UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA

FACSISTEL

INGENIERÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN

TRABAJO DE INTEGRACIÓN CURRICULAR

DISEÑO DEL CONTROL DE UN SISTEMA ROTATORIO CON ESLABÓN FLEXIBLE MEDIANTE REDES NEURONALES ARTIFICIALES

Linier Ignacio Ramos Rambay

Dirigido por:
Ing. Carlos Alberto Saldaña Enderica, M.Sc.

La Libertad - 2025

DEDICATORIA

A mis padres, ejemplo de fortaleza, trabajo y entrega inquebrantable. Este logro académico no habría sido posible sin su apoyo constante, sus consejos sabios y la confianza que siempre depositaron en mí. Cada página de esta tesis refleja el esfuerzo compartido, las enseñanzas que me transmitieron y los valores que me han guiado en el camino de la vida. Con profundo respeto y gratitud, dedico este trabajo a ustedes, como un testimonio de mi reconocimiento y del fruto de su amor incondicional.

AGRADECIMIENTO

Deseo expresar mi más sincero agradecimiento a mis padres, cuyo apoyo constante, paciencia y sacrificio han sido fundamentales en cada etapa de mi formación académica y personal.

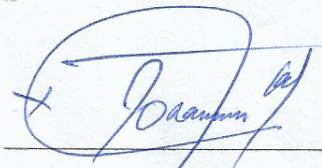
A todas las personas que, de una u otra manera, contribuyeron al desarrollo de esta investigación, les expreso mi reconocimiento y gratitud sincera.

APROBACIÓN DEL TUTOR

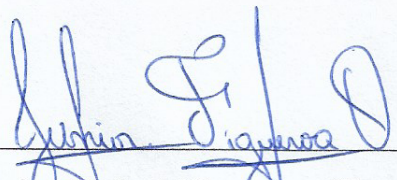
En mi calidad de tutor del trabajo de titulación denominado: **"Diseño del Control de un Sistema Rotatorio con Eslabón Flexible mediante Redes Neuronales Artificiales"**, presentado por el estudiante **Linier Ignacio Ramos Rambay**, de la carrera de **Ingeniería en Electrónica y Automatización** de la Universidad Estatal Península de Santa Elena, me permito declarar que luego de haber orientado, estudiado y revisado, lo apruebo en todas sus partes y autorizo al estudiante para que inicie los trámites legales correspondientes.

Ing. Carlos Alberto Saldaña Enderica, M.Sc.
Tutor

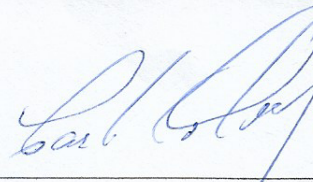
TRIBUNAL DE GRADO



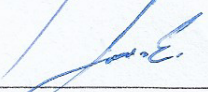
Ing. Ronald Humberto Rovira Jurado, Ph.D.
DIRECTOR DE CARRERA



Ing. Junior Rafael Figueroa Olmedo, Mgtr.
DOCENTE ESPECIALISTA



Ing. Carlos Alberto Saldaña Enderica, M.Sc.
TUTOR



Ing. Luis Enrique Chuquimarca Jiménez, Mgtr.
DOCENTE GUÍA UIC

Ing. Corina Gonzabay De La A, Mgtr.
SECRETARIA DEL TRIBUNAL

Resumen

Se ha propuesto en este proyecto de titulación el diseño y simulación de un sistema de control basado exclusivamente en Redes Neuronales Artificiales (RNA), a un sistema rotatorio con un eslabón flexible. La finalidad es lograr un seguimiento de la posición angular de la base θ, mientras se mitigan las oscilaciones de la punta del eslabón flexible α, sin necesidad de emplear controladores tradicionales ni estructuras híbridas. Para ello, se entrena un modelo de tipo Perceptrón Multicapa (MLP) mediante aprendizaje supervisado, con el fin de que aprenda directamente el modelo inverso de la planta y genere la señal de control adecuada a partir del historial de las variables del sistema.

Se procede a crear un modelo dinámico en espacio de estados, con datos de parámetros físicos reales. Este modelo se implementara en tiempo discreto utilizando la integración de Euler con paso fijo. Una vez obtenido el modelo, se simula para generar datos en lazo abierto que servirán para el entrenamiento de la red neuronal por modelo inverso. Todo este proceso se realiza en un ambiente de simulación como MATLAB/Simulink®, donde se usara el modelo entrenado como controlador en lazo cerrado, y se mide el rendimiento ante distintas señales de referencia.

Donde los resultados obtenidos demuestran que ciertas configuraciones del MLP logran un control lo mas cercano a la referencia, rápida estabilización y supresión de oscilaciones no deseadas.

Palabras clave: Control neuronal, eslabón flexible, modelo inverso, redes neuronales artificiales, seguimiento angular.

Abstract

In this degree project, the design and simulation of a control system based exclusively on Artificial Neural Networks (ANNs) is proposed for a rotary system with a flexible link. The objective is to achieve accurate tracking of the base angular position θ, while mitigating oscillations at the tip of the flexible link α, without relying on traditional controllers or hybrid structures. To this end, a Multilayer Perceptron (MLP) model is trained using supervised learning so that it can directly learn the inverse model of the plant and generate the appropriate control signal based on the history of the system variables.

A state-space dynamic model is developed using real physical parameter data. This model is implemented in discrete time using fixed-step Euler integration. Once the model is obtained, it is simulated to generate open-loop data that will be used for training the inverse-model neural network. The entire process is carried out in a simulation environment such as MATLAB/Simulink®, where the trained model is employed as a closed-loop controller, and its performance is evaluated under different reference signals.

The results obtained demonstrate that certain MLP configurations achieve control performance that closely follows the reference signal, with rapid stabilization and effective suppression of undesired oscillations.

Keywords: Neural control, flexible link, inverse model, artificial neural networks, angular tracking.

Índice

Índice de figuras

Índice de cuadros

1. Introducción

En la ingeniería de control, garantizar la estabilidad de los sistemas dinámicos es un desafío esencial para lograr que los procesos sean más seguros y autónomos, ya sea en entornos industriales o en aplicaciones robóticas. La incorporación de estrategias de control inteligentes resulta fundamental para el avance tecnológico y asegura que incluso los sistemas complejos puedan operar de manera eficiente bajo condiciones exigentes.

Los sistemas rotatorios con eslabón flexible (SREF) han atraído una atención creciente. Por el eslabón flexible aquel elemento estructural cuya elasticidad introduce grados de libertad adicionales respecto a un modelo rígido, generando vibraciones que comprometen la precisión y el desempeño dinámico del sistema. Estas oscilaciones no deseadas afectan el rendimiento, la exactitud del posicionamiento y la integridad estructural, constituyendo uno de los principales retos en el control de dichos sistemas. Este fenómeno se basa en los fundamentos clásicos y la teoría de vigas, donde se postulan las hipótesis de Euler–Bernoulli en las que se apoyan los desplazamientos y tensiones del modelo [1]. Además, en términos contemporáneos, investigaciones recientes han documentado cómo la vibración continúa siendo un obstáculo persistente en el diseño de control para SREF [2, 3].

Se han propuesto varias tácticas para afrontar estos desafíos. En [2], se empleó un controlador LQR optimizado por medio de algoritmos genéticos con el objetivo de reducir las oscilaciones. En [3], por su parte, se implementó un esquema híbrido que combinaba LQR con aprendizaje por refuerzo profundo (RL) para optimizar el seguimiento de trayectorias en situaciones de flexibilidad dinámica. Estas propuestas demuestran que se están buscando soluciones híbridas que fusionen técnicas tradicionales de control con métodos inteligentes.

En este trabajo se desarrolla un esquema puramente basado en RNA, sin dependencias en controladores clásicos auxiliares. Estas redes ofrecen la capacidad de aprender dinámicas complejas no lineales a partir de datos, lo que permite capturar y compensar deformaciones dinámicas sin estructuras de control adicionales. Además, la técnica de reducción modal, fundamental en sistemas de vibraciones [4], respalda la simplificación del sistema a modos dominantes para facilitar el entrenamiento y la implementación práctica del controlador neural. Se espera que este planteamiento logre ventajas en estabilidad y rendimiento del SREF.

1.1. Justificación

El control de sistemas rotatorios con eslabones flexibles constituye un reto importante y vigente en la ingeniería de control debido a su presencia en aplicaciones críticas como robótica ligera, manipuladores espaciales, estructuras desplegables y sistemas aeroespaciales. En estos entornos, la reducción de peso y el ahorro energético son prioritarios, pero implican un aumento de la flexibilidad estructural, lo que a su vez genera oscilaciones, acoplamientos no lineales y variaciones dinámicas que dificultan mantener precisión y estabilidad [5]. Resolver este problema es esencial para garantizar un funcionamiento seguro y eficiente en escenarios donde la confiabilidad es determinante.

A pesar de que los controladores clásicos, como PID o LQR, han sido ampliamente utilizados, su desempeño se degrada ante incertidumbres, dinámicas no modeladas o cambios en las condiciones de operación [6, 7]. Esto evidencia la necesidad de estrategias de control capaces de adaptarse a comportamientos no lineales complejos y de mantener un rendimiento estable incluso bajo perturbaciones o variaciones estructurales.

En este contexto, las RNA surgen como una alternativa prometedora, ya que pueden aprender relaciones no lineales a partir de datos sin requerir un modelo matemático exacto, adaptándose a cambios en la dinámica y compensando efectos no modelados [5, 8]. El presente trabajo propone un controlador basado exclusivamente en un Perceptrón Multicapa (MLP) entrenado para aprender el modelo inverso del sistema, eliminando la necesidad de estructuras híbridas o modelos lineales intermedios y ofreciendo un enfoque versátil y generalizable.

Los resultados logrados, corroborados en un contexto de simulación realista que incluye condiciones y parámetros físicos diversos, suponen una contribución importante para el progreso de técnicas de control inteligente empleadas en sistemas no lineales. Esto refuerza la relación entre teoría, simulación y aplicaciones prácticas.

1.2. Panorama actual

El control de sistemas rotatorios con eslabones flexibles sigue siendo un reto debido a las vibraciones y la incertidumbre en el modelado. En la última década se han investigado numerosas técnicas de control para abordarlo, desde métodos clásicos realimentados (PID no lineal con compensación de gravedad, LQR) hasta enfoques avanzados como control robusto, modos deslizantes (SMC), control adaptativo y esquemas backstepping [9].

No obstante, los métodos rígidos tradicionales suelen fallar al aplicarse en enlaces flexibles por la dinámica distribuida y no lineal, por ello la investi-

gación reciente ha recurrido cada vez más a las RNA por su capacidad para aproximar dinámicas complejas [10].

Estas redes se emplean de forma directa (por ejemplo, aprendiendo un modelo inverso del sistema para generar la acción de control) o indirecta (por ejemplo, ajustando ganancias de controladores convencionales de manera adaptativa) [10].

En la literatura reciente abundan los ejemplos: Gao et al. utilizaron RNA para la supresión de vibraciones y el seguimiento de trayectoria en un manipulador de dos eslabones flexibles [11]. También se han empleado redes neuronales recurrentes (RNN) para modelar las dinámicas directa e inversa de manipuladores flexibles, mejorando el control mediante lazos anticipativos de compensación [10].

Incluso desde décadas atrás se propuso el uso de modelos inversos neuronales (basados en perceptrones multicapa) para control de brazos flexibles – por ejemplo, Su y Khorasani (2002) presentaron un controlador inverso con dos redes MLP en un brazo flexible de una articulación, logrando reducir error de punta frente a controladores PD convencionales [12].

Un MLP entrenado puede funcionar como modelo inverso dinámico, es decir, estima la señal de control necesaria para lograr una trayectoria deseada del sistema, abordando así la no linealidad y elasticidad difícil de compensar con modelos analíticos sencillos. La estrategia de control inverso neuronal generalmente involucra entrenar la red de forma supervisada con datos del sistema (entradas de control y salidas medidas) para que la red aprenda la dinámica inversa. Por ejemplo, en un manipulador flexible de dos enlaces, Jamali et al. separaron la identificación en submodelos SISO y usaron un MLP (estructura NARX) entrenado con datos experimentales para modelar la dinámica, el MLP logró aproximar mejor el comportamiento del sistema que una red recurrente Elman, obteniendo menor error cuadrático y resultados no sesgados [13].

Esto demuestra la capacidad de las MLP para capturar la relación no lineal entre la acción de control y la respuesta del enlace flexible, lo cual es clave para un controlador inverso preciso. La ventaja de este aprendizaje supervisado es que no requiere un modelo físico exacto, sino que la red "aprende" directamente del comportamiento medido o simulado del sistema, incorporando incluso efectos difíciles de modelar. Varios trabajos recientes enfatizan recopilar datos de alta fidelidad (ya sea de experimentos o de simulaciones detalladas) para entrenar estas redes neuronales y luego validar sus desempeños en entornos de simulación avanzados antes de implementarlos en la práctica [14].

Además de los controladores puramente neuronales, enfoques híbridos que combinan RNA con técnicas de control robusto o no lineal han mostrado

resultados prometedores. Un ejemplo destacado es la combinación de Redes Neuronales con Modos Deslizantes (SMC). Lima et al. (2021) propusieron un esquema inteligente donde un controlador SMC para sistemas subactuados incorpora una red neuronal que compensa incertezas del modelo en tiempo real [8].El resultado es un controlador capaz de adaptarse continuamente a las dinámicas no modeladas y atenuar las oscilaciones flexibles.

Otro enfoque híbrido reciente combina un modelo inverso neuronal con control robusto basado en perturbaciones separadas. Por ejemplo, Li et al. (2024) desarrollaron un algoritmo inverso adaptativo para un manipulador espacial de articulación flexible donde el sistema se divide en dos subsistemas (rápido y lento) mediante teoría de perturbaciones singulares [15]. En el lazo lento, un controlador inverso genera la señal principal mientras que los términos inciertos se compensan, una red neuronal de base radial (RBF) se ajusta en línea para aproximar las dinámicas inciertas, y un término robusto adicional cancela el error de aproximación [15]. Simulaciones mostraron que este esquema logra suprimir las vibraciones de alta frecuencia causadas por la flexibilidad y mantener un seguimiento estable de la trayectoria aun con perturbaciones externas [15].

En la misma línea, se han integrado redes neuronales con Control Predictivo Modelo (MPC) para manipuladores flexibles. Un caso es el de Zhang et al. (2021), quienes propusieron un NMPC no lineal guiado por red neuronal recurrente: primero se entrenó una RNN de tres capas con activaciones ReLU para aproximar la dinámica del brazo de un grado de libertad (capturando su comportamiento no lineal), luego esa RNN se utilizó dentro de un MPC como modelo de predicción, optimizándose las acciones de control con un algoritmo evolutivo [16]. Este controlador híbrido (RNN + MPC) logró mejoras notables en simulación, cumpliendo los requisitos de precisión en el posicionamiento al mismo tiempo que redujo los sobrepicos y suprimió la vibración residual del enlace flexible [16].

Este panorama actual muestra una evolución clara hacia controladores inteligentes basados en redes neuronales para sistemas con eslabones flexibles, a menudo complementados con estrategias robustas o predictivas. Las MLP, en particular, han emergido como herramientas eficaces para construir modelos inversos del sistema, facilitando un control predictivo que contrarresta las dinámicas elásticas y no lineales. La viabilidad de estos enfoques está respaldada por numerosos trabajos: se reportan mejoras significativas en precisión de seguimiento, supresión de vibraciones y adaptabilidad frente a incertidumbres al incorporar modelos neuronales entrenados en lazo de control. Los avances revisados sugieren que un controlador RNA inverso basado en MLP, correctamente entrenado y probado en entornos como MATLAB/Simulink®, puede abordar eficazmente los desafíos de la flexibilidad estructural, sirviendo

como aporte novedoso y relevante en el control de manipuladores flexibles.

1.3. Objetivos

1.3.1. Objetivo General:

Desarrollar y evaluar un sistema de control utilizando Redes Neuronales Artificiales para un sistema rotatorio con eslabón flexible, con el propósito de mitigar las perturbaciones y minimizar las oscilaciones del sistema.

1.3.2. Objetivos Específicos:

- Realizar el modelado matemático del sistema rotatorio con eslabón flexible, con el fin de obtener una representación precisa para el diseño y prueba del sistema de control mediante redes neuronales artificiales.
- Reproducir el modelo matemático del Sistema Rotatorio con Eslabón Flexible en MATLAB/Simulink, para poder representar con exactitud las oscilaciones del eslabón y la dinámica de rotación.
- Diseñar una arquitectura de control basada en Redes Neuronales Artificiales (RNA) en MATLAB/Simulink, con el fin de mitigar las oscilaciones y controlar la posición angular del sistema rotatorio con eslabón flexible.
- Llevar a cabo pruebas de simulación con el fin de evaluar la estabilidad y el rendimiento del controlador de RNA diseñado, midiendo cuánto disminuyen las oscilaciones, cuál es el margen de error en estado estacionario, cuánto tiempo tarda en establecerse y cómo se comporta ante perturbaciones en el sistema simulado.

1.4. Fundamentos teóricos

Se abarca las bases para comprender y desarrollar el modelado de sistemas rotatorios con eslabon flexible, asi como sus estrategias de control.Para el modelado matematico se acoge varios principios que describen la dinamica de un enlace flexible y a la vez el controlador que mejora el comportamiento del sistema.

1.4.1. Eslabón flexible

En los sistemas con eslabón flexible, los enlaces no rígidos, desempeñan la transmisión de fuerzas y movimientos en distintos tramos del eslabón.

Donde esta flexibilidad resulta útil en situaciones que necesiten una respuesta mas suave, una reducción de oscilaciones y una mejor adaptabilidad frente a perturbaciones. Este tipo de comportamiento ha sido estudiado en trabajos recientes como el de Araújo [17], donde se analiza un brazo robótico de dos eslabones flexible. En dicho estudio, para modelar este tipo de sistemas, se recurre al uso de teorías de elasticidad y dinámica estructural, lo cual permitirá modelar las deformaciones del enlace a través del tiempo.

Para tener un análisis adecuado de este tipo de sistemas dinámicos, se considera tanto las fuerzas internas como externas que actúan sobre el. Para describir el comportamiento de las deformaciones y vibraciones en la parte flexible, se utiliza la teoría del haz de Euler-Bernoulli. Un aspecto crucial son las vibraciones que se producen en el eslabón flexible, ya que estas afectan la estabilidad, precisión y eficiencia del sistema. Las ecuaciones de movimiento del eslabón flexible se las puede describir mediante los principios como el de Hamilton o el Lagrange, tal como se expone en [18].

Otro metodo que captura la dinamica de este tipo de sistemas, es el de modos asumidos señalado por Lee y Alandoli [19], el cual representa la flexibilidad del eslabón de forma sencilla y sin comprometer el modelo. Esta técnica es adecuada para sistemas rotatorios con eslabón flexible, donde se debe tener en claro la relación entre el movimiento rotacional y las oscilaciones del eslabón para su correcto acoplamiento.

1.4.2. Modelado del sistema rotatorio con eslabón flexible

El modelado matemático de sistemas dinámicos, siembra las bases para entender el funcionamiento y diseñar estrategias de control adecuadas. En el caso de los sistemas rotacionales con eslabón flexible, se debe de representar el funcionamiento del eje rotatorio como las deformaciones del enlace, permitiendo reducir fenómenos como vibraciones o inestabilidad [20].

Tal como se discute en [21], el control de vibraciones en sistemas con estructuras livianas y flexibles es un reto común en ingeniería, donde el uso de modelos dinámicos precisos permite evaluar diferentes estrategias de mitigación, incluso bajo condiciones con restricciones físicas.

Una etapa clave en la formulación del modelo dinámico de sistemas rotatorios con enlaces flexibles (SREF) es el uso del principio de Hamilton, el cual permite establecer un equilibrio entre la energía cinética, la energía potencial y el trabajo de las fuerzas no conservativas para derivar las ecuaciones de movimiento del sistema. Estas formulaciones, son aplicadas y validadas en

este estudio , será simulada en un entorno como MATLAB/Simulink® para analizar el comportamiento estructural bajo diferentes condiciones [23, 22].

1.4.3. Hipótesis de Euler–Bernoulli

El sistema de estudio corresponde a un eslabón flexible unido a una base rotatoria. Para describir su comportamiento se adopta la teoría clásica de vigas de Euler–Bernoulli, la cual establece que las secciones transversales permanecen planas y perpendiculares al eje neutro durante la deformación [24]. Esta hipótesis es válida bajo la condición de pequeñas deformaciones, lo cual resulta adecuado para el presente caso [25].

Este mismo enfoque ha sido adoptado en trabajos recientes sobre el modelado del sistema rotatorio flexible, donde la teoría de Euler–Bernoulli se emplea como base para describir la dinámica de flexión [2].

