ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DESARROLLO DE UNA HERRAMIENTA COMPUTACIONAL PARA IDENTIFICACIÓN DE MODELOS LINEALES MEDIANTE OPTIMIZACIÓN DE PARÁMETROS UTILIZANDO COLONIA DE HORMIGAS Y ENJAMBRE DE PARTÍCULAS

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y CONTROL

JARA LIVISACA JESSICA LIZETTE

jessica.jara@epn.edu.ec

DIRECTOR: DR. ING. GEOVANNY DANILO CHÁVEZ GARCÍA

danilo.chavez@epn.edu.ec

Quito, noviembre 2019

AVAL

Certifico que el presente trabajo fue desarrollado por Jessica Lizette Jara Livisaca, bajo mi supervisión.

Dr.- Ing. Geovanny Danilo Chávez García
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo Jessica Lizette Jara Livisaca, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración dejo constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

JESSICA LIZETTE JARA LIVISACA

DEDICATORIA

A Dios, por regalarme la vida y sus bendiciones.

A mis padres, Fernanda y Yimer, en gratitud por el apoyo y la comprensión que me brindaron durante todos estos años.

A mis hermanos, Jesús y Jair, ustedes tienen todo para hacerlo mejor, los quiero mucho.

A mis tías Enid, Irene y a mis tíos Holger, Rodrigo por velar por mí siempre.

Y a todas las mujeres que deciden estudiar ingeniería.

Con cariño, Jessica.

AGRADECIMIENTO

A Dios, por ponerme en el lugar y momento preciso.

A mis padres Fernanda, Yimer y Margarita, por darme su cariño incondicionalmente.

A mis hermanitos Jesús y Jair por el amor sincero; ustedes son mi modelo, me enseñan mucho en cada día. También a María y Ángelo gracias por su ternura.

A mis abuelitos, por su amor infinito, sus cuidados y palabras de aliento en cada momento.

A todos mis familiares que siempre confiaron en mí y me dieron ánimo para seguir adelante. Este logro es de todos ustedes tías, tíos, primos y primas.

A mis amigas Natalia, Yazmin, Mayra, Sofía, Dayana, Talía, Katherine; gracias por los gratos momentos compartidos, por las palabras de aliento y por estar incondicionalmente. La vida nos podrá separar, pero sus recuerdos siempre permanecerán conmigo.

A la Escuela Politécnica Nacional por abrirme las puertas de su institución; a su personal docente y administrativo por su buen servicio y por formarme como profesional. En especial a la Sra. Wilmita Guerreo por su bondad y ayuda permanente más en épocas de matrículas.

Al Dr. Ing. Danilo Chávez y PhD. Ing. Oscar Camacho por el apoyo en la realización de este proyecto de titulación y por la acogida en el Laboratorio de Robótica y Sistemas Inteligentes. Al Dr. Ing. Marcelo Pozo por su apoyo incondicional durante la carrera.

A mis compañeros y luego amigos del laboratorio, gracias por brindarme su apoyo y su amistad, las risas y sus consejos; en especial a Kleber y Paul gracias por ser unas excelentes personas y brindar siempre el apoyo necesario.

Finalmente, gracias a todas aquellas personas que a lo largo de mi carrera universitaria me ayudaron y confiaron en mí. Infinitamente gracias, sin ustedes tampoco lo hubiese logrado. En forma especial a C.Pe.

Jessica. L. Jara L.

ÍNDICE DE CONTENIDO

RESUMEN

En el presente trabajo de titulación se describe el diseño, desarrollo e implementación de una herramienta computacional para la estimación de parámetros característicos e identificación de sistemas lineales de Primer Orden, Primer Orden con Retardo, Segundo Orden y Respuesta Inversa mediante los algoritmos de optimización por Colonia de Hormigas y Enjambre de Partículas, con el único conocimiento de la señal de entrada, de salida y el tiempo respuesta del sistema a identificar. La herramienta tiene el nombre de SysID, está desarrollada en MatLab® y su interfaz gráfica en App Designer®.

El uso de los algoritmos de optimización se justifica en encontrar la mejor aproximación a la estimación de cada parámetro característico de los tipos de modelos lineales antes mencionados, evitando la realización algebraica de una identificación paramétrica por métodos convencionales como curva de reacción o modelamiento matemático, esto se logra por la minimización de la función de costo, que para este proyecto de titulación es el índice de error cuadrático medio. Para identificar el tipo de sistema se realiza una identificación previa, en la cual el menor valor del índice de error cuadrático integral determina el modelo de sistema, mientras que los algoritmos de optimización realizan una estimación óptima de los parámetros característicos de este.

SysID incorpora una interfaz para la adquisición de datos por comunicación serial, y permite el ingreso de simulaciones de plantas no lineales, en un punto de operación, desde archivos MatLab-Simulink®, para la estimación de parámetros e identificación del modelo lineal del sistema. Se muestra como resultados el modelo de sistema identificado en función de transferencia, el mínimo error de identificación, el tiempo transcurrido para la identificación, las gráficas de la identificación: el sistema referencia vs. el sistema identificado y la gráfica de evolución del error por iteración en el proceso de identificación.

Las pruebas realizadas a SysID son de carácter simulado, pero se incluye la identificación del tipo de sistema a un motor dc y se realiza un análisis comparativo entre la función *IDENT* propia del software de MatLab® y la herramienta computacional desarrollada.

PALABRAS CLAVE: Algoritmos de optimización, PSO, ACO, ACO-R, sistemas lineales, identificación de sistemas, estimación de parámetros.

ABSTRACT

This project is about the design, development and implementation of a computational tool for parametric estimation and system model identification of First Order, First Order Plus Dead Time, Second Order and Inverse Response Linear Systems through Ant Colony and Particle Swarm optimization algorithms. The name of the designed computational tool is SysID. MatLab® and App Designer® software were used for the design and development of its graphic user interface.

Optimization algorithms are used to find accurate approaches for each characteristic parameters of linear models. So that an algebraic calculus for parametric identification through conventional methods as reaction curve or mathematical modeling can be avoid. This identification is achieved by minimizing a cost function, which is the integral mean square error index. The lowest value of the squared integral error index defines the model of the linear system, and the optimization algorithms estimate the optimal characteristic parameters values.

SysID incorporates a data acquisition interface using serial communication, and it allows to upload no lineal simulated plants from MatLab-Simulink® files for parameter estimation and system model identification. The results show the identified system is shown as a transfer function, the minimum value of error identification, the time elapsed for identification, the identification graphs: reference system vs. identified system and the identification error evolution graph.

The tests performed on SysID are simulated, but the identification of a dc motor is included with a comparative analysis between *IDENT* function of the MatLab® software and the developed computational tool.

KEYWORDS: Optimization algorithms, PSO, ACO, ACO-R, linear systems, system identification, parametric estimation.

1. INTRODUCCIÓN

En la actualidad la evolución de la tecnología y la automatización de los procesos industriales han construido un mercado con índices de competitividad y eficiencia elevados; esto gracias a la implementación de técnicas de control, que reducen costos y desperdicios de fabricación. Los cuales están elaborados en base a un conocimiento íntegro del comportamiento dinámico de los procesos: sus partes críticas, de su reacción a cambios de referencia y a perturbaciones.

Entonces, disponer de un modelo matemático que describa adecuadamente el proceso a controlar, es una ventaja no solo para el diseño de algoritmos de control que permitan mejorar los procesos, sino también, para los métodos de diagnóstico: supervisión y detección de fallas producto del desgaste o averías en los equipos.

El fin que persigue la identificación de sistemas es la obtención de modelos matemáticos a partir de las mediciones experimentales de: entradas o variables de control, salida o variables controladas, tiempo de simulación y perturbaciones, sin necesidad que se conozca el funcionamiento interno o la interacción de cada componente del sistema; limitándose a conocer la relación entre la entrada y salida del proceso [1]; es decir el comportamiento de la salida a un cambio de referencia en la entrada, esto se conoce como dinámica del sistema, aspecto importante para la identificación.

En consecuencia, contar con una herramienta computacional que permita encontrar los parámetros característicos a modelos de sistemas tipo caja gris, bajo los beneficios de la optimización basados en sistemas emergentes bio-inspirados representa una gran ventaja en comparación con los métodos convencionales, porque se consigue disminuir el tiempo invertido en la obtención del modelo característico de la planta y garantiza que los resultados obtenidos son lo más cercanos a los óptimos; se limita a trabajar solo con los valores de entrada (voltaje, corriente, concentración, caudal, etc.), de salida (voltaje, corriente, concentración, caudal, etc.) y el tiempo de simulación necesario; excluyendo la necesidad de una interpretación física, que en algunos casos puede ser difícil por el conjunto de leyes físicas utilizadas para describir el comportamiento. La adquisición de estos datos servirá para encontrar los parámetros característicos (Ganancia, K; Constante de tiempo, τ; Tiempo de retardo, t_0; Coeficiente amortiguamiento relativo, ξ, Frecuencia Natural, ω_n) mediante los algoritmos de optimización: Colonia de Hormigas (Ant Colony Optimization- ACO) y Enjambre de Partículas (Particle Swarm Optimization- PSO).

1.1 OBJETIVOS

El objetivo general de este proyecto de titulación es desarrollar una herramienta computacional para identificación de modelos lineales mediante optimización de parámetros utilizando colonia de hormigas y enjambre de partículas.

Los objetivos específicos de este proyecto de titulación son:

- Realizar una recopilación bibliográfica de los métodos de optimización: PSO y ACO.
- Implementar y simular los algoritmos de los métodos de optimización: PSO y ACO.
- Diseñar e implementar una herramienta computacional para la identificación de sistemas basado en PSO y ACO para sistemas lineales.
- Diseñar una interfaz gráfica para uso y demostración de la herramienta computacional desarrollada.
- Realizar una prueba experimental de identificación de parámetros a un motor dc seleccionando un método adecuado de adquisición de datos, y pruebas simuladas, y se compara los resultados obtenidos con la herramienta IDENT del software MatLab.

1.2 ALCANCE

Se realizará una recopilación bibliográfica de los métodos de optimización: PSO y ACO, con objetivo de entender su funcionamiento que se describirá en el marco teórico. Seguidamente se analizará y estudiará los algoritmos de programación y las aplicaciones de los métodos de optimización: PSO y ACO, para desarrollar el algoritmo de programación en el software de MatLab®. Después se analizará la forma adecuada para la adquisición de datos a través de una tarjeta embebida y del software de MatLab®, la misma que tiene como objetivo almacenar los datos de entrada, salida y tiempo de simulación a la planta que se realizará las pruebas de identificación de parámetros.

Se diseñará e implementará una herramienta computacional en el software de MatLab® para la identificación de sistemas lineales hasta de segundo orden y respuesta inversa, con el cual pueda obtenerse los parámetros más cercanos a los óptimos del modelo matemático representativo a una caja gris [2] mediante los algoritmos de optimización PSO y ACO. También, se desarrollará pruebas simuladas para sistemas lineales de: primer orden, primer orden con retardo, segundo orden y de respuesta inversa.

Para finalizar se desarrollará una interfaz gráfica en el software de MatLab® que permitirá escoger el tipo de sistema que se desea aproximar o la mejor aproximación al sistema real. En ambos casos se toma como función de optimización inicial al ISE (Integral Square Error- Índice integral de error cuadrático). Permitiendo el reconocimiento de las características óptimas del sistema sea este de: primer orden con retardo, primer orden sin retardo, segundo orden o respuesta inversa. De igual manera, se desarrollará de manera práctica la adquisición de datos del motor DC, para encontrar su modelo más cercano a su óptimo utilizando la herramienta computacional desarrollada y se simulará la optimización de los tipos de sistemas lineales restantes: segundo orden y respuesta inversa; por último, se comparará los resultados obtenidos con la herramienta computacional desarrollada con los obtenidos bajo la herramienta IDENT propia del software de MatLab®.

1.3 MARCO TEÓRICO

1.3.1 CONCEPTOS FUNDAMENTALES

Para la elaboración del presente trabajo de titulación es necesario realizar una revisión bibliográfica de los conceptos fundamentales que se utilizan a lo largo de su desarrollo, diseño e implementación.

1.3.1.1 Sistema

Se define como la combinación de elementos de diferente índole que interactúan entre sí para lograr un determinado objetivo. Las señales que ingresan tienen el nombre de entradas, estas son conocidas y manipuladas libremente estimulando al sistema; el resultado son señales que describen al objetivo alcanzado, a estas se las denomina salidas. Existen más señales que influyen en la evolución del sistema que son indeterminadas, a estas se las conocen como perturbaciones [3].

Figura 1.1. Sistema con entrada $\boldsymbol{u}$, perturbación $\boldsymbol{w}$ y salida $\boldsymbol{y}$.

La Figura 1.1 representa a un sistema con sus principales definiciones. El concepto de sistema puede referirse desde un motor hasta el enfriamiento de una torre de destilación, sin importar que en el primero conste de un solo elemento y el último sea un conjunto.

1.3.1.2 Modelado de un sistema

El diseño de sistemas de control requiere el modelo matemático del proceso a controlar para predecir su comportamiento futuro. Sin embargo, su obtención puede ser difícil debido a la complejidad del proceso y a su completa o parcialmente desconocida dinámica [4].

En el diseño de controladores, métodos de diagnóstico, detección de fallos, simulación y evaluación de procesos, es necesario el conocimiento de los parámetros característicos que describen a cada sistema de interés, y se espera que sea de forma fiel a la realidad. La señal de control adecuada, la detección de fallas idónea y la simulación exacta de los procesos tienen relación complementaria con el modelo matemático del sistema identificado, porque de este depende el éxito de los anteriores.

Principalmente existen dos formas de obtener el modelo matemático a un proceso:

- **Modelado**

Se obtiene el modelo matemático a partir de la descripción del proceso mediante las leyes fundamentales de la física y química que rigen su comportamiento. Por lo cual se justifica en el uso de ecuaciones diferenciales y de la matemática tradicional. En su mayoría, esta forma de identificación presenta modelos complejos con no linealidades o grandes dimensiones de estados, para solucionar este inconveniente se utiliza la reducción de modelos, pero se necesita un modelo lineal para empezar.

Su principal ventaja es la explicación del comportamiento, en distintos puntos de operación, debido a que las ecuaciones del modelado tienen un amplio rango de validez. Sin embargo, por la complejidad de los modelos obtenidos su utilidad radica en el diseño y simulación de plantas [3], más no para el diseño de sistemas de control.

- **Identificación de sistemas**

A diferencia del anterior, el modelo matemático del sistema se obtiene a partir de los datos experimentales tomados a la salida y entrada del sistema, sin tener en cuenta las leyes internas. Por tanto, los experimentos realizados son importantes para la identificación del modelo. Esta forma de identificación es útil en el diseño de sistemas de control.

En la práctica lo recomendable es utilizar ambos métodos en la identificación del modelo matemático final, así los datos adquiridos brindan gran exactitud en la identificación de los parámetros del modelo, pero el conocimiento de las leyes físicas que influyen en este facilita el proceso de identificación del modelo.

1.3.1.3 Tipos de estructuras en la identificación de sistemas

El enfoque de la identificación de los tipos de modelos de sistemas se realiza en base al conocimiento de la estructura del modelo o de la dinámica del sistema. Por la cual se debe distinguir los tipos de estructuras, estos son:

- *Caja Blanca:* la estructura del modelo está basada en principios físicos básicos (Leyes de Newton, Cinemática, Balance de energía, Balance de masas), los parámetros del modelo se calculan de las mediciones realizadas y tienen una interpretación física como lo describe en [5].
- *Caja Gris:* la estructura del modelo es parcialmente conocida, para la estimación de parámetros se utiliza la reconstrucción del sistema desde los datos adquiridos de las mediciones de las señales de entrada, salida y tiempo de simulación; estos parámetros pueden tener interpretación física [5].

- *Caja Negra:* la estructura del modelo y sus parámetros son completamente desconocidos, solo se los puede estimar por la información adquirida de la entrada-salida del proceso [6].

La Figura 1.2 muestra el esquema básico para la identificación de un modelo de sistema tipo caja gris, mediante un método de identificación basado en la adquisición de información del modelo por mediciones de sus señales de entrada y salida.

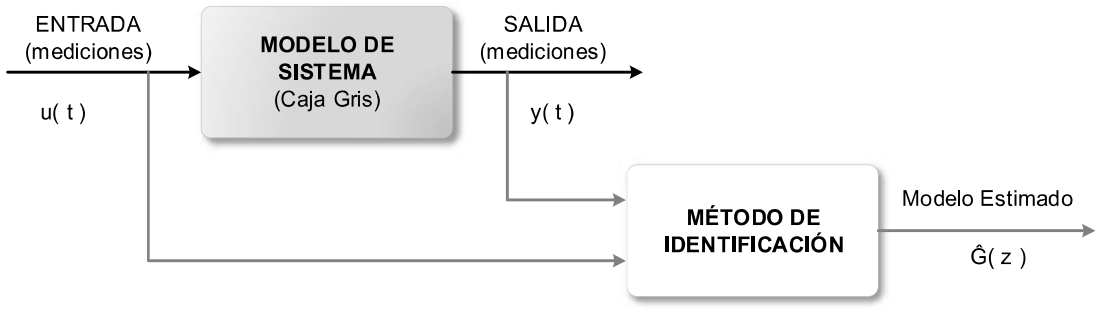


Figura 1.2. Esquema básico de identificación de modelo tipo caja gris.

La mayoría de los procesos industriales son tipo caja negra, no se conoce lo que hay en su interior y la aproximación a su modelo se resuelve por métodos empíricos de identificación. Asimismo, se habla de sistemas tipo caja gris cuando se tiene alguna característica de la planta con la cual se pueda utilizar un método fenomenológico para la identificación de su modelo, a su vez este tipo de sistemas también necesita un método empírico para respaldar el resultado de la identificación denominándose métodos semi-fenomenológicos de identificación de sistemas.

1.3.1.4 Clasificación de los métodos de identificación de sistemas

A lo largo de la historia se ha determinado la siguiente clasificación para la identificación de sistemas [3]:

- Dependiendo del tipo de modelo obtenido:
    - *Métodos no paramétricos,* obtienen modelos no paramétricos del sistema de estudio mediante técnicas como: análisis de la respuesta transitoria, análisis de la respuesta en frecuencia, análisis de correlación, análisis espectral, análisis de Fourier, etc.
    - *Métodos paramétricos,* obtienen modelos paramétricos, para ello elijen una posible estructura del modelo, un criterio de ajuste y realizan la estimación que mejor coincida con el modelo de los datos experimentales.

- Dependiendo de la aplicación:
    - *Métodos de identificación off-line (a posteriori)* se realizan cuando no se requiere un ajuste continuado del modelo y hay la certeza de que la validez de los parámetros obtenidos no se verá alterada con el paso del tiempo.
    - *Métodos de identificación on-line (identificación recursiva)* permiten que los parámetros se actualicen continuamente a partir de nuevos datos de entrada-salida del sistema en evolución.

1.3.1.5 Representaciones paramétricas de la identificación de sistemas

Casi siempre los modelos paramétricos de identificación de sistemas se describen en el domino discreto, porque la información útil para la identificación se obtiene por muestreo. Sin embargo, se puede realizar transformaciones del domino discreto al continuo para cuando se requiera un modelo continuo.

Para la realización de este proyecto se considera la representación paramétrica en función de transferencia, que se define como el cociente entre la transformada de Laplace de la salida (respuesta de la planta) y la transformada de Laplace de la entrada (señal de excitación), cuando las condiciones iniciales son cero [7], como en la Ecuación 1.1.

$$G(s) = \left.\frac{\mathcal{L}[\text{salida}]}{\mathcal{L}[\text{entrada}]}\right|_{\text{condicionaes iniciales cero}} \tag{1.1}$$

Donde:

$\mathcal{L}$: es la transformado de Laplace.

$G(s)$: función de transferencia.

La potencia más alta de s siempre está en el denominador; mientras que en el numerador su potencia debe ser menor o puede ser igual, pero nunca mayor. La Ecuación 1.2 representa una función de transferencia de orden n-ésimo, porque su potencia más alta en s es igual a n.

$$G(s) = \frac{Y(s)}{X(s)} = \frac{b_0 s^m + b_1 s^{m-1} + \cdots + b_{m-1} s + b_m}{a_0 s^n + a_1 s^{n-1} + \cdots + b_{n-1} s + b_n} \tag{1.2}$$

La dinámica de un sistema se puede representar mediante una función de transferencia porque incluye las unidades necesarias para relacionar la entrada con la salida; sin proporcionar más información acerca de la estructura física de este. Sin embargo, favorece el entendimiento de la naturaleza del sistema puesto que se puede estudiar su respuesta

a varias formas de entrada. Una función de transferencia es independiente de la magnitud y naturaleza de la función de excitación o entrada del sistema, y cuando se la desconoce se la puede establecer experimentalmente introduciendo entradas conocidas y relacionándolas con la salida del sistema [7].

1.3.2 METODOLOGÍA DE LA IDENTIFICACIÓN DE SISTEMAS

La identificación de sistemas se considera un arte, que permite construir el modelo matemático de un proceso a partir de su dinámica por medio de la medición de datos, que pueden ser en un número limitado tanto para la entrada como para la salida, en las cuales puede existir ruido a priori la información de la planta [4].

En la industria la mayoría de los procesos son de comportamiento no lineal, o son representados por modelos matemáticos de orden elevado, estos antecedentes dificultan la selección de una técnica de control adecuada y su diseño. En consecuencia, se ha desarrollado técnicas de aproximación a modelos lineales para procesos no lineales, por mencionar algunos, encontramos a: el método empírico o curva de reacción, el método de identificación que se indica en [8] y el método de sintonización de Ziegler -Nichols.

Si bien con los métodos anteriores se realiza una identificación paramétrica adecuada, su resultado depende del criterio de precisión y exactitud del practicante, por tanto, es beneficioso contar con un método para la estimación de modelos matemáticos de sistemas en el cual no intervengan estos factores. En la obtención de modelos paramétricos la señal más simple para el análisis de la respuesta de los sistemas es la función escalón o paso, la cual ha sido considera para el presente trabajo de titulación.

Una tarea ligada a la identificación de sistemas es la estimación de parámetros, que se define como la determinación experimental del valor de los parámetros que gobiernan el comportamiento dinámico del proceso, asumiendo que se conoce su estructura [9].

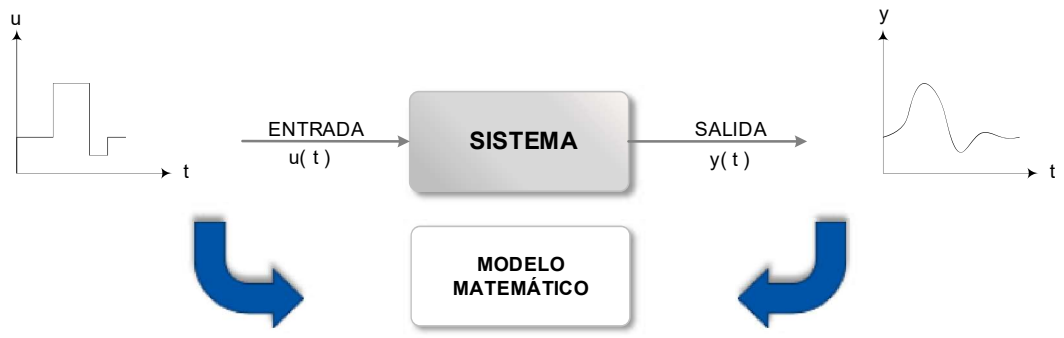


Figura 1.3. Esquema de identificación de modelos de un sistema.

En la Figura 1.3 muestra un esquema básico de identificación de modelos que no contempla el ingreso de perturbaciones. Por tanto, los modelos obtenidos de esta forma poseen una validez limitada, pero tienen la ventaja que se pueden ajustar a las condiciones y señales de entrada del proceso en curso.

El procedimiento que sigue toda identificación de modelos de sistemas se describe a continuación [3],[4],[9]:

1. *Diseño del experimento o adquisición de datos:* se determina cuáles son las entradas y salidas del sistema. Se excita al sistema mediante una señal de entrada, de la cual se registra su evolución y la evolución de la señal de salida del sistema durante un intervalo de tiempo a una taza de muestro regularmente 10 veces mayor a la frecuencia más alta de interés [9]. Los datos adquiridos deben ser lo mayormente informativos para poder identificar el modelo correcto.

2. *Examinar y pulir los datos:* en toda adquisición de datos siempre existe ruidos indeseados, tendencias o sobresaltos que es preferible que sean filtrados antes de comenzar la identificación.

3. *Seleccionar la estructura del modelo:* si lo que se desea es obtener un modelo paramétrico es necesario seleccionar una estructura adecuada que esté de acuerdo con los datos adquiridos. La elección de la estructura se facilita si se conoce alguna ley física que actúe en el proceso. Siempre se escoge el modelo que tenga la mejor aproximación o explicación a los datos observados.

4. *Criterio de ajuste:* en la estimación de los parámetros se necesita un discernimiento para medir la calidad del modelo identificado, que se logra con la minimización de un criterio de ajuste, para lograr que los parámetros estimados del modelo identificado se ajusten a la respuesta de los datos adquiridos. La elección de este criterio de ajuste depende de la información disponible y del modelo propuesto.

5. *Criterio de validación:* para finalizar con la identificación se determina si el modelo obtenido satisface el grado de exactitud requerido o proporciona un buen balance entre simplicidad y exactitud con el modelo real. Si el modelo resulta no válido, se debe revisar que el conjunto de datos adquiridos proporcione la información suficiente sobre la dinámica del sistema, que la estructura del modelo escogido describa adecuadamente al sistema real o si el criterio de ajuste es el adecuado.

El proceso de identificación es secuencial, pero al determinar la causa de una identificación errónea se lo puede repetir desde el punto correspondiente. Un organigrama del proceso de identificación de modelos de sistemas se presenta en la Figura 1.4.

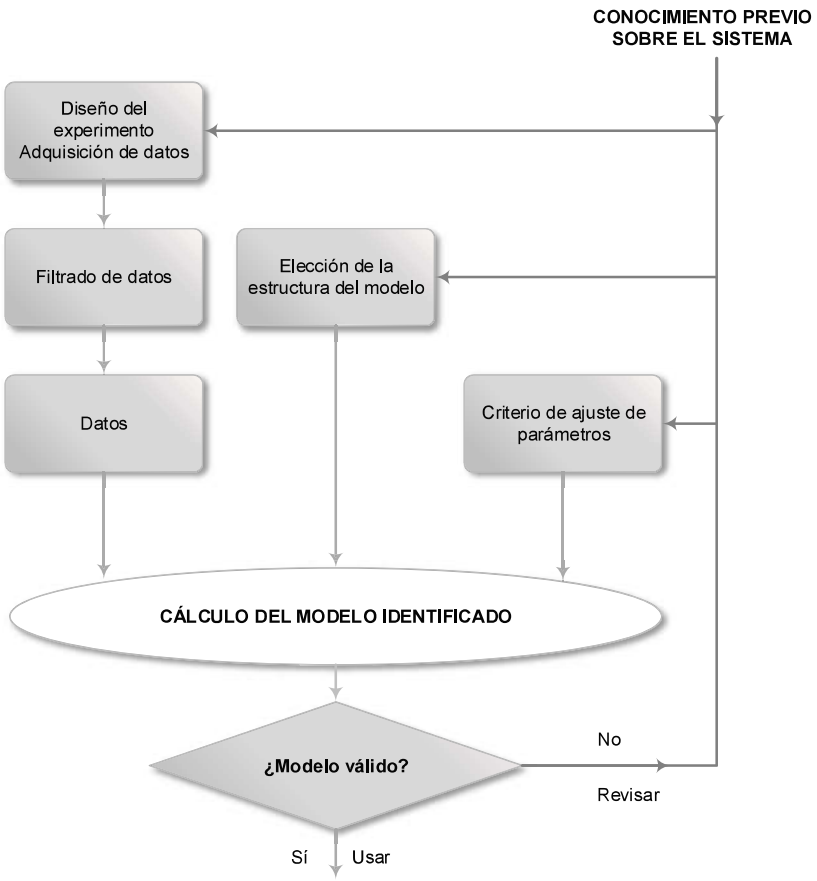


Figura 1.4. Lazo del proceso de identificación de modelos de sistemas.

1.3.2.1 Técnicas paramétricas de identificación de sistemas

Existen varias técnicas para ajustar los parámetros estimados de una identificación de sistemas, entre las cuales se destaca las siguientes:

- *Entradas especiales- respuesta a una entrada paso*

La identificación de un sistema mediante su respuesta a una entrada paso es uno de los métodos más utilizados en el control de procesos. Se trata de excitar al proceso con una entrada paso $u(t)$ y registrar la evolución de su salida $y(t)$. La Figura 1.5. muestra la respuesta del proceso a una entrada paso, como también la del modelo identificado, en función de transferencia, ante la misma entrada.

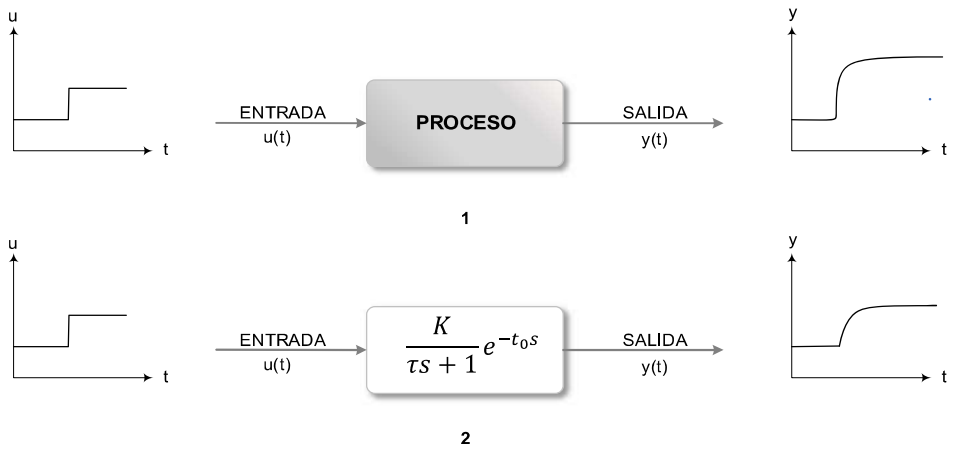


Figura 1.5. Respuesta a una entrada paso del proceso real a identificar (1). Modelo identificado para estimar los parámetros del proceso (2).

Mediante la observación de la respuesta se determina la estructura del modelo, y se reconstruyen los parámetros característicos mediante cálculos matemáticos de acuerdo con el modelo identificado [3]. Esta técnica es fácil de implementar y útil en sistemas multivariables, para identificar la relación entre las entradas con las variables del sistema.

- *Minimización del índice de error*

En la Figura 1.6. se presenta el esquema de minimización del índice de error, esta técnica consiste en estimar los parámetros característicos del modelo identificado de tal forma que cumpla con la minimización del error entre la respuesta del proceso real y la respuesta del modelo identificado. Es decir, es la obtención del mínimo valor entre la diferencia de la salida estimada y la salida del proceso real durante el tiempo de acción del proceso.

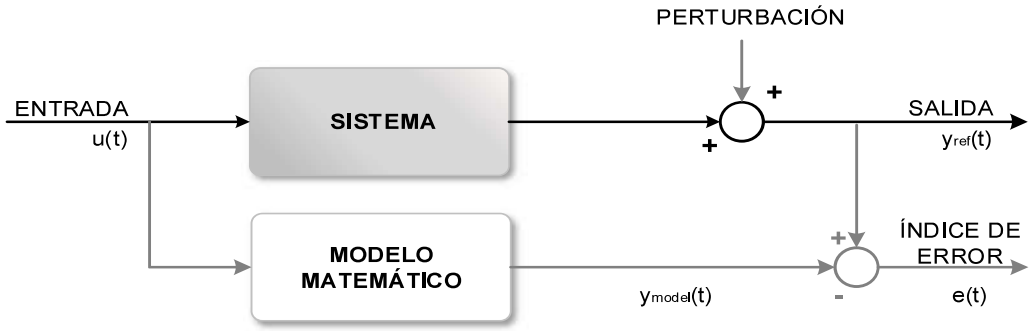


Figura 1.6. Esquema de minimización del error.

La Ecuación 1.3 presenta la forma del índice de error que se busca minimizar.

$$I(t) = \sum_{t=1}^{K} \left(y_{ref}(t) - y_{model}(t, \theta) \right)^2 \tag{1.3}$$

Donde:

$y_{ref}(t)$: salida del sistema real o referencia.

$y_{model}(t, \theta)$: salida del modelo identificado con parámetros a estimar.

θ: son los parámetros del modelo que se desea calcular.

En sistemas lineales los parámetros tienen una solución única, porque se pretende que el índice de error siempre sea el mínimo y que tienda a cero. Para sistemas no lineales se considera que en la reducción de orden tanto como en la estimación de parámetros se busca que el índice sea el menor posible o que tenga una tolerancia de error aceptable.

- *Algoritmos Computacionales*

El uso de algoritmos computacionales como redes neuronales, algoritmos de optimización o machine learning resuelven el problema de identificación de modelos como el de la estimación de parámetros y no necesariamente se debe conocer los procesos internos del sistema o describirlos en base las leyes fundamentales de la física y la química.

Todos los algoritmos computacionales trabajan con un problema de optimización, siendo la estimación de parámetros de modelos un problema del cual se busca minimizar las diferencias entre los valores del modelo real y del modelo estimado obteniendo un conjunto de constantes que describan perfectamente al modelo de caso de estudio.

Cuando se aborda problemas de optimización es necesario inicializarlos con precaución, porque en los problemas del mundo real, las soluciones óptimas no son conocidas, y por lo tanto el intervalo de inicialización y el espacio de búsqueda se eligen de acuerdo con estimaciones previas. No obstante, una inicialización cercana al valor que se lo considera como óptimo o incluso simétrica alrededor de este ocasiona un sesgo indeseado [10].

1.3.3 MODELOS MATEMÁTICOS DE SISTEMAS LINEALES

Como antecedentes se menciona que la identificación de sistemas de forma paramétrica está definida por una estructura y un número finito de constantes que relacionan las señales de entrada, salida y perturbaciones del sistema. Aun cuando se posee el modelo matemático del sistema, no es seguro que se pueda diseñar un controlador a partir de este, debido a su complejidad, la presencia de no linealidades o estados de dimensión extensos. Por esta razón, la reducción del modelo es una buena opción para llegar a este propósito, pero requiere de un modelo lineal para empezar.

Bajo tal antecedente es necesario explicar que un sistema es lineal cuando para todos los valores de x e y, y para cualquier constante $\propto$ [11], se cumplen las siguientes propiedades:

- *Homogeneidad:* cuando la función es multiplicada por una constante tienen el mismo valor si es evaluada en esa constante, y cumple con la Ecuación 1.4.

$$f(x \propto) = \propto f(x) \tag{1.4}$$

- *Superposición:* cuando la suma de un par de argumentos es igual a la suma de sus resultados, como en la Ecuación 1.5.

$$f(x + y) = f(x) + f(y) \tag{1.5}$$

Una expresión más compacta para la definición de linealidad es: dada una función $f(x)$, se dice que es lineal si, para todos los valores de x e y del dominio de f, y para cualquier número constante $\propto$ y β, se cumple la Ecuación 1.6.

$$f(\alpha x + \beta y) = \propto f(x) + \beta f(y) \tag{1.6}$$

La identidad de linealidad también se cumple para operaciones como: la derivación, la integral definida y la transformada de Laplace de una función $f(t)$ son operaciones lineales.

Es necesario enfatizar que la herramienta computacional desarrollada trabaja con la optimización de los valores de los parámetros en la representación de sistemas en forma de función de transferencia, como en la Ecuación 1.7. porque el principal interés es analizar el comportamiento a la salida del sistema a la entrada escalón, con la finalidad de obtener información útil sobre el comportamiento dinámico del sistema [11].

$$G(s) = \frac{Y(s)}{U(s)} = K\frac{(s - c_1)(s - c_2)\dots(s - c_m)}{(s - p_1)(s - p_2)\dots(s - p_n)} \tag{1.7}$$

Donde:

K: es la ganancia en estado estable del sistema.

