias a su implementación sencilla y a su optimización (referente a los pesos sinápticos).

En [11] se realizó la IS con una RNA basada en una estructura Elman. Una RNA Elman está conformada por tres capas ocultas neuronas adicionales conectadas de manera independiente en la segunda capa oculta, es decir, la salida de cada neurona de la segunda capa oculta es conectada a las neuronas independientes y estas a su vez, son señales de entrada para las neuronas de la segunda capa oculta. Los autores de [11] utilizaron a la RNA para modelar celdas de combustible con membrana de cambio de protones.

En [12] se realizó una comparación de desempeños a tres RNAs con estructuras totalmente diferentes con el fin de seleccionar la estructura con mayor capacidad de describir el comportamiento de sistemas dinámicos. Entre las estructuras utilizadas en [12] está una red neuronal reailmentada multicapa (MLFFNN por sus siglas del inglés *Multilayer feedfoward neural network*), una red neuronal recurrente diagonal (DRNN por el inglés *diagonal recurrent neural network*) y una red neuronal con entrada exógena autorregresiva (NARX del ingés *nonlinear autoregressive with exogenous inputs neural network*). En el trabajo de investigación identificaron diversos sistemas dinámicos no lineales entre los cuales se encuentra el comportamiento del eslabón de un brazo robot pero, infortunadamente, todos los sistemas identificados no provienen de datos reales. Sin importar la proveniencia de los sistemas identificados, la DRNN obtuvo un mejor desempeño midiendo la media del error al cuadrado y **el número total de parámetros**.

En [13] se presentó una revisión bibliográfica acerca de diversas técnicas para la identificación o predicción de la salud sobre agentes individuales. El trabajo explica que utilizan a las RNAs debido a estas son capaces de procesar grandes cantidades de información y que su estructura es modificable en función de la problemática que se deseé solucionar.

En [14], se propuso utilizar una RNA difusa para la identificación de diversos sistemas. La RNA propuesta es comparada con varias estructuras de RNA difusas con el fin de comprobar que su RNA tiene un mejor desempeño. Entre los sistemas identificados se encuentra la predicción de sistemas financieros como el costo futuro de metales, rastreo de precios, mercancía, etc.

Como se ha podido observar en las referencias presentadas, a pesar de que las RNAs datan de los años 40s, estas no han perdido el interés por la comunidad científica para realizar la IS. Los sistemas identificados son variados y con dinámicas altamente complejas. En cada uno de los trabajos presentados, las estructuras de RNA son diferentes; esto con el fin de esclarecer la idea de adaptabilidad de una RNA.

1.2.3. Trabajos relacionados con Redes Neuronales y Cálculo Fraccionario

En la literatura se encuentran una gran variedad de trabajos que relacionan al CF y las RNA. Se estudia la estabilidad de su dinámica, su eficiencia al encontrar parámetros para controladores, configuraciones como controladores PID, etc.

En [15] se implementó una RNA Hopfield (Hopf). El objetivo del trabajo es el demostrar que la RNA es Mittag-Leffler estable mediante el uso de la función de Lyapunov utilizando la derivada de Caputo. La RNA Hopf se muestra en la ecuación (1.1), donde $i = 1, \cdots, n$, $t \geq 0$, n corresponde al número de unidades o neuronas de la RNA, $x_i(t)$ denota al estado i, c_i es una constante, I_i es un vector de entrada el cual es constante, $f_i(\cdot)$ son las funciones de activación no lineales y a_{ij} son pesos sinápticos. La función de Mittag-Leffler generaliza a las funciones exponenciales y suelen tener comportamiento sinusoidales si estas están compuestas por dos parámetros, tal como se muestra en la ecuación (1.2).

$$ {}_{t_0}^{C}\mathcal{D}_t^{\alpha} x_i(t) = -c_i x_i(t) + \sum_{j=1}^{n} a_{ij} f_j(x_j(t)) x_j(t) + I_i, \tag{1.1} $$

$$ E_{\alpha,\beta}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(k\alpha + \beta)}. \tag{1.2} $$

En [16] se propuso un análisis de estabilidad que los mismos autores proponen con el fin de identificar los valores críticos del orden fraccionario donde pueden ocurrir bifurcaciones o comportamiento caótico en estructuras Hopf. También se comprueba que la caracterización de sistemas utilizando un orden fraccionario puede enriquecer el comportamiento de un sistema, es decir, que el modelo se comporte de manera fidedigna a la planta. El análisis de estabilidad se basa en el análisis de los eigenvalores (λ) del sistema mediante la comparación de la parte real por un complemento y la parte imaginaria de los mismos. En [17] se propuso un análisis de estabilidad, en sentido de Lyapunov, para estructuras neuronales de tipo Hopf, proponiendo un análisis en función del orden de la derivada y de la función de Lyapunov con el fin de generalizar la estabilidad de sistemas Hopf. Al igual que los trabajos anteriores, en [18] se analiza la estabilidad de sistemas Hopf, pero mediante el uso de la definición de continuidad de Lipschitz.

En [19, 20] se implementó un controlador PID fraccionario para mejorar la producción de pulpa de papel utilizando una RNA. La RNA se implementa con el fin de estimar los parámetros del controlador PID al igual que los órdenes de la derivada que la acción integral y derivativa necesitan. En [21] se propuso una metodología similar a la propuesta anteriormente para realizar el control de sistemas con PID fraccionarios, también se propone un controlador de orden fraccionario basado en RNAs; la RNA que se propone está compuesta por cuatro unidades en total, tres para la capa de entrada y una para la capa de salida. Las entradas a la RNA corresponden a la derivada fraccionaria de la señal de error y la integral fraccionaria del error en el sentido de Grünwald-Letnikov (GL), los pesos sinápticos que se muestran realizan una analogía con las ganancias de un controlador PID.

Otra de las aplicaciones que se han encontrado para las RNAs y el CF es para resolver ecuaciones diferenciales de orden fraccionario. En [22] se propuso una RNA tipo perceptrón para resolver ecuaciones diferenciales fraccionarias mediante la implementación de un algoritmo Quasi-Newton para la estimación de los pesos sinápticos. En [23–25] se propuso una metodología para resolver ecuaciones diferenciales fraccionarias no lineales y ecuaciones de Riccati mediante una RNA tipo perceptrón mediante algoritmos genéticos

y optimización por enjambres de partículas.

En [26] se utilizó una RNA para estimar el orden de derivada fraccionaria en el sentido de GL para proponer una generalización al filtro de Kalman extendido para sistemas discretos en espacio de estados. El algoritmo que proponen es robusto ante el ruido y es capaz de estimar ciertos parámetros en linea.

1.2.4. Identificación de Sistemas con Cálculo Fraccionario

Otra de las aplicaciones del CF se encuentra en la IS. En [27, 28] se propuso un modelo matemático a partir del comportamiento del circuito de Randles para estimar el comportamiento del estado de carga de una batería ácido de plomo utilizando la TLF en el sentido de RL. La estimación de los parámetros desconocidos se realizó mediante la implementación del algoritmo de Marquardt.

En [29] se propuso el método de mínimos cuadrados y de variable instrumental con el fin de estimar los parámetros de un modelo en función de transferencia tipo *output-error* (OE). El fin del trabajo consiste en estimar el comportamiento de diversos efectos en máquinas de inducción utilizando la definición de GL.

En [30] se realizó la IS mediante una estructura a bloques WH considerando que el bloque W esta construido mediante una función de transferencia OE fraccionaria al igual que el bloque H. El bloque no lineal está construido mediante la suma de funciones de base radial. El objetivo de este trabajo es el de resolver un problema de optimización con el fin de estimar sistemas.

En [31] se propuso un algoritmo para realizar la IS mediante una estructura Hammerstein basado en la definición de GL y utilizando estándares estadísticos para obtener la parametrización del modelo. La parte estática no lineal la realiza mediante una aproximación con polinomios o proponiendo una ecuación. El algoritmo muestra robustez ante el ruido y obteniendo buenos resultados en la estimación.

En [32] se presentó una metodología para realizar la IS en espacio de estados en tiempo discreto en el sentido de GL. Primeramente se realiza la identificación en tiempo continúo mediante una función de transferencia utilizando la TLF con el fin de obtener su similar en tiempo discreto. Después se propone una función de transferencia en tiempo discreto y se estiman los parámetros de la misma con el fin de representarla mediante la matriz de retroalimentación, matriz de entrada y de salida.

1.2.5. Identificación de Sistemas mediante Redes Neuronales y Cálculo Fraccionario

Con anterioridad, se han abordado trabajos acerca de la identificación de sistemas mediante el uso de diferentes técnicas. Hasta ahora, la técnica más utilizada es la identificación de sistemas en función de transferencia fraccionaria. La información acerca de la IS mediante redes neuronales y el CF es limitada.

En [33] se propuso utilizar una derivada fraccionaria discreta para procesar señales o vectores de datos. También se propone utilizar una red neuronal en espacio de estados para realizar la identificación de diversos sistemas. Lo interesante de este trabajo es que se logra obtener la derivada discreta de un vector de datos y que en conjunto con la RNA se

puede realizar la identificación de sistemas de manera satisfactoria y con un número de parámetros pequeño. En ocasiones, la magnitud de cada regresor (n_a, n_b o n_k) suele ser grande haciendo que el proceso de identificación sea tardío.

Utilizando la metodología propuesta en [33], en [34] se realizó el modelado de un ultracapacitor mediante una RNA al igual que el control del mismo. En este trabajo se muestran más datos acerca de la estructura de la RNA.

En [35] se realizó la identificación de sistemas con redes neuronales mediante el uso de una derivada fraccionaria discreta en el sentido de GL. Las entradas a la RNA están constituidas por la derivada fraccionaria discreta de la entrada y de la salida de un proceso de transferencia de calor. Para el experimento que realizan solo es necesario procesar la entrada y salida con dos ordenes de derivada diferentes.

En [36] se propuso una estructura de RNA dinámica en el sentido de RL para la identificación de sistemas en espacio de estados. La adaptación de los pesos sinápticos se realiza mediante un entrenamiento local con el análisis de una función de Lyapunov y con la solución de la ecuación de Riccati.

Se han presentado diversos trabajos con distintas metodologías proponiendo la IS con RNAs utilizando el CF y, como es posible observar, la identificación de sistemas con RNAs de orden fraccionario es un campo limitado y poco explorado.

1.3. Planteamiento del Problema

En el universo existen fenómenos altamente complejos y por ende, difíciles de controlar; en ocasiones se "discretiza" el sistema en subsistemas con el fin de hacer controladores para cada parte del sistema completo. Algunos de estos controladores se diseñan mediante el análisis del comportamiento físico del sistema con el uso de un modelo matemático capaz de cumplir con la tarea de caracterizar al sistema. La problemática recae en **la necesidad de obtener un modelo capaz de describir el comportamiento de sistemas complejos.**

Algunas veces el proceso de modelado matemático es tardío, sensible, tedioso y en ocasiones, la precisión del modelo no caracteriza el comportamiento del sistema real. Por esa razón, se utiliza la identificación de sistemas para proponer un modelo empírico generado por el conocimiento del comportamiento de las señales de entrada y de salida. El modelo empírico puede generarse como un modelo neuronal el cual está compuesto por un **alto número de parámetros de estimación que, en ocasiones, su comportamiento carece de precisión** comparándolo con la respuesta real del sistema. **Los modelos convencionales (estimados mediante técnicas de optimización clásicas), suelen proveer un desempeño aceptable rondando entre el 60 y 80 % de parentesco.**

Por otro lado, el **tiempo de computo del cálculo fraccionario es elevado y su programación es complicada**. Su mayor virtud es, a su vez su mayor defecto ya que el efecto de memoria, realiza una ponderación de cada uno de los datos ponderados para producir un valor futuro.

1.4. Hipótesis

Es posible realizar la identificación de sistemas a sistemas no lineales utilizando redes neuronales artificiales y cálculo fraccionario con el fin de proponer un modelo reducido (en función del número de parámetros) con un alto grado de precisión en la estimación. Y asemejar el comportamiento de sistemas fraccionarios a través del uso de redes neuronales con el fin de reducir el tiempo de computo.

1.5. Justificación

La mayoría de los controladores se diseñan con base en un modelo matemático y este es obtenido mediante dos formas: una de ellas es mediante las leyes físicas que rige al sistema y por último la identificación de sistemas. La IS se puede realizar de diversas maneras y con una gran variedad de modelos propuestos, en este caso, se concentrará en realizar la IS con RNA y CF.

Es normal que surjan diversos cuestionamientos acerca de las aplicaciones del CF debido a que no se cuenta con una interpretación física o geométrica **exacta**. A pesar de ello, el CF ha mostrado diversas ventajas en su aplicación como lo es la representación de sistemas no lineales de orden superior y fenómenos complejos mediante un sistema fraccionario de orden menor y por ende con un número de coeficientes o parámetros reducido. La **no localidad** que ofrece el cálculo fraccionario permite que se puedan explicar fenómenos más complejos de manera más precisa. La no localidad se basa en el **efecto de memoria** que la derivada fraccionaria contiene; este efecto consiste en ponderar el histórico o comportamientos pasados para producir el valor actual y no solo depender de una vecindad cercana a un punto de equilibrio como comúnmente se realiza. En diversos trabajos se muestra que la identificación de sistemas con CF es eficiente [5, 37–41].

Por otro lado, las RNA han mostrado ser útiles en diversos campos de la ciencia gracias a su sencilla implementación, rapidez y adaptabilidad. En este contexto, se pueden utilizar a las RNA para aproximar comportamientos no lineales como se ha propuesto en [10, 23–25, 42–44].

1.6. Objetivos

1.6.1. Objetivo General

Desarrollar algoritmos con el fin de realizar la identificación de sistemas no lineales mediante la implementación de redes neuronales de orden fraccionario.

1.6.2. Objetivos Específicos

1. Diseñar una RNA capaz de ser utilizada mediante el enfoque de CF y realizar la IS con las RNAs diseñadas.

2. Desarrollar algoritmos de aprendizaje fraccionarios y validarlos mediante una comparación con algoritmos de aprendizaje clásicos donde el orden es entero.

3. Implementar la red neuronal fraccionaria junto con el algoritmo de aprendizaje fraccionario.

4. Realizar la identificación de diversos sistemas propuestos por la literatura en función de los siguientes aspectos:

   a) Método para determinar el orden de la red neuronal fraccionaria implementada.

   b) Análisis de la velocidad de convergencia y estabilidad mediante el uso de la función de Lyapunov para cada experimento.

1.7. Metas

- Desarrollar algoritmos de los métodos de integración y derivación fraccionaria, de redes neuronales y de entrenamiento neuronal fraccionario.
- Establecer una metodología para la identificación de sistemas mediante redes neuronales introduciendo el enfoque de cálculo fraccionario.

1.8. Organización del Documento

El documento de Tesis se encuentra distribuido en los siguientes capítulos:

Capítulo 2: Se muestra una descripción general del cálculo fraccionario, funciones especiales utilizadas en el CF y las definiciones de derivada fraccionaria utilizadas para el desarrollo de este trabajo de Investigación.

Capítulo 3: Consta de una explicación general de las redes neuronales artificiales, arquitecturas, entrenamientos, descripción de factores importantes en el aprendizaje y el proceso de identificación con RNAs.

Capítulo 4: Se explica un método de integración numérica utilizando redes neuronales y técnicas de identificación de sistemas para la resolución de sistemas fraccionarios con orden variable, kernel no local y no singular.

Capítulo 5: Se propone un algoritmo de aprendizaje de orden fraccionario, una estructura de RNA basada en CF capaz de realizar la identificación de diversos sistemas de entre los cuales se encuentran retos de identificación de sistemas mecánicos y médicos.

Capítulo 6: Se presenta un algoritmo de aprendizaje fraccionario y estructura neuronal dinámica fraccionaria basada en la estabilidad de Lypunov para la identificación de diversos sistemas.

Capítlo 7: Finalmente, se presentan las conclusiones generales del trabajo de investigación, aportaciones y se plantean posibles investigaciones futuras.

CAPÍTULO 2

Cálculo Fraccionario

El 30 de Septiembre de 1695, el matemático francés Guillaume de L'Hôpital escribió al alemán Gottfried Wilhelm Leibniz preguntando la notación particular que había usado en su publicación para la derivada de n-ésimo orden $\frac{D^n x}{Dx^n}$ de una función lineal $f(x) = x$, . L'Hôpital cuestionó "¿Cuál sería el resultado si $n = 1/2$?", a lo que Leibniz respondió: "Un paradoja aparente, de la cual, un día consecuencias útiles serán escrita". Así es como con estas pequeñas pero indiscutibles frases, el cálculo fraccionario nació.

2.1. Introducción

Como se ha mencionado con anterioridad, Leibniz al proponer su notación de derivada fue cuestionado en diversas ocasiones gracias a la curiosidad que él plantó en muchos matemáticos de la época. En la carta enviada por Leibniz a L'Hôpital, no solo se hizo mención a la "posible paradoja" sino que propuso el posible resultado de la pregunta realizada por L'Hôpital donde expresa que $d^{\frac{1}{2}}x = x\sqrt{dx : x}$ y que a su vez, realizando un cambio de variables $d^e x = x[e]\sqrt{dx/x}$. En Diciembre de 1695, el matemático suizo Johann Bernoulli, preguntó a Leibniz: "De la analogía que ha mostrado, la exponenciación y derivación pueden ser transferidas a series para $d^m \bar{xy}$. Si m es un número fraccionario o irracional, puede por favor usted explicarme, ¿qué sería $d^m \bar{xy}$, una cantidad o algo más? [...]". Leibniz contestó en el mismo mes una carta donde comenzó a utilizar el término"orden general" refiriéndose al orden de derivación no entero. En su respuesta, Leibniz proponía una generalización para el problema puesto por Bernoulli finalizando con "Pienso que esto es memorable, y usted no será desagradecido por ello. Usted ha visto que las derivadas «extraordinarias» pueden ser expresadas por la composición de series geométricas infinitas". En 1697 el matemático inglés John Wallis escribió a Leibniz acerca de una productoria infinita que fuese igual a $\frac{1}{2}$ contestando que "[...] He revelado que aparte de las propiedades conocidas de y, y^2, $y^{\frac{1}{2}}$, etc. o más general y^e, o $p^e y$ y el cambio de estados dy, $d^2 y$ (o ddy), pueden ser definidos también por $d^{\frac{1}{2}}y$ o más general $d^e y$"[45, 46]. Esa fue la última referencia que se tiene acerca de Leibniz y su propuesta del uso de órdenes generalizados.

Al morir Leibniz las derivadas de orden no entero no dejaron de producir cuestionamientos en la comunidad científica. En 1738 el matemático suizo Leonhard Euler propuso arreglos algebraicos con órdenes no enteros en un trabajo de progresiones numéricas [47]. El italiano Joseph-Louis Lagrange en 1977, menciona en su ley de exponentes para operadores diferenciales de orden entero considerando que los órdenes pueden ser complejos [48]. El matemático francés Sylvestre François Lacroix, propuso la solución de derivadas fraccionarias para funciones exponenciales utilizando la función Gamma propuesta por Euler (ver sección 2.2.1) [49]. En 1820 el matemático francés Pierre-Simon Laplace definió una derivada fraccionaria para funciones representada por una integral [50]. En 182, el francés Jean-Baptiste Joseph Fourier, propone la primera definición de una integral fraccionaria como

$$\frac{d^i}{dx^i} fx = \frac{1}{2\pi} \int_{-\infty}^{\infty} d\alpha f\alpha \int_{-\infty}^{\infty} p^i cos\left(p(x-\alpha) + \frac{i\pi}{2}\right) dp, \tag{2.1}$$

donde i es una cantidad cualquiera positiva o negativa y fx puede ser una función cualquiera [51].

Desde 1695 (los orígenes del CF), se habían propuesto diversas definiciones de derivada fraccionaria pero, no fue hasta 1823 cuando el matemático noruego Niels Henrik Abel utiliza al CF para resolver un problema físico; el problema de tautócrona es la determinación de una curva en un plano $\mathbb{R}^2$ de tal manera que se estime el tiempo requerido de una partícula para deslizarse hasta la parte más baja de la curva bajo efectos gravitacionales uniformes considerando que los puntos de partida $(x_0,\, y_0)$ son independientes. La solución propuesta por Abel se basa en igualar la energía potencial y cinética, realizando cambios de variables y operaciones algebraicas se obtiene la ecuación (2.2) [52, 53].

$$k(x) = \int_0^x (x-t)^{-\frac{1}{2}} f(t)dt \Rightarrow f(x) = \frac{k}{\pi\sqrt{x}}. \tag{2.2}$$

Nótese que en la ecuación (2.2) no solamente se encuentra la solución del problema tautócrona sino una solución de una integral fraccionaria con orden $\alpha = 0.5$ [46].

En 1832 el matemático francés Joseph Liouville desarrolló dos diferentes definiciones de derivadas fraccionarias. La primera es aplicada en funciones $f(x)$ las cuales pueden ser expandidas en una combinación lineal infinita entre constantes y funciones exponenciales [54]; nótese que de la definición propuesta, el orden de derivación α está restringido de tal manera que la serie no sea convergente. La segunda definición propuesta por Liouville no tiene ninguna restricción en la selección de α pero sí en el tipo de definición en la que se desea aplicar. Para funciones del tipo $f(x) = \frac{1}{x^a}$, con un parámetro arbitrario se desarrolló la siguiente definición:

$$\mathcal{D}^\alpha x^{-a} = \frac{(-1)^\alpha \Gamma(a+\alpha)}{\Gamma(a)} x^{-a-\alpha}, \tag{2.3}$$

donde $\Gamma(\cdot)$ es la función Gamma de Euler (ver apartado 2.2.1) para cualquier orden fraccionario de derivación α. A pesar de que ambas definiciones contienen restricciones para su aplicación, Liouville utiliza estas definiciones para resolver problemas geométricos, físicos, mecánicos y ecuaciones diferenciales fraccionarias (EDF). En 1847 el alemán Georg Friedrich Bernhard Riemann deduce una definición de derivada fraccionaria buscando una generalización de la serie de Taylor. La combinación de ambas técnicas o definiciones (Liouville y Riemann) produjo lo que hoy se conoce como la primer definición de **integral fraccionaria de Riemann-Liouville**. En 1867 los matemáticos Anton Karl Grünwald y

Aleksey Vasilievich Letnikov propusieron las bases de una definición de derivada fraccionaria, utilizando una idea propuesta por Liouville, la cual se basa en utilizar el límite de diferencias finitas para órdenes fraccionarios. El método se basa en generalizar le definición de derivada clásica propuesta por Newton y reemplazar los elementos n de coeficientes binomiales por un valor no entero naciendo así la definición de **derivada fraccionaria de Grünwald-Letnikov** [55, 56].

Posteriormente, cien años después de las definiciones propuestas por Liouville, Riemann, Grünwald y Letnikov. El matemático italiano Michele Caputo, propuso una definición de derivada fraccionaria la cual, hoy es conocida como **derivada fraccionaria de Caputo**. La definición propuesta por Caputo está altamente conectada por la definición de Riemann-Liouville .

Hasta ahora se ha hablado de 300 años de historia matemática donde sucesos importantes pasaron y las personas que sembraron las bases del cálculo fraccionario a como se le conoce en la actualidad. En este capítulo se hablarán de funciones especiales en el cálculo fraccionario, diversas definiciones de derivadas fraccionarias y su forma de cálculo.

2.2. Funciones especiales

Como se ha mencionado con anterioridad, el cálculo ordinario o de orden entero es un caso particular del CF. Es importante destacar que el CF se encuentra compuesto por diversas funciones cuyo concepto matemático es relativamente simple pero es necesario para su desarrollo. En las siguientes secciones se presentarán funciones especiales utilizadas en el CF.

2.2.1. Función Gamma de Euler

La función Gamma extiende el concepto del factorial de los números complejos. La ecuación (2.4) muestra la definición de la función Gamma.

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt, \, \forall x \in \mathbb{R} \backslash \mathbb{Z}^- . \tag{2.4}$$

Existen diversas propiedades de la función Gamma, por ejemplo, si la parte real del número complejo x es positiva, entonces la integral mostrada en la ecuación (2.4) converge absolutamente ($\sum_{n=0}^{\infty} |a_n| < \infty$). Usando integración por partes se obtiene que

$$\Gamma(x+1) = x\Gamma(x), \tag{2.5}$$

generalizando la relación $n! = n(n-1)!$. Con la evaluación de $\Gamma(1) = 1$ y usando la ecuación (2.5) se deduce que el factorial es un caso particular de la función Gamma como lo muestra la ecuación (2.6).

$$\Gamma(n+1) = n\Gamma(n) = \ldots = n!\Gamma(1) = n!, \, n \in \mathbb{R} \backslash \mathbb{Z}^- . \tag{2.6}$$

La Figura 2.1 muestra el comportamiento de la función Gamma en un intervalo de los número reales.

2.2.2. Función de Mittag-Leffler

La función de Mittag-Leffler (ML) es sumamente importante en el CF debido a que es capaz de generalizar a las funciones exponenciales. Como bien se sabe, las funciones

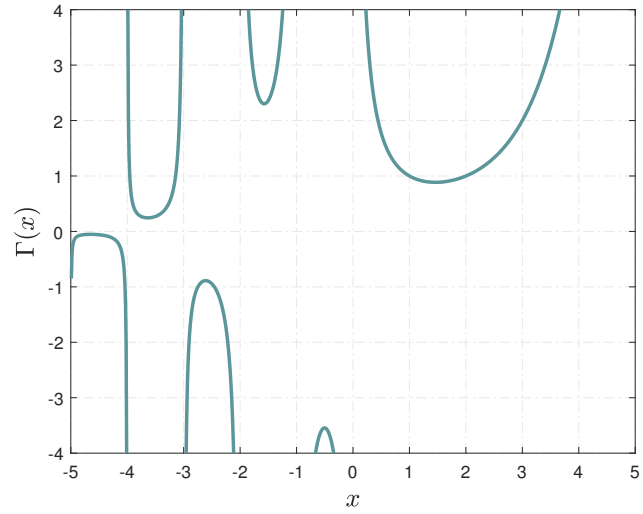


Figura 2.1: Aproximación de la función gamma de Euler.

exponenciales son las principales soluciones de las ecuaciones diferenciales de orden entero por lo que la función de ML es utilizada para obtener las soluciones de ecuaciones diferenciales fraccionarias (EDF). La ecuación (2.7) muestra la función de ML biparamétrica. Nótese que si $\alpha = \beta = 1$ se obtiene la clásica serie de Taylor para representar una función exponencial.

$$E_{\alpha,\beta}(x) = \sum_{k=0}^{\infty} \frac{x^k}{\Gamma(\alpha k + \beta)}, \alpha, \beta \in \mathbb{C},\ \mathbb{R}(\alpha) > 0,\ \mathbb{R}(\beta) > 0,\ x \in \mathbb{C}. \tag{2.7}$$

La Figura 2.2 muestra el comportamiento de la función de ML considerando que el valor de $\beta = 1$ y variando el valor de α. Tal como lo muestra la Figura, si el valor de α y β es igual a uno, se obtiene la clásica función exponencial.

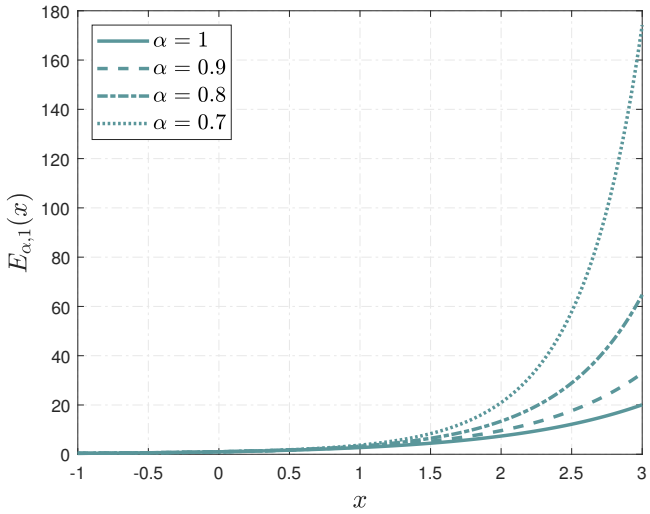


Figura 2.2: Aproximación de la función de Mittag-Leffler.

2.3. Derivadas Fraccionarias

Como se ha mostrado en la introducción de este capítulo (sección 2.1), a diferencia del cálculo ordinario, el CF está compuesto por diversas definiciones de derivada fraccionaria. Cada una cuenta con características y procesos numéricos (para su cálculo) diferentes. Bajo este contexto, se han estipulado ciertos requerimientos que una derivada fraccionaria debe cumplir [57]:

1. La derivada fraccionaria es un operador lineal.
2. La derivada fraccionaria de una función analítica es analítica.
3. Si el orden de derivación es un número entero positivo, se obtiene la derivada clásica de la función.
4. Si el orden de derivación es igual a cero, se recupera la misma función, es decir, ${}_a\mathcal{D}_b^0 f(t) = f(t)$.
5. Para cualquier $n \in \mathbb{N}$ y $\alpha \in (0, 1]$ se tiene que ${}_a\mathcal{D}_b^{\alpha}\left({}_a\mathcal{D}_b^{n} f(t)\right) = {}_a\mathcal{D}_b^{n}\left({}_a\mathcal{D}_b^{\alpha} f(t)\right)$.
6. La derivada fraccionaria tendrá propiedades no locales.

El punto cuatro y cinco hacen mención a la notación de una derivada fraccionaria. Considerando una derivada fraccionaria

$$ {}_{t_0}^{Q}\mathcal{D}_{T}^{\alpha}. \tag{2.8} $$

De la ecuación (2.8), $\mathcal{D}$ representa el operador de derivada fraccionaria, Q representa la definición o sentido de la derivada fraccionaria, t_0, T son los límites de la integral donde se realiza un barrido de los diversos comportamientos del argumento de la integral y α indica el orden fraccionario de la derivada. A lo largo del documento, la notación anterior se seguirá utilizando para representar una derivada fraccionaria.

El último punto se refiere a la capacidad de la derivada fraccionaria para guardar información de sucesos pasados, a este término se le ha definido como **efectos de memoria**. El efecto de memoria de una derivada fraccionaria lo otorga el núcleo o kernel de la derivada. El kernel se encarga de completar una convolución entre él y la función $f(t)$ como se muestra en la ecuación (2.9).

$$ (f * g)(t) = \int_{-\infty}^{\infty} f(\lambda) g(t - \lambda) d\lambda. \tag{2.9} $$

Estadísticamente, al efecto de la convolución utilizando el kernel de la derivada fraccionaria y la función $f(t)$ se le conoce como media móvil ponderada (WMA por sus siglas en inglés weighted moving average). Se le puede definir como una WMA gracias a que el kernel se encarga de ponderar los valores de la función $f(t)$ haciendo que los datos más recientes tienen mayor peso que los datos más antiguos. Se ha reiterado que el CF está formado por diversas definiciones de derivadas fraccionarias, esto debido a que cada derivada fraccionaria tiene un kernel diferente como funciones de potencia, exponenciales, funciones ML, entre otros. En las siguientes secciones se muestran las definiciones de derivadas fraccionarias utilizadas para realizar este trabajo de investigación.

2.3.1. Derivada Fraccionaria de Riemann-Liouville

La formulación común de una integral fraccionaria puede ser derivada directamente de una expresión tradicional de integrar repetitivamente una función. Este enfoque es conocido como el Enfoque de Riemann-Liouville (RL). La ecuación (2.10), muestra la formula propuesta por Cauchy de evaluar la n-ésima integral de una función $f(t)$.

$$\int \int \dots \int_0^t f(\tau)d\tau = \frac{1}{(n-1)!}\int_0^t (t-\tau)^{n-1} f(\tau)d\tau. \tag{2.10}$$

La ecuación (2.11) muestra una simplificación de la expresión (2.10) utilizando propiedades de la función Gamma.

$$\mathcal{J}^\alpha f(t) = f_\alpha(t) = \frac{1}{\Gamma(\alpha)}\int_0^t (t-\tau)^{\alpha-1} f(\tau)d\tau. \tag{2.11}$$

De la integral fraccionaria de RL, se dedujo el proceso inverso proponiendo la definición 1.

Definición 1. *Sea* $f : \mathbb{R} \to \mathbb{R} \times [a,b]$, $b > a$, $\alpha \in (0,1]$, $f \in C^1$ *entonces, la derivada fraccionaria en el sentido de Riemann-Liouville (RL) es:*

$${}_a^{RL}\mathcal{D}_b^\alpha f(t) = \frac{1}{\Gamma(1-\alpha)}\frac{d}{dt}\int_a^b f(\tau)\underbrace{(t-\tau)^{-\alpha}}_{Memoria}d\tau. \tag{2.12}$$

La derivada de RL tiene diversas características; la derivada de una constante es diferente de cero, es una de ellas. Las condiciones iniciales de un sistema o EDF deben ser fraccionarias, esto es, especificar los valores de $f, f^{(\alpha_1)}, f^{(\alpha_2)}, ..., f^{()\alpha_n)}$ cuando $t = 0$. No se tiene una explicación física clara de lo que es una condición inicial fraccionaria, en ocasiones se les define como efectos de subdifusión o pérdidas. Por otro lado se considera que la magnitud de la condición inicial fraccionaria no es medible y por lo tanto desconocida [58].

2.3.2. Derivada Fraccionaria de Grünwald-Letnikov

Opuesto al enfoque propuesto por la integral de RL que se basa en integrar repetitivamente una función, Grünwald y Letnikov propusieron el generalizar el proceso de diferenciación repetitiva basándose en la definición fundamental de la derivada mostrada en la ecuación (2.13).

$$f'(x) = \lim_{h\to 0}\frac{f(x+h)-f(x)}{h}, \tag{2.13}$$

aplicando la fórmula anterior nuevamente se obtiene,

$$\mathcal{D}^2 f(x) = \lim_{h\ to0}\frac{f(x+2h)-2f(x+h)+f(x)}{h^2}, \tag{2.14}$$

derivando n veces la ecuación (2.14) se llega a una expresión en forma de serie como (2.15).

$$\mathcal{D}^n f(x) = \lim_{h\to 0}\frac{1}{h^n}\sum_{m=0}^{n}(-1)^m\binom{n}{m}f(x-mh). \tag{2.15}$$

La expresión (2.15) puede ser generalizada por valores no enteros de n con $\alpha \in \mathbb{R}$ como se muestra en la definición 2 [59].

Definición 2. *Sea* $f : \mathbb{R} \to \mathbb{R} \times \left[t_0, \frac{T-t_0}{h}\right]$, $\alpha(0,1]$, $f \in C^1$ *entonces, la derivada fraccionaria de Grünwald-Letnikov es:*

$$ {}^{GL}_{t_0}\mathcal{D}^{\alpha}_{T} f(t) = \lim_{h\to 0} \frac{1}{h^{\alpha}} \sum_{k=0}^{\left[\frac{T-t_0}{h}\right]} \underbrace{(-1)^k \binom{\alpha}{k}}_{Memoria} f(t-kh). \tag{2.16} $$

2.3.3. Derivada Fraccionaria de Liouville-Caputo

Michele Caputo publicó en [60] una nueva definición de derivada fraccionaria. La **derivada fraccionaria de Caputo** se muestra en la definición 3.

Definición 3. *Sea* $f : \mathbb{R} \to \mathbb{R} \times [a,b]$, $b > a$, $\alpha \in (0,1]$, $f \in C^1$ *entonces, la derivada fraccionaria en el sentido de Liouville-Caputo (LC) es:*

$$ {}^{LC}_{a}\mathcal{D}^{\alpha}_{b} f(t) = \frac{1}{\Gamma(1-\alpha)} \int_a^b \frac{d}{d\tau}\left(f(\tau)\right) \underbrace{(t-\tau)^{-\alpha}}_{Memoria} d\tau. \tag{2.17} $$

Como se observa en la ecuación (2.17), la derivada LC comparte el mismo operador integral de RL. Comparado con la definición de R, la derivada en el sentido de LC tiene un operador diferencial clásico dentro de la integral. Cabe destacar que la derivada de RL y LC comparten el mismo kernel de potencia pero, en la derivada de LC, la derivada de una constante es igual a cero (tal como el caso clásico), y las condiciones iniciales del sistema son enteras (del mismo modo que el caso clásico), es decir, $f(0)$, $f'(0)$, $f''(0)$, $\ldots$, $f^{(n)}(0)$, $f^{(n-1)}(0)$ son conocidas y se tiene un significado físico preciso.

2.3.4. Derivada Fraccionaria de Caputo-Fabrizio

En 2015, los investigadores Caputo y Fabrizio proponen la siguiente definición de derivada fraccionaria:

Definición 4. *Sea* $f : \mathbb{R} \to \mathbb{R} \times [a,b]$, $b > a$, $\alpha \in (0,1]$, $f \in C^1$ *entonces, la derivada fraccionaria en el sentido de Caputo-Fabrizio (CFC) es [61, 62]:*

$$ {}^{CFC}_{a}\mathcal{D}^{\alpha}_{b} f(t) = \frac{M(\alpha)}{1-\alpha} \int_a^b \frac{d}{d\tau}\left(f(\tau)\right) \underbrace{exp\left[-\frac{\alpha(t-\tau)}{1-\alpha}\right]}_{Memoria} d\tau. \tag{2.18} $$

donde $M(\alpha)$ *es una función de normalización de tal manera que* $M(0) = M(1) = 1$.

Nótese que la derivada fraccionaria CFC de una constante es igual a cero al igual que la definición de LC (ver ecuación (2.17)). Las definiciones de RL y LC tienen un problema de singularidad en su kernel de potencia cuando $t \to \tau$, este problema de singularidad es atendido mediante la introducción del kernel exponencial mostrado en la ecuación (2.18).

2.3.5. Derivada Fraccionaria Atangana-Baleanu

En años recientes, el CF ha provocado gran interés gracias a su facilidad de modelar fenómenos físicos del mundo real. Se ha hablado de las bases del CF como de las primeras definiciones de derivadas fraccionarias existentes. En la actualidad se siguen proponiendo diversas definiciones de derivada fraccionaria con núcleos más fuertes (con respecto al

efecto de memoria). En [63] se propone una derivada fraccionaria en el sentido de Caputo, es decir, el operador derivativo clásico se encuentra dentro de la integral como es mostrado en la definición 5.

Definición 5. *Sea $f : \mathbb{R} \to \mathbb{R} \times [a, b]$, $b > a$, $\alpha \in (0, 1]$, $f \in C^1$ entonces, la derivada fraccionaria de Atangana-Baleanu (ABC) en el sentido LC es:*

$$ {}^{ABC}_{a}\mathcal{D}^{\alpha}_{b} f(t) = \frac{B(\alpha)}{1-\alpha} \int_a^b \frac{d}{d\tau} f(\tau) \underbrace{E_\alpha \left[-\alpha \frac{(t-\tau)^\alpha}{1-\alpha} \right]}_{Memoria} d\tau, \tag{2.19} $$

donde $B(\cdot)$ es una función de normalización definida como $B(0) = B(1) = 1$. La ecuación (2.19) está compuesta por una función de ML el cual, es un kernel no local. Tal como la definición de LC (ver definición 3), la derivada ABC tiene las mismas propiedades, es decir, la derivada de una constante es igual a cero y las condiciones iniciales son clásicas. La integral asociada a la ecuación (2.19) se muestra en la definición (6).

Definición 6. *La integral asociada a la derivada fraccionaria con kernel no local es representada como:*

$$ {}^{ABC}_{a}\mathcal{J}^{\alpha}_{b} f(t) = \frac{1-\alpha}{B(\alpha)} f(t) + \frac{\alpha}{B(\alpha)\Gamma(\alpha)} \int_a^b f(\tau)(t-\tau)^{\alpha-1} d\tau. \tag{2.20} $$

2.3.6. Derivada Fraccionaria-Fractal

Últimamente se ha propuesto utilizar el concepto de fractalidad en las derivadas fraccionarias con el fin de observar el comportamiento de la derivada en cualquier escala de tiempo. Cabe destacar que a la derivada fraccionaria con respecto al espacio otorga información de la fractalidad del sistema [64].

Definición 7. *Sea $f(t) \in C^1$ en un dominio $[t_0, T]$, α un orden fraccionario constante tal que $0 < \alpha \leq 1$, β un orden fraccionario constante tal que $0 < \beta \leq 1$ entonces, la derivada fraccionaria de $f(t)$ con orden fraccionario α y una dimensión fractal β está definida como*

$$ \left({}^{FFP}_{t_0}\mathcal{D}^{\alpha,\beta}_{T} f(t) \right) = \frac{1}{\Gamma(1-\alpha)} \underbrace{\frac{d}{dt^\beta}}_{Fractalidad} \int_{t_0}^{T} f(\tau) \underbrace{(t-\tau)^{-\alpha}}_{Memoria} d\tau, \tag{2.21} $$

donde

$$ \frac{d}{dt^{\beta(t)}} \Phi(t) = \lim_{t \to t_1} \frac{\Phi(t) - \Phi(t_1)}{t^{\beta(t)} - t_1^{\beta(t_1)}} = \dot{\Phi}(t) \frac{t_1^{\beta(t_1)}}{\dot{\beta}(t_1) \ln t_1 + \dfrac{\beta(t_1)}{t1}}, \tag{2.22} $$

la ecuación (2.22) muestra el comportamiento para dimensiones fractales variables. Es decir el orden de fractalidad β es una función tal que $\beta : \mathbb{R} \to \mathbb{R}$, $t \mapsto \beta(t)$. Si el valor de β es constante, se recupera la principal propiedad de fractalidad. Al igual que las definiciones de RL y LC, la derivada FFP (2.22) tiene una singularidad cuando $t = \tau$ en consecuencia, en [65] se ha propuesto una derivada fractal-fraccionaria no singular definida por la ecuación (2.23).

Definición 8. *Sea $f(t) \in C^1$ en un dominio $[t_0, T]$, α un orden fraccionario constante tal que $0 < \alpha \leq 1$, β un orden fraccionario constante tal que $0 < \beta \leq 1$ entonces, la derivada fraccionaria*

de $f(t)$ con orden fraccionario α y una dimensión fractal β con un kernel no singular está definido como:

$$\left({}_{t_0}^{FFE}\mathcal{D}_T^{\alpha,\beta} f(t)\right) = \frac{M(\alpha)}{1-\alpha} \underbrace{\frac{d}{dt^\beta}}_{\text{Fractalidad}} \int_{t_0}^{T} f(\tau) \underbrace{exp\left[\frac{-\alpha(t-\tau)}{1-\alpha}\right]}_{\text{Memoria}} d\tau. \tag{2.23}$$

donde $M(\alpha)$ es una función de normalización similar a la definición mostrada en (2.18)

Le definición mostrada en 8 está compuesta por un kernel no singular, es decir evita las indeterminaciones en su formulación. El kernel de la misma, al ser una función exponencial, genera una particularidad, es decir es no local y su ponderación con respecto a la función $f(t)$ es débil. Por lo tanto, en [65] se ha propuesto una definición de derivada fractal-fraccionaria con un kernel no singular y no local como es mostrado en (2.24).

Definición 9. *Sea $f(t) \in C^1$ en un dominio $[t_0, T]$, α un orden fraccionario constante tal que $0 < \alpha \leq 1$, β un orden fraccionario constante tal que $0 < \beta \leq 1$ entonces, la derivada fraccionaria de $f(t)$ con orden fraccionario α y una dimensión fractal β con un kernel no singular y no local está definida como:*

$$\left({}_{t_0}^{FFM}\mathcal{D}_T^{\alpha,\beta} f(t)\right) = \frac{B(\alpha)}{1-\alpha} \underbrace{\frac{d}{dt^\beta}}_{\text{Fractalidad}} \int_{t_0}^{T} f(\tau) \underbrace{E_\alpha\left[\frac{-\alpha(t-\tau)}{1-\alpha}\right]}_{\text{Memoria}} d\tau. \tag{2.24}$$

donde $B(\alpha)$ es una función de normalización denotada por la definición 5.

Cada una de las definiciones fractal-fraccionaria cuentan con una integral asociada. En las ecuación (2.25), (2.26) y (2.27) muestran las integrales asociadas a las derivadas FF mostradas en (2.21), (2.23) y (2.23), respectivamente

$${}_{t_0}^{FFP}\mathcal{J}_T^{\alpha,\beta} f(t) = \frac{1}{\Gamma(\alpha)} \int_{t_0}^{T} (t-\tau)^{\alpha-1} f(\tau) \beta \tau^{\beta-1} d\tau. \tag{2.25}$$

$${}_{t_0}^{FFE}\mathcal{J}_T^{\alpha,\beta} f(t) = \frac{(1-\alpha)\beta}{tM(\alpha)} f(t) + \frac{\alpha}{M(\alpha)} \int_{t_0}^{T} f(\tau) \beta \tau^{\beta-1} d\tau. \tag{2.26}$$

$${}_{t_0}^{FFP}\mathcal{J}_T^{\alpha,\beta} f(t) = \frac{\beta t^{\beta-1}(1-\alpha)}{B(\alpha)} f(t) + \frac{\alpha}{B(\alpha)\Gamma(\alpha)} \int_{t_0}^{T} \beta \tau^{\beta-1} (t-\tau)^{\alpha-1} f(\tau) d\tau. \tag{2.27}$$

Cabe destacar que la prueba de la ecuación (2.22), las ecuaciones (2.25), (2.26), (2.27), la existencia y unicidad de las ecuaciones (2.21), (2.23) y (2.24) se han documentado en [65].

CAPÍTULO 3

Redes Neuronales Artificiales

3.1. Introducción

Las redes neuronales artificiales (RNA) se han convertido en la familia de algoritmos de *machine learning* más populares de los últimos años. Las RNA como modelo comunicacional existen desde varias décadas en el pasado pero no ha sido hasta los últimos años con la mejora de la técnica y tecnología que se han utilizado en diversas ocasiones para desempeñarlas de vastas tareas. Reconocimiento de caracteres, imágenes, voz, predicción bursátil, generador de texto, traducción de idiomas, prevención de fraude, control de manejo, análisis genético, pronóstico de enfermedades, clasificación, etc. Son las aplicaciones que se le han dado a las RNA en los últimos años.

Como suele ocurrir en estructuras avanzadas, la complejidad de estos sistemas emerge por la interacción de elementos más simples trabajando conjuntamente, en el caso de una RNA a cada uno de estos elementos se le denomina **neurona**.

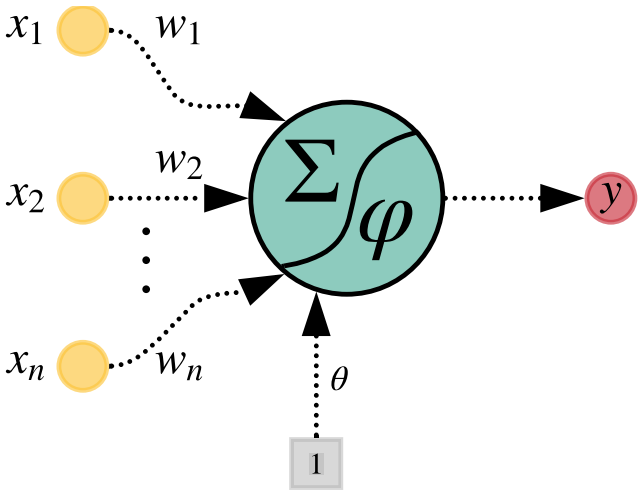


Figura 3.1: Neurona artificial con pesos y bias.

Similar a una neurona biológica, la neurona artificial tiene conexiones de entrada a través de las que recibe estímulos independientes, es decir, son entradas desconocidas ($x_1, x_2, \ldots, x_n$) para la neurona como muestra la Figura 3.1. Con los valores de entrada, la neurona procesará la información y producirá un valor de salida y ponderada por la combinación lineal de las entradas y peso parámetros denominados como pesos sinápticos ($w_1, w_2, \ldots, w_n$). La Figura 3.1 puede ser representada mediante un modelo matemático como (3.1).

$$y = \varphi\left(\sum_{k=1}^{n} w_k x_k + \theta\right), \tag{3.1}$$

donde x son los valores de entrada, w y θ son parámetros de ponderación, φ es una función y y es el valor de salida entregado por la neurona.

El siguiente capítulo muestra el significado de una red neuronal, características de las redes neuronales como su capacidad para aproximar funciones no lineales, el proceso de adaptación que las hacen útiles para la IS y diversas estructuras de red neuronal.

3.2. Redes Neuronales Artificiales

Una RNA es un modelo matemático que imita el funcionamiento del cerebro humano motivado por el hecho de que el cerebro es capaz de calcular y aprender series de patrones [66]. Al igual que el cerebro, una RNA es capaz de completar acciones no lineales, cálculo en paralelo, etc. esto gracias a las diversas arquitecturas de una RNA. El modelo neuronal, está constituido por diversas unidades de procesamiento llamadas neuronas la cual se encarga de almacenar y juntar información de los valores de entrada para producir una señal de salida.

Tal como lo muestra la ecuación (3.1) y la Figura 3.1, la neurona puede mapear no linealidades o mantener su comportamiento lineal mediante una expresión matemática llamada **función de activación**. La neurona biológica se comunica con otras neuronas mediante choques eléctricos, esta comunicación celular se le conoce como sinápsis. Del mismo modo, la neurona artificial es capaz de comunicarse con diversas neuronas artificiales mediante un parámetro ponderado conocido como **peso sináptico**. Los pesos sinápticos son optimizados mediante **algoritmos de entrenamiento**. Gracias a las vastas arquitecturas de RNAs y a su sencilla implementación, las RNAs son capaces de lidiar con diferentes tareas, es decir, sin importar la tarea o propósito, la RNA buscará adaptarse para completar el objetivo por el que fue implementada mediante la adaptación de sus pesos sinápticos con un proceso de entrenamiento [66, 67].

3.3. Arquitectura de Redes Neuronales

Como se ha mencionado con anterioridad, las RNA son estructuras complejas compuestas por una serie de elementos, nodos o unidades denominados neuronas. La Figura 3.1 muestra una neurona artificial la cual está hecha por varios elementos entre ellos está un combinación lineal de parámetros llamados pesos sinápticos y una señal de entrada x_k que es multiplicada por un peso sináptico w_k. Cabe destacar que el dominio de los pesos sinápticos son todos los números reales, es decir, se pueden definir pesos sinápticos positivos como negativos. Otro parámetro de la red neuronal que no es definido como

peso sinápticos sino como un parámetro auxiliar que permite la movilidad de la función de activación es el denominado *bías* (θ). Del mismo modo, una neurona está compuesta por funciones de activación $\varphi(\cdot)$ que permite realizar un mapeo lineal o no lineal tal que $y : \mathbb{R}^n \to \mathbb{R}, (w, x, \varphi, k) \mapsto y(w, x, \varphi, k) : x \in \mathbb{R}^n$.

3.3.1. Funciones de activación

Anteriormente se mencionó que la neurona biológica produce un potencial de acción en función de las señales que haya recibido a través de la sinápsis. Analógicamente, las neuronas artificiales cuentan con una función de activación que permite realizar el mismo fin que la neurona biológica. Con el fin de que la neurona artificial realice el mapeo no lineal, se define una función de activación no lineal y de clase uno, esto es, $\varphi(\cdot) \in C^1$ debido a que se busca que tanto la función de activación como su primera derivada sean continuas. A continuación se presentan algunas de las funciones de activación utilizadas en las RNA:

1. *Función lineal:* también conocida como función identidad es aquella donde el argumento de entrada es igual a la salida (ver Figura 3.2a).

$$\varphi(v) = v. \tag{3.2}$$

2. *Función sigmoidal:* Es la función más utilizada en las capas ocultas de una RNA ya que la función y la derivada de la función son continuas (ver Figura 3.2b).

$$\varphi(v) = \frac{1}{1 + e^{-2v}}. \tag{3.3}$$

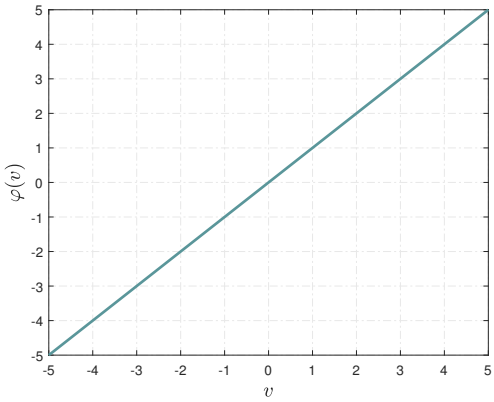
3. *Tangente hiperbólica:* La función $\tanh(\cdot)$, es requerida cuando se busca un rango diferente a la función sigmoidal ya que ésta tiene un centro en el origen y su codominio es $\Xi_v \in \{\varphi(v) : -1 \leq \varphi(v) \leq 1\}$ (ver Figura 3.2c).

$$\varphi(v) = \frac{1 - e^{-2v}}{1 + e^{-2v}} = \tanh(v). \tag{3.4}$$

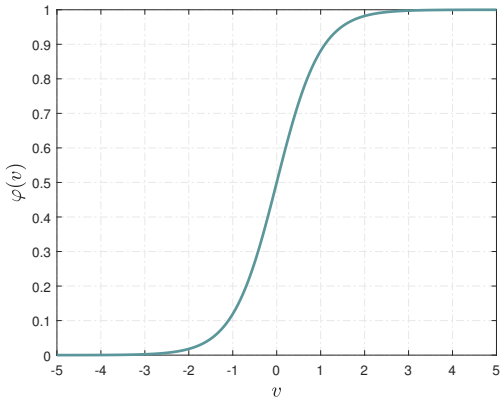
La Figura 3.2 muestra las tres funciones de activación mostradas en las ecuaciones (3.2) - (3.4).