Sea x la coordenada longitudinal a lo largo del eje neutro del eslabón, z la coordenada transversal medida desde el eje neutro, y $\alpha(x,t)$ la deflexión transversal de la viga en la dirección z. Los desplazamientos de un punto genérico de la viga se describen entonces como:

$$u_x(x,z,t) = -z\,\frac{\partial \alpha(x,t)}{\partial x}, \qquad u_z(x,t) = \alpha(x,t), \tag{1}$$

donde u_x representa el desplazamiento longitudinal y u_z el desplazamiento transversal. A partir de este campo de desplazamientos se determina la deformación longitudinal mediante:

$$\varepsilon_{xx}(x,z,t) = \frac{\partial u_x}{\partial x} = -z\,\frac{\partial^2 \alpha(x,t)}{\partial x^2}, \tag{2}$$

lo que muestra que la deformación es proporcional a la curvatura $\alpha''(x,t)$ y depende linealmente de la posición transversal z [26].

Asumiendo que el material es lineal, elástico, homogéneo e isótropo, la relación constitutiva de Hooke conduce a la tensión normal:

$$\sigma_{xx}(x,z,t) = E\,\varepsilon_{xx}(x,z,t) = -E\,z\,\alpha''(x,t), \tag{3}$$

donde E es el módulo de Young del material. Esta tensión es la responsable de los esfuerzos internos que gobiernan la dinámica de flexión de la viga [27].

Para introducir la inercia del sistema se considera la densidad de masa del material ρ (kg/m^3) y el área de la sección transversal A (m^2). El producto:

$$\mu = \rho A, \tag{4}$$

define la densidad lineal de masa (kg/m), la cual se emplea en las integrales a lo largo de la longitud L del eslabón. Con esta definición se evita ambigüedad entre la densidad volumétrica del material y la densidad distribuida de la viga [28].

1.4.4. Formulación energética del sistema

La dinámica del sistema se obtiene a partir de un enfoque energético que considera las contribuciones tanto de la base rígida como del eslabón flexible. El análisis se apoya en la formulación clásica de energía cinética y energía potencial, las cuales se emplearán en el Principio de Hamilton para derivar las ecuaciones de movimiento [29, 30].

La energía cinética total T se compone de dos partes: la asociada a la base rígida rotatoria y la correspondiente al eslabón flexible. Para la base, de momento de inercia J_{base} respecto a su eje de rotación, se tiene:

$$T_{base} = \tfrac{1}{2} J_{base}\, \dot{\theta}^2, \tag{5}$$

donde $\theta(t)$ es el ángulo de rotación. En el caso del eslabón flexible, el desplazamiento transversal absoluto de un punto a distancia x del empotramiento se expresa como:

$$y(x,t) = \alpha(x,t) + x\,\theta(t), \tag{6}$$

siendo $\alpha(x,t)$ la deflexión relativa y $x\theta(t)$ el desplazamiento debido a la rotación de la base. Su velocidad transversal se deduce como:

$$\dot{y}(x,t) = \dot{\alpha}(x,t) + x\,\dot{\theta}(t). \tag{7}$$

La energía cinética distribuida del eslabón, con densidad lineal μ, resulta entonces:

$$T_{flex} = \tfrac{1}{2} \int_0^L \mu \left(\dot{\alpha}(x,t) + x\,\dot{\theta}(t)\right)^2 dx. \tag{8}$$

La energía cinética total del sistema se expresa como la suma de ambas contribuciones:

$$T = T_{base} + T_{flex} = \tfrac{1}{2} J_{base}\, \dot{\theta}^2 + \tfrac{1}{2} \int_0^L \mu \left(\dot{\alpha}(x,t) + x\,\dot{\theta}(t)\right)^2 dx, \tag{9}$$

lo cual es consistente con la formulación estándar de sistemas rotatorios con elementos flexibles [31].

Por otra parte, la energía potencial elástica proviene de la deformación por flexión del eslabón. De acuerdo con la teoría de Euler–Bernoulli, se escribe como:

$$U = \tfrac{1}{2} \int_0^L EI \left(\frac{\partial^2 \alpha(x,t)}{\partial x^2} \right)^2 dx, \tag{10}$$

donde EI es la rigidez a flexión, producto del módulo de Young E por el segundo momento de área I de la sección transversal [24, 29].

Finalmente, el Principio de Hamilton se formula a partir del funcional:

$$\mathcal{H} = \int_{t_1}^{t_2} \big(T - U + W \big)\, dt, \tag{11}$$

donde W representa el trabajo virtual de las fuerzas externas. En este caso, la única fuerza generalizada corresponde al par aplicado por el motor en la base, $\tau(t)$, de manera que la variación del trabajo se expresa como:

$$\delta W = \tau(t)\, \delta\theta. \tag{12}$$

Estas formulaciones energéticas constituyen la base para la aplicación del Principio de Hamilton, que se utilizará para derivar las ecuaciones dinámicas del movimiento acoplado entre la base y el eslabón flexible [30, 32].

De manera similar, en [2] se plantea la obtención de las ecuaciones dinámicas del sistema a partir del principio variacional de Hamilton, confirmando la aplicabilidad de este método al caso del eslabón flexible.

1.4.5. Ecuaciones dinámicas mediante Hamilton

Con las expresiones de energía cinética, energía potencial y trabajo virtual obtenidas anteriormente, se aplica el Principio de Hamilton para deducir las ecuaciones de movimiento del sistema. Dicho principio establece que, para cualquier variación admisible de los desplazamientos $\delta\alpha(x,t)$ y $\delta\theta(t)$ con extremos fijos en los instantes t_1 y t_2, se cumple [29, 30]:

$$\delta \int_{t_1}^{t_2} \big(T - U + W \big)\, dt = 0. \tag{13}$$

Sustituyendo las expresiones de las energías y del trabajo virtual, se obtiene:

$$\delta \int_{t_1}^{t_2} \left[\tfrac{1}{2} J_{\text{base}} \dot{\theta}^2 + \tfrac{1}{2} \int_0^L \mu \left(\dot{\alpha}(x,t) + x\dot{\theta}(t) \right)^2 dx \right. \\ \left. - \tfrac{1}{2} \int_0^L EI \left(\alpha''(x,t) \right)^2 dx + \tau(t)\, \theta(t) \right] dt \;=\; 0. \tag{14}$$

Aplicando cálculo variacional a las coordenadas generalizadas se obtiene, en primer lugar, la ecuación asociada a la rotación de la base:

$$J_{base} \ddot{\theta}(t) + \mu \int_0^L \Big(x^2 \ddot{\theta}(t) + x \ddot{\alpha}(x,t) \Big) dx = \tau(t) \tag{15}$$

que refleja cómo la aceleración angular de la base está acoplada a la dinámica transversal de la viga [31]. Para la coordenada flexible $\alpha(x,t)$ se obtiene la ecuación en derivadas parciales:

$$\mu\, \ddot{\alpha}(x,t) + \mu x\, \ddot{\theta}(t) + EI\, \alpha''''(x,t) = 0, \qquad x \in (0,L), \tag{16}$$

la cual describe la propagación de ondas de flexión bajo el efecto de la aceleración de la base [32].

Las condiciones de frontera del eslabón flexible se derivan de la configuración empotrado–libre: en $x = 0$ deben anularse el desplazamiento y la pendiente:

$$\alpha(0,t) = 0, \qquad \alpha'(0,t) = 0, \tag{17}$$

mientras que en el extremo libre $x = L$ deben anularse el momento flector y el esfuerzo cortante, lo que conduce a:

$$\alpha''(L,t) = 0, \qquad \alpha'''(L,t) = 0. \tag{18}$$

Estas condiciones de frontera corresponden al caso clásico de una viga empotrada en un extremo y libre en el otro [24, 33].

El modelo dinámico continuo queda representado por la ecuación en derivadas parciales (16), acoplada con la ecuación de la base (15) y sujeta a las condiciones de frontera (17)–(18). Un procedimiento equivalente se encuentra en [2], donde se obtienen las ecuaciones acopladas de la base rotatoria y el eslabón flexible, constituyendo el punto de partida para la reducción modal [29, 32].

1.4.6. Reducción modal del sistema flexible

Las ecuaciones dinámicas del sistema continuo, obtenidas en el apartado anterior, describen con precisión el acoplamiento entre la rotación de la

base y la flexión del eslabón. Sin embargo, su forma en derivadas parciales dificulta el análisis directo y la implementación en simulaciones de control. Por esta razón, se recurre a una proyección modal, en la que la deflexión transversal $\alpha(x,t)$ se aproxima como combinación de un número finito de funciones propias de vibración [29, 32].

Sea $\{\phi_i(x)\}_{i=1}^{\infty}$ el conjunto de modos de flexión de una viga de Euler–Bernoulli empotrada en $x=0$ y libre en $x=L$. Estas funciones satisfacen el problema propio:

$$EI\,\phi_i''''(x) = \mu\,\omega_i^2\,\phi_i(x), \qquad \phi_i(0) = \phi_i'(0) = 0, \qquad \phi_i''(L) = \phi_i'''(L) = 0, \tag{19}$$

donde ω_i es la frecuencia natural del i-ésimo modo [24, 33]. La deflexión transversal se aproxima mediante una truncación modal:

$$\alpha(x,t) \;\approx\; \sum_{i=1}^{N} q_i(t)\,\phi_i(x), \tag{20}$$

donde $q_i(t)$ son las coordenadas modales generalizadas y N es el número de modos retenidos en la aproximación.

Las funciones propias cumplen relaciones de ortogonalidad tanto en masa como en rigidez:

$$\int_0^L \mu\,\phi_i(x)\,\phi_j(x)\,dx = m_i\,\delta_{ij}, \qquad \int_0^L EI\,\phi_i''(x)\,\phi_j''(x)\,dx = k_i\,\delta_{ij}, \tag{21}$$

donde m_i y k_i representan, respectivamente, la masa y la rigidez modales, y δ_{ij} es la delta de Kronecker. La frecuencia natural del modo se relaciona mediante $\omega_i^2 = k_i/m_i$. El amortiguamiento estructural puede incorporarse de forma proporcional, con un coeficiente modal $c_i = 2\zeta_i\omega_i m_i$, siendo ζ_i el factor de amortiguamiento adimensional [30]. Además, el término de acoplamiento inercial con la rotación de la base se cuantifica mediante los coeficientes:

$$\gamma_i = \mu \int_0^L x\,\phi_i(x)\,dx \tag{22}$$

que expresan la manera en que la aceleración angular de la base excita cada modo flexible [31].

Al proyectar la ecuación en derivadas parciales (16) sobre cada función propia y usar las relaciones de ortogonalidad, se obtiene, para cada coordenada modal $q_j(t)$, la ecuación diferencial ordinaria:

$$m_j\,\ddot{q}_j(t) + c_j\,\dot{q}_j(t) + k_j\,q_j(t) + \gamma_j\,\ddot{\theta}(t) = 0. \tag{23}$$

Estas N ecuaciones describen la dinámica de los modos retenidos, todos ellos acoplados únicamente a través de la aceleración de la base $\ddot{\theta}(t)$ [29]. Por su parte, la ecuación de la base toma la forma:

$$\left(J_{base} + I_a\right)\ddot{\theta}(t) + \sum_{i=1}^{N}\gamma_i\,\ddot{q}_i(t) = \tau(t), \qquad I_a = \mu\int_0^L x^2\,dx = \mu\,\frac{L^3}{3} \tag{24}$$

donde I_a representa la contribución inercial asociada a la distribución de masa del eslabón respecto al empotramiento.

En forma matricial, el sistema acoplado de N modos puede expresarse como:

$$\begin{bmatrix}\mathbf{M}_q & \boldsymbol{\gamma}\\ \boldsymbol{\gamma}^\top & J_{base}+I_a\end{bmatrix}\begin{bmatrix}\ddot{\mathbf{q}}(t)\\ \ddot{\theta}(t)\end{bmatrix} + \begin{bmatrix}\mathbf{C}_q & \mathbf{0}\\ \mathbf{0}^\top & 0\end{bmatrix}\begin{bmatrix}\dot{\mathbf{q}}(t)\\ \dot{\theta}(t)\end{bmatrix} + \begin{bmatrix}\mathbf{K}_q & \mathbf{0}\\ \mathbf{0}^\top & 0\end{bmatrix}\begin{bmatrix}\mathbf{q}(t)\\ \theta(t)\end{bmatrix} = \begin{bmatrix}\mathbf{0}\\ \tau(t)\end{bmatrix}, \tag{25}$$

donde $\mathbf{M}_q = \mathrm{diag}(m_1,\ldots,m_N)$, $\mathbf{C}_q = \mathrm{diag}(c_1,\ldots,c_N)$, $\mathbf{K}_q = \mathrm{diag}(k_1,\ldots,k_N)$ y $\boldsymbol{\gamma} = [\gamma_1,\ldots,\gamma_N]^\top$.

Este modelo modal truncado permite describir el sistema con un número reducido de coordenadas y constituye una representación más manejable para aplicaciones de control [32]. Dado que en la práctica el primer modo de flexión concentra la mayor parte de la energía del sistema, resulta habitual aproximar la dinámica completa reteniendo únicamente dicho modo. A continuación se adopta esta simplificación, dando lugar a un modelo equivalente de dos grados de libertad (2-DOF) que conserva el acoplamiento esencial entre la rotación de la base y la flexión del eslabón [29, 34].

1.4.7. Modelo simplificado de dos grados de libertad

Aunque el modelo modal presentado describe con precisión la dinámica de la viga flexible, en la práctica resulta conveniente disponer de un modelo reducido que conserve la interacción esencial entre la rotación de la base y la flexión del eslabón. En la mayoría de aplicaciones, el primer modo de vibración concentra la mayor parte de la energía dinámica del sistema, por lo que una aproximación razonable consiste en retener únicamente dicho modo [29, 34].

Para simplificar la notación y facilitar la interpretación física, se introduce una nueva variable generalizada $\alpha(t)$, definida como una coordenada modal

angular equivalente al primer modo de flexión. Esta variable se expresa en radianes y debe entenderse como una medida relativa del ángulo de deflexión del eslabón respecto a la base. Es importante señalar que, a partir de esta sección, $\alpha(t)$ se emplea con este nuevo significado angular, distinto de $\alpha(x,t)$ usado en las secciones previas para denotar la deflexión transversal en coordenadas espaciales.

De este modo, el sistema queda caracterizado por dos grados de libertad:

$$\theta(t) : \text{ángulo de la base,}$$

$$\alpha(t) : \text{ángulo relativo equivalente del eslabón flexible.}$$

Sea J_{base} el momento de inercia de la base respecto a su eje de rotación y J_{link} el momento de inercia equivalente del eslabón flexible. Si además se considera una masa puntual m_L en el extremo libre a una distancia L, la inercia total del enlace respecto a la base puede escribirse como:

$$J_T = J_{link} + m_L L^2. \tag{26}$$

La representación de la inercia equivalente en este formato se encuentra ampliamente utilizada en modelos reducidos de manipuladores flexibles [35].

La energía cinética del sistema se compone de la contribución de la base y la del eslabón flexible. Para la base se tiene:

$$T_{base} = \tfrac{1}{2} J_{base}\, \dot{\theta}^2, \tag{27}$$

mientras que para el enlace, cuya velocidad angular relativa es $(\dot{\theta} - \dot{\alpha})$, se obtiene:

$$T_{link} = \tfrac{1}{2} J_T\, (\dot{\theta} - \dot{\alpha})^2. \tag{28}$$

Así, la energía cinética total resulta:

$$T = \tfrac{1}{2} J_{base}\, \dot{\theta}^2 + \tfrac{1}{2} J_T\, (\dot{\theta} - \dot{\alpha})^2. \tag{29}$$

La energía potencial asociada a la flexión se modela mediante una rigidez torsional equivalente K_{stiff} que actúa sobre el ángulo relativo:

$$U = \tfrac{1}{2} K_{stiff}\, \alpha^2. \tag{30}$$

El uso de un resorte torsional equivalente como aproximación a la energía de flexión es común en la literatura para modelos simplificados de vigas [36].

Aplicando la formulación de Lagrange a las coordenadas generalizadas $q_i \in \{\theta, \alpha\}$, y considerando que la única fuerza externa corresponde al par aplicado $\tau(t)$ sobre la base, se obtienen las ecuaciones de movimiento acopladas [30]:

$$(J_{base} + J_T)\,\ddot{\theta}(t) - J_T\,\ddot{\alpha}(t) = \tau(t), \tag{31}$$

$$J_T\,(\ddot{\alpha}(t) - \ddot{\theta}(t)) + K_{stiff}\,\alpha(t) = 0. \tag{32}$$

En forma matricial, el modelo reducido de dos grados de libertad puede escribirse como:

$$\begin{bmatrix} J_{base} + J_T & -J_T \\ -J_T & J_T \end{bmatrix} \begin{bmatrix} \ddot{\theta}(t) \\ \ddot{\alpha}(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & K_{stiff} \end{bmatrix} \begin{bmatrix} \theta(t) \\ \alpha(t) \end{bmatrix} = \begin{bmatrix} \tau(t) \\ 0 \end{bmatrix}. \tag{33}$$

Este modelo conserva el acoplamiento fundamental entre la dinámica rígida de la base y la flexión del eslabón, constituyendo una representación simplificada que mantiene la fidelidad en el rango de frecuencias de interés [34]. A partir de este punto, el par $\tau(t)$ podrá expresarse en función del voltaje aplicado al motor eléctrico.

1.4.8. Modelo del motor dc y acoplamiento mecánico en el sistema

Para completar el modelo del sistema rotatorio con eslabón flexible es necesario incluir la dinámica del actuador. Se considera un motor de corriente continua (dc) acoplado a la base mediante una caja reductora, de modo que el voltaje aplicado en la armadura se traduce en un par sobre la coordenada θ [37].

El circuito eléctrico del motor se modela aplicando la ley de Kirchhoff de voltajes, lo que conduce a:

$$V_a(t) = R\,i(t) + L\,\frac{di(t)}{dt} + e_b(t), \tag{34}$$

donde $V_a(t)$ es el voltaje de armadura, R la resistencia, L la inductancia, $i(t)$ la corriente y $e_b(t)$ la fuerza contraelectromotriz. Esta última se relaciona con la velocidad angular del rotor mediante:

$$e_b(t) = K_b\,\dot{\theta}_m(t), \tag{35}$$

siendo K_b la constante de back-EMF y $\dot{\theta}_m(t)$ la velocidad angular del eje del motor [38]. En la práctica, la inductancia L suele ser pequeña en comparación con los términos resistivos, por lo que puede despreciarse. En tal caso, la ecuación eléctrica se simplifica a:

$$V_a(t) \approx R\,i(t) + K_b\,\dot{\theta}_m(t). \tag{36}$$

El torque desarrollado por el motor es proporcional a la corriente de armadura:

$$\tau_m(t) = K_t\, i(t), \tag{37}$$

donde K_t es la constante de torque. Esta relación lineal entre corriente y par es ampliamente documentada en modelos de motores DC [37, 38]. Si el motor acciona la base mediante una caja reductora de relación K_g, el par transmitido y la velocidad relativa se expresan como:

$$\tau(t) = K_g\, \tau_m(t) = K_t K_g\, i(t), \tag{38}$$

$$\dot{\theta}_m(t) = K_g\, \dot{\theta}(t), \tag{39}$$

donde $\theta(t)$ es el ángulo de la base. Sustituyendo la corriente a partir de (36), se obtiene:

$$i(t) = \frac{V_a(t) - K_b K_g \dot{\theta}(t)}{R}. \tag{40}$$

Finalmente, el par transmitido al eje de la base en función del voltaje aplicado resulta:

$$\tau(t) = \frac{K_t K_g}{R}\, V_a(t) - \frac{K_b K_t K_g^2}{R}\, \dot{\theta}(t). \tag{41}$$

Este par se acopla directamente a la ecuación de movimiento de la base del modelo 2-DOF. Sustituyendo en (31) se obtiene:

$$(J_{base} + J_T)\, \ddot{\theta}(t) - J_T\, \ddot{\alpha}(t) = \frac{K_t K_g}{R}\, V_a(t) - \frac{K_b K_t K_g^2}{R}\, \dot{\theta}(t), \tag{42}$$

mientras que la ecuación para la coordenada flexible se mantiene como:

$$J_T\, (\ddot{\alpha}(t) - \ddot{\theta}(t)) + K_{stiff}\, \alpha(t) = 0. \tag{43}$$

En conjunto, estas dos ecuaciones describen la dinámica combinada del sistema mecánico y del motor DC con reductora [37, 38]. El siguiente paso consiste en reescribir este modelo en la forma de espacio de estados.