$(s - c_m)$: es la representación de ceros.

$(s - p_n)$: es la representación de polos.

Para la identificación de modelos de sistemas se considera a las funciones de transferencia propias, esto significa que el grado del denominador es mayor que el grado del numerador, como se representa en la Ecuación 1.8.

$$m \leq n \tag{1.8}$$

Donde:

m: grado del numerador.

n: grado del denominador.

La respuesta en el tiempo de los sistemas lineales ante un cambio de una señal paso unitario posee características que es necesario conocer su concepto. Figura 1.7 representa la respuesta de un sistema de segundo orden al cambio de referencia de una entrada paso y posee algunos de las de las características que se describirán a continuación [7]:

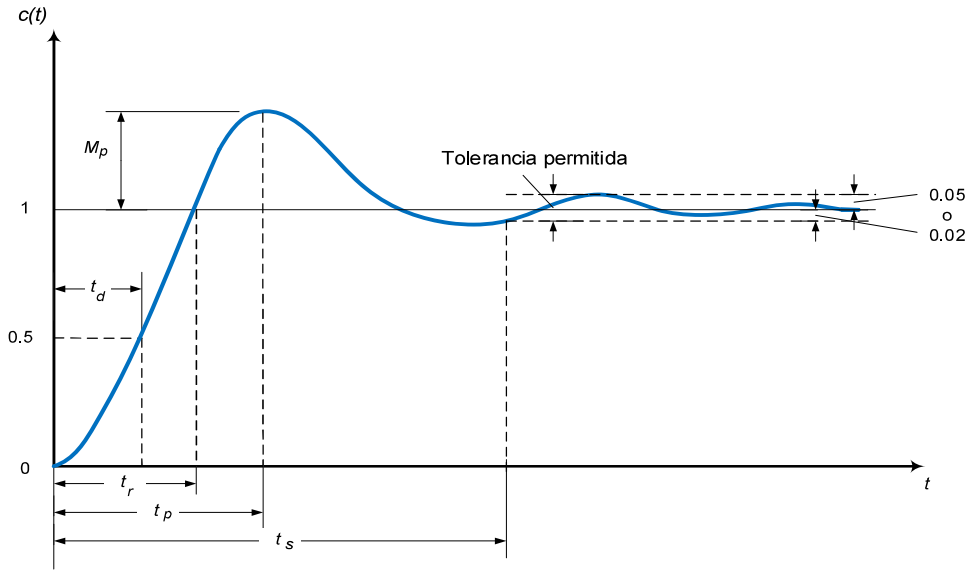


Figura 1.7. Curva de respuesta a escalón unitario con $\boldsymbol{t_d, t_r, t_p, M_p}$ y $\boldsymbol{t_s}$ [7].

- *Tiempo de respuesta:* intervalo de tiempo comprendido entre el instante en que una señal de entrada sufre un cambio brusco específico y el momento en que la señal de salida alcanza, su valor final en régimen estable y sostenido.
- *Tiempo de establecimiento, t_s:* se define como el tiempo necesario para que la respuesta a la señal paso disminuya, alcance el régimen permanente y se mantenga dentro del 5% o 2% de su valor final.
- *Tiempo de subida, t_r:* tiempo para que la respuesta a la señal paso se eleve al 10% y 90% de su valor final, por lo general para sistemas sobre amortiguados. En sistemas subamortiguados de segundo orden se usa del 0 al 100%.
- *Tiempo de retardo, t_d:* es el tiempo requerido para que la respuesta a la señal de entrada paso alcance el 50% del valor final.
- *Constante de tiempo:* es el valor del tiempo cuando el sistema alcanza el 63.2% del valor final en régimen permanente del sistema. Define la rapidez del sistema.
- *Sobrepico máximo (Porcentaje), M_p:* es el máximo pico de la curva de respuesta, se define de la resta del valor máximo y el valor en estado estable de la salida.

- *Tiempo pico, t_p:* es el tiempo que se requiere para que la respuesta alcance el primer pico del sobrepico.

En los párrafos siguientes se describe teóricamente los tipos de modelos de sistemas lineales, a los cuales se les identificará la estructura de su modelo matemático y se les estimará sus parámetros característicos a valores muy cercanos a los óptimos. La herramienta computacional propuesta trabaja con la representación en función de transferencia y con el cambio en la entrada del sistema, dado por una señal paso.

1.3.3.1 Sistema de primer orden

Los sistemas de primer orden representan físicamente a los circuitos RC, a procesos térmicos: Tanques de agitación continua (CSTR) y Tanques CSTR no adiabáticos, y a procesos que incluyan manejo de gas, nivel o similares [7]. La Ecuación 1.9 describe un sistema de Primer Orden en el domino de Laplace.

$$\frac{C(s)}{R(s)} = \frac{K}{\tau s + 1} \tag{1.9}$$

Donde:

K: es la ganancia en estado estable del proceso.

τ: es la constante de tiempo del proceso.

Gráficamente se puede calcular el valor de los parámetros por la relación entre la variación de la señal de salida sobre la variación de la señal de entrada para K, y por la relación de que para $t = \tau$ la salida $c(t)$ es 0.632 como en la Ecuación 1.10, es decir se considera la respuesta del sistema al 63,2% del estado estacionario para el cálculo de τ.

$$c(\tau) = 1 - e^{-\frac{\tau}{\tau}} = 0.632 \tag{1.10}$$

La Figura 1.8 muestra la respuesta de un sistema de primer orden a una entrada tipo paso, con condiciones iniciales cero. Para este tipo de modelo de sistema, todos los procesos que se representen por la misma función de transferencia poseen la misma salida en respuesta a la misma entrada tipo paso, con la única diferencia de las proporciones cuantitativas dadas por el parámetro K, que en sí señala cuán sensible es la salida a un cambio en la señal de entrada como se expresa en [12].

Otra de las características importantes de los sistemas de primer orden es que conforme más pequeña es la constante de tiempo τ, más rápida es la respuesta del sistema.

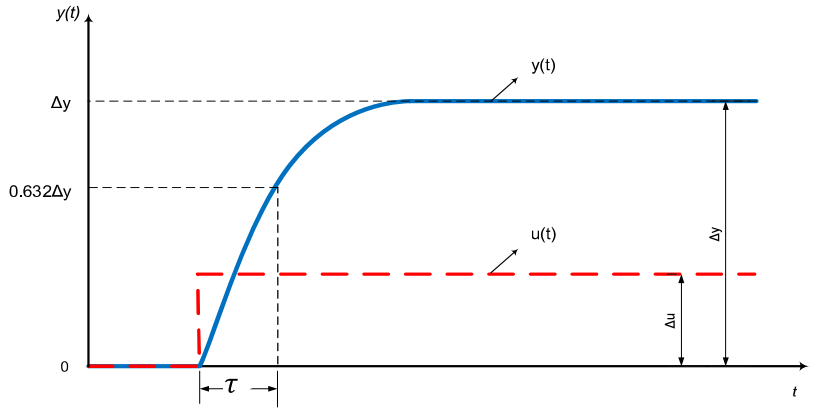


Figura 1.8. Respuesta en el tiempo de un sistema de primer orden.

Por otra parte, el tiempo de respuesta estimado razonable para este tipo de sistemas es igual a la longitud de tiempo que necesita la curva de respuesta para alcanzar el 2% del valor final que es igual a alcanzar de cuatro a cinco constantes de tiempo como se expresa en [7], [12]. Este es el tiempo suficiente para la descripción de parámetros característicos para este tipo de modelo de sistema [7].

1.3.3.2 Sistema de primer orden con retardo

También llamados F.O.P.D.T. por sus siglas en inglés (First Order Plus Delay Time), tienen la principal característica de presentar un retardo al inicio de la respuesta dinámica de la planta. Estos sistemas son la respuesta común para procesos biológicos, transporte de fluidos, de tanques de llenado y en sí cualquier proceso físico que involucre el transporte de material, energía o información [13]. La Ecuación 1.11 muestra la representación matemática de este tipo de sistema en la transformada de Laplace.

$$G(s) = \frac{K}{\tau s + 1} e^{-t_0 s} \tag{1.11}$$

Donde:

K: es la ganancia en estado estable del proceso.

τ: es la constante de tiempo del proceso.

t_0: es el tiempo muerto efectivo del proceso.

Entre las causas más comunes para presentar retardos en la salida es el tiempo de procesamiento de los sistemas de comunicación, sensores, actuadores, el trasporte de masa, energía donde la longitud de la trayectoria y la velocidad del movimiento constituyen el retardo o los propios controladores que a veces necesitan tiempo para implementar los algoritmos de control que podrían resultar complejos [13], [14],[15].

Los principales errores en el modelado de este tipo de sistemas, como se expone en [15] y se deben a:

- Error en la estimación de la ganancia, K.
- Error en la estimación del tiempo de retardo, t_0.
- Error de estimación de la constante de tiempo en estado estable, τ.
- Dinámicas no modelas, es decir polos y ceros más rápidos que el polo dominante.

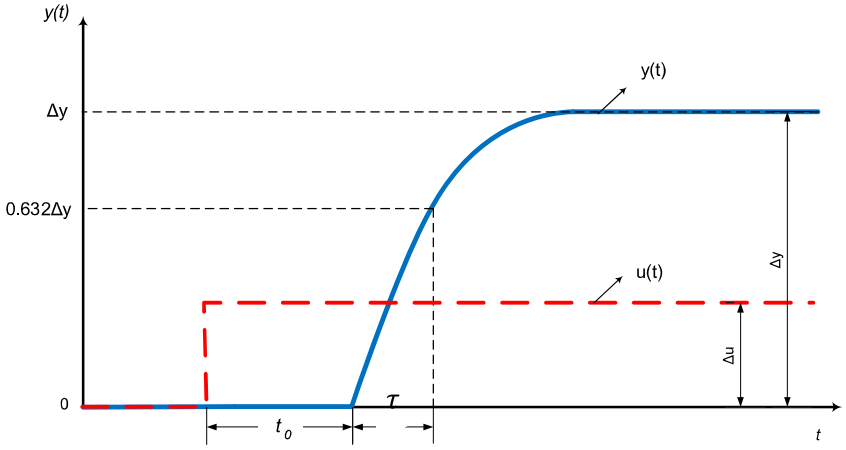


Figura 1.9. Respuesta en el tiempo de un sistema de primer orden con retardo.

La Figura 1.9 muestra la respuesta en el tiempo de un sistema de primer orden con retardo ante una señal de entrada tipo paso. Como se aprecia en la figura existe un retardo considerable desde el inicio de la función escalón de entrada a la acción de salida de la planta, esto hace que la acción de control se retarde al no haber un efecto observable alguno después del cambio de referencia [13].

1.3.3.3 Sistema de segundo orden

Todo tipo de sistema eléctrico describe su respuesta como un sistema de segundo orden y tiene como forma estándar a la Ecuación 1.12.

$$G(s) = \frac{K{\omega_n}^2}{s^2 + 2\varepsilon\omega_n s + {\omega_n}^2} \tag{1.12}$$

Donde:

K: es la ganancia del sistema en estado estable.

ε: es el factor de amortiguamiento relativo.

ω_n: es la frecuencia natural no amortiguada.

Con la ecuación anterior se puede describir la respuesta a una entrada paso de una gran variedad de sistemas dinámicos de diferente naturaleza física. El parámetro ξ describe

algunas diferencias cualitativas de la respuesta dinámica de los sistemas de segundo orden y es un parámetro fundamental para el diseño de sistemas de amortiguamiento y filtros activos. Así se tienen algunos tipos de sistemas de segundo orden dependiendo del valor que tome, estos son:

- Si $\varepsilon > 1$ es un sistema sobre amortiguado, los polos en lazo cerrado son reales negativos y diferentes. La respuesta es monótona y estable.
- Si $0 < \varepsilon < 1$ es un sistema subamortiguado, los polos en lazo cerrado son complejos conjugados con parte real negativa, es decir se encuentran en el semiplano izquierdo del plano s. Su respuesta transitoria es oscilatoria y estable.
- Si $\varepsilon = 0$ sus raíces son complejas con la parte real cero, por tanto, es un sistema sin amortiguamiento en su respuesta transitoria y presenta una respuesta de oscilaciones sostenidas indefinidamente.
- Si $\varepsilon = 1$ es un sistema críticamente amortiguado, los polos en lazo cerrado son reales e iguales. La respuesta es monótona y estable.
- Si $-1 < \varepsilon < 0$ las raíces de este sistema son complejas con la parte real positiva, la respuesta presenta oscilaciones crecientes.
- Si $\varepsilon \leq -1$ las raíces son positivas y reales, su respuesta es monótona e inestable.

Por el contrario ω_n representa la frecuencia natural no amortiguada del sistema, es decir es la frecuencia de oscilamiento si el coeficiente de amortiguamiento ε es cero. Si el sistema posee amortiguamiento se aprecia la frecuencia natural amortiguada ω_d de la Ecuación 1.13, esta siempre es menor que la frecuencia natural no amortiguada.

$$\omega_d = \omega_n\sqrt{1-\varepsilon^2} \tag{1.13}$$

Si épsilon ε, aumenta, la frecuencia natural amortiguada ω_d se reduce; pero si aumenta más de la unidad épsilon ε, la respuesta se vuelve sobre amortiguada y no oscilará. La Figura 1.10 representa la respuesta en el tiempo de un sistema de segundo orden subamortiguado a la entrada de una señal paso. Una característica importante de los sistemas de segundo orden, como lo menciona en [7] es que a igual épsilon ξ pero distinto ω_n las respuestas presentan la misma sobre elongación, muestran el mismo patrón oscilatorio y se dice que poseen la misma estabilidad relativa.

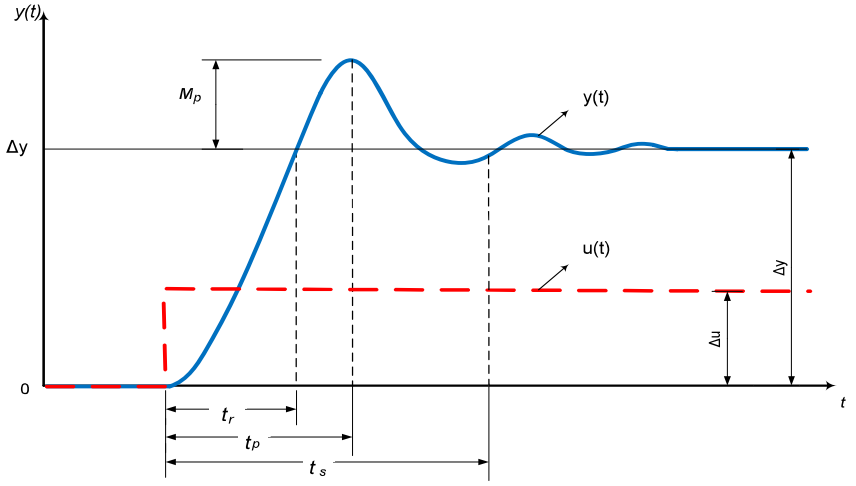


Figura 1.10. Respuesta en el tiempo de un sistema de segundo orden.

Los sistemas de segundo orden son comunes en los procesos eléctricos y sus características transitorias son más notables. Es necesario conocer estas características para escoger la acción de control adecuada para la dinámica de dicho proceso.

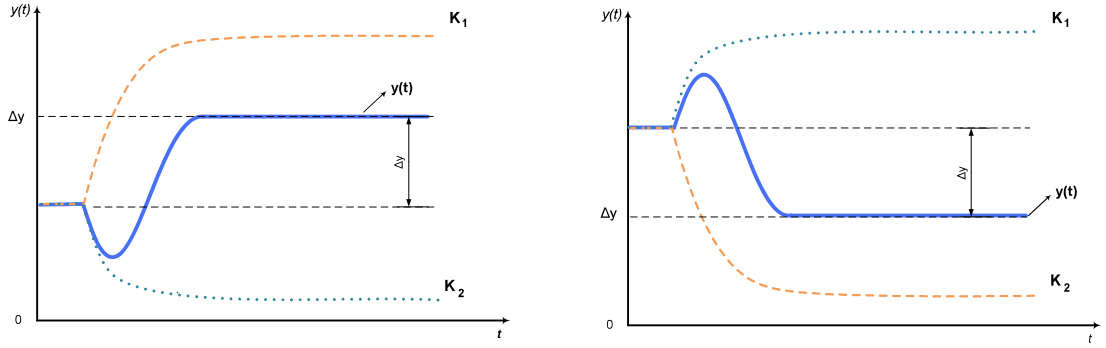
1.3.3.4 Sistemas de respuesta inversa

Un sistema de respuesta inversa se representa matemáticamente como la sustracción de dos sistemas de primer orden de dinámicas diferentes, en la Ecuación 1.14 y Ecuación 1.15 se muestra lo mencionado.

$$G(s) = G_1 - G_2 \tag{1.14}$$

$$G(s) = \frac{K_1}{(\tau_1 s + 1)} - \frac{K_2}{(\tau_2 s + 1)} \tag{1.15}$$

Existen dos tipos de sistemas de respuesta inversa, uno de ganancia positiva (a) y otro de ganancia negativa (b), al final del estado estable como se visualiza en la Figura 1.11.



a. b.

Figura 1.11. Respuesta a una entrada escalón de un sistema de respuesta inversa.

En [16] se puede elaborar una presentación alternativa de los sistemas de respuesta inversa, así se factoriza numerador con denominador obteniendo la Ecuación 1.16.

$$G(s) = \frac{K(1-\eta s)}{(\tau_1 s+1)(\tau_2 s+1)} \tag{1.16}$$

Donde:

K: es la ganancia en estado estable del sistema.

η: es el cero de la fase no mínima.

τ_1, τ_2: son las constantes de tiempo.

En la reducción de términos se considera las equivalencias de la Ecuación 1.17, derivadas del desarrollo matemático de la Ecuación 1.16.

$$K = K_1 - K_2 \tag{1.17}$$

$$\eta = \left(\frac{K_1}{\tau_1} - \frac{K_2}{\tau_2}\right)\frac{\tau_1\tau_2}{(K_1 - K_2)}$$

Sin embargo, la Ecuación 1.16 solo puede describir un sistema de respuesta inversa como el de la Figura 1.11.a si K_1 y K_2 son positivas, $K_1 > K_2$, con esto el resultado es un $K > 0$ y $\eta < 0$. Al cumplirse con las condiciones antes mencionadas al cero del sistema se lo considera negativo siempre.

La Figura 1.12. representa la respuesta en el tiempo de un sistema de respuesta inversa a una señal de entrada paso, como se observa la respuesta inicial es opuesta al valor final del sistema en estado estacionario.

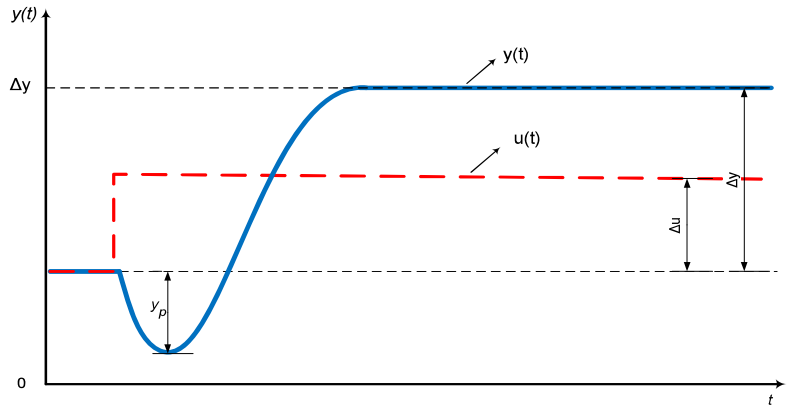


Figura 1.12. Respuesta en el tiempo de un sistema de respuesta inversa.

En general este sistema se forma por la acción de dos fenómenos físicos de dinámicas opuestas. El primero es un proceso rápido de pequeña ganancia, de este depende la

respuesta pico inversa; el segundo es un proceso lento de gran ganancia que finalmente sobrepasa a la ganancia del proceso inicial [17]. Este tipo de comportamiento se visualiza más en columnas de destilación, sistemas de calentamiento, reactores de temperatura tubulares catalíticos exotérmicos o tubulares adiabáticos.

1.3.4 CONCEPTOS FUNDAMENTALES DE LOS SISTEMAS EMERGENTES

Los sistemas emergentes están presentes en todo el entorno de forma implícita: en una colonia de hormigas, en un cardumen o en un enjambre; o también de manera explícita: en las formas de poder, en las organizaciones sociales o en el mercado de valores. Todos ejemplos anteriores tienen una sola similitud: son el resultado de la composición de sus niveles inferiores, de la interacción de sus individuos, participantes o integrantes, los mismos que poseen autonomía y características propias.

Para un mejor entendimiento se redactan a continuación los principales conceptos que describen a los sistemas emergentes.

1.3.4.1 Emergencia

El concepto de emergencia comienza con el filósofo Aristóteles, al introducir la idea que un sistema es más que la suma de sus partes. Se entiende a la emergencia como la transición de una organización simple a otra de complejidad superior [18], mediante la auto-organización, que es espontánea, y de la cual se considera que los individuos del nivel inferior actúan según sus propias leyes, emergiendo a entidades más complejas que llevan a un beneficio superior global a todo el sistema, esto sin la supervisión de algún agente.

1.3.4.2 Algoritmo emergente

Es un proceso de resolución de problemas, basado en un conjunto de reglas simples, que da una solución global compleja, sin que esta última se haya detallado explícitamente [18].

Los sistemas de insectos sociales son modelos de procesos de auto- organización no supervisados, de los cuales se ha deducido fórmulas matemáticas y modelos informáticos descentralizados. Entre los algoritmos emergentes más estudiados están: los algoritmos de colonias de hormigas, inspirados en el comportamiento de las hormigas en el proceso de búsqueda de alimento y el de enjambre de partículas, que hacen referencia a la población de aves que viajan en conjunto en busca de mejores condiciones de vida.

1.3.4.3 Estigmergia

Es una forma indirecta de comunicación, medida por modificaciones en el medio ambiente, a menudo está asociada con la flexibilidad de interacción entre individuos de una

agrupación, por ejemplo cuando el medio ambiente cambia a causa de una perturbación, los insectos en conjunto responden adecuadamente a la perturbación [18] sin necesidad que se les indique como deben actuar.

1.3.4.4 Inteligencia colectiva

Este tipo de inteligencia se refiere a la capacidad de resolver problemas a través de las interacciones de unidades simples de procesamiento de información; como es el caso de los enjambres de insectos o colonia de hormigas, es decir las interacciones simples entre los individuos del grupo pueden emerger a un comportamiento complejo e inteligente.

La inteligencia colectiva requiere de cuatro pilares para el procesamiento de información en la colonia o enjambre:

- *Coordinación y colaboración,* para regular la densidad espacio- temporal de los individuos y la asignación de sus tareas respectivamente.
- *Cooperación y deliberación,* presenta herramientas para que los mecanismos superen limitaciones individuales y permitan apoyar las decisiones del conjunto.

En los siguientes literales se realiza una descripción detallada sobre los modelos de optimización bio-inspirados que se han considerado para la elaboración de este proyecto de titulación, los mismos que corresponden a la optimización por colonia de hormigas para sistemas discretos, por enjambre de partículas para sistemas continuos y se destaca la adición de un tercer método de optimización de colonias de hormigas para sistemas continuos y discretos denominado ACO-R.

1.3.5 MODELO DE COLONIAS DE HORMIGAS

El modelo de Colonias de Hormigas se sustenta en el proceso de organización y respuesta eficiente de las hormigas, a encontrar fuentes de alimento cercanas y a evadir perturbaciones del medio ambiente. Fue introducido por [19], con su principal aplicación para la solución de problemas de optimización discreta

El proceso de búsqueda de alimento de una colonia de hormigas se indica en Figura 1.13, el punto **1** muestra a una hormiga exploradora tomar un camino al azar para llegar al alimento dejando un rastro de feromona a su paso, la cantidad de hormona aumenta de acuerdo con la cantidad de comida encontrada [18]. En el punto **2** se observa que cada integrante de la colonia toma diferentes caminos al azar para alcanzar a la fuente de alimento, con la diferencia que por el camino más corto habrá más afluencia de hormigas,

como pasa en el punto **3** donde se evidencia que toda la colonia ha emergido o convergido a un solo camino que es el más corto para la fuente de alimento.

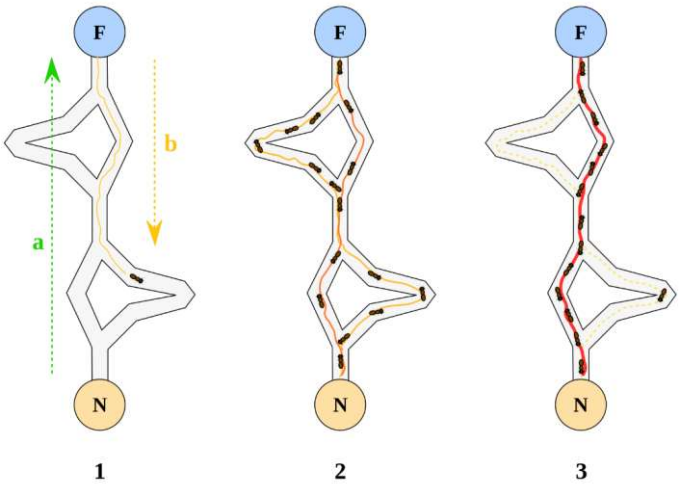


Figura 1.13. Búsqueda de alimento de la colonia de hormigas.

El número de individuos y los cambios en las condiciones medio ambientales determinan la dinámica de respuesta de la colonia y la toma de decisiones. Los patrones de las hormigas pueden cambiar precipitadamente, pero son reorientadas mediante el proceso de actualización de la feromona.

El proceso de búsqueda del mejor camino en las hormigas artificiales, que para este escrito se las denomina hormigas, se inicia con una distribución uniforme de una pequeña cantidad de feromona en todo el espacio de solución. Mediante la función de probabilidad cada hormiga selecciona los nodos que estén más próximos o que tengan una gran cantidad de feromona para la creación de caminos, una vez que todas las hormigas de esa iteración han completado sus caminos se procede a actualizar la feromona global.

Se pondera una cantidad extra de feromona para aquellos caminos que presenten el menor valor en la función de costo, los cuales tendrán mayor probabilidad de ser seleccionados en las iteraciones siguientes; asimismo, una fracción de feromona se evapora de los caminos que poseen un elevado valor en esta función. A este procedimiento se lo conoce como estigmergia y determina la calidad del camino. La función de costo es cualquier función que optimice el resultado final sea con su maximización o minimización, para este trabajo de titulación se considera la minimización de la función del índice cuadrático del error. Finalmente, el proceso se itera hasta que se cumpla una de las condiciones de parada: por el número máximo de iteraciones o el mínimo de la función de costo [20].

A continuación, se describe en detalle cada una de las reglas que se siguen para el proceso de optimización por el algoritmo de colonia de hormigas.

1.3.5.1 Regla de transición

En el proceso de construcción de la solución se crea un gráfico que limita el espacio de búsqueda, el cual toma el nombre de grafo o espacio de solución, y es un conjunto de puntos o nodos uniformemente distribuidos.

La regla de transición o función de probabilidad P [18], tiene el objetivo de escoger cual es el próximo camino para la hormiga actual k en el nodo r en la iteración t cuando elige pasar al nodo s. Esta función se define como la cantidad de feromona del camino actual elevado a un coeficiente $\propto$, por la heurística de deseabilidad del camino (r, s) elevado a un coeficiente β, sobre la suma de la feromona y heurística de deseabilidad de todos los caminos que pertenecen a la solución en construcción hasta ese momento como se muestra en la Ecuación 1.18.

$$P_{rs}^{k,\ t} = \begin{cases} \dfrac{[{\gamma_{rs}}^{t}]^{\propto}[\eta_{rs}]^{\beta}}{\Sigma_{l\ \epsilon\ S_k(r)}[{\gamma_{rl}}^{t}]^{\alpha}[\eta_{rl}]^{\beta}} & Si\ \ s\ \in\ S_k(r) \\ 0 & Si\ \ s\ \notin S_k(r) \end{cases} \qquad (1.18)$$

Donde:

${\gamma_{rs}}^{t}$: es la cantidad de feromona en el camino r, s en la iteración t.

η_{rs}: es la medida heurística de la conveniencia del camino r, s en la construcción de la solución.

${\gamma_{rl}}^{t}$: es la cantidad de feromona de todos los caminos que pertenecen a la solución actual.

η_{rl}: es la medida heurística de deseabilidad de todos los caminos que pertenecen a la solución actual.

β: coeficiente heurístico de deseabilidad de seleccionar un nodo cercano r.

$\propto$: coeficiente de intensidad de la feromona.

$S_k(r)$: es el conjunto de nodos s no visitados aún por la hormiga k y son posibles de visitar desde r.

1.3.5.2 Regla de actualización de feromona

Conforme las hormigas recorran sus respectivos caminos y hayan construido una solución, se debe incrementar el valor de la feromona del mejor camino (el que tenga su función de costo de mejor valor) o de los caminos más prometedores. Por el contrario, a los caminos subóptimos se evaporiza la feromona previamente deposita para así poder descartarlos.

Mediante este proceso se garantiza que las hormigas sigan a los mejores caminos y descarten los caminos restantes. La actualización junto con la evaporización de la feromona se presenta en la Ecuación 1.19.

$$\gamma_{rs}{}^{t} = (1-\rho)\gamma_{rs}{}^{t-1} + \sum_{k=1}^{m} \Delta\gamma_{rs}^{k,\ t} \tag{1.19}$$

Donde:

$\gamma_{rs}{}^{t}$: cantidad de feromona que se deja en el camino r, s en la iteración t.

$(1-\rho)$: taza de evaporación.

$\gamma_{rs}{}^{t-1}$: cantidad de feromona en el camino r, s en la iteración $t-1$.

m: número total de hormigas, el cual es constante.

$\Delta\gamma_{rs}^{k,\ t}$: rastro de feromona agregado por la hormiga k en la iteración t.

ρ: coeficiente de evaporización $\in (0, 1]$.

Al igual que la actualización de feromona, la evaporización de feromona es necesaria para evitar una convergencia rápida del algoritmo y al favorecer al proceso de olvido se beneficia a la exploración de nuevas áreas en el espacio de búsqueda [20].

La actualización de la feromona se puede dar al final de cada iteración, como propone [20] o al finalizar toda la secuencia del algoritmo como en [21] . La primera por lo general permite una diversificación de búsqueda mientras que la última tiende a converger más rápido.

Las soluciones con menor índice de error encontradas al principio de la exploración determinan la actualización de la feromona, con esto se logra incrementar la probabilidad de éxito para las hormigas siguientes en un espacio de búsqueda más prometedor. Por tal razón se desarrolla una ponderación extra de feromona para los mejores caminos.

1.3.5.3 Regla ponderación de feromona

La penalización al mejor camino se da por la Ecuación 1.20 que es la adición de un delta de feromona. Esta se define como el inverso a la media heurística cuando se ha finalizado el recorrido, dicha medida es la distancia total recorrida por la hormiga k en la iteración t.

$$\Delta\gamma_{rs}^{k,\ t} = \begin{cases} \dfrac{1}{L_k{}^{t}} & Si\ camino\ (r,s)\ \in\ T_k{}^{t} \\ 0 & Si\ camino\ (r,s)\ \notin\ T_k{}^{t} \end{cases} \tag{1.20}$$

Donde:

$L_k{}^t$: longitud del camino recorrido por la hormiga k en la iteración t.

$T_k{}^t$: recorrido de la hormiga k en la iteración t.

La regla de ponderación cumple un papel importante en la actualización de la feromona, de esta depende que solo a los mejores caminos se les añada una ponderación extra, y que los caminos con un elevado índice de error no posean un aumento significativo de feromona, para que sean olvidados conforme pase el tiempo de ejecución del algoritmo.

En esta aplicación práctica la longitud del recorrido, que sería la función de costo del algoritmo se la ha remplazado por el índice de error cuadrático medio, a la cual se la considera como la medida heurística del algoritmo. De esta depende el proceso de emergencia a una respuesta óptima por la iteración del conjunto de hormigas.

1.3.5.4 Flujograma del algoritmo de optimización ACO

A continuación, se presenta el procedimiento para la elaboración de un algoritmo de optimización por colonia de hormigas, en la Figura 1.14.

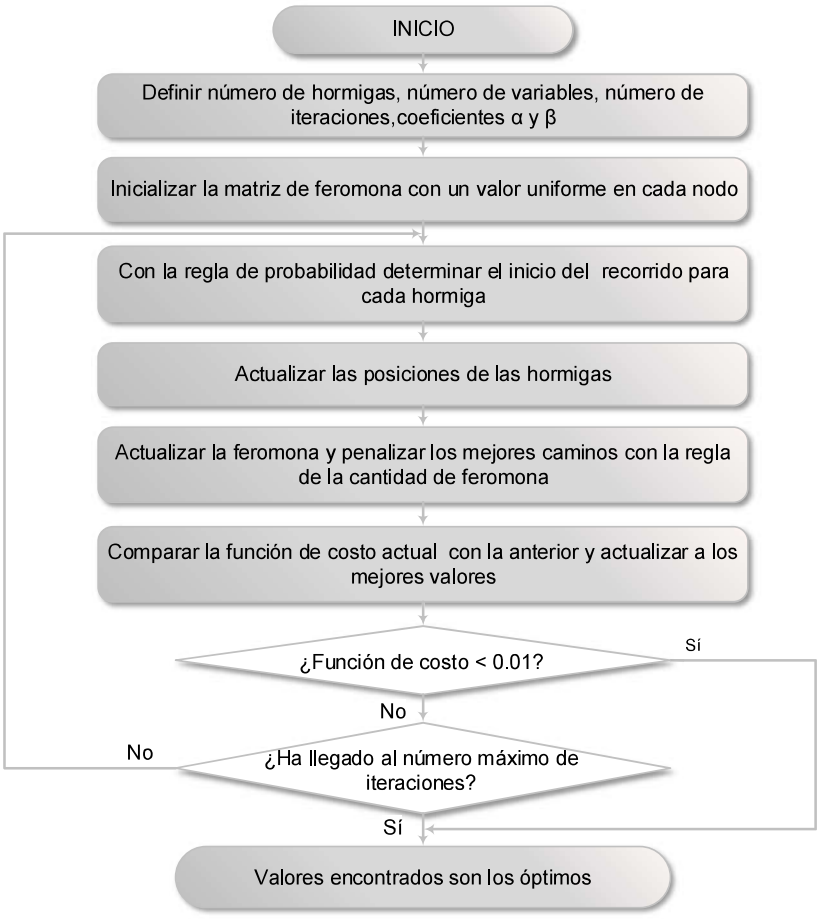


Figura 1.14. Flujograma del Método de Optimización ACO.

Como se puede apreciar en la Figura 1.14 se necesita más de una condición de parada para la finalización del flujo de programación; por consiguiente, se ha considerado a un error menor del 0,01 en la función de costo para dar fin a la ejecución del mismo, además del original que se da al llegar al número máximo de iteraciones.

1.3.5.1 Pseudo código del algoritmo de optimización ACO

Para la construcción del algoritmo utilizado en este proyecto de titulación se ha realizado un estudio de distintas fuentes bibliográficas, entre las más importantes está el pseudo código presentado a continuación, que fue tomado de [18]:

Fijar número de hormigas m
Fijar número de iteraciones n
    Repetir hasta que el sistema llegue a una buena solución
    Colocar las hormigas m en una posición inicial i
    Para $i = 1, n$
        Para $k = 1, m$
            Selección del nodo por la Ecuación 1.18
            Actualizar la posición de la hormiga
        Actualizar la feromona de cada camino por la Ecuación 1.19
    Actualizar el mejor camino

1.3.5.2 Aplicaciones del algoritmo de optimización ACO

Los algoritmos de optimización por colonia de hormigas son adecuados para problemas de optimización combinatoria (Combinatorial Optimization Problems- COPs), que acepten una representación vía grafo necesaria para imitar la búsqueda de un camino. El ejemplo más común es el Vendedor Viajero (Traveler Salesman Person- TSP) que trata de encontrar la ruta más corta para llegar a todas las ciudades que debe visitar, descrito en [20].