3.3.2. Redes Neuronales Multicapa Realimentadas

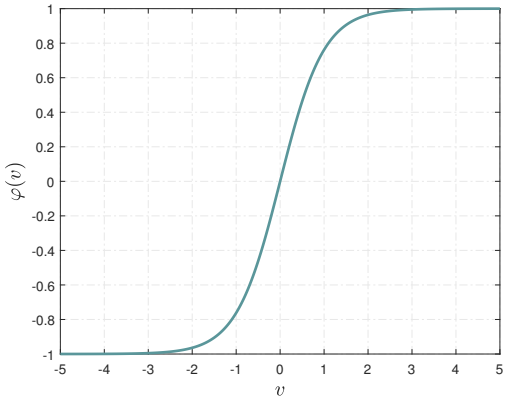
Una red neuronal multicapa realimentada (FFNN por siglas en inglés FeedForward Neural Network) es una de las arquitecturas más utilizadas en diversas aplicaciones [68–72]. En una FFNN las neuronas están organizadas en secciones llamadas capas, es decir, los nodos que se encuentran en la capa de entrada suministran información de los patrones que se desean reproducir (vector de entrada), esta información serán la entrada a la siguiente capa (primer capa oculta) mediante una ponderación realizada por pesos sinápticos. Las salidas de la capa oculta serán las entradas de la segunda capa y las salidas de las neuronas de la segunda capa serán las entradas de la tercera capa y así sucesivamente hasta llegar a la capa de salida. Esta comunicación entre las capas de la red neuronal es con el fin de que la red pueda aprender **conocimiento jerarquizado** [73]. La Figura 3.3 ilustra una RNA de dos capas ocultas con $\mathcal{P}$ neuronas en la primer capa oculta y $\mathcal{Q}$ neuronas en la siguiente capa oculta siendo esta una RNA multicapa realimentada de $\mathcal{P} - \mathcal{Q}$ neuronas, n entradas y q salidas.



(a) Función lineal o identidad.



(b) Función sigmoide.



(c) Función $\tanh(\cdot)$

Figura 3.2: Funciones de activación para neuronas artificiales.

La RNA de la Figura 3.3 se representa matemáticamente como la ecuación (3.5).

$$y_i = \varphi\left(\sum_{p=1}^{\mathcal{P}} v_{ip}\varphi\left(\sum_{j=1}^{n} w_{pj}x_j + \theta_p\right) + \iota_i\right),\ i = 1, 2, \ldots, \mathcal{Q}, \tag{3.5}$$

definiendo z_q como la señal de salida de cada neurona de la primer capa oculta permite que el modelo se simplifique de tal forma que

$$\begin{aligned} z_q &= \varphi\left(\sum_{j=1}^{n} w_{pj}x_j + \theta_p\right),\ p = 1, 2, \ldots, \mathcal{P}, \\ y_i &= \varphi\left(\sum_{p=1}^{\mathcal{P}} v_{ip}z_q + \iota_i\right),\ i = 1, 2, \ldots, \mathcal{Q}. \end{aligned} \tag{3.6}$$

Existen diversas configuraciones/arquitecturas de RNA realimentada donde, la realimentación se realiza desde la primera capa a la última o penúltima capa. Sin importar dónde se encuentre la realimentación, la esencia de una RNA realimentada es que la información siempre se distribuye hacia una sola dirección [73, 74].

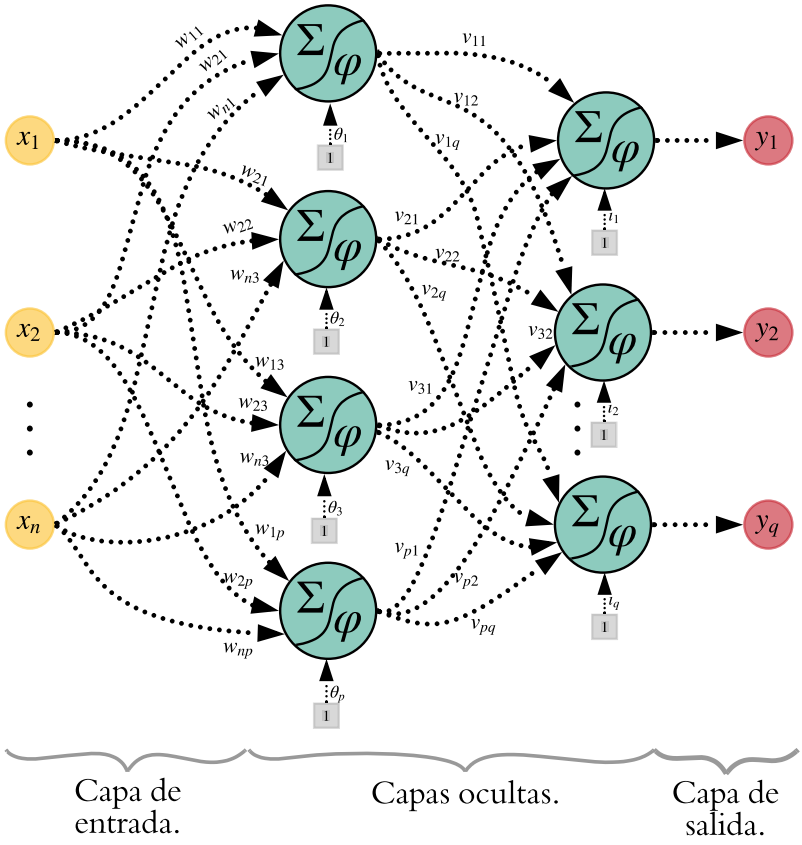


Figura 3.3: RNA multicapa realimentada con $p - q$ neuronas, n entradas y q salidas.

3.3.3. Redes Neuronales Recurrentes

Una red neuronal recurrente (RNR), a diferencia de una RNA realimentada, tiene por lo menos una retroalimentación. La retroalimentación se realiza si la neurona es suministrada mediante la salida de ella misma o si la salida de una neurona es utilizada como entrada de una neurona en una o varias capas atrás. La Figura 3.4 muestra una RNR con tres neuronas en una única capa oculta con tres elementos de regresión o retraso denotados por z^{-1}. Cada salida de las neuronas es retroalimentada y procesada con z^{-1} las cuales serán "entradas" para el nuevo procesamiento. Como se ha mencionado, una retroalimentación puede definirse como la salida de una neurona retroalimentada a sí misma, la Figura 3.4 muestra que la tercera neurona de la capa oculta cumple con dicha definición (línea morada de la Figura 3.4).

El modelo neuronal de la Figura 3.4 se muestra en la ecuación (3.7). La salida de la tercera neurona de la tercera capa se modela matemáticamente en (3.8).

$$y_q = \varphi\left(\sum_{i=1}^{2} w_{iq}x_i + w_{3q}y_1[k-1] + w_{4q}y_2[k-1] + w_{5q}\bar{x}[k-1]\right),\ q = 1, 2. \tag{3.7}$$

$$\bar{x} = \varphi\left(\sum_{i=1}^{2} w_{i3}x_i + w_{33}y_1[k-1] + w_{43}y_2[k-1] + w_{53}\bar{x}[k-1] + w_{63}\bar{x}[k]\right). \tag{3.8}$$

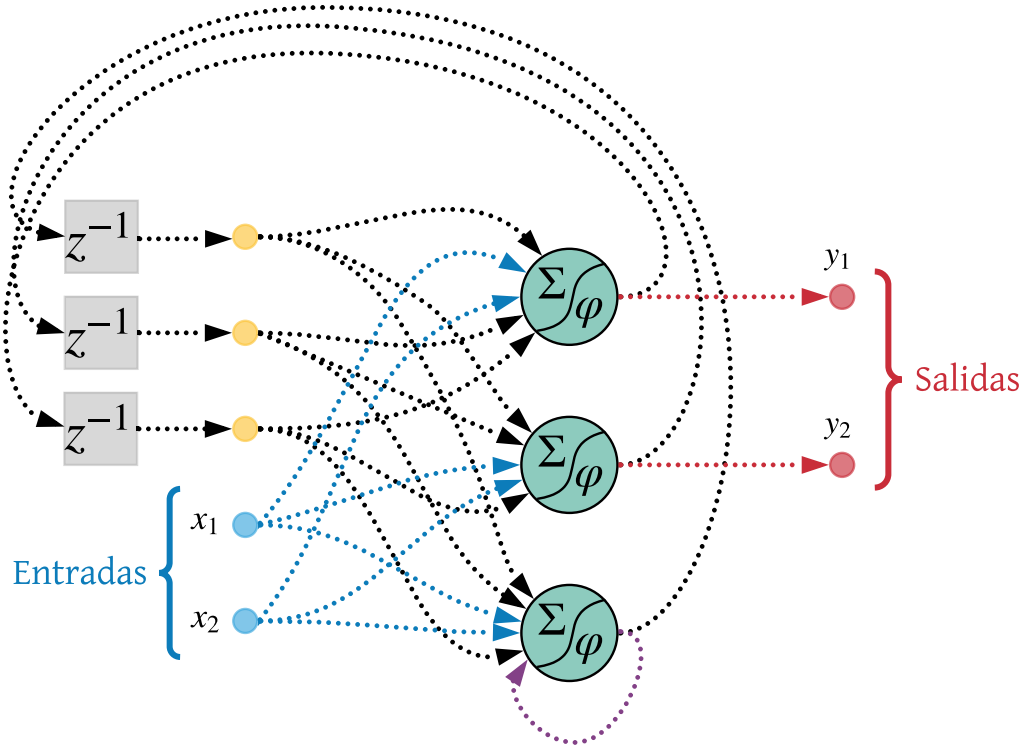


Figura 3.4: RNA recurrente con dos entradas, dos salidas y tres neuronas en la capa oculta. La retroalimentación es realizada mediante el procesamiento de cada una de las salidas mediante z^{-1}.

De las ecuaciones (3.7) y (3.8) se observa que el hecho de añadir retroalimentaciones introduce memoria a la evolución de las neuronas haciendo que la habilidad de aprendizaje cambie de manera dinámica. De esta forma la RNA será capaz de caracterizar no solo comportamientos estáticos (aproximación de funciones) sino que caracterice el comportamiento de un fenómeno variante en el tiempo. Es importante destacar que en la Figura 3.4, cada una de las neuronas no muestra ningún *bías*, esto con el fin de mostrar que no en todos los casos se necesita un *bías* para procesar la información, el hecho de añadir el *bías* queda sujeto al diseñador. Al implementar una RNA, la selección del número de neuronas, *bías*, tipo de función de activación, entrenamiento no es trivial y no existe un método o algoritmo que indique cómo seleccionar estos requerimientos por lo tanto, se realizan diversas pruebas cambiando los aspecots antes mencionando midiendo el **error de aproximación** obtenido.

3.4. Entrenamiento de Redes Neuronales

Sin lugar a dudas, una de las propiedades más importantes de las RNAs es su capacidad de aprender, entrenarse o adaptarse a su entorno mediante el ajuste adecuado de los pesos sinápticos. El **aprendizaje** neuronal se define como el proceso por el cual, los parámetros libres de una RNA son adaptados a través de un proceso de simulación del o por entorno donde la red es implementada [73]. El aprendizaje se realiza de dos maneras:

- **Entrenamiento supervisado:** En el entrenamiento supervisado, el entorno es representado por un conjunto de información de entrada y salida el cual, constituye al conjunto de entrenamiento. La RNA es expuesta al conjunto de entrenamiento y sus

parámetros (pesos sinápticos) son ajustados con base en una señal error, es decir, la diferencia entre la salida de la RNA y la salida deseada. El ajuste continúa iterativamente paso a paso con el fin de hacer que la RNA responda lo más precisamente posible al conjunto de datos de salida del conjunto de entrenamiento.

- **Entrenamiento no supervisado:** El conjunto de entrenamiento es aprendido mediante la continua interacción del entorno o experimento con el fin de minimizar un índice de desempeño muestra por muestra. Minimizando la función objetivo implica que la RNA asigne un valor o característica por cada comparación realizada.

3.5. Algoritmos de Optimización

Considerando una RNA con una capa de entrada para un vector constituido de información $\mathbf{x}(n)$, M capas ocultas y finalmente, una capa de salida con las diversas mediciones obtenidas. Teniendo ahora que,

$$\mathcal{K} = \{\mathbf{x}(k), \mathbf{y}(k)\}_{k=1}^{N}, \tag{3.9}$$

denota una muestra de entrenamiento utilizada para entrenar a la RNA de manera supervisada. Sea $\hat{y}(k)$ la salida producida por la RNA aplicando la información del vector $\mathbf{x}(k)$. La **señal de error** producida se define como

$$e(k) = y(k) - \hat{y}(\mathbf{w}, k),\ \mathbf{w} \in \mathbb{R}^q, \tag{3.10}$$

ahora, considerando una energía de error instantánea definida como

$$\mathcal{E}(\mathbf{w}, k) = \frac{1}{2}e^2(k). \tag{3.11}$$

Se le conoce como función costo u objetivo a la ecuación (3.11) la cual, es continuamente diferenciable con respecto a un parámetro desconocido $\mathbf{w}$. La función $\mathcal{E}(\mathbf{w}, k)$ mapea cada elemento de $\mathbf{w}$ de tal manera que el desempeño sea medible con el fin de encontrar una solución óptima $\mathbf{w}^*$ que satisfaga la condición

$$\mathcal{E}(\mathbf{w}^*, k) \leq \mathcal{E}(\mathbf{w}, k). \tag{3.12}$$

La ecuación (3.12) es el principal problema de optimización estipulado como: *Minimizar la función costo $\mathcal{E}(\mathbf{w})$ con respecto al vector de pesos sinápticos $\mathbf{w}$.*

Una clase de algoritmos de optimización está basada en un descenso iterativo local:

Comenzando con una condición inicial cualquiera denotada por $\mathbf{w}(0)$, se genera una secuencia de vectores de pesos $\mathbf{w}(1)$, $\mathbf{w}(2), \ldots$, de tal manera que la función costo $\mathcal{E}(\mathbf{w})$ es reducida cada iteración del algoritmo como

$$\mathcal{E}(\mathbf{w}[k+1]) < \mathcal{E}(\mathbf{w}[k]), \tag{3.13}$$

donde $[\mathbf{k}]$ es el valor anterior del vector de pesos sinápticos y $[\mathbf{k}+\mathbf{1}]$ es su valor actualizado [73].

Los algoritmos de optimización son utilizados para encontrar los parámetros óptimos de un sistema de tal manera que se cumpla el problema de optimización mostrado en la ecuación (3.11). Otra forma de expresar el problema de optimización aparte de la ecuación (3.12), es mediante la siguiente minimización

$$\mathbf{w}^* = \min_{\mathbf{w}} \mathcal{E}(\mathbf{w}, k), \tag{3.14}$$

donde $\mathbf{w}$ es el vector de pesos sinápticos que conforma (en este caso) la RNA y $\mathcal{E}(\cdot)$ es la función costo. Como se ha mencionado anteriormente, la función costo $\mathcal{E}(\cdot)$ debe ser continuamente diferenciable con respecto al parámetro que se desea optimizar y debe tener un mínimo global [75].

Para garantizar que se cumpla el problema de optimización, la condición

$$\nabla\mathcal{E}(\mathbf{w}^*) = 0, \tag{3.15}$$

debe cumplirse. Donde ∇ es el operador de gradiente de $\mathcal{E}(\mathbf{w})$,

$$\nabla\mathcal{E}(\mathbf{w}) = \left[\frac{\partial\mathcal{E}(\mathbf{w})}{\partial w_1}, \frac{\partial\mathcal{E}(\mathbf{w})}{\partial w_2}, \ldots, \frac{\partial\mathcal{E}(\mathbf{w})}{\partial w_q}\right]^T. \tag{3.16}$$

Cabe destacar que no todos los algoritmos de optimización se basan en métodos del gradiente. Existen diversos algoritmos de optimización que garantizan lo enunciado en la ecuación (3.14) por ejemplo, algoritmo por enjambre de partículas (ver apéndice B), algoritmos genéticos, bayesianos, enjambre de hormigas, Cuckoo search, etc. [76–80].

3.5.1. Métodos Basados en Gradiente

Los métodos basados en gradiente son ajustes sucesivos aplicados al vector de pesos sinápticos $\mathbf{w}$ cuyo ajuste está dirigido de manera descendiente, es decir, de manera opuesta al vector de gradiente $\nabla\mathcal{E}(\mathbf{w}) = g$. Todos los algoritmos de descenso por gradiente pueden ser representados mediante la siguiente ecuación

$$\mathbf{w}[k+1] = \mathbf{w}[k] - \eta[R^{-1}]g[k], \tag{3.17}$$

donde η es una constante positiva llamada tamaño de paso o parámetro de aprendizaje, R modifica el método del algoritmo y $\mathbf{g}[k]$ es el vector gradiente evaluado en el punto $\mathbf{w}[k]$. Usando la definición de incremento $\Delta w = w[k+1] - w[k]$ y ajustando $R = 1$ se tiene que el algoritmo se convierte en una regla corrección-error.

$$\Delta w[k] = -\eta\mathbf{g}[k]. \tag{3.18}$$

Con el fin de demostrar que el algoritmo mostrado en la ecuación (3.18) cumple con las condición de optimización mostrada en (3.13), se utiliza una expansión en una serie de primer orden de Taylor al rededor de $\mathbf{w}[k]$ para aproximar a $\mathcal{E}(\mathbf{w}[k+1])$ como

$$\mathcal{E}(\mathbf{w}[k+1]) \approx \mathcal{E}(\mathbf{w}[k]) + g^T\Delta\mathbf{w}[k], \tag{3.19}$$

utilizando la ecuación (3.18) con pequeños valores positivos de η se tiene que

$$\begin{aligned}\mathcal{E}(w[k+1]) &\approx \mathcal{E}(\mathbf{w}[k]) - \eta\mathbf{g}^T[k]\mathbf{g}[k] \\ &= \mathcal{E}(\mathbf{w}[k]) - \eta||\mathbf{g}||^2\end{aligned}. \tag{3.20}$$

Con la ecuación (3.20) se comprueba que la función costo es decreciente de iteración a iteración. El método del **gradiente descendente** converge a una solución óptima $\mathbf{w}^*$ lentamente si el parámetro η es definido adecuadamente.

3.5.2. Factor de aprendizaje η

La convergencia del algoritmo de GD depende del valor de η de tal manera que

$$\eta[k] = \eta_0, \tag{3.21}$$

donde η_0 es una constante para todo k considerando que para valores elevados de η_0 se corre el riesgo que el algoritmo diverja, bajo este contexto, si η_0 es un valor pequeño, el algoritmo podría encontrar un mínimo global pero, en consecuencia, el tiempo de computo se ve aumentado considerablemente. En [81] se propone un algoritmo llamado *aproximación estocástica* donde se considera que el coeficiente de aprendizaje cambie a través de k como muestra (3.22).

$$\eta[k] = \frac{c}{k}. \tag{3.22}$$

A pesar de que el método (3.22) es eficiente, se corre el riesgo que la estimación del parámetro diverja para valores grandes de c y valores pequeños de k.

Por otro lado, en [82] se propone un algoritmo de **búsqueda y convergencia** utilizando la ecuación (3.23).

$$\eta[k] = \frac{\eta_0}{1 + \frac{k}{\tau}}, \tag{3.23}$$

donde η_0 y τ son constantes seleccionadas por el diseñador. Se debe considerar que para los primeros valores de k, la constante de *tiempo de búsqueda* τ es más grande ocasionando que $\eta[k] = \eta_0$. Si el número de iteraciones k es más grande comparado con la constante de tiempo de búsqueda τ, el factor de de aprendizaje $\eta[k]$ se aproxima a la ecuación (3.22) donde $c = \tau\eta_0$ como es ilustrado en la Figura 3.5 [73].

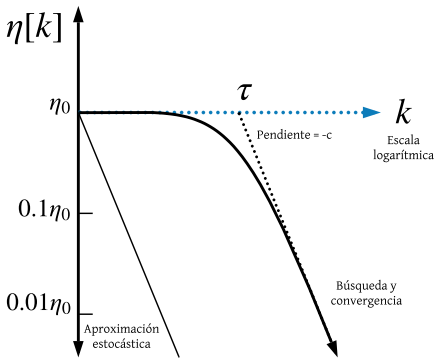


Figura 3.5: Esquemático del comportamiento del factor de aprendizaje $\eta[k]$

3.6. Identificación de Sistemas con Redes Neuronales

Como se ha mencionado, en la IS se utilizan solamente las señales de entrada y salida del sistema para proponer un modelo matemático que caracterice el comportamiento de un fenómeno físico. La Figura 3.6 muestra un esquemático de cómo se realiza la identificación de sistemas con redes neuronales.

Las señales de entrada y salida se procesan mediante retardos (expresados por Z^{-1}) de n_a, n_b y n_k unidades. Estos órdenes se caracterizan en ocasiones por ser el orden del

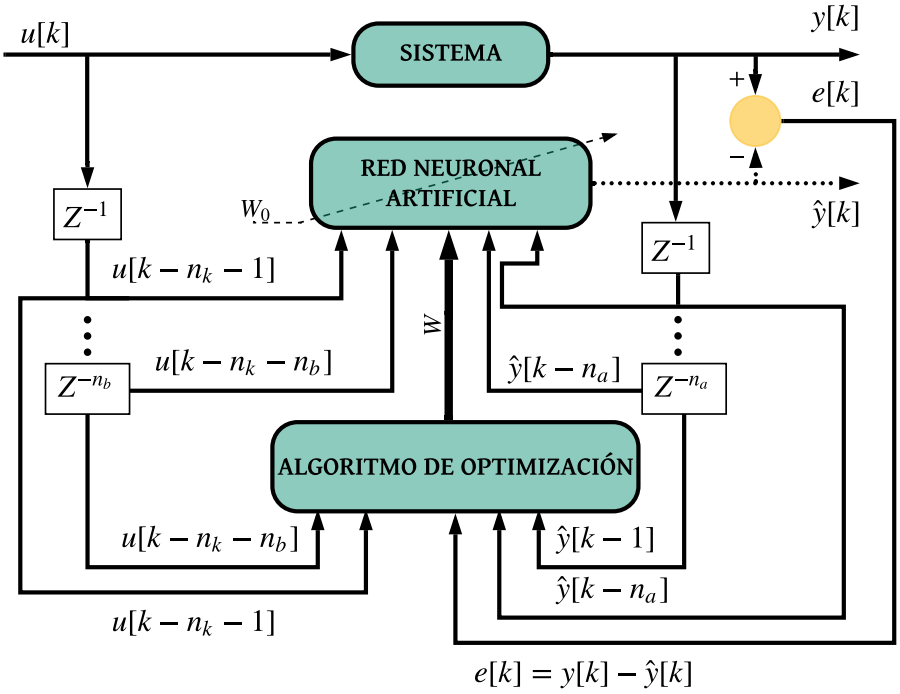


Figura 3.6: Esquema general de la identificación de sistemas con redes neuronales.

modelo propuesto para realizar la identificación y su valor es seccionado empíricamente por el usuario. En este trabajo de investigación se ha propuesto utilizar modelos neuronales para realizar la identificación por lo que el modelo neuronal utiliza las señales de entrada y salida entregadas por el usuario y con ayuda de pesos sinápticos inicializados W_0, la RNA otorga una señal de salida preliminar $\hat{y}[k]$ que es comparada con la señal de salida real $y[k]$ generando una señal de error $e[k]$. La señal de error, junto con las señales de entrada y salida procesadas, suministran información al algoritmo de aprendizaje que, analizando la tendencia o comportamiento del error, ajusta nuevos valores de pesos sinápticos de tal manera que se cumpla el problema de optimización mostrado en la sección 3.5.

CAPÍTULO 4

Redes Neuronales Artificiales para Solución de Sistemas Fraccionarios

4.1. Introducción

Existen diversas técnicas para aproximar la solución de ecuaciones diferenciales fraccionarias (EDF) no lineales. En general, es posible utilizar métodos analíticos o numéricos para resolver una EDF, por ejemplo, método de subecuaciones fraccionarias [83–85], métodos basados en homotopías [86–91], método de descomposición de Adomain [92–94], método de Wevelet [95, 96], transformadas de Laplace [97, 98].

Como se ha mencionado con anterioridad, en el CF el orden de derivación puede ser racional, constante e inclusive una función del tiempo, es decir, de orden variable (OV). La introducción del orden variable ha sido utilizada para describir problemas físicos altamente complejos como encontrar nuevos comportamientos caóticos e hipercaóticos en sistemas que carecen de tales características, encontrar soluciones de las ecuaciones de Schorödinger, análisis de la estabilidad de la ecuación del Telégrafo, entre otros [99–102]. En ocasiones, es complicado encontrar la solución de una EDF a pesar de que el orden de derivación es constante, por lo que el grado de dificultad aumenta cuando se manejan OVs, por lo anterior se proponen métodos numéricos para encontrar las soluciones de las EDFs con OVs [103–105]. Es por ello que gracias a su alta capacidad de aproximar funciones no lineales se ha propuesto utilizar una RNA para aproximar la solución de EDF con OV [106].

En [107] se utiliza una RNA para proponer una solución de EDF no lineal en el sentido LC utilizando algoritmos basados en gradiente descendente. Los resultados obtenidos fueron comparados con la solución analítica de la EDF en el sentido LC mostrando eficiencias prometedoras pero, la metodología no es capaz de conservar la principal característica de una ecuación diferencial, la cual, se basa en respetar las condiciones iniciales que hacen a ésta última única. Es decir, la trayectoria de una ecuación diferencial ordinaria (al igual que fraccionaria) es única gracias a que se establecen condiciones iniciales que deben ser respetadas. En [97] se utiliza una RNA junto con una serie de potencias para encontrar la soluciones de EDF. En [23–25] se utiliza una red neuronal realimentada para encontrar la solución de las ecuaciones diferenciales de Riccati y la ecuación de Bagley-Torkiv en el sentido LC utilizando algoritmos de optimización heurísticos . En [108] proponen una RNA para encontrar la solución de la ecuación integro-diferencial de Volterra-Fredholm

utilizando inteligencia evolutiva para el entrenamiento de la RNA. En [22] se propone una solución analítica compuesta por términos temporales y una RNA para encontrar la solución de EDFs en el sentido LC.

Como se ha mostrado, la mayoría de los trabajos revisados proponen utilizar una RNA para encontrar las soluciones de EDF gracias a su alta capacidad de aproximar funciones no lineales. En ninguno de los trabajos mencionados, se lidia con definiciones de orden variable o con kernels no singulares y no locales (kernel de ML ver ecuación (2.7)). En este capítulo se presenta una metodología basada en identificación de sistemas con RNAs para encontrar la solución de EDF con OV cuyo kernel es no singular y no local.

4.2. Derivada Fraccionaria de Orden Variable con Kernel No Local y No Singular

La definición 5 muestra la estructura matemática de una derivada fraccionaria cuyo kernel carece de localidad y singularidades sin embargo, la ecuación (2.19) está definida para órdenes que son constantes en el tiempo. Por otro lado, la definición 10 muestra una derivada fraccionaria similar a la definida en la ecuación (2.19) considerando al orden como una función variante en el tiempo.

Definición 10. *Sea* $f : \mathbb{R} \to \mathbb{R} \times [a, b]$, $b > a$, $\alpha : \mathbb{R} \to \mathbb{R}$, $t \mapsto \alpha(t)$, $f, \alpha \in C^1$ *entonces, la derivada fraccionaria de Atangana-Baleanu en el sentido LC con orden variable (ABCV) es:*

$$ {}^{ABCV}_{t_0}\mathcal{D}_t^{\alpha(t)} f(t) = \frac{B(\alpha(t))}{1-\alpha(t)} \int_{t_0}^{t} \frac{d}{d\tau} f(\tau) \underbrace{E_{\alpha(t)}\left[-\alpha(t)\frac{(t-\tau)^{\alpha(t)}}{1-\alpha(t)}\right]}_{\textit{Memoria Variable}} d\tau. \tag{4.1} $$

De la ecuación (4.1), la función de normalización $B(\cdot)$ es definida como muestra la ecuación (4.2).

$$ B(\alpha(t)) = 1 - \alpha(t) + \frac{\alpha(t)}{\Gamma(\alpha(t))}. \tag{4.2} $$

Existen diversos esquemas numéricos para mostrar el comportamiento de las derivadas fraccionarias, para la definición ABCV se realizó una modificación al método numérico propuesto en [109] (ver apéndice A).

4.3. Red Neuronal Propuesta para la solución de EDF con OV

La RNA propuesta está constituida por tres capas ocultas, una capa de entrada y una capa de salida. La capa de entrada está compuesta por elementos de procesamiento (z^{-1}) para las señales de entrada y salida del sistema fraccionario. La primer capa oculta está compuesta por dos neuronas al igual que la segunda capa oculta. La tercer capa oculta está compuesta por una neurona con el fin de combinar los efectos compuestos por las señales de entrada y salida del modelo fraccionario. Finalmente, la RNA solamente cuenta con una ramificación que denota a la única salida de la RNA. La Figura 4.1 muestra el esquemático de lo antes descrito al igual que el nombre de cada peso sináptico que conforma a la RNA.

El modelo neuronal de la Figura 4.1, es mostrado en la ecuación (4.3).

$$ \hat{y}[k] = X\varphi_3\left(z_t\varphi_2\left(v_t\varphi_1\left(J_t w_t + \theta_t\right) + v_{th}\right) + z_{\hat{y}}\varphi_2\left(v_{\hat{y}}\varphi_1\left(J_{\hat{y}} w_{\hat{y}} + \theta_{\hat{y}}\right) + v_{\hat{y}h}\right) + z_h\right) \tag{4.3} $$

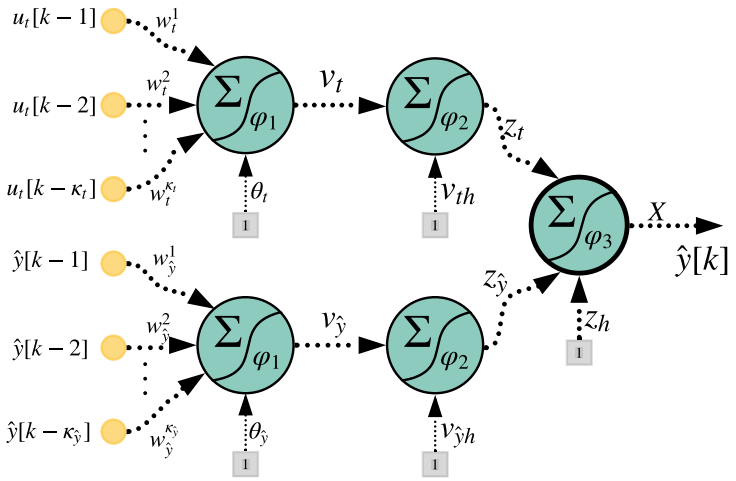


Figura 4.1: RNA propuesta para la solución de EDF con kernel no local, no singular y orden variable .

donde las funciones φ_i $(i = 1, 2, 3)$ son funciones de activación y su selección depende del usuario. X, v_t, $v_{\hat{y}}$, θ_t, $\theta_{\hat{y}} \in \mathbb{R}$, $W_t \in \mathbb{R}^{\kappa_t}$ y $W_{\hat{y}} \in \mathbb{R}^{\kappa_{\hat{y}}}$ son pesos sinápticos de la RNA. $J_t = [u_t(k-1),\, u_t(k-2), \ldots,\, u_t(k-\kappa_t)] \in \mathbb{R}^{N\times\kappa_t}$ y $J_{\hat{y}} = [\hat{y}(k-1),\, \hat{y}(k-2), \ldots,\, \hat{y}(k-\kappa_{\hat{y}})] \in \mathbb{R}^{N\times\kappa_{\hat{y}}}$ son matrices de regresión procesadas por la introducción de retardos z^{-1}. A diferencia de [22], en [110, 111] se presenta una solución general para resolver FDE en el sentido ABCV. Cabe destacar que dicha solución general considera el cambio del orden de derivación a través del tiempo satisfaciendo la condición inicial y_0 de la EDF.

4.4. Solución de EDFOV con RNA

Para encontrar la solución de la FDE se propone suponer que existe una solución $\hat{y}_N(t, \mathbf{\Omega})$ (ver ecuación (4.4)) que satisface la condición inicial del sistema fraccionario, donde $\mathbf{\Omega}$ es el vector de pesos sinápticos.

$$ {}_{t_0}^{ABCV}\mathcal{D}_t^{\alpha(t)} y(t) = f(t,y)|_{t=t_0,\, y=y_0} = y_0. \tag{4.4} $$

La ecuación (4.5), es la solución general propuesta para encontrar el comportamiento del sistema fraccionario.

$$ \hat{y}_N = y_0 + \sum_{j=1}^{\lceil \alpha(t)-1 \rceil} t^j + t^{\lceil \alpha(t) \rceil} \hat{y}(t, \mathbf{\Omega}), \tag{4.5} $$

donde $\lceil \alpha(t) \rceil$ es el valor redondeado hacia el número entero mayor de $\alpha(t)$. Es claro observar que cuando $t = 0$, la solución (4.5) cumple con la condición inicial del sistema fraccionario. Reemplazando la ecuación (4.5) en la ecuación (4.4) se tiene que

$$ {}_{t_0}^{ABCV}\mathcal{D}_t^{\alpha(t)} \hat{y}_N(t, \mathbf{\Omega}) = f(t, \hat{y}_N(t, \mathbf{\Omega})). \tag{4.6} $$

Definiendo una variable de error mediante la comparación de ${}_{t_0}^{ABCV}\mathcal{D}_t^{\alpha(t)} \hat{y}_N(t, \mathbf{\Omega})$ y $f(t, \hat{y}_N(t, \mathbf{\Omega}))$ se obtiene un problema de optimización como la ecuación (4.7).

$$\mathbf{\Omega}^* = \min_{\mathbf{\Omega}} \left\{ \frac{1}{2} \sum_{i=1}^{N} e_i^2 \right\}, \tag{4.7}$$

$e := {}_{t_0}^{ABCV}\mathcal{D}_t^{\alpha(t)} \hat{y}_N(t, \mathbf{\Omega}) - f(t, \hat{y}_N(t, \mathbf{\Omega}))$. Para encontrar la solución de (4.6) es necesario utilizar la definición mostrada en la ecuación (4.1), esto es

$${}_{t_0}^{ABCV}\mathcal{D}_t^{\alpha(t)} \hat{y}_N(t, \mathbf{\Omega}) = \frac{B(\alpha(t))}{1-\alpha(t)} \int_{t_0}^{t} \frac{d}{d\tau} \hat{y}_N(\tau, \mathbf{\Omega}) E_{\alpha(t)} \left[-\alpha(t) \frac{(t-\tau)^{\alpha(t)}}{1-\alpha(t)} \right] d\tau. \tag{4.8}$$

La integral mostrada en (4.8) se calcula con cualquier método de integración numérico, por ejemplo, utilizando el método del trapecio para realizar la integración, se define por simplicidad que

$$G(\alpha(\tau), \mathbf{\Omega}, \tau) = \dot{\hat{y}}_N(\tau, \mathbf{\Omega}) E_{\alpha(\tau)} \left[-\alpha(\tau) \frac{(t-\tau)^{\alpha(\tau)}}{1-\alpha(\tau)} \right], \tag{4.9}$$

en consecuencia, la ecuación (4.8) se reescribe como

$$f_G(t, \mathbf{\Omega}) = \frac{B(\alpha(t))}{1-\alpha(t)} \int_{t_0}^{t} G(\alpha(\tau), \mathbf{\Omega}, \tau) d\tau. \tag{4.10}$$

Una vez simplificada la expresión, se aplica el método del trapecio con altura $h = \frac{t}{N}$ considerando que t es el tiempo total de la simulación y N el número total de muestras para aproximar la expresión mostrada en la ecuación (4.10).

$$f_G(t, \mathbf{\Omega}) \approx \frac{hB(\alpha(\tau))}{2(1-\alpha(t))} \left[G(\alpha(0), \mathbf{\Omega}, 0),\ G(\alpha(h), \mathbf{\Omega}, h), \ldots, G(\alpha(t), \mathbf{\Omega}, t) \right]. \tag{4.11}$$

De acuerdo a la ecuación (4.7) y la ecuación (4.11), se define un nuevo problema de optimización como lo muestra la ecuación (4.12).

$$\mathbf{\Omega}^* = \min_{\mathbf{\Omega}} \left\{ \frac{1}{2} \sum_{k=1}^{N} f_G(t_k, \mathbf{\Omega}) - f(t_k, \hat{y}_N(t_k, \mathbf{\Omega})) \right\}. \tag{4.12}$$

El ajuste de los pesos sinápticos se realizaron con el método basado en gradiente descendente mostrado en la sección 3.5.1 del capítulo 2.

4.5. Resultados

En esta sección se presentan los resultados obtenidos de la propuesta planteada en secciones anteriores.

Recapitulando, se ha propuesto la RNA mostrada en la Figura 4.1 y modelo (4.3). La RNA propuesta cuenta con cinco neuronas en total distribuidas bajo una arquitectura *FeedForward* (realimentada) 2-2-1. Las primeras dos neuronas en la primer capa de entrada son para procesar las señales de entrada y salida del sistema fraccionario. La señal de entrada u_t está compuesta por un vector de tiempo, esto debido a que se desea que la solución neuronal propuesta esté en función del tiempo. Si las funciones de activación son definidas como funciones diferentes a la función lineal o identidad mostrada en la Figura 3.2a, la RNA está constituida por $\kappa_{\hat{y}} + \kappa_t + 10$ pesos sinápticos sino, la RNA podría

ser reducida mediante arreglos algebraicos que, en consecuencia, harán que el número de parámetros de la RNA sea reducido por debajo de $\kappa_{\hat{y}} + \kappa_t + 10$. Para los siguientes ejemplos se definió a $\varphi_2(z) = \varphi_3(z) = z$ y $\varphi_1(z) = \tanh(z)$ por lo tanto, se tiene un total de $\kappa_{\hat{y}}+\kappa_t+5$ pesos sinápticos para realizar la estimación del comportamiento de la EDFOV.

Con el fin de cuantificar los resultados obtenidos, se añaden índices de desempeño con base a la norma de la comparación entre la trayectoria de respuesta de la EDF y la RNA, así como medidas de parentesco entre las dos respuestas (FIT), ambos índices se muestran en las ecuaciones (4.13) y (4.14) respectivamente.

$$||e|| = \sqrt{\sum_{i=1}^{N} e_i^2}. \tag{4.13}$$

$$FIT = 100 * \left(1 - \frac{||y - \hat{y}||}{y - \bar{y}}\right) \%, \quad \bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i. \tag{4.14}$$

También se consideró un parámetro comparativo relacionado con el costo computacional (TS) el cual, es una de las principales ventajas del método propuesto ya que el hecho de simular una RNA consume menos tiempo de computo que la implementación de un método numérico para la solución de EDF. Cabe destacar que el valor TS es el valor promedio de 1000 repeticiones bajo las mismas condiciones, es decir, la computadora encargada de calcular el proceso no contaba con procesos en segundo plano, ni ejecutaba otro programa diferente al MATLAB 2018b

4.5.1. Ejemplo 1

Se considera la siguiente EDF en el sentido ABC con orden variable

$${}_{t_0}^{ABCOV}\mathcal{D}_t^{\alpha(t)} x(t) = -x(t), \quad x(0) = 1,\ t \in [0, 1]. \tag{4.15}$$

Es claro que la solución de la ecuación diferencial cuando $\alpha = 1$ es

$$x(t) = E_1(-t) = \sum_{k=0}^{\infty} \frac{(-t)^k}{\Gamma(k+1)} = e^{-t}. \tag{4.16}$$

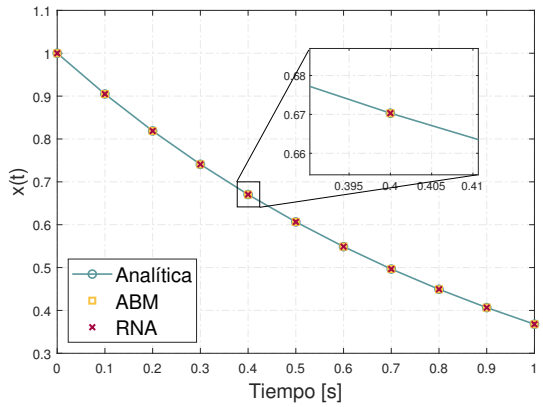
Pero, si α es una función del tiempo entonces la solución general de la ecuación (4.15) es

$$x(t) = E_{\alpha(t)}(-t^{\alpha(t)}). \tag{4.17}$$

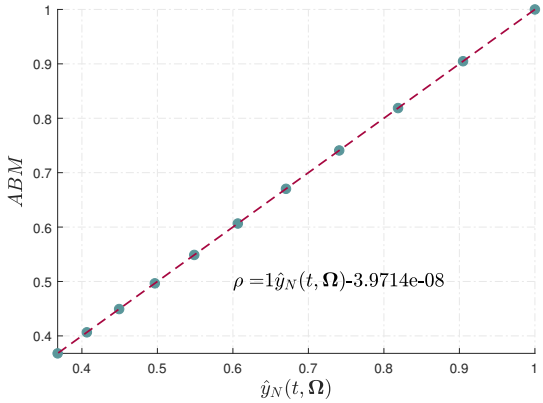
Considerando la ecuación (4.5), la solución neuronal propuesta es

$$\hat{y}_N = 1 + t\hat{y}(t, \mathbf{\Omega}). \tag{4.18}$$

Se desarrolló el algoritmo de Adams-Bashfort-Multon (ABM) [109, 112, 113] para encontrar la solución numérica de la ecuación (4.15). La Figura 4.2 muestra el comportamiento de la solución analítica de la ecuación (4.16), el algoritmo de ABM y la solución neuronal de la ecuación (4.18). Como se ha mencionado con anterioridad, encontrar la solución analítica de las EDF es de alta complejidad, dicha complejidad aumenta si se considera que el orden de derivación es variable en el tiempo, es por ello que se recurre al uso de soluciones numéricas. **La Figura 4.2a muestra que el algoritmo de ABM es capaz de asemejar el comportamiento analítico de la EDF, es por ello que se tomará**



(a) Comparación de la solución analítica, el algoritmo ABM y el método propuesto con RNA.



(b) Prueba de pendiente-intercepto para la comparativa entre el método numérico ABM y la solución neuronal.

Figura 4.2: Solución de la ecuación diferencial fraccionaria de la ecuación (4.15) y prueba de pendiente intercepto de la RNA contra el algoritmo ABM.

como referencia lo otorgado por el algoritmo ABM para medir la eficiencia del método propuesto con la RNA para futuros ejemplos. La Figura 4.2b muestra la prueba de pendiente intercepto entre el algoritmo ABM y la solución neuronal, cabe destacar que para que el modelo sea aceptable, la pendiente de la prueba debe ser cercana a uno y su intercepto cercano a cero.

De acuerdo a la selección de funciones de activación descrita con anterioridad, la red neuronal (4.3) se reduce a la ecuación (4.19).

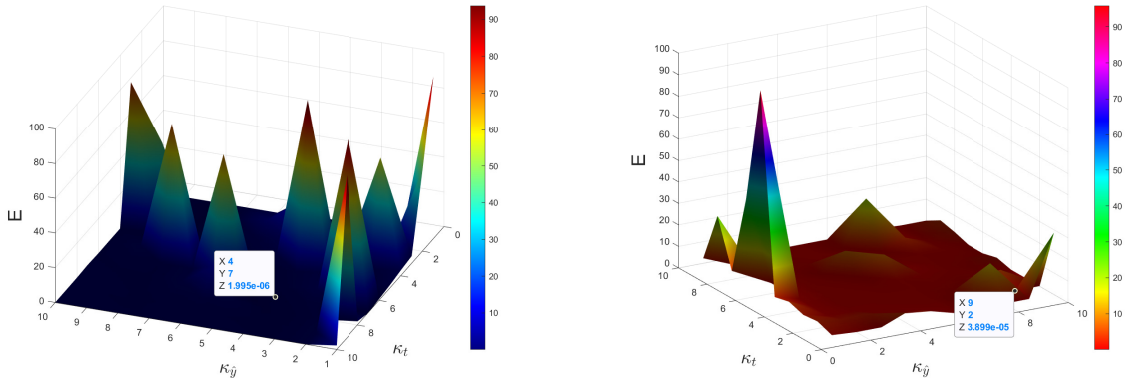
$$\hat{y}[k] = V_A \tanh(J_{\hat{y}} w_{\hat{y}} + w_{\hat{y}h}) + V_B \tanh(J_t w_t + w_{th}) + V_H \tag{4.19}$$

donde $V_A = X z_{\hat{y}} v_{\hat{y}}$, $V_B = X z_t v_t$ y $V_H = X(z_{\hat{y}} v_{\hat{y}h} + z_t v_{th} + z_h)$. Para estimar los parámetros $\kappa_{\hat{y}}$ y κ_t se utilizó un algoritmo de fuerza bruta, es decir, se propuso un conjunto de valores para $\kappa_{\hat{y}} = \{1, 2, 3, ..., 10\}$ y $\kappa_t = \{1, 2, 3, ..., 10\}$ realizando la combinación de cada uno de sus valores. En consecuencia, de acuerdo al conjunto de valores propuesto, se tendrán 100 modelos diferentes y se utiliza el modelo con mejor desempeño evaluando la función costo de la ecuación (4.20).

$$E = \frac{1}{2} \sum_{k=1}^{N} e_k^2. \tag{4.20}$$

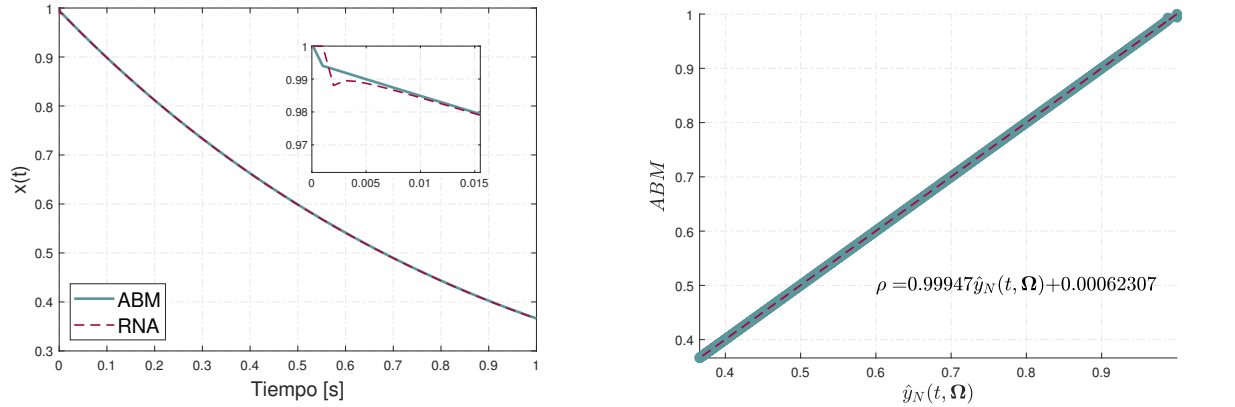
De acuerdo a la Figura 4.3a se seleccionan los valores de $\kappa_{\hat{y}} = 4$ y $\kappa_t = 7$. Con esos valores de regresión seleccionados, se estiman un total de 16 parámetros de la RNA con valores $V_A = 6.48165$, $V_B = -10.69489 \times 10^{-3}$, $V_H = 0.071811$, $w_{\hat{y}} = [0.0002,\ -0.1548,\ 0.0002,\ 0.0029]^T$, $w_t = [0.4587,\ 0.0777,\ -0.4735,\ 0.6384,\ 0.6168,\ -0.8126,\ 0.6130]^T$, $w_{\hat{y}h} = -0.094$ y $w_{th} = 0.3888$.

Se realizó la estimación de la solución de la EDF considerando ahora un orden variable como se había propuesto desde un principio. Se definió $\alpha = \tanh(3 - t)$ con un tiempo de simulación de un segundo. Se realizó el algoritmo de fuerza bruta para estimar los órdenes $\kappa_{\hat{y}}$ y κ_t como se observa en la Figura 4.3b dando como resultado $\kappa_{\hat{y}} = 9$



(a) EDF con $\alpha = 1$ (b) EDF con $\alpha = \tanh(3 - t)$

Figura 4.3: Resultado del algoritmo de fuerza bruta con $\alpha = 1$ y $\alpha = \tanh(3 - t)$



(a) Solución de la EDF con algoritmo ABM y RNA. (b) Prueba de pendiente-intercepto del ABM y RNA.

Figura 4.4: Solución de la EDF con orden variable $\alpha(t) = \tanh(3 - t)$.

y $\kappa_t = 2$ dando un total de 16 parámetros estimados (apéndice D para observar el algoritmo de fuerza bruta). Los valores de pesos sinápticos son $V_A = 0.529943$, $V_B = -0.55929$, $V_H = -0.7854$, $w_{th} = -0.57088$, $w_{\hat{y}h} = 1.7172$, $w_t = [-1.2265, 0.1984]^T$ y $w_{\hat{y}} = [-0.31, 3.55, -0.0035, -0.0013, -0.00069, -0.00043, -0.0003, -0.00022, -0.00188]^T$. La Figura 4.4a muestra la solución de la EDF con orden variable utilizando el algoritmo ABM como referencia y utilizando a la RNA como segunda alternativa para la solución de la EDF; se realizó un acercamiento para denotar que al igual que el ABM, la RNA es capaz de respetar la condición inicial de la EDF. La Figura 4.4b muestra la prueba de pendiente intercepto entre el ABM y la RNA donde la pendiente es relativamente igual a uno y el intercepto se encuentra cercano a cero que es lo que se busca para decir que el modelo o en este caso solución, es aceptable.

La Tabla 4.1 muestra el desempeño otorgado por la RNA para encontrar la solución de la ecuación diferencial mostrada en la ecuación (4.15). Cabe recalcar que los índices fueron obtenidos mediante la comparación del resultado del método numérico ABM como solución de referencia. Como se ha mencionado con anterioridad, se calcularon al rededor de 100 modelos con distintos valores de regresión $\kappa_{\hat{y}}$ y κ_t con el fin de obtener el valor óptimo de dichos regresores. Para ambos casos presentados (cos $\alpha(t) = 1$ y $\alpha(t) = \tanh(3-t)$), los resultados obtenidos son más que prometedores ya que se alcanzan

Tabla 4.1: Eficiencia de la red neuronal para resolver la EDF (4.15).

	$\alpha = 1$	$\alpha = \tanh(3-t)$
Pendiente	1	0.99947
Intercepto	0	0.000623
FIT	$99.996621\,\%$	$99.93\,\%$
$\|\|e\|\|$	2.2345×10^{-7}	0.0129
TS	$9.95115 \times 10^{-4}s$	$9.7752 \times 10^{-4}s$

eficiencias de tal manera que ambas soluciones parecen idénticas. Observando el FIT para ambos casos, son valores cercanos al 100 % que es el caso ideal. La norma del error es relativamente cero para $\alpha(t) = 1$ y muy cercana a cero con $\alpha(t) = \tanh(3-t)$ debido a la alta complejidad que un sistema fraccionario presenta cuando su orden es variable. También se realizó la prueba de pendiente intercepto donde en ambos casos, la pendiente es igual o muy cercana a uno que es el caso ideal y, por otro lado, el intercepto es igual y cercano a cero con respecto a los dos casos presentados. El TS mostrado en la Tabla 4.1 representa el tiempo de simulación utilizado para terminar el proceso de cálculo. Para comparar la eficiencia de la RNA con respecto al algoritmo ABM utilizando el valor TS, se realizó la misma prueba evaluando 1000 veces el mismo experimento y se obtuvo el promedio para ambos casos dando como resultado $0.0173s$ y $0.333811s$ respectivamente. A pesar de que el proceso computacional del algoritmo ABM es pequeño, no se compara con el costo computacional de la RNA.