1.4.9. Modelo en espacio de estados del sistema

Con las ecuaciones acopladas obtenidas, es posible construir el modelo en espacio de estados, una representación particularmente útil para el análisis

dinámico y el diseño de estrategias de control [30, 37]. Para ello se definen como variables de estado las posiciones angulares y sus velocidades asociadas:

$$x_1 = \theta(t), \qquad x_2 = \alpha(t), \qquad x_3 = \dot{\theta}(t), \qquad x_4 = \dot{\alpha}(t). \tag{44}$$

De esta forma, el vector de estado se expresa como:

$$\mathbf{x}(t) = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \theta(t) \\ \alpha(t) \\ \dot{\theta}(t) \\ \dot{\alpha}(t) \end{bmatrix}. \tag{45}$$

La entrada del sistema corresponde al voltaje aplicado en la armadura del motor:

$$u(t) = V_a(t), \tag{46}$$

y las salidas de interés son el ángulo de la base y el ángulo relativo del enlace flexible:

$$\mathbf{y}(t) = \begin{bmatrix} \theta(t) \\ \alpha(t) \end{bmatrix}. \tag{47}$$

Recordando las ecuaciones de movimiento con efecto del motor:

$$(J_{base} + J_T)\,\ddot{\theta}(t) - J_T\,\ddot{\alpha}(t) = \frac{K_t K_g}{R}\,V_a(t) - \frac{K_b K_t K_g^2}{R}\,\dot{\theta}(t), \tag{48}$$

$$J_T(\ddot{\alpha}(t) - \ddot{\theta}(t)) + K_{stiff}\,\alpha(t) = 0, \tag{49}$$

se despejan $\ddot{\theta}(t)$ y $\ddot{\alpha}(t)$ en función de los estados y la entrada. De este modo, el sistema adquiere la forma estándar:

$$\dot{\mathbf{x}}(t) = A\,\mathbf{x}(t) + B\,u(t), \qquad \mathbf{y}(t) = C\,\mathbf{x}(t) + D\,u(t), \tag{50}$$

con las matrices:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{K_{stiff}}{J_{base}} & -\frac{K_b K_t K_g^2}{R J_{base}} & 0 \\ 0 & -\left(\frac{K_{stiff}}{J_T} + \frac{K_{stiff}}{J_{base}}\right) & -\frac{K_b K_t K_g^2}{R J_{base}} & 0 \end{bmatrix}, \tag{51}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ \frac{K_t K_g}{R J_{base}} \\ \frac{K_t K_g}{R J_{base}} \end{bmatrix}, \tag{52}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \qquad D = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{53}$$

La matriz A recoge la dinámica interna del sistema: los términos $-\frac{K_{stiff}}{J_{base}}$ y $-\frac{K_{stiff}}{J_T}$ representan el efecto restaurador de la rigidez torsional, mientras que los coeficientes $-\frac{K_b K_t K_g^2}{R J_{base}}$ modelan la disipación equivalente introducida por la fuerza contraelectromotriz del motor [38]. La matriz B muestra cómo el voltaje aplicado $V_a(t)$ influye en la aceleración de los dos grados de libertad. La matriz C selecciona como salidas los ángulos de interés, y la matriz D es nula, lo que indica que no existe una relación directa de ganancia estática entre la entrada y las salidas [30, 39].

1.4.10. Redes neuronales artificiales

Las RNA surgieron como una forma de imitar, el funcionamiento del cerebro humano, permitiendo que las máquinas realicen tareas por sí mismas. Esta idea fue planteada por primera vez en 1943 por Warren McCulloch y Walter Pitts, en su trabajo titulado "Un cálculo lógico de las ideas inmanentes a la actividad nerviosa", el cual es considerado uno de los pilares fundacionales de lo que hoy conocemos como inteligencia artificial [40].

Estas redes están compuestas por unidades de procesamiento denominadas neuronas artificiales, las cuales se organizan en capas interconectadas capaces de aprender patrones complejos a partir de los datos. A diferencia de los algoritmos tradicionales, las RNA pueden generalizar comportamientos a partir de ejemplos, lo que las convierte en herramientas ideales para tareas que requieren aprendizaje y adaptación en entornos no lineales o inciertos[41].

En el ámbito del control de sistemas dinámicos, las RNA han demostrado ser eficaces al abordar problemas de modelado, identificación y control de plantas con dinámicas complejas, especialmente en presencia de no linealidades o parámetros desconocidos. Gracias a su capacidad para aproximar funciones continuas con alto grado de precisión, las RNA se utilizan en esquemas tanto directos como inversos de control, facilitando la generación de señales de control adaptativas y robustas[42].

1.4.11. Arquitectura de una red neuronal artificial básica

Una RNA básica está conformada por un conjunto de nodos o neuronas artificiales organizadas en capas: una capa de entrada, una o varias capas ocultas y una capa de salida. Cada conexión entre neuronas posee un peso sináptico, que determina la influencia que una neurona tiene sobre otra. La información fluye hacia adelante a través de la red, donde cada neurona aplica una función de activación sobre la suma ponderada de sus entradas. Las funciones de activación comúnmente utilizadas incluyen la sigmoide, tangente hiperbólica y ReLU, dependiendo del tipo de tarea y comportamiento deseado [43].

Las arquitecturas de redes neuronales artificiales se pueden clasificar en tres tipos principales según su estructura y flujo de información: redes feedforward de una sola capa, redes feedforward multicapa (MLP) y redes recurrentes. Las redes feedforward de una sola capa constan de una capa de entrada que se conecta directamente a la capa de salida, sin realizar procesamiento intermedio, lo que limita su capacidad de representar funciones no lineales complejas. Por otro lado, las redes multicapa (MLP), como la empleada en este trabajo, incorporan una o más capas ocultas que permiten una mayor capacidad de representación al capturar relaciones no lineales más complejas entre la entrada y la salida. Estas redes utilizan funciones de activación en las neuronas ocultas y, gracias a su conectividad completa entre capas adyacentes, permiten aproximar funciones arbitrarias según el teorema de aproximación universal. Finalmente, las redes recurrentes introducen conexiones hacia atrás (feedforward), permitiendo la modelación de dinámicas temporales y la memoria interna, lo que las hace adecuadas para tareas secuenciales o sistemas con retroalimentación explícita [41].

En este caso, se utilizará el MLP bajo un enfoque de control basado en modelo inverso, el cual será entrenado para generar la señal de control a partir de los estados del sistema y la referencia deseada, tema que se desarrolla en la siguiente subsección.

1.4.12. Perceptrón multicapa

El Perceptrón multicapa (MLP) es una de las redes neuronales más empleadas. Un método fue establecido por Rumelhart, Hinton y Williams [45] para que una red de este tipo fuera capaz de aprender la relación entre un grupo de patrones de entrada y sus respectivas salidas. Este procedimiento se denomina *backpropagation error* (propagación del error hacia atrás). Se ha comprobado, gracias al teorema de aproximación universal, que un MLP

con una única capa oculta y un número suficiente de neuronas es capaz de aproximar cualquier función continua definida en un subconjunto compacto de $\mathbb{R}^n$ con el nivel de precisión requerido [44].

Un MLP consiste en una capa de entrada, una de salida y al menos una oculta, no obstante, se ha evidenciado que para la mayoría de los inconvenientes basta con tener únicamente una capa oculta [46]. En este tipo de modelos, las conexiones entre nodos solo se establecen entre las neuronas de una capa específica y las neuronas de la siguiente, no existen conexiones laterales ni hacia atrás. Por consiguiente, la información se envía constantemente desde la capa de entrada a la de salida. La arquitectura de un MLP está representada en la figura 1.

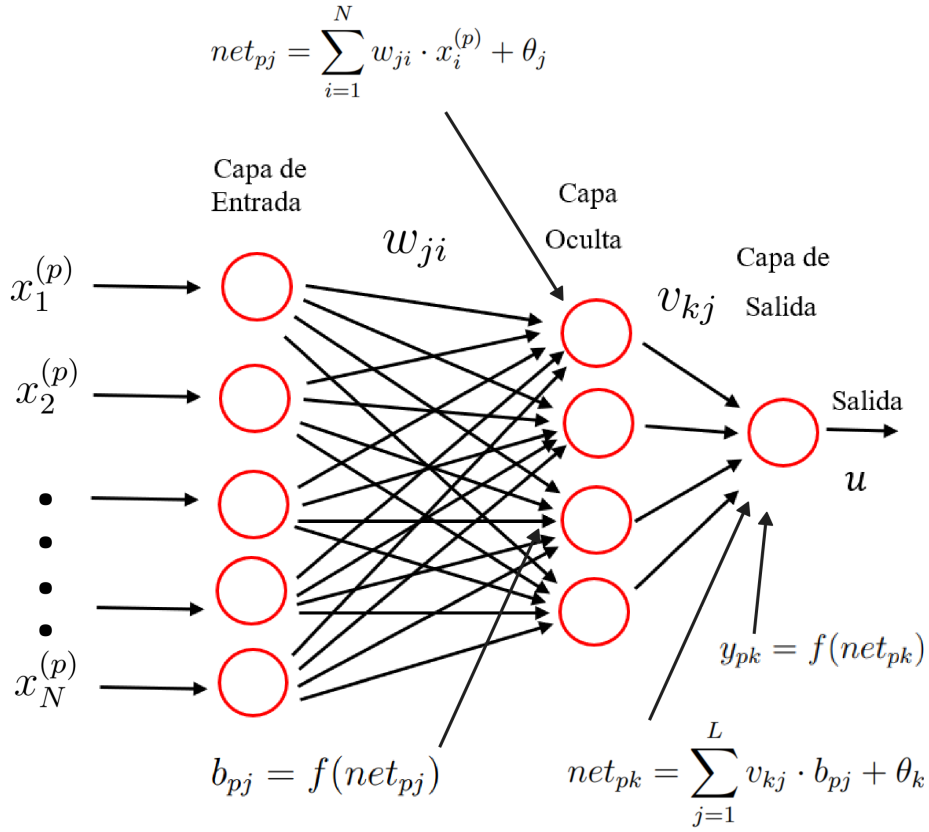


Figura 1: Arquitectura de una red MLP

Cada conexión entre neuronas está ponderada por un coeficiente denominado *peso sináptico*, el cual representa la fuerza de la conexión. En un MLP clásico de tres capas, se definen los siguientes conjuntos de pesos:

- w_{ji}: peso entre la neurona de entrada i y la neurona oculta j.
- v_{kj}: peso entre la neurona oculta j y la neurona de salida k.

Palmer, Montaño y Jiménez [47] afirman que en el algoritmo de propagación hacia adelante (forward propagation) hay una etapa de entrenamiento o aprendizaje, en la cual se ajustan los pesos de la red para que la salida obtenida por la red cuando se presenta un patrón de entrada específico coincida con la salida esperada. También existe una fase en la que se proporciona a la red un patrón de entrada y este es transmitido a través de las capas sucesivas de neuronas hasta lograr una salida.

Dado un patrón de entrada p, este se expresa como un vector columna:

$$\mathbf{x}^{(p)} = [x_1^{(p)}, x_2^{(p)}, \ldots, x_N^{(p)}]^\top \tag{54}$$

donde:

- $x_i^{(p)}$ representa la i-ésima característica del patrón p.
- N es el número total de entradas de la red.
- El superíndice (p) indica que el valor corresponde al patrón número p del conjunto de entrenamiento.

Cada neurona j de la capa oculta recibe las señales de entrada ponderadas por los pesos w_{ji}, y una constante adicional denominada umbral o sesgo θ_j. La entrada neta que recibe la neurona j ante el patrón p se define como:

$$net_{pj} = \sum_{i=1}^{N} w_{ji} \cdot x_i^{(p)} + \theta_j \tag{55}$$

donde:

- w_{ji} es el peso que conecta la entrada i con la neurona oculta j.
- θ_j es el umbral o sesgo de la neurona j, equivalente a un peso adicional asociado a una entrada constante de valor 1.
- net_{pj} representa la entrada total (lineal) a la neurona j para el patrón p.

La salida de la neurona j se calcula aplicando una función de activación no lineal (generalmente sigmoidea) a la entrada neta:

$$b_{pj} = f(net_{pj}) \tag{56}$$

donde:

- $f(\cdot)$ es la función de activación, comúnmente de tipo sigmoidal: $f(z) = \frac{1}{1+e^{-z}}$.
- b_{pj} es la salida (activación) de la neurona j de la capa oculta para el patrón p.

Las salidas de las neuronas ocultas se transmiten hacia las neuronas de la capa de salida mediante pesos v_{kj}. La entrada neta a la neurona de salida k es:

$$net_{pk} = \sum_{j=1}^{L} v_{kj} \cdot b_{pj} + \theta_k \tag{57}$$

donde:

- L es el número de neuronas en la capa oculta.
- v_{kj} es el peso que conecta la neurona oculta j con la neurona de salida k.
- θ_k es el umbral (bias) de la neurona de salida k.
- net_{pk} es la entrada neta a la neurona k para el patrón p.

Aplicando nuevamente la función de activación (que puede ser sigmoide o lineal según la tarea), se obtiene la salida final de la neurona k:

$$y_{pk} = f(net_{pk}) \tag{58}$$

donde:

- y_{pk} es la salida generada por la red para la neurona k al presentarse el patrón de entrada p.

Durante el entrenamiento de la red, se busca minimizar la diferencia entre las salidas producidas por la red y_{pk} y las salidas deseadas d_{pk} mediante una función de error. El entrenamiento es de tipo supervisado, ya que para cada entrada se conoce la salida deseada[47].

La función de error para un patrón p se define como:

$$E_p = \frac{1}{2}\sum_{k=1}^{M}(d_{pk} - y_{pk})^2 \tag{59}$$

donde:

- d_{pk} es el valor objetivo o deseado para la neurona de salida k.
- M es el número total de neuronas en la capa de salida.
- E_p es el error para un patrón especifico p.

El error total sobre todos los patrones de entrenamiento se expresa como:

$$E = \sum_{p=1}^{P} E_p \tag{60}$$

donde P es el número total de patrones de entrenamiento.

El entrenamiento del Perceptrón Multicapa (MLP) tiene como objetivo minimizar el error total E que comete la red al comparar su salida con la salida deseada para todos los patrones del conjunto de entrenamiento. Para ello, se emplea el método del gradiente descendente, el cual consiste en modificar iterativamente los pesos de la red en la dirección opuesta al gradiente del error, es decir, en la dirección que reduce más rápidamente dicho error.

El gradiente del error E respecto a un peso cualquiera de la red, por ejemplo w_{ji}, se define como la derivada parcial del error con respecto a ese peso. Matemáticamente se expresa como:

$$\frac{\partial E}{\partial w_{ji}} = \sum_{p=1}^{P} \frac{\partial E_p}{\partial w_{ji}} \tag{61}$$

donde:

- E es el error total acumulado sobre los P patrones.
- E_p es el error para un patrón específico p.
- w_{ji} es el peso que conecta la entrada i con la neurona oculta j.

Luego, el ajuste del peso w_{ji} se realiza aplicando la regla:

$$\Delta w_{ji} = -\eta \cdot \frac{\partial E}{\partial w_{ji}} = -\eta \cdot \sum_{p=1}^{P} \frac{\partial E_p}{\partial w_{ji}} \tag{62}$$

donde:

- η es la tasa de aprendizaje, un parámetro positivo que controla el tamaño del paso en cada iteración.
- Δw_{ji} es la corrección que se aplicará al peso w_{ji}.

Este procedimiento implica los siguientes pasos:

1. Se presentan todos los patrones del conjunto de entrenamiento uno a uno.
2. Para cada patrón p:
   - Se lleva a cabo una propagación hacia adelante para conseguir y_{pk}.
   - Se calcula el error E_p.
   - Se determina el gradiente $\frac{\partial E_p}{\partial w_{ji}}$ utilizando la regla de la cadena (backpropagation del error), moviéndose hacia atrás desde la capa de salida a las capas previas.
3. Una vez presentados todos los patrones, se acumulan los gradientes de todos los patrones (modo *batch*).
4. Por último, se actualizan los pesos utilizando la regla previa.

Este procedimiento se repite durante numerosas épocas o iteraciones hasta que el error total E llega a un nivel bajo o se cumple algún criterio de detención (como, por ejemplo, la saturación del error o un número máximo de épocas).

Es importante señalar que la dirección del gradiente ∇E indica el sentido de mayor aumento del error, por lo cual su opuesto $(-\nabla E)$ nos conduce a disminuirlo de la manera más eficiente posible. Este principio es el fundamento del algoritmo de entrenamiento *backpropagation*, que se usa frecuentemente en redes MLP.

El MLP, al poseer esta habilidad de aprender relaciones no lineales complejas, se convierte en una herramienta versátil para el diseño de controladores en sistemas dinámicos, sobre todo en los con comportamientos que no son lineales en gran medida y con acoplamientos, como el SREF del presente proyecto.

1.4.13. Control por modelo inverso del sistema

El control neuronal por modelo inverso es una estrategia de control que emplea la dinámica inversa del proceso para cancelar la dinámica natural del mismo. El objetivo principal es generar una señal de control que minimice el error entre la salida del sistema y la referencia deseada [48]. Este enfoque parte del principio de que, si se dispone de una aproximación adecuada del modelo inverso de la planta, la conexión en cascada entre el controlador neuronal y el proceso resulta en un mapeo cercano a la identidad entre la referencia y la salida [50].

Según [49], "El control por modelo inverso busca cancelar la dinámica de la planta, una red neuronal constituye una aproximación matemática de dicho inverso". En este trabajo se emplea un perceptrón multicapa (MLP), configurado para recibir como entradas valores actuales y retardados tanto de la señal de control $u(k), u(k-1), \ldots, u(k-N)$ como de la salida de la planta $y(k), y(k-1), \ldots, y(k-N)$. Esta incorporación de retardos permite que la red capture el comportamiento dinámico del sistema y aprenda una representación no lineal de su inverso.

Durante la fase de entrenamiento es habitual excitar la planta con señales escalón de diferentes magnitudes y tiempos de aplicación aleatorios, con el fin de cubrir un rango amplio de operación y generar una base de datos representativa [51, 52]. Posteriormente, los datos se dividen en conjuntos de entrenamiento, validación y prueba, lo cual permite ajustar la red neuronal y evitar problemas de sobreentrenamiento, garantizando así una adecuada capacidad de generalización [53, 54].

En lo referente al algoritmo de entrenamiento, el método *Levenberg–Marquardt* es uno de los más empleados en este tipo de aplicaciones debido a su rapidez de convergencia y capacidad para alcanzar errores pequeños [49, 55]. La literatura especializada también recomienda complementar este entrenamiento con técnicas de regularización y validación cruzada, con el propósito de mitigar el sobreajuste y asegurar que la red neuronal modele de forma robusta el inverso del sistema [56, 57].

En la Figura 2 se representa el esquema general del control neuronal por modelo inverso, donde el neurocontrolador genera la señal de control a partir de la referencia y señales retardadas, mientras que el neuro-emulador estima el comportamiento de la planta para calcular el error utilizado en el ajuste del controlador.

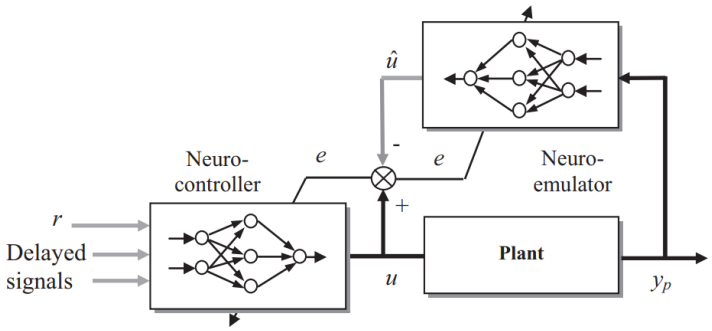


Figura 2: Control neuronal general por modelo inverso. Tomado de [58].

1.5. Métricas de evaluación

Para evaluar el desempeño del controlador neuronal y analizar la calidad del seguimiento frente a la referencia, se emplearon diversas métricas ampliamente utilizadas en control y procesamiento de señales. A continuación, se presenta una descripción breve de cada una junto con su formulación matemática.

- **Error cuadrático medio (RMSE):** mide el promedio del error cuadrático entre la señal deseada y la señal obtenida. Penaliza de forma significativa los errores de gran magnitud.

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{63}$$

- **Error absoluto medio (MAE):** representa el promedio del valor absoluto del error. Penaliza todos los errores de manera lineal.

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{64}$$

- **Coeficiente de determinación (R^2):** indica qué tan bien el modelo reproduce la variabilidad de la señal objetivo. Valores cercanos a 1 representan un ajuste adecuado.