En los ejemplos de COPs se incluyen a problemas asignación de carga horaria como el de [22]; de programación de horarios de apertura en talleres industriales como en [23]; el enrutamiento de vehículos expuesto en [24], control y programación de robots; y algunas nuevas aplicaciones como los métodos constructivos con compartición de información [25].

Por lo general para estos problemas es imposible encontrar algún algoritmo de solución óptima, por eso se usa métodos heurísticos que permitan encontrar soluciones aproximadas, es decir soluciones buenas pero no óptimas, en una cantidad de tiempo razonable [10], siendo esta la principal razón para el uso del método de optimización ACO.

1.3.6 MODELO DE COLONIA DE HORMIGAS PARA SISTEMAS CONTINUOS

Muchos de los problemas de optimización en el mundo real se los puede presentar como COPs, es decir su solución se da como una combinación o permutación de los componentes disponibles del problema, esto implica una descomposición del problema en un grupo finito de componentes, así los algoritmos de optimización combinatoria intentan encontrar la combinación o permutación óptima que da solución al problema [10].

También existen los problemas de optimización continua (Continuous Optimization Problem– CnOP) en los que se requiere escoger valores continuos para su solución. Por tanto, el uso de los algoritmos de optimización combinatoria se restringe a cuando el rango de valores continuos admisibles para la solución es en un conjunto finito de valores; esto es un inconveniente si el posible rango de solución es muy amplio y la resolución requerida es muy alta, debido al costo computacional que implicaría hallar dicha solución. Además, el espacio de solución para los CnOP está expresado como regiones de una función continua; entonces la evaluación del rastro de feromona, la adición o la evaporación de feromona y la selección de la mejor solución no se puede realizar en un punto discreto [26].

Por tal motivo, desde que el algoritmo de optimización ACO emerge como una herramienta de optimización combinatoria se lo ha modificado con el propósito de que sea útil para la solución de problemas continuos. Así surge el algoritmo de optimización ACO-R que es una modificación al algoritmo ACO para dominios continuos [10], por ser una modificación cercana, este algoritmo aborda a problemas mixtos de optimización continuos y discretos.

A continuación, se describe las diferencias del algoritmo de optimización original con el algoritmo ACO-R.

1.3.6.1 Función de densidad de probabilidad

La principal diferencia del algoritmo ACO-R con el algoritmo inicial es el uso de una distribución de probabilidad continua, es decir de una función de densidad de probabilidad (probability density function- pdf). Así en vez de que una hormiga escoja un componente del conjunto de solución mediante la Ecuación 1.18, lo hará por el muestro de una pdf.

La definición de pdf es cualquier función que cumpla con la Ecuación 1.21, tal que su integral definida sea igual a la unidad como en la Ecuación 1.22.

$$p(x) \geq 0 \quad \forall\, x \tag{1.21}$$

$$\int_{-\infty}^{\infty} p(x)dx = 1 \tag{1.22}$$

La función Gaussiana es una de las pdfs más utilizadas por su facilidad de muestreo, pero presenta algunas desventajas; la más significativa es su incapacidad de describir una situación en la que dos áreas separadas del espacio de búsqueda son prometedoras porque tienen un solo máximo. Por esta razón es preferible utilizar un kernel gaussiano que se define como la suma ponderada de varias funciones gaussianas unidimensionales [10].

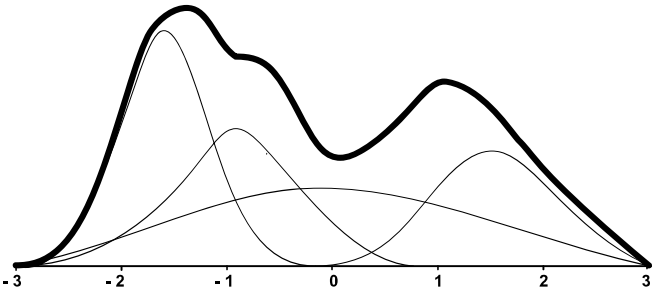


Figura 1.15 Ejemplo de Kernel pdf Gaussiano con cuatro funciones gaussianas (la ilustración se limita al rango de $\boldsymbol{x} \in [-\mathbf{3}, \mathbf{3}]$) [27].

1.3.6.2 Actualización de la feromona

La pdf de kernel gaussiano definida para la actualización de la feromona se da por la suma de varias funciones gaussianas $g_l^t(x)$ de acuerdo con el número de iteraciones t que se designen en el algoritmo como se define en la Ecuación 1.23. El uso de esta función pdf no solo proporciona una flexibilidad mayor en la forma posible de distribución de la feromona, sino que permite aún un muestreo fácil. La Ecuación 1.24 representa a una función de distribución de normal gaussiana.

$$G^t(x) = \sum_{l=1}^{k} \omega_l g_l^t(x) \tag{1.23}$$

$$g_l^t(x) = \frac{1}{\sigma_l^t \sqrt{2\pi}} e^{-\frac{(x-\mu_l^t)^2}{2{\sigma_l^t}^2}} \tag{1.24}$$

Donde:

$G^t(x)$: es el kernel gaussiano en función de cada punto x, que representan una solución posible para la función de distribución de densidad.

x: son las posibles soluciones.

ω_l: es el vector de peso o ponderación asociado a la l-iésima solución encontrada, que es una por iteración y por hormiga.

$g_l^t(x)$: es la función de distribución normal gaussiana, de la iteración t de la l-iésima solución.

μ_l^t: es el vector de media de la iteración i, de la solución l-iésima.

σ_l^t: es el vector de desviación estándar de la iteración t, de la solución l-iésima.

$\sigma_l^{t\,2}$: es el vector de varianza de la iteración t, de la solución l-iésima.

En consecuencia, la feromona no se restringe a un conjunto finito definido por una repartición lineal uniforme entre dos límites como en ACO; por el contrario, utiliza una distribución normalizada. También se diferencia en que no se descarta toda la feromona de la solución actual; cuando la matriz de feromona es actualizada se mantiene un cierto número de soluciones que son utilizadas para generar las nuevas pdfs. Es decir, las soluciones archivadas sirven para el cálculo ω, μ y σ, parámetros que sirven para el cálculo del kernel gaussiano utilizado para guiar a las hormigas en el proceso de búsqueda.

El tamaño del archivo solución guardado debe ser igual o incluso mayor a la dimensión del problema, con el fin de efectuar una solución adecuada. La matriz de feromona es actualizada agregando las soluciones generadas al archivo. En este, las soluciones son ordenadas de acuerdo con el problema de optimización; para este trabajo es de forma ascendente porque se trata de un problema de minimización, es decir se pretende guardar los valores más pequeños que representan a una mejor solución; para luego eliminar las soluciones no buenas en el mismo número de las soluciones incorporadas, garantizando que el tamaño del archivo no cambie. Este proceso garantiza que los componentes de la matriz de feromonas sean solo de las mejores soluciones encontradas y las soluciones no óptimas sean eliminadas cancelando su influencia en las próximas búsquedas.

1.3.6.3 Vector de ponderación de las funciones gaussianas

Cada una de las funciones gaussianas que son añadidas a la matriz de solución son evaluadas y clasificadas de acuerdo con su ponderación. El vector de ponderación de las funciones gaussianas está representado por ω_l, que se define como un valor proveniente de una función gaussiana de argumento l, media de 1 y de desviación estándar qk y su cálculo se presenta en la Ecuación 1.25.

$$\omega_l = \frac{1}{qk\sqrt{2\pi}} e^{-\frac{(l-1)^2}{2q^2k^2}} \tag{1.25}$$

Donde:

l: es la solución l-iésima.

k: es el tamaño de la matriz solución.

q: es el coeficiente de diversificación del proceso de búsqueda.

qk: es la desviación estándar.

q^2k^2: es igual que $(qk)^2$ y representa a la varianza.

El parámetro q evita que el algoritmo permanezca en un óptimo local, pero permite su convergencia una vez encontrado el óptimo global, cuando q es pequeño las soluciones con el mejor ranking son más seleccionadas, y con un q grande existe una distribución de probabilidad más uniforme.

1.3.6.4 Muestreo de la PDF de kernel gaussiano

Una vez determinados los elementos del vector de ponderación ω, se procede al muestreo del kernel gaussiano, el mismo que se realiza en dos fases. La primera consiste en escoger una de las funciones gaussianas que compone el kernel gaussiano y calcular su probabilidad de selección mediante la Ecuación 1.26.

$$p_l = \frac{\omega_l}{\sum_{r=1}^{k} \omega_r} \tag{1.26}$$

Donde:

p_l: es la probabilidad de elegir la función gaussiana actual.

ω_l: es el vector de ponderación de la función gaussiana actual.

ω_r: es el vector de ponderación de las funciones gaussianas de la matriz solución.

La segunda fase consiste en muestrear a la función gaussiana seleccionada, mediante un generador de números aleatorios de acuerdo con una distribución normal; esta forma se ha considerado para el proyecto de titulación. Una forma similar de muestreo es usando un generador aleatorio uniforme en conjunto con, por ejemplo, el método Box- Müller [25], para la generación de números aleatorios normalmente distribuidos. Estas fases de muestreo son lo mismo que muestrear el kernel gaussiano definido en la Ecuación 1.23.

1.3.6.5 Cálculo de la desviación estándar

El cálculo de el vector σ de desviaciones estándar se realiza solo para la solución actual seleccionada s_l, se define como a distancia promedio de esta solución con respecto a las demás soluciones archivadas por un coeficiente que representa la velocidad de convergencia como esta en la Ecuación 1.27.

$$\sigma_l^t = \xi \sum_{e=1}^{k} \frac{\left|s_l^t - s_r^t\right|}{k-1} \tag{1.27}$$

Donde:

σ_l^t: es la desviación estándar única por iteración t y por solución actual l.

s_l^t : única solución elegida para la solución actual l en la iteración t.

s_r^t: soluciones restantes de la matriz solución en la iteración t.

ξ: es el coeficiente que relaciona la distancia de desviación.

k: es el tamaño de la matriz solución.

Se debe destacar que todos los cálculos son únicos para cada hormiga por cada iteración, la desviación estándar también tiene su cálculo en una sola iteración t, esto asegura que el algoritmo se adapte a las transformaciones del problema [10]. Por tanto, la distribución de probabilidad es diferente para cada hormiga porque depende de los valores anteriores siendo esta la principal diferencia con el algoritmo original, debido a que cada pdf de kernel gaussiano es la unión de funciones gaussianas de una sola dimensión superpuestas.

1.3.6.1 Pseudo código del algoritmo de optimización ACO-R

El pseudo código presentado a continuación fue elaborado en [10]. El mismo que fue implementado en la realización de este proyecto de titulación.

Fijar número de hormigas m
Fijar número de iteraciones n
    Repetir hasta que el sistema llegue a una buena solución
    Colocar las hormigas m en una posición inicial i
    Evaluar la función de costo, separar el mejor costo inicial
    Para $i = 1, n$
        Para $k = 1, m$
            Establecer el tamaño de nueva población
            Crear el kernel gaussiano PDF
            Selección del nodo por la Ecuación 1.26
            Actualizar la posición de la hormiga
        Actualizar la matriz de costo con la nueva población
    Actualizar el mejor camino

1.3.6.2 Flujograma del algoritmo de optimización ACO-R

Para la creación del algoritmo de optimización por ACO-R se sigue el flujograma presentado en la Figura 1.16. El flujograma es de elaboración propia y establece los principales pasos que se debe considerar al resolver un CnOP.

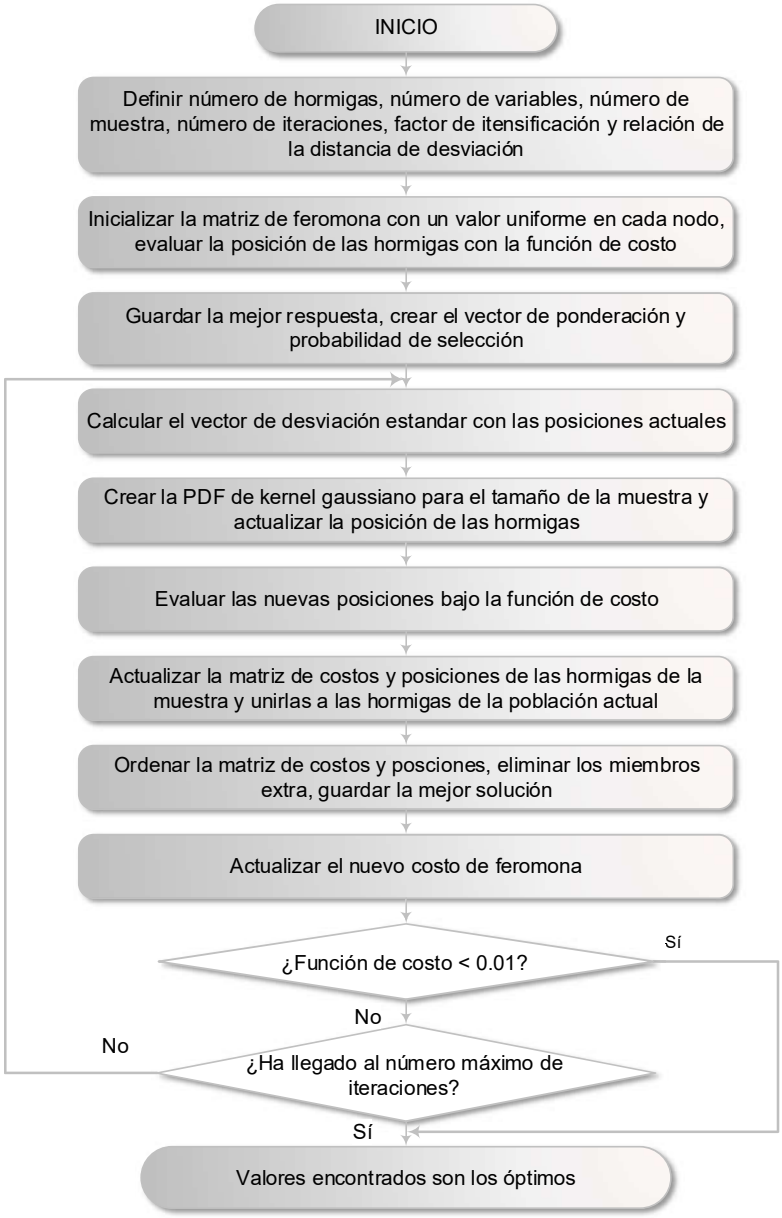


Figura 1.16 Flujograma del Método de Optimización ACO-R.

Al igual que el algoritmo anterior, en la aplicación del algoritmo ACO-R se necesita más de una condición de finalización, con el objetivo de reducir el tiempo de ejecución y el consumo de recursos del ordenador.

1.3.6.3 Aplicaciones del algoritmo de optimización ACO-R

La aplicación del algoritmo ACO-R es para la solución a problemas de optimización combinatoria de dominio continuo; la principal ventaja de este algoritmo sobre los demás, es que no necesita que la función de optimización sea continua o diferenciable, convirtiéndole en un algoritmo de aplicación más general.

El uso de una función de densidad de probabilidad para encontrar áreas prometedoras en el espacio de solución asemeja su comportamiento al de los algoritmos evolutivos; los mismos que en base a la generación actual, modifican o mutan los agentes, para que las próximas generaciones sean mejores; con la diferencia que ACO-R trabaja también con dominios discretos.

1.3.7 MODELO DE ENJAMBRE DE PARTÍCULAS

El modelo de enjambre de partículas pertenece a los algoritmos metaheurísticos basados en la inteligencia colectiva y está inspirado en el comportamiento de peces y pájaros. Se define como una población de vectores que, mediante la interacción entre cada uno de sus miembros dentro de una región definida y por ecuaciones de probabilidad, encuentran el mejor individuo en cada iteración y se desplaza en función de este, hasta converger en la mejor posición de todo el enjambre [28].

Las topologías presentes en la optimización por enjambre partículas, son:

- *GBEST (Global Best)*: es la topología mayoritariamente utilizada en la optimización por enjambre. Se describe como una población totalmente interconectada, todos los miembros pueden influir entre sí. A pesar de que contiene un mayor número de conexiones entre individuos; en la práctica, es seguir a la mejor partícula o individuo del enjambre. [18].
- *LBEST (Local Best):* topología también conocida como anillo, porque cada partícula está conectada con aquellas que se encuentran a los lados. Esto lleva a que puedan convergen de manera independiente en diversos óptimos. Esta topología es más lenta para converger a un óptimo global, porque cada individuo es influenciado por un número reducido de miembros de la población.

Los miembros del enjambre están motivados a buscar y mantener coherencia entre ellos; se organizan y se reorganizan con el propósito de producir patrones consistentes, en esto subyace la inteligencia colectiva, así también cada partícula representa a una solución candidata para el problema a optimizar en el espacio de búsqueda.

A continuación, se describe los conceptos principales como el proceso de optimización que sigue el algoritmo de enjambre de partículas.

1.3.7.1 Espacio de búsqueda

Es definido como el conjunto de todas las posibles soluciones del problema de optimización, del cual se debe encontrar la mejor solución. Las partículas están aleatoriamente distribuidas en el espacio de búsqueda, poseen memoria con lo cual pueden recordar la mejor posición donde han estado y la mejor posición que se ha encontrado en el enjambre; estos conceptos corresponden a los componentes cognitivo y social, que a su vez representan los principios del algoritmo de optimización: aprendizaje y comunicación, respectivamente.

1.3.7.2 Representación geométrica del algoritmo PSO

Cada partícula del enjambre posee un vector posición $x_i{}^t$ y un vector velocidad $v_i{}^t$; por poseer memoria también conoce su mejor vector posición local $P_i{}^t$ y su mejor posición global G^t de todo el enjambre. Su nueva posición $x_i{}^{t+1}$ es la combinación de estos tres vectores y el vector resultante es la velocidad de la partícula $v_i{}^{t+1}$, en la iteración $t+1$.

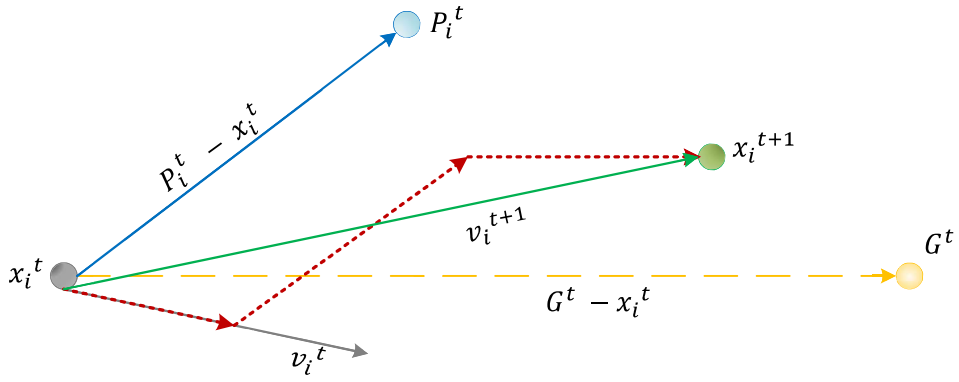


Figura 1.17. Representación geométrica de algoritmo de optimización por enjambre de partículas (PSO).

La nueva posición en $x_i{}^{t+1}$ representa a una mejor, por ser el resultado del movimiento previo de la partícula: la mejor experiencia previa de la partícula y la mejor experiencia global del enjambre. Basados en esta premisa, se intuye que la cooperación de todas las partículas resulta en encontrar la mejor posición en el espacio de búsqueda.

1.3.7.3 Inicialización del algoritmo

Se inicializa a cada partícula con un vector de posición $x_i{}^t$, que representa a una solución del problema de optimización y un vector de velocidad $v_i{}^t$, que determina la magnitud de

cambio del vector posición [18], es decir a mayor velocidad, mayor es el cambio en la posición. La Ecuación 1.28 define la posición de la partícula en la iteración $t+1$.

$$x_i{}^{t+1} = x_i{}^{t} + v_i{}^{t+1} \tag{1.28}$$

Donde:

$x_i{}^{t+1}$: es la posición de la partícula i en la iteración $t+1$.

$x_i{}^{t}$: posición actual de la partícula i en la iteración t.

$v_i{}^{t+1}$: velocidad que lleva la partícula i en la iteración $t+1$.

La actualización de la velocidad es modificada por el componente cognitivo y social de cada partícula, como también de la topología utilizada.

1.3.7.4 Actualización de la velocidad

En la topología GBEST, la velocidad de la partícula está dada por la Ecuación 1.29, donde la velocidad de la partícula i en la dimensión j durante la iteración $t+1$, es la velocidad actual de la partícula por el coeficiente de inercia, más la distribución uniforme de la mejor posición alcanzada por la partícula y en global alcanzada por el enjambre. Esta nueva posición es probablemente una mejor posición para la partícula.

$$v_{ij}{}^{t+1} = w v_{ij}{}^{t} + c_1 r_{1j}{}^{t}\left[P_{ij}{}^{t} - x_{ij}{}^{t}\right] + c_2 r_{2j}{}^{t}\left[\ G^{t} - x_{ij}{}^{t}\right] \tag{1.29}$$

Donde:

$v_{ij}{}^{t+1}$: velocidad de la partícula i en la dimensión j durante la iteración $t+1$.

w: coeficiente de inercia.

$v_{ij}{}^{t}$: velocidad de la partícula i en la dimensión j durante la iteración t.

c_1, c_2: constante de aceleración del componente cognitivo y del componente social.

$r_{1j}{}^{t}, r_{2j}{}^{t}$: números aleatorios en el rango de $[0,1]$, provenientes de distribuciones uniformes independientes.

$x_{ij}{}^{t}$: posición actual de la partícula.

$P_{ij}{}^{t}$: la mejor posición alcanzada por la partícula (componente cognitivo) en la iteración t.

G^{t}: la mejor posición global alcanzada por el enjambre hasta la iteración t (componente social).

El coeficiente de inercia controla la capacidad de exploración del algoritmo, da un equilibrio en la búsqueda de un óptimo global y del hallazgo de un óptimo local. Así en cada iteración este coeficiente se reduce gradualmente hasta ser nulo [29]; esto significa que al llegar alrededor de un óptimo global, las partículas se mantendrán alrededor este valor.

1.3.7.5 Posición individual

La dimensión de la mejor posición individual alcanzada por cada partícula se calcula de acuerdo con la Ecuación 1.30, tal que al evaluar a la función f en la iteración $t+1$, sí es mayor el valor de la función f en la posición actual, la dimensión siguiente es la misma que la actual; caso contrarío la dimensión de la posición siguiente está dada por la posición en la iteración $t+1$. Es decir, encontrar la mejor posición individual alcanzada por la partícula i se limita a minimizar la función f en esa iteración.

$$P_i{}^{t+1} = \begin{cases} P_i{}^{t} & Si\ f(x_i{}^{t+1}) \geq f\left(P_i{}^{t}\right) \\ x_i{}^{t+1} & Si\ f(x_i{}^{t+1}) < f\left(P_i{}^{t}\right) \end{cases} \tag{1.30}$$

Donde:

f: es la función que define la proximidad de la partícula con el punto óptimo. Función de costo.

1.3.7.6 Posición global

Para encontrar la mejor posición global alcanzada por el enjambre entre las mejores posiciones alcanzadas por las partículas del enjambre se cumple lo dispuesto en la Ecuación 1.31, donde la mejor posición global es el mínimo de la función de costo evaluada en cada una de las mejores posiciones encontradas por las partículas.

$$f(G^t) = min\{\ f(\ y_1{}^t\ ), \ldots, f(\ y_i{}^t\ )\ \} \parallel\ \ G^t\ \in \{\ y_1{}^t, \ldots,\ y_i{}^t\} \tag{1.31}$$

Donde:

i: es el número de partículas

G^t: es la mejor posición global del enjambre en la iteración t.

1.3.7.7 Flujograma del algoritmo de optimización PSO

En la Figura 1.18 se muestra el diagrama de flujo que describe, de forma general, el procedimiento a seguirse para la implementación de un algoritmo de optimización por PSO. Además, brinda una concepción básica del esquema de programación y su secuencia.

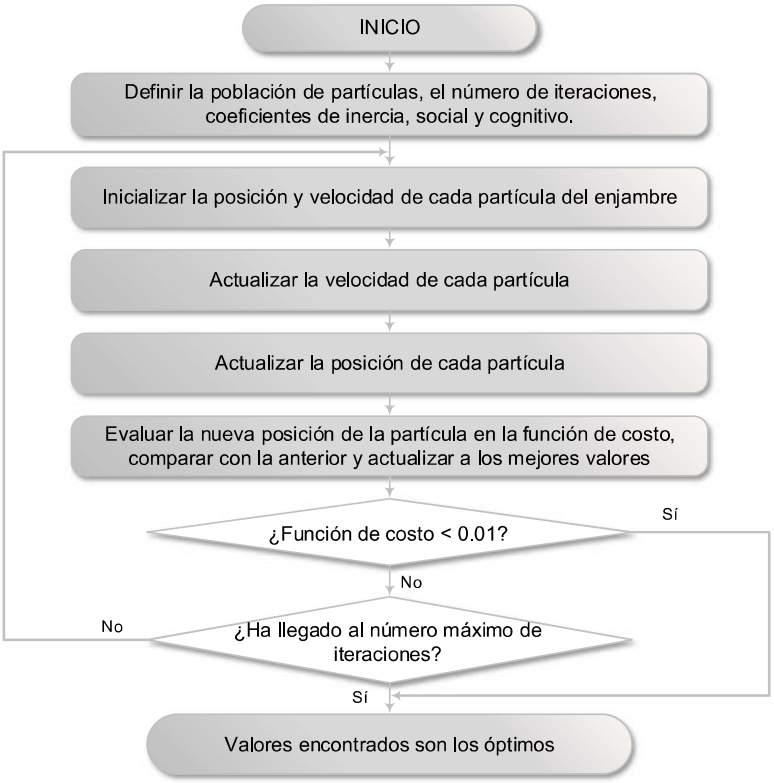


Figura 1.18. Flujograma de algoritmo de optimización por PSO.

Al igual que en los casos anteriores, en este algoritmo de optimización se integra una condición extra de parada que reduce el tiempo invertido en la identificación, comparado a cuando se realiza todo el número de iteraciones; a pesar de que se haya obtenido una muy buena aproximación del modelo matemático desde las primeras iteraciones.

1.3.7.8 Pseudo código del algoritmo de optimización PSO

A continuación, se presenta el pseudo código para la implementación de un algoritmo de optimización por enjambre de partículas, en base a [18].

Crear e inicializar un enjambre de n dimensiones
  Repetir hasta alcanzar condición de parada
    Para $i = 1, n$
      Si $f(x_i) < f(P_i)$ *entonces* $P_i = x_i$
      Si $f(P_i) < f(G)$ *entonces* $G = P_i$
    Para $i = 1, n$
      Actualizar la velocidad usando la Ecuación 1.29
      Actualizar la mejor posición de la partícula usando la Ecuación 1.30
  Actualizar la mejor posición global usando la Ecuación 1.31

1.3.7.9 Aplicaciones del algoritmo de optimización PSO

Las aplicaciones principales para el algoritmo de optimización por Enjambre de Partículas incluyen a problemas que tengan un espacio de búsqueda muy extenso puesto que las partículas se pueden alternar de forma flexible de una región a otra y pueden explorar simultáneamente en varias regiones del espacio.

La mayoría de las aplicaciones de este tipo de algoritmos incluyen la optimización de alguna función sea para minimizar o maximizar su valor con el objetivo de cumplir con la optimización de parámetros reales, optimización de funciones numéricas, entrenamiento de Redes Neuronales, Aprendizaje de Sistema Difusos, Registro de Imágenes son algunas aplicaciones que se destacan.

1.3.8 ÍNDICES DE DESEMPEÑO DE SISTEMAS

Una vez obtenido el modelo matemático estimado se necesita una medida cuantitativa que demuestre su validez, usando los índices de desempeño se puede cuantificar de diversas formas la diferencia entre la salida real y la salida estimada del sistema en un determinado intervalo de tiempo, comprobando si el modelo es aceptable.

En este proyecto de titulación, además del uso antes mencionado, los índices de desempeño son las funciones para optimizar en la estimación de los parámetros de los modelos de sistemas identificados, también llamadas funciones de costo dentro de los algoritmos de optimización ACO, ACO- R y PSO.

1.3.8.1 Error cuadrático integral

También llamado ISE por sus siglas en ingles de Integral Square Error es la media del error respecto a los valores reales de salida con los estimados en cada iteración t mediante la Ecuación 1.32. Al centrarse en el cuadrado del error penaliza los valores de error positivos como negativos, siendo útil para errores causados por cambios de referencia o al inicio de la ejecución del proceso [30]. Cuando el sistema real con el sistema estimado, llegan a un estado de similitud muy bueno el error permanece constante, convirtiendo al índice menos adecuado para la minimizar.

$$ISE = \int_0^\infty \left(y_{referencia}(t) - y_{modelo}(t)\right)^2 \tag{1.32}$$

Donde:

$y_{referencia}$: salida real o salida de referencia.

y_{modelo}: salida del modelo estimado.

1.3.8.2 Error cuadrático medio

Conocido en inglés como Mean Squared Error o MSE indica la varianza y sesgo del error a lo largo de la estimación del modelo, su forma matemática se representa en la Ecuación 1.33. Este índice penaliza a la estimación del modelo en sí, por eso mientras menor sea su valor, mejor es el modelo estimado.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_{referencia}(t) - y_{modelo}(t)\right)^2 \tag{1.33}$$

Donde:

n: es el número de muestras tomadas.

Ahora bien, es necesario cuantificar el error en cada parámetro estimado, por consecuente se hace uso del porcentaje de error en la medida, con el propósito de poseer un criterio adicional que permita determinar la eficiencia de la identificación de sistemas mediante cada uno de los algoritmos de optimización propuestos.

1.3.8.3 Error relativo

Se define como la magnitud de la diferencia entre un valor exacto y uno aproximado, dividida por la magnitud del valor exacto presentada en porcentaje, Ecuación 1.34. Esta medida permite ver qué tan lejos está un valor aproximado de uno exacto a través de un porcentaje del valor exacto. El error puede deberse al método de medición, una herramienta, a un error humano, o a las aproximaciones que se usan en el cálculo, por ejemplo, los errores de redondeo.

$$\%\, error = \frac{|aproximado - exacto|}{exacto} \times 100 \tag{1.34}$$

1.3.9 ADQUISICIÓN DE DATOS

Para una identificación de sistemas exitosa es necesario una buena adquisición de datos, por ello se presenta algunas consideraciones, y se define sus principales conceptos.

La adquisición de datos (Data Acquisition- DAQ) se define como el proceso de medir un fenómeno eléctrico o físico, mediante un sistema que consta de sensores, un hardware para la adquisición y un PC con software programable. Los sistemas DAQ basados en PC aprovechan el procesamiento, la productividad, la visualización y la conectividad de estos dispositivos, que son de uso frecuente en la industria, los mismos que proporcionan una solución más potente, flexible y rentable [31].

Figura 1.19 Partes de un sistema DAQ **[31]**.

- *Sensor:* es el elemento que convierte un fenómeno físico en una señal eléctrica que se puede medir. Dependiendo del tipo de sensor su salida puede ser en voltaje, corriente, resistencia u otro atributo eléctrico que varía con el tiempo [31].

- *Dispositivo DAQ:* es un hardware que actúa como la interfaz entre una PC y las señales del mundo exterior. Funciona como dispositivo que digitaliza señales analógicas entrantes, para que la PC pueda interpretarlas. Posee tres componentes indispensables: el circuito de acondicionamiento de señales, el convertidor analógico- digital (analog- digital converter– ADC) y un bus de comunicación [31].

- *PC:* en un sistema de adquisición la PC debe contar con un software programable que controle la operación del dispositivo DAQ y que además pueda ser usado para procesar, visualizar y almacenar los datos medidos. Así, el PC debe poseer un software controlador que le ofrezca al software de aplicación la habilidad de interactuar con el dispositivo DAQ; a la vez que el software de aplicación facilite la interacción entre la PC y el usuario [31].

Para el caso práctico de este proyecto de titulación se utiliza como dispositivo DAQ a una tarjeta Arduino® Mega 2560, como software controlador y de aplicación al entorno computacional de MatLab® de MathWorks® y como sensor se considera al encoder que posee el motor DC, del cual se adquiere bajo un acondicionamiento la de velocidad de salida del motor. A continuación, se presenta un breve resumen de lo que son estas dos plataformas de desarrollo.

- *MathWorks:* es el entorno computacional más productivo y de alto rendimiento técnico que combina la matemática comprensiva con las funciones gráficas en un lenguaje de alto nivel. Entre las facilidades que poseen están: MatLab® que permite analizar datos, desarrollar algoritmos y crear modelos matemáticos; Simulink® en el cual se ejecuta simulaciones, se genera código y se prueba y verifica sistemas embebidos [32]. Aparte posee diferentes complementos que ayudan al desarrollo de interfaces gráficas de usuarios, comunicación de dispositivos, pruebas en tiempo

real, entre algunos más. Para el desarrollo de este proyecto de titulación aparte de los softwares antes mencionados, se utiliza el complemento de App Designer® para la creación de la interfaz de usuario.

- *Arduino:* es una plataforma de software y hardware de código abierto, que ofrece una amplia gama de herramientas de software, plataformas hardware y documentación que permite el desarrollo de productos IoT (Internet of Things) tanto para estudiantes, profesionales, investigadores y cualquier persona en general [33]. Al ofrecer una plataforma IDE (Integrated Development Environment- Entorno de Desarrollo Integrado) se puede realizar cualquier tipo de aplicación por código con la depuración de este.

  Entre las características más relevantes tarjeta Arduino® MEGA 2560 están: un reloj interno de 16MHz, dos timers de 8 bits con preescalador separado y modo comparación, cuatro timers de 16 bits con preescalador separado, modo comparación y captura, 16 canales de conversión ADC de 10 bits, cuatro puertos USART seriales programables y 86 puertos de entrada/salidas programables.

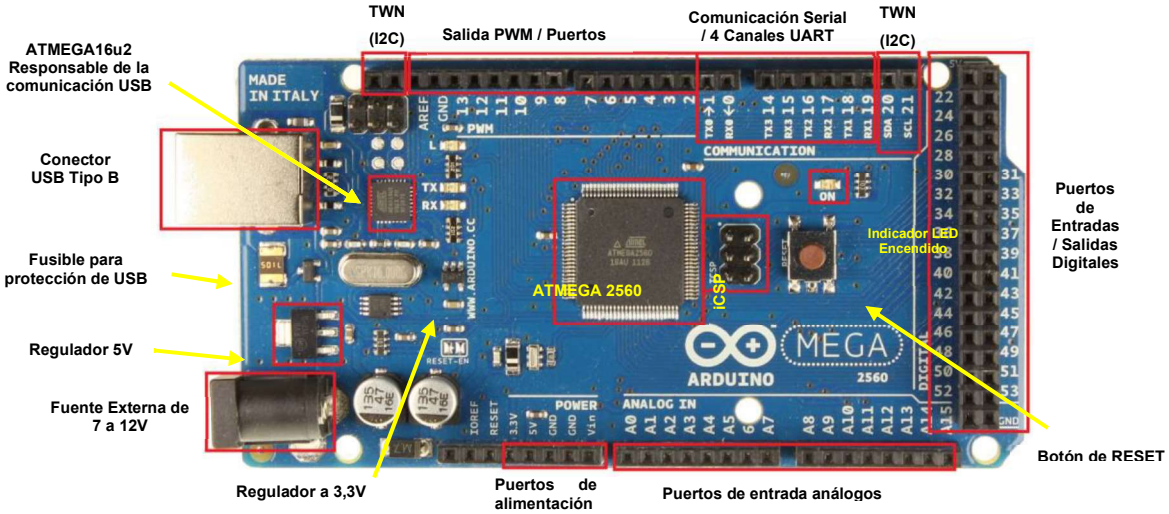


Figura 1.20 Componentes de la tarjeta Arduino® Mega 2560.