4.5.2. Ejemplo 2

Ahora se considera el sistema mostrado en la ecuación (4.21) [110]

$$ {}_{t_0}^{ABCV}\mathcal{D}_t^{\alpha(t)} x(t) = g(t) - x^2(t) \tag{4.21} $$

donde,

$$ \begin{aligned} g(t) =& (2t - 3t^2 + t^3) + \left(2t\left(1 - \alpha(t) + \frac{\alpha(t)t^{\alpha(t)}}{\Gamma(\alpha(t)+2)}\right)\right. \\ & \left. -3t^2\left(1 - \alpha(t) + \frac{2\alpha(t)t^{\alpha(t)}}{\Gamma(\alpha(t)+3)}\right) + t^3\left(1 - \alpha(t)\frac{6\alpha(t)t^{\alpha(t)}}{\Gamma(\alpha(t)+4)}\right)\right), \end{aligned} \tag{4.22} $$

con condición inicial $x(0) = 0$. La solución analítica de la ecuación diferencial está dada por la ecuación (4.23).

$$ \begin{aligned} x(t) =& 2t\left(1 - \alpha(t) + \frac{\alpha(t)t^{\alpha(t)}}{\Gamma(\alpha(t)+2)}\right) - 3t^2\left(1 - \alpha(t) + \frac{2\alpha(t)t^{\alpha(t)}}{\Gamma(\alpha(t)+3)}\right) \\ & + t^3\left(1 - \alpha(t) + \frac{6\alpha(t)t^{\alpha(t)}}{\Gamma(\alpha(t)+4)}\right). \end{aligned} \tag{4.23} $$

Considerando la ecuación (4.5), la solución neuronal está dada por la ecuación (4.24).

$$ \hat{y}_N = t\hat{y}(t, \mathbf{\Omega}). \tag{4.24} $$

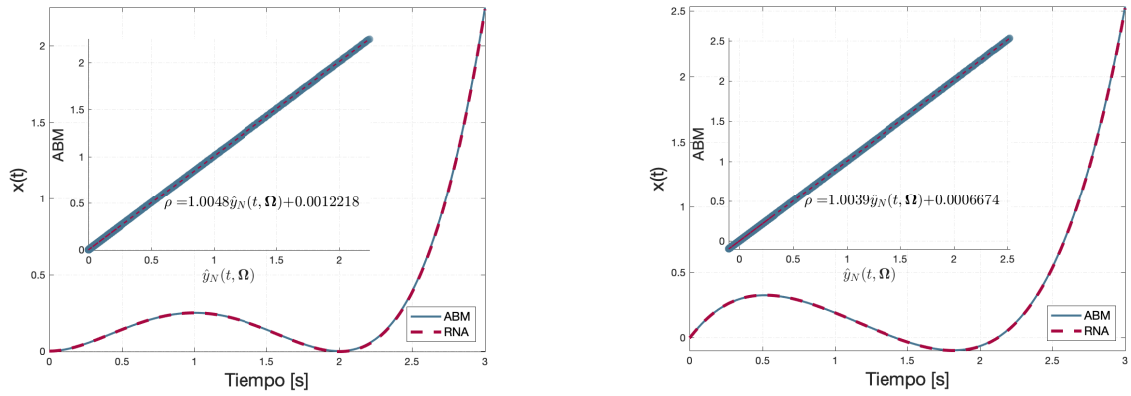
La Figura 4.5 muestra la solución de la ecuación diferencial con su aproximación mediante el algoritmo ABM con orden constante $\alpha = 0.95$ (ver 4.5a) y orden variable $\alpha(t) = \frac{1}{1+e^{-t}}$ (ver 4.5b). También, la Figura 4.5 muestra las pruebas de pendiente-intercepto bajo la comparación del comportamiento entregado por la RNA y el algoritmo ABM.

Con el fin de mostrar que el hecho de simular el comportamiento de la EDF por medio de la RNA requiere menos costo computacional, se realizó la prueba de exigencia computacional antes mencionada. Al igual que el costo computacional, la Tabla 4.2 muestra el FIT, la norma del error y los resultados de la prueba de pendiente intercepto.

Tabla 4.2: Desempeño otorgado por la RNA. El desempeño fue medido mediante la solución entregada por el algoritmo de ABM.

	$\alpha = 0.95$	$\alpha(t) = \frac{1}{1+e^{-t}}$
Pendiente	1.0048	1.0039
Intercepto	0.001221	0.000667
FIT	99.11724	99.37232
$\|\|e\|\|$	0.2161	0.1859
TS	4.2569×10^{-4}	4.3697×10^{-4}

Al simular el algoritmo de ABM en 1000 ocasiones, se obtuvo un tiempo de computo promedio de $0.01112s$ y $0.0094s$ para $\alpha = 0.95$ y $\alpha(t) = \frac{1}{1+e^{-t}}$ respectivamente. Comparando el tiempo de computo del algoritmo ABM con la RNA (mostrado en la Tabla 4.2) se puede concluir que la diferencia es amplia.



(a) Solución analítica y solución por la RNA con $\alpha = 0.95$

(b) Solución analítica y solución por la RNA con $\alpha(t) = \frac{1}{1+e^{-t}}$

Figura 4.5: Soluciones analíticas y solución por la RNA de la EDF con orden constante $\alpha = 0.95$ y orden variable $\alpha(t) = \frac{1}{1+e^{-t}}$

4.5.3. Ejemplo 3

En el siguiente ejemplo se considera el oscilador caótico de Willamowski-Rössler el cual se representa mediante el siguiente sistema de EDF con orden variable [110, 111]

$$
\begin{aligned}
{}_{t_0}^{ABCV}\mathcal{D}_t^{\alpha(t)} x_1(t) &= -\left(x_3(t) + x2(t)\right), \\
{}_{t_0}^{ABCV}\mathcal{D}_t^{\alpha(t)} x_2(t) &= x_1(t) + \varepsilon_1 x_2(t), \\
{}_{t_0}^{ABCV}\mathcal{D}_t^{\alpha(t)} x_3(t) &= \varepsilon_2 + x_3(t)(x_1(t) - \varepsilon_3),
\end{aligned}
\tag{4.25}
$$

donde $\varepsilon_1 = 0.1$, $\varepsilon_2 = 0.1$, $\varepsilon_3 = 14$, $x_1(0) = 4.87623$, $x_2(0) = 4.87623$, $x_3(0) = 0.1278$. Para este ejemplo se definió el orden variable de $\alpha(t) = \tanh(t+1)$ [110]. La ecuación (4.26) muestran las soluciones neuronales propuestas para resolver el sistema de EDF, es importante mencionar que para la metodología propuesta, se propone una RNA para resolver cada uno de los estados por separado como si la variable de estado no depende más que de ella misma.

$$
\begin{aligned}
\hat{y}_{N1} &= 4.87623 + t\hat{y}(t, \mathbf{\Omega}_1), \\
\hat{y}_{N2} &= 4.87623 + t\hat{y}(t, \mathbf{\Omega}_2), \\
\hat{y}_{N3} &= 0.12799 + t\hat{y}(t, \mathbf{\Omega}_3).
\end{aligned}
\tag{4.26}
$$

Se definieron las funciones de activación como $\varphi_1 = \varphi_2 = \tanh(\cdot)$ y $\varphi_3(z) = z$. De acuerdo a la configuración de funciones de activación establecidas, se estiman $\kappa_{\hat{y}} + \kappa_t + 9$ pesos sinápticos del modelo neuronal mostrado en la ecuación (4.27).

$$
\hat{y}[k] = Z_A \tanh\left(v_{\hat{y}} \tanh\left(J_{\hat{y}} w_{\hat{y}} + w_{\hat{y}h}\right) + v_{\hat{y}h}\right) + Z_B \tanh\left(v_t \tanh\left(J_t w_t + w_{th}\right) + v_{th}\right) + Z_H,
\tag{4.27}
$$

donde, $Z_A = Xz_{\hat{y}}$, $Z_B = Xz_t$ y $Z_H = Xz_h$. Se utilizó nuevamente el algoritmo de fuerza bruta para encontrar lo órdenes $\kappa_{\hat{y}}$ y κ_t en un conjunto de valores enteros entre uno y diez, los resultados de la optimización de los regresores se muestran en la Tabla 4.3. Los pesos sinápticos estimados para encontrar la solución del estado $x_1(t)$ son $Z_A = 1.56223$, $Z_B = 26.261711$, $Z_H = -3.69889$, $v_t = 16.8104$, $v_{th} = -39.8019$, $v_{\hat{y}} = -0.0203$, $v_{\hat{y}h} = -0.3412$, $w_{\hat{y}} = [0,\ 0.06,\ 0.0055,\ 0.00006515,\ 0,\ 0.00153,\ -0.000579]^T$ y $w_t = [0.6233,\ 0.6009,\ -0.8293,\ 0.5952,\ 0.3506,\ 0.017,\ 1.1825]^T$.

Para el estado $x_2(t)$ se estimaron los siguientes pesos sinápticos $Z_A = 22.2233$, $Z_B = -0.02687$, $Z_H = 0.2364$, $v_t = 15.4640$, $v_{th} = 0.00125$, $v_{\hat{y}} = -0.0187$, $v_{\hat{y}h} = 0.4131$, $w_{\hat{y}} = [0,\ -0.0405,\ -0.0005,\ 0.001,\ 0.0004,\ -0.0019,\ -0.0004]^T$ y

$w_t = [0.6097,\ 0.5885,\ -0.8403,\ 0.5856,\ 0.3422,\ 0.0097,\ 1.1761]^T$.

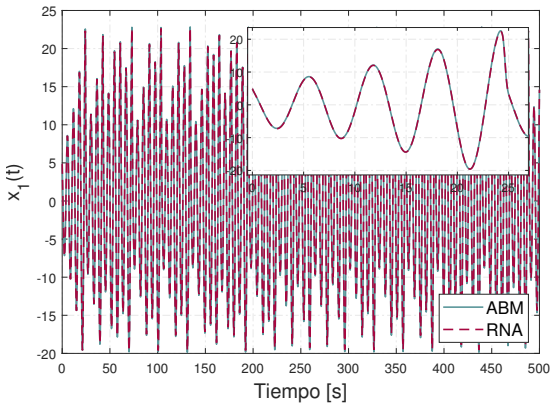
Y finalmente para el estado $x_3(t)$ se estimaron los pesos $Z_A = 0.00498$, $Z_B = 17.3858$, $Z_H = 0.8734$, $v_t = 11.0145$, $v_{th} = -0.0505$, $v_{\hat{y}} = 0.00316$, $v_{\hat{y}h} = 0.2728$,

$w_{\hat{y}} = [0,\ 0.0909,\ 0.0005,\ 0.1,\ 0.014]^T$ y $w_t = [-95.57,\ -7.2719,\ 79.4373,\ 22.681,\ 20.9501]^T$.

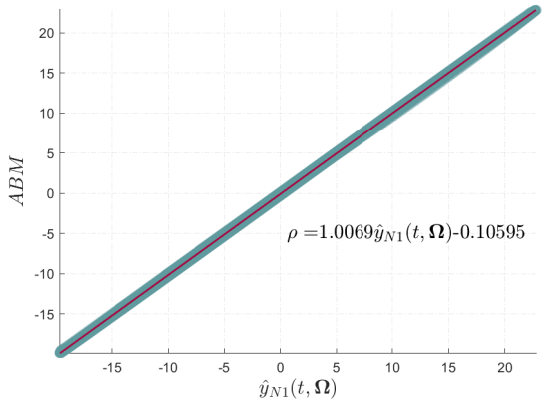
Las Figuras 4.6a, 4.7a, 4.8a muestran la solución obtenida mediante la implementación del algoritmo ABM y la solución neuronal propuesta. En dichas figuras, se realizó un acercamiento al principio de la simulación para denotar que se cumple la condición inicial establecida. Las Figuras 4.6b, 4.7b, 4.8b muestran la prueba de pendiente intercepto que, como se ha mencionado con anterioridad, la pendiente debe ser uno o cercano al uno y el intercepto debe ser cero o cercano a dicho valor. En las pruebas de pendiente-intercepto realizadas se obtuvieron pendientes cercanas a uno e interceptos cercano a cero concluyendo que la estimación de las soluciones del sistema de EDF con orden variable es válida.

Tabla 4.3: Eficiencia de la red neuronal para resolver el sistema de EDF para el oscilador de Rössler.

	$x_1(t)$	$x_2(t)$	$x_3(t)$
$\alpha(t)$	$\tanh(t+1)$	$\tanh(t+1)$	$\tanh(t+1)$
Pendiente	1.0069	0.99685	0.99922
Intercepto	0.01059	-0.07921	0.003071
FIT	98.3441 %	98.6513 %	95.7015 %
$\|\|e\|\|$	42.016	33.2411	42.7595
TS	0.02147	0.02128	0.02028



(a) Solución de la EDF con algoritmo ABM y RNA para el estado $x_1(t)$.

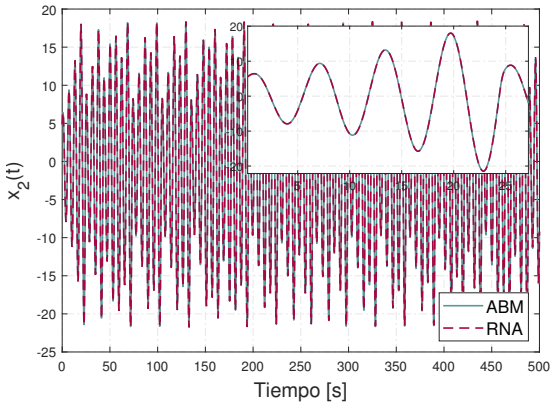


(b) Prueba de pendiente-intercepto del ABM y RNA para el estado $x_1(t)$.

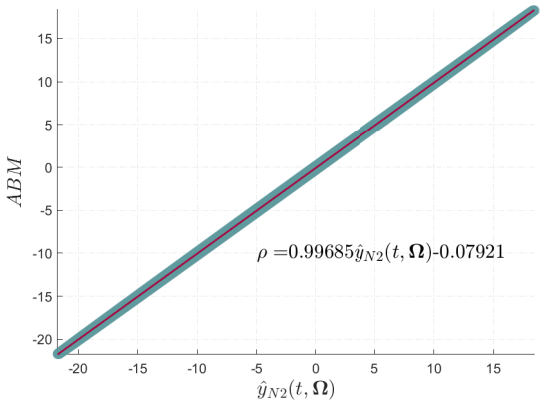
Figura 4.6: Solución del estado $x_1(t)$ para el oscilador caótico de Rössler con $\alpha(t) = \tanh(t+1)$.

La Tabla 4.3 muestra el desempeño de la red neuronal para estimar la solución de cada estado del sistema de EDF del oscilador de Rössler. En la misma tabla se muestra que la norma del error es elevada en cada caso debido a la cantidad de puntos evaluados, es por eso que no se ha fiado solamente de un parámetro para evaluar el desempeño del método propuesto. Por otro lado el FIT, que mide el parentesco entre dos señales, es elevado para cada uno de los estados. Cabe destacar que para todos los estados se realizó la prueba de pendiente-intercepto donde se obtuvieron resultados prometedores. En la misma Tabla se muestra el parámetro TS añadido para medir el costo computacional necesario para realizar los cálculos con el fin de encontrar la solución de la EDF; el algoritmo ABM no permite medir el costo computacional necesario para cada uno de los estados por separado, por lo que se sumó el promedio de cada experimento realizado por la RNA, es decir, se realizaron 1000 repeticiones de cada estado para registrar el tiempo que costó el procesamiento, la suma de estos promedios sería el costo computacional total utilizado por la metodología propuesta que es de $0.06303s$ mientras que el tiempo de computo utilizado con el algoritmo ABM es de $1.4007s$ que es 22 veces más que el tiempo utilizado por la RNA.

Por último en la Figura 4.9 muestra el comportamiento de los tres estados en conjunto y el comportamiento de cada estado de la RNA. Cabe destacar que se introdujo un apuntador para mostrar que las condiciones iniciales son iguales.

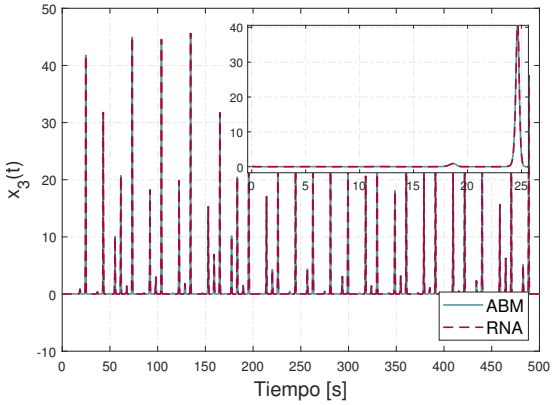


(a) Solución de la EDF con algoritmo ABM y RNA para el estado $x2(t)$.

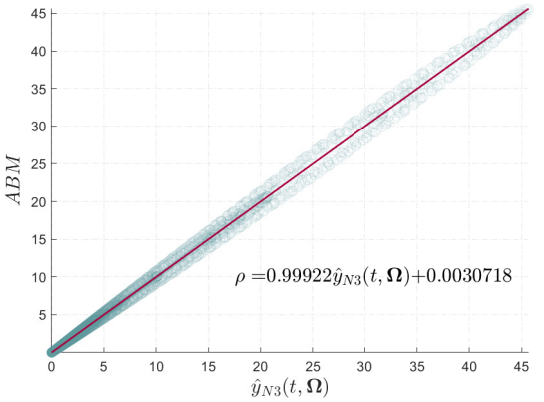


(b) Prueba de pendiente-intercepto del ABM y RNA para el estado $x_2(t)$.

Figura 4.7: Solución del estado $x_2(t)$ para el oscilador caótico de Rössler con $\alpha(t) = \tanh(t+1)$.

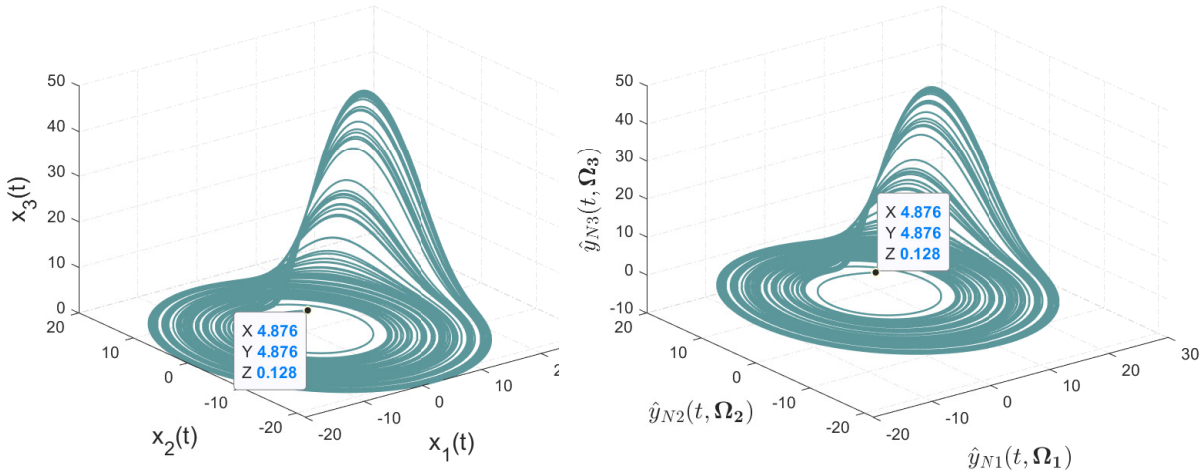


(a) Solución de la EDF con algoritmo ABM y RNA para el estado $x_3(t)$.



(b) Prueba de pendiente-intercepto del ABM y RNA para el estado $x_3(t)$.

Figura 4.8: Solución del estado $x_3(t)$ para el oscilador caótico de Rössler con $\alpha(t) = \tanh(t+1)$.



(a) Soluciones conjuntas producidas por el algoritmo ABM.

(b) Soluciones conjuntas generadas por el método utilizando RNAs.

Figura 4.9: Comportamiento caótico del oscilador de Rössler utilizando el algoritmo ABM y el método propuesto utilizando RNAs.

CAPÍTULO 5

Identificación de Sistemas con Redes Neuronales Fraccionarias

5.1. Introducción

La IS es utilizada en diversos campos de la ingeniería, [114–116]. Las RNA han sido una herramienta útil para realizar la IS de manera satisfactoria [42–44, 117]. Entre las muchas características de las RNAs, se encuentra la habilidad de aproximar funciones no lineales debido al uso de funciones de activación no lineales, su habilidad de procesar diversas señales de entrada y salida, y la facilidad de adaptación mediante algoritmos de entrenamiento aplicados sobre los pesos sinápticos que conforman a la RNA [118]. A pesar de que los enfoques de IS con RNAs son eficientes en términos de precisión [42], un número elevado de parámetros son comúnmente utilizados para lograr esa eficiencia [119]. Con base en lo anterior, diversos trabajos han sido desarrollados con el fin de reducir la complejidad de la IS con modelos neuronales. Por ejemplo, en [120–122], pruebas empíricas fueron utilizadas aumentando el número de parámetros del modelos propuesto hasta alcanzar el grado de precisión definido obteniendo resultados prometedores pero dejando a un lado la relación entre simplicidad del modelo y la precisión que ofrece el mismo [120, 123]. En la literatura se encuentran diversas técnicas propuestas para cumplir con el objetivo de reducir el número de parámetros de un modelo neuronal, por ejemplo, la eliminación de lazos sinápticos donde el peso no ejerce una ponderación importante [124–128], optimización por enjambre de partículas (PSO) [129], recombinación arquitectónica de valor singular [130], algoritmos genéticos (GA) [119, 123, 131, 132], entre otros.

Como se ha mencionado con anterioridad, para realizar la IS es necesario proponer un modelo el cual, asemejará el comportamiento del sistema real con ayuda de la optimización del mismo. El modelo propuesto en este trabajo está basado en modelos neuronales como se proponen en [42–44]. En [42–44] se propone una metodología para reducir la complejidad del modelo neuronal mediante arreglos algebráicos y condiciones de entrenamiento. A diferencia de estos últimos trabajos, se utilizará la metodología basada en CF que se propone en [133] que es una de las contribuciones de este trabajo de investigación. El CF no es ajeno a la identificación de sistemas, de hecho ya se ha aplicado para este fin en distintos sistemas [134, 135].

Diversos trabajos afirman que una de las características de utilizar el CF para la estimación de modelos es la reducción de la complejidad del dicho modelo [37–41]. Por ejemplo, en [38] diversos ejemplos son proveídos como sistemas lineales, no lineales, datos experimentales de intercambiadores de calor con el fin de ser identificados mediante modelos de primer, segundo y de orden fraccionario. En todos los ejemplos mostrados, el modelo de orden fraccionario alcanza un mejor desempeño con un número de parámetros reducido. En [39] se demostró que un modelo fraccionario simple es capaz de representar un modelo de orden superior. En este trabajo se enfatiza un hecho importante, esto es, que los modelos fraccionarios son capaces de caracterizar el comportamiento de sistemas de orden infinito. En [40] se aprovecha el hecho anterior para proponer un método para reducir el número de parámetros de sistemas complejos. En [41] el CF es utilizado para reducir la complejidad de sistemas masa-resorte-amortiguador demostrando que los modelos fraccionarios permiten reducir el número de parámetros sin comprometer la eficiencia del modelo, es decir, que el modelo fraccionario es capaz de igualar la eficiencia del modelo entero con un número de parámetros reducido.

Hasta ahora se han presentado diversos trabajos donde se utilizan las ventajas del CF para la IS mediante diversas técnicas de identificación y modelado. En este capítulo se planteará una RNA fraccionaria (RNAF) para la IS. Los sistemas identificados en este capítulo son dos *benchmarks* (secadora de cabello [136] y modelos de histéresis [137]) y un tercer sistema biológico (predicción de glucosa en la sangre).

5.2. RNA Propuesta

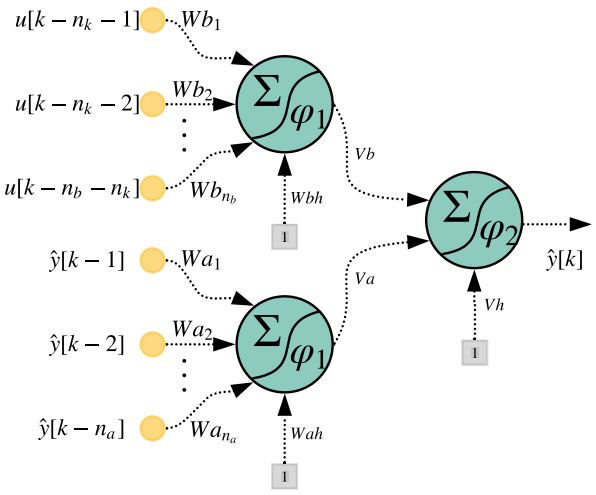


Figura 5.1: RNAF 2-1 propuesta para la identificación de sistemas.

La Figura 5.1 muestra el diseño de la red neuronal utilizada para la propuesta de identificación de sistemas que se abordará en este capítulo. La capa de entrada está compuesta por señales de entrada y salida procesadas con regresores (z^{-1}) que permiten adquirir el conocimiento del comportamiento del sistema en su pasado. La primer capa oculta está compuesta por dos neuronas encargadas de sumar las dinámicas de las señales de entrada y salida por separado. La segunda capa de oculta está compuesta por una neurona encargada de procesar el conocimiento jerarquizado. Finalmente, la RNA tiene una singular ramificación denotada como salida. La representación matemática de la RNA mostrada en la Figura 5.1 está expresada en la ecuación (5.1).

$$\hat{y}[k] = \varphi_2\left(V_b\varphi_1\left(\sum_{i=1}^{n_b} Wb_i u(k-i-n_k) + Wbh\right) + V_a\varphi_1\left(\sum_{i=1}^{n_a} Wa_i\hat{y}(k-i) + Wah\right) + Vh\right), \tag{5.1}$$

donde $Wb \in \mathbb{R}^{n_b}$, $Wa \in \mathbb{R}^{n_a}$, Wbh, Wah, Va, Vb, $Vh \in \mathbb{R}$ son pesos sinápticos $u[k]$ y $\hat{y}[k]$ son las señales de entrada y salida respectivamente. n_a y n_b son los órdenes del modelo neuronal y n_k representa el retardo del sistema.

5.3. Algoritmo de Aprendizaje Fraccionario

En este capítulo se aprovecharán las ventajas del CF para proponer un algoritmo de aprendizaje fraccionario para entrenar el modelo neuronal de la Figura 5.1 representado en la ecuación (5.1). Como se mencionó en el capítulo 3, uno de los métodos de optimización más utilizados para el entrenamiento de una RNA son los basados en gradiente. Con el fin de demostrar las diversas características de la metodología propuesta, se recapitulará lo mostrado en el capítulo 3. Los algoritmos basados en gradiente son capaces de encontrar el valor óptimo de parámetros de acuerdo a la minimización de una función costo u objetivo, por ejemplo (5.2)

$$\mathcal{E}[x,k] = \frac{1}{2}(y[k] - \hat{y}[x,k])^2, \tag{5.2}$$

donde $y[k]$ y $\hat{y}[x,k]$ son las salidas medidas y estimadas respectivamente en el instante k y x es el parámetro que compone a $\hat{y}[x,k]$. Con el fin de encontrar el valor óptimo de x es necesario resolver la siguiente ecuación en diferencias

$$x_j[k+1] = x_j[k] - \eta_j[t|k]\frac{\partial\mathcal{E}[x_j,k]}{\partial x_j[k]}, \tag{5.3}$$

donde j representa el j-ésimo elemento de $\mathbf{x}$, $\mathbf{x}$ es el vector de parámetros, η_j representa el factor de aprendizaje mostrado en la sección 3.5.2 y $\mathcal{E}$ es una función costo. Como se ha mencionado con anterioridad, en este capítulo se propone optimizar el valor de x_j mediante un algoritmo de optimización fraccionario. Debido a que la ecuación (5.3) es una ecuación discreta, se ha propuesto utilizar una derivada fraccionaria discreta como lo es la definición de GL mostrada en la definición 2 y ecuación (2.16). El algoritmo de optimización basado en el descenso por el gradiente fraccionario es mostrado en la ecuación (5.4).

$$\Delta^{\alpha}x_j = -\eta[t|k]\frac{\partial\mathcal{E}[x_j,k]}{\partial x_j[k]}. \tag{5.4}$$

5.4. Método de Estabilidad de Lyapunov

Considere un sistema dinámico el cual satisface

$$\dot{x} = f(x(t),t), \quad x(t_0) = x_0,\ x \in \mathbb{R}, \tag{5.5}$$

donde $x(t) \in \mathbb{D} \subseteq \mathbb{R}^n$ representa la solución de (5.5), $\mathbb{D}$ un conjunto abierto que contiene al origen y $f : \mathbb{D} \mapsto \mathbb{R}^n$ una función continua y $f(x^*) = 0$ entonces [138]:

i) El punto de equilibrio x^* es estable en el sentido de Lyapunov en $t = t_0$ si para cualquier $\varepsilon > 0$ existe una función $\delta(t_0,\varepsilon) > 0$ tal que

$$||x(t_0) - x^*|| < \delta \Rightarrow ||x(t) - x^*|| < \varepsilon, \quad \forall t \geq t_0. \tag{5.6}$$

ii) El punto de equilibrio x^* es estable asintóticamente en el sentido de Lyapunov si existe una función $\delta > 0$ tal que

$$||x(t_0) - x^*|| < \delta \Rightarrow \lim_{t \mapsto \infty} ||x(t) - x^*|| = 0. \tag{5.7}$$

iii) El punto de equilibrio x^* es estable exponencialmente si es asintótiamente estable y si existen $c_1,\ c_2,\ \delta > 0$ tal que

$$||x(t_0) - x^*|| < \delta \Rightarrow ||x(t) - x^*|| \leq c_1||x(t_0) - x^*||e^{-c_2 t}, \quad \forall t \geq t_0. \tag{5.8}$$

5.4.1. Segundo método de estabilidad de Lyapunov

En [139] Lyapunov propuso dos métodos demostrando la estabilidad de sistemas. El primero fue desarrollado mediante la solución de series convergentes. El segundo método, el cual es conocido ahora como el *criterio de estabilidad de Lypunov* o *método directo de Lyapunov*, se basa en el uso de una *función de Lyapunov* $V(x)$ positiva. Considerando un sistema como la ecuación (5.5) y un punto de equilibrio $x = 0$. Considerando una función $V : \mathbb{R}^n \mapsto \mathbb{R}$ tal que

- $V(x) = 0 \iff x = 0.$
- $V(x) > 0 \iff x \neq 0.$
- $\dot{V}(x) = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i} f_i(x) = \nabla V f(\mathbf{x}), \quad \forall x \neq 0.$ El sistema (5.5) es estable si $\dot{V}(x) < 0,\ para\, x \neq 0.$

5.4.2. Extensión del método directo de Lyapunov para sistemas fraccionarios

Sea un sistema fraccionario

$${}_0\mathcal{D}_t^{\alpha} x(t) = f(t, x), \tag{5.9}$$

donde $\alpha \in (0, 1]$, $\mathcal{D}$ es el operador fraccionario en el sentido RL o LC. x^* es el punto de equilibrio de (5.9) de tal modo que $f(x^*) = 0$.

En secciones anteriores se han definido los conceptos de estabilidad en el sentido de Lyapunov. El método directo de Lyapunov se basa en encontrar una función que cumpla con los requisitos que el método estipula (ver sección 5.4.1). Para proponer una extensión del segundo método de Lyapunov se debe introducir el concepto de estabilidad Mittag-Leffler o estabilidad ML.

Definición 11. *La solución de* (5.9) *es estable ML si*

$$||x(t)|| \leq (m(x(t_0)))E_{\alpha}(-\lambda(t - t_0)^{\alpha}))^{c_1} \tag{5.10}$$

donde t_0 es el tiempo inicial, $\alpha \in (0, 1], \lambda \geq 0, b > 0, m(0) = 0, m(x) \geq 0$ y $m(x)$ es localmente Lipschitz en $x \in \mathbb{R}^n$ con una constante de Libschitz m_0.

Utilizando el método directo de Lyapunov, se obtiene una estabilidad asintótica del sistema (5.5). Ahora, se presentará un extensión al segundo método de Lyapunov para sistemas fraccionarios para mostrar la estabilidad ML [140].

Sea $x = 0$ un punto de equilibrio de (5.9) y $\mathbb{D} \subset \mathbb{R}^n$ es un dominio con contiene al origen. Sea una función $V(t, x(t)) : [0, \infty) \times \mathbb{D} \to \mathbb{R}$ continuamente diferenciable y localmente Lipschitz con respecto a x tal que

$$c_1||x||^{d_1} \leq V(t, x(t)) \leq c_2||x||^{d_1 d_2}, \tag{5.11}$$

$$_0\mathcal{D}_t^{\alpha} V(t, x(t)) \leq c_3||x||^{d_1 d_2}, \tag{5.12}$$

donde $t \geq 0$, $x \in \mathbb{D}$, $\alpha \in (0, 1]$, $c_1, c_2, c_3, d_1, d_2 > 0$. Entonces, el punto de equilibrio $x = 0$ es estable ML si las suposiciones (5.11) y (5.12) se mantienen en $\mathbb{R}^n$. La prueba de la extensión de la estabilidad de Lyapunov se encuentra en [140].

5.5. Estabilidad del Algoritmo de Aprendizaje Fraccionario Propuesto

Para probar la estabilidad del algoritmo propuesto se define un función de Lyapunov discreta

$$V[x, k] = \mathcal{E}[x, k] = \frac{1}{2} e^2[x, k]. \tag{5.13}$$

El cambio de la función de Lyapunov (5.13) es

$$\Delta^{\alpha} V[x, k] = \frac{V[x, k+1] - V[x, k]}{\sigma^{1-\alpha}} = \frac{1}{2s} \left(e^2[x, k+1] - e^2[x, k] \right), \tag{5.14}$$

donde $e[x, k] := y[k] - \hat{y}[x, k]$, $y[k]$ y $\hat{y}[x, k]$ representan las señales de salida medida y de la red neuronal respectivamente, x el parámetro que se desea optimizar, σ es un parámetro temporal auxiliar con el fin de dimensionar el tiempo [141], $\alpha \in (0, 1]$ representa el orden de la derivada fraccionaria y s una variable auxiliar para representar $\sigma^{1-\alpha}$. Si $\Delta e[x, k] = s^{-1}(e[x, k+1] - e[x, k])$, entonces

$$\begin{aligned} e[x, k+1] &= s^{-1} \Delta^{\alpha} e[x, k] + e[x, k], \\ \Rightarrow e^2[x, k+1] &= s^{-2} \Delta^{\alpha} e^2[x, k] + 2 s^{-1} e[x, k] \Delta e[x, k] + e^2[x, k], \end{aligned} \tag{5.15}$$

sustituyendo (5.15) en (5.14) se tiene

$$\begin{aligned} \Delta^{\alpha} V[x, k] &= \frac{1}{2s} \left(s^{-2} \Delta^{\alpha} e^2[x, k] + 2 s^{-1} e[x, k] \Delta^{\alpha} e[x, k] + e^2[x, k] - e^2[x, k] \right), \\ \Delta^{\alpha} V[x, k] &= \frac{\Delta^{\alpha} e[x, k]}{s^3} \left(\frac{1}{2} \Delta^{\alpha} e[x, k] + e[x, k] s \right). \end{aligned} \tag{5.16}$$

Se aproxima $e[x, k+1]$ mediante una serie de Taylor con $n = 1$ en el punto $x_0 = x[k]$ tal que

$$e[x, k+1] = e[x, k] + \frac{\partial e[x, k]}{\partial x[k]} s \Delta^{\alpha} x[k] \Rightarrow \Delta^{\alpha} e[x, k] = \frac{\partial e[x, k]}{\partial x[k]} \Delta^{\alpha} x[k], \tag{5.17}$$

sustituyendo la ecuación (5.17) en (5.16) se obtiene

$$\Delta^{\alpha}V[x,k] = \frac{1}{s^3}\frac{\partial e[x,k]}{\partial x[k]}\Delta^{\alpha}x[k]\left(\frac{1}{2}\frac{\partial e[x,k]}{\partial x[k]}\Delta^{\alpha}x[k] + e[x,k]s\right), \tag{5.18}$$

considerando que $\Delta^{\alpha}x[k]$ es el algoritmo de aprendizaje fraccionario mostrado en (5.4), entonces

$$\Delta^{\alpha}V[x,k] = \frac{1}{s^3}\frac{\partial e[x,k]}{\partial x[k]}\left(-\eta[t|k]\frac{\partial \mathcal{E}[x,k]}{\partial x[k]}\right)\left(\frac{1}{2}\frac{\partial e[x,k]}{\partial x[k]}\left(-\eta[t|k]\frac{\partial \mathcal{E}[x,k]}{\partial x[k]}\right) + e[x,k]s\right). \tag{5.19}$$

Utilizando la regla de la cadena con

$$\begin{aligned}\frac{\partial e[x,k]}{\partial x[k]} &= \frac{\partial e[x,k]}{\partial \hat{y}[x,k]}\frac{\partial \hat{y}[x,k]}{\partial x[k]} = -\frac{\partial \hat{y}[x,k]}{\partial x[k]}\\ \frac{\partial \mathcal{E}[x,k]}{\partial x[k]} &= \frac{\partial e[x,k]}{\partial x[k]}\frac{\partial e[x,k]}{\partial \hat{y}[x,k]}\frac{\partial \hat{y}[x,k]}{\partial x[k]} = -e[x,k]\frac{\partial \hat{y}[x,k]}{\partial x[k]},\end{aligned} \tag{5.20}$$

se sustituye en (5.19) como

$$\begin{aligned}\Delta^{\alpha}V[x,k] &= -\frac{1}{s^3}\frac{\partial \hat{y}[x,k]}{\partial x[k]}\left(\eta[t|k]e[x,k]\frac{\partial \hat{y}[x,k]}{\partial x[k]}\right)\left(e[x,k]s - \frac{1}{2}\frac{\partial \hat{y}[x,k]}{\partial x[k]}\left(\eta[t|k]e[x,k]\frac{\partial \hat{y}[x,k]}{\partial x[k]}\right)\right),\\ \Delta^{\alpha}V[x,k] &= -\frac{1}{2s^3}\left(e[x,k]\frac{\partial \hat{y}[x,k]}{\partial x[k]}\right)^2\left(2\eta[t|k]s - \eta^2[t|k]\left(\frac{\partial \hat{y}[x,k]}{\partial x[k]}\right)^2\right).\end{aligned} \tag{5.21}$$

El término

$$2\eta[t|k]s - \eta^2[t|k]\left(\frac{\partial \hat{y}[x,k]}{\partial x[k]}\right)^2, \tag{5.22}$$

de la ecuación (5.21) debe ser mayor a cero con el fin de que la derivada de la función de Lyapunov sea negativa definida y así cumplir con el criterio de estabilidad. Por lo tanto, se considera que el factor de compensación temporal es siempre un valor mayor a cero y se resuelve la desigualdad

$$2\eta[t|k]s - a^2\eta^2[t|k] \geq 0, \tag{5.23}$$

considerando que $a^2 = \left(\frac{\partial \hat{y}[x,k]}{\partial x[k]}\right)^2$. Las soluciones de la desigualdad (5.23) en función del factor de aprendizaje η son

$$\begin{aligned}\eta[t|k] &\geq 0,\ si\, a = 0,\\ 0 \leq \eta[t|k] &\leq \frac{2s}{a^2},\ si\, a > 0.\end{aligned} \tag{5.24}$$

De acuerdo a las desigualdades de (5.24), para cualquier valor positivo de η la derivada de la función de Lyapunov (5.21) es estable asintóticamente, culminando así la prueba de estabilidad del algoritmo de aprendizaje fraccionario propuesto en la ecuación (5.4). Con el fin de evitar divergencias numéricas debido al alto valor que se le puede asignar al factor de aprendizaje, se han mencionado en la sección 3.5.2 algunas formas para definir

este parámetro. En [42–44, 133] se recomienda que el valor del factor de aprendizaje varíe en un rango de $0 \leq \eta[t|k] \leq 1$.

5.6. Entrenamiento Fraccionario Neuronal

De acuerdo a la ecuación que caracteriza el comportamiento neuronal propuesto (ver ecuación (5.1)) se tienen un total de $n_a + n_b + 5$ pesos sinápticos. Aplicando el esquema numérico en el sentido de GL mostrado en la ecuación (A.3) del apéndice A se presentan los entrenamientos fraccionarios para cada peso sináptico de la RNA mostrada en la Figura 3.6.

$$
\begin{aligned}
Va[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Va[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Va[t(k)-j], \\
Vb[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Vb[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Vb[t(k)-j], \\
Vh[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Vh[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Vh[t(k)-j], \\
Wa[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wa[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Wa[t(k)-j], \\
Wb[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wb[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Wb[t(k)-j], \\
Wah[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wah[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Wah[t(k)-j], \\
Wbh[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wbh[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Wbh[t(k)-j].
\end{aligned}
\tag{5.25}
$$

El efecto del CF es análogo a una media móvil ponderada, debido a que el kernel de la derivada se encarga de ponderar los valores más recientes con mayor peso que los datos más antiguos. Por lo tanto, los pesos sinápticos del entrenamiento (5.25) tiene propiedades no locales, es decir, para producir el valor actual de la ecuación diferencial fraccionaria, necesita de su información pasada. Ahora se le define como red neuronal artificial fraccionaria (RNAF) a una RNA o modelo neuronal cuyos pesos sinápticos son entrenados mediante algoritmos con propiedades no locales como la ecuación (5.25).

5.7. Resultados

En esta sección se muestran los resultados obtenidos de la implementación de la metodología propuesta. Con el fin de comparar los resultados obtenidos de la identificación de diversos sistemas, la RNA fue entrenada mediante el algoritmo fraccionario y el algoritmo clásico. Además, si el sistema fue identificado mediante otras metodologías, los resultados fueron incluidos para realizar la comparación.

La comparación es desarrollada en términos del número de parámetros n_p y el criterio de Akaike mediante el cálculo de la predicción final del error (FPE por sus siglas en inglés *Final Prediction Error*) la cual proporciona una medida de calidad del modelo al simular el modelo propuesto con una base de datos distinta con la que fue estimado. Para medir la precisión del modelo, se ha utilizado la raíz del error medio cuadrático (RMSE), medidas de parentesco (FIT) y por último la prueba de pendiente intercepto. Cabe destacar que el caso ideal es cuando $y[k] = \hat{y}[x, k]$ donde el FIT sería igual al $100\,\%$, el $RMSE = 0$, pendiente igual a uno y un intercepto igual a cero. El cálculo del FIT se ha mostrado en la ecuación (4.14), los demás índices mencionados se muestran en las ecuación (5.26), (5.27) y (5.28).

$$FPE = \frac{1}{N}\sum_{i=1}^{N} e_i^2(k, \hat{x}(k)) \left(\frac{1 + \frac{d}{N}}{1 - \frac{d}{N}}\right), \tag{5.26}$$

donde N es el número de valores que conforman a la base de datos utilizada para realizar la estimación, $e(t)$ es la señal producida mediante la comparación de la salida medida $y[k]$ y la salida del modelo neuronal $\hat{y}[k, \hat{x}[k]]$, $\hat{x}$ representa el vector de parámetros estimados y d representa el número total de parámetros estimados [4].

$$RMSE = \sqrt{\frac{\sum_{k=1}^{N}(y[k] - \hat{y}[k, \hat{x}])^2}{N}}, \tag{5.27}$$

donde $y[k]$ es la señal de salida real del sistema, $\hat{y}[k, \hat{x}]$ es la salida producida por el modelo neuronal y $\hat{x}$ es el vector de parámetros estimados por el algoritmo de entrenamiento fraccionario.

$$\begin{aligned} m &= \frac{\sum_{k=1}^{N}(y[k] - \bar{y})\left(\hat{y}[k] - \bar{\hat{y}}\right)}{\sum_{k=1}^{N}(y[k] - \bar{y})^2} \\ b &= \bar{\hat{y}} - m\bar{y} \end{aligned} \tag{5.28}$$

donde $y[k]$ es la señal de salida medida, $\bar{y}$ la media de la señal de salida medida, $\hat{y}$ la señal de salida estimada por la RNA, $\bar{\hat{y}}$ la media de la señal de salida estimada por la RNA, m y b representan la pendiente y el intercepto respectivamente de la prueba pendiente-intercepto.

5.7.1. Sistema 1: Secadora de Cabello

El sistema está compuesto por una malla hecha por cables resistivos encargados de calentar el flujo de aire en la entrada de la tubería del dispositivo. La señal de entrada del sistema es el voltaje aplicado al dispositivo y la salida es la temperatura del aire a través de del conducto de aire medida mediante un termopar ubicado en el orificio de salida del dispositivo. La base de datos, la cual está disponible en el portal Web DaISy (por sus sigas del inglés *Database for the Identification of Systems [136]*), está compuesta por 1000 muestras, infortunadamente no se cuenta con el periodo de muestreo como para saber la duración del experimento. La Figura 5.2 muestra las señales de entrada y salida del sistema donde

la señal de entrada es una señal binaria pseudo-aleatoria (PRBS del inglés *pseudo-random binary signal*).

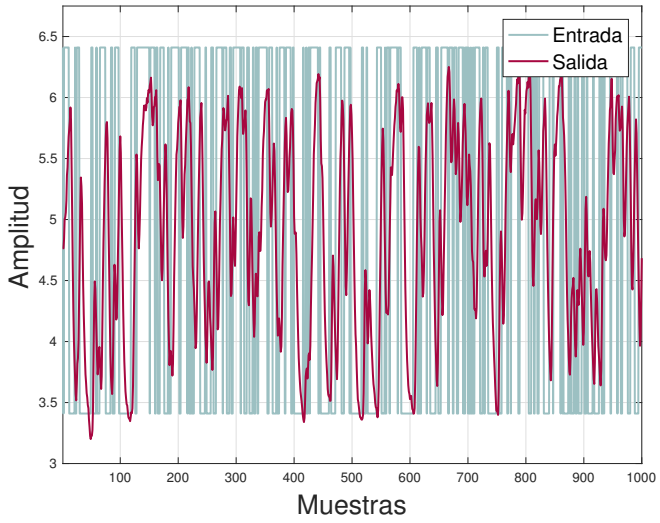


Figura 5.2: Señales de entrada y salida para sistema de secado.

Como se ha mostrado en secciones anteriores, el modelo neuronal puede ser manipulado mediante operaciones algebraicas dependiendo de la selección de las funciones de activación de la RNAF. De la Figura 5.1 las funciones de activación se definieron como $\varphi_1(z) = \tanh(z)$ y $\varphi_2(z) = z$, por lo tanto el modelo neuronal se expresaría como (5.29).

$$\hat{y}[k] = Vb \tanh\left(\sum_{i=1}^{n_b} Wb_i u[k-i-n_k] + Wbh\right) + Va \tanh\left(\sum_{i=1}^{n_a} Wa_i \hat{y}[k-i] + Wah\right) + Vh. \tag{5.29}$$

Como se ha mencionado al principio de esta sección, el modelo neuronal dado por la ecuación (5.29) es entrenado 1) por el algoritmo de entrenamiento de orden fraccionario y 2) por el algoritmo basado en descenso del gradiente propuesto en [43]. En ambos casos, los ordenes n_a, n_b y tiempo muerto n_k fueron optimizados por el cambio de todos los parámetros en un rango de 1 a 10 con un paso igual a uno, es decir, se aplicó el algoritmo de fuerza bruta mostrado en el apéndice D. Un total de 1000 modelos fueron estimados para cada uno de los algoritmos de entrenamiento. El mejor resultado de los 1000 modelos con la RNAF y el mejor modelo de los 1000 estimados por la RNA fuero escogidos para realizar la comparación. El orden de derivación α fue optimizado por el algoritmo de optimización por enjambre de partículas mostrado en el apéndice C. El PSO fue utilizado para la reducción de una función costo mostrada en la ecuación (5.27) durante 10 épocas. El sistema ha sido identificado mediante otra metodología basada en normas matriciales (NNSI) propuesta en [1] considerando sus condiciones de entrenamiento y validación. Los mejores dos modelos obtenidos del algoritmo de fuerza bruta antes mencionado fueron entrenados por el 30 % del total de los datos y validados por el 70 % del total, tal como se propone con el modelo NNSI. Con esta validación se permite realizar una comparación justa con los modelos propuestos y el modelo NNSI. La comparación fue realizada en términos de precisión mediante la medición del FIT validando el modelo con los datos de entrenamiento (FIT_{tr}) y los datos de validación (FIT_v), el RMSE en ambas situaciones ($RMSE_{tr}$ y $RMSE_v$), la prueba de pendiente-intercepto con ambos conjuntos de datos,

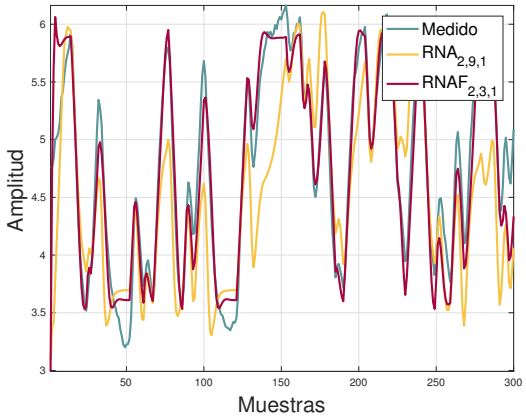
número de parámetros n_p y el FPE.

Después de la optimización de los ordenes y retardos de n_a, n_b y n_k para la RNAF se obtuvieron 2, 3 y 1 respectivamente. Mientras que para el modelo clásico se obtuvieron los valores de 2, 9 y 2 respectivamente. Mediante la optimización del orden de derivación con el uso del PSO se obtuvo un valor de $\alpha = 0.9678127473$. La Tabla 5.1 muestra el desempeño del modelo con el entrenamiento local, el modelo no local y el modelo NNSI. El valor del RMSE, tanto en los datos de entrenamiento como validación, el modelo fraccionario otorga un mejor desempeño que el modelo tradicional. Con el fin de comparar los modelos de manera visual, las figuras 5.3a y 5.3c muestran los resultados obtenidos de manera temporal con los datos de entrenamiento y validación respectivamente mediante la simulación del modelo considerando que solamente se tiene la señal de entrada y en consecuencia, la salida producida es retroalimentada para cerrar el lazo de la RNAF y la RNA.

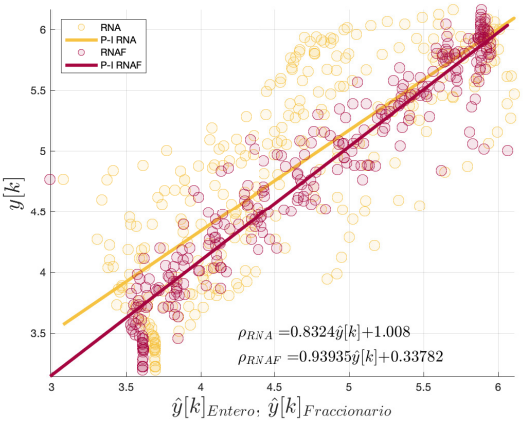
Las figuras 5.3b y 5.3d muestran las pruebas de pendiente-intercepto para los datos de entrenamiento y validación respectivamente, donde el modelo RNAF se encuentra más cercano a la idealidad, es decir la pendiente del modelo RNAF se encuentra más cercana a la unidad que el modelo RNA y del mismo modo, el intercepto del modelo RNAF es más cercano al cero que el modelo RNA. Comparando el modelo con la metodología propuesta en este trabajo y el modelo entrenado convencionalmente [43], la RNAF muestra un mejor desempeño con un número de parámetros inferior. El número de parámetros n_p se ve reflejado directamente en la calidad del modelo la cual, es calculada mediante el FPE donde el modelo RNAF obtuvo un mejor desempeño que su contra parte clásica. Realizando la comparación con el modelo NNSI, el modelo RNAF es superior con respecto a la medición del FIT. Infortunadamente, en [1] se carece de información acerca de su metodología como para realizar más comparaciones de los modelos.

Tabla 5.1: Desempeño de la RNAF, RNA y el modelo NNSI. 300 datos fueron utilizados para entrenar los modelos neuronales y estimar el orden fraccionario α. La validación del modelo fue realizada con las últimas 700 muestras del conjunto de datos.