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} \tag{65}$$

- **Integral del error absoluto (IAE):** cuantifica el error absoluto acumulado a lo largo del tiempo. Es útil para evaluar errores persistentes.

$$\text{IAE} = \int_0^T |e(t)|\, dt \tag{66}$$

- **Integral del error cuadrático (ISE):** mide el error cuadrático acumulado en el tiempo. Otorga mayor peso a los errores grandes.

$$\text{ISE} = \int_0^T e^2(t)\, dt \tag{67}$$

- **Integral del error absoluto ponderado en el tiempo (ITAE):** pondera el error absoluto multiplicándolo por el tiempo, penalizando más los errores que ocurren en etapas tardías.

$$\text{ITAE} = \int_0^T t\, |e(t)|\, dt \tag{68}$$

- **Integral del error cuadrático ponderado en el tiempo (ITSE):** similar al ITAE, pero penalizando de manera cuadrática los errores, resaltando aquellos que se mantienen o aumentan en el tiempo.

$$\text{ITSE} = \int_0^T t\, e^2(t)\, dt \tag{69}$$

1.6. Marco contextual

El presente trabajo de titulación se desarrolla en la carrera de Ingeniería en Electrónica y Automatización de la Universidad Estatal Península de Santa Elena, haciendo uso de los recursos disponibles en el Laboratorio de Control de Procesos. Dicho laboratorio constituye principalmente un espacio de trabajo académico en el que los estudiantes pueden desarrollar proyectos de simulación y análisis. Dentro de este marco, MATLAB/Simulink® es la plataforma central de desarrollo debido a sus capacidades avanzadas permite representar el comportamiento del sistema rotatorio con eslabón flexible, analizar su respuesta frente a distintas condiciones de operación y verificar la viabilidad del controlador RNA propuesto antes de su futura aplicación en entornos reales.

El principal beneficiario directo de este trabajo es la comunidad académica de la Universidad, dado que la metodología propuesta y los resultados

obtenidos podrán servir como material de referencia en asignaturas vinculadas a control automático, robótica, simulación de sistemas.

A nivel externo, la propuesta resulta pertinente para sectores en los que se requiera el diseño de estrategias de control para sistemas flexibles, tales como la robótica colaborativa, la automatización industrial, el área aeroespacial o el diseño de manipuladores ligeros. La metodología planteada es escalable y replicable, lo que facilita su adaptación a diferentes contextos y condiciones de operación.

2. Métodos y diseño experimental

El presente capítulo describe los enfoques metodológicos utilizados para el desarrollo del modelo matemático, la creación del controlador basado en RNA, así mismo las métricas utilizadas para evaluar su desempeño.

2.1. Métodos de investigación

2.1.1. Investigación documental

Se realizó una revisión bibliográfica detallada sobre el modelado dinámico de sistemas mecánicos flexibles, los fundamentos matemáticos aplicables (como la teoría de Euler-Bernoulli), y el uso de RNA como técnicas de control no lineal. Esta etapa de documentación permitió la creación del marco teórico requerido para respaldar el empleo de un Perceptrón Multicapa (MLP) como controlador inverso, basándose en investigaciones científicas actuales acerca del control de manipuladores flexibles y de la inteligencia artificial implementada en sistemas dinámicos.

2.1.2. Investigación analítica

Se formularon las ecuaciones matemáticas que describen la dinámica del sistema rotatorio flexible, considerando tanto la base como el eslabón elástico. A partir de estas ecuaciones se construyó un modelo en espacio de estados, que fue posteriormente utilizado como base para la generación de datos de simulación. Esta etapa también incluyó el análisis de estabilidad, resonancia y vibración, con el objetivo de identificar las variables críticas que afectan el comportamiento del sistema.

2.1.3. Modelado y simulación

Se desarrolló un entorno de simulación en MATLAB/Simulink que integra el modelo matemático del sistema con las condiciones dinámicas correspondientes. En este entorno se diseñó un conjunto de pruebas virtuales en las que se introdujeron señales de referencia escalonadas y perturbaciones controladas. El entorno sirvió como banco de pruebas tanto para validar el modelo como para generar los datos de entrenamiento y validación de la red neuronal.

2.1.4. Investigación aplicada

Se entrenó una red neuronal artificial tipo MLP como modelo inverso del sistema, empleando el conjunto de datos generados en la simulación. La red

fue incorporada como controlador dentro del entorno MATLAB/Simulink, en sustitución de un esquema de control clásico. Se evaluó el desempeño del controlador neuronal mediante métricas de seguimiento de referencia, reducción de vibraciones y robustez ante perturbaciones, demostrando la aplicabilidad del enfoque propuesto en el contexto de sistemas no lineales con dinámica flexible.

2.2. Descripción del proyecto

El presente proyecto de titulación se enfoca en el diseño y simulación de un sistema de control basado en RNA para un sistema rotatorio con un eslabón flexible, el cual está inspirado en el Rotary Flexible Link de Quanser. Se busca desarrollar una estrategia de control que permita seguir una referencia dada y minimizar la oscilacion en la punta del enlace flexible.

Para lograr este cometido, primero se lleva a cabo el modelado matemático del sistema, considerando las dinámicas del eje rotatorio y las propiedades mecánicas del eslabón flexible, asi mismo el acople entre ambos componentes. Este modelo teórico sirve como base para el desarrollo del controlador neuronal, teniendo en cuenta que el sistema presenta comportamientos no lineales.

Una vez obtenido el modelo matemático, se procede a usar un entorno de simulación como MATLAB/Simulink (R2022b)®. En donde permite emular el comportamiento del sistema, incluyendo la presencia de perturbaciones externas y variaciones en los parámetros del modelo. La simulación considera dos variables para el control: el ángulo de rotación (θ), que determina la posición angular del eje principal, y el ángulo de flexión (α), que representa la deflexión del eslabón flexible.

Dado que el entorno de MATLAB/Simulink (R2022b)® permite simular el sistema, se propone la aplicación de un control basado en RNA, el cual se entrena para adaptarse a las dinámicas cambiantes del sistema. Donde a diferencia de los controladores tradicionales como el PID en sistemas no lineales presentan una efectividad limitada, las RNA poseen la capacidad de aprender dependiendo el datasaet correspondiente para su desenvolvimiento.

Cabe destacar que este proceso, sienta las bases para futuras implementaciones físicas donde la metodología empleada permite identificar posibles desafíos en una implementación física, tales como la sensibilidad a ruido en los sensores, la necesidad de un procesamiento en tiempo real y las limitaciones asociadas a la potencia del actuador.

2.3. Componentes de la propuesta

Se presenta a la parte lógica del proyecto, o sea, el software que se empleará en la investigación, mencionando las herramientas que posibilitan la simulación y el análisis del comportamiento de sistemas dinámicos. Todo esto contribuye a lograr una mejor comprensión de cómo se comporta el sistema y a poner a prueba distintas soluciones.

2.3.1. Componentes lógicos

Software MATLAB

Este software resalta por su ambiente interactivo y la utilización de librerías especializadas, lo que le da ventajas particulares frente a otros programas de cálculo matemático. Cada una de estas bibliotecas incluye bloques independientes, lo que posibilita que el usuario emule, implemente y verifique diversos tipos de sistemas dinámicos dependientes del tiempo. Simulink® también está incluido en el entorno de MATLAB®, lo que proporciona acceso a una extensa gama de herramientas que facilitan el diseño de algoritmos y la visualización y análisis de los resultados generados en las simulaciones [59].

Deep Learning Toolbox

Deep Learning Toolbox™ es una herramienta que proporciona al usuario un conjunto de funciones, bloques y aplicaciones gráficas para Simulink®, todos ellos creados con el propósito de simplificar la simulación, prueba y desarrollo de redes neuronales artificiales [60].

2.4. Desarrollo controlador RNA

Esta sección detalla las fases realizadas para diseñar, entrenar y evaluar un controlador de tipo Perceptrón multicapa (MLP) basado en RNA, con una perspectiva de modelo inverso, para el sistema rotatorio que tiene un eslabón flexible. El proceso total se fraccionó en cuatro etapas: simulación de lazo abierto, creación del conjunto de datos, entrenamiento del modelo neuronal y puesta en funcionamiento del controlador de lazo cerrado.

2.4.1. Simulación del sistema en lazo abierto

Para generar los datos necesarios en el entrenamiento del modelo inverso basado en RNA, se llevó a cabo una simulación en lazo abierto del SREF.

Esta simulación permitió observar el comportamiento dinámico del sistema ante distintas excitaciones y recopilar las señales de entrada y salida correspondientes.

Tomando en cuenta una integración de tipo Euler con un paso de simulación de $dt = 0{,}02$ segundos, el sistema fue modelado en el dominio del tiempo discreto a través de su representación en espacio de estados. Para el modelo linealizado del sistema, se utilizó un punto de operación que está centrado en la base rotatoria, la flexión del eslabón y la dinámica del motor.

Los parámetros físicos empleados en dicha simulación se presentan en el Cuadro 1. Estos valores permiten caracterizar de manera precisa la dinámica mecánica y eléctrica del sistema, constituyendo la base para la correcta generación de los datos de entrenamiento de la RNA.

Cuadro 1: Parámetros físicos del sistema rotatorio con eslabón flexible.

Parámetro	**Descripción**	**Valor**
L	Longitud del eslabón flexible	0,45 m
m_l	Masa distribuida del eslabón	0,0008 kg
J_{link}	Inercia del eslabón	0,0042 kg·m^2
m_b	Masa puntual de la base	0,05 kg
J_b	Inercia de la base	$m_b \cdot L^2$
J_{load}	Inercia total del eslabón + base	$J_{link} + J_b$
J_{base}	Inercia de la base motriz	0,002 kg·m^2
R	Resistencia eléctrica del motor	2,6 Ω
K_g	Relación de engranaje	70
K_t	Constante de torque del motor	0,00767 N·m/A
K_b	Constante de fuerza contraelectromotriz	0,00767 V·s/rad
K_m	Constante del motor	0,00767
f_c	Frecuencia natural de vibración	1,9 Hz
K_{stiff}	Constante de rigidez del eslabón	$(2\pi f_c)^2 \cdot J_{load}$

Para generar las señales de excitación durante la simulación, se utilizó una entrada tipo escalón aleatorio, generada con un valor distinto cada 2 segundos, dentro del rango $[-1{,}5, 1{,}5]$ V. Este tipo de señal permite obtener una variedad de cambios, lo cual favorece el aprendizaje por modelo inverso de la red neuronal.

El modelo fue simulado durante 1000 segundos, y se registraron las salidas del sistema en cada instante de muestreo, correspondientes a:

- Ángulo de la base rotatoria $\theta(t)$
- Ángulo de flexión del eslabón $\alpha(t)$
- Velocidad angular de la base $\dot{\theta}(t)$
- Velocidad angular del eslabón $\dot{\alpha}(t)$

En la Figura 3 se presenta el resultado de la simulación en lazo abierto, incluyendo la entrada aplicada y las salidas obtenidas.

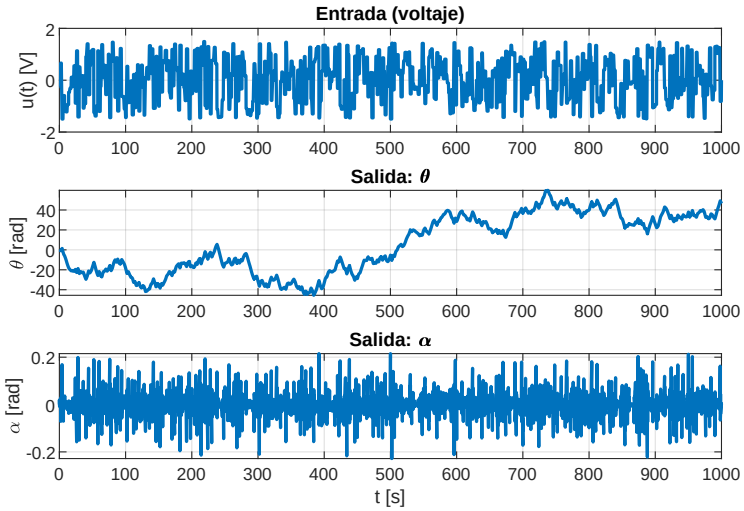


Figura 3: Simulación en lazo abierto del sistema ante entrada escalonada aleatoria

Toda esta información fue almacenada en un archivo de datos para ser procesada posteriormente en la etapa de generación del conjunto de entrenamiento para la red neuronal.

2.4.2. Generación del conjunto de datos para entrenamiento

Después de la simulación en lazo abierto, se crea el conjunto de datos que será utilizado para capacitar la red neuronal del modelo inverso. En esta fase, las variables del sistema se eligen y se organizan en vectores de entrada y salida que capturan apropiadamente la dinámica del proceso.

El conjunto de datos incluye una serie de regresores históricos de las salidas, sus derivadas, la referencia deseada y la acción de control aplicada. La finalidad es que, con esta información pasada, la red neuronal tenga la capacidad de predecir la entrada óptima $u(k)$ para que el sistema continúe siguiendo la referencia deseada $\theta_{\text{ref}}(k)$.

Estructura del conjunto de entrenamiento

El vector de entrada para la red neuronal en el instante de tiempo k está compuesto por 11 regresores, organizados como se muestra en el Cuadro 2.

Cuadro 2: Regresores utilizados como entradas del modelo inverso.

Regresor	**Descripción**	**Símbolo**
1	Ángulo de la base actual	$\theta(k)$
2	Ángulo de la base en $k-1$	$\theta(k-1)$
3	Ángulo de la base en $k-2$	$\theta(k-2)$
4	Ángulo del eslabón en $k-1$	$\alpha(k-1)$
5	Ángulo del eslabón en $k-2$	$\alpha(k-2)$
6	Velocidad angular de la base en $k-1$	$\dot{\theta}(k-1)$
7	Velocidad angular de la base en $k-2$	$\dot{\theta}(k-2)$
8	Velocidad angular del eslabón en $k-1$	$\dot{\alpha}(k-1)$
9	Entrada aplicada en $k-1$	$u(k-1)$
10	Entrada aplicada en $k-2$	$u(k-2)$
11	Entrada aplicada en $k-3$	$u(k-3)$

Como se observa en el Cuadro 2, los regresores incluyen tanto variables de posición y velocidad como las entradas pasadas aplicadas al sistema. Esta combinación busca capturar la memoria dinámica del proceso, lo que permite que la RNA pueda aproximar de manera más precisa la relación inversa entre la referencia deseada y la señal de control $u(k)$.

La salida deseada para cada muestra del dataset es la acción de control $u(k)$ aplicada en ese instante. El conjunto final contiene pares ordenados de la forma:

$$\text{Entrada: } \mathbf{x}_k = [\theta_{\text{ref}}(k), \theta(k-1), \theta(k-2), \ldots, u(k-3)]^\top, \quad \text{Salida: } u(k) \tag{70}$$

Procesamiento y visualización

Finalmente, los datos se graficaron para verificar su coherencia y posteriormente fueron almacenados en un archivo `.mat` para ser empleados en la etapa de entrenamiento. La Figura 4 muestra un ejemplo de la visualización de los regresores empleados.

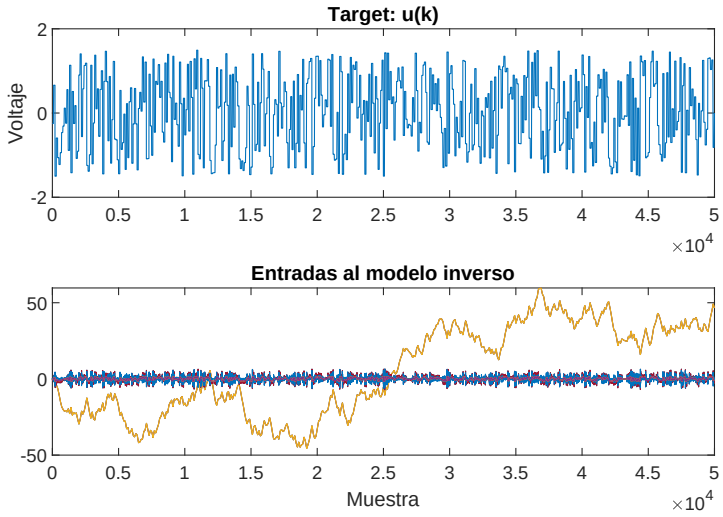


Figura 4: Visualización de los regresores del conjunto de entrenamiento.

La Figura 4 muestra los regresores que se consideraron para entrenar el modelo inverso. Las variables utilizadas como entradas, las cuales equivalen a las salidas del sistema que se han medido, están en la parte inferior; mientras que la señal de control $u(k)$, que representa la salida esperada del modelo, se encuentra en la parte superior. Esta etapa es fundamental, ya que la elección y el orden adecuados de los regresores influyen directamente en el aprendizaje del modelo inverso, lo cual permite capturar la dinámica real del sistema.

2.4.3. Entrenamiento del modelo inverso con RNA

En esta fase se entrena una red neuronal artificial (RNA) del tipo Perceptrón Multicapa (MLP) para que actúe como modelo inverso del SREF. El objetivo es que, dada una referencia deseada y un conjunto de regresores históricos del sistema, la red prediga la señal de control $u(k)$ que debe aplicarse para seguir dicha referencia.

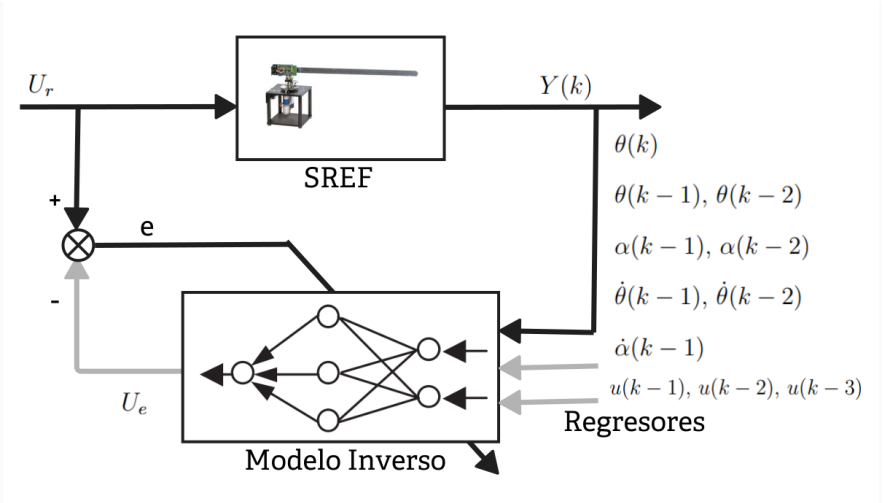


Figura 5: Esquema del proceso de identificación de la dinámica inversa del SREF.

El diagrama de la Figura 5 resume el proceso de identificación inversa. Este esquema permite visualizar claramente cómo se organiza el flujo de información durante el entrenamiento de la RNA, destacando la relación entre los regresores empleados y la señal de control que la red debe aprender a predecir.

Arquitectura de la red neuronal

Se utilizó una red *feedforward* con entrenamiento supervisado mediante el algoritmo *Levenberg-Marquardt*, conocido por su rápida convergencia en tareas de regresión no lineal [47]. Para determinar la mejor configuración, se evaluaron distintas combinaciones de capas ocultas y funciones de activación, las cuales se resumen en la Cuadro 3.

Cuadro 3: Configuraciones probadas para la red neuronal MLP.

Configuración	**Capas Ocultas**	**Funciones de Activación**
C1	[15]	*logsig – purelin*
C2	[20]	*logsig – purelin*
C3	[20, 10]	*logsig – logsig – purelin*
C4	[25, 15]	*logsig – logsig – purelin*
C5	[12, 2]	*logsig – logsig – purelin*
C6	[30, 20]	*logsig – logsig – purelin*

Tras entrenar las diferentes configuraciones mostradas en el Cuadro 3, se obtuvieron las curvas de error cuadrático medio (MSE) correspondientes a las arquitecturas con desempeño más representativo. En la Figura 6 se presentan las trayectorias del error para cuatro de las seis redes evaluadas. En todos los casos se observa que las curvas de entrenamiento y validación tienden a converger de manera estable, lo que indica que las arquitecturas seleccionadas poseen una adecuada capacidad de aprendizaje del modelo inverso. No obstante, cada red presenta características particulares en cuanto a la rapidez de descenso del error y la suavidad en las trayectorias, reflejando el efecto de la variación en el número de neuronas y capas ocultas.