En una discretización de señales se necesita algunos conceptos, de los cuales destacamos los siguientes:

- *Teorema de Nyquist-Shannon:* este teorema establece que, para una reconstrucción en el dominio temporal de una señal adquirida, se necesita que la frecuencia de muestreo cumpla con la Ecuación 1.35, es decir que al menos sea dos veces mayor que la frecuencia más alta de la señal que se desea reconstruir.

$$f_s \geq 2f_{máx.} \tag{1.35}$$

  Donde:

f_s: es la frecuencia de muestreo.

$f_{máx.}$: es la frecuencia más alta de la señal.

La ecuación anterior se cumple siempre que $f_{mín.} = 0$. Cuando no es así se debe aplicar el criterio de ancho de banda Bw que se expresa en la Ecuación 1.36 y la Ecuación 1.37.

$$Bw = f_{máx.} - f_{mín.} \tag{1.36}$$

$$f_s > 2Bw_. \tag{1.37}$$

Donde:

Bw: es el ancho de banda

$f_{máx.}$: es la frecuencia máxima de la señal a muestrear.

$f_{mín.}$: es la frecuencia mínima de la señal a muestrear.

Finalmente, el tiempo de muestreo en base a la frecuencia de muestreo, se expresa en la Ecuación 1.38.

$$T_m = \frac{1}{f_s} \tag{1.38}$$

Donde:

T_m: es el tiempo de muestreo.

- *Resolución:* es el valor mínimo de variación de voltaje de entrada necesario para cambiar en un bit la salida digital, su cálculo se presenta en la Ecuación 1.39.

$$resolución = \frac{V_{fe}}{(2^n - 1)} \tag{1.39}$$

Donde:

n: es el número de bits del convertidor.

V_{fe}: es la tensión de salida del fondo de escala o voltaje máximo de salida.

Al tener en consideración estos conceptos se garantiza una adecuada discretización del sistema continuo; y en consecuencia su buen modelamiento. No obstante, existen limitantes como el tiempo de adquisición de datos del sensor; por eso se debe elegir un sensor de buena resolución y de características adecuadas en función del sistema real. Asimismo, una excesiva adquisición de datos no es lo adecuado por la cantidad de

memoria de almacenamiento que utiliza; al igual que con pocos datos no se puede realizar una buena aproximación discreta del sistema a causa de la perdida de información.

En la práctica existen otros conceptos que son necesarios para la elección del tiempo de muestreo de sistemas con una entrada paso como se expone en [34], así se tiene que:

- Para la elección del tiempo de muestreo (Tm) debe cumplirse la Ecuación 1.40, que relaciona la constante de tiempo del sistema en lazo cerrado con la toma de muestras.

$$T_m = \frac{\tau_{sistema}}{10} \tag{1.40}$$

Donde:

T_m: es el tiempo de muestreo de la planta.

$\tau_{sistema}$: es la constante de tiempo dominante del sistema.

Como se dice en [35], de la experiencia obtenida de los casos prácticos se tiene que el muestreo de alrededor de 1s son para procesos de flujo, nivel, presión y temperatura; y para los sistemas electromecánicos son en el orden de los milisegundos.

1.3.10 CONSIDERACIONES IMPORTANTES PARA EL MODELAMIENTO DE PLANTAS REALES

Para el modelamiento de plantas reales también se debe considerar algunas nociones que tiene relevancia en el momento de elegir las señales y el tipo de entrada para cual ser realizaran las pruebas y adquisición de datos. Se describen a continuación, los más imprescindibles:

- *Elección de las señales a medir:* lo primero es reconocer que señales se deben registrar: la señal que se puede manipular para excitar al sistema durante el experimento será la señal de entrada; la señal que refleja los cambios que tiene el sistema, será la señal de salida; y el tiempo. Estas son las señales que intervienen en la identificación de sistemas.
- *Elección del tipo de entrada:* para sistemas lineales es suficiente con utilizar una señal de entrada de dos niveles, puede ser una señal paso. En sistemas no lineales, se considera el punto de operación, y es preferible adquirir los datos bajo ese mismo entorno.

Con estos conocimientos establecidos se puede asegurar que la prueba práctica a desarrollar en este trabajo de titulación tiene más probabilidad de éxito.

2. METODOLOGÍA

El tipo de metodología utilizada es aplicativo porque se ha desarrollado una herramienta computacional, al mismo tiempo que cumple con características de ser descriptivo y explicativo a medida que se ha desarrollado en base a una teoría de modelos de optimización bio-inspirados. Por la complejidad de este tiempo de información se ha realizado una descripción detallada de los algoritmos de optimización aplicados en este trabajo práctico.

En la recolección de información principalmente fue por consulta en artículos científicos y libros de carácter relevante, con el fin de comprender la teoría que engloba la metaheurística de los modelos de optimización basados en inteligencia colectiva, tanto como a la teoría de los modelos matemáticos a sistemas lineales, la identificación de sistemas y la estimación de parámetros, su importancia para la automatización y estudio de técnicas de control, detección de fallos y simulación de procesos en la realidad.

En la primera fase, se describe el desarrollo del algoritmo ACO en base a la bibliografía consultada y se ajusta los parámetros que modifican su comportamiento de acuerdo con el problema a identificar. Como segundo se realiza la descripción del algoritmo de optimización ACO-R, se presenta sus parámetros modificables con su importancia y los valores que se han considerado como default para la identificación y estimación de parámetros del caso tipo de estudio. Posteriormente se describe al algoritmo de optimización PSO, a cada uno de los parámetros correspondientes a este e igualmente se establece un rango de los valores que se debe considerar para la modificación de los estos conforme la necesidad del problema tipo a identificar. Se desarrolla la herramienta computacional y la adquisición de datos, por medio de un Arduino® Mega 2560 y el entorno de MatLab para la realización del caso práctico. Adicionalmente se diseña la interfaz gráfica en App Designer® que posee las funcionalidades de escoger el modelo lineal al cual se desea aproximar y determinar cuál es el mejor modelo de aproximación, para un conjunto de datos obtenidos experimentalmente, en ambos casos la función de optimización inicial es el error cuadrático integral (ISE- Integral Square Error). A continuación, se complementa a la interfaz gráfica con el sistema de adquisición de datos antes mencionado, se realiza las simulaciones pertinentes a los cuatro tipos de modelos lineales y la optimización de parámetros e identificación del modelo de sistema a un motor DC. Finalmente, se compara los resultados obtenidos de la herramienta computacional con la herramienta IDENT de MatLab®, se describen los resultados obtenidos y se redacta las conclusiones y recomendaciones pertinentes.

2.1 IDENTIFICACIÓN DE SISTEMAS POR ALGORITMOS COMPUTACIONALES

Como se expuso en el marco teórico la identificación de sistemas puede realizarse mediante algoritmos computacionales, en el desarrollo de este trabajo de titulación se considera a dos algoritmos de optimización: colonia de hormigas- ACO y enjambre de partículas- PSO. Sin embargo, por la naturaleza del primer algoritmo es necesario considerar un tercer método de optimización con el fin de lograr una comparación justa en los resultados obtenidos por la herramienta computacional; así se integra el algoritmo de optimización ACO- R para sistemas discretos y continuos. Finalmente, para fines de esta aplicación práctica, el conjunto de estos algoritmos forma la herramienta computacional que se la ha denominado como SysID.

El objetivo de involucrar a estos algoritmos de optimización es encontrar la mejor estimación de parámetros característicos posible a la identificación del modelo matemático del sistema referencia, como una alternativa al uso a métodos convencionales de identificación como la curva de reacción o el modelamiento matemático. La Figura 2.1 muestra el esquema básico de como la herramienta computacional SysID consigue modificar iterativamente los parámetros estimados a unos mejores, mediante la minimización de la función de costo.

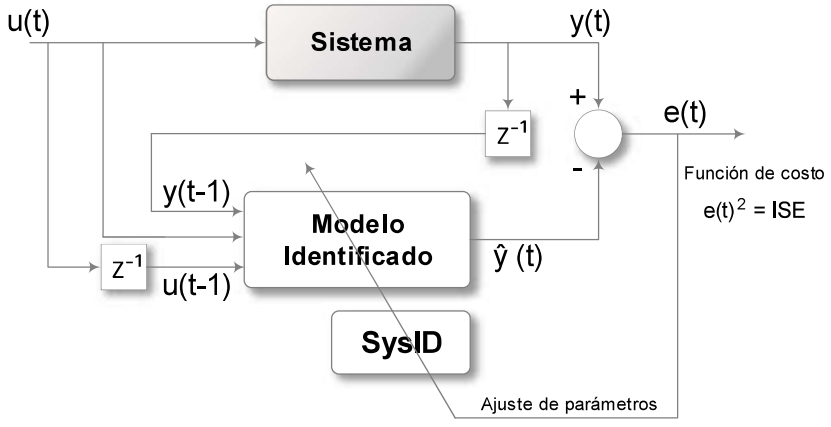


Figura 2.1. Principio de estimación de parámetros e identificación de sistemas por la herramienta computacional SysID **[36]**.

Para los tres algoritmos de optimización, la función de costo en un inicio fue el índice de error ISE entre la salida del sistema identificado y el sistema real. El éxito de la identificación depende de la minimización de este, al punto de que los parámetros estimados obtenidos sean la aproximación óptima posible al sistema de referencia.

2.2 PRESENTACIÓN DE LOS SISTEMA LINEALES TIPO

A continuación, se describen los sistemas lineales tipo a identificar mediante la herramienta computacional SysID.

2.2.1 SISTEMA DE PRIMER ORDEN

El sistema tipo de primer orden a identificar responde a la Ecuación 2.1.

$$G(s) = \frac{8}{(3s+1)} \tag{2.1}$$

Este modelo de sistema representa en la vida real a procesos térmicos de tanques de agitación continua (CSTR) [37]. La Figura 2.2. es la respuesta que se desea identificar y modelar en base a la estimación de parámetros mediante los algoritmos de optimización previamente descritos.

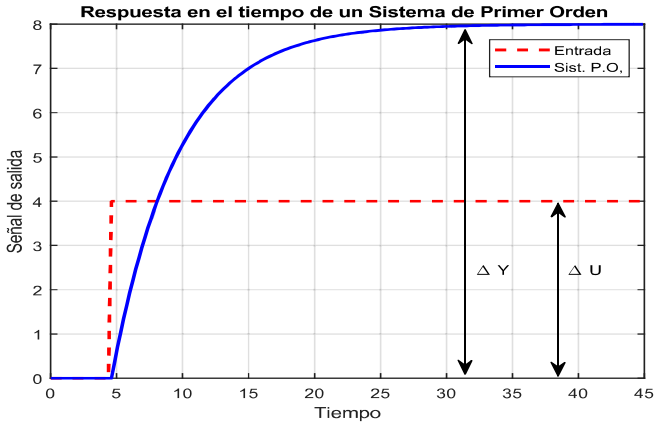


Figura 2.2. Respuesta del sistema de primer orden a identificar.

Al ser un sistema tipo se puede modificar sus parámetros característicos en la herramienta computacional, pero para una identificación adecuada SysID posee un campo de entrada en el cual se especifica los límites referenciales para la creación del espacio de búsqueda.

2.2.2 SISTEMA DE PRIMER ORDEN CON RETARDO

El sistema tipo de primer orden con retardo a identificar responde a la Ecuación 2.2.

$$G(s) = \frac{8}{(3s+1)} e^{-7s} \tag{2.2}$$

Este modelo presenta, ante un cambio de referencia, un tiempo muerto al inicio de su respuesta de salida, el cual se debe principalmente por el transporte físico de un fluido a través de una tubería como es en el caso de los tanques de llenado [37]. La Figura 2.3 muestra la respuesta al cambio escalón de un sistema de primer orden con retardo.

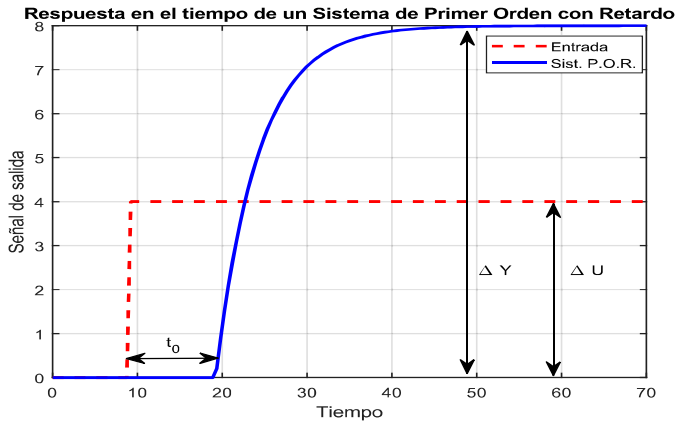


Figura 2.3. Respuesta del sistema de primer orden con retardo a identificar.

Como se mostró en la parte teórica son tres los parámetros característicos a identificarse con la herramienta computacional para este tipo de modelo de sistema. Al igual que el caso anterior, es importante establecer el límite en el espacio de búsqueda, porque en uno muy reducido, existe el riesgo de no encontrar el valor óptimo para cada parámetro.

2.2.3 SISTEMA DE SEGUNDO ORDEN

El sistema tipo de segundo orden a identificar responde a la Ecuación 2.3.

$$G(s) = \frac{8}{(s^2 + 0.4s + 1)} \tag{2.3}$$

Estos son la respuesta más común de los sistemas reales ante un cambio de referencia de una entrada escalón; aquí tenemos a los sistemas masa- resorte, fuerza y a los sistemas eléctricos [37]. La Figura 2.4 muestra la respuesta de un sistema de segundo orden a un cambio de referencia de una entrada escalón.

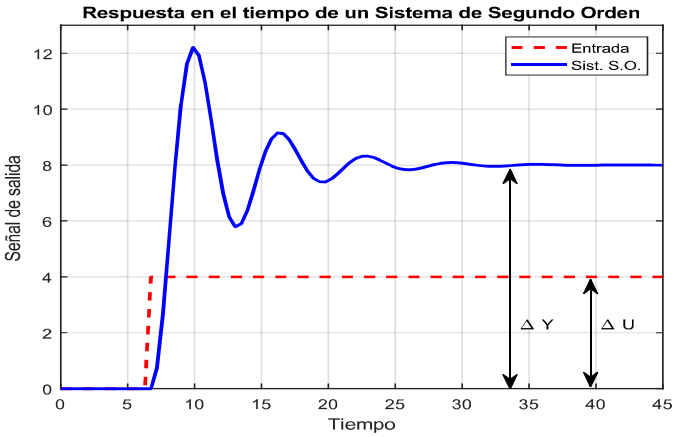


Figura 2.4. Respuesta del sistema de segundo orden a identificar.

Al igual que los casos anteriores el espacio de búsqueda debe estar de acuerdo con el parámetro de ganancia del sistema y cuando la respuesta llega al estado estable.

2.2.4 SISTEMA DE RESPUESTA INVERSA

El sistema tipo de respuesta inversa a identificar es expresado en una forma alternativa, como la resta de dos sistemas de primer orden con dinámicas diferentes, como se muestra en la Ecuación 2.4. En la teoría se expone los criterios que deben cumplirse para utilizar esta representación.

$$G(s) = \frac{8}{(5s+1)} - \frac{3}{(0.5s+1)} \tag{2.4}$$

Como se aprecia los parámetros a estimar son a cuatro; esto implica un tiempo de ejecución mayor, con un gasto computacional elevado, debido al número de iteraciones y miembros (tanto hormigas como partículas) que deben intervenir en la estimación de los parámetros. Por tanto, se cree conveniente trabajar mediante su forma recursiva dada por el método de Balaguer [8]. El desarrollo de este método para el fin que se ha planteado en este proyecto de titulación se presenta a continuación.

La Ecuación 2.5 presenta la forma recursiva de identificación a sistemas de respuesta inversa sin retardo y adimensional, mediante el método de Balaguer.

$$G(s) = \frac{K_{RI}(-b\tau s+1)}{(\tau s+1)(a\tau s+1)} \tag{2.5}$$

Donde el valor de K_{RI} está determinado por el cociente entre la variación de la entrada sobre la salida, Ecuación 2.6, valor que se puede obtener computacionalmente por código.

$$K_{RI} = \frac{\Delta U}{\Delta Y} \tag{2.6}$$

Resolviendo así la estimación a solo los tres parámetros restantes a, b y τ, con esto se asegura una optimización más eficiente en un menor tiempo.

El sistema final de respuesta inversa a identificar es el de la Ecuación 2.7.

$$G(s) = \frac{5(-2.2s+1)}{(5s+1)(0.5s+1)} \tag{2.7}$$

Antes de adentrarnos en el esclarecimiento de la implementación y diseño de la herramienta computacional, es necesario explicar que los sistemas tipo expuestos en los párrafos anteriores han sido utilizados únicamente para parametrizar los valores que necesitan cada uno de los algoritmos.

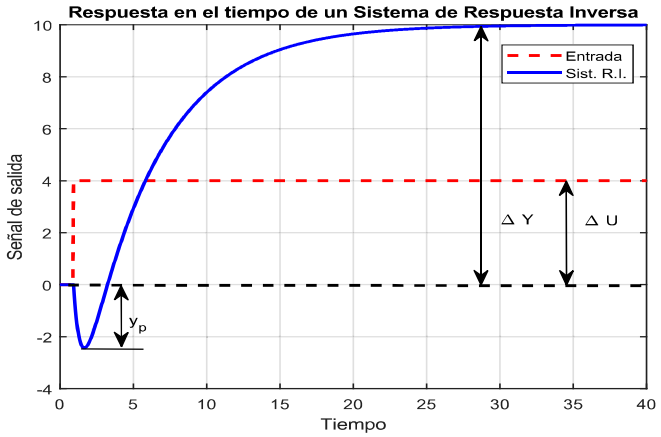


Figura 2.5. Gráfica del sistema de respuesta inversa a identificar.

Asimismo, para otros modelos que no sean los modelos tipo, se ha realizado una identificación previa, esto con el fin de delimitar el espacio de búsqueda de cada algoritmo. Por tanto, se utilizó el método de dos puntos para la identificación previa de los sistemas de primer orden y primer orden con retardo en el cual se pretende tomar el tiempo cuando los valores de la señal de salida están al 63% y 28% de la variación de la salida, siendo sus parámetros de ganancia, constante de tiempo y tiempo de retardo determinadas por las operaciones matemáticas que se presentan en [38]; las ecuaciones de máxima elongación, tiempo de establecimiento y de tiempo de crecimiento de [7] se las han utilizado para determinar el valor del coeficiente de amortiguamiento, ganancia y frecuencia natural no amortiguada para los sistemas de segundo orden, y el método de Balaguer de [8] para encontrar los parámetros característicos de los sistemas de respuesta inversa.

2.3 IMPLEMENTACIÓN DEL ALGORITMO ACO

Para la implementación del algoritmo de optimización ACO se interpreta cuáles son los parámetros que deben manipularse de acuerdo con el problema a optimizar. De estos depende el resultado de la estimación del modelo, es decir que se encuentre la solución óptima y con un gasto computacional adecuado.

La Figura 2.6 presenta el diagrama de flujo para la implementación del algoritmo ACO, que consta de dos condiciones de parada, una cuando el algoritmo llega a su número máximo de iteraciones y otra cuando la función de costo se reduce al 0.01. Pero en la práctica se justifica la necesidad de una tercera condición; en el caso de que el algoritmo, pasado un cierto intervalo de iteraciones, no realice ningún cambio en el último mejor valor de la función de costo, se pueda detener y presentar esos resultados como los más cercanos a los óptimos porque la inversión de más iteraciones y de tiempo computacional no refleja mejores resultados a los obtenidos con anterioridad.

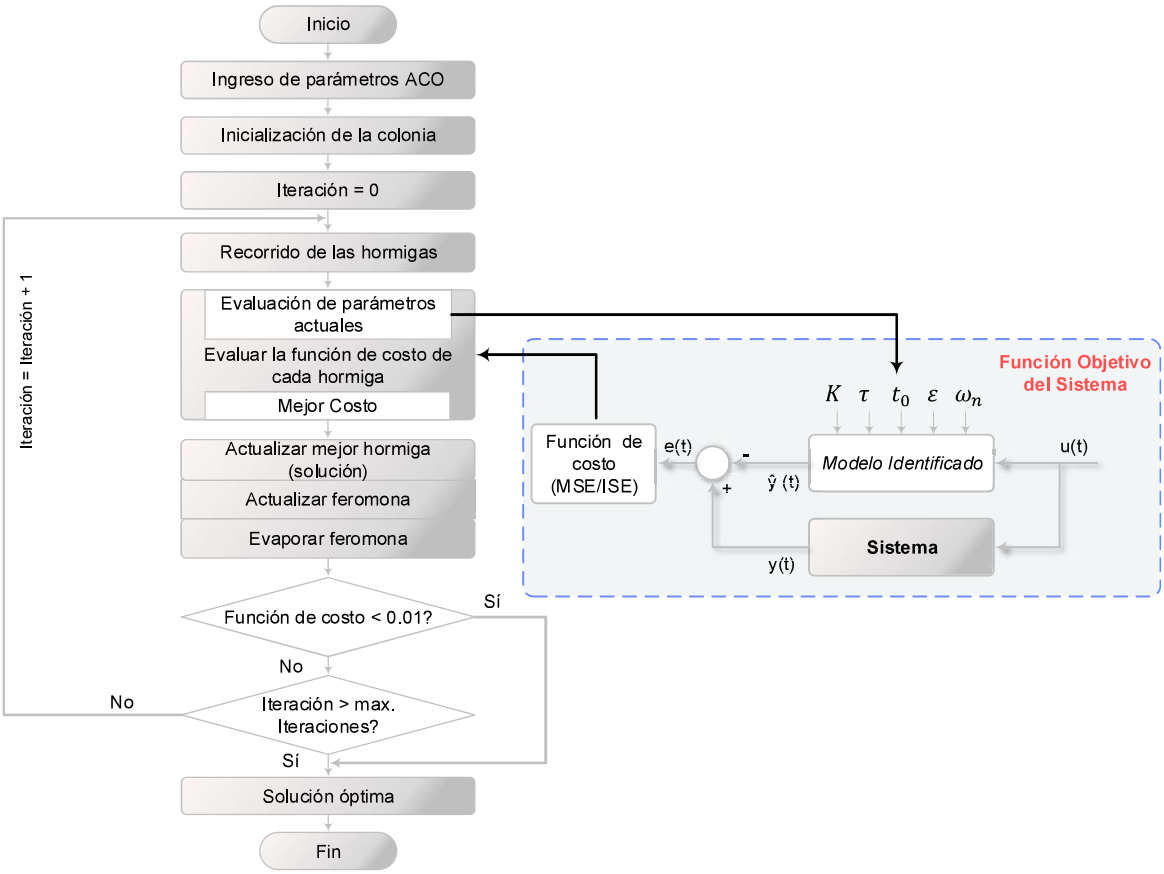


Figura 2.6 Diagrama de flujo en la implementación del ACO.

Por esto se ha establecido una comparación entre el último valor de la función de costo y el anterior, en un número de 40 iteraciones; rango en el cual, si no se ha disminuido la función de costo actual, se establecerá como valor más cercano óptimo al encontrado hasta esa iteración. Esto asegura un mejor rendimiento de SysID en el tiempo de ejecución.

Los parámetros del algoritmo fueron establecidos de acuerdo con la necesidad de los casos de estudio y se los ha determinado en base a prueba y error. De igual manera estos parámetros quedan a consideración del usuario en la herramienta computacional, pero los valores de la Tabla 2.1 son los estimados para una ejecución adecuada en la identificación de modelos y estimación de parámetros para los sistemas tipo presentados anteriormente. Estos valores se los ha impuesto como por defecto en SysID.

Tabla 2.1. Parámetros del algoritmo ACO

Parámetro	Valor
Número de iteraciones	100
Número de hormigas	200
Coeficiente Alfa (α)	0.1
Coeficiente Beta (β)	0.05
Coeficiente de evaporación Rho (ρ)	0.2

Como se mostró en la teoría, la metaheurística de los algoritmos que simulan la inteligencia colectiva se basan en cuatro pilares fundamentales para el procesamiento de información y la emergencia de soluciones. Así, el algoritmo de optimización ACO posee dos parámetros que manipulan estos pilares, estos son: la intensidad de la feromona, α; y la heurística de deseabilidad de un nodo, β. Tales parámetros deben interactuar entre sí de manera que permitan una exploración amplia, pero a la vez puedan converger a una solución óptima a una velocidad adecuada.

2.3.1 DESCRIPCIÓN DE PARÁMETROS

En los párrafos siguientes se detalla la correlación entre estos parámetros y algunos otros que influyen en el desempeño del algoritmo.

- *Número de iteraciones:* es el número de lazos de repetición en la ejecución del recorrido del conjunto de todas las hormigas. Es un parámetro ajustable bajo criterio porque después de un número determinado de iteraciones, es posible que los resultados no mejoren. Sin embargo, demandará más tiempo de ejecución, hecho que ocasiona un mayor gasto computacional, esto es un resultado indeseado para la herramienta.

- *Número de hormigas:* es un parámetro que se lo ha elegido de forma heurística en base a las pruebas realizadas al problema de optimización, siendo que al existir un reducido número de hormigas no existirán los efectos esperados de cooperación, igualmente de haber hormigas en exceso provocará un sistema computacional deficiente.

- *Coeficiente de intensificación,* $\propto$ *:* parámetro ajustable que controla la influencia de la intensidad de feromona; es decir, mientras más alto sea este valor las hormigas prestan mayor atención al rastro de feromona. Sí $\propto = 0$, el coeficiente β es el que actúa, provocando que los nodos más cercanos tengan mayor probabilidad de ser seleccionados.

- *Coeficiente de deseabilidad,* β*:* parámetro ajustable para controlar la heurística de deseabilidad de un nodo, es decir tiene mayor influencia el nodo cuya distancia sea más cercana al nodo actual. Si $\beta = 0$, el único coeficiente que trabaja es $\propto$, esto significa que solo la feromona actúa, llegando a escoger los mismos rastros de feromonas sin importar si el camino es el más corto o no, esto conduce a determinar soluciones subóptimas.

- *Coeficiente de evaporación, ρ:* parámetro ajustable, útil en el proceso de evaporación de la feromona, necesario para olvidar rastros de feromona infructuosos, permitiendo a las hormigas avanzar hacia mejores soluciones. Este coeficiente debe estar en el límite de $0 < \rho \leq 1$.

Es necesario explicar que el espacio de solución (búsqueda) o grafo, dentro del algoritmo implementado, se define como el conjunto de nodos equidistantes delimitados por un límite inferior y un límite superior; estas variables se pueden modificar en la interfaz de la herramienta computacional. También, en el espacio de búsqueda se debe determinar el número de nodos que se desea, en esta aplicación se ha considerado de 1×10^5. Por último, este vector columna de nodos se lo debe repetir de acuerdo con el número parámetros que se desean encontrar, dependiendo del tipo de modelo que se ha de identificar. Con estos cuatro parámetros el espacio de soluciones queda determinado, donde cada uno de los nodos son posibles soluciones al problema de optimización.

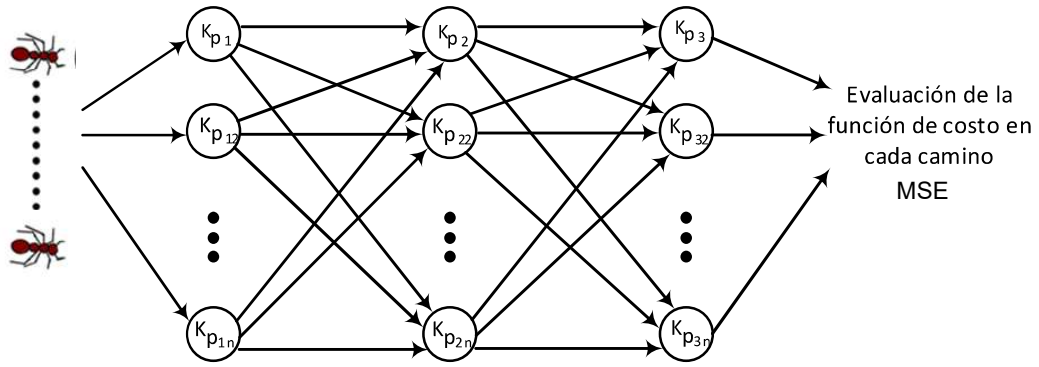


Figura 2.7 Ilustración de la creación del espacio de soluciones o grafo.

En la implementación de los cuatro tipos de modelos de sistemas a identificar cambia el número de parámetros, como los límites del grafo. En la Tabla 2.2. se muestra los sistemas tipo con el número de parámetros a estimar y sus límites correspondientes. En un inicio, estos valores fueron implementados en la herramienta computacional SysID en el algoritmo de optimización ACO.

Tabla 2.2. Variables del espacio de soluciones para el algoritmo ACO.

Tipo de Sistema	Número de Parámetros	Valores Paramétricos por Identificar	Límite Inferior	Límite Superior
Primer Orden	2	$K = 8;\ \tau = 3$	-10	10
Primer Orden con Retardo	3	$K = 8;\ \tau = 3;\ t_0 = 7$	0	10
Segundo Orden	3	$K = 8;\ \xi = 0.2;\ \omega_n = 1$	-10	10
Respuesta Inversa	3	$a\tau = 5;\ b\tau = 2.2;\ \tau = 0.5$	-10	10

Los sistemas lineales descritos en la sección anterior son los sistemas por identificarse y estimar de manera óptima sus parámetros; con excepción del sistema de respuesta inversa, con el cual se trabajó mediante su forma recursiva por el método de Balaguer [8].

En el caso de los sistemas de primer orden con retardo los límites del grafo de soluciones solo se establece valores positivos debido a la constante de tiempo de retardo o tiempo muerto t_0 que no admite números negativos.

Como se expuso en los párrafos anteriores, los modelos tipo sin identificación previa se utilizaron para parametrizar al algoritmo; pero en la práctica para los distintos casos de prueba que se ejecutaron, se realizó una identificación previa; esto ayuda a limitar el espacio de búsqueda a regiones más prometedoras. De los valores determinados por la identificación previa se escoge de cada uno, un rango con holgura de un 70% a 50% del valor determinado, tanto para los límites superior e inferior del espacio de búsqueda, este porcentaje de holgura es para que el algoritmo tenga, a su vez, un espacio de búsqueda lo suficientemente disperso, pero con valores cercanos al óptimo y así pueda realizar su heurística de optimización.

2.4 IMPLEMENTACIÓN DEL ALGORITMO ACO-R

En la implementación del algoritmo ACO-R se consideran algunos aspectos, entre los más importantes están: la definición de la función gaussiana a utilizar, el valor del factor de intensificación de selección y el radio de desviación. Una correcta selección de los valores para estos parámetros acontece en una optimización eficiente con un costo computacional razonable.

En la Figura 2.8 se muestra el diagrama de flujo que sigue la implementación del algoritmo ACO-R, se evidencia que las principales diferencias con el algoritmo original son: la existencia de una evaluación inicial del algoritmo, la reserva de un archivo solución de cardinalidad del tamaño de muestra, la adición de nuevas soluciones de acuerdo al valor de optimización dado por un kernel gaussiano de funciones normalmente distribuidas al archivo solución guardado, la formación de un nuevo archivo solución que se ordena de acuerdo al peso de la función de costo, la eliminación de las soluciones menos óptimas para almacenar igual número de elementos que la población original y de la cual se toman los mejores valores para repetir el todo el proceso de selección de la mejor solución en cada iteración.

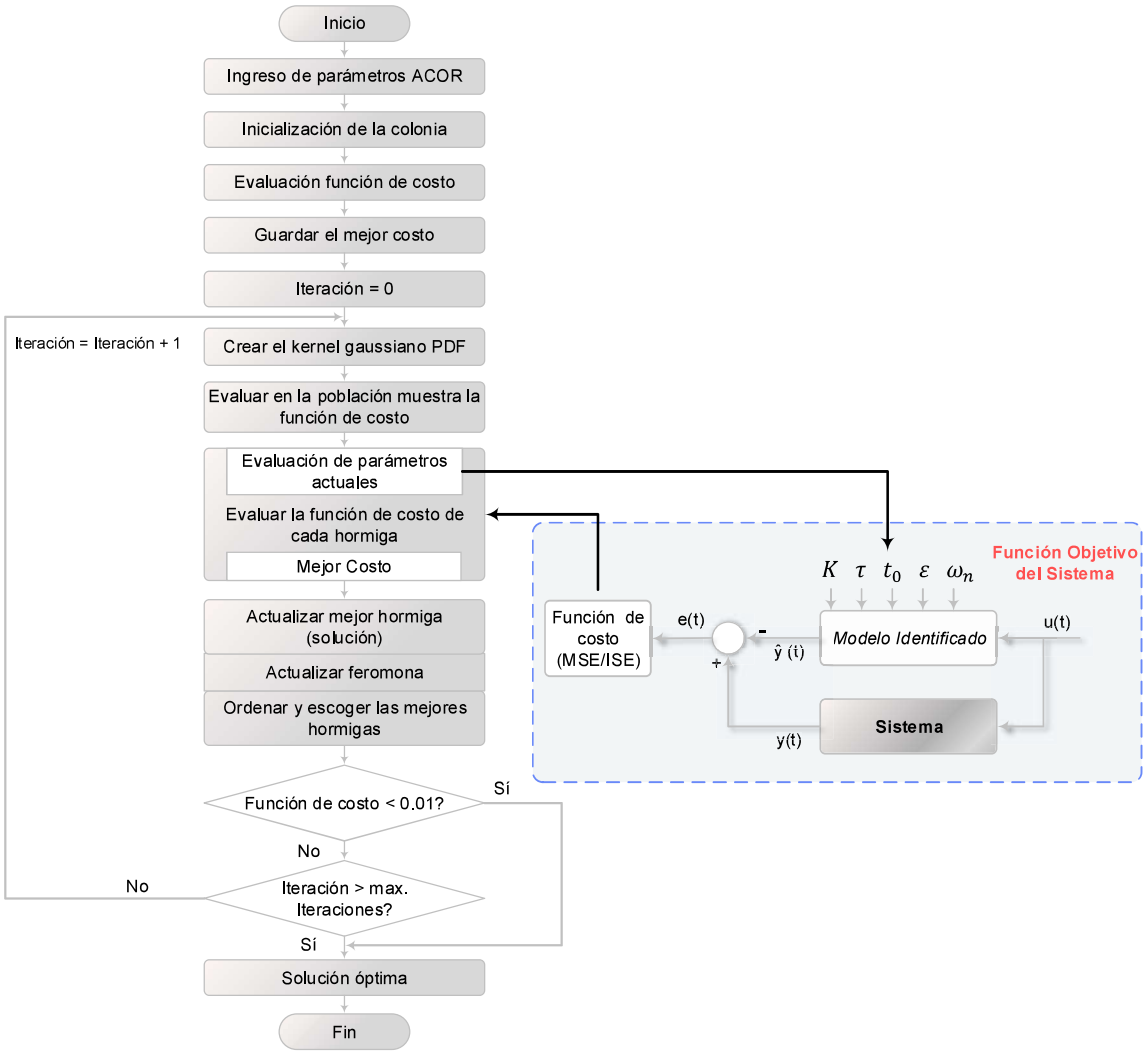


Figura 2.8. Diagrama de flujo de la implementación del algoritmo ACO-R

Como en el caso anterior también se trabaja con una tercera condición de parada.