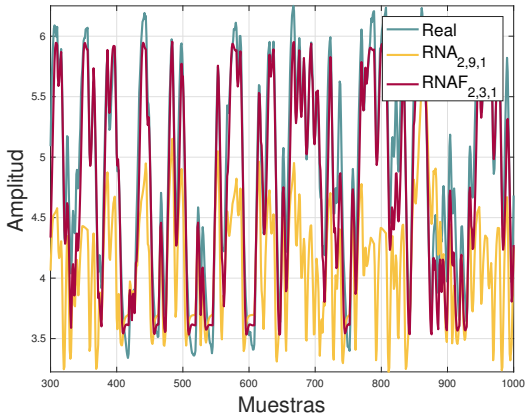
Modelo	n_p	FIT		$RMSE$	
		Entrenamiento	Validación	Entrenamiento	Validación
RNAF	11	94.19 %	93.65 %	0.264789575053522	0.333519
RNA	16	87.11 %	80.34 %	0.591260203914813	1.112078
NNSI	-	-	90.2 %	-	-
Modelo	FPE	Pendiente		Intercepto	
		Entrenamiento	Validación	Entrenamiento	Validación
RNAF	0.0754508	0.93935	0.97754	0.33782	0.15062
RNA	0.3889788	0.8324	1.1641	1.008	0.15062
NNSI	-	-	-	-	-



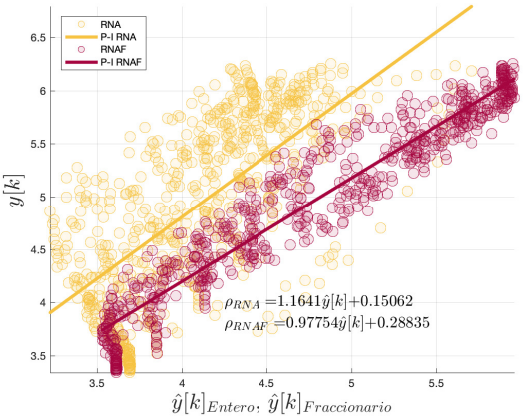
(a) Simulación del modelo entero (RNA) y el modelo fraccionario (RNAF) con los datos de **entrenamiento**.



(b) Prueba de pendiente-intercepto (P-I) de los modelos RNA y RNAF con los datos de **entrenamiento**.



(c) Simulación de la RNA y la RNAF con los datos de **validación**.

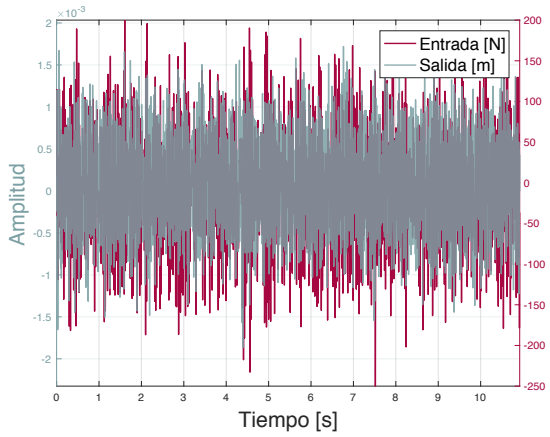


(d) Prueba de pendiente-intercepto (P-I) de los modelos RNA y RNAF con los datos de **validación**.

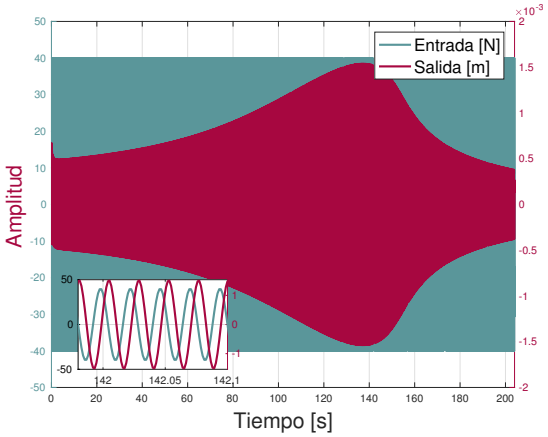
Figura 5.3: Comportamiento temporal del sistema con el modelo fraccionario RNAF ($[n_a, n_b, n_k, \alpha] = [2, 3, 1, 0.9678127473]$) y el modelo clásico RNA ($[n_a, n_b, n_k] = [2, 9, 2]$) con 300 muestras del conjunto de entrenamiento y 700 muestras del conjunto de validación [1].

5.7.2. Sistema 2: Modelo de Histéresis de Bouc-Wen

Con el fin de probar la flexibilidad de la metodología propuesta, un segundo sistema ha sido identificado. El sistema consiste en el modelo de Bouc-Wen, el cual es utilizado para representar efectos de histéresis en ingeniería mecánica [137, 142, 143]. La señal de entrada y salida corresponden a una fuerza aplicada a un sistema amortiguado y la posición de una masa respectivamente. Las señales fueron muestreadas a una frecuencia de $750Hz$. La base de datos total está compuesta por dos partes, entrenamiento y validación. Los datos de entrenamiento están conformados por 8192 muestras (ver Figura 5.4a) y 153000 muestras para la validación (ver Figura 5.4b). El entrenamiento y validación fue realizada de acuerdo a lo propuesto en [137].



(a) Simulación de la RNA y la RNAF con los datos de **validación**.



(b) Prueba de pendiente-intercepto (P-I) de los modelos RNA y RNAF con los datos de **validación**.

Figura 5.4: Señales de entrada y salida para realizar el entrenamiento de los modelos fraccionario y entero (lado izquierdo) y señales de entrada-salida para realizar la validación de los modelos (lado derecho).

$$\hat{y}[k] = Vb\tanh\left(\sum_{i=1}^{n_b} Wb_i u[k-i-n_k] + Wbh\right) + Va\tanh\left(\sum_{i=1}^{n_a} Wa_i \hat{y}[k-i] + Wah\right) + Vh. \tag{5.30}$$

El modelo neuronal para la identificación de este sistema está dado por la ecuación (5.30). Los datos de entrenamiento (8192 muestras) fueron utilizados para la optimización de los órdenes y retardos. Del mismo modo que el ejemplo anterior, los ordenes y retardo fueron obtenidos por el cambio de n_a, n_b y n_k en un rango definido. Para este ejemplo el rango de cambio para n_a y n_b se definió entre $\Omega_{n_a,n_b} \in \{1, 2, \ldots, 20\}$ y el rango de valores para encontrar el retardo natural óptimo se definió como $\Omega_{n_k} \in \{1, 2, \ldots, 10\}$. Se generaron 4000 modelos diferentes por cada método de aprendizaje (el fraccionario para la RNAF y el clásico para la RNA), el mejor modelo fue seleccionado de cada algoritmo. Al igual que el ejemplo anterior, el PSO ha sido utilizado para encontrar el valor óptimo del orden fraccionario α. Después de la optimización, los órdenes y retardo óptimos se definieron como $[n_a, n_b, n_k, \alpha] = [10, 1, 1, 0.99788]$ para la red neuronal fraccionaria y para el modelo neuronal de orden entero se estimaron $[n_a, n_b, n_k] = [12, 5, 1]$.

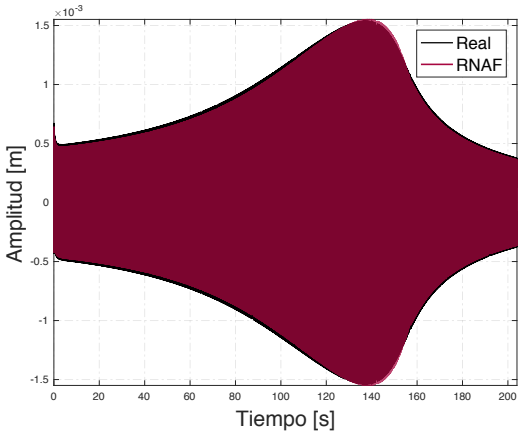
De acuerdo a los requisitos del benchmark propuesto en [137], los modelos son evaluados por pruebas de simulación y predicción con el conjunto de datos referente a la validación compuesto por 153000 muestras donde se calcula el RMSE y aunado a ello, se calcularon los otros parámetros de desempeño mostrados en el sistema anterior.

Las figuras 5.5a y 5.5b muestran la prueba de predicción para los modelos neuronales (fraccionario y entero respectivamente). Cabe destacar que la prueba de predicción no es capaz de garantizar que la estimación es eficiente, debido a que el sistema neuronal es alimentado con las señales de entrada y salida con las que fue entrenado.

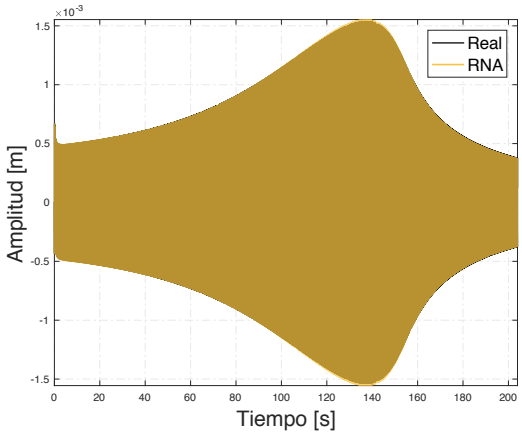
Tabla 5.2: Desempeño de la RNAF, RNA y diversas metodologías encontradas en la literatura para la identificación del modelo con histéresis Bouc-Wen. La validación fue realizada mediante las pruebas de predicción (Pred) y simulación (Sim) a lo largo de 153000 muestras que conforman los datos de validación.

Model	n_p	FIT		$RMSE$	
		Pred	Sim	Pred	Sim
RNAF	16	96.80 %	72.98 %	6.89×10^{-6}	1.79×10^{-4}
RNA	22	96.96 %	58.31 %	2.01×10^{-5}	2.76×10^{-4}
[144]	-	-	-	0.0687	1.38
[145]	67	-	-	-	3.88×10^{-4}

Model	FPE		Pendiente		Intercepto	
	Pred	Sim	Pred	Sim	Pred	Sim
RNAF	1.12×10^{-5}	1.68×10^{-4}	1.012	0.91	3.86×10^{-9}	2.50×10^{-8}
RNA	3.83×10^{-5}	3.54×10^{-4}	0.994	0.82	6.69×10^{-7}	4.23×10^{-6}
[144]	-	-	-	-	-	-
[145]	-	-	-	-	-	-



(a) Predicción del modelo RNAF con los datos de validación.



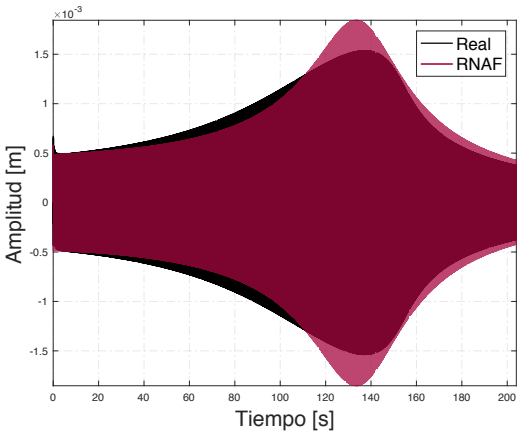
(b) Predicción del modelo RNA con los datos de validación.

Figura 5.5: Predicción de modelo fraccionario ($[n_a, n_b, n_k, \alpha] = [10, 1, 1, 0.99788]$) y modelo entero ($[n_a, n_b, n_k] = [12, 5, 1]$).

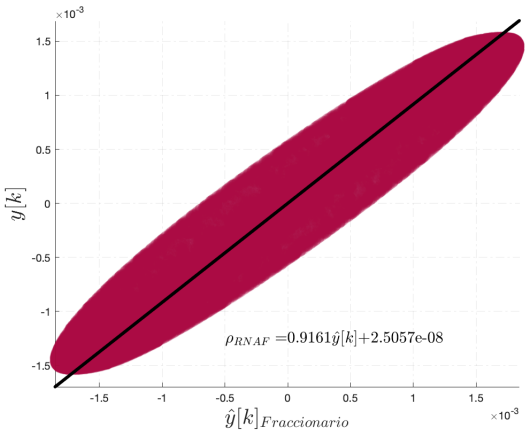
Por otro lado, la prueba de simulación se caracteriza por utilizar la señal de entrada como una señal excitadora que en consecuencia producirá una señal de salida (otorgada por el modelo neuronal). La señal producida, es retroalmientada al modelo neuronal para que funja como una señal de entrada más y producir nueva información proveniente del modelo neuronal. La Figura 5.6 muestra la prueba de simulación del modelo neuronal fraccionario. De la prueba de simulación, se realizó la prueba de pendiente-intercepto con el fin de mostrar la relación entre la señal de salida medida (señal de salida real del sistema) y la señal producida por el modelo neuronal fraccionario. Del mismo modo, la Figura 5.7 muestra los resultados de simulación y pendiente-intercepto para el modelo de orden entero.

Cabe destacar que las figuras 5.6b y 5.7b muestran que el intercepto es cercano a cero o relativamente igual a cero, debido a la amplitud de la señal que se desea estimar, es decir, ya que la magnitud de la señal real del sistema es pequeña, el valor del intercepto será pequeño o cercano a cero. En la Tabla 5.2 se observa que, de hecho, el modelo neuronal

fraccionario alcanza una mejor precisión con los demás modelos excepto en la prueba de predicción donde el modelo de orden entero obtuvo un mejor FIT. Aún así, en las pruebas de simulación, n_p, FIT en simulación, RMSE, etc. El modelo neuronal fraccionario obtuvo un mejor desempeño que los demás incluyendo los modelos reportados en la literatura [144, 145].

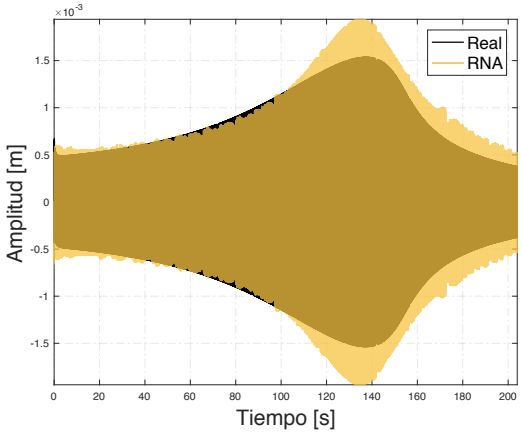


(a) Simulación del modelo neuronal fraccionario con los datos de validación.

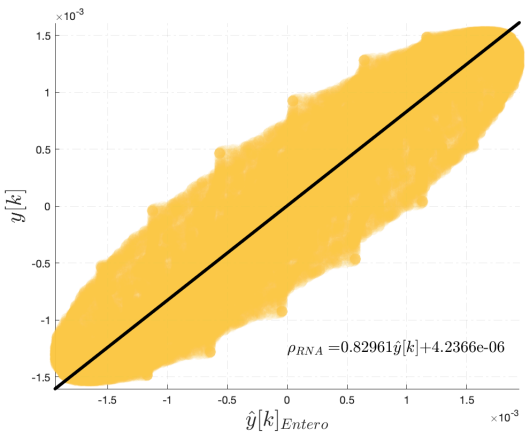


(b) Prueba de pendiente-intercepto con la simulación del modelo RNAF de orden reducido.

Figura 5.6: Simulación del modelo neuronal fraccionario reducido con $[n_a, n_b, n_k, \alpha] = [10, 1, 1, 0.99788]$ y prueba de pendiente-intercepto.



(a) Simulación del modelo neuronal de orden entero con los datos de validación.



(b) Prueba de pendiente-intercepto con la simulación del modelo neuronal de orden entero.

Figura 5.7: Simulación del modelo neuronal de orden entero con $[n_a, n_b, n_k] = [12, 5, 1]$ y prueba de pendiente-intercepto.

5.7.3. Sistema 3: Predicción de Glucosa

La diabetes de tipo uno (DT1) es una enfermedad crónica que ocurre cuando el páncreas es incapaz de producir la insulina que el cuerpo necesita. La insulina es una hormona la cual regula la glucosa en la sangre (GS) ayudando a las células a convertir la glucosa en energía para el cuerpo. En este sentido, la DT1 deteriora el control de la GS ocasionando riesgos de sufrir hipoglicemias (GS < 70mg/dL) o hiperglicemias (GS > 180mg/dL). Las hiperglicemias ocasionan severas complicaciones en el sistema cardiovascular como

deterioro de vasos sanguíneos, deterioro en órganos, etc. Por otro lado, la hipoglicemia ocasiona riesgos de tener convulsiones, coma, y por último, la muerte. En consecuencia, los pacientes con DT1 deben optimizar sus dosis de insulina con el fin de mantener los niveles de GS en un rango entre [70, 180] mg/dL [146]. El tratamiento para pacientes con DT1 consiste en el suministro exógeno de dosis de insulina mediante inyecciones repetitivas a lo largo de su vida. Recientemente, una nueva tecnología llamada páncreas artificiales (PA) ha sido desarrollada con el fin de mejorar o acabar con el tratamiento basado en la inyección punzante de insulina. Los PA combinan un monitoreo continuo de glucosa (MCG) con bombas de insulina utilizando algoritmos de control encargados de regular las dosis de insulina que el cuerpo requiere.

La predicción de GS (PGS) ayuda a mejorar el control de la GS. Por ejemplo en [147] y [148], dos grupos de investigación presentan dos sistemas los cuales, basados en la PGS recomiendan la dosis óptima de insulina suministrada manualmente para atender a pacientes con DT1. El resultado de los dos sistemas propuestos se vio reflejado en al promedio de concentración de glucosa plasmática (HbA1c), la cual es un indicador relacionado con las complicaciones que la diabetes produce. Otros ejemplos donde PGS es utilizado para el mejoramiento del control de la GS fueron presentados en [149] y [150], donde eventos hipoglicémicos se redujeron mediante la reducción o suspensión del suministro de insulina cuando la predicción de GS mostraba niveles por debajo del umbral definido como 70 mg/dL. Bajo el mismo propósito, en [151, 152], se ha propuesto utilizar un PA en lazo cerrado realizando la PGS para optimizar el suministro de insulina requerido mediante un algoritmo de control predictivo (MPC) [153].

Cabe destacar que el proceso de PGS se realiza mediante el uso de un modelo matemático que prediga los niveles de glucosa bajo los efectos de diversas variables. En la literatura se encuentran diversos modelos enfocados unicamente en describir el comportamiento de la GS [153–157]. En este sentido, en este trabajo se ha propuesto utilizar modelos basados en redes neuronales artificiales, debido a que han demostrado que son útiles para el modelado de sistemas complejos [42–44, 110, 111, 116, 133]. De hecho, las RNAs ya han sido utilizadas para predecir el nivel de GS, por ejemplo en [158] un modelo LSTM (Long-Short Term-Memory) es comparado con modelos basados en regresión de vector soporte (Support Vector Regression) y modelos autoregresivos de media móvil. El modelo propuesto en [158] es una RNA recurrente bidireccional (BRNN), es decir, los estados de las entradas fluyen hacia delante y hacia atrás iterativamente. Cada una de las ramificaciones (hacia adelante y hacia atrás) está compuesta por 4 neuronas en la capa de entrada, ocho, 64 y ocho neuronas distribuidas en tres capas ocultas respectivamente para producir predicciones de 30, 40 y 60 minutos. En [159] se propone una RNA convulusiva (CNN) para predecir la concentración de GS a 30 y 60 minutos. La RNA propuesta utiliza los datos del nivel de glucosa en la sangre, comidas e insulina como señales de entrada. En [160] un modelo neuronal LSTM recibe señales de niveles de glucosa sanguínea para suministrar a una RNA entrenada mediante el método del gradiente para predicciones a 30 y 60 minutos.

El uso del CF no es ajeno al modelado de DT1. Por ejemplo, en [161] se propone utilizar el modelo mínimo de Bergman utilizando una derivada fraccionaria en el sentido de Caputo para observar el efecto de la insulina en la concentración GS. Entre otros ejemplos, en [162] se propone utilizar un controlador PID fraccionario ($PI^{\lambda}D^{\mu}$), un observador no lineal de orden fraccionario [163], un observador no lineal adaptable con modos deslizantes [164] y un controlador difuso de nivel [165] fueron propuestos para regular el nivel de glucosa en la sangre. A pesar de que los trabajos presentados que relacionan al CF con la predicción de glucosa mostraron resultados prometedores gracias al uso de la no localidad

en el CF, ninguna de ellos fue validado con datos provenientes de pacientes reales.

Método

A. *Entradas del Modelo*

Como es propuesto en [166], utilizando la información de carbohidratos ingeridos por **comidas** y los niveles de **insulina** en el sistema es posible obtener una predicción de glucosa eficaz. En este sentido, se ha propuesto utilizar para el modelo neuronal las variables IOB, COB y GS como entradas al modelo neuronal.

La insulina activa o IOB por sus siglas en inglés (insulin on board) se define como la insulina inyectada que sigue teniendo un efecto en la concentración de glucosa en la sangre. El IOB es calculado mediante la ecuación (5.31).

$$u^{IOB}[k] = \sum_{n=1}^{N} I[k-n]h_{IOB}[n], \tag{5.31}$$

donde $I[n]$ es la cantidad de insulina en U/h entregada por una bomba de insulina en el n-ésimo instante de tiempo. El COB e la porción de comida que sigue teniendo un efecto sobre el comportamiento de la GS. El COB es calculado mediante la ecuación (5.32)

$$u^{COB}[k] = \sum_{n=1}^{N} CHO[k-n]h_{COB}[n], \tag{5.32}$$

donde $CHO[n]$ es la cantidad carbohidratos en gramos ingeridos en el n-ésimo instante de tiempo. Cabe destacar que la porción de CHO es declarada por cada uno de los pacientes dependiendo de la hora que en su propia dieta se hayan definido. En (5.31) y (5.32), $N = 48$ para considerar ocho horas de datos y $h[n] = h[n\tau_s]$ con $\tau_s = 10min$ está dada por la ecuación (5.33):

$$h[n] = \left(1 + \frac{n}{\tau}\right) e^{-\frac{n}{\tau}}, \tag{5.33}$$

donde $\tau = 50min$ para (5.31) y para (5.32), $\tau = 40min$ de acuerdo a [167].

B. *Modelo Neuronal Propuesto*

Similar al modelo presentado en la ecuación (5.1) y Figura 5.1, el modelo neuronal propuesto para predicción de glucosa se muestra en la Figura 5.8. Nótese que la diferencia entre la estructura neuronal de la Figura 5.8 y la Figura 5.1 se basa en el número de entradas que se han definido para estimar la señal de salida y el número de neuronas, es decir, el usuario es capaz de utilizar el número de neuronas que este desee. Por lo tanto, el modelo neuronal de (5.1) tiene algunas variaciones en su representación matemática como lo muestra (5.34) con funciones de activación $\varphi_1(z) = \tanh(z)$ y $\varphi_2(z) = z$.

$$\hat{y}^{GS}[k] = \sum_{i=1}^{nn} \left(Vb_i \tanh\left(\sum_{j_1=1}^{n_{b1}} u^{IOB}[k-j_1-n_{k1}]Wb^i_{1,j_1} + \sum_{j_2=1}^{n_{b2}} u^{COB}[k-j_2-n_{k2}]Wb^i_{2,j_1} + Wbh_i \right) + Va_i \tanh\left(\sum_{j=1}^{n_a} \hat{y}[k-j]Wa^i_j + Wah_i \right) + Vh \right) \tag{5.34}$$

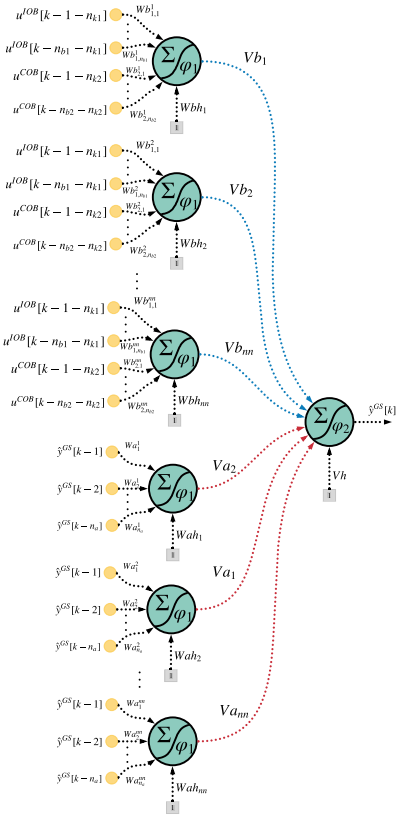


Figura 5.8: Estructura de red neuronal propuesta para la predicción de glucosa en la sangre.

donde $Va_i \in \mathbb{R}^{nn}$, $Vb \in \mathbb{R}^{nn}$, $Vh \in \mathbb{R}$, $Wah \in \mathbb{R}^{nn}$, $Wbh \in \mathbb{R}^{nn}$, $Wb_{1,i} \in \mathbb{R}^{n_{b1} \times nn}$, $Wb_{2,i} \in \mathbb{R}^{n_{b2} \times nn}$, $Wa_i \in \mathbb{R}^{n_a \times nn}$ son pesos sinápticos, nn es el número de neuronas para procesar las señales de entrada y salida, $u^{IOB}[k]$ es la entrada de insulina activa o IOB, $u^{COB}[k]$ representa la ingesta de carbohidratos o COB, $\hat{y}^{BG}[k]$ es la concentración de glucosa en la sangre, n_a, n_{b1} y n_{b2} es el orden del modelo neuronal y n_{k1} y n_{k2} representan los tiempos muertos del sistema.

C. *Entrenamiento Fraccionario de la RNA*

De acuerdo a lo mostrado en la sección 5.6, es posible realizar el entrenamiento de una red neuronal de manera no local como se muestra en la ecuación (5.25). Utilizando la definición discreta de derivada fraccionaria en el sentido GL mostrada en la ecuación (2.16) y esquema numérico mostrado en (A.3), el entrenamiento de la RNA de (5.34) es:

$$
\begin{aligned}
Va[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Va[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Va[t(k)-j], \\
Vb[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Vb[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Vb[t(k)-j], \\
Vh[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Vh[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Vh[t(k)-j], \\
Wa[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wa[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Wa[t(k)-j], \\
Wb_{1,i}[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wb_{1,i}[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} W_{1,i}[t(k)-j], \\
Wb_{2,i}[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wb_{2,i}[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Wb_{2,i}[t(k)-j], \\
Wah[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wah[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Wah[t(k)-j], \\
Wbh[k+1] &= -\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wbh[t(k)]}h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} Wbh[t(k)-j].
\end{aligned}
\tag{5.35}
$$

D. *Optimización de órdenes n_a, n_{b1}, n_{b2}, tiempos muertos n_{k1}, n_{k2} y α.*

Para el caso de la RNA de orden entero, se utilizó el algoritmo de fuerza bruta midiendo el valor RMSE (ver ecuación (5.27)) para optimizar los órdenes (n_a, n_{b1} y n_{b2}) y tiempos muertos ($n_k k1$ y n_{k2}) como también el número de neuronas. Los rangos de optimización se definieron entre uno y cinco para los órdenes y tiempos muertos de la RNA y de 1 a 20 para el número de neuronas. Por tanto, el número total estimados corresponde a 62500 modelos diferentes. Después de haber aplicado el algoritmo de fuerza bruta, se encontraron los órdenes $n_a = 5$, $n_{b1} = 5$, $n_{b2} = 1$, $n_{k1} = 3$ y $n_{k2} = 3$. La Figura 5.9 muestra el resultado de la variación de las neuronas con los valores óptimos $n_a = 5$, $n_{b1} = 5$, $n_{b2} = 1$, $n_{k1} = 3$ y $n_{k2} = 3$. Es importante destacar que de la Figura 5.9, se seleccionó aquel valor que cumpliese un equilibrio entre un valor pequeño de RMSE, un valor pequeño de nn y que las muestras RMSE fueran uniformes, es decir, que los datos no estuvieran contaminados con valores atípicos. Bajo este contexto, el valor mínimo de RMSE encontrado fue con $nn = 3$ y $RMSE = 17.1$ pero está altamente contaminado por valores atípicos, por lo tanto se definió como $nn = 10$ ya que este no se encuentra contaminado por valores atípicos y su valor RMSE está cercano al mínimo con $RMSE = 17.45$. Es de suma importancia mencionar que cada muestra corresponde al promedio RMSE de 1000 repeticiones, es decir, el programa se ejecutó en 1000 ocasiones con condiciones iniciales aleatorias con el fin de mostrar que el algoritmo es robusto y obtener una mejor exactitud ante el valor RMSE. Con los órdenes definidos y al igual que el número de neuronas se calcularon 151 pesos sinápticos con el método del gradiente convencional mostrado en el apéndice A.

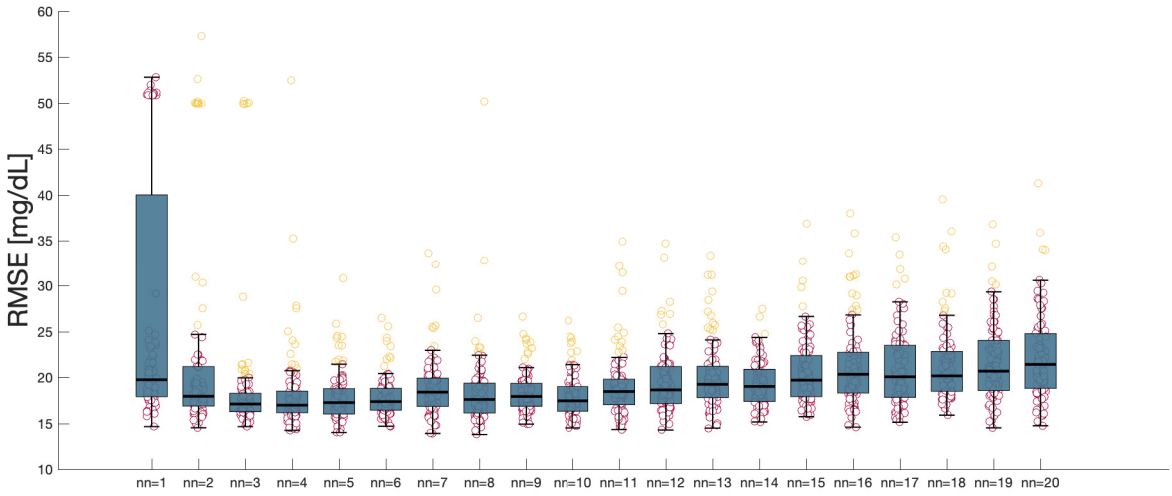


Figura 5.9: Selección del número de neuronas para el modelo neuronal mostrado en la Figura 5.8.

Debido a que se busca reducir el número de parámetros mediante el uso del algoritmo de aprendizaje fraccionario, de la RNA de la Figura 5.8, se definió el número de neuronas igual a uno. Por otro lado, se utilizó una combinación entre el algoritmo de fuerza bruta y el PSO (ver apéndice C) para encontrar los valores de los órdenes (n_a, n_{b1} y n_{b2}) y tiempos muertos ($n_k k1$ y n_{k2}) y el orden fraccionario α. Después de haber concluido con la combinación de algoritmos (fuerza bruta y PSO), se encontraron los órdenes $n_a = 3$, $n_{b1} = 11$ y $n_{b2} = 7$, tiempos muertos $n_{k1} = 2$, $n_{k2} = 2$ y orden fraccionario $\alpha = 0.9998721$.

Evaluación experimental

A. *Base de datos*

La base de datos utilizada para el entrenamiento, pruebas y validación del modelo neuronal fraccionario propuesto están conformados por datos de 30 pacientes adultos con DT1 tratados mediante un pancras artifical. Los pacientes fueron parte de programa autorizado por la autoridad nacional de seguridad francesa (ANSM) al rededor de 12 hospitales. Los datos utilizados para entrenar a la RNA y RNAF fueron muestreados por el sistema Dexcom 5G CGM cada 5 min, la insulina suministrada por la bomba fue muestreada cada minuto y finalmente, los datos acerca de los carbohidratos fueron ingresados por los pacientes manualmente. Las tres señales fueron remuestreadas con un periodo de 10 min.

B. *Evaluación de pruebas*

Se procesaron los datos de 30 pacientes con DT1 por 45 días. La Figura 5.10 muestra el proceso realizado para separar los datos en entrenamiento y validación. Por cada paciente, los datos fueron separados en subconjuntos de entrenamiento compuestos por 30 días (referentes a 8640 muestras) y un subconjunto de validación compuesto por 15 días (referente a 4320 muestras). Como se ha mencionado con anterioridad, el modelo fue entrenado mediante la información proveniente de 30 pacientes, por lo tanto el conjunto de datos de entrenamiento y pruebas está compuesto por 259200 muestras.

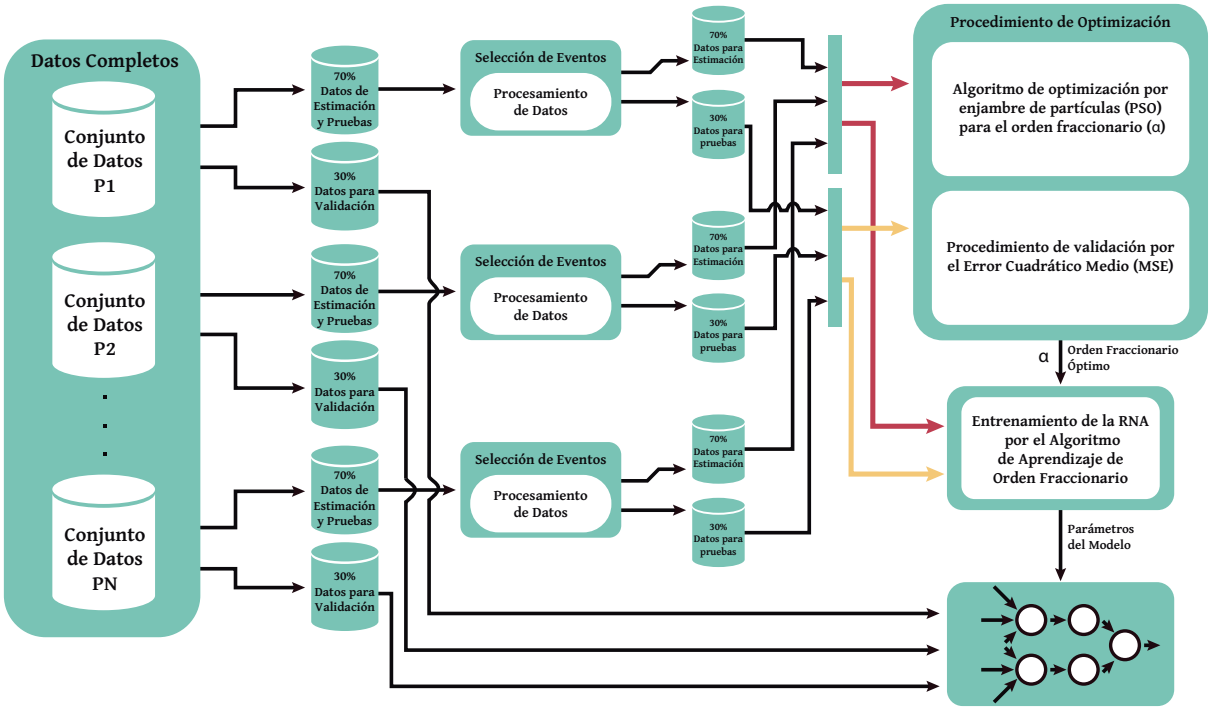


Figura 5.10: La base de datos fue separada en datos de entrenamiento y datos de validación. De cada subconjunto, diversos eventos fueron seleccionados para optimizar y entrenar al modelo neuronal fraccionario.

De los 30 subconjuntos de entrenamiento, se definieron eventos donde el sistema mostrara una mayor dinámica esto sucede durante la ingesta de alimentos. Cada evento está formado por cuatro horas de muestreo, dos horas antes y dos horas después de haber ingerido alimentos. La Figura 5.11 muestra un ejemplo de los eventos seleccionados para realizar el entrenamiento de las RNAs.

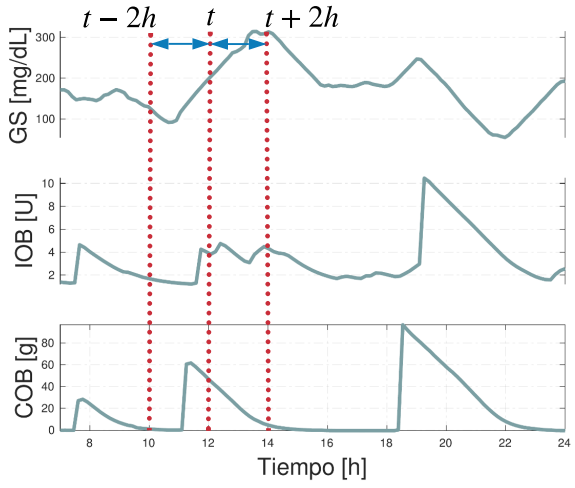


Figura 5.11: Ejemplo de extracción de eventos del conjunto de datos de entrenamiento. Los niveles de GS, IOB y COB son extraídos al rededor de una comida (2 horas antes y 2 horas después).

El 70 % de cada uno de los subconjuntos de información de entrenamiento fue utilizado para la optimización del orden fraccionario α y validado sobre el 30 % restante mediante el algoritmo PSO. Finalmente, la validación fue realizada mediante la predicción de

los niveles de glucosa referentes a 15 días (los cuales la RNAF no consideró durante su entrenamiento). Los horizontes de predicción (HP) propuestos para evaluar la eficiencia de la RNAF fueron de 30 y 60 minutos. El instrumento de medición utilizado para cuantificar la eficiencia del modelo neuronal fraccionario fue el RMSE más la desviación estándar sobre los datos de validación de cada paciente.

C. *Prueba de Predicción*

El objetivo de esta identificación es predecir el comportamiento de la glucosa en "n muestras en el futuro". Para realizar la validación se desarrolló un algoritmo que permitiera completar dicho objetivo. Primeramente se debe saber qué valor entre n_a, $n_{b_1} + n_{k_1}$ y $n_{b_2} + n_{k_2}$ es mayor. El algoritmo espera hasta que el vector (entrada o salidas) con mayor dimensión sea llenado con la información necesaria. Una vez que todos los vectores tienen un valor determinado, la RNAF arroja la primera predicción (1 muestra en el futuro) "p_1". En el siguiente paso, la RNAF utiliza esa primera predicción para realizar una predicción dos muestras adelante "p_2". El proceso se repite hasta llegar a la predicción "p_n" donde la predicción dependerá unicamente de las predicciones realizadas por la RNA y no de los valores anteriormente sabidos.

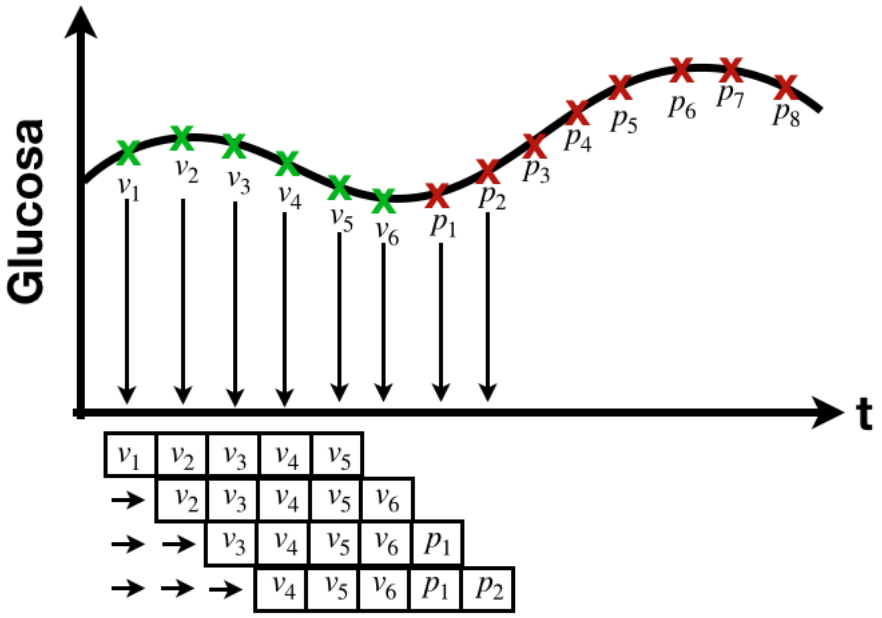


Figura 5.12: Esquema de prueba de predicción.

La Figura 5.12 muestra un esquemático del proceso de predicción considerando que la máxima combinación de los órdenes es cinco y que se está prediciendo cada 10 minutos según la frecuencia de muestreo que se utilizó para realizar la identificación del sistema.

D. Indicador de Desempeño

El indicador de desempeño es el RMSE (ver ecuación (5.27)) debido a que la mayoría de los autores utilizan este indicador para evaluar el desempeño de los modelos propuestos.

Resultados

Se realizaron pruebas con 10, 15, 20, 30 y 60 minutos como horizonte de predicción (HP) para la RNA y la $RNAF$ bajo las mismas condiciones, es decir, para realizar una

comparación justa se utilizó la misma base de datos para validación. Cabe recordar que se está utilizando la información de 30 pacientes durante 15 días (en vida cotidiana) con un tiempo de muestreo de 5 min. La Figura 5.13 muestra el desempeño de cada RNA entrenada de forma fraccionaria y convencional para los diversos HP que se han mencionado donde la RNAF muestra obtener un mejor desempeño a lo largo de la evaluación de los 30 pacientes que la RNA.

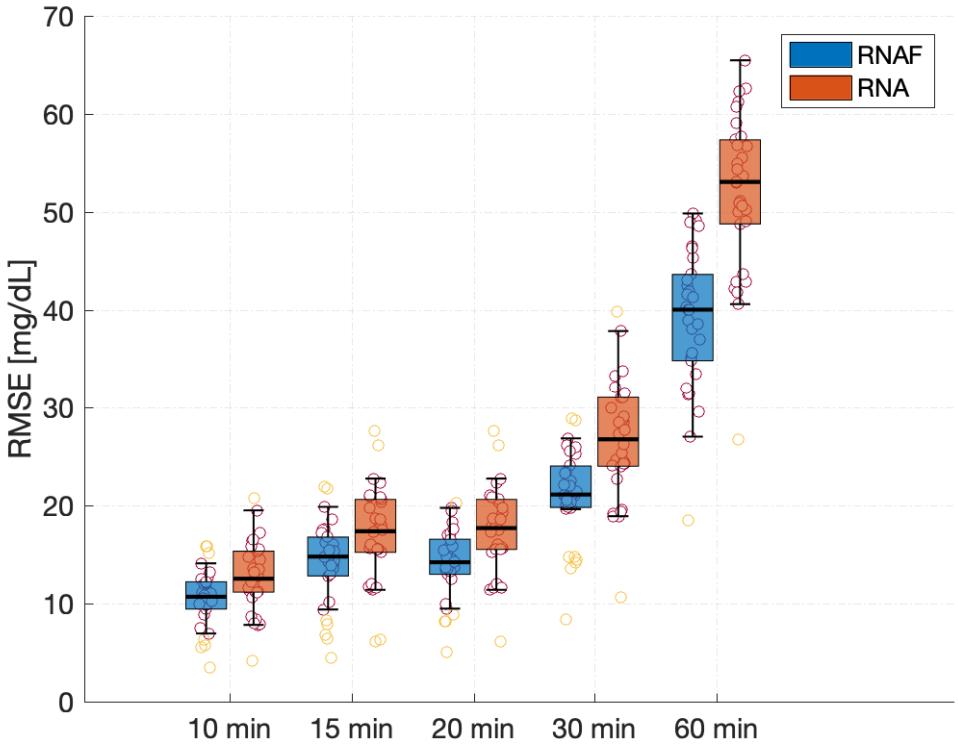
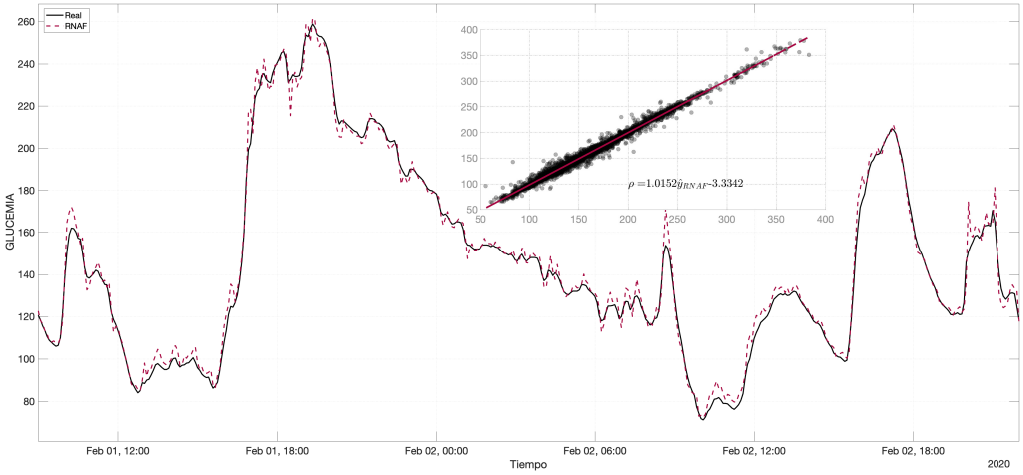


Figura 5.13: Desempeño de la RNAF y RNA con diversos HP.

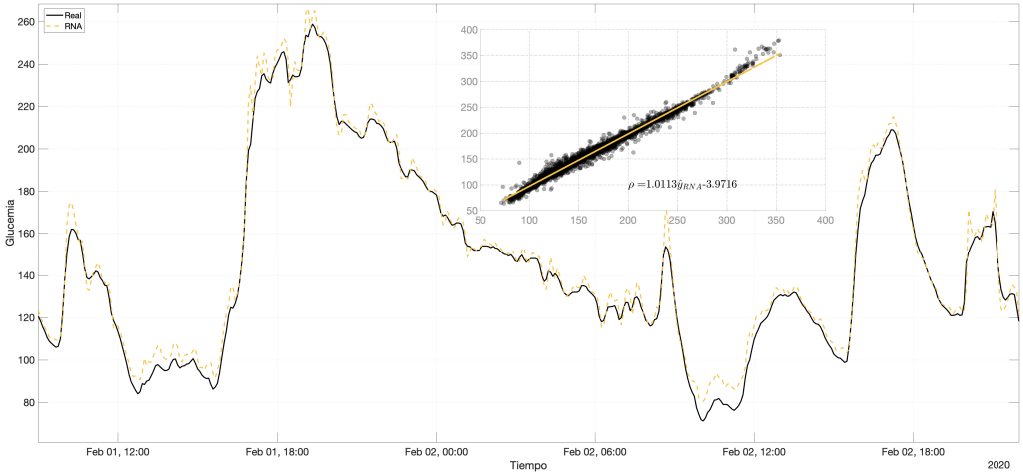
La Tabla 5.3 muestra el RMSE (promedio $\pm$ std) de modelos basados en RNAs para la predicción de glucosa con horizontes de 30 y 60 min encontrados en la literatura, el modelo fraccionario propuesto y el modelo neuronal entrenado de forma convencional.

Tabla 5.3: Comparación entre diferentes desempeños encontrados en la literatura y el desempeño alcanzado por el modelo fraccionario propuesto y convencional para predicciones de 30 y 60 min.

Referencia	$\tilde{RMSE} \pm std(RMSE)$[mg/dL] de predicciones	
	30 min	60 min
LSTM [158]	21.747	36.918
CNN [159]	21.07 ± 2.35	$\mathbf{33.27 \pm 4.79}$
BRNN [160]	30.6 ± 11.6	65.7 ± 27
RNAF	$\mathbf{21.0019986 \pm 4.7657}$	39.02535 ± 7.2772
RNA	26.554484 ± 6.1696	51.8724 ± 8.27



(a) Predicción realizada por la RNAF.



(b) Predicción realizada por la RNA.

Figura 5.14: Predicciones de 10 minutos a un paciente aleatorio seleccionado entre 30 por la RNAF y la RNA sobre periodos de desayuno, comida y cena.

La Figura 5.14 muestra la predicción del modelo neuronal fraccionario 5.14a y el modelo neuronal convencional 5.14b. El comportamiento mostrado es de un paciente seleccionado al azar en un intervalo de tiempo que fuera afable con el fin de mostrar dos periodos alimenticios (desayuno, comida y cena). Las siguientes predicciones se realizarán bajo el mismo paciente en el mismo intervalo de tiempo considerando diversos HP.

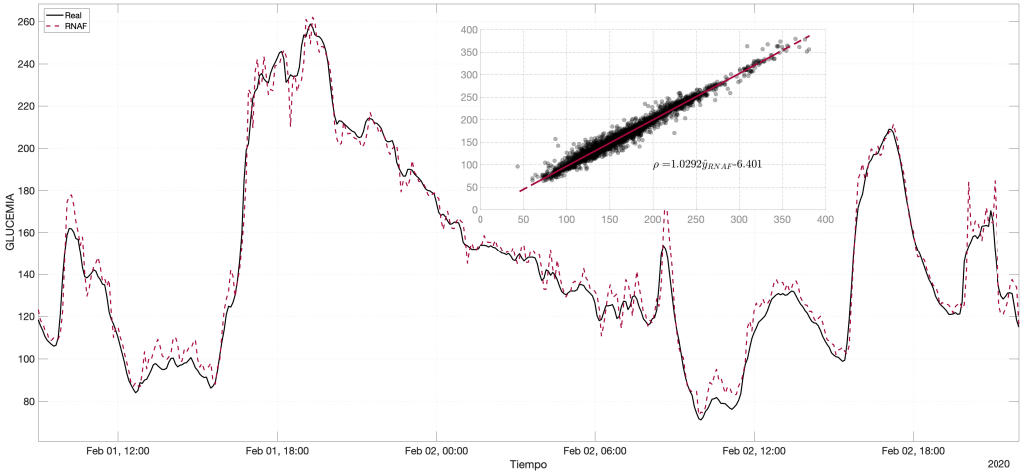
Discusión

De la Figura 5.9 se seleccionaron diez neuronas para el modelo neuronal convencional que, con el orden encontrado mediante el algoritmo de fuerza bruta, 151 parámetros fueron optimizados. Por otro lado, con el fin de reducir el número de parámetros de la RNA, se definió el número de neuronas igual a uno para el caso fraccionario que, junto con los ordenes de los regresores n_a, n_{b1}, n_{b2}, n_{k1} y n_{k2}, se optimizaron 26 parámetros solamente. Si la calidad del modelo fuese evaluada (FPE visto en la ecuación (5.26)) el modelo RNAF es mejor que su contra parte convencional.

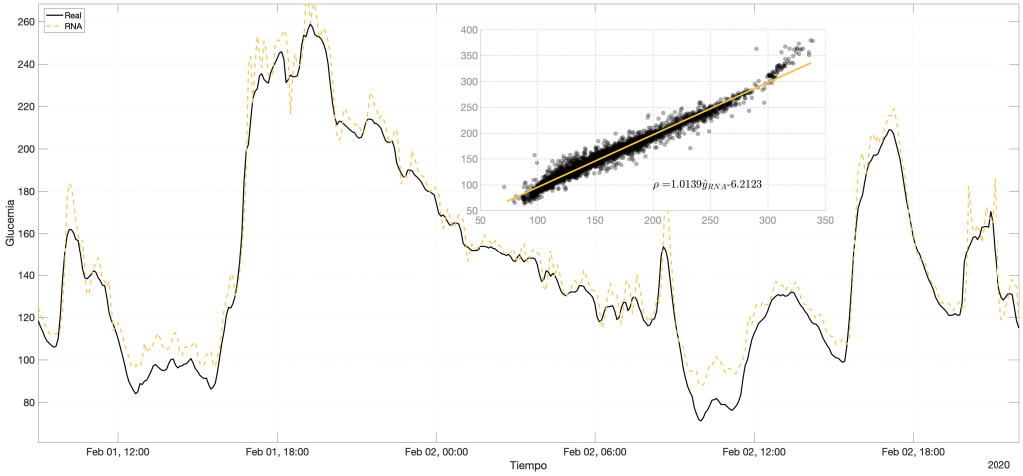
Los órdenes y retardos naturales (tiempos muertos) del modelo neuronal fraccionario permitieron considerar la dinámica de la concentración de GS a lo largo de 30 min, IOB de $t-110$ a $t-20$ min y COB de $t-70$ a $t-20$ min considerando a t como el tiempo actual. Por otro lado el modelo convencional utilizó la dinámica la concentración de GS a lo largo de 50 min, IOB de $t-30$ a $t-10$ min y COB de $t-50$ a $t-30$ min. El impacto de las tres variables de salida (GS) y entrada (IOB y COB) son importantes para las predicciones realizadas por la RNAF ya que se consideran HP dentro de los intervalos mencionados. Se realizaron pruebas de pendiente intercepto en cada una de las predicciones realizadas donde la RNA mostró tener un mejor desempeño para el paciente actual pero, gracias a lo mostrado en la Figura 5.13 se puede concluir que el paciente mostrado en las figuras 5.14, 5.16, 5.15, 5.17 y 5.18 solo fue un caso particular donde la RNA fue superior. Es importante denotar que conforme el HP se eleva, la predicción comienza a carecer de precisión; este comportamiento es totalmente normal debido a que se están prediciendo HP con pocas muestras del pasado, si el HP es pequeño entonces la predicción será más precisa.