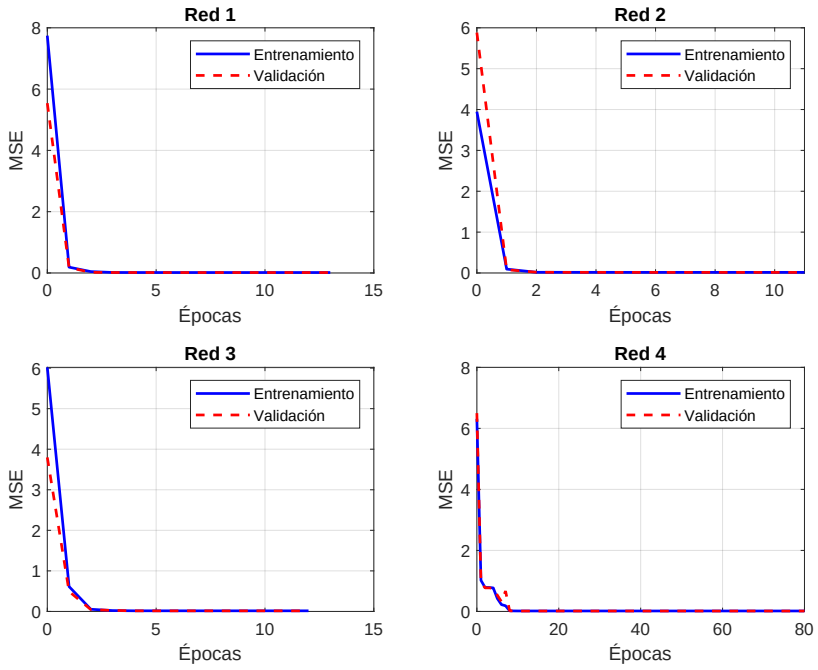


Figura 6: Curvas de MSE para cuatro de las arquitecturas neuronales evaluadas durante el entrenamiento del modelo inverso.

Parámetros de entrenamiento

Los parámetros empleados en el entrenamiento de la red se presentan en el Cuadro 4. Se seleccionaron estos valores para garantizar que el modelo convergiera de manera adecuada sin sobreajustarse, manteniendo un balance entre la capacidad de generalización y la precisión.

Cuadro 4: Parámetros de entrenamiento de la red neuronal.

Parámetro	**Valor**
Algoritmo	Levenberg-Marquardt (`trainlm`)
Número máximo de épocas	600
Error objetivo	10^{-4}
Tasa de validación	10 %
Proporción de entrenamiento	85 %
Normalización	`mapminmax` en entradas y salidas

La comparación entre la salida real del sistema y la que estima la red neuronal artificial entrenada se presenta en la Figura 7. Se nota una buena coincidencia, lo que muestra que la red consiguió aprender la relación inversa del sistema.

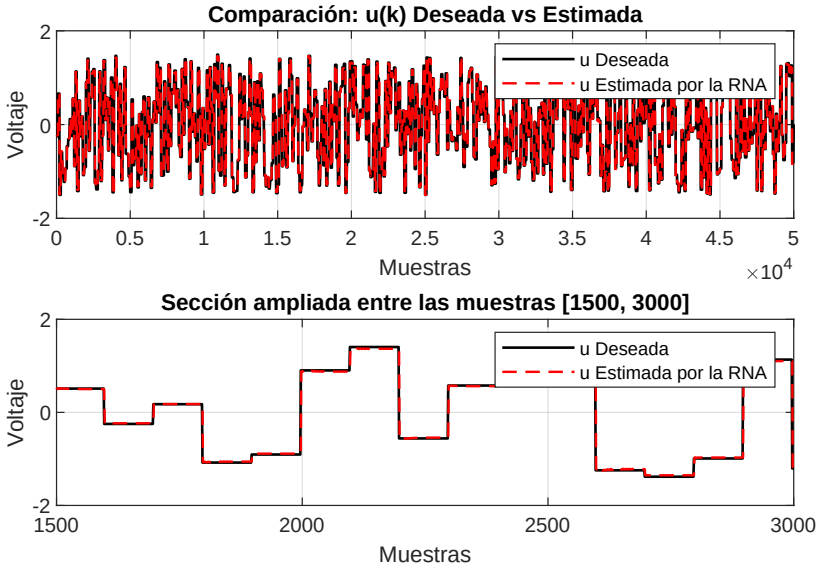


Figura 7: Comparación entre $u(k)$ real y $u(k)$ estimado por la RNA

Exportación del modelo entrenado

Una vez concluido el proceso de entrenamiento, la red neuronal entrenada fue exportada y guardada como un archivo `.mat` para su posterior implementación como controlador:

```
save red_inversa_MLP_con_orden_y_historial.mat net
```

Este modelo será integrado en el sistema de control en la siguiente fase, en donde se pondrá a prueba en simulaciones en lazo cerrado para evaluar su capacidad de seguimiento de la referencia.

2.4.4. Implementación del controlador neuronal

Una vez entrenada la red neuronal artificial (RNA) del tipo MLP como modelo inverso del sistema, se procedió a su implementación como controlador. En esta etapa se busca evaluar la capacidad del controlador por modelo inverso para generar la señal de control $u(k)$ que permita al sistema rotatorio con eslabón flexible seguir una referencia deseada $\theta_{\text{ref}}(k)$.

Antes de presentar el esquema implementado en Simulink, es conveniente ilustrar la estructura conceptual del controlador basado en el modelo inverso. Esta representación muestra cómo la RNA utiliza la referencia y los regresores del sistema para producir la acción de control aplicada a la planta.

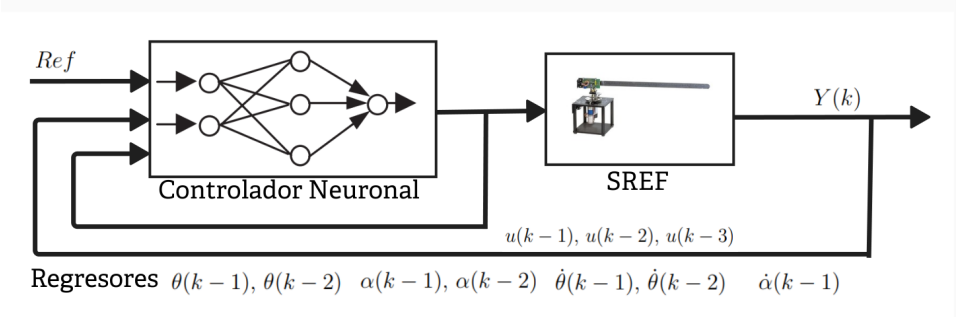


Figura 8: Diagrama del controlador neuronal por modelo inverso del SREF.

La Figura 8 muestra la estructura conceptual del controlador neuronal implementado a partir del modelo inverso entrenado. Este diagrama resume el flujo de información en lazo cerrado, donde la RNA genera en cada instante la señal de control necesaria para hacer que el sistema siga la referencia.

La RNA fue cargada desde el archivo `.mat` generado durante el entrenamiento, y posteriormente exportada a Simulink mediante el comando `gensim(net)`. Con ello se diseñó una estructura modular en la cual el bloque de red neuronal recibe como entradas los regresores correspondientes: referencia actual, salidas pasadas del sistema y entradas de control previas.

El controlador entrega como salida la estimación $\hat{u}(k)$, que es aplicada a la planta simulada en espacio de estados.

En la Figura 9 se presenta el esquema general del modelo implementado en Simulink.

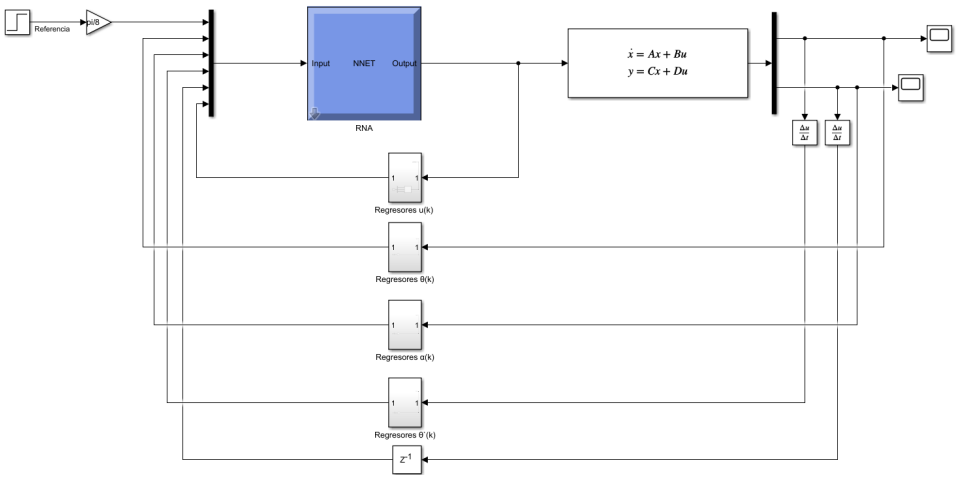


Figura 9: Estructura general del contolador RNA en Simulink

En este modelo, el bloque de red neuronal recibe los siguientes regresores, estructurados en un vector de entrada:

- $\theta(k-1)$, $\theta(k-2)$
- $\alpha(k-1)$, $\alpha(k-2)$
- $\dot{\theta}(k-1)$, $\dot{\theta}(k-2)$
- $\dot{\alpha}(k-1)$
- $u(k-1)$, $u(k-2)$, $u(k-3)$

Esta implementación permite ejecutar simulaciones para verificar el desempeño del sistema controlado ante diferentes secuencias de referencia. Los resultados obtenidos se discutirán a detalle en el siguiente capítulo, dedicado al análisis de desempeño del controlador.

3. Resultados

En la sección anterior se describió en detalle el diseño del controlador neuronal basado en el modelo inverso del (SREF). Para ello, se entrenaron múltiples RNA del tipo (MLP), con diferente número de capas ocultas, funciones de activación y parámetros de entrenamiento, para su posterior simulacion en lazo cerrado.

Se presenta los resultados obtenidos, al usar diferentes señales de referencia, permitiendo analizar el desempeño de las RNA mediante métricas como, el error cuadrático medio (RMSE), el sobrepaso, el tiempo de establecimiento y la estabilidad general del sistema. Además, se comparan las mejores y peores configuraciones entrenadas, y se incluye un análisis de robustez ante variaciones en la señal de referencia y en los parámetros del sistema.

3.1. Requerimientos del sistema de control

A continuación los requerimientos que orientan el diseño y la evaluación del sistema de control basado en redes neuronales artificiales (RNA). En el Cuadro 5 se resumen los criterios principales que garantizan el correcto funcionamiento del sistema rotatorio controlado, considerando seguimiento, rapidez, estabilidad y robustez.

Cuadro 5: Requerimientos principales del sistema de control RNA.

N.°	**Requerimiento**	**Descripción**
1	Seguimiento de referencia	La salida angular $\theta(k)$ debe seguir la referencia $\theta_{ref}(k)$ con un error estacionario menor al 2 %.
2	Estabilidad del sistema	El sistema tiene que conservar la estabilidad BIBO en situaciones normales de funcionamiento.
3	Rapidez de respuesta	La duración del tiempo de establecimiento frente a una entrada tipo escalón no debe exceder los 8 s.
4	Limitación del sobreimpulso	El sobreimpulso máximo permitido será menor al 5 %.
5	Robustez ante perturbaciones	El sistema tiene que restaurar la referencia cuando ocurran alteraciones de hasta ±5 % del valor nominal.

Como se aprecia en el Cuadro 5, los requerimientos definen los objetivos de desempeño que se esperan alcanzar con las diferentes configuraciones de las RNA. El cumplimiento de estos parámetros se verifica posteriormente mediante los resultados experimentales y las métricas de error obtenidas para cada tipo de señal de referencia.

3.2. Restricciones del sistema de control

Las restricciones establecen las condiciones bajo las cuales se realizaron las simulaciones y entrenamientos de las RNA. Estas limitaciones son necesarias para acotar el comportamiento del modelo experimental y asegurar coherencia entre los escenarios de prueba. En el Cuadro 6 se presentan las principales restricciones aplicadas durante el desarrollo del sistema.

Cuadro 6: Restricciones principales del sistema de control RNA.

N.°	**Restricción**	**Descripción**
1	Modelo físico considerado	El modelo dinámico representa un sistema rotatorio con un eslabón flexible. Se incluyen las variables angulares θ y α, sin considerar fricción no lineal, acoplamientos externos ni cargas adicionales.
2	Simulación discreta	Teniendo en cuenta el modelo identificado del sistema, las simulaciones se ejecutan en un dominio discreto con un tiempo de muestreo de $T_s = 0{,}02$ segundos.
3	Tipo de señales de referencia	Se utilizan señales de entrada del tipo: (i) escalón sencillo, (ii) escalón periódico constante y (iii) escalón periódico variable, con períodos y amplitudes determinados para cada situación.
4	Arquitectura de la RNA	Se emplea una red MLP que posee dos capas ocultas. funciones de activación como `purelin` y `logsig`, y la capacitación a través del algoritmo `trainlm`.
5	Entorno de simulación	Se desarrolla en un ambiente computacional como MATLAB/-SIMULINK y se evalua los resultados sin aplicasrlos en tiempo real.

Estas restricciones nombradas en el Cuadro 6 son las que rigen la interpretación correspondiente de los resultados evaluador posteriormente.

3.3. Influencia de la arquitectura de la red MLP en el desempeño del modelo inverso

Durante el proceso de entrenamiento se evaluaron distintas configuraciones de redes neuronales MLP que difieren en el número de capas ocultas y la cantidad de neuronas por capa, como se muestra en el Cuadro 7.

Cuadro 7: Configuraciones probadas para la red neuronal MLP.

Configuración	**Capas Ocultas**	**Funciones de Activación**
C1	[15]	*logsig – purelin*
C2	[20]	*logsig – purelin*
C3	[20, 10]	*logsig – logsig – purelin*
C4	[25, 15]	*logsig – logsig – purelin*
C5	[12, 2]	*logsig – logsig – purelin*
C6	[30, 20]	*logsig – logsig – purelin*

Este análisis ayuda a comprender cómo esta configuración de arquitectura influye en la capacidad de la red para aprender el modelo inverso del SREF y en el comportamiento dinámico obtenido durante las simulaciones de seguimiento de θ y α.

Los resultados de la investigación estructural de las RNA se presentan en cuatro secciones: (i) el impacto del número de capas ocultas, (ii) el impacto de la cantidad de neuronas, (iii) la mejor configuración elegida, y (iv) los lineamientos generales de diseño que se extraen de las pruebas experimentales.

3.3.1. Efecto del número de capas ocultas

El Cuadro 8 muestra el efecto de modificar el numero de capas de la red, comparando los casos con una y dos capas ocultas. Se observa que una única capa presenta limitaciones para modelar la dinámica completa del sistema, mientras que dos capas permiten una representación jerárquica más rica y un mejor seguimiento de las variables del sistema.

Cuadro 8: Influencia del número de capas ocultas en el comportamiento del modelo inverso.

Efecto del número de capas ocultas		
N.°	**Configuración**	**Comportamiento y análisis**
1	**Una capa oculta (C1–C2)**	Limitaciones en la representación de dinámicas no lineales que son complejas. Aprenden la tendencia general del sistema, sin embargo, no consiguen replicar correctamente los acoplamientos entre θ y α. Como consecuencia, se da un seguimiento que incluye sobretiros y oscilaciones debido a una representación insuficiente de la dinámica inversa.
2	**Dos capas ocultas (C3–C6)**	La primera capa aprende las combinaciones esenciales de las variables (derivadas, retardos y ángulos), y la segunda capa mejora esas conexiones. Esto da como resultado un seguimiento más preciso y estable de θ, con una reducción en las oscilaciones de α. Sin embargo, un número muy alto de capas o neuronas puede generar sobre entrenamiento.

El uso de dos capas ocultas proporciona un equilibrio adecuado entre precisión y estabilidad para modelar la naturaleza inversa y acoplada del SREF.

3.3.2. Efecto del número de neuronas por capa

El Cuadro 9 resume la influencia de la cantidad de neuronas por capa sobre la capacidad de ajuste y la estabilidad de la red. En general, se encontró que un número moderado de neuronas proporciona un buen equilibrio entre precisión, rapidez de aprendizaje y comportamiento físico coherente del modelo inverso.

Cuadro 9: Influencia del número de neuronas del modelo inverso.

Efecto del número de neuronas por capa		
N.°	**Configuración**	**Comportamiento y análisis**
1	**Pocas neuronas**	Capacidad de ajuste escasa, con respuestas más lentas y la presencia de sobretiros. La red no logra retratar completamente la dinámica del sistema, lo que produce un rendimiento estable aunque lento.
2	**Cantidad moderada de neuronas (C3–C5)**	Equilibrio adecuado entre la estabilidad y la precisión. Sin causar sobreoscilaciones, capturan correctamente la relación no lineal entre θ, α y sus derivadas. Ofrecen un comportamiento suave de α y un seguimiento de θ con un sobretiro que es moderado..
3	**Exceso de neuronas (C6)**	La habilidad de ajuste se incrementa, pero a la vez lo hace la sensibilidad al ruido y el peligro de sobreentrenamiento. Se registran sobretiros grandes, oscilaciones constantes y pérdida de generalización porque la red tiene la tendencia a memorizar los datos utilizados para el entrenamiento.

Los resultados experimentales confirman que mantener una cantidad moderada de neuronas (entre 10–25 en la primera capa y 2–6 en la segunda) permite una modelación adecuada sin pérdida de estabilidad ni sobreajuste.

3.3.3. Selección de la configuración óptima

El Cuadro 10 presenta el análisis detallado de la configuración óptima seleccionada C5 = [12, 2], la cual mostró el mejor desempeño global en términos de estabilidad y generalización.

Cuadro 10: Análisis de la configuración óptima seleccionada (C5 = [12, 2]).

Configuración óptima del modelo inverso		
N.°	**Característica**	**Descripción y justificación**
1	**Arquitectura [12, 2]**	Dos capas ocultas, con una segunda capa de tamaño más pequeño, funcionan como un cuello de botella regularizador, este impide que el ruido se amplifique y optimiza la generalización del modelo inverso.
2	**Desempeño dinámico**	Muestra el comportamiento más adecuado entre estabilidad y rapidez: Oscilación mínima y respuesta suave de α junto con el seguimiento de θ con sobretiro moderado.
3	**Mandos de control**	Las señales $u(t)$ obtenidas son físicamente coherentes, sin picos abruptos ni saturaciones. Esto demuestra que la red aprendió una función inversa estable y realista.
4	**Comparación con otras configuraciones**	Supera a las redes de una sola capa (C1–C2), que mostraron oscilaciones, y a las más grandes (C6), que evidenciaron un sobreajuste. Su desempeño en general la posiciona como la arquitectura mejor calificada.

3.3.4. Criterios generales de selección

Finalmente, El Cuadro 11 resume los criterios generales de diseño derivados de las observaciones anteriores. Estos lineamientos sirven como guía práctica para la elección de la arquitectura MLP en sistemas con características similares.

Cuadro 11: Criterios generales para la selección de la arquitectura MLP.

Criterios generales para la red neuronal MLP		
N.°	**Criterio**	**Descripción y justificación**
1	Número de capas ocultas	Emplear al menos dos capas ocultas en sistemas con acoplamientos, una sola capa suele ser insuficiente para capturar las no linealidades dinámicas del modelo inverso.
2	Número de neuronas por capa	Mantener un tamaño moderado (por ejemplo: en la primera capa de 10 a 25 y en la segunda de 2 a 6). Aunque un número excesivo de neuronas aumenta el riesgo de sobre entrenamiento y oscilaciones, una cantidad muy baja limita la capacidad.
3	Validación por escenarios	Para evaluar la fortaleza y la generalización más allá de los patrones de entrenamiento, se realiza una validación empleando referencias variadas (escalón variable, simple o constante).
4	Consistencia de datos	Para captar un comportamiento positivo, es necesario garantizar un conjunto de datos que sea rico en la dinámica del sistema.

3.4. Evaluación con señal tipo escalón

Se procede a analizar el desempeño de las seis configuraciones de redes neuronales MLP implementadas como controladores inversos, ante una señal de referencia tipo escalón simple. Se busca evaluar la capacidad de seguimiento de la variable $\theta(t)$ como la reducción de oscilaciones en la punta del eslabón flexible $\alpha(t)$, según la referencia aplicada.

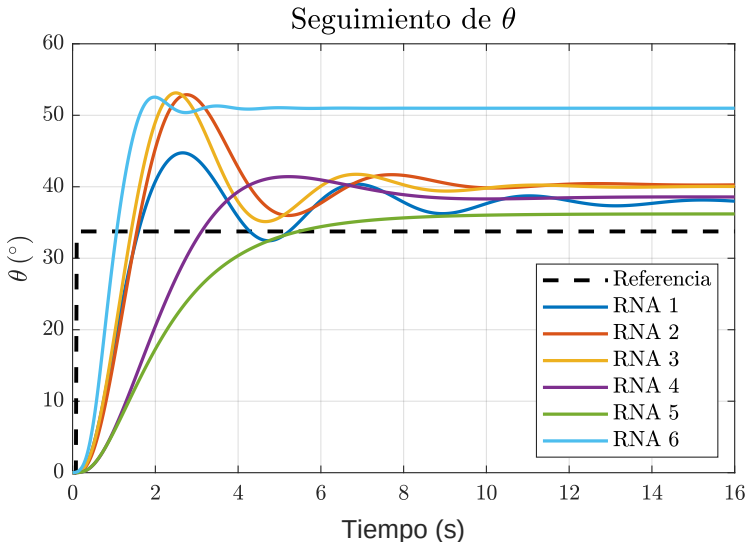


Figura 10: Comparación de la respuesta de $\theta(t)$ para las seis configuraciones de RNA ante un escalón Simple.