En todos los casos de optimización es importante que el algoritmo evite quedarse en un óptimo local, y que una vez encontrado el óptimo global pueda converger rápidamente; por ende, necesita una estrategia para diversificarse. El problema radica en que ACO-R no sabe si la región seleccionada contiene un óptimo local o global. Por tanto, debe hacer una suposición inteligente, basándose en la diversificación (mayor robustez), o en la intensificación (mayor velocidad de convergencia- mayor eficiencia). La solución para lidiar con este problema es utilizar dos parámetros que definen el balance entre estos dos criterios, así tenemos al factor de intensificación de selección q y a la relación de la distancia de desviación ξ [10].

En la descripción de parámetros se trata a fondo los anteriormente mencionados, y algunos otros que son necesarios para afianzar el comportamiento del algoritmo en la estimación e identificación óptima de los modelos de sistemas.

Los parámetros establecidos para el funcionamiento adecuado del algoritmo ACO-R se presentan en la Tabla 2.3, los cuales han sido seleccionados heurísticamente para los casos tipo presentados en la sección anterior, de igual manera la modificación de estos parámetros queda a consideración del usuario en la herramienta computacional. Sin embargo, los valores por defecto que se han establecido son los que se presentan en la Tabla 2.3.

Tabla 2.3. Parámetros del algoritmo ACO-R

Parámetro	Valor
Número de iteraciones	200
Número de hormigas	20
Factor de intensificación (q)	0.5
Relación de la distancia de desviación (ξ)	1
Tamaño de muestra	60

Otra de las diferencias fundamentales en la implementación del algoritmo ACO- R, es que el factor de intensificación, q; es igual al balance entre la diversificación que resulta en mayor robustez, y la intensificación que se desempeña como una mayor velocidad de convergencia. Por tanto, se ha escogido un coeficiente de alrededor del 0,5 considerando como valor máximo a 1; esto con la finalidad de poseer en igual proporción los dos beneficios que representa este parámetro. El coeficiente de relación de la distancia de desviación, ξ, influye en como la memoria de largo plazo es utilizada; es decir, la búsqueda es menos sesgada en los espacios que ya se han explorado en el archivo solución. Por tanto, este parámetro también influye en la velocidad de convergencia del archivo solución.

2.4.1.1 DESCRIPCIÓN DE PARÁMETROS

La descripción de cada parámetro se presenta en el siguiente apartado. Esto servirá como guía para la modificación de los parámetros del algoritmo por parte del usuario.

- *Número de iteraciones:* el número de iteraciones en la aplicación queda al criterio del usuario, pero se recomienda que sea en un número que beneficie al costo computacional y que esté de acuerdo con el resultado obtenido de las pruebas realizadas; en este trabajo se ha considerado 200 iteraciones como máximo procedente de las pruebas realizadas. Además, la eficiencia del algoritmo depende de los otros parámetros que son más influyentes.

- *Número de hormigas:* representa al tamaño de la población y es la cardinalidad mínima que debe poseer la matriz tamaño de muestra para lograr una ejecución adecuada del algoritmo. Al igual que en los anteriores algoritmos una cantidad

exagerada de miembros provoca ineficiencia por parte del algoritmo, pero da mayor veracidad de respuesta; mientras que una cantidad pequeña se traduce en una velocidad de convergencia más rápida, pero hay la posibilidad de caer en un mínimo local. Tanto para el cálculo del peso de solución como el de la desviación estándar es un factor influyente, porque está en relación inversa con cada uno de estos.

- *Factor de intensificación de selección,* q*:* también llamado peso de selección es el parámetro diseñado para controlar la diversificación del proceso de búsqueda. El cálculo del peso para cada función gaussiana depende de este factor, así mientras es más pequeño, las soluciones mejor clasificadas son altamente preferidas; al contrario, si es grande, el valor de probabilidad se vuelve uniforme para todo el rango de soluciones logrando que la búsqueda sea más diversificada y el algoritmo actúe con mayor robustez. Sin embargo, una mayor robustez generalmente significa una menor eficiencia, es decir una velocidad de convergencia más lenta. Cuando $q = 0$, significa que solo la función gaussiana asociada con la mejor solución encontrada hasta ese momento es usada para generar las próximas soluciones [10]. Por tanto, el valor seleccionado es de 0.5.

- *Relación de la distancia de desviación,* ξ*:* el comportamiento de este parámetro se asemeja a la taza de evaporización de feromona en el algoritmo ACO cuando es mayor a cero. Mientras mayor es su valor, menor es la velocidad de convergencia del algoritmo y viceversa. Además, influye en cómo se utiliza a la matriz de muestra o también llamada matriz de soluciones anteriores, haciendo que la búsqueda sea menos sesgada en los puntos del espacio de búsqueda que ya fueron explorados.

- *Tamaño de muestra:* junto con la taza de aprendizaje (factor de intensificación de selección, q) es uno de los parámetros más influyentes en la robustez del algoritmo, es decir mientras más lenta es la taza de aprendizaje y cuanto mayor es el tamaño del archivo solución más robusto es el algoritmo, pero menor es la velocidad de convergencia. Por tanto, el tamaño del archivo solución es de 20 hormigas y el tamaño de muestra es de 60 para no tener un tamaño exagerado y se puede tener un balance entre velocidad y robustez.

Como se mencionó en un principio un concepto importante para la ejecución del algoritmo ACO-R es el kernel gaussiano de funciones pdf, que para esta aplicación se ha considerado a la función normal de distribución o también llamada campana de Gauss. Con la modificación de que para el parámetro de tiempo muerto t_0, de sistemas de primer orden con retardo; al existir en la distribución normal de valores negativos, estos no son

admisibles computacionalmente para este parámetro. Por tal razón, se ha modificado a dicha distribución por una aleatoria positiva exclusiva para este parámetro evitando que adopte valores negativos. Esta modificación está dentro de la definición del kernel por tanto se puede aplicar en la implementación del algoritmo.

Al igual que el algoritmo anterior, para ACO-R también se define un espacio de búsqueda, con la diferencia que es uniforme y de valores positivos; porque las nuevas posiciones alcanzadas por el algoritmo se basan en la suma de la distancia promedio entre las soluciones. Estas soluciones provienen del cálculo de la desviación estándar, que es la distancia promedio entre la solución seleccionada y las soluciones en el archivo, multiplicada por el coeficiente de la relación de desviación, Ecuación 1.27. La probabilidad generada al muestrear la función normal de distribución, que trabaja con valores positivos como negativos, sirve para encerrar el espacio de búsqueda entre los valores cercanos al punto óptimo global. Esto se muestra en la Figura 2.9, donde se evidencia que la generación de resultados que satisfacen la necesidad del problema solo depende de la heurística del algoritmo.

En la Figura 2.9 se observa cómo se genera las funciones de densidad de probabilidad; así, se tiene un espacio de búsqueda de valores cercanos al óptimo dada por una distribución inicial de los costos. Supongamos que las primeras hormigas al evaluarse en la función de costo, que se llamará para esta explicación U, resultan en una población P que tiene un óptimo inicial en O, que sería en el caso del algoritmo ACO-R la población de muestra.

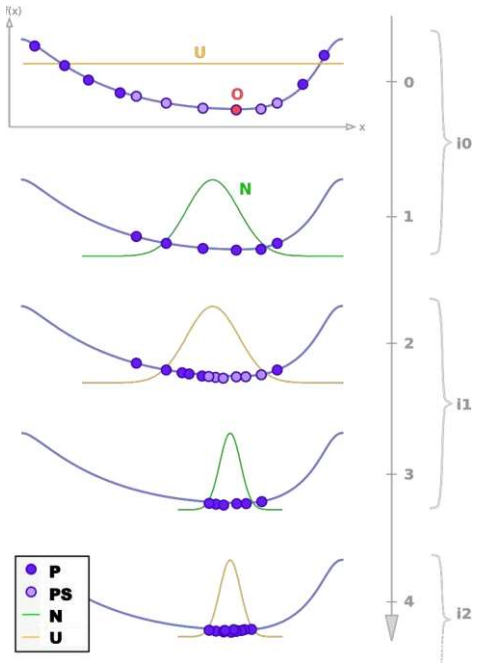


Figura 2.9 Espacio de búsqueda del algoritmo ACO-R.

De la población de muestra se escogen las mejores soluciones PS, que servirán para generar las nuevas funciones de densidad de probabilidad gaussianas, denominadas en la gráfica N. Con cada iteración la pdf centra su distribución alrededor del óptimo inicial, resultando en escoger el mejor óptimo al finalizar la ejecución del algoritmo.

Las variables que intervienen en la definición del espacio de búsqueda son: el número de parámetros a identificar para cada tipo de modelo de sistema, el límite superior e inferior del espacio de búsqueda, en la Tabla 2.4 se muestra dichos valores, los mismos fueron implementados, en un inicio, en SysID.

Tabla 2.4 Variables para la creación del espacio de búsqueda para el algoritmo ACO-R.

Tipo de Sistema	Número de Parámetros	Valores Paramétricos por Identificar	Límite Inferior	Límite Superior
Primer Orden	2	$K = 8;\ \tau = 3$	0.1	10
Primer Orden con Retardo	3	$K = 8;\ \tau = 3;\ t_0 = 7$	0.1	10
Segundo Orden	3	$K = 8;\ \xi = 0.2;\ \omega_n = 1$	0.1	10
Respuesta Inversa	3	$a\tau = 5;\ b\tau = 2.2;\ \tau = 0.5$	0.1	10

Los valores del límite inferior y superior de la tabla anterior fueron iniciales y específicos para los sistemas tipo de los casos de estudio. Por tanto, para otros casos se realiza una identificación previa, para delimitar el espacio de búsqueda, debido a que, si el espacio de búsqueda está únicamente entre los límites anteriores, no sería un espacio de búsqueda coherente con los diversos casos de plantas que existen.

Lo que se pretende es optimizar los valores ya identificados previamente para que sean los más cercanos a los óptimos, en los cuales la función de costo es la mínima; como también tener beneficios adicionales como la reducción en el tiempo de ejecución del algoritmo, que a pesar de que sea un algoritmo de rápida convergencia, al hacer esta modificación se ha visualizado una mejora en su rendimiento. Además, como ya se dijo, estos parámetros previamente identificados se les ha proporcionado una holgura del 70% al 50%, con el fin de que el algoritmo también cumpla con su papel de optimizar.

2.5 IMPLEMENTACIÓN DEL ALGORITMO PSO

En la implementación del algoritmo de optimización PSO, al igual que en la implementación de los anteriores, se considera modificables los parámetros del algoritmo que dependen del problema, porque en base a estos se puede afinar la búsqueda de la mejor solución y disminuir el gasto computacional de la herramienta desarrollada.

La Figura 2.10 muestra el flujograma de la implementación del algoritmo PSO para la identificación de modelos de sistemas y estimación de parámetros.

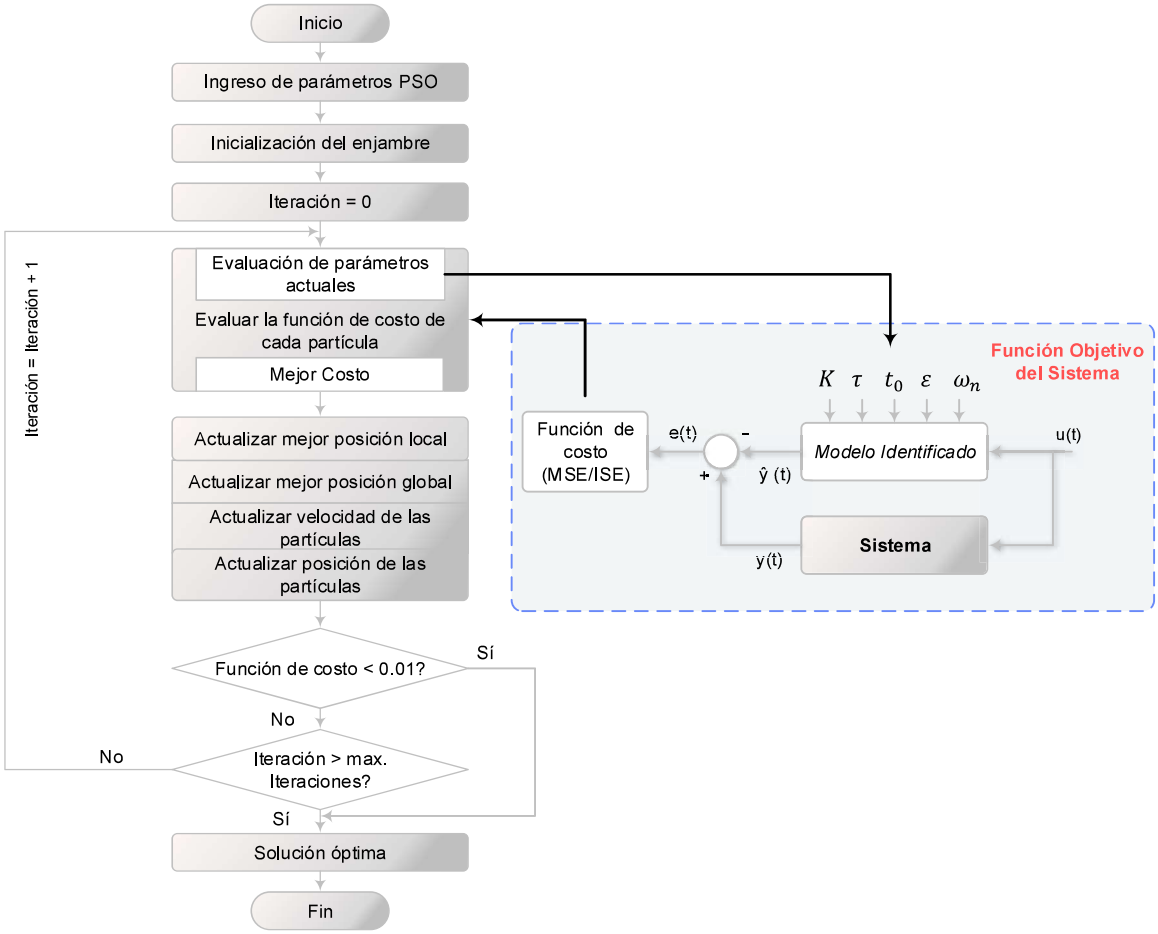


Figura 2.10. Diagrama de flujo en la implementación del PSO.

A diferencia de los dos algoritmos anteriores el algoritmo de optimización PSO es evolutivo y no constructivo; por tanto, existen algunas discrepancias en la definición de parámetros y criterios, además de que están bio-inspirados en diferentes ambientes como se expone en la teoría. En la elaboración de este proyecto de titulación, se escoge los valores adecuados de estos parámetros en correspondencia a los casos de estudio planteados, mediante prueba y error. No obstante, en la herramienta computacional quedan de libre discernimiento del usuario, pero como valores por defecto se presentan los de Tabla 2.5.

Tabla 2.5. Parámetros del algoritmo PSO

Parámetro	Valor
Número de iteraciones	200
Tamaño del enjambre	40
Coeficiente del componente cognitivo (c_1)	1.5
Coeficiente del componente social (c_2)	2
Peso de Inercia	1

El objetivo de este algoritmo es buscar de forma eficiente el mejor espacio de solución, como se muestra en la Figura 2.11, mediante la dirección de las partículas hacia la mejor solución encontrada de las iteraciones anteriores, con la intención de encontrar una única solución que para este caso práctico, minimiza la función de costo al final del proceso de iteraciones siendo está la solución conocida como óptimo global.

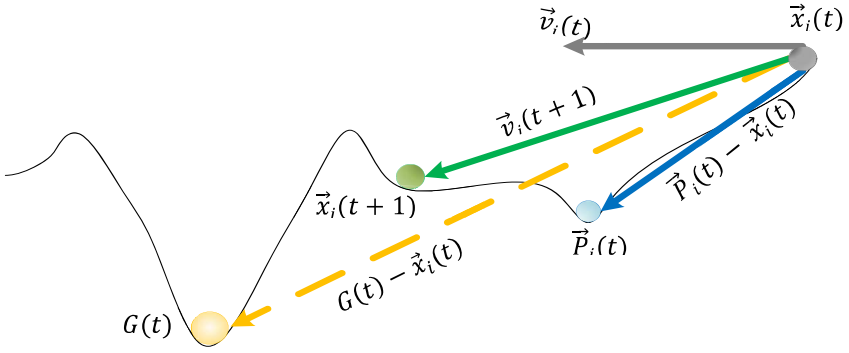


Figura 2.11. Evolución del algoritmo PSO.

Al igual que en los casos anteriores una sola condición de finalización del algoritmo no basta, se necesita una condición extra que resguarde el resultado y mejore el tiempo computacional invertido en la identificación. Así se añade la condición de disminuir al 0,01 de la función de costo, que es el índice MSE entre el modelo real y el modelo identificado; lo cual es práctico para los casos que ya se ha encontrado una buena estimación a pesar de que el algoritmo se encuentre aún en las primeras iteraciones.

Pero qué pasa si el algoritmo no encuentra una solución adecuada que satisfaga a la condición de para anterior, deberá esperar que se realice todo el número de iteraciones establecido en la Tabla 2.5, para que finalice el proceso de optimización. Entonces, porque no se desea un gasto infructuoso de tiempo de ejecución en el algoritmo, se establece una tercera condición en la cual, si pasado un margen de 40 iteraciones no se ha mostrado un cambio en la función de costo, se detiene el algoritmo y finaliza el proceso de optimización.

En las pruebas realizadas se comprobó que el margen de 40 iteraciones era el necesario, debido a que un valor mayor de iteraciones el cambio en la minimización de la función de costo no refleja el tiempo computacional invertido, como también el valor de los parámetros optimizados son iguales a los que se habían obtenido antes. Un valor menor de iteraciones por el contrario se presentaba el riesgo de que el algoritmo pueda seguir optimizando y que fue innecesariamente truncado.

2.5.1 DESCRIPCIÓN DE PARÁMETROS

A continuación, se redacta una descripción detallada de los criterios de selección considerados para definir los parámetros del algoritmo que intervienen en el proceso de optimización.

- *Número de iteraciones:* dependen del problema, muy pocas iteraciones terminan la búsqueda muy rápido, muchas iteraciones alargaran el tiempo de búsqueda y el procesamiento computacional, sin llegar a buenos resultados. Mediante las pruebas realizadas se ha escogido un número de 200 iteraciones, por presentar el mejor tiempo de ejecución como de minimización de la función de costo.

- *Tamaño del enjambre:* determina el número de partículas en el enjambre, a mayor cantidad de miembros mejores son los resultados, sin embargo, el tiempo computacional también incrementa; al realizar las pruebas se seleccionó el doble de la cantidad de partículas con que finalmente se trabajó, evidenciándose un costo computacional exagerado y generándose el mismo resultado con el tamaño de 40 partículas, al igual que al seleccionarse un tamaño menor del enjambre no se llegó al mismo resultado de optimización.

- *Coeficientes de aceleración:* son el componente cognitivo c_1 y el componente social c_2. Si $c_2 > c_1$ la partícula es atraída con más fuerza hacia su mejor posición, en cambio si $c_2 < c_1$ la partícula es atraída con mayor fuerza hacia la posición global del enjambre, en cambio cuando $c_2 = c_1$ la partícula se dirige al promedio entre el mejor global y la mejor posición personal. Por tanto, de las pruebas realizadas se determina que $c_1 = 1.5$ y $c_2 = 2$, son los adecuados para conseguir un balance en la diversificación del algoritmo.

- *Peso de inercia:* es el valor que se agrega a la velocidad actual, está presente en la construcción del vector de actualización de la velocidad. Por lo general se mantiene en un valor igual a la unidad y es muy importante porque de este depende que el algoritmo converja más rápido a una solución óptima.

Para el algoritmo de optimización PSO el espacio de búsqueda es el mismo en los cuatro tipos de sistemas a identificar, es decir es positivo y con límites superior e inferior, los mismos que sirven para limitar la velocidad de las partículas. Este espacio de búsqueda también está limitado por el número de parámetros que se desean optimizar dependiendo del tipo de sistema; así, se tiene espacios de búsqueda de 2 a 3 vectores columna. Al igual

que en los otros algoritmos el número de parámetros está determinado por el tipo de sistema que se requiera identificar y estimar sus parámetros.

Tabla 2.6 Variables para la creación del espacio de búsqueda del algoritmo PSO.

Tipo de Sistema	Número de Parámetros	Valores Paramétricos por Identificar	Límite Inferior	Límite Superior
Primer Orden	2	$K = 8;\ \tau = 3$	0.01	10
Primer Orden con Retardo	3	$K = 8;\ \tau = 3;\ t_0 = 7$	0.01	10
Segundo Orden	3	$K = 8;\ \xi = 0.2;\ \omega_n = 1$	0.01	10
Respuesta Inversa	3	$a\tau = 5;\ b\tau = 2.2;\ \tau = 0.5$	0.01	10

Limitar la velocidad con la que pueden moverse las partículas en el espacio de solución genera que las partículas no se dispersen de manera que no puedan convergen a un valor cercano al punto óptimo. Un criterio adicional que mejora el desempeño del algoritmo PSO es añadir un coeficiente de amortiguamiento para el peso de peso inercia, con el fin de que conforme pasen las iteraciones la inercia disminuya en la actualización de la velocidad logrando que los coeficientes del componente cognitivo como del componente social tengan más peso; esto ocurre al llegar a los últimos lazos de iteración cuando mejor es la información que se posee de estos.

Al igual que en los algoritmos anteriores, en PSO también se trabaja con una identificación previa, porque como ya se dijo no se cumple autónomamente la identificación de sistemas si el rango de búsqueda queda delimitado en los límites de la Tabla 2.6; porque cada parámetro característico está dentro de un rango numérico de diferente valor. A razón de esto, el espacio de búsqueda debe ser de límites diferentes para cada parámetro, para que se encuentren en una región con valores prometedores donde se halle el valor más cercano al óptimo. Los nuevos límites están entre el 70% al 50% del valor previo identificado, esto con el fin de que exista un espacio de búsqueda lo suficiente extenso para que se realice la metaheurística de optimización para cada parámetro; con esto se logra un mejor rendimiento en el tiempo de ejecución del algoritmo puesto que posee un espacio de búsqueda más prometedor.

2.6 DISEÑO DE LA INTERFAZ GRÁFICA

La interfaz gráfica se ha creado en la herramienta computacional App Designer® de software de MatLab®, bajo la consideraciones de la norma ISA 101 para el desarrollo de interfaces hombre máquina [39].

Figura 2.12 Interacción del usuario con la interfaz y las aplicaciones de MatLab®.

La interfaz desarrollada responde al diagrama de flujo presentado en la Figura 2.13. Como se la ha realizado en una visualización gráfica en forma de pestañas, la primera pestaña que se observa es la de Presentación, las siguientes corresponden a las opciones de identificación por el algoritmo ACO, por el algoritmo ACO-R o por el algoritmo PSO.

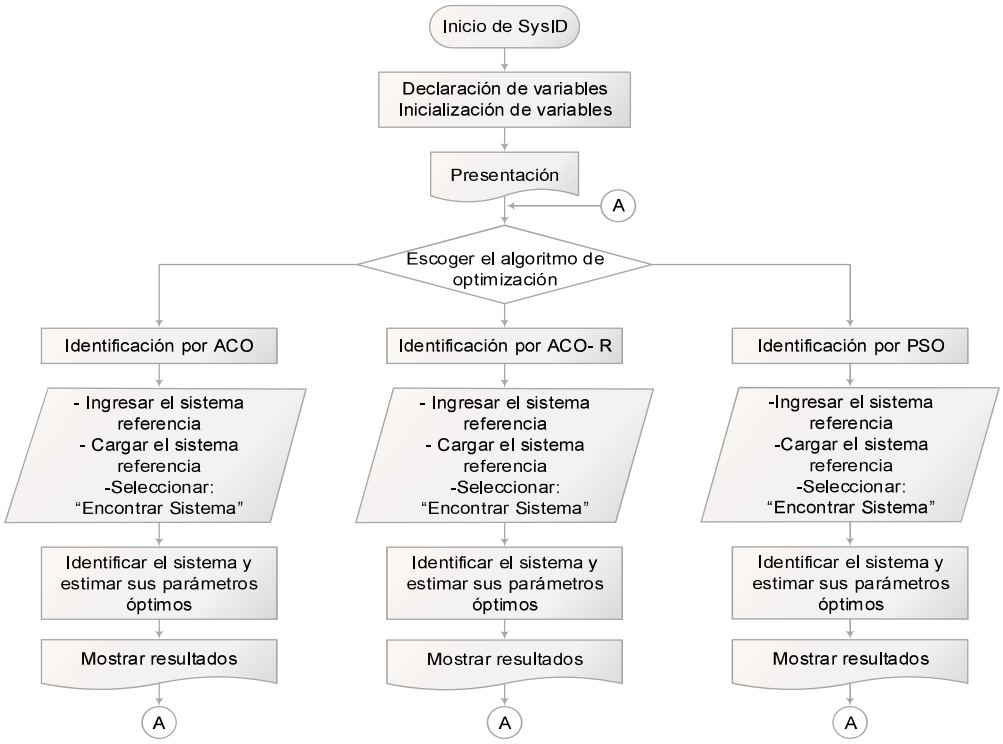


Figura 2.13. Diagrama general del funcionamiento de SysID.

En la Figura 2.14 se muestra la pestaña de Presentación de la herramienta computacional y la de identificación por el algoritmo ACO-R. Se muestra la distribución para la presentación de resultados, tanto de los parámetros optimizados, la función de transferencia del sistema, las gráficas de evolución del error y la del modelo identificado.

En la herramienta computacional diseñada también se visualiza el tiempo de ejecución del algoritmo en minutos, el número de iteraciones invertidas en la identificación, el error relativo de la identificación, el índice de error cuadrático y de error medio cuadrático. Igualmente posee la opción de guardar los resultados en archivos .fig o .svg, como la captura de pantalla de los resultados obtenidos y la presentación de un gráfico de barras para la comparación de resultados obtenidos.

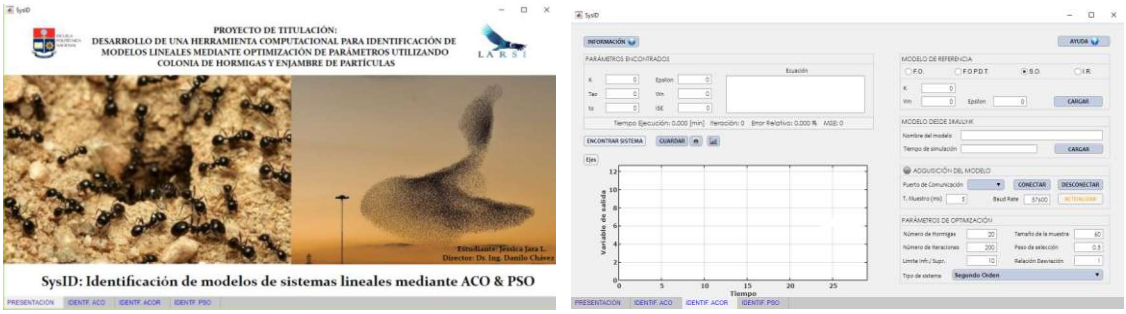


Figura 2.14. Visualización de la interfaz gráfica. Pestañas: Presentación e Identificación mediante ACO.

La herramienta computacional cuenta con los botones de: información y ayuda. En el primero se despliega la teoría básica sobre los algoritmos de optimización, con el fin de facilitar un concepto y conocimiento general del alcance de los algoritmos de optimización bio- inspirados implementados en la herramienta computacional. En el segundo botón se encuentra la respuesta a las principales interrogantes que pueden surgir en el manejo de la herramienta, esto con la finalidad de brindar una mejor experiencia al usuario.

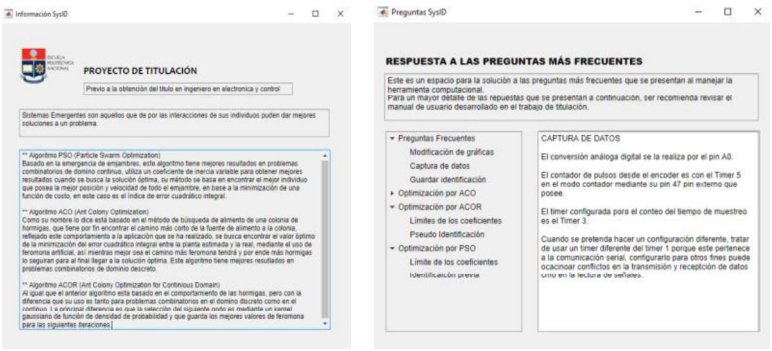


Figura 2.15. Visualización de las ventanas de Información SysID y Preguntas SysID.

La Figura 2.16. muestra la distribución de los componentes de visualización para la presentación de los de parámetros del modelo identificado y el ingreso de datos referencia de forma manual. Asimismo, la herramienta cuenta con la integración de modelos desde Simulink® tanto para identificar el tipo de sistema como para la estimación de sus parámetros con valores cercanos a los óptimos. También existe la opción de adquisición de datos de referencia de forma serial por medio de una tarjeta embebida para los sistemas reales, que en las pruebas realizadas fue para el caso práctico de identificación del modelo de sistema de un motor DC.

La presentación de resultados se habilita, dependiendo del tipo de modelo de sistema identificado, los espacios correspondientes para la visualización de los parámetros optimizados y en el espacio de Ecuación se muestra la función de transferencia del sistema identificado.

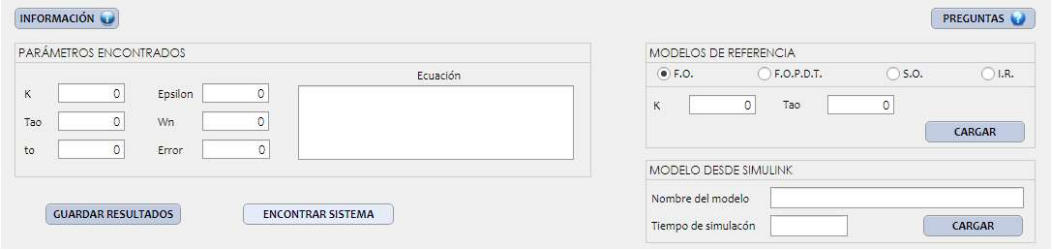


Figura 2.16. Visualización de los resultados y la entrada de los parámetros de referencia.

El entorno gráfico de visualización de la herramienta computacional se muestra en la Figura 2.17., que es donde se presenta la curva resultante de la identificación del modelo. También se cuenta con la opción Graficar error, que al ser seleccionado gráfica el índice de error por iteración en la identificación del modelo y la estimación de los parámetros más cercanos a los valores óptimos.

El panel de parámetros de optimización contiene los parámetros modificables de los algoritmos de optimización implementados, que están a libre criterio y necesitad del usuario. Igualmente existe la posibilidad de escoger el tipo de sistema al cuál se desea aproximar el sistema referencia, al seleccionarlo la herramienta solo cumple con la función de estimar los parámetros a valores muy cercanos a los óptimos del tipo de modelo de sistema a identificar; o a su vez se puede escoger la opción encontrar la mejor identificación dejando a la herramienta aproximar al mejor resultado obtenido en base a la comparación del menor índice de error, en un inicio ISE pero en base a las pruebas se toma al índice MSE para el proceso de identificación.

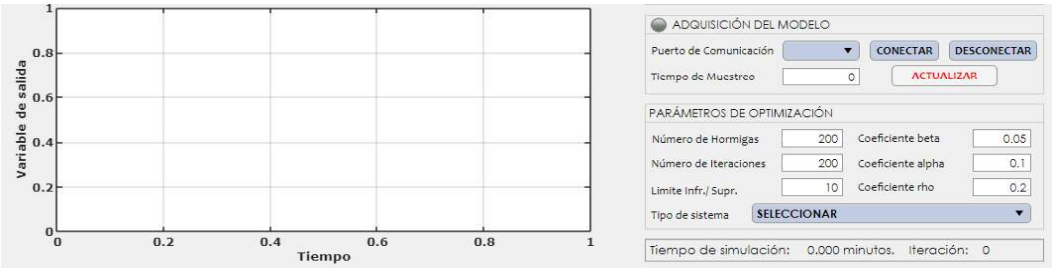


Figura 2.17. Visualización de la gráfica de identificación, del tiempo de simulación, la adquisición de datos y los parámetros modificables de los algoritmos.

En la Figura 2.18 se observa el diseño general de la herramienta computacional SysID para identificación de sistemas lineales mediante algoritmos de optimización ACO, ACO-R y PSO. En el diseño de la herramienta se ha considerado que el producto final sea de uso sencillo, con una interfaz gráfica que permita el acceso los parámetros de los algoritmos para que se puedan modificar en correspondencia con las necesidades de los sistemas a

identificar o a la necesidad del usuario, en un software de buen rendimiento computacional y un hardware fácil de manipular en la adquisición de datos para esta aplicación.

Figura 2.18 Diseño de la herramienta computacional SysID.

La Figura 2.19. muestra el diagrama de flujo completo correspondiente a la herramienta computacional SysID, la misma se ha realizado en base en la teoría del proceso de identificación de sistemas presentado en el capítulo anterior.

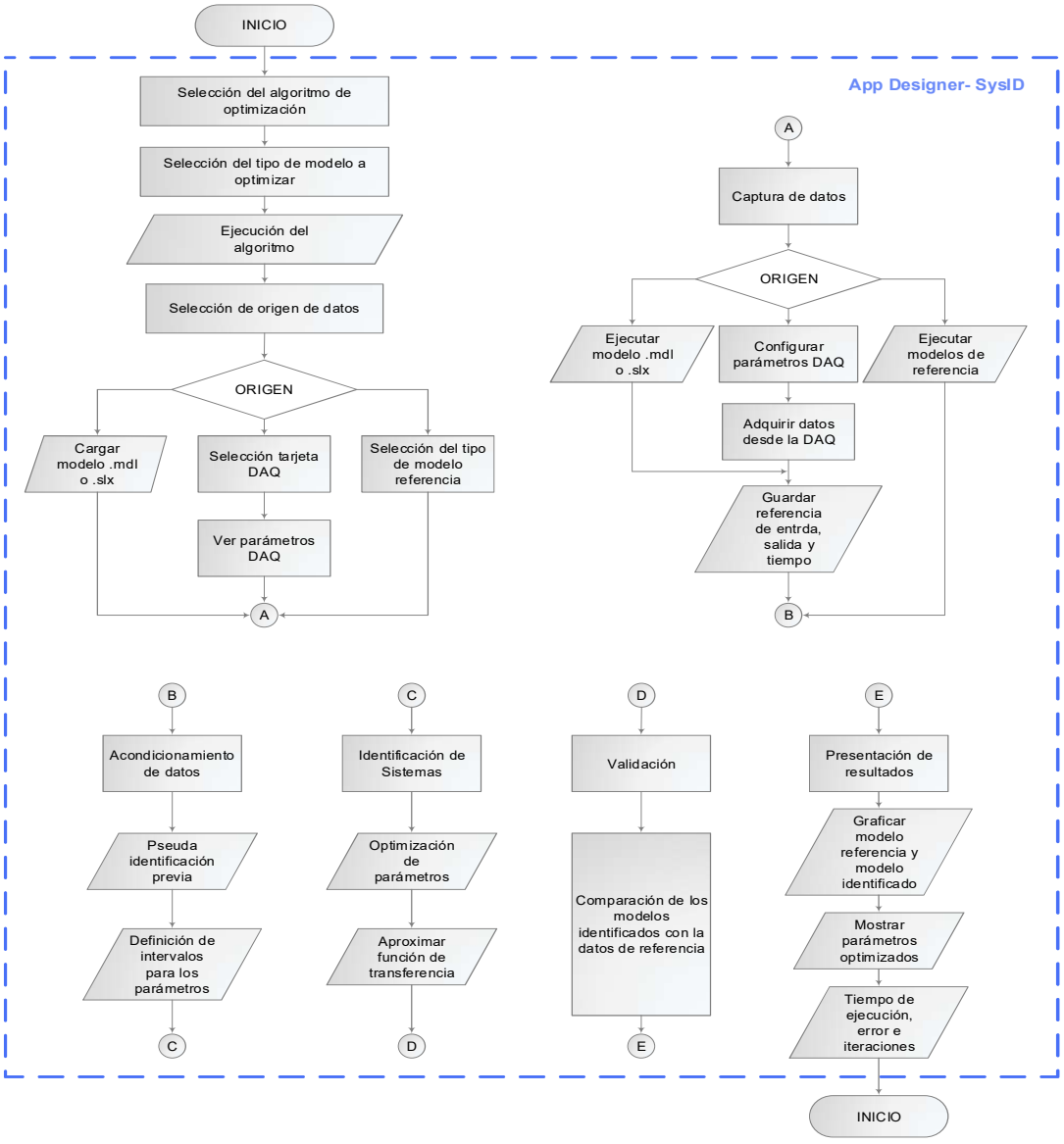


Figura 2.19. Diagrama de flujo de SysID.