Visualmente, la diferencia entre el desempeño entregado por la RNAF y la RNA es similar, y es cierto, esto debido a que el orden $\alpha = 0.998721$ es cercano a uno (el caso clásico o el entrenamiento convencional); a pesar de ello, el hecho de cambiar el orden de derivación permitió reducir el número de neuronas y por ende el número de parámetros de la RNAF.

Ahora, hablando de los desempeños mostrados en la Tabla 5.3, el modelo RNAF es superior para las predicciones de 30 min y el modelo propuesto en [158] ha sido el mejor modelo para la predicción de 60 min. Infortunadamente la Tabla 5.3 solamente puede realizar una comparación justa entre los valores RMSE obtenido de la predicción del modelo RNAF y el modelo RNA ya que la validación de estos fue realizada bajo la misma base de datos. Es decir, los modelos propuestos en [158–160] fueron evaluados con otro tipo de base de datos. En [158] por ejemplo, se realizó la evaluación sobre 20 pacientes con DT1 compuesta por 26 conjuntos de información con 1500 muestras en cada uno de ellos (39000 muestras en total) correspondientes a cinco días de pruebas (con un periodo de muestreo de cinco minutos); de acuerdo a la estructura de red neuronal mostrada se optimizaron 16384 parámetros para realizar la predicción. En [160] se evaluó el modelo sobre 32 pacientes con DT1 durante siete días de vida cotidiana muestreados cada 5 minutos (2016 muestras en total). Cabe destacar que los modelos propuestos fueron evaluados sobre 30 pacientes con DT1 con una duración de 15 días muestreados cada 5 minutos dando un total de 129600 muestras en total con 151 y 26 parámetros para el modelo RNA y RNAF respectivamente. En este contexto, los modelos propuestos han mostrado un mejor desempeño en función del valor RMSE y el número de parámetros que los modelos encontrados en la literatura. Solo el modelo propuesto en [159] ha sido evaluado bajo condiciones similares ya que se realizó la validación sobre 10 pacientes con DT1 con tres meses de pruebas con un modelo neuronal compuesto por 524288 parámetros (según la información que el artículo muestra).

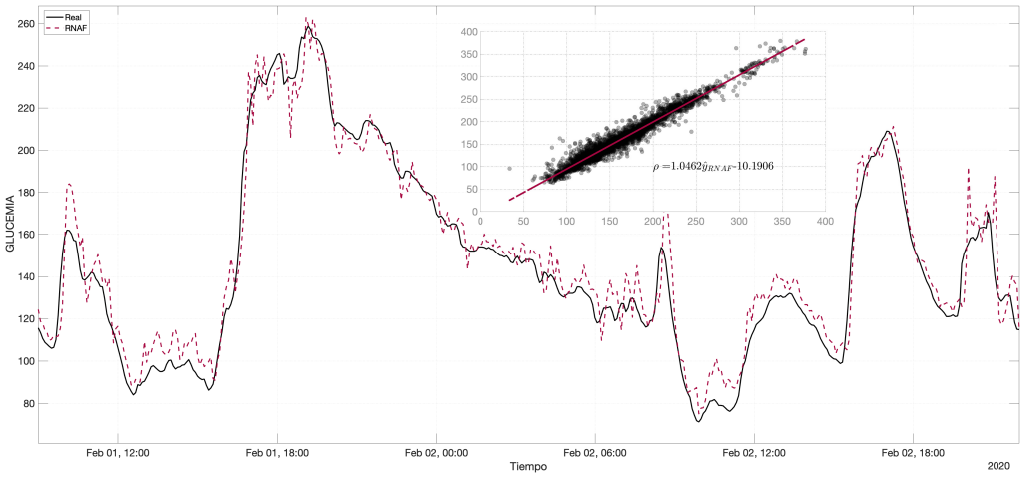


(a) Predicción realizada por la RNAF.

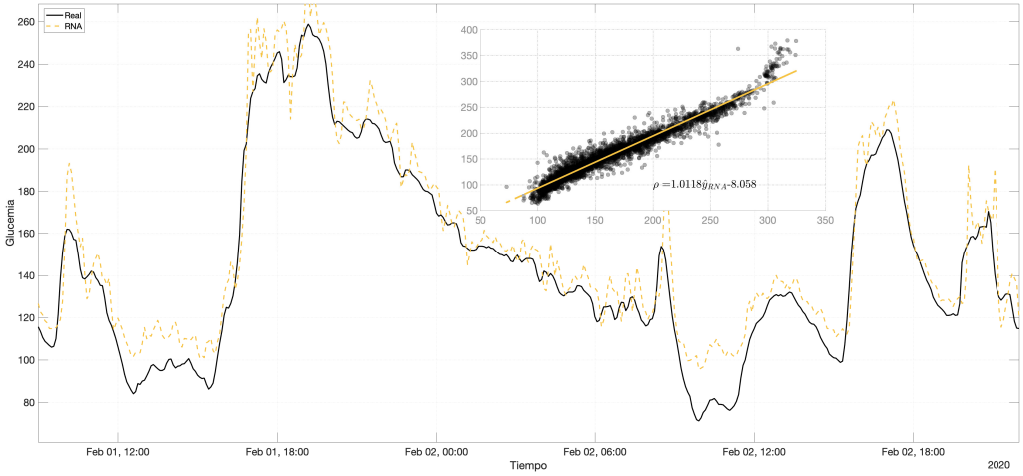


(b) Predicción realizada por la RNA.

Figura 5.15: Predicciones de 15 minutos a un paciente aleatorio seleccionado entre 30 por la RNAF y la RNA sobre periodos de desayuno, comida y cena.

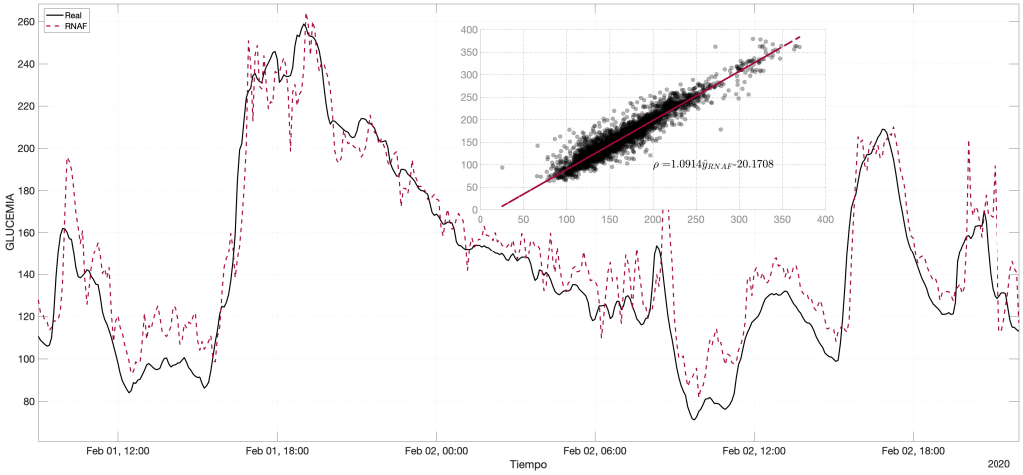


(a) Predicción realizada por la RNAF.

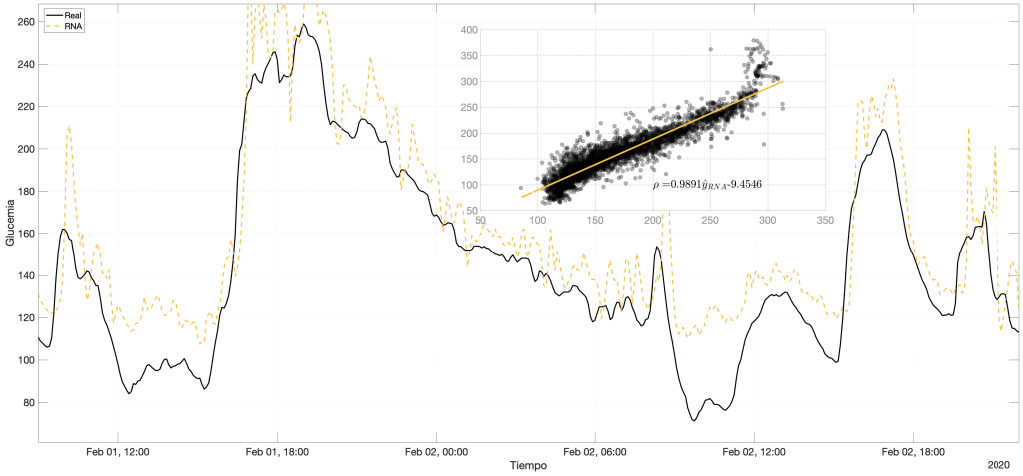


(b) Predicción realizada por la RNA.

Figura 5.16: Predicciones de 20 minutos a un paciente aleatorio seleccionado entre 30 por la RNAF y la RNA sobre periodos de desayuno, comida y cena.

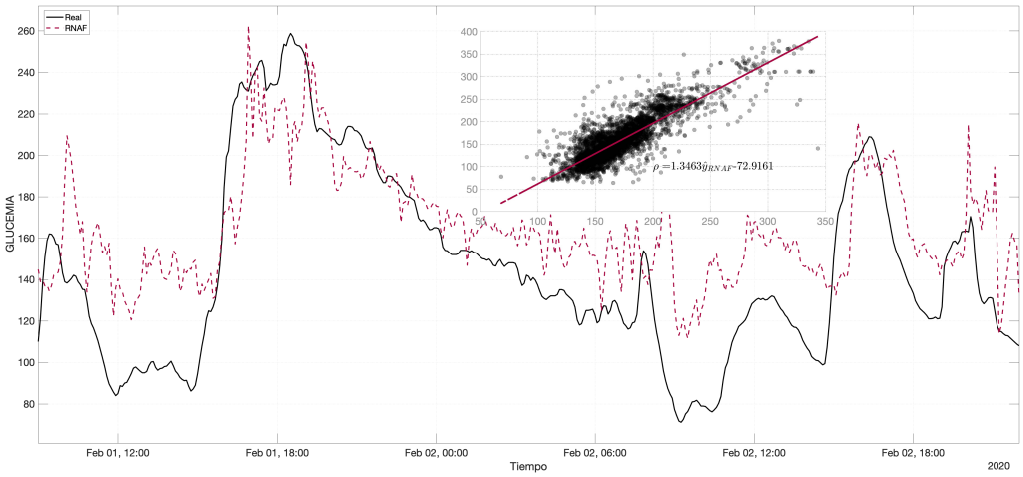


(a) Predicción realizada por la RNAF.

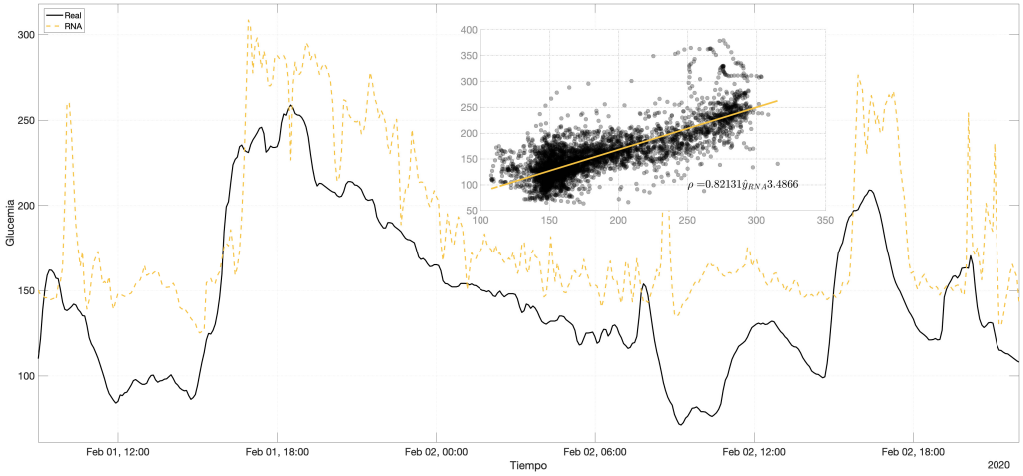


(b) Predicción realizada por la RNA.

Figura 5.17: Predicciones de 30 minutos a un paciente aleatorio seleccionado entre 30 por la RNAF y la RNA sobre periodos de desayuno, comida y cena.



(a) Predicción realizada por la RNAF.



(b) Predicción realizada por la RNA.

Figura 5.18: Predicciones de 60 minutos a un paciente aleatorio seleccionado entre 30 por la RNAF y la RNA sobre periodos de desayuno, comida y cena.

CAPÍTULO 6

Neuro-Adaptavilidad Fractal-Fraccionaria.

6.1. Introducción

Muchos de los controladores son diseñados con base en el comportamiento del modelo de los cuales, en los capítulos 1 y 3, se han explicado los dos métodos de obtener dicho modelo. En este contexto, se ha decidido aprovechar las ventajas de la IS para realizar el modelado de diversos sistemas [168, 169]. Como se ha mencionado con anterioridad, en este trabajo de investigación se ha optado por utilizar a las redes neuronales como modelo de optimización, es decir, estimar los parámetros de una RNA con el fin de simular el comportamiento físico de un sistema gracias a que la RNAs tienen la capacidad de aproximar funciones no lineales, capacidad de procesar grandes capacidades de información [118, 119, 170, 171]. Diversos trabajos de investigación encontrados en la literatura aseguran que las RNAs son eficientes para realizar el modelado de sistemas físicos [172], donde el modelo analítico es difícil de obtener[167, 173].

En [174, 175] se utiliza la definición de RL para la IS mediante modelos Hammerstein fraccionarios de sistemas simulados y sistemas reales como intercambiadores de calor. En [133] se utiliza una RNA utilizando un algoritmo de entrenamiento fraccionario, la no localidad del algoritmo permitió que el entrenamiento encontrase nuevas trayectorias de solución a pesar de iniciar con las mismas condiciones iniciales en cada simulación; el algoritmo fraccionario implementado ayudó a reducir el número de parámetros que el modelo necesita para estimar el comportamiento del sistema real en comparación de su contra parte convencional (orden entero). En [36] se propone una RNA con dinámica fraccionaria para realizar la IS de dos tanques acoplados, la RNA propuesta, es desarrollada utilizando la defunción de RL y mediante un algoritmo de aprendizaje fraccionario basado en las ecuaciones de Riccati y Lyapunov. En [176], basado en la teoría de Lyapunov, una RNA en el sentido de Caputo fue entrenada para la IS de un sistema simulado. En [177], una RNAF en el sentido de Caputo fue utilizada para la identificación de una turbina de viento utilizando el algoritmo PSO para su adaptación.

En el capítulo 1 se ha mencionado acerca de la definición geométrica que una derivada fraccionaria representa, la cual está definida como tiempo cósmico ya que la derivada fraccionaria permite encontrar soluciones en diferentes escalas de tiempo que el cálculo convencional es incapaz de caracterizar. Del mismo modo que el tiempo es medido en diversas escalas, el espacio también puede ser medido en **diferentes escalas de espacio**

definido como espacio fractal.

Como se ha mencionado al principio de este parrafo, el CF es capaz de adaptarse a diversos problemas donde, entre ellos, se encuentra la IS. Los trabajos encontrados en la literatura se basan en el concepto de derivada fraccionaria con kernel de potencia (RL y LC) sin embargo, no se han encontrado trabajos que relacionen el concepto de **fractalidad** para la IS.

6.2. Balance entre definiciones fraccionarias y fraccionarias-fractales

La ecuación (2.21) ha mostrado la definición fractal-fraccionaria con un kernel de potencia (FFP). El concepto matemático de fractalidad está dado por (2.22) cuya comprobación se ha demostrado en [65, 178, 179]. Cabe destacar que si el orden de fractalidad β es constante, la propiedad de fractalidad se expresa como la ecuación (6.1).

$$\frac{d}{dt^{\beta}}\Phi(t) = \frac{t^{1-\beta}}{\beta}\dot{\Phi}(t). \tag{6.1}$$

Si la igualdad (6.1) es utilizada en la ecuación (2.21) se llegaría a una definición como

$$\begin{aligned} \left({}_{t_0}^{FFP}\mathcal{D}_T^{\alpha,\beta} f(t)\right) &= \frac{1}{\Gamma(1-\alpha)} \underbrace{\frac{d}{dt^{\beta}}}_{\text{Fractalidad}} \int_{t_0}^{T} f(t) \underbrace{(t-\tau)^{-\alpha}}_{\text{Memoria}} d\tau, \\ \Rightarrow \left({}_{t_0}^{FFP}\mathcal{D}_T^{\alpha,\beta} f(t)\right) &= \frac{1}{\Gamma(1-\alpha)} \underbrace{\frac{t^{1-\beta}}{\beta}\frac{d}{dt}}_{\text{Fractalidad}} \int_{t_0}^{T} f(t) \underbrace{(t-\tau)^{-\alpha}}_{\text{Memoria}} d\tau, \end{aligned} \tag{6.2}$$

considerando que

$$\Phi(t) = \int_{t_0}^{T} f(t)(t-\tau)^{-\alpha} d\tau. \tag{6.3}$$

De la ecuación (6.2) se encuentra una relación entre la definición FFP y la definición de RL expuesta en la definición 1. Dicha relación se realiza ya que de la ecuación (6.2) se encuentra la definición de RL (ver ecuación (2.12)) implícitamente como se muestra en la ecuación (6.4).

$${}_{t_0}^{FFP}\mathcal{D}_T^{\alpha,\beta} f(t) = \frac{t^{1-\beta}}{\beta} \underbrace{\frac{1}{\Gamma(1-\alpha)}\frac{d}{dt}\int_{t_0}^{T} f(t)(t-\tau)^{-\alpha} d\tau}_{\text{Definición RL}}. \tag{6.4}$$

Por lo tanto, se encuentra una relación directamente proporcional entre la definición FFP y la definición de RL como lo muestra (6.5)

$${}_{t_0}^{FFP}\mathcal{D}_T^{\alpha,\beta} f(t) = \frac{t^{1-\beta}}{\beta} {}_{t_0}^{RL}\mathcal{D}_T^{\alpha} f(t). \tag{6.5}$$

Es importante denotar que la definición FFP es una generalización del caso fraccionario por ejemplo, si $\beta = 1$ se obtiene la derivada fraccionaria clásica de Riemann-Liouville pero, si el valor de beta es un numero racional, el comportamiento de la definición RL tendrá comportamientos fractales.

6.2.1. Balance entre definiciones fraccionarias

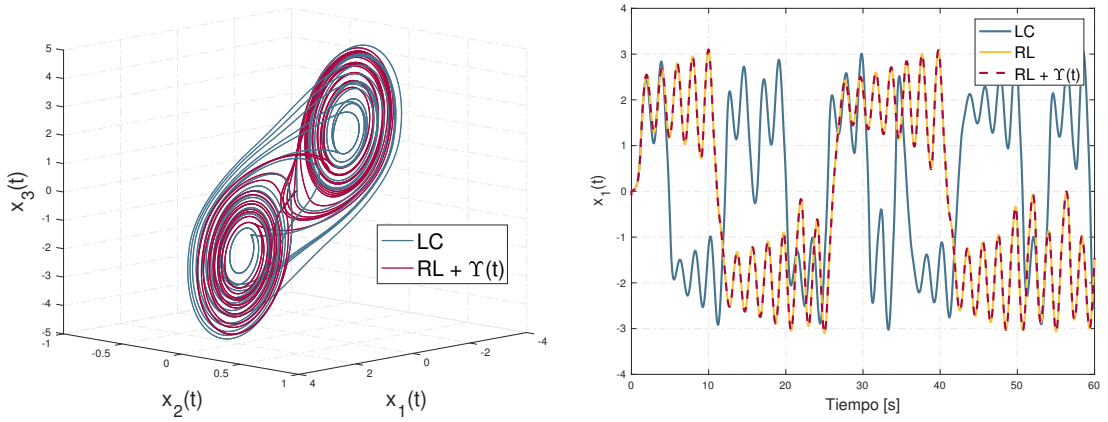
En diversos trabajos se propone una relación entre las dos derivadas (2.17) y (2.12). Por ejemplo, en [180, 181] muestran una relación entre las dos derivadas como se muestra en la ecuación (6.6).

$$\left({}_0^{LC}\mathcal{D}_t^{\alpha}x\right)(t) = \left({}_0^{RL}\mathcal{D}_t^{\alpha}x\right)(t) - \frac{t^{-\alpha}}{\Gamma(1-\alpha)}x(0) \tag{6.6}$$

En [140] se presenta la comprobación de la ecuación (6.6) pero, al realizar la simulación de dicha igualdad, la aproximación carece de precisión. Para demostrar la afirmación anterior se realizó la simulación de un atractor caótico conocido como *Circuito de Chua* gobernado por el sistema de ecuaciones diferenciales (6.7) [182].

$$\begin{aligned}
{}_0\mathcal{D}_{60}^{0.98}x_1(t) &= \alpha(x_2 - x_1 - a_2x_1 - 0.5(a_1 - a_2)(|x1+1| - |x1-1|),\\
{}_0\mathcal{D}_{60}^{0.98}x_2(t) &= x_1 - x_2 + x_3,\\
{}_0\mathcal{D}_{60}^{0.98}x_3(t) &= -\beta x_2.
\end{aligned} \tag{6.7}$$

La Figura 6.1a muestra el comportamiento del sistema caótico con cada no de sus estados. Como bien se aprecia, existe una diferencia entre el comportamiento del sistema con el sentido de LC y RL. En la Figura 6.1a no se alcanzan a apreciar todas las diferencias debido a la perspectiva con la que fue tomada la imagen. Por lo tanto se agregó el comportamiento de uno de sus estados en la Figura 6.1b con el fin de constatar que la compensación denotada como $\Upsilon(t)$ no es capaz de igualar ambas definiciones. En la Figura 6.1b se agregó el comportamiento de la definición de RL sin la compensación $\Upsilon(t)$ mostrando que, dicha compensación no aporta en lo absoluto al balance entre las dos definiciones.



(a) Comportamiento caótico del Circuito de Chua.

(b) Comportamiento temporal del estado $x_1(t)$ del Circuito de Chua.

Figura 6.1: Diferencias entre definiciones LC y RL.

Otra de las aproximaciones propuestas entre las dos derivadas consiste en una compensación mediante series de Taylor propuesto en [183] y expresado en (6.8).

$$\left({}_0^{C}\mathcal{D}_t^{\alpha}x\right)(t) = \left({}_0^{RL}\mathcal{D}_t^{\alpha}x\right)(t) - T_{m-1}[f;t_0](t), \tag{6.8}$$

En [183] muestran diversos ejemplos para probar que su aproximación es válida y útil pero, solo ha sido para aproximar funciones básica como x, e^x, $\sin(x)$, etc. Todas las formas

de compensación para relacionar ambas derivadas son interesantes pero, ninguna de ellas relaciona la fractalidad de una derivada fraccionaria como se muestra en la ecuación (6.9). Debido a la fractalidad añadida, se propone un término de compensación adaptable $\Phi(t_k)$ como se muestra en la ecuación (6.9).

$$\left({}_0^C\mathcal{D}_t^{\alpha}x\right)(t) = \frac{t^{1-\beta}}{\beta}\left({}_0^{RL}\mathcal{D}_t^{\alpha}x\right)(t) - \Phi(t_k), \tag{6.9}$$

donde $\Phi(t_k)$ es una red neuronal como se muestra en la ecuación (6.10) y Figura 6.2.

$$\begin{aligned}\Phi(t_k) = \varphi_3\left(Z_a\varphi_2\left(Va\varphi_1\left(\sum_{i_a=1}^{n_a} W_{a_{i_a}}\tilde{x}(k-i_a) + W_{ah}\right) + V_{ah}\right) + \right.\\ \left. Z_b\varphi_2\left(Vb\varphi_1\left(\sum_{i_b=1}^{n_b} W_{b_{i_b}}t(k-i_b-n_k) + W_{bh}\right) + V_{bh}\right) + Z_h\right),\end{aligned} \tag{6.10}$$

donde Z_a, Z_b, Z_h, Va, Vb, V_{ah}, V_{bh}, W_{ah} y $W_{bh} \in \mathbb{R}$, $W_a \in \mathbb{R}^{n_a}$, $W_b \in \mathbb{R}^{n_b}$ son pesos sinápticos no locales, φ_1, φ_2 yφ_3 son funciones de activación, t es el vector de tiempo y $\tilde{x}(t) = \left({}_0^C\mathcal{D}_t^{\alpha}x\right)(t) - \left({}_0^{RL}\mathcal{D}_t^{\alpha}x\right)(t)$.

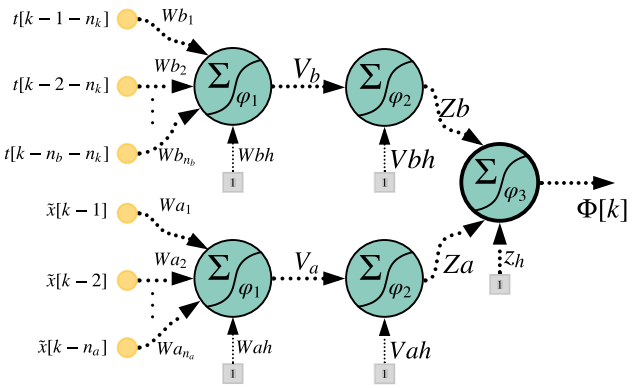


Figura 6.2: arquitectura de red neuronal fraccionaria compuesta por dos neuronas para procesar el tiempo y la diferencia entre ambas definiciones de derivada, dos neuronas en la segunda capa oculta y una neurona de salida..

Como se ha podido observar, la estructura de red neuronal se ha definido como una red neuronal fraccionaria debido a que los pesos sinápticos que forman a la red neuronal y que permiten su adaptabilidad, son entrenados mediante le implementación de algoritmos no locales, es decir, los pesos sinápticos tienen un comportamiento no local. Como se ha mostrado en el capítulo 5, se ha propuesto un algoritmo de aprendizaje basado en el gradiente de forma fraccionaria. Bajo la misma premisa se ha diseñado un algoritmo con capacidades fraccionarias y fractales considerando la ecuación (6.5), por lo tanto, el entrenaimiento fractal-fraccionario es como se propone en la ecuación (6.11).

$$
\begin{aligned}
Wa[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wa(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Wa[t(k)-j]\right],\\
Wb[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wb(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Wb[t(k)-j]\right],\\
Wah[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wah(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Wah[t(k)-j]\right],\\
Wbh[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Wbh(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Wbh[t(k)-j]\right],\\
Va[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Va(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Va[t(k)-j]\right],\\
Vb[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Vb(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Vb[t(k)-j]\right],\\
Vah[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Vah(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Vah[t(k)-j]\right],\\
Vbh[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Vbh(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Vbh[t(k)-j]\right],\\
Za[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Za(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Za[t(k)-j]\right],\\
Zb[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Zb(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Zb[t(k)-j])\right],\\
Zh[k+1] &= \beta t^{\beta-1}\left[-\eta[t(k)]\frac{\partial \mathcal{E}}{\partial Zh(t_k)}h^{\alpha}-\sum_{j=0}^{k}c_j^{(\alpha)}Zh[t(k)-j]\right].
\end{aligned}
\tag{6.11}
$$

6.3. Método Neuro-Adaptable Fractal-Fraccionario

El método neuro-adaptable está basado en la teoría de estabilidad establecida por Lyapunov [139]. Primeramente, debe considerarse un sistema no lineal como la ecuación (6.12):

$$
\Sigma_{NL} := \begin{cases} \left({}_0\mathcal{D}_t^{\alpha}x\right)(t) = f(t,x(t)) + g(t,u(t)), \\ y(t) = x(t), \end{cases} \tag{6.12}
$$

y una red neuronal adaptable como muestra la ecuación (6.13),

$$\Sigma_{RNDF} := \begin{cases} \left({}_0\mathcal{D}_t^{\alpha}\hat{x}\right)(t) = A\hat{x}(t) + Bu(t) + W_1\varphi_x(\hat{x}(t)) + W_2\varphi_u(u(t)) + \Psi_x + \Psi_u + Le, \\ \hat{y}(t) = \hat{x}(t), \end{cases} \tag{6.13}$$

donde $A \in \mathbb{R}^{n\times n}$, $A = A^T < 0$, $B \in \mathbb{R}^{n\times n_u}$, $B = B^T < 0$ y $L \in \mathbb{R}^{n\times n}$, $L = L^T > 0$, n_u representan el número de entradas del sistema, $W_1 \in \mathbb{R}^{n\times nn_{\hat{x}}}$ y $W_2 \in \mathbb{R}^{n\times nn_u}$ son matrices de pesos sinápticos, $\varphi_x \in \mathbb{R}^{nn_{\hat{x}}}$ y $\varphi_u \in \mathbb{R}^{nn_u}$ son funciones de activación, Ψ_x y Ψ_u son términos de compensación, y $nn_{\hat{x}}$ y nn_u son el número de neuronas para procesar señales de salida y entrada respectivamente.

Teorema 1. *Considere una red neuronal dinámica como la ecuación (6.13) y un sistema similar al mostrado en la ecuación (6.12). Los algoritmos de adaptación para los pesos sinápticos W_1 y W_2 que garantizan la estabilidad y convergencia del sistema en el sentido de LC son:*

$${}_0\mathcal{D}_t^{\gamma_1}W_1(t) = \varphi_1(\hat{x})e^T, \tag{6.14}$$

$${}_0\mathcal{D}_t^{\gamma_2}W_2(t) = \varphi_2(u)e^T. \tag{6.15}$$

Para realizar la prueba del teorema se debe considerar que la derivada fraccionaria cumpla con la regla de la cadena. Debido a que las derivadas de RL (ver ecuación (2.12)) y Fractal-Fractional (ver ecuación (2.21)) no cumplen con la regla de la cadena se utiliza una relación entre ambas considerando la derivada fraccionaria en el sentido de Caputo (ver ecuación (2.17)) con ayuda de un término de compensación adaptable como se muestra en la ecuación (6.9). De este modo, se puede seguir con la demostración del teorema sin perder las propiedades de fractalidad que la definición FFP otorga. El siguiente lemma muestra la comprobación de que la definición fraccionaria de Caputo cumple con la regla de la cadena:

Lemma 1. *Sea $x(t) \in \mathbb{R}$ una función continua y derivable. Entonces, para cualquier instante de tiempo $t \geq t_0$ [184]*

$$\frac{1}{2}{}_{t_0}^{C}\mathcal{D}_t^{\alpha}x^2(t) \leq x(t){}_{t_0}^{C}\mathcal{D}_t^{\alpha}x(t), \quad \forall\alpha \in (0,1]. \tag{6.16}$$

Demostración. Para probar que la ecuación (6.16) es verdadera, se expresa un equivalente como se muestra en la ecuación (6.17).

$$x(t){}_{t_0}^{C}\mathcal{D}_t^{\alpha}x(t) - \frac{1}{2}{}_{t_0}^{C}\mathcal{D}_t^{\alpha}x^2(t) \geq 0, \quad \forall\alpha \in (0,1]. \tag{6.17}$$

Utilizando la definición de la derivada de Caputo mostrada en la ecuación (2.17), las funciones se definen como:

$${}_{t_0}^{C}\mathcal{D}_t^{\alpha}x(t) = \frac{1}{\Gamma(1-\alpha)}\int_{t_0}^{t}\frac{\dot{x}(\tau)}{(t-\tau)^{\alpha}}d\tau, \tag{6.18}$$

$$\frac{1}{2}{}_{t_0}^{C}\mathcal{D}_t^{\alpha}x^2(t) = \frac{1}{\Gamma(1-\alpha)}\int_{t_0}^{t}\frac{\dot{x}(\tau)x(\tau)}{(t-\tau)^{\alpha}}d\tau. \tag{6.19}$$

Asociando las expresiones (6.18) y (6.19) se redefine la expresión (6.17) como

$$\frac{1}{\Gamma(1-\alpha)}\int_{t_0}^{t}\frac{[x(t)-x(\tau)]\dot{x}(\tau)}{(t-\tau)^{\alpha}}d\tau \geq 0. \tag{6.20}$$

Haciendo un cambio de variable se obtiene la ecuación (6.21).

$$\frac{1}{\Gamma(1-\alpha)}\int_{t_0}^{t}\frac{y(\tau)\dot{y}(\tau)}{(t-\tau)^{\alpha}}d\tau \leq 0. \tag{6.21}$$

Realizando una integración por partes se llega a la solución mostrada en la ecuación (6.22).

$$-\left[\frac{y^2(\tau)}{2\Gamma(1-\alpha)(t-\tau)^2}\right]\Big|_{\tau=t}+\left[\frac{y_0^2}{2\Gamma(1-\alpha)(t-t_0)^{\alpha}}\right]+\frac{\alpha}{2\Gamma(1-\alpha)}\int_{t_0}^{t}\frac{y^2(\tau)}{(t-\tau)^{\alpha+1}}d\tau \geq 0. \tag{6.22}$$

Utilizando la regla de L'Hopital se llega a la expresión mostrada en la ecuación (6.23).

$$\frac{y_0^2}{2\Gamma(1-\alpha)(t-t_0)^{\alpha}}+\frac{\alpha}{2\Gamma(1-\alpha)}\int_{t_0}^{t}\frac{y^2(\tau)}{(t-\tau)^{\alpha+1}}d\tau \geq 0. \tag{6.23}$$

Todas las expresiones de la ecuación (6.23) resultan positivas por lo que se concluye la prueba afirmando el **Lemma 1**.

□

Una vez que se ha demostrado que la regla de la cadena se cumple con la definición de Caputo se comienza a realizar la prueba del Teorema como sigue a continuación.

Demostración. Primeramente se define una función de Lyapunov como se muestra en la ecuación (6.24).

$$V(t)=\frac{1}{2}e^Te+\frac{1}{2}tr\left(W_1W_1^T\right)+\frac{1}{2}tr\left(W_2W_2^T\right). \tag{6.24}$$

Se define un error de predicción y su dinámica utilizando la derivada fraccionaria en el sentido de Caputo considerando que $\Delta_x := f(x) - A\hat{x}$ y $\Delta_u := g(u) - Bu$ como se muestra en la ecuación (6.25)

$$e := y-\hat{y} \Rightarrow {}_0^C\mathcal{D}_t^{\alpha}e(t)=\Delta_x+\Delta_u-W_1\varphi_1(\hat{x})-W_2\varphi_2(u)-\Psi_x-\Psi_u-Le, \tag{6.25}$$

sustituyendo la ecuación (6.25) en la ecuación (6.24), se tiene que:

$$\begin{aligned}{}_0^C\mathcal{D}_t^{\alpha}V(t) =& e^T\Delta_x+e^T\Delta_u-e^TW_1\varphi_1(\hat{x})-e^TW_2\varphi_2(u)-e^T\Psi_x-e^T\Psi_u-e^TLe+tr\left(W_1\dot{W}_1^T\right)\\ &+tr\left(W_2\dot{W}_2^T\right).\end{aligned} \tag{6.26}$$

Utilizando la desigualdad de normas de Cauchy-Schwarz y que $||\cdot||_2 \leq ||\cdot||_1$ se tiene que:

$$e^T\Delta_x \leq ||e^T||_1||\Delta_x||_2, \quad e^T\Delta_u \leq ||e^T||_1||\Delta_u||_2. \tag{6.27}$$

Considerando que $||\Delta_x|| \leq \delta_x$, $||\Delta_u|| \leq \delta_u$ y usando la propiedad de la norma uno $||e^T||_1 = e^T sign(e)$ se llega a las expresiones mostradas en la ecuación (6.28).

$$e^T\Delta_x \leq e^T sign(e)\delta_x, \quad e^T\Delta_u \leq e^T sign(e)\delta_u, \tag{6.28}$$

donde δ_x y δ_u son incertidumbres y se seleccionan con base en el conocimiento previo del sistema. Ahora, considerando que el producto de $e^T W_1 \varphi_1(\hat{x})$ y $e^T W_2 \varphi_2(u)$ es un escalar, es posible asociar lo mostrado en la ecuación (6.29).

$$\begin{aligned} tr\left(W_1 {}_0^C\mathcal{D}_t^\alpha W_1^T - W_1\varphi_1(\hat{x})e^T\right) &= tr\left(W_1\left({}_0^C\mathcal{D}_t^\alpha W_1^T - \varphi_1(\hat{x})e^T\right)\right), \\ tr\left(W_2 {}_0^C\mathcal{D}_t^\alpha W_2^T - W_2\varphi_2(u)e^T\right) &= tr\left(W_2\left({}_0^C\mathcal{D}_t^\alpha W_2^T - \varphi_2(u)e^T\right)\right). \end{aligned} \tag{6.29}$$

Sustituyendo las ecuaciones (6.28) y (6.29) en la función de Lyapunov (6.26) se tiene que:

$$\begin{aligned} {}_0^C\mathcal{D}_t^\alpha V(t) = e^T\left(sing(e)\delta_x - \Psi_x\right) + e^T\left(sing(e)\delta_u - \Psi_u\right) + tr\left(W_2\left({}_0^C\mathcal{D}_t^\alpha W_2^T - \varphi_2(u)e^T\right)\right) + \cdots \\ \cdots + tr\left(W_1\left({}_0^C\mathcal{D}_t^\alpha W_1^T - \varphi_1(\hat{x})e^T\right)\right) - e^T Le. \end{aligned} \tag{6.30}$$

Con el fin de eliminar términos mixtos, que hacen que la función candidata de Lyapunov no sea negativa definida, se realizan las siguiente igualaciones.

$${}_0^C\mathcal{D}_t^\alpha \hat{W}_1(t) = \varphi_1(\hat{x})e^T, \tag{6.31}$$

$${}_0^C\mathcal{D}_t^\alpha \hat{W}_2(t) = \varphi_2(u)e^T, \tag{6.32}$$

$$\Psi_x = sign(e)\delta_x, \tag{6.33}$$

$$\Psi_u = sign(e)\delta_u. \tag{6.34}$$

Las ecuaciones (6.31) y (6.32) son similares a las ecuaciones (6.14) y (6.15) respectivamente. Con los términos mixtos eliminados, la derivada de la función de Lyapunov se expresa como la ecaución (6.35).

$${}_0^C\mathcal{D}_t^\alpha V(t) = -e^T Le,\ L > 0. \tag{6.35}$$

Ya que la derivada de la función de Lyapunov (ver ecuación (6.35)) es negativa definida se concluye con el teorema. □

6.4. Resultados

En esta sección se presentan los diversos resultados obtenidos. El desempeño de la red neuronal es medido mediante la prueba de pendiente-intercepto donde idealmente, la pendiente debe ser igual a uno y el intercepto igual a cero. Con el fin de evaluar la efectividad del modelo neuronal propuesto, también se calculará el FIT (ver ecuación (4.14)), la norma del error (ver ecuación (4.13)) y la fórmula de Akaike (ver ecuación (5.26)). Las bases de datos utilizadas fueron divididas en dos partes, una para realizar el entrenamiento (70 % del total) y la segunda para realizar la validación (30 % del total). Dentro de la parte de entrenamiento, se divide en dos partes para realizar una estimación (70 % de los datos correspondientes al conjunto de datos de entrenamiento) y para pruebas (30 % de los datos de entrenamineto) con el fin de optimizar los parámetros de la red neuronal, ordenes fraccionarios, pesos sinápticos, parámetros libres (L), etc.

6.4.1. Sistema 1: Sistema de dos tanques en cascada

El objetivo es el de modelar el nivel de liquido de los dos tanques de un sistema de dos tanques conectados en cascada en escala. El esquemático del sistema se muestra en la Figura 6.3.

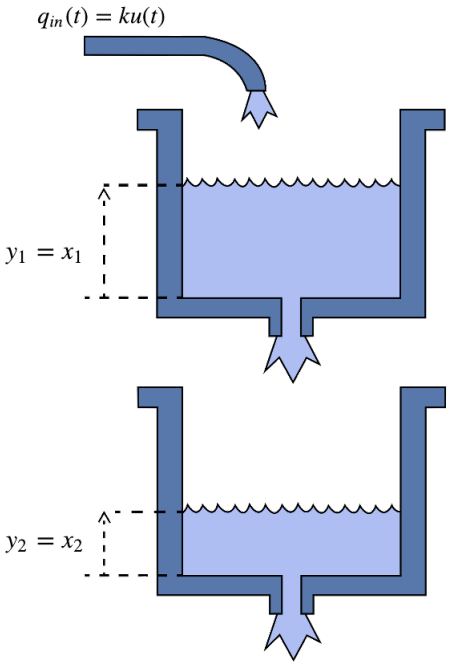


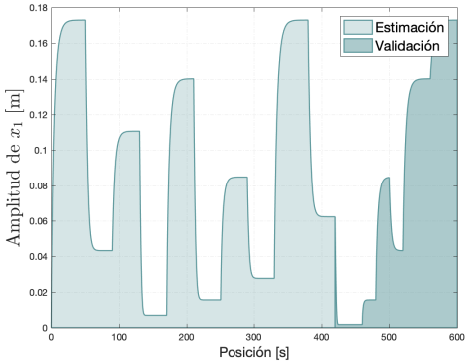
Figura 6.3: Sistema de dos tanques en cascada.

El problema original de la base de datos era el solamente saber el nivel de líquido del tanque inferior [185]. Con el fin de agregar un grado de dificultad mayor, se utilizaron las ecuaciones que gobiernan el comportamiento del nivel de líquido en tanques como se muestran en las ecuaciones (6.36). En la Figura 6.4 se muestran las señales de entrada y las dos señales de salida del sistema.

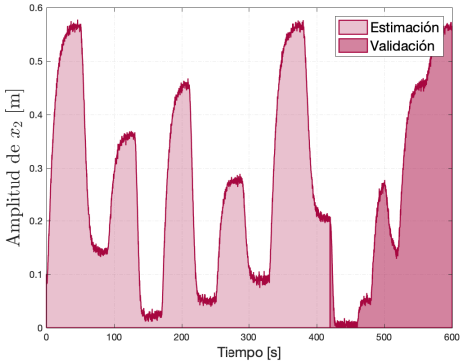
$$\begin{aligned} \frac{d}{dt}x_1(t) &= \frac{1}{A_1}\left(ku(t) - a_1\sqrt{2gx_1(t)}\right) \\ \frac{d}{dt}x_2(t) &= \frac{1}{A_2}\left(a_1\sqrt{2gx_1(t)} - a_2\sqrt{2gx_2(t)}\right) \\ y(t) &= [x_1(t),\, x_2(t)]^T \end{aligned} \tag{6.36}$$

Como se ha mencionado con anterioridad, el modelo neuronal mostrado en la ecuación (6.10) compensará la fractalidad de la derivada fractal-fraccionaria como se muestra en la ecuación (6.9). Se realizaron las aproximaciones de las diferencias entre los estados (diferencia entre FFP y Caputo) $x_1(t)$ y $x_2(t)$ con la red neuronal propuesta en la ecuación (6.10) considerando que para la RNDF mostrada en la ecuación (6.13) se tiene que $A = I$, $B = 2 * I$, $L = 75 * I$, $\delta_x = 0.0005$, $\delta_u = 0.00324$, $nn_x = 2$, $nn_u = 2$, $\alpha_1 = 0.791592$, $\alpha_2 = 0.877902$, $\gamma_1 = 0.98$ y $\gamma_2 = 0.96$. Los parámetros anteriores fueron optimizados mediante la implementación del algoritmo PSO.

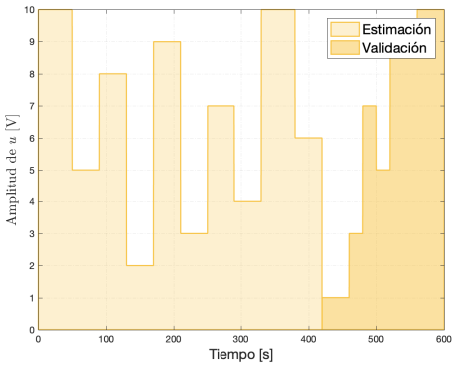
Las figuras 6.5a y 6.5c muestran las comparativas entre la derivada de Caputo y la derivada FFP con la compensación para los estados $x_1(t)$ y $x_2(t)$. Los resultados mostrados



(a) Datos de estimación y validación calculados mediante el modelo (6.36) para el estado $x_1(t)$



(b) Datos de estimación y validación para el estado $x_2(t)$.



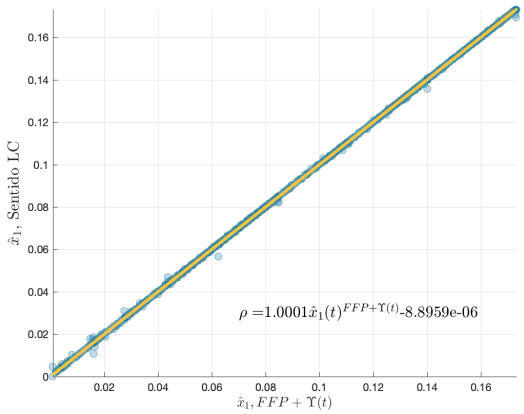
(c) Señal de entrada al sistema. Se muestran los datos utilizados para la estimación y la validación.

Figura 6.4: Datos de estimación y validación de cada salida $x_1(t)$ y $x_2(t)$, y la entrada $u(t)$.

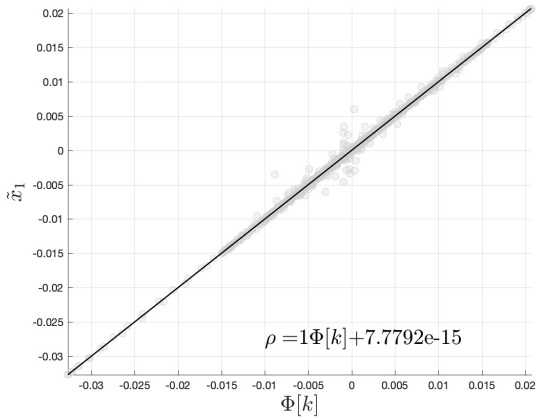
en las figuras 6.5b y 6.5d son referentes a la aproximación realizada por la red neuronal fractal-fraccionaria (NFF) al error de predicción de cada estado $\tilde{x}_1(t)$ y $\tilde{x}_2(t)$. Observando la prueba de pendiente-intercepto de la Figura 6.5, se percibe que la aproximación es valida ya que para todos los casos presentados, la pendiente es cercana a uno e intercepto cercano a cero o relativamente igual a cero.

Una vez que se han presentado las aproximaciones realizadas por la NFF de la ecuación (6.10), se muestran los resultados otorgados por la RNDF de la ecuación (6.13) cuyos parámetros se han mostrado con anterioridad. Las figuras 6.6a, 6.6b y 6.6c muestran las estimaciones realizadas para el estado $x_1(t)$ el cual, carece de ruido en la medición debido a que es el estado "simulado"por las ecuaciones (6.36). Del mismo modo pero con ruido en las mediciones, las figuras 6.6d, 6.6e y 6.6f muestran los resultados de la identificación del estado $x_2(t)$.

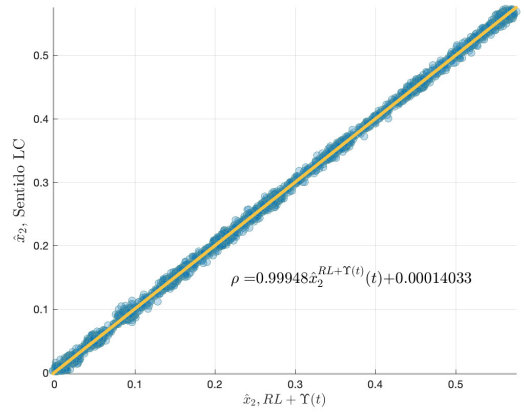
Nótese que de la pendiente-intercepto de los estados $x_1(t)$ y $x_2(t)$ en el sentido LC muestran una menor dispersión de los datos, es decir, la precisión de la derivada en el sentido LC (ver figuras 6.6b y 6.6e) es mayor que la derivada de FFP (ver figuras 6.6a y 6.6d) . De acuerdo a lo mostrado en las figuras 6.5b y 6.5d, el desempeño del conjunto de las redes neuronales artificiales (la compensación neuronal NNF mostrada en (6.10) y la RNDF mostrada en (6.13)) es similar al desempeño de la RNDF procesada con la derivada



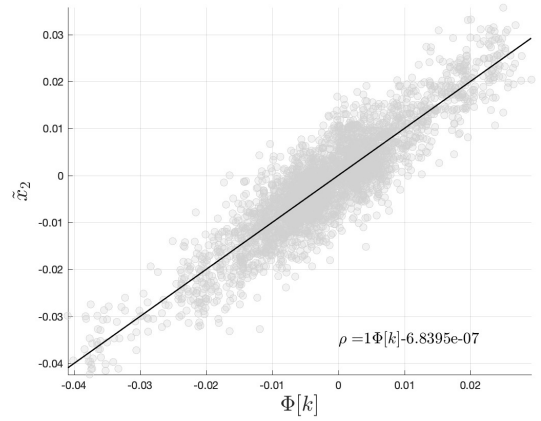
(a) Comparación entre la respuesta en sentido Caputo y la respuesta de Riemann-Liouville más la compensación neuronal FF para el estado $x_1(t)$.



(b) Pendiente-Intercepto de la salida NFF de la ecuación (6.10) para el estado $x_1(t)t$.



(c) Comparación entre la respuesta en sentido Caputo y la respuesta de Riemann-Liouville más la compensación neuronal FF para el estado $x_2(t)$.



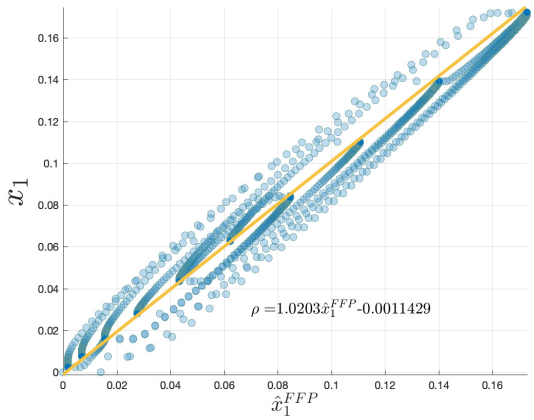
(d) Pendiente-Intercepto de la salida NFF de la ecuación (6.10) para el estado $x_2(t)t$.

Figura 6.5: Resultados obtenidos de las comparaciones con base en la ecuación (6.9) para los estados $x_1(t)$ y $x_2(t)$.

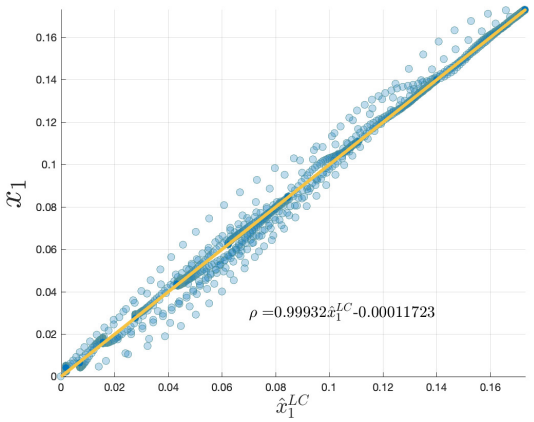
fraccionaria en el sentido LC.

Con fin de cuantificar el desempeño de las diferentes definiciones de derivada fraccionaria. Se han utilizado los índices mencionados al inicio de esta sección mostrados en la Tabla 6.1. De la Tabla 6.1 es posible observar que el desempeño de la definición de LC es similar al desempeño $FFP + NFF$ gracias a que se utiliza una red neuronal fractal NFF para poder compensar el comportamiento fractal de la definición FFP **esto implica que se defina una relación entre un comportamiento fractal (FFP) y una derivada fraccionaria (LC).** Del mismo modo, de la Tabla 6.1, la definición con un mejor desempeño son las basadas en Caputo y es totalmente justificable es decir, como se ha mencionado con anterioridad, la derivada de RL (que es la base de la derivada FFP) no tiene ciertas características importantes que una derivada convencional sí, como la derivada de una constante (que es diferente de cero) y más importante, que las condiciones iniciales deben definirse de manera fraccionaria. También se ha mencionado que se utiliza a GL para

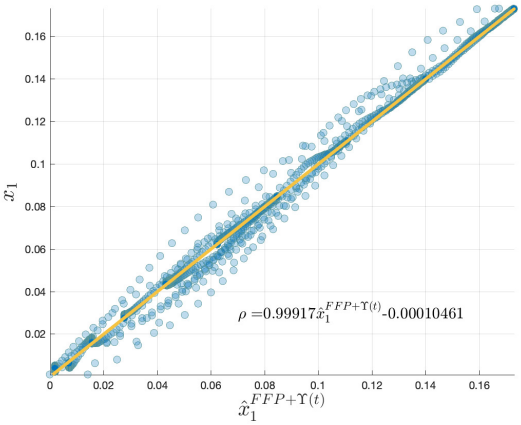
poder calcular el comportamiento de RL, en consecuencia, solo se pueden encontrar las soluciones a funciones simples como funciones algebraicas, exponenciales, trigonométricas, etc. [3, 21, 58].