En la Figura 10 se puede observar que no todas las configuraciones RNA tienen un correcto seguimeinto de referencia. La RNA 6 (curva celeste) cuenta con un sobrepaso excesivo y se estabiliza en un valor final muy superior a la referencia. La RNA 3 (naranja) y la RNA 2 (roja) también presentan sobreelongaciones, aunque eventualmente logran estabilizarse cerca del valor deseado.

Por otra parte, la RNA 1 (de color azul) muestra una oscilación moderada que disminuye con el tiempo. La RNA 4 (en color morado) presenta un incremento más paulatino, sin picos altos, lo que permite una estabilización mas lenta y mayor a la referencia.

Mientras que, la RNA 5 (verde claro) muestra el comportamiento más eficiente, logra un seguimiento con bajo sobrepaso, estabilización rápida lo mas cercana a la referencia. Este comportamiento denota que en las configuraciones evaluadas, la RNA 5 obtiene el mejor resultado entre velocidad de respuesta, precisión y estabilidad.

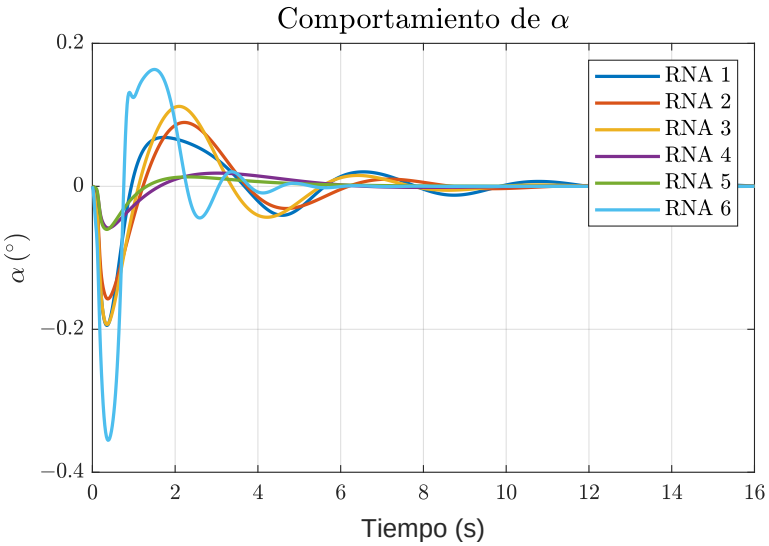


Figura 11: Comparación de la respuesta del ángulo del eslabón $\alpha(t)$ para las seis configuraciones de RNA.

La Figura 11 muestra el comportamiento de la punta del eslabón flexible $\alpha(t)$. Se observa que la RNA 6 (celeste) presenta la mayor amplitud de oscilaciones que persiste durante un tiempo considerable, lo que indica que su capacidad de amortiguación puede ser perjudicial para el sistema. En cambio la RNA 1 (azul), RNA 2 (roja) y la RNA 3 (naranja) muestran oscilaciones moderadas, que disminuyen poco a poco hasta la estabilización.

Por el contrario, la RNA 4 (morada) y la RNA 5 (verde claro) muestran un comportamiento más estable. Sobre todo, la RNA 5 nuevamente presenta un mejor comportamiento al tener una curva con oscilaciones baja y rápida disipación, con esto la RNA 5 no solo es efectiva en el seguimiento de la referencia, sino también en la reducción de las oscilaciones del eslabón flexible.

Cuadro 12: Parámetros dinámicos observados en las respuestas ante señal escalón.

Parámetros de respuesta de los controles RNA						
N.°	**Diseño**	**Sobreelong. (θ)**	**T. estab. [s]**	**Valor final [°]**	**Punta del eslabón (α)** (Rango de oscilación)	**Asent. máx.**
1	RNA - C1	32.6 %	11.2800	37.92	[-0.1946 - 0.0684]	33.7500
2	RNA - C2	56.7 %	8.5200	40.28	[-0.1575 - 0.0894]	33.7500
3	RNA - C3	57.4 %	7.6800	40.02	[-0.1934 - 0.1118]	33.7500
4	RNA - C4	22.7 %	7.4800	38.55	[-0.0583 - 0.0184]	33.7500
5	RNA - C5	7.3 %	7.5600	36.20	[-0.0603 - 0.0132]	33.7500
6	RNA - C6	55.7 %	2.2000	50.99	[-0.3553 - 0.1635]	33.7500

Cuadro 13: Comparación de métricas de error para cada red neuronal ante una señal escalón.

RNA	**RMSE**	**MAE**	**R^2**	**IAE**	**ISE**	**ITAE**	**ITSE**
1	7.6990	5.4709	-9.4704	109.4850	1186.5131	840.0724	4077.4698
2	9.9954	8.2431	-16.6477	164.9611	1999.7175	1349.4845	10000.8271
3	9.5792	7.8250	-15.2088	156.5936	1836.6672	1287.0993	9084.7523
4	9.7163	7.1423	-15.6762	142.9398	1889.8024	1030.5302	6142.6258
5	9.1722	5.1073	-13.8607	102.2241	1684.2105	529.4320	2672.3264
6	17.4475	17.1498	-52.7721	343.1674	6091.4085	3442.3005	59368.3495

Los Cuadros 12 y 13 refuerzan lo observado gráficamente. La RNA 5 presenta los valores más bajos en sobreelongación (7.3 %) y error medio absoluto (MAE = 5.1), así como el menor índice ITAE, lo que indica mejor comportamiento transitorio y estabilidad. Asimismo, sus oscilaciones en α son de baja amplitud, lo que la convierte en una opción confiable y eficiente.

En cambio, la RNA 6, a pesar de ser la red más compleja en términos arquitectónicos, presenta un rendimiento deficiente: una elongación excesiva y una oscilación del eslabón muy amplia, además de valores de error significativamente superiores. Esto evidencia que una red más complicada no necesariamente significa un mejor control.

3.5. Evaluación con entrada escalón periódico variable

En esta sección se analiza el desempeño de las seis configuraciones de redes neuronales MLP ante una entrada de tipo escalón periódico con amplitudes

variables. Lo que permite evaluar la capacidad del controlador RNA ante cambios dinámicos de la referencia.

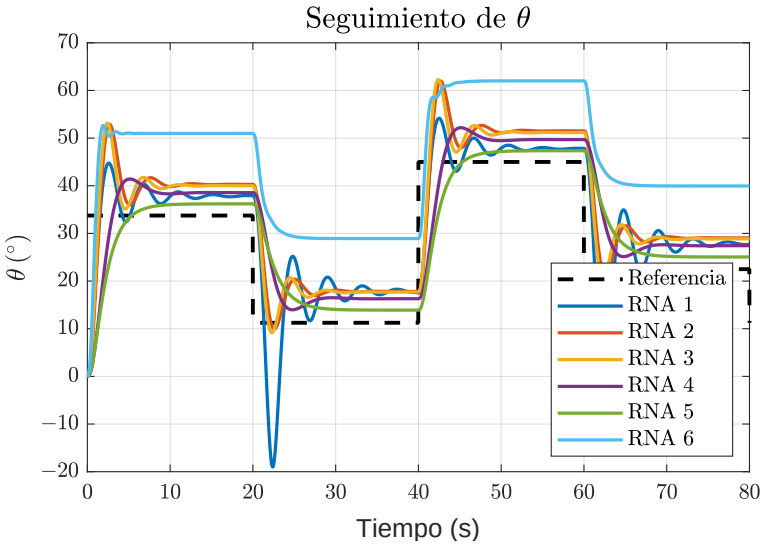


Figura 12: Respuesta del sistema para θ ante escalón periódico variable usando las seis configuraciones RNA.

La Figura 12 muestra la respuesta de las seis configuraciones de redes neuronales artificiales ante una señal de entrada tipo escalón periódico con amplitudes variables. Donde la RNA 1 (línea azul) muestra un comportamiento con grandes sobreelongaciones que denotan una deficiencia en el seguimeinto de referencia donde los cambios son bruscos. RNA 2 (línea roja) y RNA 3 (línea naranja) a pesar que el seguimiento es mas cercano a la referencia, presenta oscilaciones moderadas en el cambio de cada escalón. En cambio la RNA 4 (línea morada) tiene un seguimiento más uniforme en comparacion a las redes anteriores aunque no alcanza la referencia con exactitud en todos los escalones.

Mientras que la RNA 5 (línea verde claro) cuenta con una respuesta lenta en algunos tramos, sin embargo, presenta un comportamiento muy estable, con bajas oscilaciones y un valor muy cercano a la referencia.

Por último, la RNA 6 (línea celeste) alcanza los valores de referencia con rapidez, pero presenta un desplazamiento considerable respecto a la referencia. Este análisis determina que la configuración RNA 5 logra el mejor resultado entre estabilidad, bajo sobreimpulso y seguimiento de referencia, mientras que RNA 6 y RNA 1 muestran los peores desempeños.

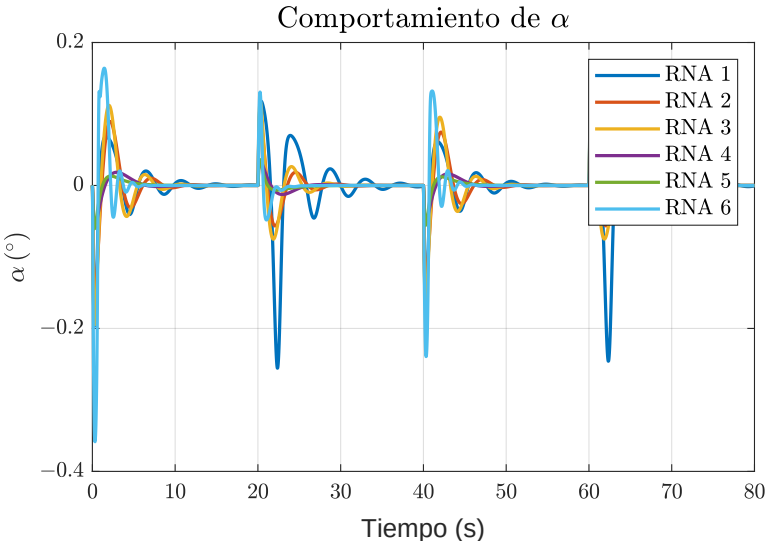


Figura 13: Comportamiento del ángulo del eslabón α para las distintas configuraciones RNA con entrada variable.

En la Figura 13 se presenta el comportamiento de la punta del eslabón flexible $\alpha(t)$ frente a una referencia escalonada de tipo periódico. La RNA 1 (línea azul) muestra oscilaciones grandes en los cambios descendentes de la señal escalón, aunque posteriormente se atenúan después de varios segundos. La RNA 2 y RNA 3 (línea roja) reducen levemente las oscilaciones respecto a RNA 1, pero aún presentan cambios fuertes ante cada escalón variable. Mientras que la RNA 4 (línea morada) tiene una clara reducción de oscilaciones y una respuesta mas suave durante el tiempo de ejecución, no obstante la RNA 5 (línea verde claro) presenta un comportamiento mejor a la red anterior, donde frente a escalones pronunciados las oscilaciones son mínimas. En cambio la RNA 6 (línea celeste) denota oscilaciones intensas similares a la RNA 1, aunque esta red responde rápido, esto genera una tardanza en disipar las vibraciones.

Dado el análisis, las configuraciones RNA 4 y especialmente RNA 5 resultan ser las más efectivas para mitigar las oscilaciones de $\alpha(t)$.

Los parámetros de desempeño dinámico correspondientes a cada una de las configuraciones de red neuronal evaluadas se presentan en el Cuadro 14, considerando características clave como la sobreelongación, el tiempo de establecimiento, el valor final alcanzado por la variable θ, y el rango de oscilación del ángulo del eslabón α.

Cuadro 14: Parámetros de respuesta para controladores con entrada de tipo escalón periódico variable.

Parámetros de respuesta – Entrada escalón periódico variable						
N.°	**Diseño**	**Sobreelong. (θ)**	**T. estab. [s]**	**Valor final [°]**	**Punta del eslabón (α)** (Rango de oscilación)	**Asent. máx.**
1	RNA - C1	381.5 %	73.9800	27.73	[-0.2556 – 0.1178]	42.9192
2	RNA - C2	451.0 %	68.2400	29.06	[-0.1586 – 0.0921]	50.7357
3	RNA - C3	453.6 %	67.4400	28.86	[-0.1950 – 0.1168]	51.0267
4	RNA - C4	363.9 %	67.1600	27.43	[-0.0586 – 0.0348]	40.9349
5	RNA - C5	320.9 %	67.7800	25.06	[-0.0605 – 0.0361]	36.1056
6	RNA - C6	451.2 %	64.2000	39.96	[-0.3585 – 0.1640]	50.7633

Por otro lado, el Cuadro 15 resume las métricas cuantitativas de error, tales como RMSE, MAE, R^2, IAE, ISE, ITAE e ITSE, las cuales aportan una evaluación integral del desempeño general de cada controlador bajo una señal de entrada escalonada periódica.

Cuadro 15: Métricas de desempeño para entrada escalón periódico variable.

RNA	**RMSE**	**MAE**	**R^2**	**IAE**	**ISE**	**ITAE**	**ITSE**
1	8.8691	6.5317	0.5029	522.1624	6280.3234	20022.8260	223073.4985
2	9.1393	7.6936	0.4721	615.1249	6669.3143	23273.9122	223548.4519
3	8.7681	7.3761	0.5141	589.7227	6137.4787	22370.7007	205547.5155
4	8.9462	6.8002	0.4942	543.6551	6390.3964	20005.5536	202088.5271
5	8.3926	5.2464	0.5549	419.3452	5623.0328	14944.0715	167810.0312
6	18.3018	17.9072	-1.1168	1432.3069	26783.6701	57311.4578	1065170.3955

De acuerdo con los resultados mostrados, la configuración RNA - C5 demuestra el mejor rendimiento en estabilidad y supresión de oscilaciones, obteniendo un coeficiente de determinación $R^2 = 0{,}5549$, un error cuadrático medio (RMSE) y error acumulado (IAE e ITAE) bajo, lo cual demuestra su capacidad de seguimiento ante una exigencia notable.

En cambio, la RNA - C6 presenta un comportamiento deficiente. Sus métricas de error son elevadas en en comparación con las demás configuraciones, a demás su sobre elongación y oscilaciones indican una deficiencia en la capacidad de control lo cual representaría inestabilidad si se implementara en un sistema físico.

Cabe resaltar que el rango de oscilación de $\alpha(t)$ fue más contenido en las configuraciones C4 y C5, lo cual demuestra la precisión de control del sistema rotatorio con eslabón flexible.

3.6. Evaluación ante señal escalón periódico constante

En esta sección se analiza el comportamiento del sistema controlado por redes neuronales MLP cuando la entrada consiste en un escalón periódico constante. Esta señal permite observar la capacidad de seguimiento ante cambios regulares y repetitivos.

En la Figura 14 se muestra el comportamiento de seguimiento de la variable θ frente a una señal de referencia escalón constante, de las seis configuraciones de redes neuronales artificiales desarrolladas.

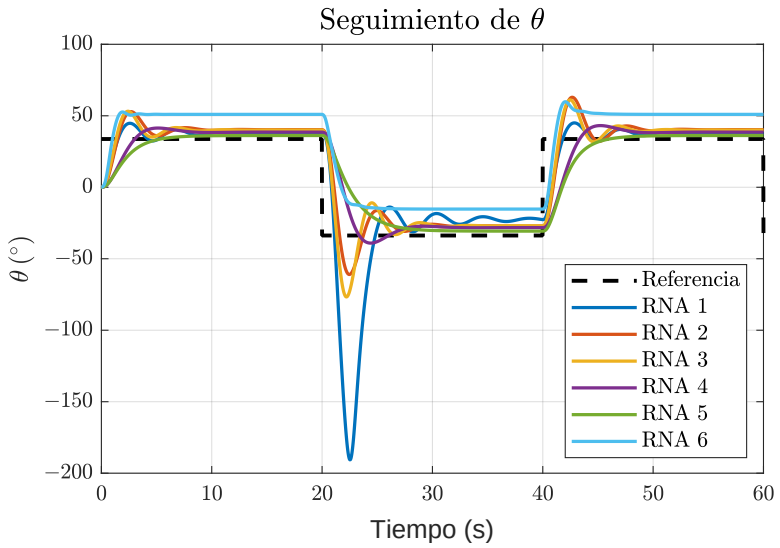


Figura 14: Respuesta angular θ para redes RNA ante señal de entrada escalón periódico constante.

La RNA 1 (línea azul) presenta una severa sobre elongación en el seguimiento a los cambios del escalon.

Mientras que la RNA 2 (línea roja) y la RNA 3 (línea naranja) muestran una respuesta más controlada que RNA 1, sin embargo presenta sobrepasos moderados y una oscilacion controlada. En cambio la RNA 4 (línea morada) y la RNA 5 (línea verde claro) destacan por su buen desempeño. Ambas redes presentan una respuesta con menor sobreelongación y un tiempo de establecimiento corto. En particular, RNA 5 demuestra un seguimiento suave cercano a la referencia sin generar oscilaciones pronunciadas. Por último, la RNA 6 (línea celeste) también muestra un comportamiento no deseado. A pesar de tener un rapido accionamiento al seguir la referencia, permanece en un valor superior a esta sin corregir su error de estado estacionario.

La Figura 15 presenta el comportamiento del eslabon flexible α, frente a los cambios generados por una señal de referencia escalón constante.

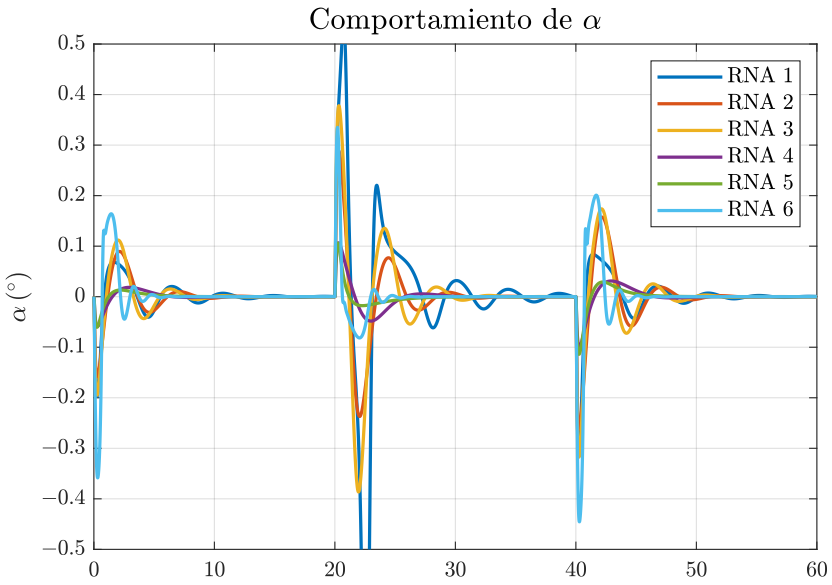


Figura 15: Respuesta de oscilación α del eslabón flexible ante escalón periódico constante.

La RNA 1 (línea azul) presenta oscilaciones fuertes después de cada cambio de referencia, con sobrepasos elevados, lo que deduce a un control deficiente en la reducción de oscilaciones. En cambio las RNA 2 (línea roja) y RNA 3 (línea naranja) también generan oscilaciones, aunque menos intensas que la red anterior. En contraste, las RNA 4 (línea morado) y RNA 5 (línea verde claro) muestran un comportamiento más estable, con oscilaciones de α menores y rápida disipación de estas. En especial, la RNA 5 logra una respuesta más suave y controlada. Finalmente, la RNA 6 (línea celeste) tiene un comportamiento similar a la RNA 1, con grandes oscilaciones y comportamiento no adecuado para el control del sistema.

Los resultados obtenidos presentan que las configuraciones RNA 4 y RNA 5 ofrecen un comportamiento más robusto al momento de menorar las vibraciones en la punta del eslabón.

Los desempeños de los controladores RNA ante una entrada tipo escalón periódico constante aparecen en los Cuadros 17 y 16. Mientras que la segunda tabla muestra las métricas de error, la primera incluye los parámetros más importantes de la respuesta dinámica.