En los siguientes párrafos, se describen los diagramas de flujo de cada algoritmo implementado dentro del entorno de App Designer, con el propósito de entender la secuencia de ingreso de datos para la identificación de sistemas mediante SysID.

Cada uno de los algoritmos de optimización utilizados tienen diagramas de flujo semejantes, para su implementación dentro de la herramienta computacional. La programación de la herramienta es en forma secuencial y ordenada por medio de funciones. A continuación, se muestra el diagrama de flujo para el algoritmo de optimización por colonia de hormigas.

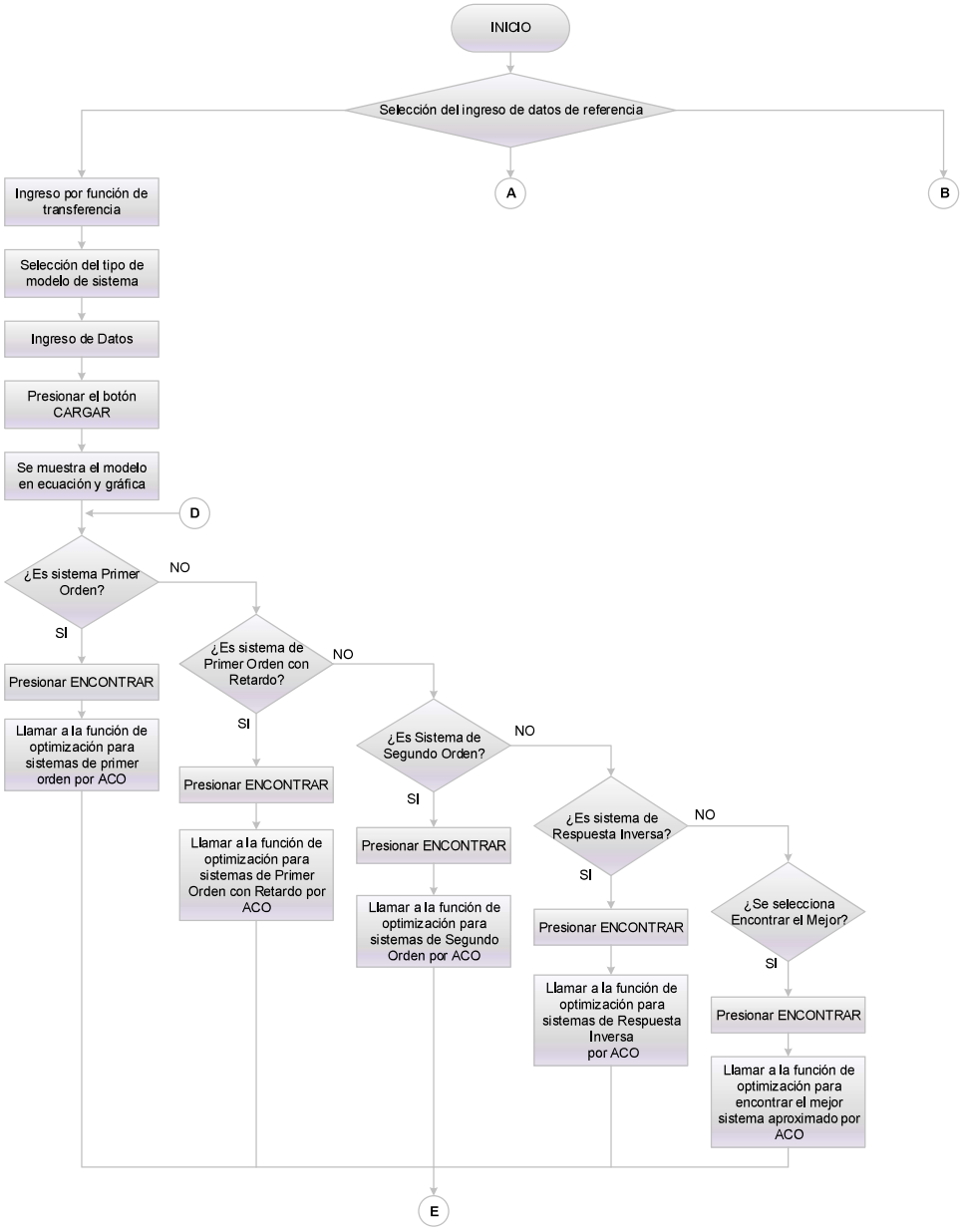


Figura 2.20. Diagrama de flujo de selección del modelo referencia por SysID.

En la Figura 2.20 se muestra la selección de los datos de referencia ingresando la función de transferencia, esto con el fin de realizar las pruebas simuladas para los sistemas tipo a identificar. Existen dos formas adicionales para el ingreso de los datos referenciales: una es desde modelos en Simulink® y la otra desde la adquisición de datos mediante la comunicación serial con un sistema embebido.

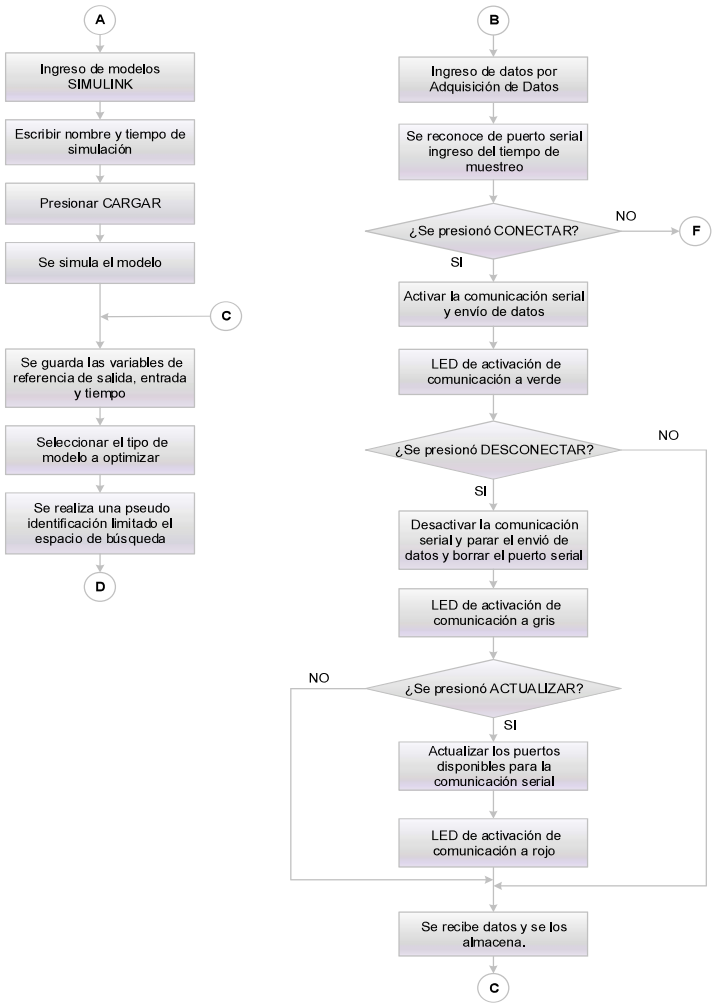


Figura 2.21. Diagrama de flujo de dos tipos de ingreso de sistemas referencia.

Con el uso de estas dos formas de adquisición de datos de referencia, se expande el beneficio que presenta SysID porque se permite encontrar la función de transferencia de un modelo no lineal que pueda reducir su dinámica a un sistema lineal de primer orden con o sin retardo, segundo orden o respuesta inversa en un punto de operación y bajo condiciones iniciales conocidas. Esto convierte a la herramienta computacional desarrollada en un método de verificación de las técnicas convencionales de identificación de sistemas y reducción de modelos como son el análisis de la repuesta de un sistema a una señal paso o por cálculo geométrico. Para lograr este objetivo a los datos adquiridos se les realiza una identificación inicial para aproximar y limitar el rango de valores del

espacio de búsqueda en el cual se encuentran los valores más cercano a los óptimos de cada parámetro del modelo identificado; al igual que ayuda en la ejecución del algoritmo porque disminuye la cantidad de iteraciones procesadas, reduciendo notablemente el tiempo de búsqueda de los valores cercanos a los óptimos, como muestra la Figura 2.22.

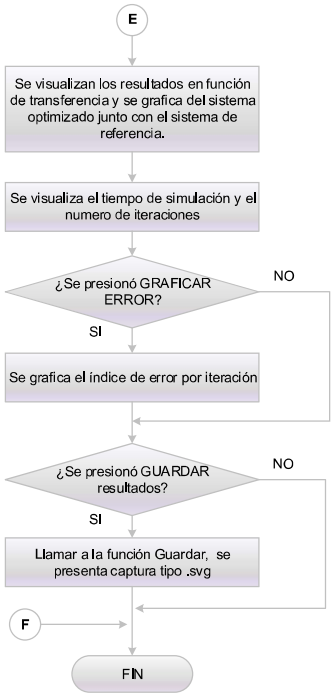


Figura 2.22. Diagrama de flujo de la presentación y almacenamiento de resultados.

Finalmente, se presenta los resultados de la identificación y de la optimización a la estimación de parámetros de los sistemas, como se muestra en el diagrama de flujo de la Figura 2.22., en el área de gráficas se muestra el sistema identificado junto con el sistema referencia con sus respectivas etiquetas, como también se muestra los valores considerados como óptimos y el resultado en función de transferencia. Además, se presenta el número de iteraciones realizado y el tiempo transcurrido hasta encontrar los valores cercanos a los óptimos de los parámetros del sistema identificado. También incluye una opción de selección de gráfica del error por iteración que se obtuvo en la estimación de parámetros, esta opción se habilita una vez se haya mostrado los resultados de la identificación. Igualmente, la herramienta SysID presenta una opción de captura de datos, que se la realiza por captura de pantalla de los datos de identificación y la gráfica de identificación en formato de imagen .svg.

En el algoritmo de optimización en por colonia de hormigas en tiempo continuo, el diagrama de flujo es semejante al presentado anteriormente y posee las mismas funcionalidades.

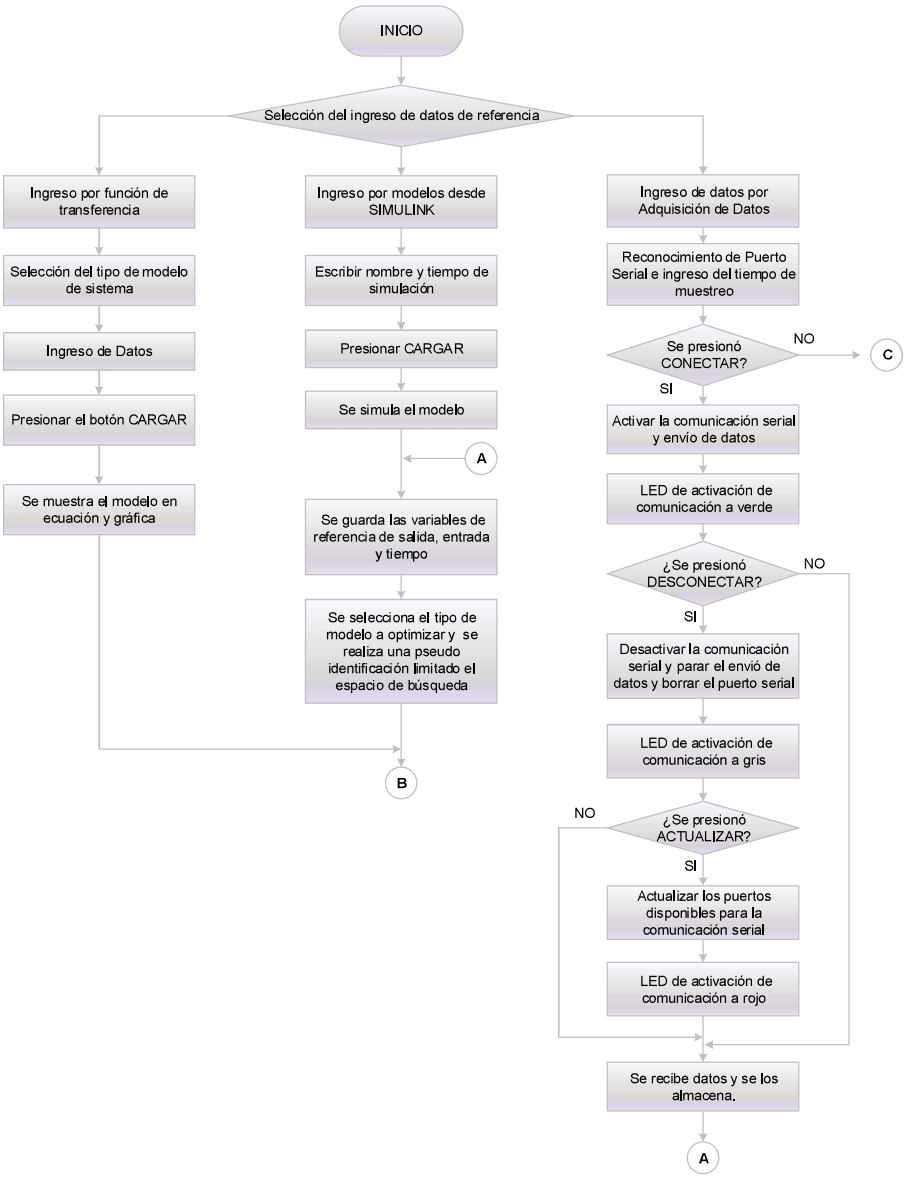


Figura 2.23. Diagrama de flujo del algoritmo ACO-R para el ingreso de datos de referencia.

Como lo muestra la Figura 2.23 la funcionalidad del ingreso de datos es la misma que la descrita anteriormente en el algoritmo ACO, pero cabe destacar que en la adquisición de datos posee tres botones el primero: CONECTAR para habilitar puertos de comunicación serial, envío de datos de entrada, de salida y de tiempo de muestreo; un segundo botón: DESCONECTAR para deshabilitar el puerto serial una vez finalizada la adquisición de datos y por último un tercer botón: ACTUALIZAR que se lo utiliza para reconocer los puertos de comunicación serial disponibles luego de una reconexión de la tarjeta de adquisición o para el reconocimiento de nuevas tarjetas.

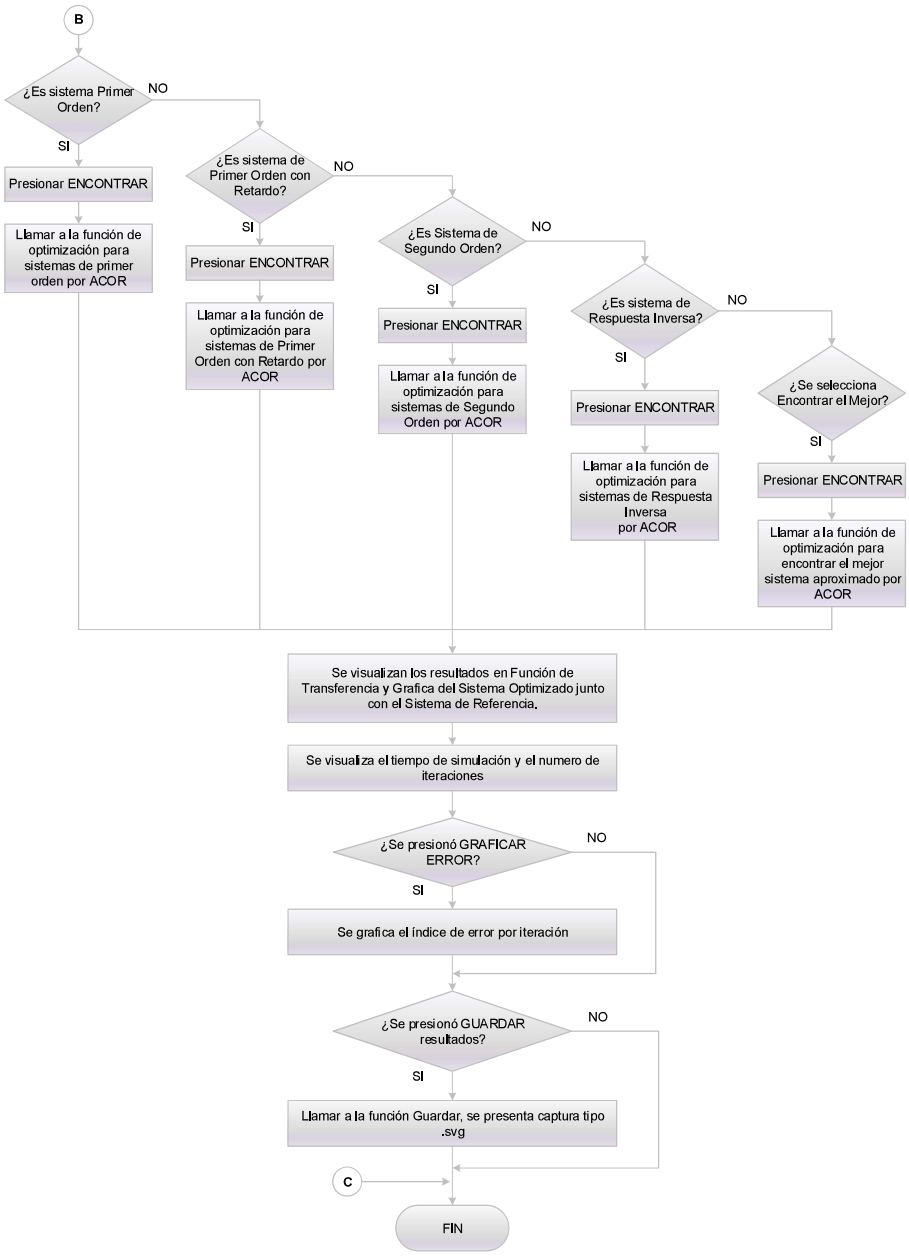


Figura 2.24. Diagrama de flujo de la selección del tipo de sistema identificado y visualización de resultados.

La Figura 2.24 presenta el diagrama de flujo para la selección de la función de identificación dependiendo del tipo de sistema que se quiera encontrar la estimación de parámetros o como en el último caso se escoge la opción encontrar el mejor para que identifique por sí mismo el mejor sistema y estime los valores cercanos a los óptimos de los parámetros.

Para finalizar, se muestra el diagrama de flujo global de la implementación del algoritmo PSO en la herramienta computacional diseñada en App Designer de Matlab®, el mismo que cuenta con las mide funcionalidades descritas previamente.

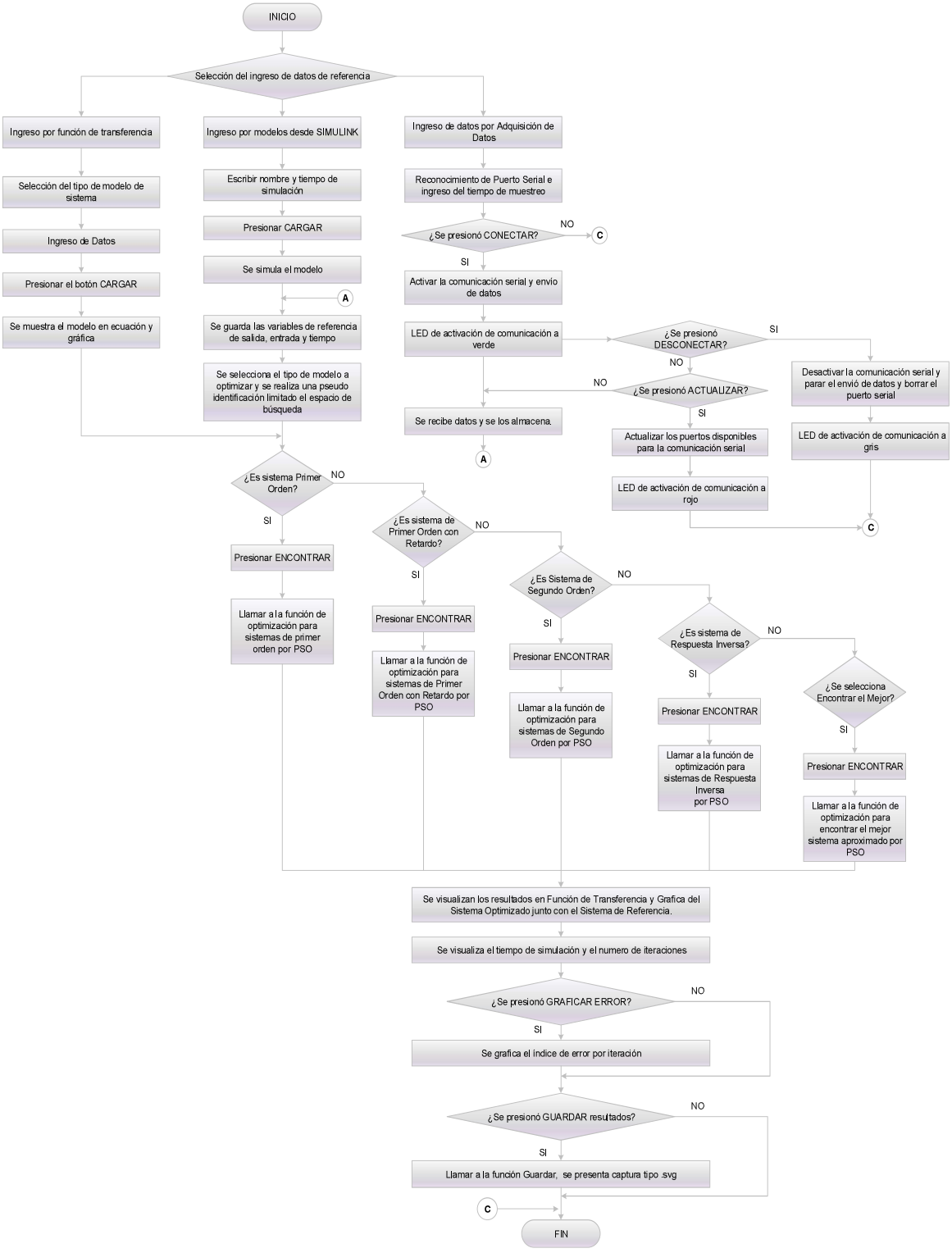


Figura 2.25. Diagrama de flujo completo de la implementación del algoritmo PSO.

Una vez culminada la fase de desarrollo y diseño de la interfaz gráfica de la herramienta computacional se inician las pruebas tanto simuladas como físicas, para lo cual se necesita considerar algunos aspectos de importancia.

2.7 ADQUISICIÓN DE DATOS

En la adquisición de datos se utiliza un ARDUINO® MEGA 2560, para determinar el tiempo de muestreo de esta tarjeta se ejecuta un código sencillo donde se utiliza una función que cuenta los microsegundos que transcurren desde que se obtiene una muestra hasta la siguiente muestra, de ese experimento se obtiene un tiempo de muestreo es de 112 µs, eso equivale a una frecuencia 8.920 KHz como lo muestra la Ecuación 2.8.

$$f_{arduino} = \frac{1}{112\mu s} = 8.920\ KHz \tag{2.8}$$

Se demuestra que la frecuencia que pose esta tarjeta embebida es suficiente para realizar una buena discretización del sistema motor DC. Ahora, se necesita un programa que permita tomar datos en un tiempo de muestreo (Tm) especificado por el usuario pero que se encuentre dentro de un rango determinado que garantice una adecuada discretización de las señales. También, se necesita una conversión análoga- digital del voltaje de entrada y un control del paso de alimentación al motor, con el fin de muestrear su dinámica, la misma que se visualiza en el cambio de la señal de entrada; como se emplea un motor con encoder se necesita contar los pulsos recibidos para poder transformarlos en velocidad y el tiempo de muestreo, que es el número de intervalos de tiempo tomados por el usuario.

Figura 2.26. Sistema de adquisición de datos SysID.

2.7.1 PROGRAMACIÓN DEL TIEMPO DE MUESTREO

Para la adquisición de datos se utiliza el timer 3 de 16 bits y su configuración es en modo comparación o CTC (Clear Time on Compare). El timer cuenta el intervalo del tiempo de muestreo. Cada vez que este y el registro de comparación sean iguales se lanza una interrupción al microcontrolador para que tome los datos de las señales de entrada, salida y tiempo. Seguidamente el contador se reinicia en cero para volver a contar un tiempo más

de muestreo, así sucesivamente. Para este propósito se debe configurar algunos registros de acuerdo con las siguientes tablas.

Tabla 2.7 Descripción de los modos de generación de forma de onda

Mode	WGMn3	WGMn2 (CTCn)	WGMn1 (PWMn1)	WGMn0 (PWMn0)	Timer/Counter Mode of Operation	TOP	Update of OCRnx at	TOVn Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCRnA	Immediate	MAX

La Tabla 2.7. muestra la configuración del timer en modo comparación o CTC como está en el modo 4. El timer 3 posee dos registros que deben ser configurados el TCCR3A, TCCR3B y TCCR3C, los cuales se han configurado de acuerdo con las tablas en [40].

TCCR3A es el primer registro del timer 3 en modo CTC sus bits 7:2 son para obtener de los pines señales PWM, lo cual no se necesita y por tanto se carga el valor a cero lógico. Los bits 1:0 son configuraciones para usar en el modo 4 de la Tabla 2.7, así tenemos que estos pines van a cero lógico (0L).

TCCR3B es el segundo registro del timer 3 para configurarlo en modo comparación. Sus bits 7:6 son para disminuir el ruido y especificar el flanco de captura cuando se usa el pin de captura de entrada, como no los utilizamos estos van a 0L; el bit 5 está reservado por cuanto ira a 0L; los bits 4:3 son la continuación de la configuración del modo 4, que es el modo CTC; por tanto, el bit WGM33 va a 0L y el bit WGM32 va a uno lógico (1L); en los bits 2:0 se configura el pre escalador mediante la siguiente tabla y bajo la Ecuación 2.9.

Tabla 2.8 Descripción de los bits de selección de reloj.

CSn2	CSn1	CSn0	Description
0	0	0	No clock source. (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/1 (No prescaling
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on Tn pin. Clock on falling edge
1	1	1	External clock source on Tn pin. Clock on rising edge

$$f_{OC1A} = \frac{f_{clk\_I/O}}{2N(1 + OCR1A)} \tag{2.9}$$

La frecuencia se calcula a partir de la velocidad nominal del motor, en este caso el motor que se utiliza en la prueba práctica es de 4400 rpm, y por datos de placa se conoce que el encoder tiene 100 pulsos por revolución. Por tanto, mediante la conversión de la Ecuación 2.10 se obtiene los pulsos por segundo, para después convertirlos a frecuencia.

$$\frac{4400\,rev}{min} \times \frac{100\,pulsos}{1\,rev} \times \frac{1min}{60\,s} = 7333.33\frac{pulsos}{s} \tag{2.10}$$

Esa sería la resolución de pulsos que se toman si planta responde en segundos, pero el tiempo de establecimiento de la planta está en 60 milisegundos, por consiguiente, es necesario que se haga el equivalente en milisegundos, así se tiene que:

$$\frac{7333.33\,pulsos}{s} \times \frac{1\,s}{1000\,ms} = \frac{7.33\,pulsos}{ms} \tag{2.11}$$

Este último valor refleja la cantidad pulsos que se adquiriría en cada milisegundo, pero es un tiempo de muestreo demasiado pequeño lo cual no es conveniente debido a una pérdida de datos que puede existir por un tiempo tan pequeño. Por tanto, se considera un tiempo de muestreo a cada 5 milisegundos en un margen de hasta los 10 milisegundos.

Considerando el tiempo más pequeño de muestreo de 5 milisegundos se tiene una resolución de 36.65 pulsos y en el de 10 milisegundos es de 73,3 pulsos; ninguno de los dos valores supera la resolución de 16 bits que es hasta 65536 por tanto no se debe tomar consideraciones extra.

Una vez establecidos estos dos márgenes de tiempo de muestreo se procede a calcular el preescalador, así para 5 milisegundos se tiene una frecuencia de 200 Hz y para 10 milisegundos son 100 Hz. El cálculo del preescalador se realiza de acuerdo con la Ecuación 2.12, Ecuación 2.13 y Ecuación 2.14.

$$OCR3A = \frac{f_{clk\_I/O}}{2f_{OC1A}N} - 1 \tag{2.12}$$

$$OCR3A = \frac{16\,MHz}{2(100\,Hz)8} - 1 = 9999 \tag{2.13}$$

$$OCR3A = \frac{16\,MHz}{2(200\,Hz)8} - 1 = 4999 \tag{2.14}$$

Del resultado de las ecuaciones anteriores se posee en bits el valor que el registro OCR3A, estos valores no superan la resolución de 16 bits con la que se trabaja, por tanto, se

considera un pre escalador de $N = 8$. De esta manera los últimos bits del registro TCCR3B se escriben en el bit 2: un 0L, en el bit 1: un 1L y en el bit 0: un 0L.

TCCR3C es el tercer registro de este registro no necesitamos ningún tiempo de comparación forzada por tanto todo el registro va a 0L.

OCR3A es el registro que guarda el valor máximo o TOP hasta el cual va a contar el timer 3, como este depende del tiempo de muestreo que ingrese el usuario se necesita que se defina en variable para ser calculada en el microprocesador. Al trabajar en un tiempo de muestreo (Tm) variable se debe encontrar la ecuación que rige el valor de OCR3A en función de este. A continuación, se presenta el cálculo correspondiente.

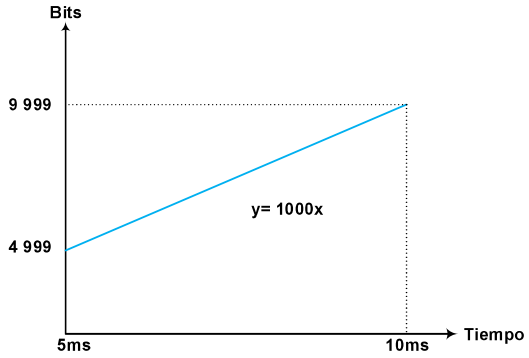


Figura 2.27. Gráfica que relaciona al Tm con los bits del registro OCR3A.

Ecuación de una recta:

$$y = mx \tag{2.15}$$

$$m = \frac{y_2 - y_1}{x_2 - y_1} = \frac{9\,999\ bits - 4\,999\ bits}{10\ ms - 5\ ms} = \frac{1\,000\ bits}{ms} \tag{2.16}$$

Por tanto, la ecuación final sería

$$y = 1\,000x \tag{2.17}$$

Donde:

y: es el valor de OCR3A

x: es el tiempo en milisegundos en un rango de 5 ms a 10 ms.

Todos los cálculos anteriores son para asegurar que la ecuación que rige el valor del OCR3A en tiempo de muestreo variable no desbordará los 16 bits de resolución, si se trabaja con un pre escalador de 8.

TIMSK3 es el registro de interrupciones del timer 3, esta lanza una interrupción cada vez que se alcanza el tiempo de muestreo, del cual solo nos interesa que el bit 1: que es la habilitación de interrupción en modo captura este activa, es el único bit que ira a 1L.

SEI es el comando que habilita las interrupciones de forma serial, el cual también debe estar activado.

2.7.2 PROGRAMACIÓN DEL TIMER 5 EN MODO CONTADOR

En timer 5 de 16 bits que tiene al pin 47 como pin externo, se lo configura en modo normal como contador, para que cuente los pulsos cada que exista un flanco de bajada y subida, hasta el máximo de 0xFF. Para lograr este objetivo se deben configurar tres registros:

TCCR5A es el registro que se configura según el modo que se utilizará al timer, en este caso se considera el modo 0 mostrado en la Tabla 2.7. Así los bits 7:2 se escriben a 0L porque no necesitamos señales PWM, y los bits 1:0 van a 0L como lo especifica en la tabla.

TCCR5B es el segundo registro para configurarse en el timer 5, como este se lo utiliza en modo contador no se necesita configurar los bits de 7:3 por tanto se escriben a 0L, de los bits 2:0 se configura de acuerdo con la Tabla 2.8, como se necesita contar los flancos de bajada de la señal del encoder, se utiliza en modo fuente de reloj externo en el pin 47 así en el bit 2:1 se lo configura a 1L y el bit 0: a 0L.

TCCR5C se configura todo a 0L, porque los bits 7:5 se los utiliza cuando se quiere forzar la comparación en la unidad generación de formas de onda, y los bits 4:0 están reservados.

TCNT5 es la localidad de 16 bits donde se guarda los valores que el timer 5 cuenta a lo largo del tiempo, esta es la localidad a la cual se ingresa para saber el número de pulsos contados en cada tiempo de muestreo, cada vez que se extrae el número de pulsos se reinicia la variable para que no exista una sobre escritura de valores a causa de un desbordamiento de esta localidad.

2.7.3 PROGRAMACIÓN DE LA CONVERSIÓN ANÁLOGA DIGITAL

El converso ADC del Arduino® Mega 2560 transforma una señal análoga de voltaje de entrada a valores digitales de 10 bits de resolución. Este microcontrolador posee quince puertos (pines) de conversión análoga a digital, de los cuales se ha utilizado el pin A0 para la conversión. En los siguientes párrafos se describe el proceso de configuración.

ADMUX es uno de los registros a configurarse para activar la conversión ADC, el bit 7:6 se configura para que la referencia de voltaje sea interna de 1.1V con un capacitor externo al pin AREF, esto es con los valores 1L y 0L como corresponde; se escoge la alineación

de los datos de conversión a la derecha esto es que el bit 5 este en 0L; el bit 4:0 representan cuál de los puertos análogos se establece como puerto de conversión análoga digital y si se necesita una ganancia de selección como el puerto que utilizamos es el A0 por tanto estos bits van a 0L.

Tabla 2.9 Selección de la fuente de disparo automático para ADC

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free Running mode
0	0	1	Analog Comparator
0	1	0	External Interrupt Request 0
0	1	1	Timer/Counter0 Compare Match A
1	0	0	Timer/Counter0 Overflow
1	0	1	Timer/Counter1 Compare Match B
1	1	0	Timer/Counter1 Overflow
1	1	1	Timer/Counter1 Capture Event

ADCSRB este registro se configura para el control y estatus de la conversión ADC, así para e bit 7:3 están reservados por tanto su valor es a 0L. Los últimos bits 2:0 se configuran para seleccionar la fuente que da paso a la conversión análoga, de la Tabla 2.9 se escoge la conversión en free running porque el Timer 3 se configuro para tomar valores en cada tiempo de muestreo desde en la conversión ADC; por tanto, estos últimos bits van a 0L.

DIDR0 es el registro que deshabilita la entrada digital de los pines de conversión utilizados, para que solo se cumpla la función ADC; como se usa únicamente el pin A0, el bit 0 es el único a 1L los demás estarán en 0L.

2.8 CONFIGURACIÓN DE LA COMUNICACIÓN SERIAL

Para la configuración de la comunicación serial entre App Designer® y Arduino® se ha utilizado la herramienta Control Instrument en Simulink de MatLab®, así la configuración del puerto es de forma asincrónica, permitiendo la escritura y lectura de datos.