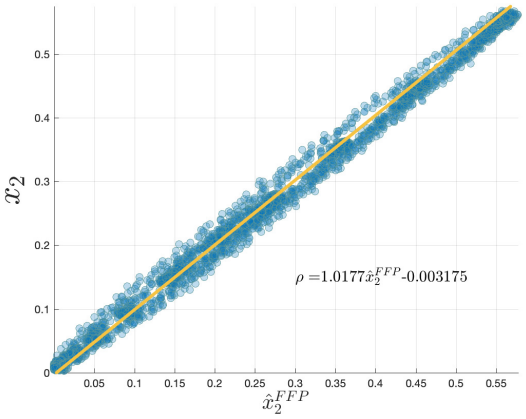
(a) Identificación del estado $x_1(t)$ mediante la definición de RL.



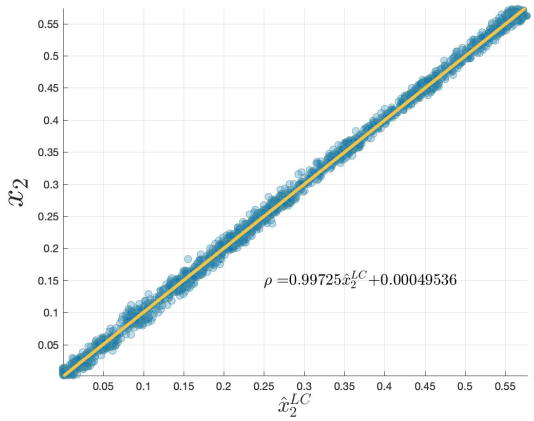
(b) Identificación del estado $x_1(t)$ mediante la definición de LC.



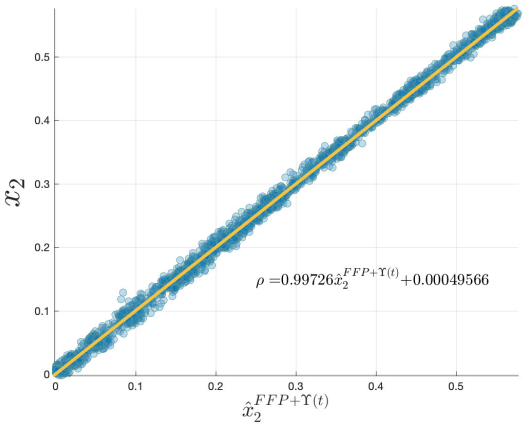
(c) Definición RL más compensación NFF para el estado $x_1(t)$



(d) Identificación del estado $x_2(t)$ mediante la definición de RL.



(e) Identificación del estado $x_2(t)$ mediante la definición de LC.



(f) Definición RL más compensación NFF para el estado $x_2(t)$

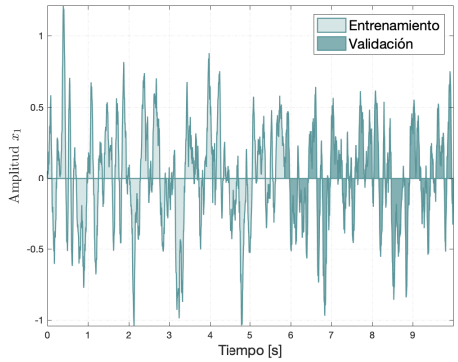
Figura 6.6: Desempeño de la estimación realizada por el conjunto de RNAFs (RNDF más NNF). Los resultados fueron obtenidos evaluando a la RNAFs con los datos de validación mostrados en la Figura 6.4

Tabla 6.1: Desempeño de cada definición de derivada fraccionaria utilizada.

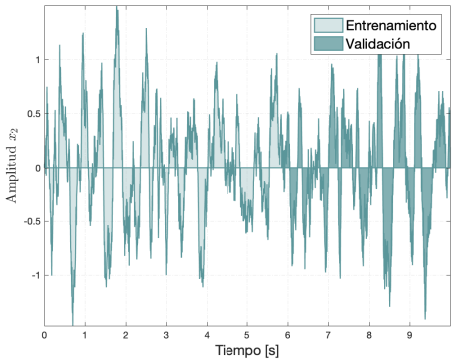
		FFP	LC	$FFP + NFF$
$x_1(t)$	$\|\|e\|\|$	0.2917	0.1619	0.1607
	FIT	91.12 %	95.04 %	95.1 %
	$Pendiente$	1.0203	0.99932	0.99917
	$Intercepto$	-0.00114	-0.000117	0.00010
$x_2(t)$	$\|\|e\|\|$	0.7366	0.4114	0.3753
	FIT	92.7349 %	95.95 %	96.29 %
	$Pendiente$	1.0177	0.99725	0.99726
	$Intercepto$	-0.003175	0.00049	0.00049

6.4.2. Sistema 2: Brazo reproductor de disco compacto

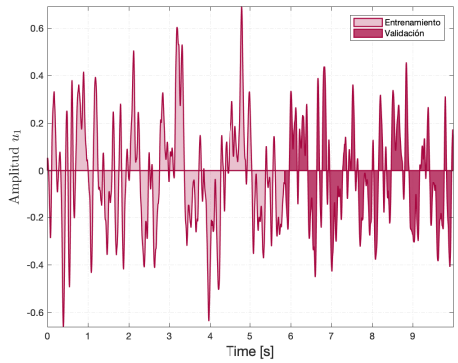
El siguiente sistema corresponde a un brazo mecánico de un reproductor de CD. Las señales de entrada y salida representan fuerzas ejercidas por actuadores y la posición del brazo respectivamente. La base de datos está compuesta por 2048 muestras, 1200 muestras son utilizadas para el entrenamiento y pruebas de la RNDF y una red de orden convencional RNC (caso clásico) [186]. El sistema mostrado en la Figura 6.7 contiene dos señales de entrada diferentes, por lo tanto la dimensión de "B" es diferente a las dimensiones utilizadas anteriormente. Bajo las mismas condiciones de simulación que el artículo propone, se evaluó el desempeño de la red neuronal propuesta.



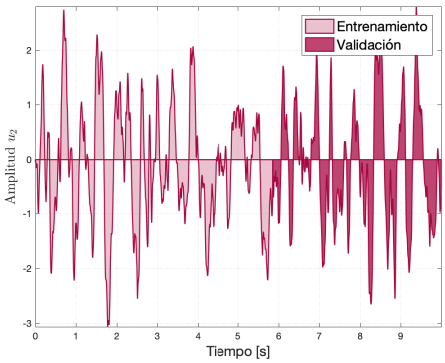
(a) Salida $x_1(t)$ del sistema.



(b) Salida $x_2(t)$ del sistema.



(c) Entrada $u_1(t)$



(d) Entrada $u_2(t)$

Figura 6.7: Datos de entrenamiento y validación para el modelo neuronal FFP y entero.

Tabla 6.2: Parámetros de la red neuronal fractal-fraccionaria y red neuronal de orden entero.

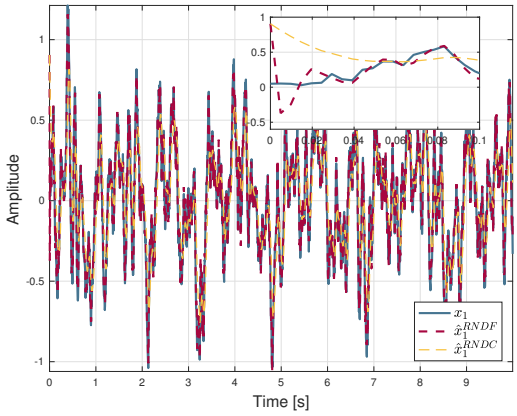
RNDF	RNC
$A = -3I$	$A = -15.463I$
$L = 4.514692I$	$L = 38.048236972I$
$B = -I$	$B = -2I$
$\delta_x = 0.0001$	$\delta_x = 0.0001$
$\delta_u = 0.1$	$\delta_u = 0.1$
$nn_x = 2$	$nn_x = 13$
$nn_u = 2$	$nn_u = 13$
$\alpha_1 = 0.8384526$	$\alpha_1 = 1$
$\alpha_2 = 0.98459318$	$\alpha_2 = 1$
$\beta = 0.927534$	$\beta = 1$
$\gamma = 0.99173$	$\gamma = 1$

Los parámetros de la RNDF y la RNC se muestran en la Tabla 6.2, donde nn_x representa el número de neuronas utilizadas para la estimación de las salidas, es decir $W_1 \in \mathbb{R}^{2\times 2}$, nn_u representa el número de neuronas necesarias para procesar la información referente a las señales de entrada con $W_2 \in \mathbb{R}^{2\times 2}$, α_i representa el orden fraccionario para los estados de salida, β el orden fractal utilizado para la compensación neuronal NFF, γ es el orden fraccionario del entrenamiento de los pesos sinápticos. La validación se realizó en los últimos 848 datos los cuales, son diferentes al conjunto de datos utilizados para realizar el entrenamiento como lo muestra la Figura 6.7. Al igual que en los casos anteriores, se realizó la prueba de pendiente-intercepto sobre los datos de validación como se muestra en la Figura 6.8 para la estimación del caso fractal-fraccionario (RNDF) y el caso clásico (RNC).

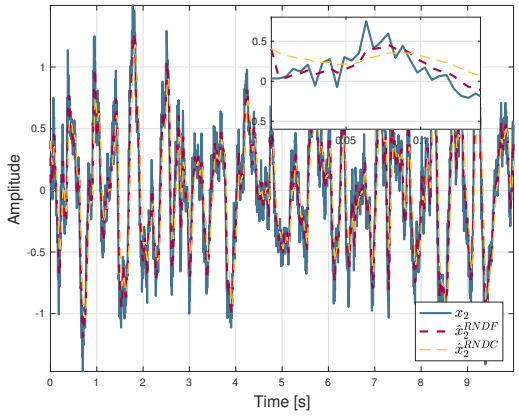
Las figuras 6.8a y 6.8b muestran el comportamiento de ambas redes neuronales (RNDF referente a la fractal-fraccionaria y RNC referida a la clásica). Se realizó un acercamiento al inicio de la simulación para denotar que ambos sistemas comienzan bajo las mismas condiciones iniciales pero, el sistema RNDF es capaz de adaptarse con mayor velocidad que el sistema de orden entero. Debido a que la prueba de pendiente-intercepto sobre los datos de validación muestran que ambos sistemas (RNDF y RNC) son aptos para representar el comportamiento del sistema real (ver figuras 6.8c, 6.8d, 6.8e, 6.8f), se ha decidido utilizar otros criterios de desempeño como los muestra la Tabla 6.3.

Tabla 6.3: Desempeño de ambos modelos neuronales (RNDF y RNC) y un modelo neuronal (MLP) encontrado en la literatura para el reproductor de discos.

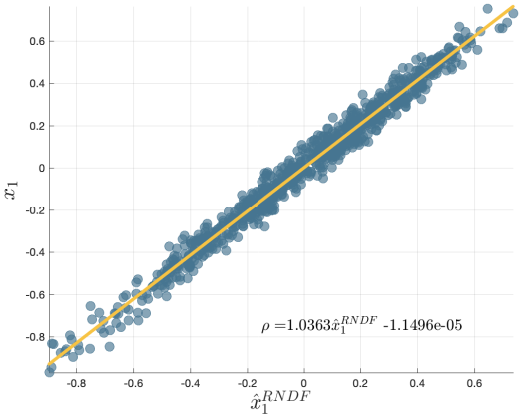
		RNDF	RNC	MLP [186]
x_1	FIT	85.15564 %	38.91198 %	66.0735 %
	$\|\|e\|\|$	2.40898	9.91354	-
	n_p	10	29	20689
	FPE	0.0022	0.0506	-
x_2	FIT	72.366808 %	44.775875 %	78.479213 %
	$\|\|e\|\|$	6.57414	13.13823	-
	n_p	10	29	20689
	FPE	0.0155	0.3006	-



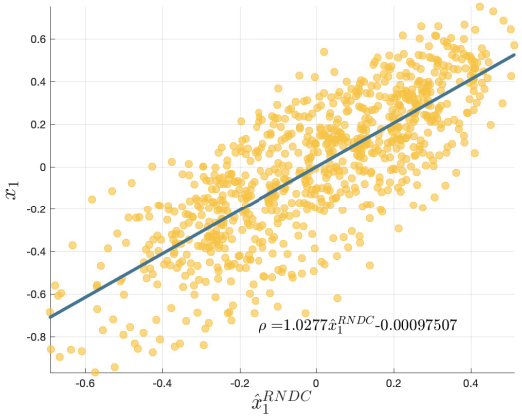
(a) Comportamiento temporal de los sistemas real, RNDF y RNC para el estado $x_1(t)$.



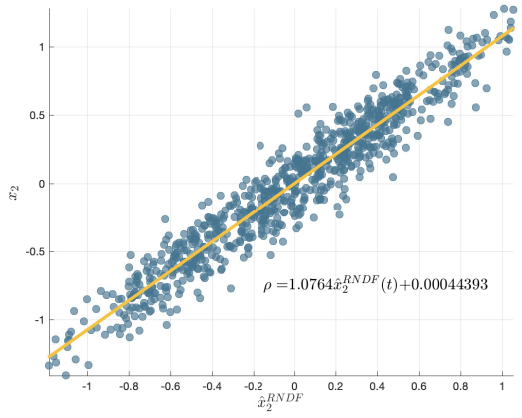
(b) Comportamiento temporal de los sistemas real, RNDF y RNC para el estado $x_2(t)$.



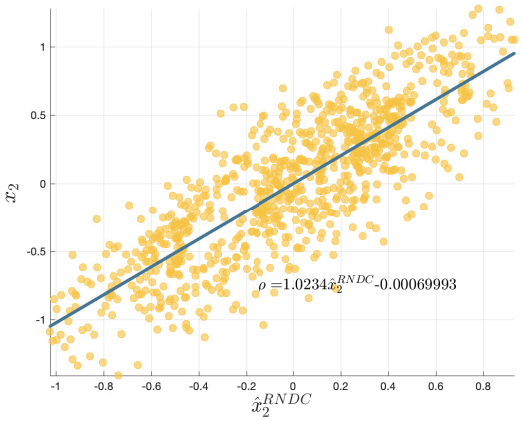
(c) Pendiente-intercepto de la RNDF contra el comportamiento del estado $x_1(t)$.



(d) Pendiente-intercepto de la RNC contra el comportamiento del estado $x_1(t)$.



(e) Pendiente-intercepto de la RNDF contra el comportamiento del estado $x_2(t)$.



(f) Pendiente-intercepto de la RNC contra el comportamiento del estado $x_2(t)$.

Figura 6.8: Resultados temporales sobre el total de la base de datos y prueba de pendiente-intercepto sobre los datos de validación.

Con ayuda de la Tabla 6.2, muestra que la RNDF necesita de dos neuronas para estimar el comportamiento del sistema mientras que la RNC necesita de 13 neuronas y por ende, el número de parámetros es distinto y con una diferencia alta como lo muestra la Tabla 6.3.

De la Tabla 6.3 se puede concluir que el modelo RNDF es superior en la mayoría de los criterios de desempeño con un número de parámetros mínimo para realizar la estimación. La excepción que no permite decir que la RNDF es superior en todos los sentidos, se encuentra en el FIT del estado $x_2(t)$, donde el modelo encontrado en la literatura de una MLP [186] ofrece una mejor aproximación pero, a costo de estimar un modelo neuronal MLP de 20689 (por cada estado) parámetros, mientras que la RNDF solo necesita 10 parámetros para estimar la misma salida; la diferencia del desempeño no es significativa por lo tanto, es aquí donde la decisión de qué modelo utilizar para la representación del sistema la debe tomar el usuario o el ingeniero en control.

Con el fin de probar el desempeño del algoritmo RNDF se realizaron 1000 repeticiones del sistema ajustando condiciones iniciales aleatorias por cada repetición en un rango entre cero y diez. El valor RMSE fue calculado para cada uno de los estados al igual que el tiempo de ejecución. Se consideró medir el tiempo de ejecución para denotar que este tipo de RNA permite procesar la información más rápido y compite con los algoritmos convencionales.

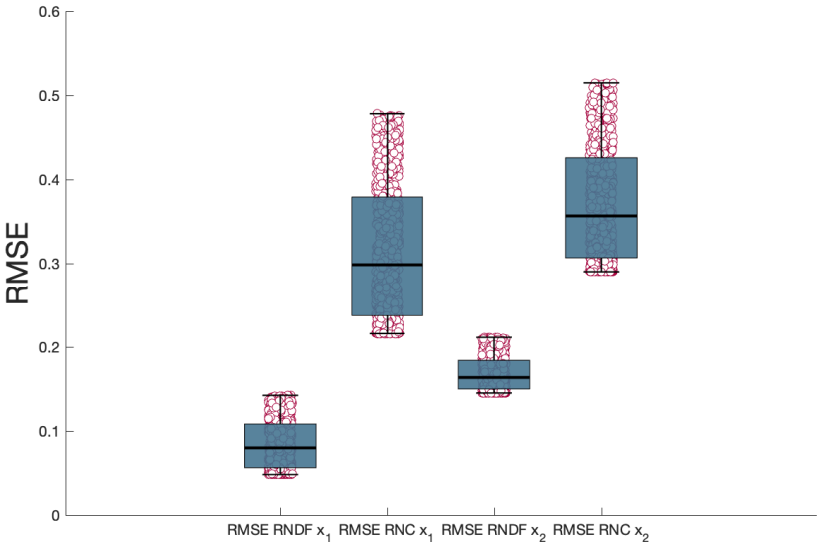


Figura 6.9: RMSEs obtenidos con los datos de validación después de 1000 repeticiones.

La Figura 6.9 muestra que a pesar de que el sistema RNDF ha sido evaluado con condiciones iniciales diferentes, en 1000 ocasiones sigue obteniendo un mejor resultado que el modelo de orden entero. Infortunadamente no es posible mostrar un diagrama similar a la Figura 6.9 para representar el tiempo de computo debido a que en ambos casos (RNDF y RNC) los resultados se ven con baja precisión. El tiempo de ejecución del algoritmo fue tomado mediante una computadora Macintosh con procesador 2.5 GHz Intel Core i7 y 16 GB 1600 MHz DDR3 de memoria RAM. Los procesos secundarios de la computadora fueron totalmente cerrados para ambas pruebas; el tiempo de ejecución promedio (*mean* $\pm$ *std*) por parte de la RNDF es de 0.1526 ± 0.0115 mientras que el modelo RNC es de 0.1591 ± 0.0154. Con los tiempos de ejecución mencionados, el modelo RNDF es capaz de competir con un modelo de orden entero probando que, el costo computacional (una de las desventajas de la aplicación del CF) es despreciable para esta estructura de RNA.

6.4.3. Sistema 3: Estimación de temperatura de un motor eléctrico

El siguiente sistema es un motor síncrono de imán permanente (PMSM del inglés permanent magnet synchronous motor) utilizado para un banco de pruebas. El PMSM representa un prototipo de un equipo de manufactura original. Las muestras fueron tomadas a 2 Hz. El conjunto de datos consta de diversas sesiones de pruebas diferenciadas por una etiqueta de identificación. La durabilidad de cada prueba es distinta entre una y seis horas [187, 188].

El motor es excitado por ciclos de manejo utilizando una velocidad y par de referencia. El sistema utiliza corrientes y voltajes en coordenadas d/q (para saber a qué se refieren las coordenadas d/q favor de ver el apéndice E). Estas señales son el resultado de una estrategia de control estándar con el fin de alcanzar las señales de referencia de velocidad y par mecánico. Las señales de velocidad y par son igualmente incluidas en el *benchmark*. Las señales de salida del sistema constan de la temperatura de superficie del rotor (pm o x_1), temperatura del diente del estator medida mediante un sensor térmico (s_t o x_2), la temperatura de la horquilla del estator (s_y o x_3), temperatura del devanado del estator (s_w o x_4) y el par mecánico inducido por la corriente de entrada. Las señales de entrada pertenecen al voltaje del componente d (u_d o u_1), voltaje del componente q(u_q o u_2), corriente del componente d (i_d o u_3), corriente del componente q (i_q o u_4) y la velocidad del motor (ω o u_5) como lo muestra el diagrama 6.10.

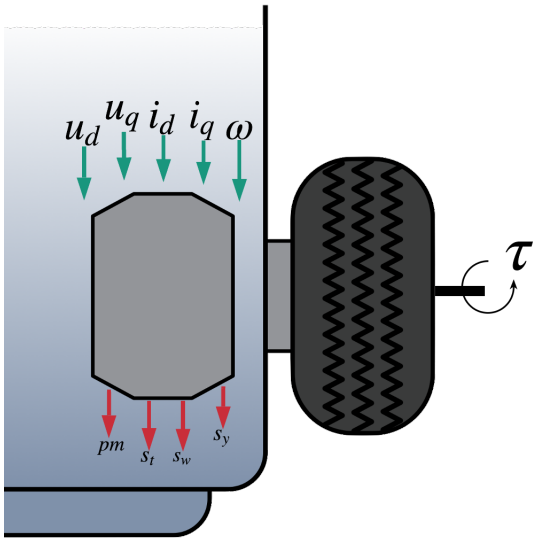
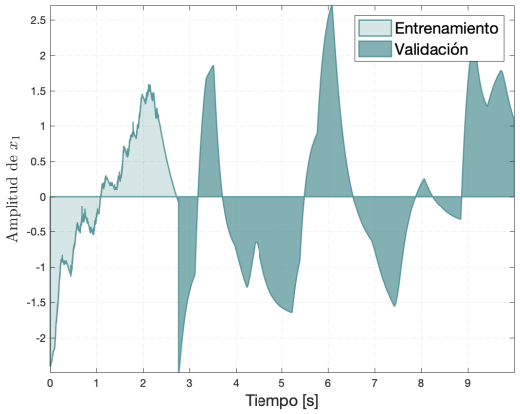
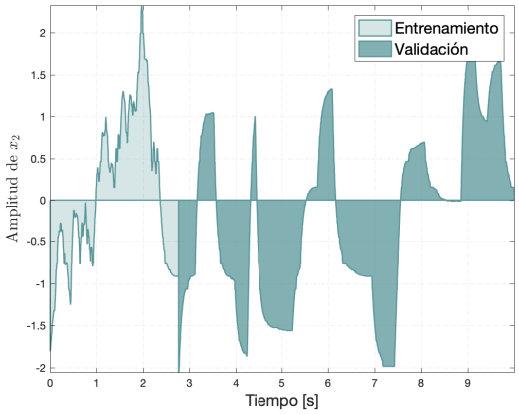


Figura 6.10: Esquemático de señales de entrada y salida del sistema.

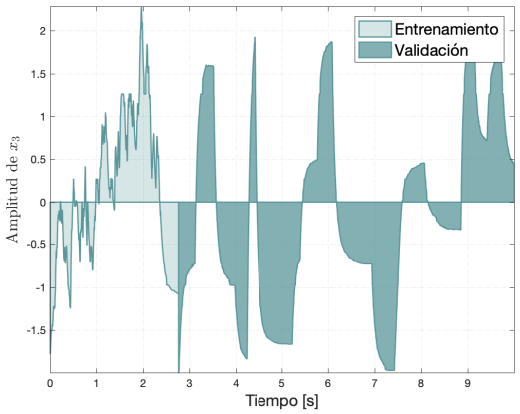
En los sistemas anteriores, se utilizaban una relación 70-30 y 60-40 para realizar el entrenamiento de las redes neuronales y la validación de las mismas. Debido a que la base de datos está compuesta por diversas sesiones de prueba, se consideró utilizar una de las sesiones seleccionada de manera aleatoria para realizar el entrenamiento y pruebas de las RNAs dando como resultado profileID = 73 compuesto por 16785 muestras. Del mismo modo, para realizar la validación del modelo se seleccionó una sesión de manera aleatoria dando como resultado profileID = 83 correspondiente a 43970 como se muestra en las figuras 6.11 (señales de salida) y 6.12. Cabe destacar que en total, se utilizarán 60755 muestras separadas en 27 % para realizar el entrenamiento y pruebas con el PSO y 72 % para realizar la validación.



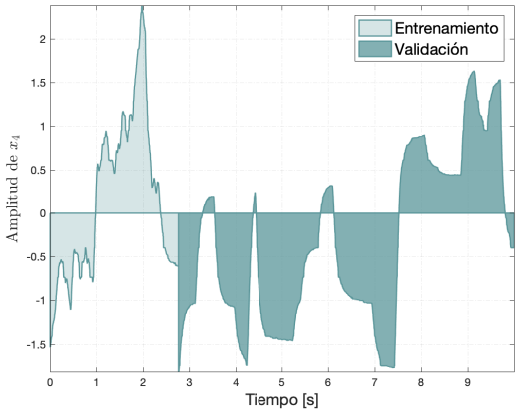
(a) Temperatura del rotor $x_1(t)$.



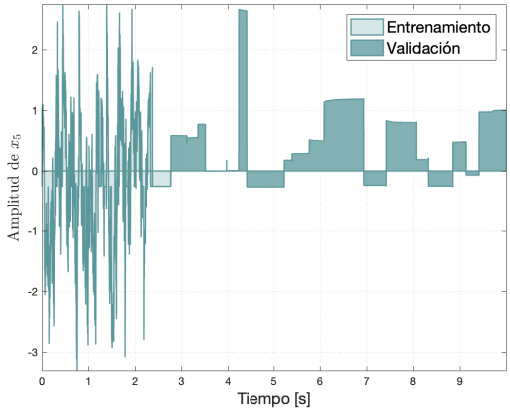
(b) Temperatura del diente del estator $x_2(t)$.



(c) Temperatura del devanado del estator $x_3(t)$.

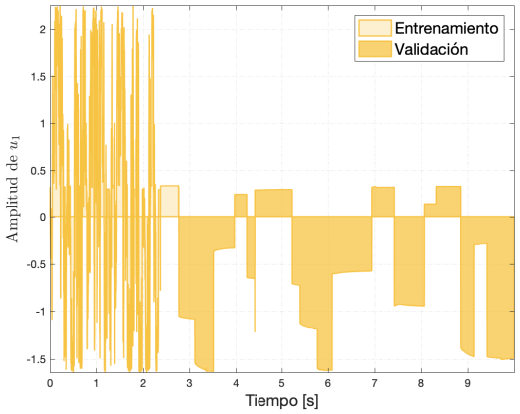


(d) Temperatura de la horquilla del estator $x_4(t)$.

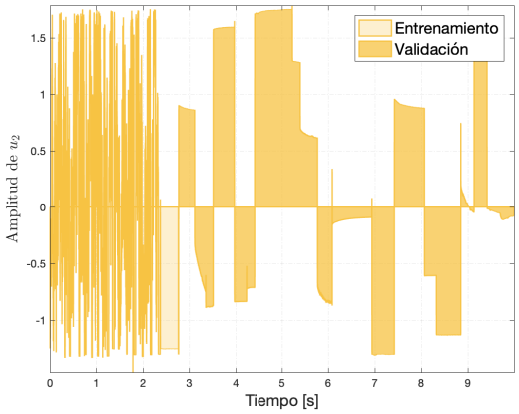


(e) Par mecánico $x_5(t)$.

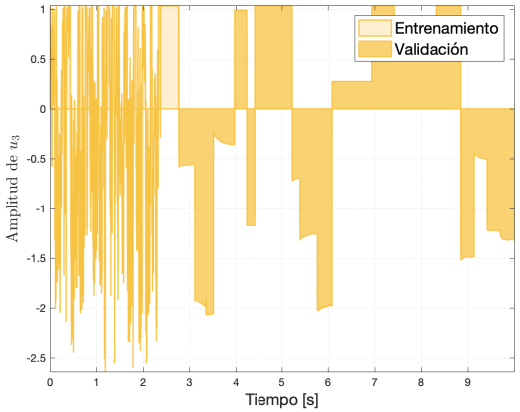
Figura 6.11: Salidas del sistema para realizar el entrenamiento y validación de las RNAs.



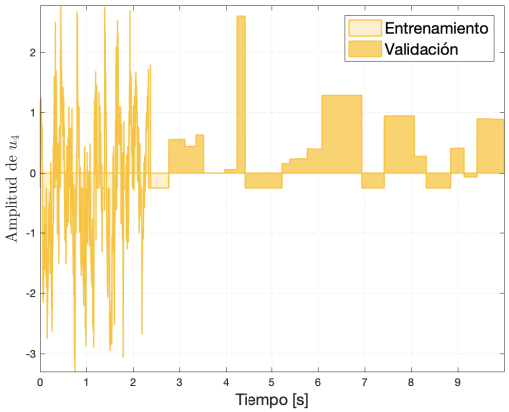
(a) Componente D de Voltaje $u_1(t)$.



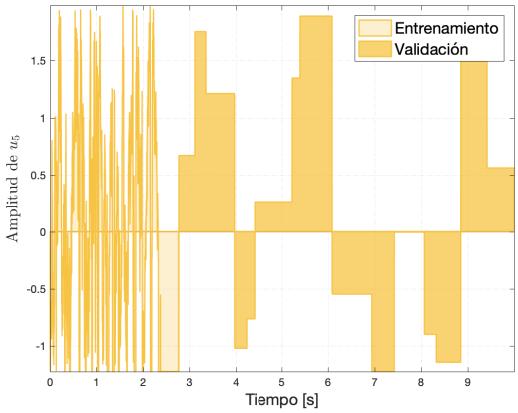
(b) Componente Q de Voltaje $u_2(t)$.



(c) Componente D de Corriente $u_3(t)$.



(d) Componente Q de Corriente $u_4(t)$.



(e) Velocidad del Motor $u_5(t)$

Figura 6.12: Señales de entrada para entrenamiento y validación de las RNAs.

Las señales de entrada-salida fueron normalizadas entre el valor máximo del valor absoluto de cada una de las señales con el fin de aprovechar la no linealidad de la función de activación. Cabe destacar que para este sistema se utilizaron funciones de activación del tipo $\tanh(\cdot)$ o variaciones de la misma.

Para las redes neuronales fractal-fraccionaria y clásica (RNDF y RNC) se obtuvieron los parámetros mostrados en la Tabla 6.4 mediante el PSO. Nótese que del 27 % de los datos

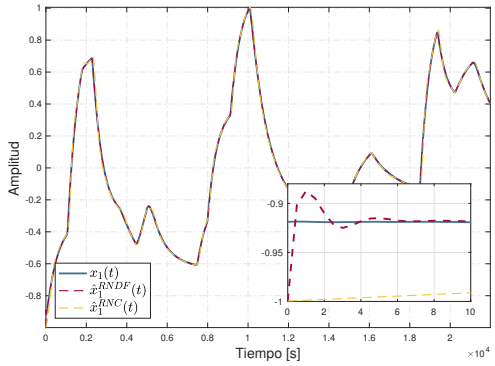
de entrenamiento se utiliza el 70 % del 27 % para realizar la estimación de los parámetros referente a 11749 muestras y el resto es utilizado para validar la optimización del PSO con 5035 muestras; es importante notar que en realidad las RNAs están siendo entrenadas con una quinta parte de 60755 muestras que conforman el total de los datos.

Las figuras 6.13a, 6.14a, 6.15a, 6.16a y 6.17a muestran el comportamiento temporal de cada una de las salidas conformadas por los datos de validación. Se realizó un acercamiento para denotar que las condiciones iniciales del sistema fraccionario y entero son iguales pero diferentes al sistema real. En dichas figuras, se observa que la RNDF tiene una respuesta más rápida que la RNC.

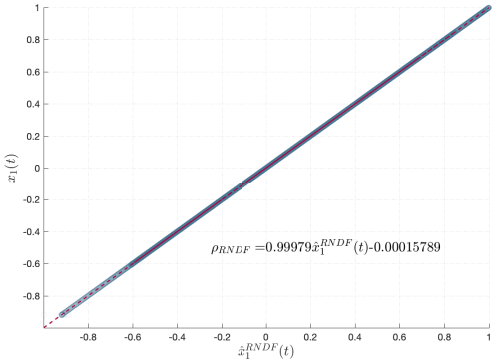
Las pruebas de pendiente-intercepto de las figuras 6.13b, 6.14b, 6.15b, 6.16b y 6.17b referentes a la RNDF muestran que el sistema fractal-fraccionario es altamente aprovechable. La exactitud de la respuesta es alta debido a que la comparación entre la respuesta de la RNDF y el estado de salida $x_i(t)$, $i = 1, 2, 3, 4, 5$ están uniformemente distribuidas. No obstante, el sistema de orden entero RNC de las figuras 6.13c, 6.14c, 6.15c, 6.16b y 6.17c muestra buenos resultados pero de acuerdo a la Tabla 6.5 necesita un número alto de parámetros ocasionando que la calidad del modelo RNC sea deficiente. La comparación entre los modelos es vasta, ya que se está comparando un modelo con un grado de eficiencia mayúsculo con un número de parámetros reducido con un modelo cuya eficiencia en altamente aceptable a costa de estimar un número de parámetros extenso.

Tabla 6.4: Parámetros de los modelos RNDF y RNC encontrados mediante el algoritmo PSO.

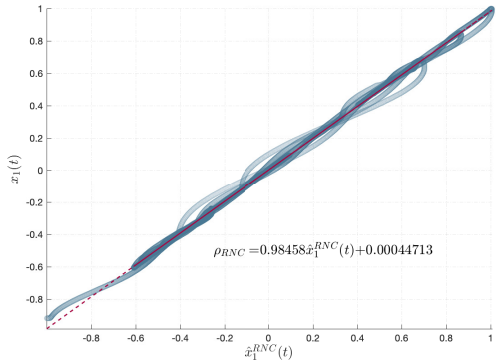
RNDF	RNC
$A = 51.136I$	$A = 491.263I$
$L = 5.315818 * I$	$L = 524.8673I$
$B = 0.8651I$	$B = 2.76814I$
$\delta_x = 0.00052730864$	$\delta_x = 0.00254306$
$\delta_u = 0.1$	$\delta_u = 0.1$
$nn_x = 5$	$nn_x = 20$
$nn_u = 5$	$nn_u = 20$
$\alpha = 0.9487$	$\alpha = 1$
$\beta = 0.8642$	$\beta = 1$
$\gamma = 0.9825$	$\gamma = 1$



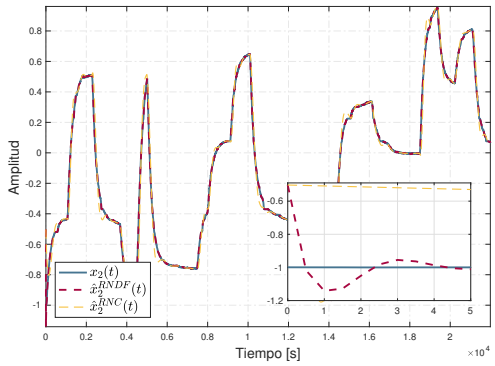
(a) Componente D de Voltaje $u_1(t)$.



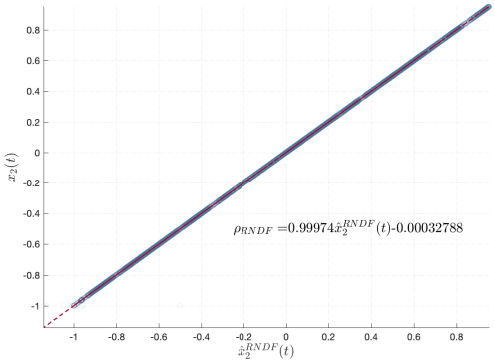
(b) Componente Q de Voltaje $u_2(t)$.



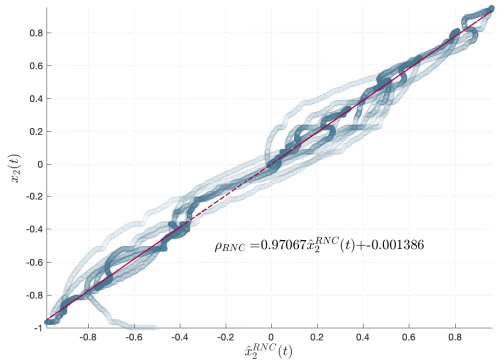
(c) Componente D de Corriente $u_3(t)$.



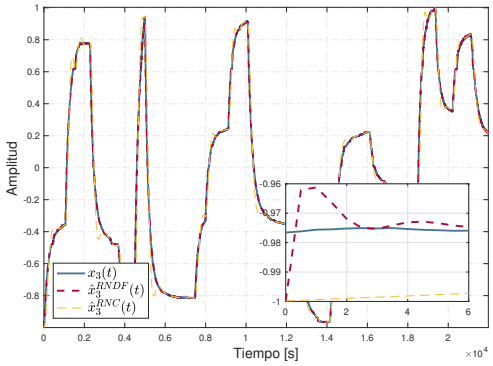
(a) Componente D de Voltaje $u_1(t)$.



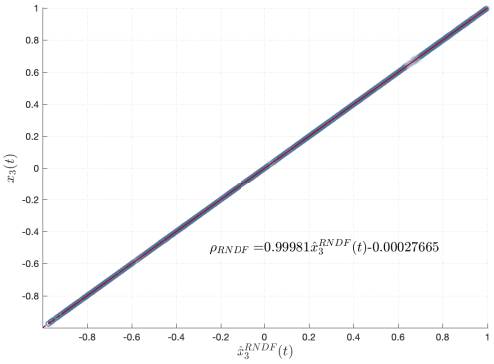
(b) Componente Q de Voltaje $u_2(t)$.



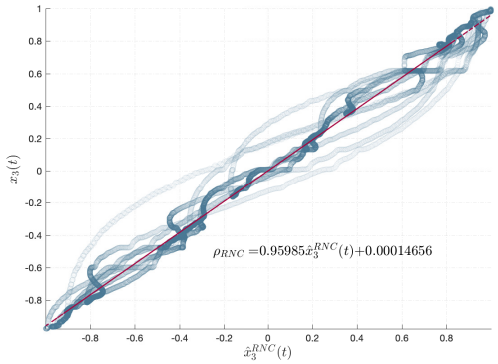
(c) Componente D de Corriente $u_3(t)$.



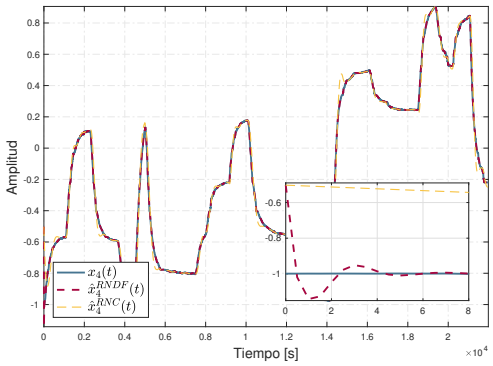
(a) Componente D de Voltaje $u_1(t)$.



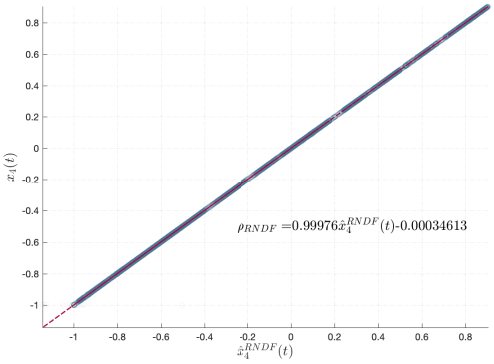
(b) Componente Q de Voltaje $u_2(t)$.



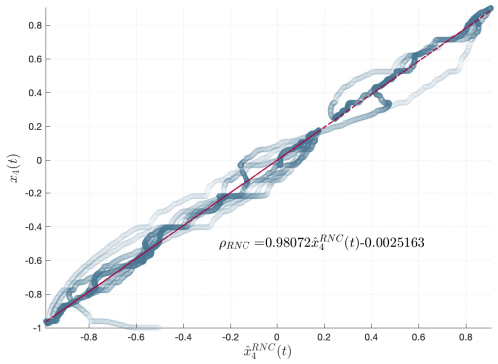
(c) Componente D de Corriente $u_3(t)$.



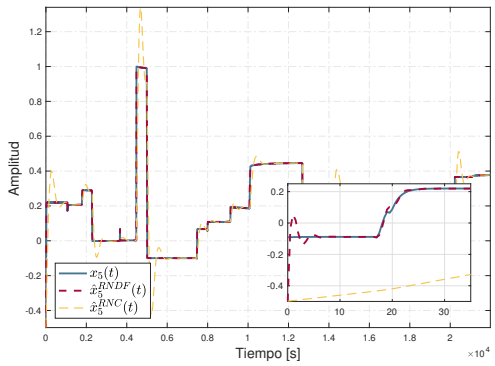
(a) Componente D de Voltaje $u_1(t)$.



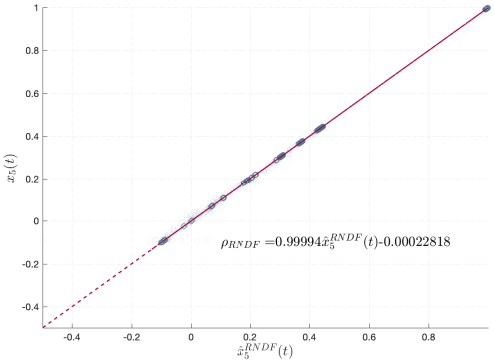
(b) Componente Q de Voltaje $u_2(t)$.



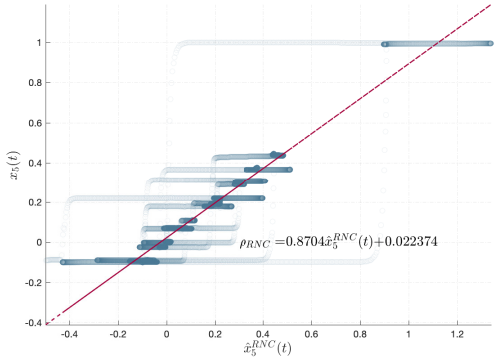
(c) Componente D de Corriente $u_3(t)$.



(a) Componente D de Voltaje $u_1(t)$.



(b) Componente Q de Voltaje $u_2(t)$.



(c) Componente D de Corriente $u_3(t)$.

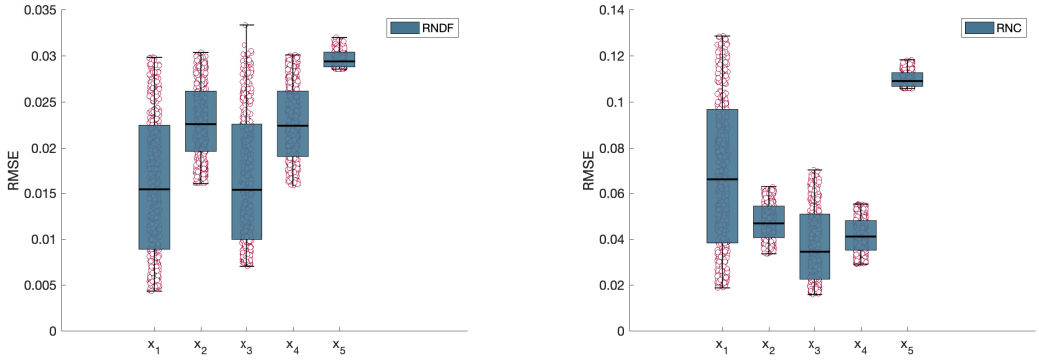
Tabla 6.5: Desempeño de ambos modelos neuronales (RNDF y RNC).

		RNDF	RNC
x_1	FIT	99.88 %	95.2059
	$\|\|e\|\|$	0.1041	4.4735
	n_p	15	23
	FPE	2.46×10^{-7}	4.56×10^{-4}
x_2	FIT	99.48 %	89.41 %
	$\|\|e\|\|$	0.5545	11.4886
	n_p	15	23
	FPE	6.99×10^{-6}	0.003
x_3	FIT	99.83 %	87.75 %
	$\|\|e\|\|$	0.1924	4.3246
	n_p	15	23
	FPE	8.42×10^{-7}	0.0047
x_4	FIT	99.59 %	91.02 %
	$\|\|e\|\|$	0.5533	10.1283
	n_p	15	23
	FPE	6.96×10^{-6}	0.0023
x_5	FIT	98.96 %	61.79 %
	$\|\|e\|\|$	0.4989	18.4308
	n_p	15	23
	FPE	5.66×10^{-7}	0.0077

La Tabla 6.5 muestra los desempeños obtenidos por ambas RNAs (RNDF y RNC). La

RNDF es superior en todos los sentidos a la RNA convencional. Para la RNDF se estiman un total de 75 parámetros mientras que para la RNC 115 parámetros. De acuerdo al criterio de Akaike [4], la calidad del modelo propuesto depende de un criterio de desempeño y el número de parámetros, es por ello que la eficiencia de la RNDF es mayor con respecto a la RNC.

Por último, se realizó el mismo experimento del sistema dos que consiste en realizar 1000 repeticiones cambiando las condiciones iniciales en cada iteración de forma aleatoria. Del mismo modo los parámetros de las RNAs fueron cambiados aleatoriamente pero cumpliendo las condiciones establecidas en el desarrollo matemático del método de adaptación neuronal.



(a) Desempeño de la RNDF.

(b) Desempeño de la RNC.

Figura 6.18: RMSE medido a través de realizar 1000 repeticiones con condiciones y parámetros aleatorios para las redes FFP y convencional.

Las figuras 6.18a y 6.18b muestran los resultados obtenidos de las 1000 repeticiones realizadas midiendo el valor RMSE del ecuación (5.27) para considerar el número de datos utilizados para realizar la validación. Del mismo modo, se calculó el costo computacional para cada una de las RNAs considerando el tiempo de computo como criterio de desempeño en una computadora Windows 10 un procesador RYZEN 5 2600 AMD 3.4Ghz 6 Cores Socket AM4 y 16 GB de memoria RAM DDR4 de 3000 MH. Al iniciar cada prueba se verificó que los procesos secundarios hayan sido cerrados, es decir, se verificó que la computadora solamente estuviera trabajando con el software de cálculo numérico MATLAB 2018b. Los resultados son mostrados en la Figura 6.19.

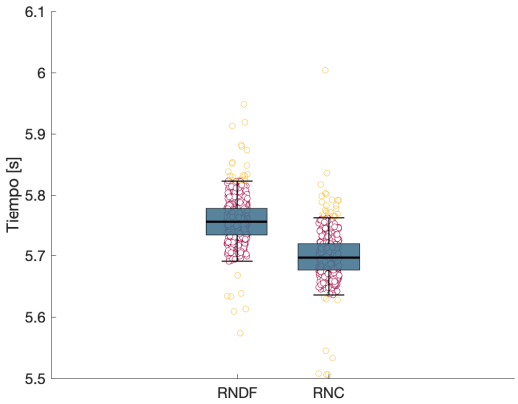


Figura 6.19: Costo computacional de cada red neuronal.

CAPÍTULO 7

Conclusiones y Trabajos Futuros

7.1. Conclusiones

Se ha probado que las RNA son capaces de realizar la identificación de sistemas físicos e inclusive, sistemas fraccionarios con un alto índice de precisión gracias a su capacidad de aproximar funciones no lineales mediante el uso de **funciones de activación** de carácter no lineal. También, gracias a que las RNA cuentan con la importante propiedad de adaptabilidad, es posible implementarlas en diversas tareas en cualquier tipo de ciencia donde se utilicen datos numéricos para describir un fenómeno. El entrenamiento de una RNA depende en gran medida del procesamiento de la información, ya que su comportamiento futuro dependerá altamente de la información observada por la misma red tal como si fuera un niño en etapas de desarrollo cognitivo. Lo anterior se establece ya que los niños en su etapa Pre-Operacional (entre dos y siete años) comienzan a tener y reconocer los cambios que sufre su vida cotidiana.

Con respecto al CF, la implementación del CF ha sido una perfecta justificación para modelar sistemas físicos gracias al efecto de memoria que este ofrece. Es normal que se tengan incertidumbres acerca del CF debido a que no se tiene una interpretación física o geométrica de él mismo pero, sí se han propuesto diversos conceptos que engloban al CF por ejemplo, el término de tiempo cósmico que se basa en una escala de tiempo diferente a la cotidiana, pérdidas de potencia, disipación de calor, fractalidades, entre otros, son términos que describen al CF de manera física [3, 58].

7.1.1. Conclusión capítulo 2

En este capítulo se presentaron algunos hecho históricos que dieron pie a que el CF se cimentara a como se le conoce hoy.

Se presentaron funciones que son útiles en el cálculo fraccionario. Por ejemplo la función de ML la cual, es importante en el CF ya que permite proponer soluciones analíticas de ecuaciones diferenciales fraccionarias, al igual que en caso clásico que utiliza funciones exponenciales para proponer soluciones de ecuaciones diferenciales ordinarias, la función de ML es la solución de las EDF ya que es capaz de generalizar a la función exponencial como se ha mostrado en la sección 2.2.2 y la ecuación (2.7) [189]. Al igual que la generalización ofrecida por la función de ML, la función Gamma expresada en la ecuación (2.4),

generaliza a los números factoriales. La diferencia entre la función Gamma y el operador factorial es que el operador factorial solo puede calcular el factorial de números enteros positivos mientras que la función Gamma es capaz de calcular el factorial de cualquier número complejo cuya parte Real no sea entera negativa.

A pesar de que se han presentado diversas definiciones de derivada fraccionaria, las derivadas de RL, GL y LC son la base del desarrollo del CF. Gracias a su formulación analítica mediante una extensión de la ecuación de Cauchy, la derivada de RL es la principal entre las tres definiciones mencionadas con anterioridad [58]. La necesidad de conocer las condiciones iniciales del sistema de manera fraccionaria ocasiona que la derivada de RL no sea tan utilizada para explicar fenómenos físicos.

La derivada de GL es similar a la derivada de RL aún y que su planteamiento fue totalmente diferente. La derivada de GL es capaz de proponer soluciones numéricas a EDF ya que su implementación es discreta y contiene las mismas características que la derivada de RL.

Por otro lado, la derivada de LC es la más utilizada en la ingeniería porque esta puede explicar de manera física el comportamiento de los fenómenos de la naturaleza debido a que requiere condiciones iniciales clásica al resolver ecuaciones diferenciales fraccionarias (al igual que las ecuaciones diferenciales ordinarias). Cabe destacar que se le conoce como "derivada artificial" a la definición de LC debido a que **no** fue formulada de manera analítica como sus iguales (RL y GL). La formulación de la derivada de LC permitió que la derivada de una constante fuera igual a cero (caso contrario a la derivada de RL) y que las condiciones iniciales del sistema fraccionario estén definidas bajo la física del sistema, es decir, se conoce la posición, velocidad, aceleración, etc., del sistema. Y finalmente, gracias a su estructura matemática, la derivada de LC, puede ser resuelta utilizando la ecuación integro-diferencial de Volterra.

A lo largo del capítulo se han presentado diversas definiciones de derivada fraccionaria. Cada una de las definiciones tiene características y tal vez «problemas» particulares. Uno de estos problemas son las singularidades que las definiciones de RL y LC tienen dentro de su kernel cuando $t = \tau$; el problema de singularidad fue expuesto y resuelto en [57, 61, 62] mediante la introducción de una función local y no singular como lo es la función exponencial dando así origen a la definición CFC.

En los últimos años se ha utilizado al cálculo fraccionario para generalizar soluciones o encontrar comportamientos anómalos que el cálculo convencional es incapaz de caracterizar. En este sentido, se presentó la definición de Atangana-Baleanu (ver ecuación (2.19)) la cual, es capaz de evitar singularidades y localidades mediante el uso de una función de ML en su formulación [102].

Las derivadas Fraccionarias-Fractales mostradas en las definiciones 7, 8 y 9 fueron propuestas en [64, 65] con el fin de aprovechar el comportamiento fractal de la derivada ordinaria que las definiciones de LC y RL necesitan para su formulación. El hecho de introducir un efecto fractal en la definición hace que se puedan explicar nuevos "comportamientos físicos" ya que no solo involucran el cambio de la función a través del tiempo sino del tiempo-espacio. Es importante destacar que el el cálculo diferencial utilizado en la vida cotidiana, es solamente un caso particular del CF y que el cálculo fraccionario es un caso particular del calculo Fraccionario-Fractal; la Figura 7.1 muestra un esquemático

de las generalizaciones mencionadas al igual que los casos particulares.