Cuadro 16: Parámetros de respuesta dinámica para cada controlador basado en RNA (escalón periódico constante).

Parámetros de respuesta de los controles basados en RNA (escalón periódico constante)						
N.°	**Diseño**	**Sobreelong. (θ)**	**T. estab. [s]**	**Valor final [°]**	**Punta del eslabón (α)** (Rango de oscilación)	**Asent. máx.**
1	RNA - C1	-233.4 %	51.5000	37.94	[-1.081 – -0.540]	156.9031
2	RNA - C2	-286.5 %	50.3400	40.28	[-0.262 – -0.288]	96.6876
3	RNA - C3	-281.0 %	49.5200	40.02	[-0.386 – -0.377]	94.8444
4	RNA - C4	-227.8 %	47.8200	38.55	[-0.100 – -0.106]	76.8857
5	RNA - C5	-207.3 %	48.1000	36.20	[-0.114 – -0.106]	69.9549
6	RNA - C6	-277.2 %	44.3400	50.99	[-0.445 – -0.335]	93.5604

Se observa en el Cuadro 16 que las configuraciones C4 y C5 presentan tiempos de establecimiento más bajos (47.82 s y 48.10 s) y rangos de oscilación de la variable α reducidos ([−0,10, −0,11]), lo que refleja una mayor estabilidad. En contraste, las configuraciones C1 y C6 evidencian sobre elongaciones grandes (–233.4 % y –277.2 %) y tiempos de establecimiento más largos (51.50 s y 44.34), con oscilaciones más amplias en la punta del eslabón.

Cuadro 17: Métricas de desempeño para controladores RNA ante señal de entrada tipo escalón periódico constante.

RNA	**RMSE**	**MAE**	**R^2**	**IAE**	**ISE**	**ITAE**	**ITSE**
1	31.7209	16.0445	0.1166	1282.9760	80474.3587	54425.4854	3403155.1594
2	15.4694	10.4371	0.7899	834.2291	19100.4458	32438.9853	735540.4518
3	15.8182	10.5975	0.7803	847.0670	19974.0633	33297.9880	780650.7628
4	16.8459	9.6940	0.7509	774.7553	22658.5612	29764.4846	870525.6143
5	17.7625	9.4204	0.7230	752.8363	25193.9029	29147.5748	977208.7090
6	22.7026	20.3605	0.5475	1628.4212	41207.5519	65823.4133	1670674.9717

Se determina en el Cuadro 17 que C4 y C5 alcanzan valores relativamente bajos de RMSE (16.85 y 17.76) y MAE (9.69 y 9.42), así como errores acumulados IAE cercanos a 750, en comparación con C1, que presenta un RMSE de 31.72 y un IAE de 1282.98. Asimismo, el coeficiente de determinación R^2 es notablemente más alto en C4 (0.75) y C5 (0.72) que en C1 (0.12) y C6 (0.55), confirmando una mejor capacidad de seguimiento de la referencia.

Como se evidenció en las pruebas realizadas, las configuraciones C4 y C5 logran un equilibrio positivo entre estabilidad, rapidez y precisión, según lo demuestra la información. En cambio C1 y C6 presentaron comportamiento no deseado, con sobre elongación elevados, y estabilidad lejos de la referencia, lo que las hace inapropiadas para aplicaciones que exigen robustez.

3.7. Evolución del error durante el entrenamiento del modelo inverso

Se seleccionó la arquitectura correspondiente a la configuración C5, al haber demostrado el mejor desempeño general tanto en precisión como en estabilidad frente a las distintas señales de prueba. Esta red fue entrenada mediante aprendizaje supervisado utilizando el algoritmo Levenberg-Marquardt (`trainlm`), con una estructura de dos capas ocultas compuestas por 12 y 2 neuronas respectivamente, funciones de activación sigmoidal (`logsig`) y una capa de salida lineal (`purelin`).

La evolución del error durante el proceso de entrenamiento puede observarse en la Figura 16.

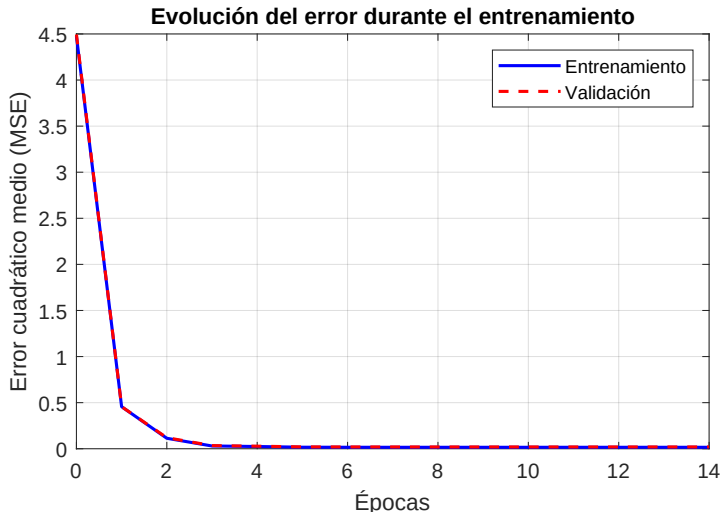


Figura 16: Evolución del error cuadrático medio (MSE) durante el entrenamiento de la red neuronal C5.

Se puede visualizar que el error baja en las primeras cinco épocas y se estabiliza rápidamente, con valores próximos a cero tanto en la validación como en el entrenamiento. El hecho de que esta convergencia sea rápida y estable indica que la red neuronal ha conseguido asimilar de manera eficaz la relación inversa entre las variables del sistema y la entrada de control. Asimismo, el hecho de que las dos curvas se superpongan muestra una buena habilidad para generalizar, algo fundamental para prevenir el sobreajuste y asegurar un buen rendimiento al utilizarla en lazo cerrado.

3.8. Robustez y estabilidad del sistema de control

Para evaluar la estabilidad y robustez del sistema ante perturbaciones externas, se introdujo una alteración abrupta sobre la referencia de la variable θ, controlada mediante la red neuronal C5, previamente identificada como la mejor configuración durante las pruebas anteriores.

La Figura 17 muestra cómo, ante la perturbación aplicada alrededor del segundo 30, la señal de salida θ presenta una desviación transitoria, pero es capaz de recuperar rápidamente su seguimiento a la referencia sin comprometer la estabilidad del sistema. Esta respuesta evidencia la robustez del controlador neuronal, al lograr mantener el desempeño ante variaciones inesperadas.

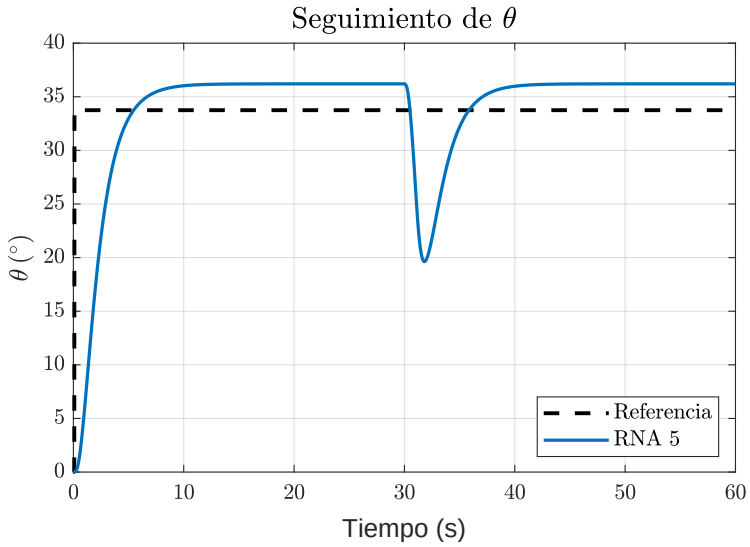


Figura 17: Respuesta de θ ante una perturbación externa usando la red C5.

En cuanto al comportamiento de la variable α, representativa de la flexión del eslabón, la Figura 18 revela un pequeño incremento localizado en el instante de la perturbación. No obstante, dicha oscilación es rápidamente amortiguada, lo cual sugiere que el sistema no solo es robusto, sino que mantiene un control eficiente de las vibraciones estructurales.

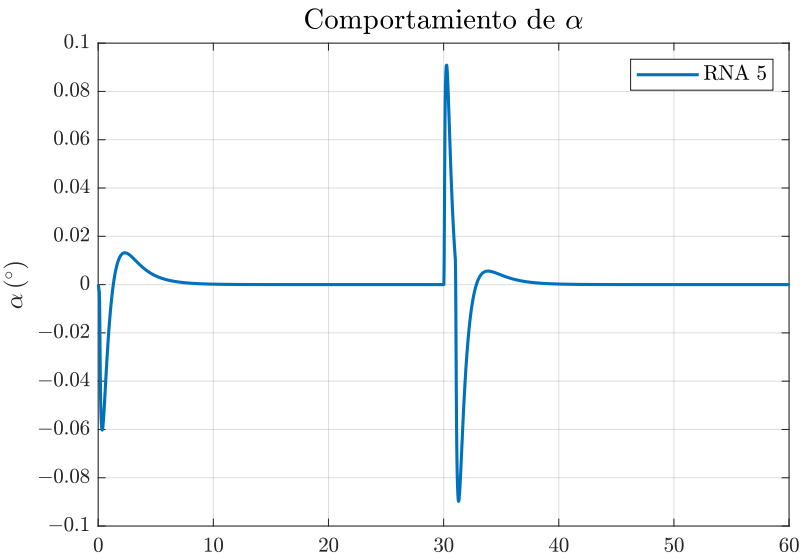


Figura 18: Respuesta del ángulo del eslabón flexible α ante perturbación.

Por último, la señal de control producida por la red C5 se puede ver en la Figura 19. Se percibe que esta responde de forma limitada a la modificación, sin mostrar saturaciones ni interrupciones bruscas. Este comportamiento verifica que la red puede generalizar apropiadamente y no necesita hacer esfuerzos de control desmesurados para evitar perturbaciones.

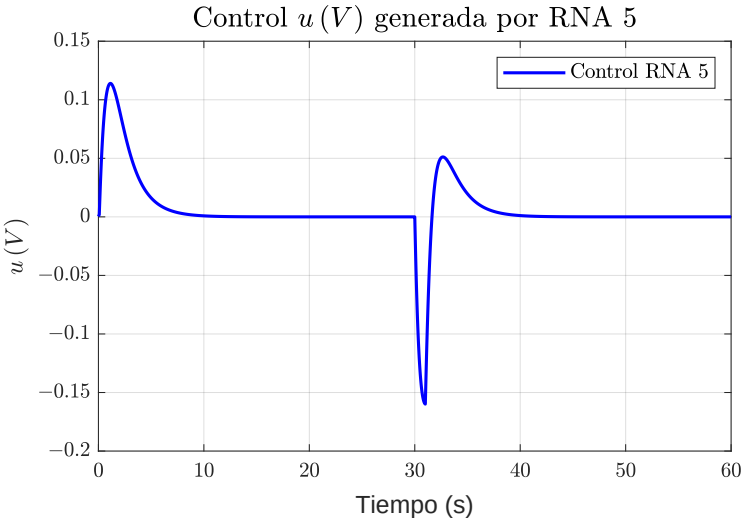


Figura 19: Señal de control $u(k)$ generada por la red neuronal C5 ante perturbación.

En conjunto, estas evidencias experimentales permiten concluir que el sistema controlado mediante la red neuronal C5 presenta un comportamiento estable y robusto, tolerando adecuadamente perturbaciones externas tanto en la variable controlada como en las dinámicas internas del sistema.

3.8.1. Análisis de estabilidad BIBO en tiempo discreto

El análisis de estabilidad de un sistema dinámico controlado por una red neuronal artificial (RNA) puede abordarse mediante técnicas indirectas, dado que las redes neuronales no permiten obtener una expresión explícita del modelo en lazo cerrado. En este contexto, se optó por utilizar herramientas de identificación de sistemas para obtener una función de transferencia que aproxime la dinámica del sistema cuando es controlado por la red neuronal entrenada.

Estabilidad BIBO

Una de las maneras más usadas para analizar la estabilidad de sistemas discretos es a través del criterio BIBO (Bounded Input Bounded Output). Este principio determina que un sistema es BIBO estable si la salida se mantiene acotada frente a cualquier entrada limitada. Para los sistemas lineales invariantes en el tiempo (LTI) de tiempo discreto, es posible llevar a cabo un análisis de esta condición, observando los polos de su función de transferencia en el plano z.

Criterio en el dominio z

Sea un sistema descrito por una función de transferencia discreta de la forma:

$$G(z) = \frac{b_0 + b_1 z^{-1} + \cdots + b_m z^{-m}}{1 + a_1 z^{-1} + \cdots + a_n z^{-n}}, \tag{71}$$

donde b_i y a_i son coeficientes reales. El sistema será BIBO estable si y solo si todos los polos (raíces del denominador) se encuentran estrictamente dentro del círculo unitario, es decir:

$$|p_i| < 1, \quad \forall i. \tag{72}$$

Este criterio es directo y sencillo de aplicar una vez se tiene la función de transferencia identificada.

3.8.2. Identificación del modelo en lazo cerrado

Para llevar a cabo el análisis, se registraron las señales de referencia (θ_{ref}) y salida (θ) del sistema al aplicar un escalón. A partir de estos datos, se

utilizó la herramienta *System Identification Toolbox* de MATLAB para estimar una función de transferencia discreta de segundo orden que represente el comportamiento del sistema en lazo cerrado bajo el control de la RNA.

La función de transferencia identificada es la siguiente:

$$G(z) = \frac{0{,}0002971\, z^{-1}}{1 - 1{,}966\, z^{-1} + 0{,}9667\, z^{-2}} \tag{73}$$

El sistema tiene un tiempo de muestreo de $T_s = 0{,}02$ s. A partir de esta función, se procedió a calcular los polos y sus respectivos módulos, obteniéndose:

- Polos: $p_{1,2} = 0{,}9830 \pm 0{,}0203i$
- Módulos: $|p_1| = |p_2| = 0{,}9832$

Dado que ambos polos tienen módulo menor que 1, se concluye que el sistema en lazo cerrado con la RNA es BIBO estable, ya que todos los polos se encuentran dentro del círculo unitario.

La evaluación matemática indirecta de la estabilidad posibilita validar el rendimiento del controlador neuronal sin que sea imprescindible saber explícitamente cuál es el modelo interno del sistema. La identificación realizada proporciona una representación cercana, aunque suficiente, del comportamiento dinámico, lo que posibilita la aplicación de criterios teóricos bien fundamentados como el de estabilidad BIBO. Esta validación se suma al análisis gráfico expuesto en las partes previas, en el que también se demostró una conducta firme y sólida del sistema frente a perturbaciones.

4. Conclusiones y recomendaciones

En esta sección se presenta las conclusiones generales del trabajo de titulación, destacando los principales logros alcanzados y el cumplimiento de los objetivos planteados. Además, se proponen recomendaciones que podrían orientar investigaciones futuras o mejoras en la implementación del sistema de control basado en redes neuronales artificiales desarrollado en este estudio.

4.1. Conclusiones

El presente trabajo de titulación permitió diseñar, entrenar y validar un sistema de control basado exclusivamente en RNA para un sistema rotatorio con eslabón flexible, empleando un enfoque de modelo inverso. El controlador neuronal desarrollado demostró su capacidad para seguir referencias y reducir las oscilaciones propias de la flexibilidad estructural, sin recurrir a controladores tradicionales adicionales, validando así su viabilidad como única estrategia de control.

Se creó y se validó un modelo dinámico en MATLAB/ Simulink® que representa la interacción entre el eslabón flexible y la base rotatoria. Este modelo, que fue creado a partir de una formulación energética y aplicado en un espacio de estados, logró captar las oscilaciones elásticas del eslabón α y la dinámica rotacional θ del sistema. Esto sentó unas bases sólidas para las fases posteriores del trabajo.

Para el diseño del controlador, se optó por una arquitectura de red neuronal MLP optimizada para producir la señal de control a partir de un conjunto de regresores que contenía datos históricos sobre la posición, la velocidad y las entradas previas. La configuración C5 fue la que alcanzó el mejor desempeño, logrando un balance apropiado entre estabilidad, precisión y habilidad de generalización.

Las pruebas exhaustivas comprendieron en evaluar el sistema ante señales de escalón simple, escalón periódico constante y variable, además de perturbaciones externas, mostrando que el controlador neuronal C5 preservó un seguimiento estable y próximo a la referencia con un 8 % de error en estado estacionario y el mejor tiempo de establecimiento (8 s). Asi mismo el sistema volvió a obtener la referencia incluso cuando se aplicó una perturbación del ± 5 % sobre el valor nominal.

En conjunto, los resultados alcanzados confirman que el enfoque de control por RNA basado en modelo inverso es una solución eficaz y robusta para sistemas rotatorios con eslabones flexibles. No obstante, su éxito depende de una adecuada preparación del dataset y de un diseño cuidadoso de

la arquitectura de la red. Estos hallazgos no solo cumplen con los objetivos planteados, sino que también abren la posibilidad de aplicar este método a otros sistemas dinámicos no lineales en ámbitos académicos e industriales.

4.2. Recomendaciones

A partir de los resultados obtenidos y las lecciones aprendidas durante el desarrollo de esta investigación, se proponen las siguientes recomendaciones para trabajos futuros:

- Explorar la implementación de controladores híbridos, donde una RNA de modelo inverso trabaje en conjunto con un esquema de control tradicional (como LQR, PID adaptativo o control por modos deslizantes), para aumentar la robustez ante condiciones más adversas.
- Implementar un esquema de aprendizaje en línea (online learning) que permita a la RNA actualizar sus pesos dinámica mente frente a cambios en la planta o a la aparición de incertidumbres no consideradas durante el entrenamiento.
- Tener una selección adecuada de la calidad y el orden del conjunto de datos durante el entrenamiento, las muestras deben de ser variadas y no redundantes para un correcto aprendizaje.
- Comprobar su capacidad de generalización fuera del entorno simulado, poniendo a prueba el sistema en un ambiente experimental real con sensores y actuadores físicos.
- Analizar mas arquitecturas de redes neuronales (como las LSTM o las convolucionales) u otros métodos de aprendizaje por refuerzo, que podrían mejorar aún más la capacidad para controlar y adaptarse.

Este proyecto de titulación representa un progreso importante para la automatización y el control inteligente de los sistemas mecánicos flexibles. Los resultados obtenidos son alentadores y posibilitan que se lleven a cabo estudios venideros en el área del control neuronal avanzado aplicado a sistemas reales.

Referencias

[1] P. Akella, E. Hemingway, and O. M. O'Reilly, "A Visualization Tool for the Vibration of Euler–Bernoulli and Timoshenko Beams," *The Mathematica Journal*, vol. 19, no. 1, pp. 49–59, 2017.

[2] C. A. Saldaña Enderica, J. R. Llata, and C. Torre-Ferrero, "Optimization of Q and R Matrices with Genetic Algorithms to Reduce Oscillations in a Rotary Flexible Link System," *Robotics*, vol. 13, no. 6, p. 84, 2024, doi: 10.3390/robotics13060084.

[3] C. Saldaña Enderica, J. R. Llata, and C. Torre-Ferrero, "Guided Reinforcement Learning with Twin Delayed Deep Deterministic Policy Gradient for a Rotary Flexible-Link System," *Robotics*, vol. 14, no. 6, art. 76, 2025. doi: 10.3390/robotics14060076.

[4] J. Yang, Y. Xu, and M. Chen, "Model Reduction of Flexible Manipulators," Carnegie Mellon University, Pittsburgh, PA, USA, Tech. Rep. CMU-RI-TR-92-08, June 1992.

[5] D. Subedi, I. Tyapin, and G. Hovland, "Review on Modeling and Control of Flexible Link Manipulators," *Modeling, Identification and Control*, vol. 41, pp. 141–163, 2020, doi: 10.4173/MIC.2020.3.2.

[6] H. H. A. Talebi, R. V. Patel, and K. Khorasani, *Control of Flexible-Link Manipulators Using Neural Networks*. London, U.K.: Springer, 2001. doi: 10.1007/BFb0110411.

[7] E. Alandoli, M. Z. A. Rashid, and M. Sulaiman, "A comparison of PID and LQR controllers for position tracking and vibration suppression of flexible link manipulator," *Journal of Theoretical and Applied Information Technology*, vol. 95, pp. 2949–2955, July 2017.

[8] G. Lima, D. Porto, A. Oliveira, and W. Moreira Bessa, "Intelligent control of a single-link flexible manipulator using sliding modes and artificial neural networks," *Electronics Letters*, vol. 57, 2021, doi: 10.1049/ell2.12300.