Para la adquisición de datos primero se envía el tiempo de muestreo por el puerto serial del computador al Arduino, se inicia la lectura de datos por el puerto serial; pero no se inicializa la adquisición de datos si no hasta esperar un lapso de tres iteraciones del tiempo de muestreo. Esto con el propósito de dar paso a la alimentación del motor al mismo tiempo que se adquieren las señales de entrada de voltaje, salida de pulsos y el contador de los tiempos de muestreo realizados.

Una vez que se ha adquirido un tamaño de muestras considerables se cierra la comunicación serial. Los datos adquiridos por la tarjeta son enviados de forma serial a la

interfaz gráfica, dentro de esta se los transforma y se los almacena en forma de vectores. Se procede a realizar la identificación del modelo de sistema y la estimación de valores cercanos a los óptimos para los parámetros característicos del sistema.

La comunicación serial se la ha realizado por medio de Simulink, la configuración de los bloques se ha realizado según indica [41]. En la Figura 2.28 y en la Figura 2.29 se representa gráficamente la escritura (envío) y la lectura (recepción) de los datos entre el hardware de adquisición y el software de la herramienta computacional SysID.

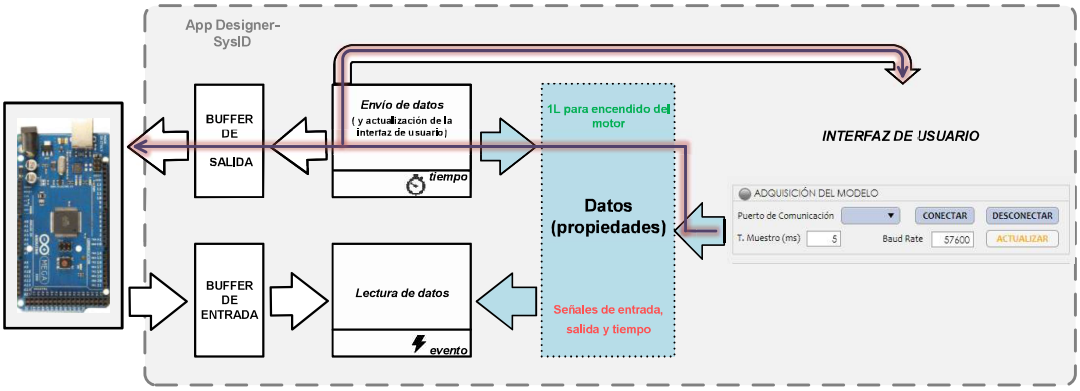


Figura 2.28 Envío del tiempo de muestreo desde SysID.

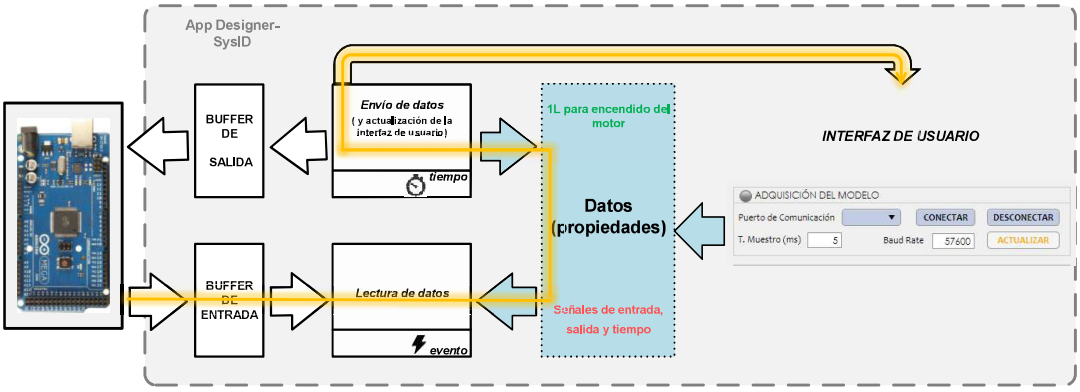


Figura 2.29 Recepción de datos por SysID.

2.9 CARACTERÍSTICAS DEL EQUIPO HARDWARE

Para un buen desempeño de la aplicación SysID se recomienda como mínimo utilizar un computador de las siguientes características.

- Procesador: Intel ® Core (TM) i7-5500U CPU @ 2.40GHz.
- RAM: 8.00 GB

Para la tarjeta de adquisición se puede utilizar cualquiera que tenga disponibilidad de:

- Mínimo tres timers (uno con pin externo).

- Un puerto de conversión análoga digital.
- Un reloj interno mínimo de 16MHz
- Tres puertos de entrada- salida digitales.

2.10 CARACTERÍSTICAS DEL SOFTWARE

El desarrollo de esta herramienta computacional se la ha realizado en bajo las siguientes condiciones de software por tanto se recomienda utilizar las mismas de ser posible.

- Tipo de sistema: Sistema operativo de 64 bits, procesador x64
- Edición de Windows: Windows 10 Education
- Edición de MatLab: MatLab® 2019b

2.11 CASO PRÁCTICO: IDENTIFICACIÓN DEL MODELO DE SISTEMA A UN MOTOR DC

La Figura 2.30 muestra el procedimiento que se realizó en la ejecución de la prueba práctica para identificar el modelo motor DC con encoder acoplado, del cual se considera la lectura de pulsos del encoder transformados a rpm como la señal de salida de la planta, y la conversión análoga digital de la entrada del voltaje como señal de entrada de la planta y un contador de tiempo de muestreo como el vector tiempo del sistema en milisegundos.

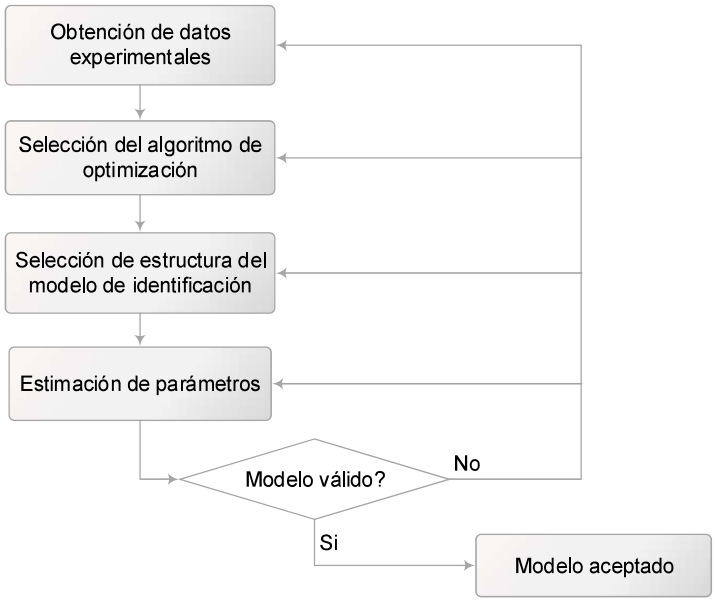


Figura 2.30. Proceso de identificación de un modelo de sistema a una planta real.

La estructura del modelo se selecciona de acuerdo con las gráficas obtenidas de los datos adquiridos, en este caso se probó para las estructuras de primer orden y de segundo orden.

La estimación de parámetros se hace por la minimización de la función de costo que es el índice medio cuadrático de error (MSE). Una vez encontrados los valores más cercanos a los óptimos se válida la respuesta del modelo mediante la superposición de gráficas entre el sistema identificado y el sistema real.

2.11.1 CARACTERÍSTICAS DEL MOTOR DC Y ENCODER

En el caso práctico se ha considerado un servo motor DC con encoder incorporado, de la marca Japón Motors, el mismo que tiene aplicaciones en la alimentación de papel para las copiadoras de la marca EPSON. A continuación, se presenta sus datos de placa.

- Tipo: DSE38BE27-001
- P/N: AX060121
- Tensión nominal 24 V dc
- Velocidad nominal: 4400 rpm
- Diámetro del motor: 33 mm
- Longitud del motor: 101.8 mm (incluido el encoder)
- Diámetro del eje de salida: 5mm
- Longitud del eje de salida: 25mm
- Potencia de salida: 19 W
- Cables de alimentación: Rojo (+) y Negro (-)

El enconder utilizado en el caso práctico está acoplado al motor dc y sus datos característicos del encoder son los siguientes:

- Modelo: GP1A30R
- Pulsos: 100 pulsos por revolución
- Voltaje de alimentación: 5 V dc
- Cables de alimentación: Rojo (+5Vdc) y Verde (-Gnd)
- Cable de señal A: Amarillo
- Cable de señal B: Blanco

2.11.2 OBTENCIÓN DE DATOS EXPERIMENTALES

Para la obtención de los datos experimentales además de realizar el programa de adquisición se diseña una pequeña placa PBC para resguardar el voltaje y corriente que ingresan a la tarjeta embebida como la fidelidad de los datos adquiridos.

A continuación, se describe cada uno de los circuitos utilizados para este propósito y finalmente se presenta el diseño total de la placa en la Figura 2.31.

- Circuito de accionamiento de la entrada de voltaje

El circuito utilizado es un accionamiento por relé con activación de bobina a 5 Vdc, esto con el fin de iniciar la adquisición de datos justo con la alimentación del motor. Se coloca un diodo 1N4004 entre la bobina del relé para evitar las posibles corrientes inversas, y un transistor NPN 2N3904 al que se conecta su base al pin de activación del sistema de alimentación desde el programa.

- Circuito de acondicionamiento para la señal del encoder

Para evitar posibles entradas de voltaje mayor a 5 Vdc al pin A0 de la conversión ADC, se coloca un dio Zener entre la señal del encoder y el GND del encoder con Arduino®.

- Circuito de acondicionamiento para la señal de voltaje de 5 Vdc

El circuito de acondicionamiento es un divisor de voltaje que se ha diseñado en base a la alimentación del motor y considerando una resistencia de 1KΩ como condición inicial.

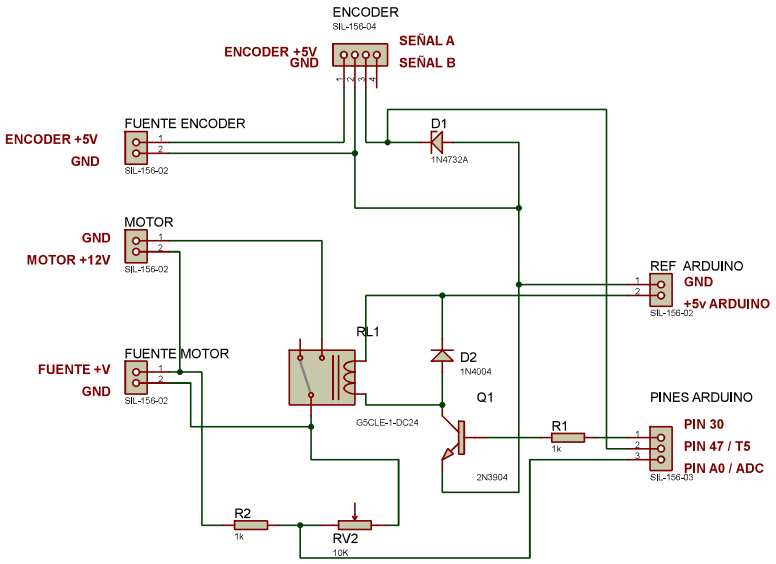


Figura 2.31. Circuito de acondicionamiento de las señales del Motor DC a Arduino®.

Con el software y hardware de adquisición listos se procede a muestrear las señales de entrada, salida y tiempo, que se las utiliza con las dos herramientas computacionales ya antes mencionadas para la identificación del modelo de sistema y para la estimación de parámetros característicos. Se presenta en la Figura 2.32 señal del voltaje de ingreso al motor. La prueba se realizó a 12 VDC para un tiempo de muestreo de 5ms.

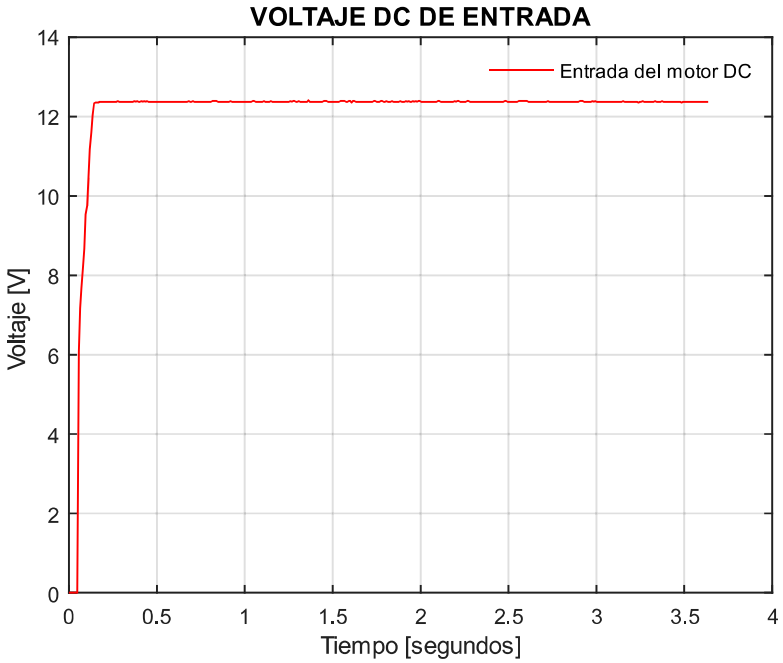


Figura 2.32. Datos adquiridos de voltaje.

La Figura 2.33 muestra la velocidad de salida del motor en rpm obtenida en la adquisición de datos.

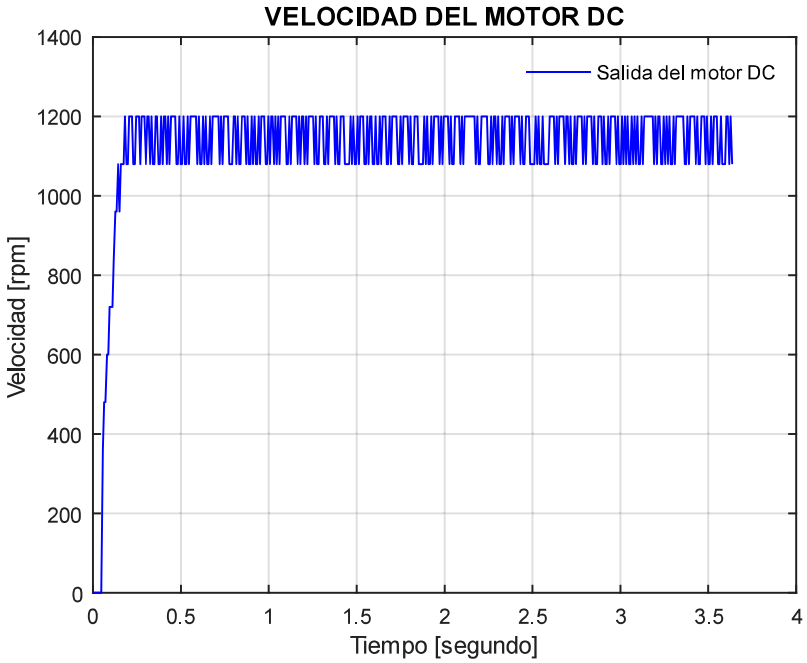


Figura 2.33. Datos adquiridos de velocidad.

El tiempo en la adquisición de datos es en milisegundos, pero en las gráficas para una mejor visualización se ha trasformado en segundos, y en ese mismo orden se hará la identificación del modelo de sistema.

3. RESULTADOS Y DISCUSIÓN

En un inicio, se presenta el análisis detallado de los resultados obtenidos mediante la herramienta computacional a la identificación y estimación de parámetros de los sistemas tipo mencionados anteriormente; para lo cual se hace una comparación de los algoritmos de optimización cuando se ha cumplido con dos diferentes condiciones de parada; y para el caso práctico se realiza una comparación de SysID con la función IDENT propia de MatLab® con lo cual se pretende establecer comparaciones entre las ventajas y diferencias que presentan estos dos softwares de identificación; para concluir se presenta la eficiencia de la herramienta y de cada uno de los algoritmos mediante un gráfico de barras.

Para el análisis de resultados de los tres algoritmos de optimización implementados, y con el propósito de realizar una comparación justa se consideró elaborarlos en iguales condiciones es decir en un mismo lenguaje de programación, bajo iguales intervalos de inicialización de búsqueda, con igual metodología para el ajuste de parámetros y con las mismas condiciones de parada. Así para obtener un grupo de datos significativos para su comparación se realiza 20 pruebas independientes para cada modelo tipo de sistema en los tres algoritmos implementados, con el propósito de presentar el desempeño medio y la taza de éxito al encontrar un valor muy cercano al óptimo global, como la cuantificación de su precisión y exactitud de respuesta.

En las 10 primeras ejecuciones se considera como condición de parada llegar a un valor menor al 0.01 de la función de costo, mediante esta prueba se pretende evaluar el porcentaje de exactitud y precisión en encontrar el valor más cercano al óptimo global; con las 10 ejecuciones restantes la condición de parada es llegar al número máximo de 200 iteraciones con esto se hace un análisis del número de iteraciones necesarias y del costo computacional invertido.

Cada uno de los parámetros de los algoritmos de optimización de la herramienta SysID se los seleccionó mediante el proceso de prueba y error; con fundamento en la teoría como en la experiencia adquirida durante el proceso de prueba. Además, se enfatiza que en un inicio se trabajó con el índice ISE como función de optimización de los algoritmos, pero no se obtuvo el resultado deseado, debido a que este índice al llegar al error muy pequeño permanece constante. En consecuencia, se consideró como función de minimización o función de costo al índice MSE, que indica el sesgo y la varianza del valor estimado y el real. Estos valores son el reflejo de cuan alejado está el promedio del valor estimado y el real, es decir mientras menor sea su valor más eficiente es la estimación y más consistente el modelo referencia con el modelo resultado de la identificación.

3.1 RESULTADOS DE LA IDENTIFICACIÓN DE SISTEMAS DE PRIMER ORDEN

Los resultados de la identificación del sistema tipo de primer orden mediante la herramienta computacional SysID alcanzaron, en promedio, la exactitud del 99.7% entre las variables de referencia y los valores optimizados como lo muestra la Figura 3.1.

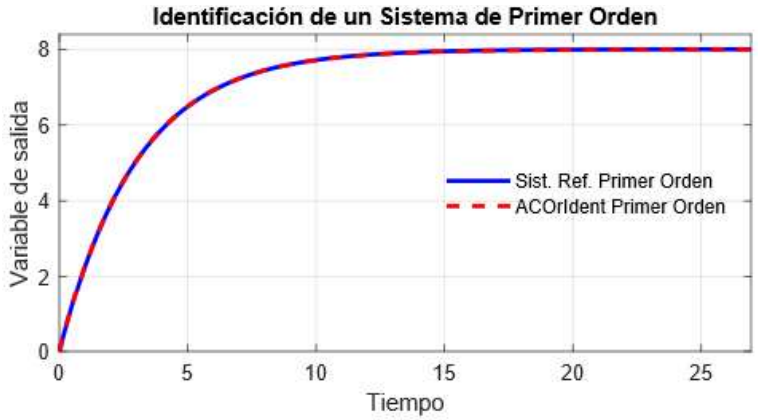


Figura 3.1 Identificación del sistema de primer orden por algoritmo ACO.

La Tabla 3.1 muestra los valores promedio de los resultados de la optimización y los valores referencia del sistema tipo de primer orden, se aprecia que cada algoritmo optimiza a valores muy próximos entre sí. Estos valores de optimización promedio son de la Tabla C.1, Tabla C.2, Tabla C.3 de la parte de Anexos. También se ha calculado el error en porcentaje para cada parámetro, resultando en errores menores al 0.5%. Además, se aprecia que el algoritmo PSO tiene el menor porcentaje de error.

Tabla 3.1 Parámetros estimados por SysID

	PARÁMETROS ESTIMADOS PARA SISTEMAS DE PRIMER ORDEN						
	Referencia	Valores Promedio					
		ACO	Error (%)	ACO-R	Error (%)	PSO	Error (%)
K	8	7.9776	0.28	7.9926	0.09	7.9842	0.19
τ	3	3.0139	0.46	3.0149	0.49	2.9991	0.03

Como se había mencionado con anterioridad, para interpretar el alcance de la aplicación de algoritmos de optimización bioinspirados en la identificación de sistemas y estimación de parámetros, se guardó los resultados de 20 pruebas de identificación realizadas al modelo tipo de primer orden, estas pruebas están en el Anexo C. A continuación, se presentan los resultados promedios obtenidos para dos de las condiciones de parada implementadas en la herramienta computacional.

Tabla 3.2 Valores promedio de la identificación del sistema tipo de primer orden.

	CONDICIONES DE PARADA				
	Índice de error < 0.01			Máximo de 200 iteraciones	
	Valores Promedio			Valores Promedio	
	MSE	Tiempo [min]	Número iteraciones	MSE	Tiempo [min]
ACO	0.00413	0.5336	2.7	0.000638	5.3403
ACO-R	0.00375	0.1294	3.4	1.11e-30	1.2526
PSO	0.00553	0.1135	3.3	1.11e-30	0.8757

La Tabla 3.2 muestra los resultados obtenidos bajo dos tipos de condiciones de parada. Los resultados para la primera condición son similares entre sí, pero el de menor índice de error, tiempo de ejecución aceptable y un número de iteraciones promedio es el algoritmo ACO-R. Sin embargo, cabe destacar que cada uno de los algoritmos se lidera en al menos una categoría, así es que el algoritmo ACO posee el menor número de iteraciones, el algoritmo PSO posee el menor tiempo de ejecución del algoritmo, mientras que el algoritmo ACO-R posee el menor índice de error.

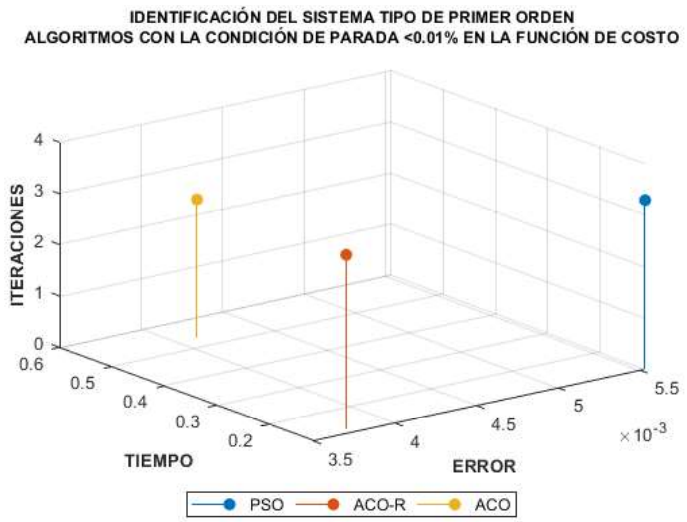


Figura 3.2. Comparación de los algoritmos a una condición de parada.

En la Figura 3.2. se ilustran las características promedio obtenidas de la Tabla 3.2 cuando la condición de parada es que la función de costo es menor a 0.01; de la cual es más fácil interpretar que el algoritmo ACO-R posee las mejores respuestas para la identificación de sistemas de primer orden.

La Figura 3.3 muestra la gráfica de barras de la identificación de sistemas tipo de primer orden bajo la condición de parada de un máximo de iteraciones de 200; así se evidencia que el mejor algoritmo de identificación en base al tiempo y al mínimo índice de error es el algoritmo PSO con una notable superioridad en comparación con los demás.

Del gráfico de barras los tres algoritmos poseen valores de su función de costo muy cercanos a cero, pero a un costo computacional elevado; lo cual refleja que es necesario dos condiciones de parada. Por tanto, para la herramienta computacional se implementa condiciones por el número de máximo de iteraciones y por el porcentaje mínimo de error.

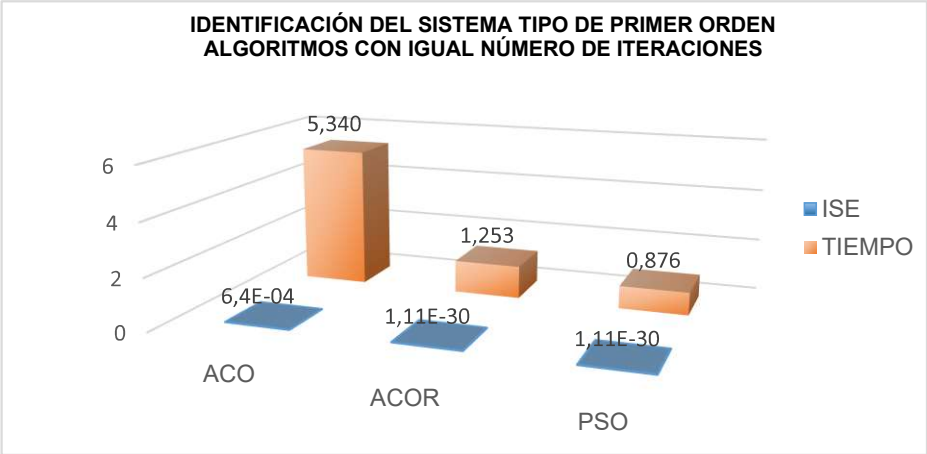


Figura 3.3. Gráfico de barras comparativo de las pruebas de identificación.

Ahora bien, se tiene un escenario adicional cuando la estimación de parámetros no llega a la condición mínima de error y tampoco cambia su costo de optimización dentro de un número considerable de iteraciones, lo cual sugiere que ha llegado a su solución óptima, y por lo cual no es necesario esperar que el algoritmo cumpla con el número máximo de iteraciones. Para este escenario se ha implementado una tercera condición, que dado un número máximo de repeticiones del mismo índice de error se detenga la búsqueda y se presente el resultado, para esta condición se ha considerado un máximo de 40 iteraciones.

Mediante estas condiciones de parada se mejora el tiempo de ejecución del algoritmo y por tanto el costo computacional, a la vez que no se interfiere en la búsqueda de los valores cercanos a los óptimos para los parámetros característicos del sistema identificado.

3.2 RESULTADOS DE LA IDENTIFICACIÓN DE SISTEMAS DE PRIMER ORDEN CON RETARDO

En los párrafos siguientes se discute los resultados obtenidos de la identificación y estimación de parámetros del sistema tipo de primer orden con retardo como se muestra en la Figura 3.4.

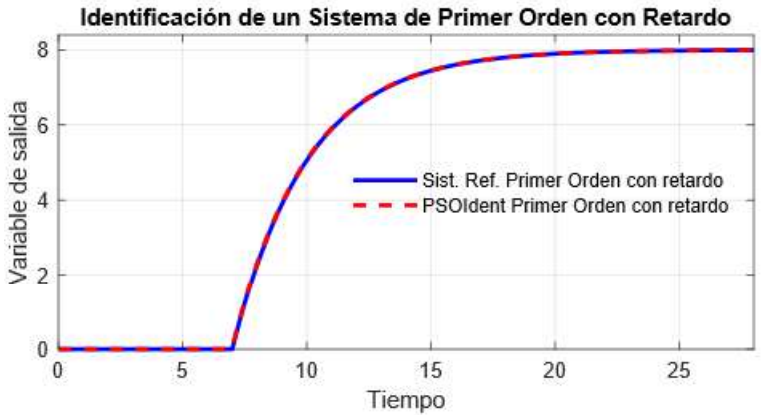


Figura 3.4 Identificación del sistema por el algoritmo PSO.

La Tabla 3.3 muestra los parámetros referencia y los parámetros promedio obtenidos de las pruebas realizadas a la identificación de sistemas por SysID para el sistema tipo de primer orden con retardo. Estas pruebas se encuentran en la Tabla C.4, Tabla C.5 y Tabla C.6 del Anexo C. También, se ha calculado el error de cada parámetro con respecto a la referencia, utilizando la Ecuación 1.34.

El algoritmo de optimización PSO posee el menor error de medida alrededor del 0.14%; por tanto, se lo considera como el mejor algoritmo de identificación para este tipo de modelo de sistema. Se evidencia que el parámetro con mayor dificultad para su estimación es la constante de tiempo con un 5% de error, esto no favorece a los algoritmos ACO y ACO-R.

Tabla 3.3 Parámetros estimados por SysID

	PARÁMETROS ESTIMADOS PARA SISTEMAS DE PRIMER ORDEN CON RETARDO						
	Referencia	**Valores Promedio**					
		ACO	**Error (%)**	**ACO-R**	**Error (%)**	**PSO**	**Error (%)**
K	8	8.0186	0.23	8.0567	0.70	7.9896	0.13
τ	3	3.1386	4.62	3.1721	5.43	3.0042	0.14
t_0	7	6.8806	1.74	6.9373	0.89	6.9901	0.14

En la Tabla 3.4 se muestra el promedio de las mediciones realizadas a la estimación de parámetros e identificación de modelo ante las dos condiciones de parada mencionadas.

Se observa que el algoritmo PSO posee una ventaja de 41 veces menor en el tiempo de ejecución comparado con el algoritmo ACO; y un 9% en el tiempo de ejecución comparado con el algoritmo ACO-R. Se destaca que ACO-R también refleja mejores resultados que la forma tradicional del algoritmo ACO, esto debido a su heurística de búsqueda que se basa en funciones de densidad de probabilidad normalmente distribuidas provenientes de las antiguas mejores soluciones, son utilizadas en las futuras iteraciones.

Tabla 3.4 Valores promedio de a dos de las condiciones de parada.

	CONDICIONES DE PARADA				
	Índice de error < 0.01			Máximo de 200 iteraciones	
	Valores Promedio			Valores Promedio	
	MSE	Tiempo [min]	Número iteraciones	MSE	Tiempo [min]
ACO	0.01244	7.6148	230	0.00921	6.681
ACO-R	0.00553	0.1989	9.3	1.34E-30	2.040
PSO	0.00422	0.1822	7.9	2.25E-30	1.461

De la tabla anterior, el algoritmo ACO no cumple con la condición de parada establecida en un valor menor al 0,01 de la función de costo. De los 10 casos de prueba realizados para este algoritmo, se tiene un acierto del 70% con la condición de parada antes mencionada, a pesar de que se le ha proporcionado un número mayor de iteraciones.

Los resultados de esta prueba siguieren dos acciones: la primera, una identificación previa para encontrar un espacio de búsqueda prometedor; y la segunda, implementar una tercera condición de parada. Como se expuso anteriormente, si al llegar a un valor cercano al óptimo y no registrar ningún cambio en la función de minimización luego de 40 iteraciones, se finaliza el proceso de optimización. Estos dos criterios se implementaron en SysID.

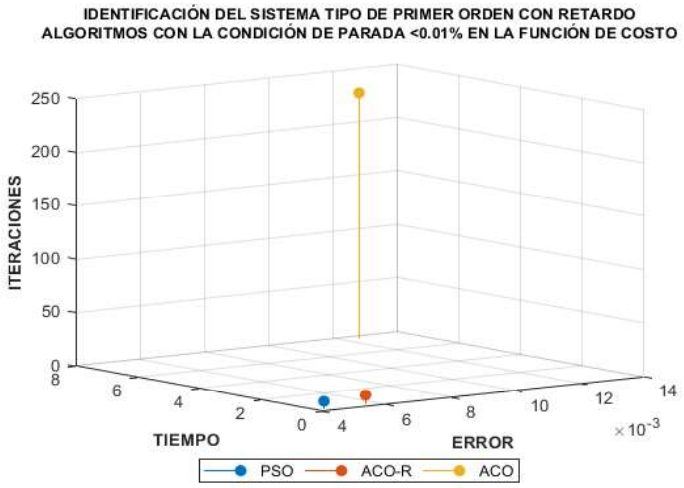


Figura 3.5. Comparación de resultados promedio de la identificación del sistema.

La Figura 3.5 muestra la comparación de los tres algoritmos de optimización implementados, de acuerdo con el número de iteraciones, tiempo de ejecución e índice MSE. Se aprecia que tanto el algoritmo de optimización por enjambre de partículas y el de optimización por colonia de hormigas para el dominio continuo tienen valores semejantes; esto es coherente con la teoría porque los dos algoritmos trabajan de forma evolutiva.

Para la segunda prueba de optimización con condición de parada del número máximo de iteraciones se tiene el gráfico de barras de la Figura 3.6, en la cual se aprecia que el algoritmo PSO posee el mejor tiempo como el mejor índice de error alcanzado, seguido del algoritmo ACO-R y finalmente el algoritmo ACO; demostrando una superioridad del 4,57 veces en el tiempo de ejecución en comparación con el último algoritmo mencionado y de 39,63% con el algoritmo ACO-R.

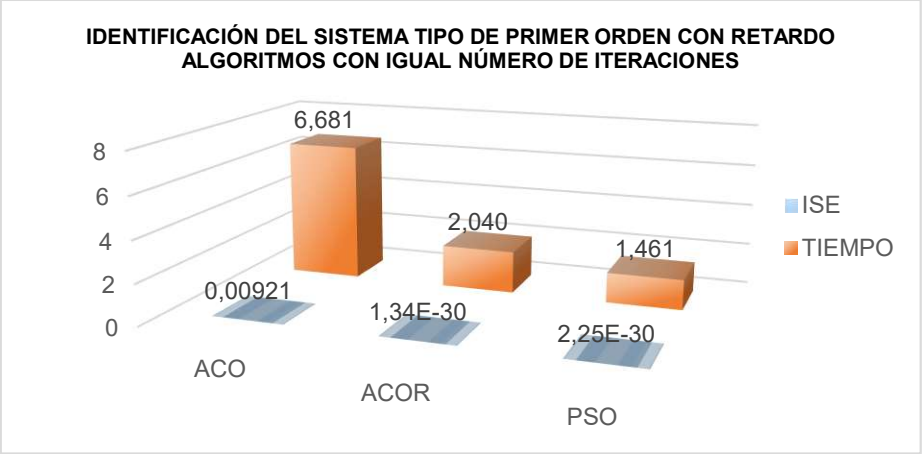


Figura 3.6. Gráfico de barras para la comparación de error y tiempo de ejecución.

En contraste con los resultados obtenidos, se aprecia que bajo un mayor número de iteraciones el algoritmo ACO no manifiesta un mejor resultado; pero si demuestra que utiliza un mayor tiempo de ejecución y un mayor costo computacional.

3.3 RESULTADOS DE LA IDENTIFICACIÓN DEL SISTEMA TIPO DE SEGUNDO ORDEN

La Figura 3.7 muestra la identificación del sistema tipo de segundo orden, descrito en el segundo capítulo, bajo la optimización mediante el algoritmo PSO; la exactitud del modelo y estimación de los parámetros es del 97.82%.

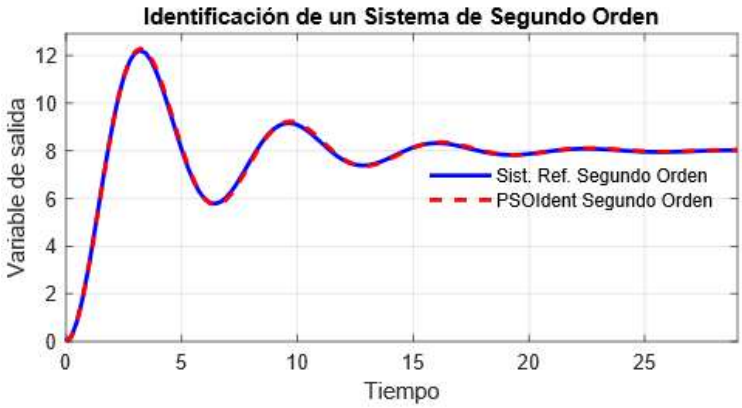


Figura 3.7. identificación del sistema tipo de segundo orden por PSO.

La Tabla 3.5 presenta un resumen de la estimación de parámetros realizada por la herramienta computacional propuesta y la referencia del modelo tipo de segundo orden. Los valores promedio expuestos están en base a la Tabla C.7, Tabla C.8 y Tabla C.9 del Anexo C, al igual que en los algoritmos anteriores el PSO posee los menores porcentajes de error; mientras que el algoritmo ACO presenta el mayor porcentaje de error en la estimación del valor cercano al óptimo para el coeficiente de amortiguamiento, ε.