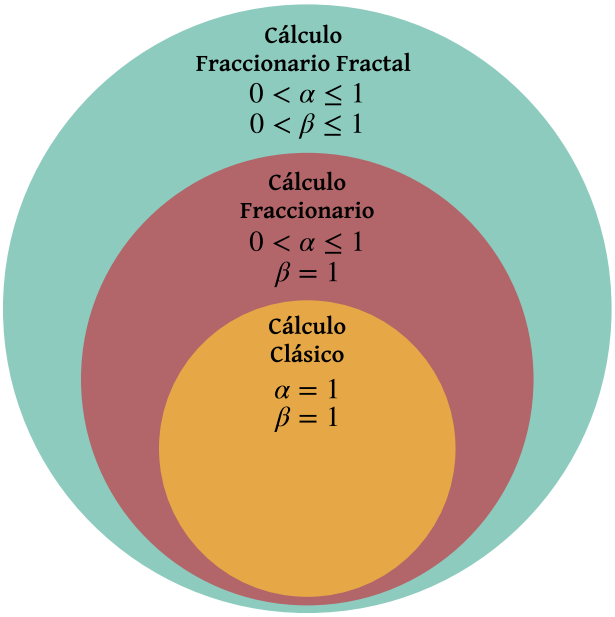


Figura 7.1: Casos generalizados y particulares del cálculo.

7.1.2. Conclusión capítulo 3

En este capítulo se presentó el concepto de neurona artificial y la analogía que esta tiene con una neurona biológica. Existe una relación estrecha entre la neurona artificial y biológica, desde su arquitectura hasta sus funcionalidades, es decir, una neurona biológica está constituida principalmente por un soma, dendritas, axón y el proceso de sinápsis que permite la comunicación con otras neuronas mientras que, una neurona artificial, analógicamente está formada un punto suma (análogo al soma), ramificaciones de entrada (análogas a las dendritas), una salida (análoga al axón) y pesos sinápticos que ponderan la comunicación con otras neuronas artificiales (análogos a la sinápsis). Desde otro punto de vista, las neuronas artificiales al igual que las neuronas biológicas, son capaces de distribuir información, procesar y aprender de la misma.

Se explicó que una RNA es el conjunto (en ocasiones complejo) de varias neuronas conectadas entre sí con el fin de imitar un comportamiento deseado. Se habló de arquitecturas realimentadas y retroalimentadas cuya principal característica es el efecto de memoria que las RNA retroalimentadas son capaces de procesar mediante la introducción de un elemento de retardo z^{-1}. Cabe destacar que existen diversas configuraciones de RNA más no arquitecturas, es decir, es posible encontrar configuraciones de tipo anillo, celulares, etc., pero estas son solo arquitecturas realimentadas o retroalimentadas.

Finalmente, el capítulo termina con la explicación del proceso de identificación de sistemas por medio de redes neuronales. A pesar de que en la actualidad existen modelos matemáticos propuestos de forma analítica, en ocasiones el proceso que conlleva es riguroso y tedioso ya que no solo es analizar la física que rodea al sistema sino estimar los parámetros que permiten transformar la energía en relación entrada-salida. Este tipo de complicaciones hace que el usuario se incline hacia el modelado por medio de la identificación de sistemas que solamente necesita saber la señal de entrada con la que se excita el sistema y la señal de salida producida por dicha entrada, se propone un modelo

matemático y se estiman los parámetros del modelo propuesto. Cabe destacar que el proceso de IS es similar al proceso de entrenamiento de una red neuronal ya que ambos necesitan dos señales (entrada y salida), sufren un procesos de optimización con el fin de que el modelo (ya sea neuronal o de otra clase) se ajuste a la señal deseada.

7.1.3. Conclusión capítulo 4

En este capítulo se ha presentado una metodología con base en la teoría de identificación de sistemas considerando al tiempo como una señal de entrada con el fin de producir la solución de EDFOV. Se presentaron diversos ejemplos para mostrar que la metodología propuesta es eficaz para sistemas simples como para sistemas complejos satisfaciendo una de las principales características de las EDO, la cual, establece que la trayectoria de la solución de una EDO es única y depende del la condición inicial. Para ello, se propuso una solución general que respeta la condición inicial de la ecuación diferencial y que está en función del comportamiento de una RNA. La RNA se adapta con el fin de completar la tarea asignada sin importar el tipo de EDF y más importante, el orden de derivación ya que puede ser constante o variable.

Los resultados obtenidos fueron comparados con el algoritmo más utilizado en la literatura gracias a su alto grado de precisión. El algoritmo de Adams-Bashfort-Multon (ABM) es capaz de generalizar inclusive al algoritmo numérico de Runge-Kutta que es uno de los más precisos. Los resultados de las comparaciones fueron prometedores con respecto a las mediciones de la norma del error, FIT y la prueba de pendiente-intercepto.

Una de las comparaciones más importantes que diferencia la metodología propuesta con el algoritmo ABM es el costo computacional, en cada ejemplo mostrado se midió el costo computacional necesario para otorgar una solución, para ello, se realizó el procedimiento (referente a entregar la solución de la EDF) 1000 veces registrando el tiempo que una computadora con procesador i5 y 16GB de RAM necesita para completar la tarea. En conclusión, dados los ejemplos mostrados, la metodología propuesta con la RNA necesita un costo computacional más pequeño que el uso del algoritmo de ABM gracias a su rapidez para procesar grandes cantidades de información.

El desarrollo de esta metodología se vio reflejado en publicaciones en revistas indexadas como:

- *Solving fractional differential equations of variable-order involving operators with Mittag-Leffler kernel using artificial neural networks* [110].
- *New numerical approximation for solving fractional delay differential equations of variable order using artificial neural networks* [111].
- *A novel method to solve variable-order fractional delay differential equations based in lagrange interpolations* [190].

7.1.4. Conclusión capítulo 5

Se ha propuesto un nuevo algoritmo de optimización para entrenar redes neuronales. El algoritmo está basado en el descenso por el gradiente utilizando la derivada fraccionaria de GL. Se ha presentado un criterio de estabilidad para garantizar que el algoritmo es estable. Se comprobó mediante ejemplos de diversos sistemas que el hecho de que a los pesos sinápticos de la RNA se le otorguen componentes no locales ha permitido que el

número de parámetros disminuya.

Se ha mencionado que la trayectoria de solución de una ecuación diferencial depende de las condiciones iniciales de la misma, por otro lado, en el CF, la no localidad permite encontrar diferentes trayectorias de solución con las mismas condiciones iniciales gracias al parámetro "libre" de la definición de derivada fraccionaria α. Con la introducción de α, se tienen grados de libertad infinitos lo cual no es posible con el cálculo diferencial convencional. Al agregar efectos no locales a la estimación de los pesos sinápticos se genera, en consecuencia y debido a que una RNA está conformada por pesos sinápticos, que la RNA tenga efectos no locales por lo que se ha introducido el concepto de RNAF como una red neuronal artificial fraccionaria.

Para mostrar el desempeño del algoritmo fraccionario propuesto, tres sistemas fueron identificados (dos benchmarks y un sistema médico). Los resultados obtenidos demostraron que las estimaciones realizadas por la RNAF fueron mejores y con un número de parámetros reducido.

En el caso del sistema médico, la optimización del orden de derivación fraccionario (al igual que en los demás experimentos), se realizó mediante el PSO. El PSO optimizó un orden cercano a la unidad haciendo que tuvieran desempeños similares o que la diferencia entre estos no fuera vasta pero, es importante notar que el hecho de realizar una pequeña variación al orden de la derivada permitió reducir casi seis veces el número de parámetros del modelo convencional. Para este sistema, la RNAF no obtuvo gran diferencia en el desempeño obtenido debido a que de acuerdo al resultado del PSO ($\alpha \to 1$) no se puede optimizar lo que ya está optimizado.

La metodología propuesta fue publicada en una revista indexada dando como resultado la siguiente referencia:

- *Fractional order neural networks for system identification* [133]

7.1.5. Conclusión capítulo 6

Se presentó una nueva metodología de adaptación de redes neuronales para la identificación de sistemas aprovechando la fractalidad de la definición (FFP). Entre los resultados se muestra que la eficiencia del algoritmo propuesto es capaz de igualar o mejorar la adaptabilidad del caso clásico, es decir, cuando el orden de derivación es igual a uno. Se realizaron diversas pruebas empíricas para poder seleccionar el número de neuronas para cada modelo propuesto (FFP y clásico) con el fin de encontrar el modelo óptimo para cada metodología. También se propuso una optimización por enjambre de partículas para encontrar los parámetros óptimos necesarios para la red neuronal. Algunos de los parámetros de la red neuronal fueron seleccionados empíricamente como δ_x y δ_u que son incertidumbres entre el modelo real (sistema real) y el modelo propuesto.

Debido a que se buscaba aprovechar la fractalidad de la derivada FFP y encontrar una relación entre la derivada FFP y Caputo. Se propuso una compensación neuronal que ayudaría a encontrar esa relación. La compensación neuronal fue propuesta por una red neuronal de tres capas separadas para ajustar el margen error. Con el fin de "modelar" el efecto de fractalidad, es decir que la compensación neuronal tuviera naturaleza fractal, se entrenó a la red neuronal que conforma a la compensación neuronal con un algoritmo de optimización fractal. Este proceso se llevó a cabo en cada uno de los sistemas presentados y se muestra en las tablas de parámetros como β.

Se presentó la estimación de diversos sistemas los cuales fueron analizados de distintas maneras es decir, el primer sistema está constituido por una sola señal de entrada con dos salidas siendo que el sistema original solamente otorga el comportamiento de una salida; para obtener el comportamiento de la salida faltante, se analizó la física del sistema para obtener la salida faltante. El segundo y tercer sistema presentan múltiples salidas y multiples entradas (MIMO) ocasionando que el rango de búsqueda se amplíe por parte del PSO. Según la prueba de pendiente-intercepto ambos modelos son aceptables, por lo tanto se propuso medir el desempeño mediante el FIT, la norma del error y el FPE con el fin de tener un criterio cuantitativo de la selección del modelo.

En cada uno de los experimentos se realizaron 1000 repeticiones con el fin de probar que el algoritmo funciona ante diversas condiciones de operación dadas por las condiciones iniciales de los estados de estimación y los pesos sinápticos. En las 1000 repeticiones se midió el costo computacional mediante la medición del tiempo que un equipo de computo necesita bajo las mismas condiciones concluyendo que el sistema fraccionario es capaz de competir con los algoritmos convencionales.

7.2. Trabajos Futuros

Como se ha explicado con anterioridad, el algoritmo de optimización fraccionario fue propuesto utilizando el método de gradiente descendente pero, este no es el único método de optimización basado en gradiente. Por lo que define como trabajo futuro:

1. Proponer métodos de optimización basados en gradiente para resolver la ecuación diferencial que caracteriza el descenso del parámetro considerando el algoritmo de LM, Gauss-Newton, Newton-Rapson, etc.

2. El fundamento matemático de la RNA fraccionaria mostrada en el capítulo 5 consiste en calcular mediante la regla de la cadena el cambio de la función de activación con respecto al parámetro que se desea optimizar. En este caso, se da la oportunidad de cambiar la regla de la cadena (derivadas parciales) por derivadas fractales.

3. Extrapolar la metodología a no solo algoritmos basados en gradiente sino a algoritmos heurísticos donde se utilicen derivadas temporales o espaciales como el PSO, cuckoo search, enjambre de hormigas, vuelo de murciélagos, etc.

4. Utilizar el algoritmo de optimización propuesto para definir redes neuronales de orden fraccionario para realizar la clasificación de patrones con el uso de *Machine Learning* y *Deep Learning*.

5. Utilizar el algoritmo propuesto para diversas estructuras diseñadas especialmente para la predicción de características como lo son las LSTM.

Con respecto a las estructuras de RNA:

1. Tal y como los pesos sinápticos, las funciones de activación también son parte de una RNA por lo que se proponen utilizar funciones de activación expresadas en términos de funciones especiales utilizadas en el CF como la función gamma (2.4), función de ML (2.7), entre otras.

2. El campo de las derivadas fraccionarias discretas es un tema que está reluciendo en años recientes, un posible nicho de oportunidad se presenta en el uso de derivadas fraccionarias discretas Δ^{α} para el procesamiento de información.

APÉNDICE A

Métodos Numéricos Para Derivadas Fraccionarias

En esta sección se presentan diversos métodos numéricos para emular el comportamiento de las diferentes definiciones de derivadas fraccionarias que se han utilizado para el desarrollo de este trabajo de investigación.

A.1. Esquema Numérico para la Derivada Fraccionaria de Grünwald-Letnikov

Para presentar el esquema numérico de la derivada fraccionaria en el sentido de GL, se presenta nuevamente la definición de derivada fraccionaria como

Definición 12. *Sea* $x : \mathbb{R} \to \mathbb{R} \times [t_0, \frac{T-t_0}{h}]$, $\alpha \in (0, 1]$, $x \in C^1$ *entonces, la definición de GL está dada por la ecuación* (A.1)*:*

$$ {}^{GL}_{t_0}\mathcal{D}^{\alpha}_{T} x(t) = f(t, x(t)) = \lim_{h \to 0} \frac{1}{h^{\alpha}} \sum_{k=0}^{\frac{T-t_0}{h}} (-1)^k \binom{\alpha}{k} x(t - kh), \tag{A.1} $$

donde $(-1)\binom{\alpha}{k}$ *representan coeficientes binomiales* $c_j^{(\alpha)}$, $(j = \{0, 1, 2, \ldots\})$ *calculados por la ecuación* (A.2)

$$ \begin{aligned} c_0^{(\alpha)} &= 1, \\ c_j^{(\alpha)} &= \left(1 - \frac{1+\alpha}{j}\right) c_{j-1}^{(\alpha)}, \quad j > 0. \end{aligned} \tag{A.2} $$

Definición 13. *Por lo tanto el esquema numérico para resolver la ecuación* (A.1) *está dado por la ecuación* (A.3) *[191]*

$$ x(t_k) = f(t_k, x(t_k))h^{\alpha} - \sum_{j=0}^{k} c_j^{(\alpha)} x(t_k - j). \tag{A.3} $$

A.2. Esquema Numérico para la Derivada Fraccionaria de Liouville-Caputo

Considerando la siguiente EDF en el sentido LC:

$$ {}_0^{LC}\mathcal{D}_t^{\alpha} f(t) = g(t, f(t)),\ f^k(0) = f_0^k,\ k = 0, 1, \ldots, n, n-1, \tag{A.4} $$

considerando nuevamente la ecuación de la definición LC:

$$ {}_0^{LC}\mathcal{D}_t^{\alpha} f(t) = \frac{1}{1-\alpha}\int_0^t \frac{d}{d\tau} f(\tau)(t-\tau)^{1-\alpha} d\tau. \tag{A.5} $$

Con la condición inicial de (A.4), se genera una única solución en el intervalo $t \in [0, T]$, por lo que la ecuación (A.5) satisface la ecuación integro diferencial de Volterra [192]:

$$ f(t) = f_0 + \frac{1}{\Gamma(\alpha)}\int_0^t (t-\tau)^{\alpha-1} g(\tau, f(\tau)) d\tau,\ t < T. \tag{A.6} $$

La solución de la ecuación (A.6) es conocido como el método de Adams-Bashforth-Multon (ABM). El método está basado en una solución iteratuva como [192, 193]:

$$ \begin{aligned} f_{k+1}^P &= f_0 + \frac{1}{\Gamma(\alpha)}\sum_{j=0}^{k} b_{j,k+1} g(t_j, f_j) \\ f_{k+1} &= f_0 + \frac{1}{\Gamma(\alpha)}\left(\sum_{j=0}^{k} a_{j,k+1} g(t_j, f_j) + a_{j,k+1} g(t_{k+1}, f_{k+1}^P)\right), \end{aligned} \tag{A.7} $$

donde,

$$ \begin{aligned} a_{j,k+1} &= \frac{h^{\alpha}}{\alpha(\alpha+1)} \begin{cases} k^{\alpha+1} - (k-\alpha)(k+1)^{\alpha}, & j = 0, \\ (k-j+2)^{\alpha+1} + (k-j)^{\alpha+1} - 2(k-j+1)^{\alpha+1}, & 1 \le j \le k, \\ 1, & j = k+1 \end{cases} \\ b_{j,k+1} &= \frac{h\alpha}{\alpha}((k+1-j)^{\alpha} - (k-j)^{\alpha}), \quad j = 0, 1, 2, 3, \ldots, k. \end{aligned} \tag{A.8} $$

A.3. Esquema Numérico para la Derivada Fraccionaria de Atangana-Baleanu-Caputo

El método ABM para la derivada fraccionaria de ABC es descrito considerando [109, 110]

$$ {}_0^{ABC}\mathcal{D}_t^{\alpha} f(t) = g(t, f(t)), \quad f^k(0) = f_0^k, \quad k = 0, 1, ..., n-1, \tag{A.9} $$

donde $\alpha(t) > 0$ y ${}_0^{ABC}\mathcal{D}_t^{\alpha(t)}$ es la derivada fraccionaria en el sentido de Atangana-Baleanu-Caputo (ABC). La ecuación(A.9) tiene una solución única en $t \in [0, T]$, esta solución puede ser reescrita utilizando la integral asociada a la definición de Atangana-Baleanu como sigue:

$$ f(t) = f_0 + \frac{1-\alpha(t)}{B(\alpha(t))} g(t, f(t)) + \frac{\alpha(t)}{B(\alpha(t))\Gamma(\alpha(t))}\int_0^t g(u, f(u))(t-u)^{\alpha(t)-1} du, \tag{A.10} $$

donde $B(\alpha(t))$ es una función de normalización de tal manera que $B(0) = B(1) = 1$. Ahora el esquema numérico utilizando cuadraturas trapezoidales es definido como

$$\begin{aligned} f_{i+1}^{P} &= f_0 + \frac{1-\alpha}{B(\alpha)} g(t_{i+1}, f_{i+1}) + \frac{\alpha}{\Gamma(\alpha)B(\alpha)} \sum_{j=0}^{i} b_{j,i+1} g(t_j, f_j), \\ f_{i+1} &= f_0 + \frac{1-\alpha}{B(\alpha)} g(t_{i+1}, f_{i+1}^{P}) + \frac{\alpha}{B(\alpha)} \left[\frac{h^\alpha}{\Gamma(\alpha+2)} g(t_{i+1}, f_{i+1}^{P}) + \right. \\ &\left. + \frac{h^\alpha}{\Gamma(\alpha+2)} \sum_{j=0}^{i} a_{j,i+1} g(t_j, f_j) \right], \end{aligned} \tag{A.11}$$

donde,

$$\begin{aligned} a_{j,i+1} &= \begin{cases} i^{\alpha+1} - (i-\alpha)(i+1)^\alpha, & j = 0, \\ (i-j+2)^{\alpha+1} + (i-j)^{\alpha+1} - 2(i-j+1)^{\alpha+1}, & 1 \le j \le i, \end{cases} \\ b_{j,i+1} &= \frac{h^\alpha}{\alpha}((i+1-j)^\alpha - (i-j)^\alpha), \qquad j = 0, 1, 2, ..., i. \end{aligned} \tag{A.12}$$

A.4. Esquema Numérico Fraccionario-Fractal-Riemann-Liouville

Considerando el problema general de Cauchy

$$\begin{aligned} {}_0^{FFP}\mathcal{D}_t^{\alpha,\beta} x(t) &= f(t, x(t)), \\ x(0) &= x_0, \end{aligned} \tag{A.13}$$

donde $f(t, x(t)) \in C^1$, $x(t) \in C^1$. Se utiliza la integral asociada a la definición FFP mostrada en (2.25)

$$x(t) = \frac{1}{\Gamma(\alpha)} \int_0^t (t-\tau)^{\alpha-1} f(\tau, x(\tau)) \beta \tau^{\beta-1} d\tau, \tag{A.14}$$

en el punto $t = t_{m+1}$ la ecuación (A.14) puede ser reformulada como

$$x(t) = \frac{1}{\Gamma(\alpha)} \int_0^{t_{m+1}} (t_{m+1}-\tau)^{\alpha-1} f(\tau, x(\tau)) \beta \tau^{\beta-1} d\tau, \tag{A.15}$$

$$\Rightarrow x_{m+1} = \frac{1}{\Gamma(\alpha)} \sum_{j=0}^{m} \int_{t_j}^{t_{j+1}} (t_{m+1}-\tau)^{\alpha-1} f(\tau, x(\tau)) \beta \tau^{\beta-1} d\tau, \tag{A.16}$$

por simplicidad

$$\begin{aligned} F(t, x(t)) &= f(t, x(t)) \beta \tau^{\beta-1}, \\ \therefore x_{m+1} &= \frac{1}{\Gamma(\alpha)} \sum_{j=0}^{m} \int_{t_j}^{t_{j+1}} (t_{m+1}-\tau)^{\alpha-1} F(\tau, x(\tau)) d\tau, \end{aligned} \tag{A.17}$$

en el intervalo $[t_j, t_{j+1}]$ se pueden aplicar aproximaciones polinomiales con la interpolación de Lagrange para aproximar a la función $F(t, x(t))$

$$F(t, x(t)) \cong \frac{t - t_{j-1}}{t_j - y_{j-1}} F(t_j, x(t_j)) - \frac{t - t_j}{t_j - t_{j-1}} F(t_{j-1}, x(t_{j-1})). \tag{A.18}$$

Reemplazando la ecuación (A.18) en la ecuación (A.17) se deduce la siguiente aproximación

$$x_{m+1} = \frac{1}{\Gamma(\alpha)} \sum_{j=0}^{m} \int_{t_j}^{t_{j+1}} (t_{m+1}-\tau)^{\alpha-1} \left[\frac{\tau - t_{j-1}}{t_j - t_{j-1}} F(t_j, x(t_j)) - \frac{\tau - t_j}{t_j - t_{j-1}} F(t_{j-1}, x(t_{j-1})) \right] d\tau, \tag{A.19}$$

ahora, utilizando cuadraturas trapezoidales, es posible aproximar la integral mediante

$$\begin{aligned} x_{m+1} = \sum_{j=0}^{m} \Bigg[& \frac{(\Delta t)^\alpha}{\Gamma(\alpha+2)} F(t_j, x_j) \left\{ (m+1-j)^\alpha (m-j+2+\alpha) - (m-j)^\alpha (m-j+2+2\alpha) \right\} \\ & \frac{(\Delta t)^\alpha}{\Gamma(\alpha+2)} F(t_{j-1}, x_{j-1}) \left\{ (m+1-j)^{\alpha+1} - (m-j)^\alpha (m-j+1+\alpha) \right\} \Bigg]. \end{aligned} \tag{A.20}$$

Sustituyendo la ecuación (A.17) en la ecuación (A.20) se obtiene el esquema numérico para resolver EDF en sentido FFP.

$$\begin{aligned} x_{m+1} = \sum_{j=0}^{m} \Bigg[& \frac{(\Delta t)^\alpha}{\Gamma(\alpha+2)} f(t_j, x(t_j)) \beta \tau^{\beta-1} \left\{ (m+1-j)^\alpha (m-j+2+\alpha) - (m-j)^\alpha (m-j+2+2\alpha) \right\} \\ & \frac{(\Delta t)^\alpha}{\Gamma(\alpha+2)} f(t_{j-1}, x(t_{j-1})) \beta \tau^{\beta-1} \left\{ (m+1-j)^{\alpha+1} - (m-j)^\alpha (m-j+1+\alpha) \right\} \Bigg]. \end{aligned} \tag{A.21}$$

La comprobación de existencia y unicidad del esquema numérico es desarrollado en [65].

APÉNDICE B

Algoritmos de Optimización basados en Gradiente

Los algoritmos de optimización, aprendizaje, entrenamiento, etc. son usados para adaptar los pesos sinápticos de una arquitectura de red neuronal definida. El objetivo de esta adaptación es minimizar la función costo, por ejemplo (3.11). Los algoritmos de aprendizaje comúnmente utilizados para la adaptación de los pesos sinápticos son los de propagación hacia atrás o retropropagación (RP o BP), que se basan principalmente en la regla de la cadena para obtener las derivadas parciales correspondientes al parámetro que se desea estimar. La forma general de los algoritmos para adaptar los pesos sinápticos está dada por la ecuación (B.1).

$$\hat{w}(k+1) = \hat{w}(k) - \eta[R^{-1}]\frac{\partial E}{\partial \hat{w}}, \tag{B.1}$$

donde R modifica el tipo del algoritmo, η es el factor de aprendizaje, $\frac{\partial \mathcal{E}}{\partial \hat{w}}$ es el gradiente de la función objetivo con respecto al parámetro que se desea estimar ($\hat{w}(k)$) en la k-ésima iteración.

B.1. Método de Gradiente Descendente

Como se mencionó con anterioridad, la ecuación (B.1) generaliza a gran parte de los algoritmos de optimización. En este caso si $R = 1$ se obtiene el algoritmo del gradiente descendente como se expresa en la ecuación (B.2).

$$\hat{w}(k+1) = \hat{w}(k) - \eta\frac{\partial E}{\partial \hat{w}}, \tag{B.2}$$

donde $\hat{w}(k+1)$ es el valor del parámetro en la siguiente época, ciclo o iteración, $\hat{w}(k)$ es el valor del parámetro actual, $\frac{\partial \mathcal{E}}{\partial \hat{w}}$ es la razón de cambio de la función costo con respecto al parámetro que se desea estimar y η es el factor de aprendizaje.

En la ecuación (B.2), el parámetro η juega un papel importante en la adaptación de los pesos sinápticos pero es de mayor importancia su papel en el algoritmo de optimización debido a que define la velocidad de convergencia del algoritmo. En los algoritmos de optimización el valor de η debe encontrarse delimitado en un rango pequeño. Si se define un valor de η relativamente pequeño, significa que la velocidad de convergencia será lenta pero, si se define el valor de η relativamente grande, entonces se tendrá una convergencia

rápida pero podrían presentarse complicaciones como oscilaciones en el comportamiento de la función costo con respecto al número de épocas o inestabilidad. Para evadir la problemática que produce la selección de un valor de η constante, se implementó un algoritmo de "búsqueda y convergencia" sugerido en [118] y mostrado en la sección 3.5.2.

B.2. Método de Newton

La idea tras el método de Newton es tal que una función objetivo $\mathcal{E}(\hat{w})$ debe ser minimizada con una aproximación local mediante una función cuadrática. La función $\mathcal{E}(\hat{w})$ cerca al punto $\hat{w}(k)$ ($k = 1, 2, ..., N$) puede ser aproximada mediante la serie truncada de Taylor:

$$\mathcal{E}(\hat{w}) \cong \mathcal{E}(\hat{w}(k)) + \Delta\hat{w}^T \nabla\mathcal{E}(\hat{w}(k)) + \frac{1}{2}\Delta\hat{w}^T \nabla^2\mathcal{E}(\hat{w}(k))\Delta\hat{w}. \tag{B.3}$$

Realizando ciertas operaciones se tiene que:

$$\hat{w}(k+1) = \hat{w}(k) - \left[\nabla^2\mathcal{E}(\hat{w})\right]^{-1} \nabla\mathcal{E}(\hat{w}), \tag{B.4}$$

donde $\left[\nabla^2\mathcal{E}(\hat{w})\right]$ se puede definir como la matriz Hessiana de la función objetivo con respecto al parámetro que se desea optimizar y $\nabla\mathcal{E}(\hat{w})$ es la matriz Jacobiana de la función objetivo.

Utilizando la generalización de la ecuación (B.1), considerando que $R = \nabla^2\mathcal{E}(\hat{w})$ y que el valor de $\eta = 1$ se puede llegar a la misma expresión (B.4).

En general, el método de Newton tiene un resultado de convergencia rápido, pero requiere la evaluación de la primera y segunda derivada de la función objetivo y del calculo del inverso de la matriz Hessiana lo cual contrae problemas de singularidades. Además, si el punto inicial ($\hat{w}_0$) se encuentra distante del mínimo global, la matriz Hessiana podría no ser positiva definida y el algoritmo podría diverger.

B.3. Método de Levenberg-Marquardt

El algoritmo de Levenberg-Marquardt es una modificación al algoritmo de Newton, el cual evita ciertas desventajas que tiene el algoritmo de Newton. La técnica de Levenberg-Marquardt [194] aproxima el comportamiento de la matriz Hessiana por la matriz $\nabla^2\mathcal{E} = (J(\hat{w})^T J(\hat{w}) + \mu I)$, donde μ es un factor positivo con efectos (para optimización) parecidos al comportamiento de η en el algoritmo del gradiente descendente e I es la matriz identidad. Partiendo del método de Newton mostrado en la ecuación (B.4), se tiene que $\nabla\mathcal{E}(\hat{w}) \approx 2J(\hat{w})^T e(\hat{w})$ donde $J(\hat{w})^T$ es la matriz Jacobiana transpuesta de la función objetivo, $e(\hat{w})$ son los residuos de la estimación y $\nabla^2\mathcal{E}(\hat{w}) \approx 2J(\hat{w})^T J(\hat{w})$. Realizando ciertas operaciones se llega a la expresión mostrada en la ecuación (B.5) la cual es el algoritmo de Levenberg-Marquardt.

$$\hat{w}(k+1) = \hat{w}(k) - \left[J(\hat{w})^T J(\hat{w}) + \mu I\right]^{-1} J(\hat{w})^T e(\hat{w}). \tag{B.5}$$

Nótese que $[J(\hat{w})^T J(\hat{w}) + \mu I]$ es una matriz simétrica no singular, aún si la matriz Hessiana (aproximada por $J(\hat{w})^T J(\hat{w})$) es singular. Además, esta matriz será positiva definida con la selección apropiada de μ. Por otro lado, el parámetro μ debería ser pequeño para asegurar una convergencia rápida. El algoritmo de Levenberg-Marquardt posee una similitud al algoritmo de Newton. Además, el algoritmo puede converger a un valor óptimo a pesar que el valor inicial $\hat{w}_0$ sea relativamente pobre (es decir, que el valor inicial se encuentre distante del valor óptimo $\hat{w}^*$) al igual que el método del gradiente descendente.

En conclusión, si el valor de $\mu \to \infty$ el algoritmo de Levenberg-Marquardt tenderá a comportarse como el método del gradiente descendente pero, si $\mu \to 0$ el algoritmo tendrá las capacidades de convergencia iguales al algoritmo de Newton. Para asegurar la mejor selección del valor de μ se utilizó un algoritmo de "búsqueda y convergencia" [118, 195] al igual que en el caso del gradiente descendente considerando un perfil de curva diferente. El algoritmo de búsqueda y convergencia para μ está dado por la ecuación (B.6).

$$\mu = \mu_0 \left(1 + \frac{k}{5 * k_0}\right), \tag{B.6}$$

donde μ_0 es el valor inicial que puede ser calculado por la traza de la matriz Hessiana [195], k_0 es igual $\frac{N}{3}$, N es el número total de datos que se están utilizando para realizar la estimación del sistema y k es el k-ésimo instante de tiempo discreto.

APÉNDICE C

Optimización por Enjambre de Partículas

El algoritmo por enjambre partículas (PSO del inglés *Particle Swarm Optimization*) es una algoritmo de optimización encargado de minimizar una función objetivo como la ecuación (3.11) [76, 77]. El PSO trabajo mediante una población (llamada enjambre) compuesto por diversas soluciones (llamadas partículas). Las trayectorias de las partículas son gobernadas mediante la ecuación (C.1) dentro de un espacio de solución

$$\begin{aligned} \nu_{i+1} &= \omega\nu_i + c_1(p_{ai} - x_i) + c_2(p_{bi} - x_i), \\ x_i &= x_i + \nu_i, \end{aligned} \tag{C.1}$$

donde c_1 y c_2 son dos constantes positivas que representan aceleraciones, ω es un factor de momentum la cual provee un balance entre exploración local y global, la $i-$ésimas partícula es representada como $X = (x_1, x_2, \ldots, x_S)$, $P = (p_1, p_2, \ldots, p_S)$ es la posición de la partícula evaluada mediante la función costo y $V = (\nu_1, \nu_2, \ldots, \nu_S)$ es el cambio de posición (velocidad). El parámetro **g** es la posición mejor evaluada por medio de la función costo. El movimiento de cada partícula está guiado por la ecuación (C.1) a través del espacio de solución pero, cuando una partícula obtiene un mejor resultado, es decir, un mejor **g**, las demás partículas comenzarán a converger a esa posición. El objetivo es encontrar una solución Ω_1 para el cual $f(\Omega_1) \leq f(\Omega_2), \forall\Omega_2$ en el espacio de búsqueda, esto significa que Ω_1 es la mejor posición que optimiza el problema. Un ejemplo del algoritmo es presentado a continuación:

Algorithm 1 Optimización por Enjambre de Partículas

procedure PSO
  for i = 1 **to** el tamaño del enjambre **do**
    Inicializar X aleatoriamente dentro de los límites $[X_{min}, X_{max}]$;
    Inicializar V aleatoriamente dentro de los límites $[V_{min}, V_{max}]$;
  Evaluar cada partícula;
  Identificar la mejor posición P y definir **g**;
  for i = 1 **to** número de épocas **do**
    for j = 1 **to** tamaño del enjambre **do**
      $\nu_{i+1} = \omega\nu_i + c_1(p_{ai} - x_i) + c_2(p_{bi} - x_i)$;
      $x_i = x_i + \nu_i$;
      $P_{i+1} = P_i$;
      Evaluar $f(x_i)$;
      if $f(P_{i+1}) < f(x_i)$ **then**
        Actualizar P_{i+1};
      if $f(\mathbf{g}) < f(P_{i+1})$ **then**
        Actualizar g;

APÉNDICE D

Algoritmo de Fuerza Bruta

El algoritmo de fuerza bruta es un método iterativo para encontrar los parámetros óptimos para problemas de bajo grado de complejidad basado en experimentaciones empíricas. En este trabajo de investigación, se ha utilizado este algoritmo para la búsqueda del número de salidas pasadas n_a, entradas pasadas n_b y para encontrar el tiempo muerto de los sistemas n_k. Esto último es un paso importante para el método de identificación ya que en ocasiones es necesario obtener información del comportamiento pasado del sistema.

Como se ha mencionado anteriormente, se realizan diversas pruebas y se evalúa el error de cada una de las pruebas. El error es evaluado bajo estándares que el propio usuario define, es decir, se define una función objetivo dependiendo de un criterio ya sea basado en el número de parámetros, la norma del error, el FIT, el valor RMSE, la pendiente-intercepto, etc.

Algorithm 2 Fuerza Bruta

```
1:  procedure FB
2:      Inicializar n_a dentro de los límites n̄_k y n_k;
3:      Inicializar n_a dentro de los límites n̄_b y n_b;
4:      Inicializar n_a dentro de los límites n̄_a y n_a;
5:      for i_nk = n_k to n̄_k do
6:          for i_nb = n_b to n̄_b do
7:              for i_na = n_a to n̄_a do
8:                  Estimar los parámetros del modelo
9:                  Calcular el desempeño del modelo optimizado
10:                 Guardar el desempeño calculado
11:                 Guardar los valores de n_a, n_b y n_k
12:     Graficar el desempeño
13:     Seleccionar el modelo de acuerdo a los criterios del usuario
```

El método consiste primeramente en definir un subespacio de soluciones compuesto por los intervalos mínimo y máximo de valores de n_a, n_b y n_k. Dentro de bucles compuestos por los valores de regresión, se evalúa el desempeño del modelo propuesto. Una vez que todos los bucles han concluido, dependerá del usuario seleccionar el modelo que cumpla con todos los estándares que él mismo haya establecido. Un ejemplo del algoritmo de fuerza bruta se muestra en 2.

APÉNDICE E

Coordenadas D/Q

La transformación de Park o D/Q convierte las componentes 'abc' de un sistema trifásico a otro sistema de referencia 'dq0'. El objetivo de la transformación consiste en convertir los valores trifásicos 'abc', variables senoidalmente en el tiempo, a valores constantes 'dq0', en régimen permanente. El vector con las componentes del nuevo sistema de referencia $[x_r]$ se obtiene multiplicando el vector de coordenadas trifásicas $[x]$ por una matriz de transformación $[T]$, según la expresión (E.1)

$$\begin{bmatrix} x_d \\ x_q \\ x_0 \end{bmatrix} = [x_r] = [T] \cdot [x] = [T] \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix}, \tag{E.1}$$

donde la matriz de transformación $[T]$ está dada por la ecuación (E.2).

$$T = \sqrt{\frac{2}{3}} \begin{bmatrix} \cos(\theta) & \cos(\theta - \frac{2\pi}{3}) & \cos(\theta + \frac{2\pi}{3}) \\ -\sin(\theta) & -\sin(\theta - \frac{2\pi}{3}) & -\sin(\theta + \frac{2\pi}{3}) \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}, \end{bmatrix} \tag{E.2}$$

donde θ (expresada en la ecuación (E.3)) es el ángulo de referencia rotativa (ejes D-Q) como lo muestra la Figura E.1

$$\theta = \int_0^t \omega t dt + \theta_0, \tag{E.3}$$

donde ω es la velocidad angular de referencia D/Q y θ_0 es el ángulo inicial de la referencia D/Q. Cuando la velocidad angular ω es constante, la trnasformación se puede expresar según la expresión (E.4).

$$T = \sqrt{\frac{2}{3}} \begin{bmatrix} \cos(\omega t + \theta_0) & \cos(\omega t + \theta_0 - \frac{2\pi}{3}) & \cos(\omega t + \theta_0 + \frac{2\pi}{3}) \\ -\sin(\omega t + \theta_0) & -\sin(\omega t + \theta_0 - \frac{2\pi}{3}) & -\sin(\omega t + \theta_0 + \frac{2\pi}{3}) \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \tag{E.4}$$

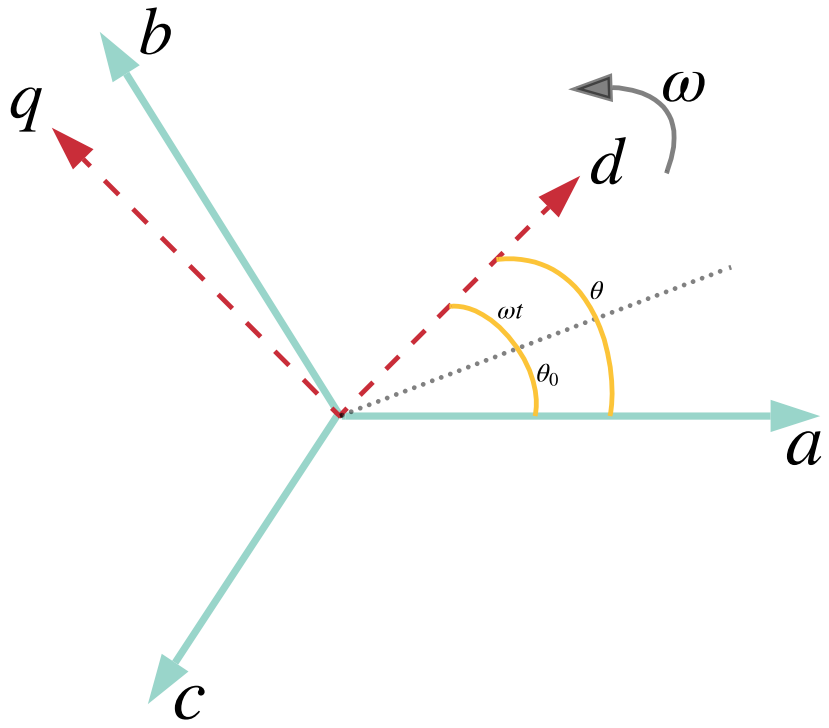


Figura E.1: Sistemas de referencia trifásico y D/Q.

APÉNDICE F

Productos Obtenidos

Durante el desarrollo doctoral se han realizado diversas publicaciones referentes al CF, relaciones entre el CF y RNAs, control, etc. Este apéndice muestran las publicaciones realizadas al igual que la participación en conferencias internacionales.

Chaos, Solitons and Fractals 103 (2017) 382–403

Contents lists available at ScienceDirect

Chaos, Solitons and Fractals

Nonlinear Science, and Nonequilibrium and Complex Phenomena

journal homepage: www.elsevier.com/locate/chaos

Solving fractional differential equations of variable-order involving operators with Mittag-Leffler kernel using artificial neural networks

C.J. Zúñiga-Aguilar[a], H.M. Romero-Ugalde[b], J.F. Gómez-Aguilar[c,*], R.F. Escobar-Jiménez[a], M. Valtierra-Rodríguez[d]

[a] Centro Nacional de Investigación y Desarrollo Tecnológico, Tecnológico Nacional de México, Interior Internado Palmira S/N, Col. Palmira, C.P. 62490, Cuernavaca, Morelos, México

[b] University Grenoble Alpes, F-38000 Grenoble, France, CEA LETI MINATEC Campus, F-38054 Grenoble France

[c] CONACyT - Centro Nacional de Investigación y Desarrollo Tecnológico, Tecnológico Nacional de México, Interior Internado Palmira S/N, Col. Palmira, C.P. 62490, Cuernavaca, Morelos, México

[d] Facultad de Ingeniería, Universidad Autónoma de Querétaro, Campus San Juan del Río, Río Moctezuma 249, Col. San Cayetano, C.P. 76807, México

ARTICLE INFO

Article history:
Received 14 April 2017
Revised 28 June 2017
Accepted 28 June 2017

Keywords:
Fractional calculus
Nonlinear fractional differential equations
Atangana-Baleanu-Caputo fractional derivative
Variable-order fractional derivative
Artificial neural networks

ABSTRACT

In this paper, we approximate the solution of fractional differential equations using a new approach of artificial neural network. We consider fractional differential equations of variable-order with Mittag-Leffler kernel in Liouville–Caputo sense. With this new neural network approach, it is obtained an approximate solution of the fractional differential equation and this solution is optimized using the Levenberg–Marquardt algorithm. The neural network effectiveness and applicability were validated by solving different types of fractional differential equations, the Willamowski-Rössler oscillator and a multi-scroll system. The solution of the neural network was compared with the analytical solutions and the numerical simulations obtained through the Adams-Bashforth-Moulton method. To show the effectiveness of the proposed neural network different performance indices were calculated.

1. Introduction

Fractional calculus (FC) is a generalization of integration and differentiation to non-integer order. Recent studies in science and engineering demonstrated that the dynamics of many systems can be described more accurately by means of differential equations of non-integer order, for instance bioengineering, viscoelasticity, diffusion, chaos theory, physics, electromagnetism, and many others [1–8]. There are several approximation techniques to solve nonlinear fractional partial differential equations. In general, we can use direct and indirect implementation techniques for numerical approximation of the fractional operator, several numerical and analytical methods have been developed, for example, fractional subequation method [9–11], the homotopy perturbation method [12–14], the variational iteration method [15–18], homotopy perturbation transform method [19–21], Adomian decomposition method [22–24], Wavelet Method [25,26], Laplace transforms [27,28]. Several definitions of fractional order derivatives have been proposed, including: Grünwald–Letnikov, Riemann–Liouville, Weyl, Riesz and the Liouville–Caputo representation. In 2016 Atangana and Baleanu presented another version of fractional derivatives, which uses the generalized Mittag-Leffler function as the non-singular and non-local kernel. This definition has all the benefits of the fractional operators of Riemann–Liouville or Liouville–Caputo [29–32]. Several studies have shown that, many complex physical problems can be described with great success via variable-order derivatives (VO), [33–36]. A novel study underlining the advantages of using these derivatives rather than constant order fractional derivative was presented in [37]. Some applications include processing of geographical data, diffusion processes and groundwater flow equation [38–40]. The equations described by the VO derivatives are highly complex, difficult to handle analytically, it is therefore advisable to research their solutions numerically [41–43]. In [43], the authors proposed a modification to the ABM method to solve differential equations of fractional order in Liouville–Caputo sense. The integration step is function of the delay and the number of samples. The authors showed that the numerical errors decay quickly by reducing the integration step size.

Artificial neural networks (ANNs) have also been successfully applied on many scientific and engineering fields thanks to their robustness [44] and their capacity to approximate nonlinear behaviors or functions. Cybenko established in [45] that a nonlinear

* Corresponding author.
E-mail addresses: jgomez@cenidet.edu.mx, jfranciscogoma@hotmail.com (J.F. Gómez-Aguilar).

http://dx.doi.org/10.1016/j.chaos.2017.06.030
0960-0779/

EPJ Plus

EPJ.org
your physics journal

Eur. Phys. J. Plus (2018) **133**: 75 DOI 10.1140/epjp/i2018-11917-0

New numerical approximation for solving fractional delay differential equations of variable order using artificial neural networks

C.J. Zúñiga-Aguilar, A. Coronel-Escamilla, J.F. Gómez-Aguilar, V.M. Alvarado-Martínez, H.M. Romero-Ugalde

Società Italiana di Fisica

Springer

EPJ Plus

EPJ.org
your physics journal

Eur. Phys. J. Plus (2018) **133**: 13 DOI 10.1140/epjp/i2018-11853-y

Robust control for fractional variable-order chaotic systems with non-singular kernel

C.J. Zu niga-Aguilar, J.F. Gómez-Aguilar, R.F. Escobar-Jiménez and H.M. Romero-Ugalde

Società Italiana di Fisica

Springer

Chaos, Solitons and Fractals 115 (2018) 177–189

Contents lists available at ScienceDirect

Chaos, Solitons and Fractals

Nonlinear Science, and Nonequilibrium and Complex Phenomena

journal homepage: www.elsevier.com/locate/chaos

FPGA implementation and control of chaotic systems involving the variable-order fractional operator with Mittag–Leffler law

L.F. Ávalos-Ruiz[a], C.J. Zúñiga-Aguilar[a], J.F. Gómez-Aguilar[b,*], R.F. Escobar-Jiménez[a], H.M. Romero-Ugalde[c]

[a] Tecnológico Nacional de México/CENIDET, Interior Internado Palmira S/N, Col. Palmira, C.P. 62490, Cuernavaca Morelos, México
[b] CONACyT-Tecnológico Nacional de México/CENIDET, Interior Internado Palmira S/N, Col. Palmira, C.P. 62490, Cuernavaca Morelos, México
[c] Diabeloop SA, 155 Cours Berriat, F-38000 Grenoble, France

ARTICLE INFO

Article history:
Received 13 June 2018
Revised 17 August 2018
Accepted 21 August 2018

Keywords:
Fractional calculus
Variable-order fractional operators
Nonlinear systems
Chaos
LabVIEW software
FPGA implementation

ABSTRACT

This paper presents the simulation and control implementation on a Field Programmable Gate Array (FPGA) for a class of variable-order fractional chaotic systems by using sliding mode control strategy. Four different fractional variable-order chaotic systems via Atangana–Baleanu–Caputo fractional-order derivative were considered; Dadras, Aizawa, Thomas and 4 Wings attractors. A methodology has been developed to construct variable-order fractional chaotic systems using LabVIEW® software for its implementation in the National Instruments myRio-1900 (Xilinx FPGA Z-7010)® device. The variable-order fractional differential equations and the control law were solved using the variable-order Adams algorithm. Finally, simulation results show that FPGA provides high-speed realizations with the desired accuracy and demonstrate the effectiveness of the proposed sliding mode control.

1. Introduction

Fractional calculus is the mathematical generalisation of classical calculus, this mathematical tool has been used in the recent decades for modeling real world problems in many field's science, technology and engineering [1–8]. Fractional-Order Differential Equations (FODE's) have increasingly attracted attention for the evaluation of dynamical systems. The fractional derivative operator is non-local, which expresses that the system's response will be affected at any time by all previous responses. FODE's give an exact description of different physical phenomena, also, FODE's give a description of the inherent relation of different processes with memory and hereditary properties [9]. In the literature there are several definitions of fractional-order derivatives, for instance, Riemann–Liouville, Grünwald–Letnikov, Liouville–Caputo, Caputo–Fabrizio and Atangana–Baleanu [10–25]. The order of the fractional derivative can be interpreted as the index of memory of the system. This fractional-order can be real, rational or irrational, or even complex. Samko, in [26], stated that the fractional integrals and derivative can be generalized introducing in the fractional order a function of time or space or space-time variables $q(x, t)$. Several studies have been reported in the literature employing this proposal with excellent results [27–32].

Chaos, as a very interesting nonlinear phenomenon, has been intensively studied in the last decades. In the literature, there are several works on different control techniques applied to a variety of chaotic systems [33–36]. The chaos control problem in a fractional order brushless DC motor was studied by the authors in [37], in this work, the sliding mode control, robust control, and extended back-stepping techniques were represented in the Liouville–Caputo sense. The chaos control for a general class of chaotic systems based on the sliding mode control theory was studied by Wang et al. [38]. The authors used feedback controllers to guarantee asymptotic stability of the chaotic systems. Yin et al. [39] presented a sliding mode control law for controlling a class of fractional-order chaotic systems. Authors in [40] developed a modified sliding mode approach for synchronizing fractional-order chaotic systems using neural networks. In recent years, the hardware implementation of fractional chaotic systems has increasingly attracted attention. Nevertheless, the implementation of the algorithms is complicated due to their memory dependence and the hardware requires the use of high-order integer order systems. Several digital implementations of chaotic systems have been implemented on FPGAs.

FPGAs are well-known for their processing speed and hardware flexibility. The maind advantage of this technology is its low con-

* Corresponding author.
E-mail address: jgomez@cenidet.edu.mx (J.F. Gómez-Aguilar).

https://doi.org/10.1016/j.chaos.2018.08.021
0960-0779/

Chaos, Solitons and Fractals 130 (2020) 109444

Contents lists available at ScienceDirect

Chaos, Solitons and Fractals

Nonlinear Science, and Nonequilibrium and Complex Phenomena

journal homepage: www.elsevier.com/locate/chaos

Frontiers

Fractional order neural networks for system identification

C.J. Zuñiga Aguilar[a], J.F. Gómez-Aguilar[b], V.M. Alvarado-Martínez[a], H.M. Romero-Ugalde[c,*]

[a] Tecnológico Nacional de México/CENIDET, Interior Internado Palmira S/N, Col. Palmira. C.P. 62490, Cuernavaca, Morelos, México
[b] CONACyT - Tecnológico Nacional de México/CENIDET, Interior Internado Palmira S/N, Col. Palmira. C.P. 62490, Cuernavaca, Morelos, México
[c] ADynSys SAS, 428 Calle Segunda, Col. La Herradura, C.P. 35027, Gomez Palacio Durango, México

ARTICLE INFO

Article history:
Received 10 June 2019
Revised 1 September 2019
Accepted 12 September 2019

Keywords:
Fractional calculus
Neural networks
Black box modeling
System identification

ABSTRACT

Neural networks and fractional order calculus have shown to be powerful tools for system identification. In this paper we combine both approaches to propose a fractional order neural network (FONN) for system identification. The learning algorithm was generalized considering the Grünwald-Letnikov fractional derivative. This new black box modeling approach is validated by the identification of three different systems (two benchmark systems and a real system). Comparisons vs others approaches showed that the proposed FONN model reached better accuracy with less number of parameters.

1. Introduction

System identification is currently used in many engineering fields [1–3] since the models performed by this approach are usually easy to obtain and accurate enough for accomplishing a given task.

Neural networks (NN) have been largely used for system identification purposes and have shown to be a powerful tool [4–7]. Among the features of neural networks that allow system identification, we found the ability to approximate nonlinear functions due to the use of nonlinear activation functions, the ability to process many inputs and outputs, and the automatic adaptation of synaptic weights via a learning algorithm [8].

Although neural network-based system identification approaches are efficient in terms of accuracy [5,9], a large number of parameters is usually required to achieve this [10]. In this sense, many works have been performed in order to reduce the complexity of neural networks-based system identification models. For instance, in [11–13], trial and error approach was used, in order to manually increase the number of parameters of the proposed model until a given accuracy is reached. Good results were reached in previous works, however, this technique may not lead to the "best compromise" between the model simplicity and the approximation accuracy [11,14]. Similar techniques, for instance "pruning" [15–19], particle swarm optimization [20], singular value architectural recombination [21], and genetic algorithms [10,14,22,23], have been employed to improve the compromise, simplicity and accuracy, by automatically optimizing the structure of the neural network. However, in these techniques, a training is required each time the neural network evolves [24], increasing the computational cost. Different methods for leading with the compromise between complexity and accuracy are based on the design of the neural network [5–7,25–28]. Advantage of these approaches is that a single training is required, since the neural network does not evolve. The neural network is designed to ensure a good balance between accuracy and complexity after the training, which is performed only once.