[9] C. Pan, H. Huang, C. Chen, L. Wu, and J. Xiao, "Neuro-adaptive command-filtered backstepping control of uncertain flexible-joint manipulators with input saturation via output," 2025, doi: 10.21203/rs.3.rs-5889236/v1.

[10] M. Sasaki, M. Takeda, J. Muguro, and W. Njeri, "Trajectory Control of Flexible Manipulators Using Forward and Inverse Models with Neural Networks," *Vibration*, vol. 8, no. 3, p. 48, 2025, doi: 10.3390/vibration8030048.

[11] H. Gao, W. He, C. Zhou, and C. Sun, "Neural Network Control of a Two-Link Flexible Robotic Manipulator Using Assumed Mode Method," *IEEE Transactions on Industrial Informatics*, vol. PP, pp. 1–1, Mar. 2018, doi: 10.1109/TII.2018.2818120.

[12] Z. Su and K. Khorasani, "A neural-network-based controller for a single-link flexible manipulator using the inverse dynamics approach," *IEEE Transactions on Industrial Electronics*, vol. 48, pp. 1074–1086, Jan. 2002, doi: 10.1109/41.969386.

[13] A. Jamali, I. Mat Darus, P. Samin, and M. Tokhi, "Intelligent modeling of double link flexible robotic manipulator using artificial neural network," *Journal of Vibroengineering*, vol. 20, pp. 1021–1034, Mar. 2018, doi: 10.21595/jve.2017.18575.

[14] M. Azeez, A. Abdelhaleem, S. Elnaggar, K. Moustafa, and K. Atia, "Optimized sliding mode controller for trajectory tracking of flexible joints three-link manipulator with noise in input and output," *Scientific Reports*, vol. 13, Aug. 2023, doi: 10.1038/s41598-023-38855-7.

[15] S. Li, L. Meng, K. Fang, and F. Liu, "Neural Network Adaptive Inverse Control of Flexible Joint Space Manipulator Considering the Influence of Gravity," *Sensors*, vol. 24, no. 21, p. 6942, 2024, doi: 10.3390/s24216942.

[16] A. Zhang, Z. Lin, B. Wang, and Z. Han, "Nonlinear Model Predictive Control of Single-Link Flexible-Joint Robot Using Recurrent Neural Network and Differential Evolution Optimization," *Electronics*, vol. 10, no. 19, p. 2426, 2021, doi: 10.3390/electronics10192426.

[17] J. M. Araújo, J. Bettega, N. J. B. Dantas, C. E. T. Dórea, D. Richiedei, and I. Tamellin, "Vibration Control of a Two-Link Flexible Robot Arm with Time Delay through the Robust Receptance Method," *Applied Sciences*, vol. 11, no. 21, p. 9907, 2021, doi: 10.3390/app11219907.

[18] S. S. Rao, *Mechanical Vibrations*. Upper Saddle River, NJ, USA: Pearson Prentice Hall, 2004.

[19] T. S. Lee and E. A. Alandoli, "A critical review of modelling methods for flexible and rigid link manipulators," *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 42, p. 508, 2020, doi: 10.1007/s40430-020-02602-0.

[20] D. C. Karnopp, D. L. Margolis, y R. C. Rosenberg, *System dynamics: modeling, simulation, and control of mechatronic systems*. John Wiley & Sons, 2012.

[21] A. Wills, D. Bates, A. Fleming, B. Ninness, and S. Moheimani, "Model Predictive Control Applied to Constraint Handling in Active Noise and Vibration Control," *IEEE Transactions on Control Systems Technology*, vol. 16, pp. 3–12, Feb. 2008, doi: 10.1109/TCST.2007.903062.

[22] E. Arnáez Braschi, *Enfoque práctico del control moderno. Con aplicaciones en Matlab*, Cap. 1, Universidad Peruana de Ciencias Aplicadas (UPC), 2005. [En línea]. Disponible en: http://hdl.handle.net/10757/345716. doi: 10.19083/978-612-4191-28-2.

[23] C. Pan, J. Chen, Y. Yu, X. Liu, y J. Zhang, "Dynamic modeling and vibration analysis of a flexible manipulator with a rotating base using assumed mode method," *Applied Sciences*, vol. 12, no. 13, p. 6496, 2022. doi: 10.3390/app12136496.

[24] S. P. Timoshenko, *Theory of Elastic Stability*, 2nd ed. New York, NY, USA: McGraw-Hill, 1961.

[25] W. Weaver, S. P. Timoshenko, and D. H. Young, *Vibration Problems in Engineering*, 5th ed. New York, NY, USA: Wiley, 1990.

[26] L. Meirovitch, *Analytical Methods in Vibrations*. New York, NY, USA: Macmillan, 1967.

[27] K. J. Bathe, *Finite Element Procedures*. Upper Saddle River, NJ, USA: Prentice Hall, 1996.

[28] S. S. Rao, *Mechanical Vibrations*, 6th ed. Upper Saddle River, NJ, USA: Pearson, 2017.

[29] R. D. Blevins, *Formulas for Natural Frequency and Mode Shape*, 2nd ed. Malabar, FL, USA: Krieger, 2001.

[30] L. Meirovitch, *Methods of Analytical Dynamics*. New York, NY, USA: Dover, 2010.

[31] S. S. Rao, *Vibration of Continuous Systems*. Hoboken, NJ, USA: John Wiley & Sons, 2007.

[32] M. Benosman and G. Le Vey, "Control of flexible manipulators: A survey," *Robotica*, vol. 22, no. 5, pp. 533–545, 2004.

[33] S. P. Timoshenko and J. N. Goodier, *Theory of Elasticity*, 3rd ed. New York, NY, USA: McGraw-Hill, 1970.

[34] M. W. Spong, "Modeling and control of elastic joint robots," *Journal of Dynamic Systems, Measurement, and Control*, vol. 109, no. 4, pp. 310–319, Dec. 1987.

[35] B. Siciliano and O. Khatib (eds.), *Springer Handbook of Robotics*, 2nd ed. Cham, Switzerland: Springer, 2016.

[36] S. K. Dwivedy and P. Eberhard, "Dynamic analysis of flexible manipulators, a literature review," *Mechanism and Machine Theory*, vol. 41, no. 7, pp. 749–777, July 2006.

[37] C. L. Phillips and R. D. Harbor, *Feedback Control Systems*, 6th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2010.

[38] R. Krishnan, *Electric Motor Drives: Modeling, Analysis, and Control*. Upper Saddle River, NJ, USA: Prentice Hall, 2001.

[39] K. Ogata, *Modern Control Engineering*, 5th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2010.

[40] S. J. Russell, P. Norvig, and E. Davis, *Artificial Intelligence: A Modern Approach*, 3rd ed., Prentice Hall Series in Artificial Intelligence. Upper Saddle River, NJ: Prentice Hall, 2010. [Online]. Available: `https://books.google.com.ec/books?id=8jZBksh-bUMC`

[41] S. Haykin, *Neural Networks and Learning Machines*, 3rd ed., Pearson Education, 2009.

[42] K. S. Narendra and K. Parthasarathy, "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990. doi: 10.1109/72.80202.

[43] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[44] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[45] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1, D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA: MIT Press, 1986, pp. 318–362.

[46] K.-I. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, vol. 2, no. 3, pp. 183–192, 1989. doi: [10.1016/0893-6080(89)90003-8](https://doi.org/10.1016/0893-6080(89)90003-8).

[47] A. Palmer, J. Montaño, and R. Jiménez, "Tutorial sobre Redes Neuronales Artificiales: El Perceptrón Multicapa," *[Revista]*, vol. 5, Jul. 2001.

[48] A. Atehortúa, "Control neuronal por modelo inverso aplicado a sistemas dinámicos," *Revista Colombiana de Control*, vol. 15, no. 2, pp. 45–54, 2017.

[49] V. A. Rodríguez-Toro, J. E. Garzón, and J. A. López, "Control neuronal por modelo inverso de un servosistema usando algoritmos de aprendizaje Levenberg–Marquardt y Bayesiano," *arXiv preprint arXiv:1106.2744*, 2011. [Online]. Available: https://arxiv.org/abs/1106.2744

[50] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990. doi: 10.1109/72.80202

[51] K. J. Hunt, D. Sbarbaro, R. Zbikowski, and P. J. Gawthrop, "Neural networks for control systems—A survey," *Automatica*, vol. 28, no. 6, pp. 1083–1112, 1992. doi: 10.1016/0005-1098(92)90053-I

[52] M. Nørgaard, O. Ravn, N. K. Poulsen, and L. K. Hansen, *Neural Networks for Modelling and Control of Dynamic Systems: A Practitioner's Handbook*. London, UK: Springer, 2000.

[53] M. T. Hagan, H. B. Demuth, and M. H. Beale, "Neural network training issues: overfitting and generalization," in *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, 1996, pp. 127–132.

[54] O. Nelles, *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*. Berlin, Germany: Springer, 2001.

[55] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 989–993, 1994. doi: 10.1109/72.329697

[56] F. D. Foresee and M. T. Hagan, "Gauss-Newton approximation to Bayesian learning," in *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, 1997, pp. 1930–1935. doi: 10.1109/IJCNN.1997.614194

[57] M. T. Hagan, H. B. Demuth, M. H. Beale, and O. De Jesús, *Neural Network Design*, 2nd ed. Stillwater, OK, USA: Martin Hagan, 2014. [Online]. Available: https://hagan.okstate.edu/nnd.html

[58] N. Siddique and H. Adeli, *Computational Intelligence: Synergies of Fuzzy Logic, Neural Networks and Evolutionary Computing*. Cambridge, MA, USA: MIT Press, 2013.

[59] A. A. Rubio and G. V. Raffo, *Manual de Simulink para la asignatura de Teoría de Sistemas*.

[60] The MathWorks, Inc., *Deep Learning Toolbox, Version R2024a*, Natick, MA, USA: The MathWorks, Inc., 2024. [Online]. Available: `https://www.mathworks.com/products/deep-learning.html`

Anexos

A. Simulación en Lazo Abierto

```
% Script para la Simulación en Lazo Abierto de un Sistema Flexible
% Este script define los parámetros del sistema y realiza una simulación
% en lazo abierto con entradas aleatorias para generar un dataset.

clear; close all; clc;
rng(1);

%% 1. Parámetros y Modelo del Sistema
L      = 0.45;
ml     = 0.0008;
Jlink  = 0.0042;
mb     = 0.05;
Jb     = mb * L^2;
Jload  = Jlink + Jb;
Jbase  = 0.002;
R      = 2.6;
Kg     = 70;
Kt     = 0.00767;
Kb     = 0.00767;
Km     = 0.00767;
fc     = 1.9;
Kstiff = (2*pi*fc)^2 * Jload;

A = [ 0, 0, 1, 0; 0, 0, 0, 1; 0, Kstiff/(Jbase*L), -(Km*Kg)^2/(R*Jbase),
    0; ...
      0, -Kstiff*(Jload+Jbase)/(L*Jbase*Jload), (Km*Kg)^2/(R*Jbase), 0];
B = [ 0; 0; (Km*Kg)/(R*Jbase); -(Km*Kg)/(R*Jbase)];
C = eye(4);
D = zeros(4,1);

%% 2. Parámetros de Simulación y Entradas
dt   = 0.02;
tfin = 1000;
t    = 0:dt:tfin;
N    = numel(t);
u    = zeros(N,1);
tt_u = 0;
while tt_u < tfin
    val = 1.5 * (2*rand - 1);
    u(t>=tt_u & t<tt_u+2) = val;
    tt_u = tt_u + 2;
end
rng(99);
theta_ref = zeros(N,1);
Tesc_ref = 10;
num_escalones_ref = ceil(tfin/Tesc_ref);
valores_ref = zeros(num_escalones_ref,1);
amp = 1.5;
for k = 1:min(6, num_escalones_ref)
    valores_ref(k) = rand*amp;
end
for k = 7:num_escalones_ref
    nuevo = 0.7*valores_ref(k-1) + 0.3*amp*(2*rand - 1);
    valores_ref(k) = max(min(nuevo, amp), -amp);
end
```

```
tt_ref = 0;
for k = 1:num_escalones_ref
    theta_ref(t >= tt_ref & t < tt_ref + Tesc_ref) = valores_ref(k);
    tt_ref = tt_ref + Tesc_ref;
end

%% 3. Simulación y Gráficas
x = zeros(4,1);
Y = zeros(N,4);
for k = 1:N
    Y(k,:) = (C*x + D*u(k)).';
    dx = A*x + B*u(k);
    x  = x + dt*dx;
end

f1 = figure('Color','w');
subplot(3,1,1); plot(t, u, 'LineWidth', 1.5);
grid on; ylabel('u(t) [V]'); title('Entrada');
subplot(3,1,2); plot(t, Y(:,1), 'LineWidth', 1.5);
grid on; ylabel('\theta [rad]'); title('Salida \theta');
subplot(3,1,3); plot(t, Y(:,2), 'LineWidth', 1.5);
grid on; ylabel('\alpha [rad]'); xlabel('t [s]'); title('Salida \alpha');

%% 4. Guardado de Datos
U = u(:);
Yout = Y;
save('dataset_simulacion_lazo_abierto.mat', 'U', 'Yout', 'theta_ref', 't')
    ;
```

Listing 1: Script de simulación en lazo abierto para la generación del dataset

B. Generación del Dataset

```
%% Script para la Generación del Dataset de la Red Neuronal
% Este script carga los datos de la simulación y los procesa para crear un
% dataset estructurado (X, Tgt) para el entrenamiento de una red neuronal
% de control inverso.

% --- Inicialización ---
clear; close all; clc;

% 1) Carga de Datos
load('datos_ajustados_lazo_abi_con_ref.mat');

% 2) Configuración y Pre-asignación
N = length(U);
nmax = 4;
Ndata = N - nmax;
X = zeros(11, Ndata);
Tgt = zeros(1, Ndata);

% 3) Construcción del Dataset
for i = 1:Ndata
    k = i + nmax;
    % Entradas (historial): theta, alpha, dtheta, dalpha, u
    X(1, i) = Yout(k, 1);
    X(2, i) = Yout(k-1, 1);
```

```
    X(3, i) = Yout(k-2, 1);
    X(4, i) = Yout(k-1, 2);
    X(5, i) = Yout(k-2, 2);
    X(6, i) = Yout(k-1, 3);
    X(7, i) = Yout(k-2, 3);
    X(8, i) = Yout(k-1, 4);
    X(9, i) = U(k-1);
    X(10, i) = U(k-2);
    X(11, i) = U(k-3);
    % Salida (objetivo)
    Tgt(1, i) = U(k);
end

% 4) Visualización y Guardado
figure('Color','w');
subplot(2,1,1); plot(Tgt, 'LineWidth', 1.5);
title('Señal Objetivo (u(k))'); ylabel('Voltaje [V]'); grid on;
subplot(2,1,2); plot(X', 'LineWidth', 1.5);
title('Entradas al Modelo Inverso'); ylabel('Valores'); xlabel('Muestra');
     grid on;
save('dataset_inverso_MLP.mat', 'X', 'Tgt');
```

Listing 2: Script para la preparación del dataset para el modelo de control inverso

C. Entrenamiento de la Red Neuronal

```
%% Script para el Entrenamiento de una Red Neuronal para Control Inverso
% Este script entrena una red neuronal MLP como un modelo de control
% inverso y la guarda para su posterior uso en lazo cerrado.

% --- Inicialización ---
clear; close all; clc;

% 1) Carga del Dataset
load('dataset_inverso_MLP_con_orden_y_historial.mat');

% 2) Configuración de la Red
hiddenLayers = [20 10];
net = feedforwardnet(hiddenLayers, 'trainlm');
net.layers{1}.transferFcn = 'logsig';
net.layers{2}.transferFcn = 'logsig';
net.layers{3}.transferFcn = 'purelin';
net.inputs{1}.processFcns  = {'mapminmax'};
net.outputs{2}.processFcns = {'mapminmax'};

% División de datos
net.divideFcn = 'dividerand';
net.divideParam.trainRatio = 0.85;
net.divideParam.valRatio   = 0.10;
net.divideParam.testRatio  = 0.05;

% Parámetros de entrenamiento
net.trainParam.epochs     = 600;
net.trainParam.goal       = 1e-4;
net.trainParam.min_grad   = 1e-6;
net.trainParam.showWindow = true;
```

```
% 3) Entrenamiento de la Red
[net, tr] = train(net, X, Tgt);

% 4) Visualización y Guardado
figure('Color', 'w');
plot(tr.epoch, tr.perf, 'b', 'LineWidth', 1.5); hold on;
plot(tr.epoch, tr.vperf, 'r--', 'LineWidth', 1.5);
legend('Entrenamiento','Validación');
xlabel('Épocas'); ylabel('Error cuadrático medio (MSE)');
title('Evolución del error durante el entrenamiento');
grid on;
save('red_inversa_MLP_con_orden_y_historial.mat', 'net', 'tr');
```

Listing 3: Script para el entrenamiento de la red neuronal de control inverso

D. Control en Lazo Cerrado

```
%% Script para la Simulación en Lazo Cerrado con Red Neuronal
% Este script simula el sistema en lazo cerrado usando una red neuronal
% pre-entrenada para seguir una trayectoria de referencia.

% --- Inicialización ---
clear; close all; clc;

% 1) Carga de la Red
load('red_inversa_MLP_con_orden_y_historial.mat');

% 2) Parámetros y Modelo del Sistema
L = 0.45; ml = 0.0008; Jlink = 0.0042;
mb = 0.05; Jb = mb*L^2; Jload = Jlink+Jb;
Jbase = 0.002; R = 2.6; Kg = 70;
Kt = 0.00767; Kb = 0.00767; Km = 0.00767;
fc = 1.9; Kstiff = (2*pi*fc)^2 * Jload;
A = [0, 0, 1, 0; 0, 0, 0, 1; 0, Kstiff/(Jbase*L), -(Km*Kg)^2/(R*Jbase), 0;
     ...
     0, -Kstiff*(Jload+Jbase)/(L*Jbase*Jload), (Km*Kg)^2/(R*Jbase), 0];
B = [0; 0; (Km*Kg)/(R*Jbase); -(Km*Kg)/(R*Jbase)];
C = [1 0 0 0; 0 1 0 0];
D = zeros(2,1);

% 3) Simulación y Referencia
dt = 0.02; tfin = 300; t = 0:dt:tfin; N = numel(t);
rng(99);
theta_ref = zeros(N,1);
Tesc = 20; amp = 1.5; num_escalones = ceil(tfin/Tesc);
valores = zeros(num_escalones,1);
fase_inicial = 6;
for k = 1:min(fase_inicial, num_escalones)
    valores(k) = rand*amp;
end
for k = fase_inicial+1:num_escalones
    nuevo = 0.7*valores(k-1) + 0.3*amp*(2*rand-1);
    valores(k) = max(min(nuevo, amp), -amp);
end
tt = 0;
for k = 1:num_escalones
```

```
    theta_ref(t >= tt & t < tt+Tesc) = valores(k);
    tt = tt + Tesc;
end

% 4) Inicialización y Bucle de Simulación
x = zeros(4,1); Y = zeros(N,2); U = zeros(N,1);
u_hist = zeros(3,1); theta_hist = zeros(3,1); dtheta_hist = zeros(2,1);
alpha_hist = zeros(2,1); dalpha_hist = 0;
for k = 4:N
    entradaRNA = [
        theta_ref(k); theta_hist(1); theta_hist(2); alpha_hist(1);
        alpha_hist(2); dtheta_hist(1); dtheta_hist(2); dalpha_hist;
        u_hist(1); u_hist(2); u_hist(3)
    ];
    u = net(entradaRNA);
    u = max(min(u, 5), -5);
    dx = A*x + B*u;
    x = x + dt*dx;
    y = C*x + D*u;
    Y(k,:) = y.'; U(k) = u;
    theta_hist = [y(1); theta_hist(1:2)];
    dtheta_hist = [x(3); dtheta_hist(1)];
    alpha_hist = [y(2); alpha_hist(1)];
    dalpha_hist = x(4);
    u_hist = [u; u_hist(1:2)];
end

% 5) Visualización de Resultados
figure('Color','w');
subplot(3,1,1); plot(t, theta_ref, 'k--', 'LineWidth',1.5); hold on;
plot(t, Y(:,1), 'b', 'LineWidth',1.5);
ylabel('\theta [rad]'); title('Referencia vs \theta');
legend('\theta_{ref}', '\theta'); grid on;
subplot(3,1,2); plot(t, Y(:,2), 'r', 'LineWidth',1.5);
ylabel('\alpha [rad]'); title('Ángulo del eslabón \alpha');
grid on;
subplot(3,1,3); plot(t, U, 'm', 'LineWidth',1.5);
xlabel('t [s]'); ylabel('u [V]');
title('Señal de control u(k)'); grid on;
```

Listing 4: Script de simulación de control en lazo cerrado con red neuronal