In this paper, we follow the neural network design method proposed in [5–7], to derive balanced simplicity-accuracy system identification models. Furthermore, we take advantage of fractional order calculus to reduce even more the number of parameters of the proposed neural network-based models. In fact, many FC-based models have been used to approximate the dynamic of different systems [29–32]. For instance, in [29] and [30] a chickenpox disease fractional model is proposed using three different kernels (power law, exponential decay and Mittag-Leffler type) to determine the system with highest performance on real data in two different databases respectively. In [31], an epidemiological model (dengue fever) is proposed analyzing the behavior, the existence and uniqueness of three different fractional derivative definition, the results were compared against the classical model where the fractional model has a better performance. In [32], a physical model named as blood ethanol concentration model has been investigated proposing different types of kernels, the fractional models has been compared against the classical model. The parameters of the model and the fractional order have been optimized by

* Corresponding author.
E-mail address: hector.m.romero.ugalde@gmail.com (H.M. Romero-Ugalde).

https://doi.org/10.1016/j.chaos.2019.109444
0960-0779/

Chaos, Solitons and Fractals 126 (2019) 266–282

Contents lists available at ScienceDirect

Chaos, Solitons and Fractals

Nonlinear Science, and Nonequilibrium and Complex Phenomena

journal homepage: www.elsevier.com/locate/chaos

A novel method to solve variable-order fractional delay differential equations based in lagrange interpolations

C.J. Zúñiga-Aguilar[a], J.F. Gómez-Aguilar[b,*], R.F. Escobar-Jiménez[a], H.M. Romero-Ugalde[c]

[a] Tecnológico Nacional de México/CENIDET. Interior Internado Palmira S/N, Col. Palmira, Cuernavaca, Morelos, C.P. 62490, México
[b] CONACyT-Tecnológico Nacional de México/CENIDET. Interior Internado Palmira S/N, Col. Palmira, Cuernavaca, Morelos, C.P. 62490, México
[c] Diabeloop SA, 155 Cours Berriat, Grenoble, F-38000, France

ARTICLE INFO

Article history:
Received 29 April 2019
Revised 8 June 2019
Accepted 12 June 2019

Keywords:
Fractional calculus
Mittag-Leffler kernel
Lagrange interpolation
Fractional delay differential equations
Variable-order fractional operators

ABSTRACT

In this work, we present a novel numerical method based on the fundamental theorem of fractional calculus and the Lagrange polynomial interpolation to solve numerically fractional delay differential equations. We focus on the fractional derivative with power-law, exponential decay and Mittag-Leffler kernel of Liouville-Caputo type with constant and variable-order. The numerical methods were applied to simulate the Duffing attractor, El-Niño/Southern-Oscillation, and Ikeda systems.

1. Introduction

Fractional calculus (FC) is the generalization of ordinary calculus. Over the past decades, fractional differentiation has singled out as outstanding mathematical tools to portray more accurately many real-world problems. In literature, there are many definitions of fractional derivatives, for instance Riemann-Liouville (with power-law kernel), Caputo-Fabrizio (with exponential decay law) and Atangana-Baleanu (based on the non-singular and non-local generalized stretched Mittag-Leffler function) [1–19]. Involving these fractional derivatives, new applications have been discussed in view of the fractional calculus, for example, [11–21] and the references cited therein.

In general, most of the fractional differential equations and fractional delay differential equations are not solvable towards exacts solutions. Therefore, there has been significant interest in developing novel numerical methods for solving these equations. Several works have extending standard numerical methods such as the Adams-Bashforth method (based on an iterative algorithm of prediction and correction) to solve these equations [22,23]. In [24,25], the authors introduced a novel predictor-corrector method to numerically solve fractional differential equations. The author showed the accurate and time efficient compared with other methods. Also, in [26], the authors studied the system of first order delay differential using spline functions, and studied the stability and the error analysis.

Samko in 1993 introduced the variable-order fractional differential equations. These fractional operators can be considered as a generalization of fractional operators of constant order [27]. Several studies [28–32] have shown that, many complex physical problems can be described with great success via variable-order (VO) derivatives. In [33], the authors introduced the delayed two parameters Mittag-Leffler type matrix function and obtained an explicit formula of solutions to linear nonhomogeneous fractional delay differential equations via the variation of constants method. In [34], the authors developed conditioned pseudospectral schemes for solving fractional delay differential equations. Based on artificial neural networks, in [35], approximate solutions of the fractional delay differential equations (linear systems with delay, nonlinear systems with delay and Newton-Leipnik oscillator) were obtained. Synaptic weights were optimized using the Levenberg-Marquardt algorithm. The solution of the neural network was compared with the analytical solutions and the numerical simulations obtained through the Adams-Bashforth-Moulton method. Another numerical implementations of VO fractional derivatives are given in [36–38]. Recently, the authors in [39] proposed novel generalize numerical schemes for simulating variable-order fractional differential operators with power-law, exponential-law and Mittag-Leffler kernel. These schemes combine the fundamental theorem of fractional calculus and the two-step Lagrange polynomial interpolation [40,41]. Numerical examples were applied

* Corresponding author.
E-mail address: jgomez@cenidet.edu.mx (J.F. Gómez-Aguilar).

https://doi.org/10.1016/j.chaos.2019.06.009
0960-0779/

BLOOD GLUCOSE PREDICTION WITH A FRACTIONAL ORDER NEURAL NETWORK

Carlos Jesus Zuñiga-Aguilar[1,2], José Francisco Gómez-Aguilar[2,3], Sylvia Franc[4], Guillaume Charpentier[4], Maéva Doron[5], Pierre Yves Benhamou[6], Héctor Manuel Romero-Ugalde[1]

[1] Diabeloop SA, 155 Cours Berriat, F-38000 Grenoble, France; [2] Centro Nacional de Investigación y Desarrollo Tecnologico, Tecnologico Nacional de Mexico, Interior Internado Palmira S/N, Col. Palmira, C. P. 62490, Cuernavaca, Morelos, Mexico; [3] CONACyt, Interior Internado Palmira S/N, Col. Palmira, C. P. 62490, Cuernavaca, Morelos, Mexico; [4] Centre Hospitalier Sud-Francilien, Department of Diabetes and Endocrinology, Corbeil-Essonnes, France and Centre d'Etudes et de Recherche pour l'Intensification du Traitement du Diabète (CERITD), Corbeil-Essonnes, France; [5] Univ. Grenoble, CEA, LETI, F-38000, Grenoble, France; [6] CHU Grenoble/ Endocrinology and Université Grenoble Alpes, Grenoble, France

BACKGROUND AND AIMS

Blood Glucose (BG) prediction models need to be improved in order to ameliorate BG regulation. Neural Networks (NN) and fractional order calculus are powerful tools for black box modeling.

The aim of this work is to combine both approaches to propose the first NN model with fractional order learning algorithm to improve BG prediction.

METHOD

Clinical trial NCT02987556

- Multicenter open-label randomized controlled crossover study.
- Performed at 12 hospitals in France.
- Patients with:
  - glycated haemoglobin (HbA1c) < 10%
  - already treated with insulin pump therapy (> 6 months)

Database

- Ten T1D patients (aged > 18 years) wearing the DBLG1 System.
- Ten days of data for training
- Five days of data unseen by the NN during training for test.

The NN model *(Figure 1)*, which uses BG, Insulin on board, and carbohydrates on board as inputs, consists of a recurrent three-layer NN with 2-2-1 neurons in the input, hidden and output layers, respectively.

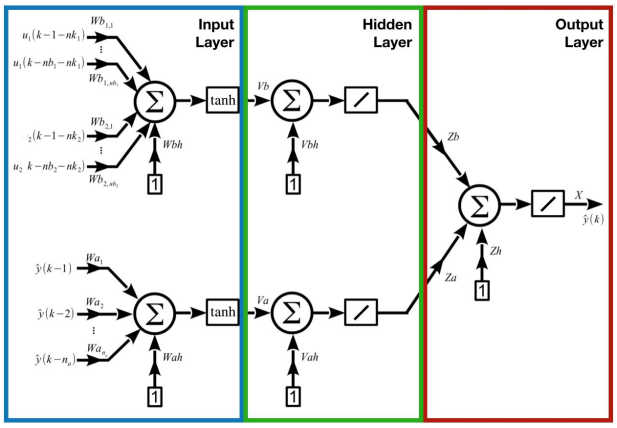


***Figure 1** - 2-2-1 recurrent neural network model*

The NN was trained by the following fractional order learning rules which were derived by the Grünwald-Letnikov fractional derivative [4].

$$Wa(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Wa(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Wa(t_k)$$

$$Vb(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Vb(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Vb(t_k)$$

$$Wah(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Wah(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Wah(t_k)$$

$$Vbh(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Vbh(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Vbh(t_k)$$

$$Wb_1(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Wb_1(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Wb_1(t_k)$$

$$Za(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Za(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Za(t_k)$$

$$Wb_2(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Wb_2(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Wb_2(t_k)$$

$$Zb(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Zb(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Zb(t_k)$$

$$Wbh(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Wbh(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Wbh(t_k)$$

$$Zh(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Zh(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Zh(t_k)$$

$$Va(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Va(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Va(t_k)$$

$$X(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial X(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} X(t_k)$$

$$Vah(t_{k+1}) = -\eta(t_k)\frac{\partial E}{\partial Vah(t_k)}h^\alpha - \sum_{j=0}^{k} c_j^{(\alpha)} Vah(t_k)$$

RMSE was computed to evaluate accuracy on BG prediction 30-, and 60-min-ahead on the 5 days of test (10 patients).

RESULTS

Figure 2 shows RMSE reached by the proposed NN-based model on the 10 test subsets (10 patients, 5 days).

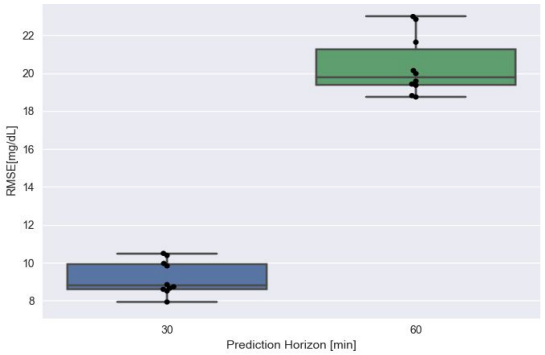


***Figure 2** - RMSE boxplots. Each boxplot is composed of 10 points (10 patients). Each point is the mean RMSE computed during the 5 days of validation.*

As expected, RMSE ± std increases when prediction horizon increases.

Table 1 displays results reached by the proposed model and also results reported in the literature.

Reference	Method	RMSE (mean ± std) [mg/dL]	
		30 min	60 min
Li et al. (2018) [1]	LSTM-CNN	21 ± 2.35	33.27 ± 4.79
Chen et al. (2018) [2]	DRNN-LSTM	19.04	-
Martinson et al. (2018) [3]	LSTM	20 ± 2.5	33.2 ± 3.2
Proposed NN model	**NN**	**9.19 ± 0.89**	**20.35 ± 1.5**

***Table 1.** Accuracy reached by the proposed FONN model vs performance reported in the literature.*

In *Table 1* we can observe that for the two prediction horizons the proposed NN model reached better performance than other methods included in the comparison.

ACKNOWLEDGEMENT

Carlos Jesús Zúñiga Aguilar acknowledges the support provided by CONACyT through the assignment of doctoral fellowship. José Francisco Gómez Aguilar acknowledges the support provided by CONACyT: Cátedras CONACyT para jóvenes investigadores 2014.

CONCLUSIONS

This work presented a NN-based BG prediction model trained by a fractional order learning algorithm.
Compared with results reported in the literature the proposed model:

- reached the best performance on predictions 30- and 60-min-ahead and,
- is the simplest of the compared approaches.

In future works, the possibility to predict hypo- and hyperglycemia events by the proposed model will be evaluated.

REFERENCES:

[1] Li, K., Daniels, J., Liu, C., Herrero-Vinas, P., & Georgiou, P. (2019). Convolutional recurrent neural networks for glucose prediction. IEEE Journal of Biomedical and Health Informatics.
[2] Chen, J., Li, K., Herrero, P., Zhu, T., & Georgiou, P. (2018). Dilated Recurrent Neural Network for Short-time Prediction of Glucose Concentration. In KHD@ IJCAI (pp. 69-73).
[3] Martinsson, J., Schliep, A., Eliasson, B., Meijner, C., Persson, S., & Mogren, O. (2018). Automatic blood glucose prediction with confidence using recurrent neural networks. In 3rd International Workshop on Knowledge Discovery in Healthcare Data, KDH@ IJCAI-ECAI 2018, 13 July 2018 (pp. 64-68).
[4] Petráš, I. (2011). Fractional-order nonlinear systems: modeling, analysis and simulation. Springer Science & Business Media.

H. M. Romero-Ugalde / hector@diabeloop.fr, hector.m.romero.ugalde@gmail.com

Bibliografía

[1] Z. Liu, A. Hansson, and L. Vandenberghe, "Nuclear norm system identification with missing inputs and outputs," *Systems & Control Letters*, vol. 62, no. 8, pp. 605–612, 2013.

[2] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[3] I. Podlubny, "Geometric and physical interpretation of fractional integration and fractional differentiation," *arXiv preprint math/0110241*, 2001.

[4] L. Ljung, "System identification," *Wiley Encyclopedia of Electrical and Electronics Engineering*, pp. 1–19, 1999.

[5] P. Vázquez-Guerrero, "Identificación de modelos wiener-hammerstein utilizando cálculo fraccionario," 2016.

[6] L. C. A.-D. L. Ríos, "Identificación de modelos hammerstein-wiener utilizando métodos de cálculo fraccionario," 2019.

[7] I. E. Guerrero-Ozuna, "Análisis dinámico de un sistema de regeneradores de energía para establecer acciones de control, enfoque basado en cálculo fraccionario.," 2016.

[8] H. M. Romero, "Identificación de sistemas utilizando redes neuronales: Un enfoque basado en balance entre sencillez, precisión y costo computacional.," 2013.

[9] U. Flores, "Identificación de sistemas no lineales mediante las estructuras narx y hammerstein-wiener," 2011.

[10] C. J. Zúñiga-Aguilar, "Estimador m de huber para la identificación de un modelo en red neuronal.," 2016.

[11] D. Yu, Y. Wang, H. Liu, K. Jermsittiparsert, and N. Razmjooy, "System identification of pem fuel cells using an improved elman neural network and a new hybrid optimization algorithm," *Energy Reports*, vol. 5, pp. 1365–1374, 2019.

[12] R. Kumar, S. Srivastava, J. Gupta, and A. Mohindru, "Comparative study of neural networks for dynamic nonlinear systems identification," *Soft Computing*, vol. 23, no. 1, pp. 101–114, 2019.

[13] R.-T. Wu and M. R. Jahanshahi, "Data fusion approaches for structural health monitoring and system identification: past, present, and future," *Structural Health Monitoring*, vol. 19, no. 2, pp. 552–586, 2020.

[14] C. Luo, C. Tan, X. Wang, and Y. Zheng, "An evolving recurrent interval type-2 intuitionistic fuzzy neural network for online learning and time series prediction," *Applied Soft Computing*, vol. 78, pp. 150–163, 2019.

[15] J. Chen, Z. Zeng, and P. Jiang, "Global mittag-leffler stability and synchronization of memristor-based fractional-order neural networks," *Neural Networks*, vol. 51, pp. 1–8, 2014.

[16] E. Kaslik and S. Sivasundaram, "Nonlinear dynamics and chaos in fractional-order neural networks," *Neural Networks*, vol. 32, pp. 245–256, 2012.

[17] J. Yu, C. Hu, and H. Jiang, "α-stability and α-synchronization for fractional-order neural networks," *Neural Networks*, vol. 35, pp. 82–87, 2012.

[18] L. Chen, Y. Chai, R. Wu, T. Ma, and H. Zhai, "Dynamic analysis of a class of fractional-order neural networks with delay," *Neurocomputing*, vol. 111, pp. 190–194, 2013.

[19] W.-j. Shan and W. Tang, "A neural network fractional order pid controller for folpd process," pp. 10459–10463, 2016.

[20] S. Das, S. Saha, A. Mukherjee, I. Pan, and A. Gupta, "Adaptive gain and order scheduling of optimal fractional order pilamdadμ controllers with radial basis function neural-network," pp. 1–6, 2011.

[21] I. Petráš, "A note on fractional-order non-linear controller: Possible neural network approach to design," pp. 603–608, 2016.

[22] M. Pakdaman, A. Ahmadian, S. Effati, S. Salahshour, and D. Baleanu, "Solving differential equations of fractional order using an optimization technique based on training artificial neural network," *Applied Mathematics and Computation*, vol. 293, pp. 81–95, 2017.

[23] M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, "A new stochastic approach for solution of riccati differential equation of fractional order," *Annals of Mathematics and Artificial Intelligence*, vol. 60, no. 3-4, pp. 229–250, 2010.

[24] M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, "Solution of fractional order system of bagley-torvik equation using evolutionary computational intelligence," *Mathematical Problems in Engineering*, vol. 2011, 2011.

[25] M. A. Z. Raja, M. A. Manzar, and R. Samar, "An efficient computational intelligence approach for solving fractional order riccati equations using ann and sqp," *Applied Mathematical Modelling*, vol. 39, no. 10-11, pp. 3075–3093, 2015.

[26] D. Sierociuk and A. Dzieliński, "Fractional kalman filter algorithm for the states, parameters and order of fractional system estimation," 2006.

[27] J. Sabatier, M. Aoun, A. Oustaloup, G. Grégoire, F. Ragot, and P. Roy, "Fractional system identification for lead acid battery state of charge estimation," *Signal processing*, vol. 86, no. 10, pp. 2645–2657, 2006.

[28] B. Wang, S. E. Li, H. Peng, and Z. Liu, "Fractional-order modeling and parameter identification for lithium-ion batteries," *Journal of Power Sources*, vol. 293, pp. 151–161, 2015.

[29] A. Khadhraoui, K. Jelassi, and J.-C. Trigeassou, "Least squares and instrumental variable techniques for global identification of fractional differential equation," in *2014 International Conference on Electrical Sciences and Technologies in Maghreb (CISTEM)*, pp. 1–5, IEEE, 2014.

[30] L. Vanbeylen, "A fractional approach to identify wiener–hammerstein systems," *Automatica*, vol. 50, no. 3, pp. 903–909, 2014.

[31] D. Ivanov, "Identification discrete fractional order hammerstein systems," in *2015 International Siberian Conference on Control and Communications (SIBCON)*, pp. 1–4, IEEE, 2015.

[32] A. Dzieliński, D. Sierociuk, and G. Sarwas, "Some applications of fractional order calculus," *Bulletin of the Polish Academy of Sciences: Technical Sciences*, vol. 58, no. 4, pp. 583–592, 2010.

[33] D. Sierociuk, G. Sarwas, and A. Dzieliński, "Discrete fractional order artificial neural network," *acta mechanica et automatica*, vol. 5, pp. 128–132, 2011.

[34] G. Sarwas, D. Sierociuk, and A. Dzieliński, "Ultracapacitor modeling and control with discrete fractional order artificial neural network," in *Proceedings of the 13th International Carpathian Control Conference (ICCC)*, pp. 617–622, IEEE, 2012.

[35] D. Sierociuk and I. Petráš, "Modeling of heat transfer process by using discrete fractional-order neural networks," pp. 146–150, 2011.

[36] A. Boroomand and M. B. Menhaj, "On-line nonlinear systems identification of coupled tanks via fractional differential neural networks," pp. 2185–2189, 2009.

[37] B. Goodwine and K. Leyden, "Recent results in fractional-order modeling in multi-agent systems and linear friction welding," *IFAC-PapersOnLine*, vol. 48, no. 1, pp. 380–381, 2015.

[38] J. Mayes, *Reduction and approximation in large and infinite potential-driven flow networks*. Citeseer, 2012.

[39] S. K. Verma and S. K. Nagar, "Approximation and order reduction of fractional order siso system," pp. 1–6, 2016.

[40] M. Taha, D. Abualnadi, and O. Hasan, "Model order reduction using fractional order systems," pp. 199–204, 2016.

[41] J. P. Hollkamp, M. Sen, and F. Semperlotti, "Model-order reduction of lumped parameter systems via fractional calculus," *Journal of Sound and Vibration*, vol. 419, pp. 526–543, 2018.

[42] H. M. R. Ugalde, J.-C. Carmona, V. M. Alvarado, and J. Reyes-Reyes, "Neural network design and model reduction approach for black box nonlinear system identification with reduced number of parameters," *Neurocomputing*, vol. 101, pp. 170–180, 2013.

[43] H. M. R. Ugalde, J.-C. Carmona, J. Reyes-Reyes, V. M. Alvarado, and C. Corbier, "Balanced simplicity–accuracy neural network model families for system identification," *Neural computing and applications*, vol. 26, no. 1, pp. 171–186, 2015.

[44] H. M. R. Ugalde, J.-C. Carmona, J. Reyes-Reyes, V. M. Alvarado, and J. Mantilla, "Computational cost improvement of neural network models in black box nonlinear system identification," *Neurocomputing*, vol. 166, pp. 96–108, 2015.

[45] G. W. Leibniz, *Leibnizens mathematische schriften*, vol. 1. Asher, 1849.

[46] M. Weilbeer, *Efficient numerical methods for fractional differential equations and their analytical background*. Papierflieger, 2005.

[47] L. Euler, "De progressionibus transcendentibus seu quarum termini generales algebraice dari nequeunt," *Commentarii academiae scientiarum Petropolitanae*, pp. 36–57, 1738.

[48] J. L. Lagrange, *Sur une nouvelle espèce de calcul rélatif à la différentiation & à l'intégration des quantités variables*. Académie royale des sciences et belles lettres, 1775.

[49] S. F. Lacroix, "Traité du calcul différentiel et du calcul intégral tome 3," *Traité du calcul différentiel et du calcul intégral*, 1819.

[50] P. S. Laplace, *Théorie analytique des probabilités*. Courcier, 1820.

[51] J. B. J. baron Fourier, *Théorie analytique de la chaleur*. F. Didot, 1822.

[52] N. H. Abel, "Auflösung einer mechanischen aufgabe.," *Journal für die reine und angewandte Mathematik*, vol. 1826, no. 1, pp. 153–157, 1826.

[53] N. Abel, "Solution de quelques problèmes à l'aide d'intégrales définies," *Oeuvres*, vol. 1, pp. 11–27, 1881.

[54] J. Liouville, *Mémoire sur quelques questions de géométrie et de mécanique, et sur un nouveau genre de calcul pour résoudre ces questions*. 1832.

[55] A. K. Grunwald, "Uber"begrente"derivationen und deren anwedung," *Zangew Math und Phys*, vol. 12, pp. 441–480, 1867.

[56] A. Letnikov, "An explanation of the concepts of the theory of differentiation of arbitrary index (russian), moscow matem," *Sbornik*, vol. 6, pp. 413–445, 1872.

[57] M. Caputo and M. Fabrizio, "On the notion of fractional derivative and applications to the hysteresis phenomena," *Meccanica*, vol. 52, no. 13, pp. 3043–3052, 2017.

[58] I. Podlubny, *Fractional differential equations: an introduction to fractional derivatives, fractional differential equations, to methods of their solution and some of their applications*. Elsevier, 1998.

[59] A. Loverro, "Fractional calculus: history, definitions and applications for the engineer," *Rapport technique, Univeristy of Notre Dame: Department of Aerospace and Mechanical Engineering*, pp. 1–28, 2004.

[60] M. Caputo, "Linear models of dissipation whose q is almost frequency independent," *Annals of Geophysics*, vol. 19, no. 4, pp. 383–393, 1966.

[61] M. Caputo and M. Fabrizio, "Applications of new time and spatial fractional derivatives with exponential kernels," *Progr. Fract. Differ. Appl*, vol. 2, no. 2, pp. 1–11, 2016.

[62] M. Caputo and M. Fabrizio, "A new definition of fractional derivative without singular kernel," *Progr. Fract. Differ. Appl*, vol. 1, no. 2, pp. 1–13, 2015.

[63] A. Atangana and D. Baleanu, "New fractional derivatives with nonlocal and non-singular kernel: theory and application to heat transfer model," *arXiv preprint arXiv:1602.03408*, 2016.

[64] A. Atangana, "Fractal-fractional differentiation and integration: Connecting fractal calculus and fractional calculus to predict complex system," *Chaos, solitons & fractals*, vol. 102, pp. 396–406, 2017.

[65] A. Atangana and A. Shafiq, "Differential and integral operators with constant fractional order and variable fractional dimension," *Chaos, Solitons & Fractals*, vol. 127, pp. 226–243, 2019.

[66] C. Zecchin, "Online glucose prediction in type-1 diabetes by neural network models," 2014.

[67] S. Grossberg, *How does a brain build a cognitive code?* Springer, 1982.

[68] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541)*, vol. 2, pp. 985–990, IEEE, 2004.

[69] P. Baldi and R. Vershynin, "The capacity of feedforward neural networks," *Neural networks*, vol. 116, pp. 288–311, 2019.

[70] G. Dudek, "Generating random weights and biases in feedforward neural networks with random hidden nodes," *Information Sciences*, vol. 481, pp. 33–56, 2019.

[71] H. F. Lui and W. R. Wolf, "Construction of reduced-order models for fluid flows using deep feedforward neural networks," *Journal of Fluid Mechanics*, vol. 872, pp. 963–994, 2019.

[72] D. E. Millán-Ocampo, A. Parrales-Bahena, J. G. González-Rodríguez, S. Silva-Martínez, J. Porcayo-Calderón, and J. A. Hernández-Pérez, "Modelling of behavior for inhibition corrosion of bronze using artificial neural network (ann)," *Entropy*, vol. 20, no. 6, p. 409, 2018.

[73] S. Haykin, *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[74] S. Haykin, *Neural Networks and Learning Machines, 3/E*. Pearson Education India, 2010.

[75] Z. Zhang, "Parameter estimation techniques: A tutorial with application to conic fitting," *Image and vision Computing*, vol. 15, no. 1, pp. 59–76, 1997.

[76] Y. Shi *et al.*, "Particle swarm optimization: developments, applications and resources," vol. 1, pp. 81–86, 2001.

[77] I. C. Trelea, "The particle swarm optimization algorithm: convergence analysis and parameter selection," *Information processing letters*, vol. 85, no. 6, pp. 317–325, 2003.

[78] A. Bate, M. Lindquist, I. R. Edwards, S. Olsson, R. Orre, A. Lansner, and R. M. De Freitas, "A bayesian neural network method for adverse drug reaction signal generation," *European journal of clinical pharmacology*, vol. 54, no. 4, pp. 315–321, 1998.

[79] J. Cai, X. Ma, L. Li, Y. Yang, H. Peng, and X. Wang, "Chaotic ant swarm optimization to economic dispatch," *Electric Power Systems Research*, vol. 77, no. 10, pp. 1373–1380, 2007.

[80] E. Valian, S. Mohanna, and S. Tavakoli, "Improved cuckoo search algorithm for feedforward neural network training," *International Journal of Artificial Intelligence & Applications*, vol. 2, no. 3, pp. 36–43, 2011.

[81] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[82] C. Darken, J. Chang, J. Moody, *et al.*, "Learning rate schedules for faster stochastic gradient search," vol. 2, 1992.

[83] H.-C. Ma, D.-D. Yao, and X.-F. Peng, "Exact solutions of non-linear fractional partial differential equations by fractional sub-equation method," *Thermal Science*, vol. 19, no. 4, pp. 1239–1244, 2015.

[84] S. T. Mohyud-Din, T. Nawaz, E. Azhar, and M. A. Akbar, "Fractional sub-equation method to space–time fractional calogero-degasperis and potential kadomtsev-petviashvili equations," *Journal of Taibah University for Science*, vol. 11, no. 2, pp. 258–263, 2017.

[85] E. M. Zayed and H. M. A. Rahman, "On using the modified variational iteration method for solving the nonlinear coupled equations in the mathematical physics," *Ricerche di matematica*, vol. 59, no. 1, pp. 137–159, 2010.

[86] Y. Zhang, C. Cattani, and X.-J. Yang, "Local fractional homotopy perturbation method for solving non-homogeneous heat conduction equations in fractal domains," *Entropy*, vol. 17, no. 10, pp. 6753–6764, 2015.

[87] J. Singh, P. K. Gupta, K. Rai, *et al.*, "Homotopy perturbation method to space–time fractional solidification in a finite slab," *Applied Mathematical Modelling*, vol. 35, no. 4, pp. 1937–1945, 2011.

[88] M. Kushwaha *et al.*, "Homotopy perturbation method for a limit case stefan problem governed by fractional diffusion equation," *Applied Mathematical Modelling*, vol. 37, no. 5, pp. 3589–3599, 2013.

[89] D. Kumar, J. Singh, and S. Kumar, "Analytic and approximate solutions of space-time fractional telegraph equations via laplace transform," *Walailak Journal of Science and Technology (WJST)*, vol. 11, no. 8, pp. 711–728, 2014.

[90] S. Irandoust-Pakchin, M. Javidi, and H. Kheiri, "Analytical solutions for the fractional nonlinear cable equation using a modified homotopy perturbation and separation of variables methods," *Computational Mathematics and Mathematical Physics*, vol. 56, no. 1, pp. 116–131, 2016.

[91] D. Kumar, J. Singh, and S. Kumar, "Numerical computation of nonlinear fractional zakharov–kuznetsov equation arising in ion-acoustic waves," *Journal of the Egyptian Mathematical Society*, vol. 22, no. 3, pp. 373–378, 2014.

[92] Y. Xu, Y. Pei, and F. Dong, "An adaptive tikhonov regularization parameter choice method for electrical resistance tomography," *Flow Measurement and Instrumentation*, vol. 50, pp. 1–12, 2016.

[93] Z. Wang, H. Liu, N. Huang, Q. Sun, and X. Li, "Mid-infrared raman amplification and wavelength conversion in dispersion engineered silicon-on-sapphire waveguides," *Journal of Optics*, vol. 16, no. 1, p. 015206, 2013.

[94] N. Khodabakhshi, S. M. Vaezpour, and D. Baleanu, "Numerical solutions of the initial value problem for fractional differential equations by modification of the adomian decomposition method," *Fractional Calculus and Applied Analysis*, vol. 17, no. 2, pp. 382–400, 2014.

[95] S. Saha Ray and A. Gupta, "Numerical solution of fractional partial differential equation of parabolic type with dirichlet boundary conditions using two-dimensional legendre wavelets method," *Journal of Computational and Nonlinear Dynamics*, vol. 11, no. 1, 2016.

[96] M. Yi and J. Huang, "Wavelet operational matrix method for solving fractional differential equations with variable coefficients," *Applied Mathematics and Computation*, vol. 230, pp. 383–394, 2014.

[97] H. Jafari and H. K. Jassim, "Numerical solutions of telegraph and laplace equations on cantor sets using local fractional laplace decomposition method," *International Journal of Advances in Applied Mathematics and Mechanics*, vol. 2, no. 3, pp. 144–151, 2015.

[98] P. Goswami and R. T. Alqahtani, "On the solution of local fractional differential equations using local fractional laplace variational iteration method," *Mathematical Problems in Engineering*, vol. 2016, 2016.

[99] C. Li and G. Chen, "Chaos and hyperchaos in the fractional-order rössler equations," *Physica A: Statistical Mechanics and its Applications*, vol. 341, pp. 55–61, 2004.

[100] A. Bhrawy and M. Zaky, "An improved collocation method for multi-dimensional space–time variable-order fractional schrödinger equations," *Applied Numerical Mathematics*, vol. 111, pp. 197–218, 2017.

[101] D. Valério and J. S. Da Costa, "Variable-order fractional derivatives and their numerical approximations," *Signal Processing*, vol. 91, no. 3, pp. 470–483, 2011.

[102] A. Atangana, "On the stability and convergence of the time-fractional variable order telegraph equation," *Journal of Computational Physics*, vol. 293, pp. 104–114, 2015.

[103] S. Yaghoobi, B. P. Moghaddam, and K. Ivaz, "An efficient cubic spline approximation for variable-order fractional differential equations with time delay," *Nonlinear Dynamics*, vol. 87, no. 2, pp. 815–826, 2017.

[104] A. Atangana and R. T. Alqahtani, "Stability analysis of nonlinear thin viscous fluid sheet flow equation with local fractional variable order derivative," *Journal of Computational and Theoretical Nanoscience*, vol. 13, no. 5, pp. 2710–2717, 2016.

[105] B. Parsa Moghaddam, S. Yaghoobi, and J. Tenreiro Machado, "An extended predictor–corrector algorithm for variable-order fractional delay differential equations," *Journal of Computational and Nonlinear Dynamics*, vol. 11, no. 6, 2016.

[106] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[107] H. Qu and X. Liu, "A numerical method for solving fractional differential equations by using neural network," *Advances in Mathematical Physics*, vol. 2015, 2015.

[108] B. S. Kashkaria and M. I. Syam, "Evolutionary computational intelligence in solving a class of nonlinear volterra–fredholm integro-differential equations," *Journal of Computational and Applied Mathematics*, vol. 311, pp. 314–323, 2017.

[109] B. S. T. Alkahtani, "Chua's circuit model with atangana–baleanu derivative with fractional order," *Chaos, Solitons & Fractals*, vol. 89, pp. 547–551, 2016.

[110] C. Zúñiga-Aguilar, H. Romero-Ugalde, J. Gómez-Aguilar, R. Escobar-Jiménez, and M. Valtierra-Rodríguez, "Solving fractional differential equations of variable-order involving operators with mittag-leffler kernel using artificial neural networks," *Chaos, Solitons & Fractals*, vol. 103, pp. 382–403, 2017.

[111] C. Zúñiga-Aguilar, A. Coronel-Escamilla, J. Gómez-Aguilar, V. Alvarado-Martínez, and H. Romero-Ugalde, "New numerical approximation for solving fractional delay differential equations of variable order using artificial neural networks," *The European Physical Journal Plus*, vol. 133, no. 2, p. 75, 2018.

[112] M. E. Yalçin, J. A. Suykens, J. Vandewalle, and S. Özoğuz, "Families of scroll grid attractors," *International Journal of Bifurcation and Chaos*, vol. 12, no. 01, pp. 23–41, 2002.

[113] S. Ma, Y. Xu, and W. Yue, "Numerical solutions of a variable-order fractional financial system," *Journal of Applied Mathematics*, vol. 2012, 2012.

[114] H. Ning and X. Jing, "Identification of partially known non-linear stochastic spatio-temporal dynamical systems by using a novel partially linear kernel method," *IET Control Theory & Applications*, vol. 9, no. 1, pp. 21–33, 2014.

[115] G. Pillonetto, T. Chen, A. Chiuso, G. De Nicolao, and L. Ljung, "Regularized linear system identification using atomic, nuclear and kernel-based norms: The role of the stability constraint," *Automatica*, vol. 69, pp. 137–149, 2016.

[116] C. Corbier and H. M. R. Ugalde, "Low-order control-oriented modeling of piezoelectric actuator using huberian function with low threshold: pseudolinear and neural network models," *Nonlinear Dynamics*, vol. 85, no. 2, pp. 923–940, 2016.

[117] S.-T. Pan and C.-C. Lai, "Identification of chaotic systems by neural network with hybrid learning algorithm," *Chaos, Solitons & Fractals*, vol. 37, no. 1, pp. 233–244, 2008.

[118] A. Cochocki and R. Unbehauen, *Neural networks for optimization and signal processing*. John Wiley & Sons, Inc., 1993.

[119] S.-T. Tzeng, "Design of fuzzy wavelet neural networks using the ga approach for function approximation and system identification," *Fuzzy Sets and Systems*, vol. 161, no. 19, pp. 2585–2596, 2010.

[120] M. Witters and J. Swevers, "Black-box model identification for a continuously variable, electro-hydraulic semi-active damper," *Mechanical Systems and Signal Processing*, vol. 24, no. 1, pp. 4–18, 2010.

[121] J. de Jesús Rubio, "Fuzzy slopes model of nonlinear systems with sparse data," *Soft Computing*, vol. 19, no. 12, pp. 3507–3514, 2015.

[122] W. Yu and X. Li, "Fuzzy identification using fuzzy neural networks with stable learning algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 12, no. 3, pp. 411–420, 2004.

[123] S. M. R. Loghmanian, H. Jamaluddin, R. Ahmad, R. Yusof, and M. Khalid, "Structure optimization of neural network for dynamic system modeling using multi-objective genetic algorithm," *Neural Computing and Applications*, vol. 21, no. 6, pp. 1281–1295, 2012.

[124] J. P. Donate, X. Li, G. G. Sánchez, and A. S. de Miguel, "Time series forecasting by evolving artificial neural networks with genetic algorithms, differential evolution and estimation of distribution algorithm," *Neural Computing and Applications*, vol. 22, no. 1, pp. 11–20, 2013.

[125] D. Leite, P. Costa, and F. Gomide, "Evolving granular neural networks from fuzzy data streams," *Neural Networks*, vol. 38, pp. 1–16, 2013.

[126] J. Rubio, A. Pascual-Iserte, J. del Olmo Alòs, and J. Vidal, "Dynamic base station switch on/off strategies for sustainable wireless networks," pp. 289–293, 2014.

[127] L. Biao, L. Qing-chun, J. Zhen-hua, and N. Sheng-fang, "System identification of locomotive diesel engines with autoregressive neural network," pp. 3417–3421, 2009.

[128] C. Endisch, P. Stolze, P. Endisch, C. Hackl, and R. Kennel, "Levenberg-marquardt-based obs algorithm using adaptive pruning interval for system identification with dynamic neural networks," pp. 3402–3408, 2009.

[129] H.-W. Ge, F. Qian, Y.-C. Liang, W.-l. Du, and L. Wang, "Identification and control of nonlinear systems by a dissimilation particle swarm optimization-based elman neural network," *Nonlinear Analysis: Real World Applications*, vol. 9, no. 4, pp. 1345–1360, 2008.

[130] C.-K. Goh, E.-J. Teoh, and K. C. Tan, "Hybrid multiobjective evolutionary design for artificial neural networks," *IEEE Transactions on Neural Networks*, vol. 19, no. 9, pp. 1531–1548, 2008.

[131] W. Xie, Y. Zhu, Z. Zhao, and Y. Wong, "Nonlinear system identification using optimized dynamic neural network," *Neurocomputing*, vol. 72, no. 13-15, pp. 3277–3287, 2009.

[132] L. dos Santos Coelho and M. W. Pessôa, "Nonlinear identification using a b-spline neural network and chaotic immune approaches," *Mechanical Systems and Signal Processing*, vol. 23, no. 8, pp. 2418–2434, 2009.

[133] C. Zúñiga-Aguilar, J. Gómez-Aguilar, V. Alvarado-Martínez, and H. Romero-Ugalde, "Fractional order neural networks for system identification," *Chaos, Solitons & Fractals*, vol. 130, p. 109444, 2020.

[134] S. Qureshi, A. Yusuf, A. A. Shaikh, M. Inc, and D. Baleanu, "Fractional modeling of blood ethanol concentration system with real data application," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 29, no. 1, p. 013143, 2019.

[135] S. Qureshi and A. Yusuf, "Fractional derivatives applied to mseir problems: Comparative study with real world data," *The European Physical Journal Plus*, vol. 134, no. 4, p. 171, 2019.

[136] K. Favoreel-Leuven, *Data of a laboratory setup acting like a hair dryer*, 1994 (accessed April 20, 2020). https://homes.esat.kuleuven.be/~smc/daisy/daisydata.html.

[137] J. Noël and M. Schoukens, "Hysteretic benchmark with a dynamic nonlinearity," in *Workshop on nonlinear system identification benchmarks*, pp. 7–14, 2016.

[138] A. Banakar, "Lyapunov stability analysis of gradient descent-learning algorithm in network training," *ISRN Applied Mathematics*, vol. 2011, 2011.

[139] A. Lyapunov, "Lectures in theoretical mechanics," *Kiev Izdatel Naukova Dumka*, 1982.

[140] Y. Li, Y. Chen, and I. Podlubny, "Stability of fractional-order nonlinear dynamic systems: Lyapunov direct method and generalized mittag–leffler stability," *Computers & Mathematics with Applications*, vol. 59, no. 5, pp. 1810–1821, 2010.

[141] J. F. Gómez-Aguilar, H. Yépez-Martínez, C. Calderón-Ramón, I. Cruz-Orduña, R. F. Escobar-Jiménez, and V. H. Olivares-Peregrino, "Modeling of a mass-spring-damper system by fractional derivatives with and without a singular kernel," *Entropy*, vol. 17, no. 9, pp. 6289–6303, 2015.

[142] R. Bouc, *Forced vibrations of mechanical systems with hysteresis*. 1967.

[143] Y.-K. Wen, "Method for random vibration of hysteretic systems," *Journal of the engineering mechanics division*, vol. 102, no. 2, pp. 249–263, 1976.

[144] J. Belz, T. Münker, T. O. Heinz, G. Kampmann, and O. Nelles, "Automatic modeling with local model networks for benchmark processes," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 470–475, 2017.

[145] A. Esfahani, P. Dreesen, K. Tiels, J.-P. Noël, and J. Schoukens, "Using a polynomial decoupling algorithm for state-space identification of a bouc-wen system," 2016.

[146] D. Boiroux, A. K. Duun-Henriksen, S. Schmidt, K. Nørgaard, N. K. Poulsen, H. Madsen, and J. B. Jørgensen, "Adaptive control in an artificial pancreas for people with type 1 diabetes," *Control Engineering Practice*, vol. 58, pp. 332–342, 2017.

[147] A. Albisser, S. Sakkal, and C. Wright, "Home blood glucose prediction: validation, safety, and efficacy testing in clinical diabetes," *Diabetes technology & therapeutics*, vol. 7, no. 3, pp. 487–496, 2005.

[148] I.-Y. Jo, S.-H. Yoo, D. Y. Lee, C.-Y. Park, and E. M. Kim, "Diabetes management via a mobile application: a case report," *Clinical nutrition research*, vol. 6, no. 1, pp. 61–67, 2017.

[149] S. Garg, R. L. Brazg, T. S. Bailey, B. A. Buckingham, R. H. Slover, D. C. Klonoff, J. Shin, J. B. Welsh, and F. R. Kaufman, "Reduction in duration of hypoglycemia by automatic suspension of insulin delivery: the in-clinic aspire study," *Diabetes technology & therapeutics*, vol. 14, no. 3, pp. 205–209, 2012.

[150] M. Stenerson, F. Cameron, D. M. Wilson, B. Harris, S. Payne, B. W. Bequette, and B. A. Buckingham, "The impact of accelerometer and heart rate data on hypoglycemia mitigation in type 1 diabetes," *Journal of diabetes science and technology*, vol. 8, no. 1, pp. 64–69, 2014.

[151] P. Y. Benhamou, E. Huneker, S. Franc, M. Doron, G. Charpentier, D. Consortium, *et al.*, "Customization of home closed-loop insulin delivery in adult patients with type 1 diabetes, assisted with structured remote monitoring: the pilot wp7 diabeloop study," *Acta diabetologica*, vol. 55, no. 6, pp. 549–556, 2018.

[152] P.-Y. Benhamou, S. Franc, Y. Reznik, C. Thivolet, P. Schaepelynck, E. Renard, B. Guerci, L. Chaillous, C. Lukas-Croisier, N. Jeandidier, *et al.*, "Closed-loop insulin delivery in adults with type 1 diabetes in real-life conditions: a 12-week multicentre, open-label randomised controlled crossover trial," *The Lancet Digital Health*, vol. 1, no. 1, pp. e17–e25, 2019.

[153] R. Hovorka, V. Canonico, L. J. Chassin, U. Haueter, M. Massi-Benedetti, M. O. Federici, T. R. Pieber, H. C. Schaller, L. Schaupp, T. Vering, *et al.*, "Nonlinear model predictive control of glucose concentration in subjects with type 1 diabetes," *Physiological measurement*, vol. 25, no. 4, p. 905, 2004.

[154] R. Hovorka, F. Shojaee-Moradie, P. V. Carroll, L. J. Chassin, I. J. Gowrie, N. C. Jackson, R. S. Tudor, A. M. Umpleby, and R. H. Jones, "Partitioning glucose distribution/transport, disposal, and endogenous production during ivgtt," *American Journal of Physiology-Endocrinology and Metabolism*, vol. 282, no. 5, pp. E992–E1007, 2002.

[155] R. Hovorka, H. Jayatillake, E. Rogatsky, V. Tomuta, T. Hovorka, and D. T. Stein, "Calculating glucose fluxes during meal tolerance test: a new computational approach," *American Journal of Physiology-Endocrinology and Metabolism*, vol. 293, no. 2, pp. E610–E619, 2007.

[156] P. Li, L. Yu, Q. Fang, and S.-Y. Lee, "A simplification of cobelli's glucose–insulin model for type 1 diabetes mellitus and its fpga implementation," *Medical & biological engineering & computing*, vol. 54, no. 10, pp. 1563–1577, 2016.

[157] B. Grosman, E. Dassau, H. C. Zisser, L. Jovanovič, and F. J. Doyle III, "Zone model predictive control: a strategy to minimize hyper-and hypoglycemic events," *Journal of diabetes science and technology*, vol. 4, no. 4, pp. 961–975, 2010.

[158] J. M. V. B. L. Sun, Qingnan and S. G. Mougiakakou, "Predicting blood glucose with an lstm and bi-lstm based deep neural network," pp. 1–5, 2018.

[159] K. Li, J. Daniels, C. Liu, P. Herrero-Vinas, and P. Georgiou, "Convolutional recurrent neural networks for glucose prediction," *IEEE journal of biomedical and health informatics*, 2019.

[160] S. Fiorini, C. Martini, D. Malpassi, R. Cordera, D. Maggi, A. Verri, and A. Barla, "Data-driven strategies for robust forecast of continuous glucose monitoring time-series," pp. 1680–1683, 2017.

[161] S. Coman, C. Boldisor, and L. Floroian, "Fractional adaptive control for a fractional-order insuline-glucose dynamic model," pp. 887–892, 2017.

[162] M. Goharimanesh, A. Lashkaripour, and A. Abouei Mehrizi, "Fractional order pid controller for diabetes patients," *Journal of Computational Applied Mechanics*, vol. 46, no. 1, pp. 69–76, 2015.

[163] I. N'Doye, H. Voos, M. Darouach, J. G. Schneider, and N. Knauf, "An unknown input fractional-order observer design for fractional-order glucose-insulin system," pp. 595–600, 2012.

[164] H. Delavari, H. Heydarinejad, and D. Baleanu, "Adaptive fractional-order blood glucose regulator based on high-order sliding mode observer," *IET Systems Biology*, vol. 13, no. 2, pp. 43–54, 2018.

[165] H. Heydarinejad, H. Delavari, and D. Baleanu, "Fuzzy type-2 fractional backstepping blood glucose control based on sliding mode observer," *International Journal of Dynamics and Control*, vol. 7, no. 1, pp. 341–354, 2019.

[166] C. Zecchin, A. Facchinetti, G. Sparacino, G. De Nicolao, and C. Cobelli, "Neural network incorporating meal information improves accuracy of short-time prediction of glucose concentration," *IEEE transactions on biomedical engineering*, vol. 59, no. 6, pp. 1550–1560, 2012.

[167] H. M. Romero-Ugalde, M. Garnotel, M. Doron, P. Jallon, G. Charpentier, S. Franc, E. Huneker, C. Simon, and S. Bonnet, "Arx model for interstitial glucose prediction during and after physical activities," *Control Engineering Practice*, vol. 90, pp. 321–330, 2019.

[168] X. Tang, L. Zhang, and X. Wang, "Sparse augmented lagrangian algorithm for system identification," *Neurocomputing*, vol. 330, pp. 403–411, 2019.

[169] M. Baumann, C. Weissinger, and H.-G. Herzog, "System identification and modeling of an automotive bidirectional dc/dc converter," pp. 1–5, 2019.

[170] S. Chen, S. Billings, and P. Grant, "Non-linear system identification using neural networks," *International journal of control*, vol. 51, no. 6, pp. 1191–1214, 1990.

[171] A. S. Poznyak, E. N. Sanchez, and W. Yu, *Differential neural networks for robust nonlinear control: identification, state estimation and trajectory tracking*. World Scientific, 2001.

[172] X. Han, W.-F. Xie, Z. Fu, and W. Luo, "Nonlinear systems identification using dynamic multi-time scale neural networks," *Neurocomputing*, vol. 74, no. 17, pp. 3428–3439, 2011.

[173] G. Khalaj, H. Yoozbashizadeh, A. Khodabandeh, and A. Nazari, "Artificial neural network to predict the effect of heat treatments on vickers microhardness of low-carbon nb microalloyed steels," *Neural Computing and Applications*, vol. 22, no. 5, pp. 879–888, 2013.

[174] N. I. Chaudhary, S. Zubair, M. S. Aslam, M. A. Z. Raja, and J. T. Machado, "Design of momentum fractional lms for hammerstein nonlinear system identification with application to electrically stimulated muscle model," *The European Physical Journal Plus*, vol. 134, no. 8, p. 407, 2019.

[175] K. Hammar, T. Djamah, and M. Bettayeb, "Identification of fractional hammerstein system with application to a heating process," *Nonlinear Dynamics*, vol. 96, no. 4, pp. 2613–2626, 2019.

[176] M.-R. Rahmani and M. Farrokhi, "Identification of neuro-fractional hammerstein systems: a hybrid frequency-/time-domain approach," *Soft Computing*, vol. 22, no. 24, pp. 8097–8106, 2018.

[177] Z. Aslipour and A. Yazdizadeh, "Identification of wind turbine using fractional order dynamic neural network and optimization algorithm," *International Journal of Engineering*, vol. 33, no. 2, pp. 277–284, 2020.

[178] A. Atangana and S. Qureshi, "Modeling attractors of chaotic dynamical systems with fractal–fractional operators," *Chaos, Solitons & Fractals*, vol. 123, pp. 320–337, 2019.

[179] A. Atangana and M. A. Khan, "Validity of fractal derivative to capturing chaotic attractors," *Chaos, Solitons & Fractals*, vol. 126, pp. 50–59, 2019.

[180] C. Li, D. Qian, and Y. Chen, "On riemann-liouville and caputo derivatives," *Discrete Dynamics in Nature and Society*, vol. 2011, 2011.

[181] T. Chiranjeevi and R. K. Biswas, "Discrete-time fractional optimal control," *Mathematics*, vol. 5, no. 2, p. 25, 2017.

[182] B. Sobota and M. Guzan, "Virtualization of chua's circuit state space," pp. 127–163, 2019.

[183] R. Garrappa, E. Kaslik, and M. Popolizio, "Evaluation of fractional integrals and derivatives of elementary functions: Overview and tutorial," *Mathematics*, vol. 7, no. 5, p. 407, 2019.

[184] N. Aguila-Camacho, M. A. Duarte-Mermoud, and J. A. Gallegos, "Lyapunov functions for fractional order systems," *Communications in Nonlinear Science and Numerical Simulation*, vol. 19, no. 9, pp. 2951–2957, 2014.

[185] MathWorks, *Data of tanks system*, 2015 (accedido 4 Abril 2020). https://la.mathworks.com/help/ident/examples/two-tank-system-c-mex-file-modeling-of-time-continuous-siso-system.html.

[186] S. Dudul and A. Ghatol, "Identification of a typical cd player arm using a two-layer perceptron neural network model," vol. 2, pp. 1157–1162, 2003.

[187] W. Kirchgässner, O. Wallscheid, and J. Böcker, "Deep residual convolutional and recurrent neural networks for temperature estimation in permanent magnet synchronous motors," pp. 1439–1446, 2019.

[188] W. Kirchgässner, O. Wallscheid, and J. Böcker, "Empirical evaluation of exponentially weighted moving averages for simple linear thermal modeling of permanent magnet synchronous machines," pp. 318–323, 2019.

[189] J. F. Gómez-Aguilar, V. F. Morales-Delgado, M. A. Taneco-Hernández, D. Baleanu, R. F. Escobar-Jiménez, and M. M. Al Qurashi, "Analytical solutions of the electrical rlc circuit via liouville–caputo operators with local and non-local kernels," *Entropy*, vol. 18, no. 8, p. 402, 2016.

[190] C. Zúñiga-Aguilar, J. Gómez-Aguilar, R. Escobar-Jiménez, and H. Romero-Ugalde, "A novel method to solve variable-order fractional delay differential equations based in lagrange interpolations," *Chaos, Solitons & Fractals*, vol. 126, pp. 266–282, 2019.

[191] A. Coronel-Escamilla, J. Gómez-Aguilar, M. López-López, V. Alvarado-Martínez, and G. Guerrero-Ramírez, "Triple pendulum model involving fractional derivatives with different kernels," *Chaos, Solitons & Fractals*, vol. 91, pp. 248–261, 2016.

[192] A. Atangana and J. Gómez-Aguilar, "Decolonisation of fractional calculus rules: breaking commutativity and associativity to capture more natural phenomena," *The European Physical Journal Plus*, vol. 133, no. 4, p. 166, 2018.

[193] C. Li and C. Tao, "On the fractional adams method," *Computers & Mathematics with Applications*, vol. 58, no. 8, pp. 1573–1588, 2009.

[194] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[195] K. Madsen, H. B. Nielsen, and O. Tingleff, "Methods for non-linear least squares problems," 2004.