Tabla 3.5 Parámetros estimados por SysID

	PARÁMETROS ESTIMADOS PARA SISTEMAS DE SEGUNDO ORDEN						
	Referencia	**Valores Promedio**					
		ACO	**Error (%)**	**ACO-R**	**Error (%)**	**PSO**	**Error (%)**
K	8	7.9889	0.14	8.0680	0.85	7.9898	0.13
ε	0.2	0.2413	20.65	0.2035	1.75	0.2037	1.85
w_n	1	1.0270	2.70	1.0061	0.61	0.9980	0.20

De las pruebas de identificación y estimación de parámetros realizadas se obtuvieron los resultados promedio que se muestran en la Tabla 3.6. Nuevamente, el algoritmo PSO tiene un mayor rendimiento en comparación con los algoritmos restantes. Al igual que en la identificación del modelo de primer orden con retardo, el algoritmo ACO no cumple con la condición de parada de llegar a un valor menor al 0,01 en su función de costo, a pesar de que se le ha proporcionado un mayor número de iteraciones.

Tabla 3.6 Promedio de respuesta en la identificación del sistema tipo de segundo orden.

	CONDICIONES DE PARADA				
	Índice de error < 0.01			**Máximo de 200 iteraciones**	
	Valores Promedio			**Valores Promedio**	
	MSE	**Tiempo [min]**	**Número iteraciones**	**MSE**	**Tiempo [min]**
ACO	0.3457	6.1931	230.2	0.32636	5.946
ACO-R	0.0070	0.5694	62	5.50E-30	1.384
PSO	0.0055	0.2897	42.1	5.50E-30	1.047

También se interpreta de la tabla anterior que el algoritmo ACO que posee índices de error muy similares para las dos condiciones de parada; considerando que en para una condición de parada en las pruebas se ha adicionado 30 iteraciones más.

De la misma manera el algoritmo PSO utiliza alrededor del 32% menos iteraciones que el algoritmo ACO-R y el 81.71% menos que el algoritmo ACO para obtener un resultado menor al 0.01 en la función de costo. De igual manera, se evidencia el algoritmo PSO utiliza el 4,67% y el algoritmo ACO-R el 9,19% del tiempo de ejecución empleado por el algoritmo tradicional ACO.

Para la condición de parada de un máximo de 200 iteraciones los algoritmos PSO y ACO-R poseen muy buenos resultados con un tiempo menor en el 17.65% y 23.27% del tiempo empleado por el algoritmo ACO y con índices de error diferenciablemente menores que el alcanzado por este. Como se precia en la tabla anterior a pesar de que al algoritmo ACO se le ha provisto de mayor número de iteraciones el resultado obtenido no justifica esta acción, porque no mejora significativamente su índice de error.

La Figura 3.8 evidencia con claridad que el mejor algoritmo para la condición de parada de un valor de la función de costo menor a 0.01 es el algoritmo PSO, cabe mencionar que las pruebas realizadas fueron a un sistema tipo de segundo orden subamortiguado.

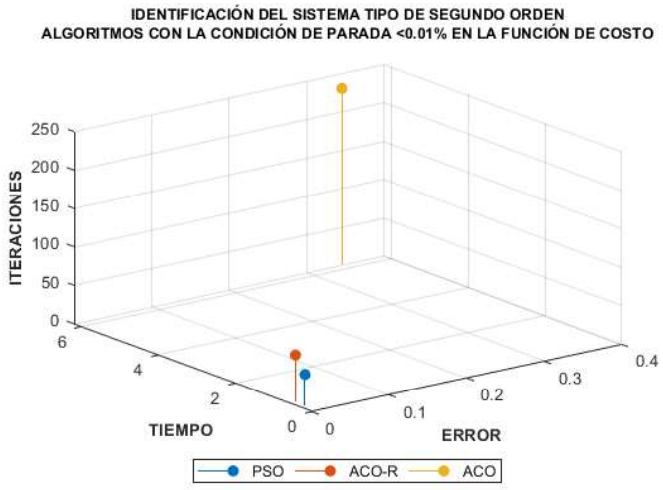


Figura 3.8. Gráfica comparativa del promedio de los algoritmos implementados.

Al comparar estos resultados con los obtenidos anteriormente, se interpreta que conforme la dificultad del modelo a identificar aumenta, sea por el número de parámetros o el rango de valores donde se debe optimizar, incrementa a la vez el número de iteraciones necesarias para alcanzar la condición de parada establecida; s tres algoritmos involucrados en la identificación del sistema.

Por tanto, se considera que para un mejor rendimiento de la herramienta computacional se necesita una identificación previa, con el fin de que los algoritmos tengan un espacio de búsqueda mucho más limitado en el cual se garantice que se encuentran los valores adecuados para la optimización de cada uno de los parámetros característicos del modelo de sistema; dejando de lado el uso de un espacio de búsqueda fijo para todo el algoritmo. De esta manera cada parámetro característico a optimizarse tendrá su propio espacio de búsqueda en un rango aumentado en el 50% más y menos de los valores obtenidos de la identificación previa.

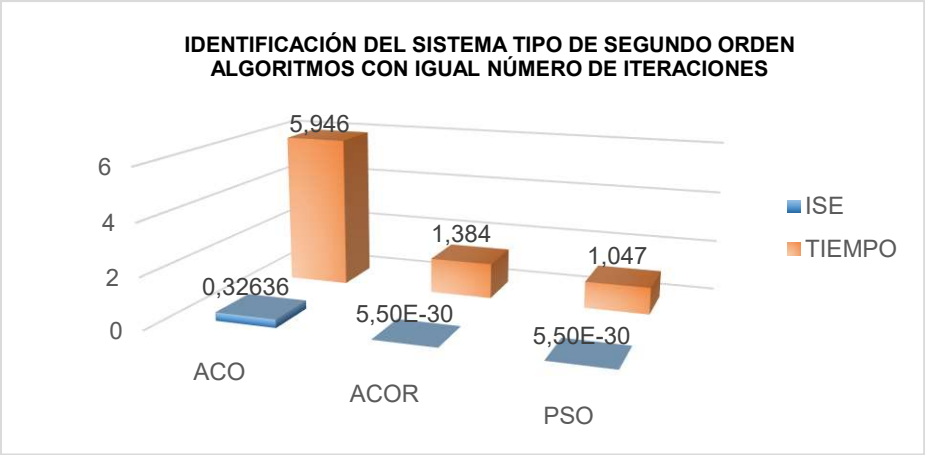


Figura 3.9. Comparación de los algoritmos en identificación del sistema de segundo orden.

La Figura 3.9 muestra el gráfico de barras en el cual, debido a la naturaleza heurística y la forma de desarrollo del algoritmo de optimización ACO, se visualiza una diferencia muy significativa tanto en el tiempo ejecución empleado, para alcanzar la condición de parada de un máximo de iteraciones, como el resultado de su índice de error. Esto se relaciona con el hecho de que este algoritmo es constructivo y no evolutivo como lo son el algoritmo ACO-R y PSO. A pesar de que estos tres algoritmos posean algún artificio que les ayuda a en su heurística de búsqueda de los valores más cercanos al óptimo global; para el algoritmo ACO se necesitaría un coeficiente adicional que le permita una mejor relación entre su convergencia y divergencia al encontrar una mejor solución para el problema de optimización. Esto pone en evidencia que la volatilidad de la feromona y en sí la dinámica constructiva del algoritmo de optimización no es lo suficiente robusta en comparación con los otros dos algoritmos bioinspirados utilizados.

3.4 RESULTADOS DE LA IDENTIFICACIÓN DE SISTEMAS DE RESPUESTA INVERSA

A continuación, se discuten los resultados obtenidos de la identificación del sistema tipo de respuesta inversa, bajo los tres algoritmos de optimización implementados. La Figura 3.10 muestra el resultado obtenido mediante la herramienta SysID por el algoritmo de optimización PSO; la misma que evidencia una estimación de parámetros del 64.13% de exactitud a los parámetros originales, y posee 30% mayor de exactitud comparado con la estimación del modelo por Balaguer realizado analíticamente.

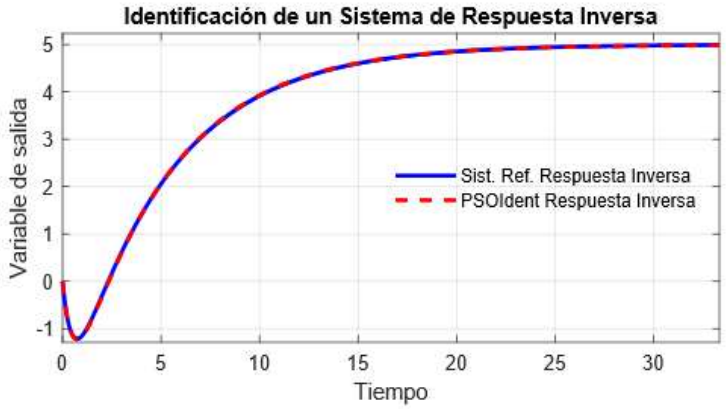


Figura 3.10. Identificación por el algoritmo PSO al sistema tipo de respuesta inversa.

En los sistemas de respuesta inversa, como se expuso en el Capítulo 2, se utiliza la forma recursiva de Balaguer para estimar sus parámetros característicos, esto permite reducir los parámetros de optimización de cinco a tres. Este cambio de parámetros a optimizar representa una ventaja significativa al reducir el tiempo de ejecución del algoritmo.

Tabla 3.7 Parámetros estimados por SysID

	PARÁMETROS ESTIMADOS PARA SISTEMAS DE RESPUESTA INVERSA						
	Referencia	**Valores Promedio**					
		ACO	**Error (%)**	**ACO-R**	**Error (%)**	**PSO**	**Error (%)**
KRI	5	-	-	-	-	-	-
τ	0.5	1.11	124.80	0.79	57.74	0.757	51.48
$b\tau$	2.2	2.77	25.91	2.84	29.23	2.832	28.73
$a\tau$	5	7.06	41.16	6.41	28.28	6.370	27.40

En la Tabla 3.7 se presenta los valores más cercanos a los óptimos identificados por la herramienta SysID, junto con la referencia del sistema tipo de respuesta inversa en su forma recursiva de Balaguer; de la cual se observa que todos los parámetros presentan un error relativo considerable, y aun así el algoritmo PSO y ACO-R son los de mejor resultado. El algoritmo ACO, por su parte presenta la mejor optimización del parámetro $b\tau$ que es la estimación que menor error presenta. La causa de que los errores en la estimación de parámetros de este sistema tipo sean mayores, al de los otros casos, es porque conforme aumenta la dificultad en el modelo se verá reflejado en el mayor valor del índice de error.

En la Tabla 3.8 se demuestra lo que se expuso en el párrafo anterior, al compararla con la Tabla 3.6 del promedio de identificación del sistema tipo de segundo orden; porque se evidencia que a pesar de que poseen el mismo número de variables a optimizar y que la respuesta gráfica posea similitudes en formas sinusoidales, se tiene un menor número de iteraciones para alcanzar la condición de parada de un índice de error menor al 0,01. Al mismo tiempo, que para la condición de parada de un máximo de 200 iteraciones los

resultados del índice de error en comparación al tiempo invertido no reflejan una ventaja significativa.

Tabla 3.8 Promedio de la identificación del sistema tipo de respuesta inversa.

	CONDICIONES DE PARADA				
	Índice de error < 0.01			Máximo de 200 iteraciones	
	Valores Promedio			Valores Promedio	
	MSE	Tiempo [min]	Número iteraciones	MSE	Tiempo [min]
ACO	0.00506	2.9421	25.1	3.26E-03	9.61
ACO-R	0.00714	0.2191	7.3	1.38E-03	3.23
PSO	0.00679	0.1971	5.1	4.77E-04	2.38

De los datos promedio, de la Tabla 3.8, se dice que el algoritmo PSO utiliza solo el 7,44% del tiempo de ejecución y el 20,31% de las iteraciones totales del algoritmo ACO; de igual forma ocupa un 10,04% menos del tiempo de simulación y 30,14% menos del número de iteraciones requerido en el algoritmo ACO-R. En la condición de parada por el número de iteraciones, los valores del índice de error presentados en la tabla anterior son similares entre sí, con lo cual prevalece que el mejor algoritmo de identificación es el algoritmo PSO, por tener el menor valor del índice de error.

Al igual que en las pruebas realizadas con anterioridad el algoritmo PSO posee la mejor respuesta de optimización con el valor promedio más pequeño entre el número de iteraciones, tiempo de ejecución del algoritmo e índice de error, como se muestra en la Figura 3.11.

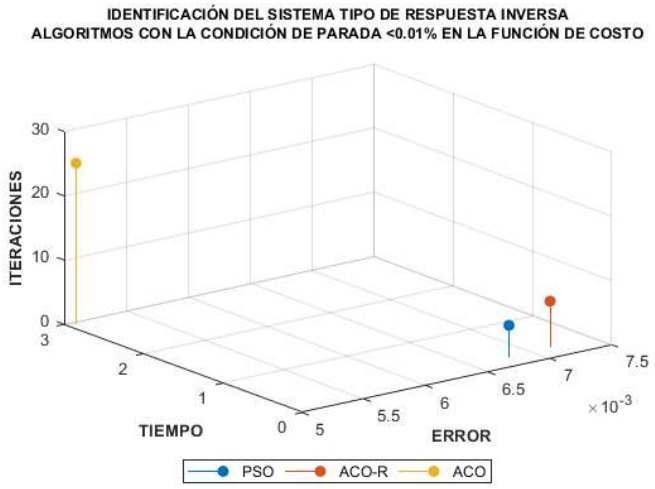


Figura 3.11. Comparación del promedio de identificación de los algoritmos implementados.

Del gráfico de barras de la Figura 3.12 se aprecia lo que se expuso en párrafos anteriores, donde el costo computacional empleado por el algoritmo ACO es el mayor de todos. Por

otra parte, a pesar de que se realizó las pruebas a un mayor número de iteraciones, no se obtiene un mejor índice de error en comparación con la condición de parada anterior. Igual que en los otros casos para superar esta dificultad se realiza una identificación previa de un modelo de sistema recursivo para los sistemas de respuesta inversa para la identificación de los modelos de respuesta inversa se presentan mejores resultados al poseer un rango semejante de error.

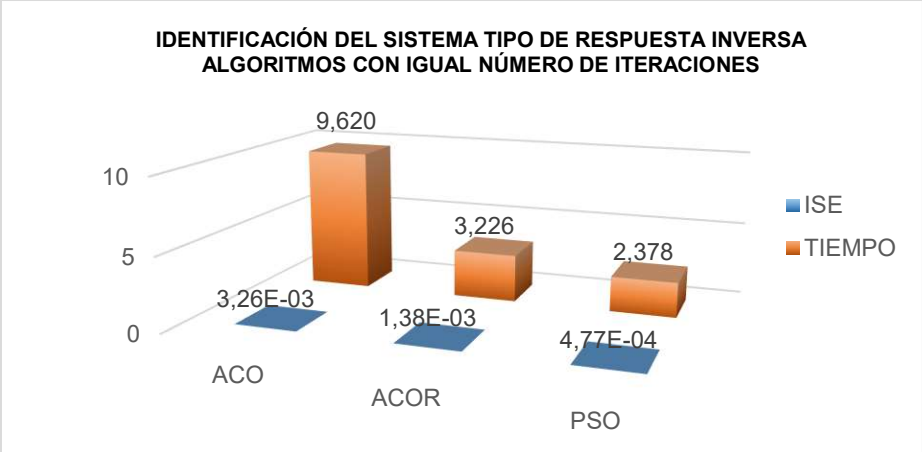


Figura 3.12. Gráfica de barras del error y tiempo de identificación a una condición de parada de 200 iteraciones.

Se destaca que de todas las pruebas realizas, ninguno de los algoritmos implementados cae en el problema de óptimos locales, esto se refleja en que tanto el algoritmo ACO, PSO y ACO-R poseen valores similares en el error MSE, en las pruebas realizadas bajo las dos condiciones de parada.

Dentro de los algoritmos ACO-R y PSO se utiliza funciones de probabilidad mediante comandos propios del software de MatLab®, como son la distribución normal aleatoria y la distribución uniforme aleatoria estas distribuciones a veces dan valores negativos que no pueden ser utilizados, algunas veces por la constitución del algoritmo y otras por el modelo de sistema como es el caso de los modelos de primer orden con retardo en su parámetro tiempo de retardo. Sin embargo, se tiene sistemas con ganancia negativa en su parámetro K y en los sistemas de respuesta inversa al parámetro $b\tau$ se lo considera positivo, pero cuando para el sistema es de ganancia positiva será negativo. Entonces por las consideraciones anteriores se modificó a estos algoritmos para que siempre que exista una probabilidad negativa se transforme en positiva, en los casos de primer orden con retardo y respuesta inversa; lo cual no ha interferido en la optimización de parámetros como se puede apreciar en los resultados obtenidos de todas las pruebas realizadas a los cuatro tipos de sistema lineales.

De las pruebas de identificación de sistemas realizadas en simulación, para los cuatro tipos de modelos de sistemas lineales mediante los tres algoritmos de optimización, se evidencia que el algoritmo PSO es el que presenta un mejor desempeño. La Tabla 3.9 muestra el análisis comparativo en porcentaje, de las principales características de identificación: tiempo de ejecución del algoritmo y número de iteraciones, cuando la condición de parada es del índice de error menor al 0.01.

Tabla 3.9 Análisis comparativo de los algoritmos implementados en SysID

	CONDICIÓN DE PARADA: índice de error < 0.01							
	Valores Promedio [%]							
	Primer Orden		**Primer Orden con Retardo**		**Segundo Orden**		**Respuesta Inversa**	
	Tiempo	**Iteraciones**	**Tiempo**	**Iteraciones**	**Tiempo**	**Iteraciones**	**Tiempo**	**Iteraciones**
ACO	78.73	-22.22	97.61	96.57	95.32	81.71	93.30	79.68
ACO-R	12.29	2.94	8.39	15.05	49.12	32.09	10.04	30.14

De la tabla anterior se muestra que el algoritmo PSO tiene un tiempo de ejecución 91,24% mejor en la identificación de los sistemas tipo y un número de iteraciones del 58,93% mejor en comparación con el algoritmo ACO; mientras que en comparación con el algoritmo ACO-R presenta el 19,96% en el tiempo de ejecución y el 20,06% en el número de iteraciones. Además, se revela un valor negativo para el porcentaje del mejor número de iteraciones que posee el algoritmo PSO con respecto al ACO, esto es por los resultados promedio obtenidos en la Tabla 3.2, en la cual el algoritmo ACO posee el menor número de iteraciones, pero a un mayor tiempo de ejecución. Con respecto al índice de error cuadrático medio, el algoritmo ACO posee el menor en la identificación de los sistemas de respuesta inversa, y el algoritmo ACO- R en los sistemas de primer orden, en los casos restantes el mejor índice lo presenta el algoritmo PSO.

3.5 RESULTADOS DEL CASO PRÁCTICO

En el siguiente apartado, se describe los resultados obtenidos de la identificación del modelo a un sistema real realizada mediante SysID, y su respectiva comparación con los resultados obtenidos de la herramienta propia de MatLab® System Identification Toolbox antes llamada IDENT. Los datos adquiridos son de un motor DC al cambio de referencia de una entrada paso de voltaje de 0- 12 VDC.

3.5.1 RESULTADOS DE LA HERRAMIENTA SYSID

Los resultados obtenidos por la herramienta SysID en la identificación del modelo del motor DC se presentan en la Tabla 3.10. De la cual se aprecia que el algoritmo ACO-R posee el menor índice de error, al igual que el menor error entre los parámetros de la identificación

previa (considerados como reales) y los parámetros estimados. En cambio, el menor número de iteraciones lo posee el algoritmo ACO y el menor tiempo de ejecución es el del algoritmo PSO. En consecuencia, el algoritmo ACO- R es el mejor algoritmo para la identificación del modelo de un motor DC.

Tabla 3.10 Cuadro comparativo de la identificación de la planta real

	PARÁMETROS ESTIMADOS POR SysID		RECURSOS UTILIZADOS POR CADA ALGORITMO		
	K	Tao	MSE	Iteraciones	Tiempo
ACO	92.73	0.01896	3673	44	1.873
ACO-R	92.80	0.01654	3659	66	0.185
PSO	92.82	0.01654	3663	52	0.106
IDENTIFICACIÓN PREVIA	87.30	0.05940	-	-	-

En la Tabla 3.11 se corrobora con los resultados obtenidos de la Tabla 3.10; porque el algoritmo ACO- R presenta el menor porcentaje de error y el mejor porcentaje de exactitud dado por la normalización del índice cuadrático de error (NRMSE), lo que demuestra que es el mejor algoritmo para encontrar la estimación más cercana al óptimo para los parámetros de los modelos de sistema lineal. El índice NRMSE es exclusivo para el caso práctico, con el propósito de poseer un valor más comparable la identificación por SysID con el Toolbox System Identification de MatLab®.

Además, se presenta el error de cada parámetro en porcentaje, considerado como real al estimado previamente con la herramienta computacional.

Tabla 3.11 Cuadro comparativo de los porcentajes de estimación del modelo

	PARÁMETROS ESTIMADOS POR SysID		INDICES Y PORCENTAJES DEL MODELO IDENTIFICADO		
	Error K %	Error Tao %	Error %	ISE	Exactitud %
ACO	6.219	219.19	5.348	13360	64.35
ACO-R	6.300	178.45	5.325	13310	64.38
PSO	6.323	178.45	5.330	13320	64.37

La Ecuación 3.1 presenta el modelo de sistema en función de transferencia que se muestra en la herramienta computacional al finalizar la estimación, la Ecuación 3.2 es la misma que la anterior con el termino s en 1. La exactitud total del modelo identificado es del 64.38%. con referencia. El índice de error MSE es de 3659, su porcentaje de error está en 5.235%, siendo estos los menores de las pruebas realizada. En la Figura 3.13 se muestra el sistema identificado en rojo y el sistema referencia en rojo.

$$G(s) = \frac{92.8}{0.01654s + 1} \tag{3.1}$$

$$G(s) = \frac{5610}{s + 60.46} \tag{3.2}$$

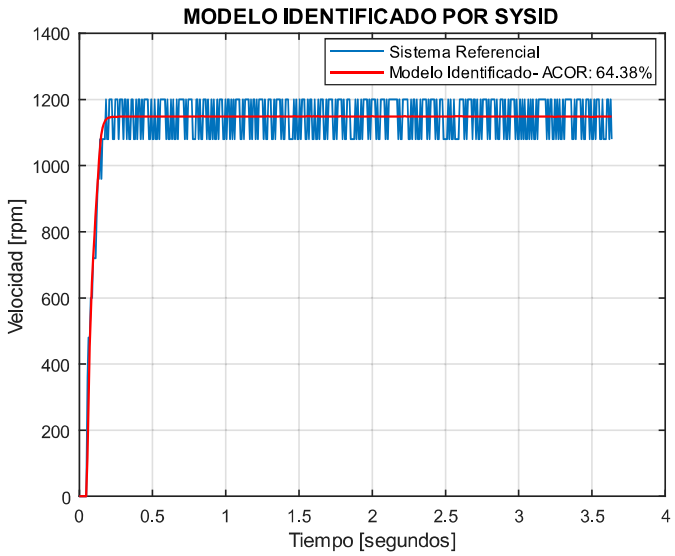


Figura 3.13. Identificación del motor dc a un sistema de primer orden mediante ACO.

Se destaca que la aproximación de este sistema modelo es para un sistema de primer orden; no se le considera de retardo porque el estímulo en la entrada y la reacción del sistema es en el mismo instante. Si se aprecia el sistema con un ligero retardo es debido a la entrada del sistema referencia.

De igual manera se identificó un modelo de sistema de segundo orden para el caso práctico mediante la optimización ACO-R de la herramienta SysID. Los resultados de la identificación se muestran en la Tabla 3.12 y Tabla 3.13. Se tiene un índice de error cuadrático medio del 3662.8 esto se puede interpretar como una exactitud del 64.37% del sistema identificado con respecto al sistema real.

Tabla 3.12 Cuadro comparativo de la identificación de la planta real

	PARÁMETROS ESTIMADOS POR SysID			**RECURSOS UTILIZADOS POR CADA ALGORITMO**		
	K	ε	W_n	**MSE**	**Iteraciones**	**Tiempo**
ACO-R	92.79	6.322	764.2	3662.8	200	0.641
IDENTIFICACIÓN PREVIA	23.57	0.5731	1.9242	-	-	-

La identificación solo se muestra con el algoritmo ACO- R porque este fue el único algoritmo que pudo estimar los parámetros más cercanos a los óptimos del sistema de segundo

orden para el caso práctico. El algoritmo ACO y PSO no tuvieron buenos resultados, motivo por el cual no se los considera para el análisis respectivo.

De las pruebas realizadas tanto para este caso práctico, como para simulación, el algoritmo ACO- R, al poseer una metaheurística de distribución de densidades de probabilidad, ha sido beneficioso porque le permite al algoritmo redirigir su espacio de búsqueda a valores promotores donde se pueden encontrar los valores más cercanos a los óptimos. Por otra parte, los algoritmos restantes no poseen algún método que les permita cambiar su espacio de búsqueda una vez definido en la herramienta.

Los errores de cada uno de los parámetros característicos del sistema identificado se los muestra en la Tabla 3.13. Para le cálculo de estos, se consideró como reales a los obtenidos de la identificación previa por eso son de porcentajes elevados. Sin embargo, si se tuviese los valores de las otras identificaciones se observaría si el sistema encontrado por ACO- R corresponde a los valores más cercanos a los óptimos.

Tabla 3.13 Cuadro comparativo de los porcentajes de estimación del modelo

	PARÁMETROS ESTIMADOS POR SysID			**INDICES Y PORCENTAJES DEL MODELO IDENTIFICADO**		
	ERROR K %	**ERROR ε %**	**ERROR W_n%**	**ERROR %**	**ISE**	**EXACTITUD %**
ACO-R	293.72	919.36	396.15	5.327	13330	64.37

La Ecuación 3.3. presenta el sistema en función de transferencia y la Figura 3.14 muestra la comparación gráfica del modelo identificado con los datos reales obtenidos.

$$G(s) = \frac{54190000}{s^2 + 9662s + 584000} \tag{3.3}$$

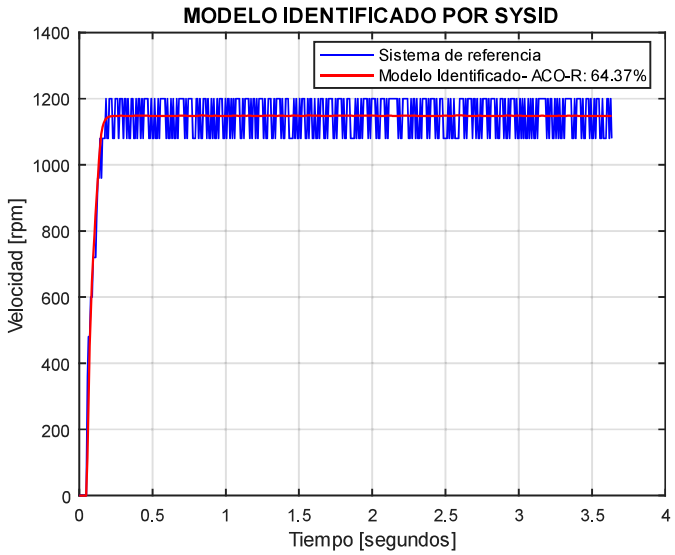


Figura 3.14. Comparación del modelo de segundo orden optimizado y el sistema real.

3.5.2 RESULTADOS DE LA HERRAMIENTA IDENT

Para el análisis de resultados y con la finalidad de obtener una comparación adecuada se realizó un estudio de los diferentes métodos que posee la herramienta System Identification de MatLab®, para determinar el método que sería el adecuado en la comparación.

De los métodos que utiliza la herramienta propia de MatLab para identificar sistemas, está el Método de Punto Interior (Interior-Point Method) mediante la función fmincon. Este método es un algoritmo que resuelve problemas de optimización convexa lineal y no lineal, trata de minimizar una función lineal sobre un conjunto convexo, mediante el establecimiento de barreras en el conjunto factible y garantizando que el número de iteraciones del algoritmo está limitado por el polinomio de la solución y su precisión [42].

Al igual que SysID, la herramienta System Identification posee un método de inicialización con el cual se pretende encontrar un espacio de búsqueda prometedor en el cual se encuentren los valores más cercanos a los óptimos del modelo. Para las pruebas se utiliza las condiciones iniciales estimadas por la aproximación de variables instrumentales (IV-Instrumental Variable Estimation). El método de variables instrumentales se utiliza para estimar las relaciones de los datos de referencia y al trabajar con modelos lineales cuando hay una relación entre el error de los parámetros y los parámetros estimados [43].

Para el uso de System Identification de MatLab®, después de escoger el método de inicialización: IV, el método de búsqueda: Interior- Point y las condiciones iniciales por estimación; no se especificó ningún otro parámetro, como tampoco se modificó los valores predeterminados para el método de búsqueda, esto con el objetivo de ser imparcial en la manipulación de las dos herramientas y que las ambas herramientas al compararse estén sujetas a las mismas condiciones predeterminadas con las que fueron graduadas.

La Figura 3.15 y la Ecuación 3.4 presentan el resultado de identificación de un modelo de primer orden para el caso práctico, este posee la exactitud del 63.4% y un MSE de 3865; este índice parece muy elevado, pero es por las dimensiones de los datos adquiridos. Los reportes de esta identificación están en el Anexo D.

$$G(s) = \frac{7011}{s + 75.55} \tag{3.4}$$

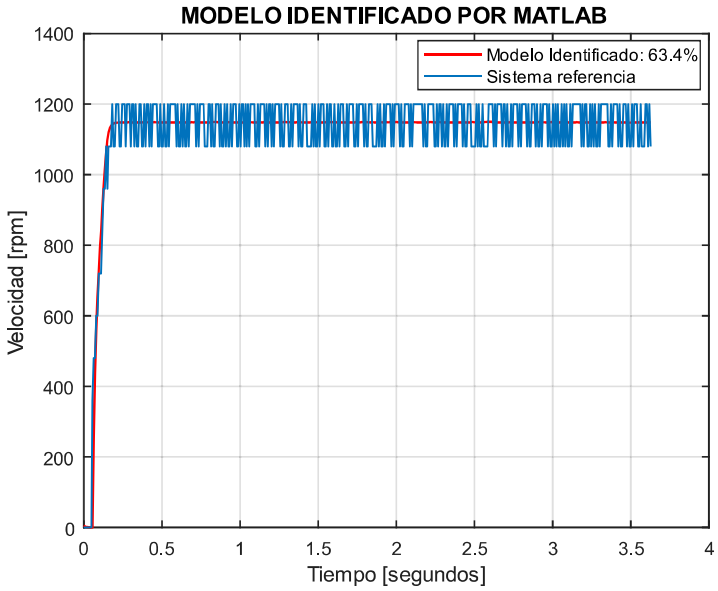


Figura 3.15. Comparación del modelo identificado y es sistema real.

De igual manera se identificó un sistema de segundo orden para los datos adquiridos, la Ecuación 3.5 y Figura 3.16 muestra el resultado en función de transferencia. El modelo posee una exactitud del 54.8%, con un índice de error cuadrático medio de 5894.

$$G(s) = \frac{359000}{s^2 + 186.3s + 3856} \tag{3.5}$$

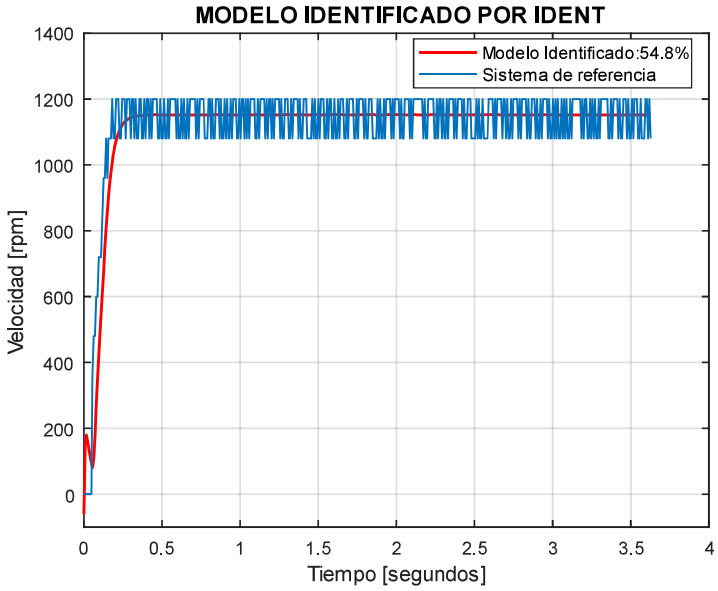


Figura 3.16. Comparación del modelo real y el modelo identificado por IDENT.

3.5.3 COMPARACIÓN DE RESULTADOS CON IDENT DE MATLAB

Se recalca que en la realización de esta herramienta computacional se empleó la versión 2019b del software de MatLab®, versión en la cual la función IDENT ha sido remplazada por la aplicación System Identification. Sin embargo, son exactamente iguales. Como resultado se tiene una comparación del funcionamiento de SysID con la aplicación antes mencionada en las funcionalidades que competen dentro del trabajo de titulación como son:

- Identificación de la función de transferencia del modelo.
- Identificación del mejor modelo lineal.

De la comparación de los resultados obtenidos con la herramienta de Matlab y SysID, se obtiene un 0.98% de mejora del modelo de sistema de primer orden y un 9,57% en el modelo de segundo orden identificado por SysID; en base a estos resultados se infiere que la SysID realiza una mejor identificación de sistemas reales que la herramienta System Identification de MatLab.

En las dos identificaciones se obtuvo que el mejor modelo de sistema para los datos adquiridos es el modelo de primer orden, a pesar de que los modelos de segundo orden también han demostrado muy buenas aproximaciones. La Figura 3.17 muestra la comparación gráfica de los modelos de sistema de primer orden identificados por las dos herramientas y la referencia de velocidad del modelo real.

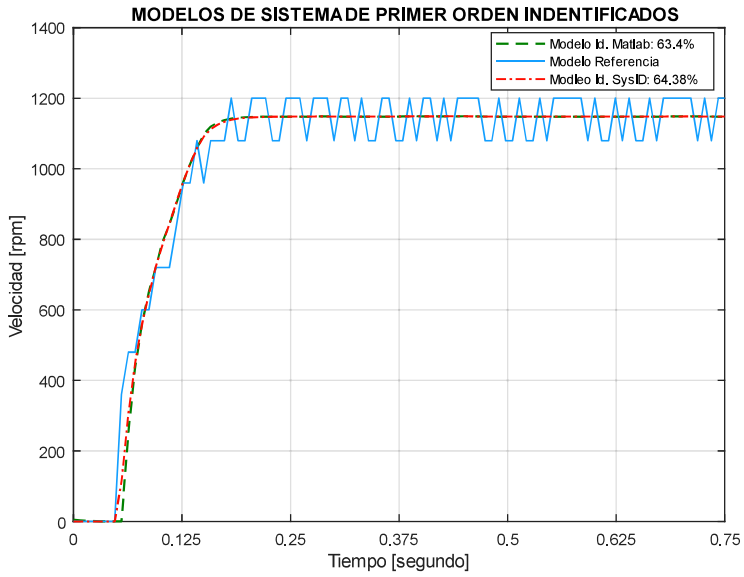


Figura 3.17. Comparación gráfica de la salida de los modelos identificados de primer orden.

La Figura 3.18 muestra la comparación gráfica de los modelos de sistema de segundo orden identificados por las herramientas y la referencia de velocidad del modelo real.

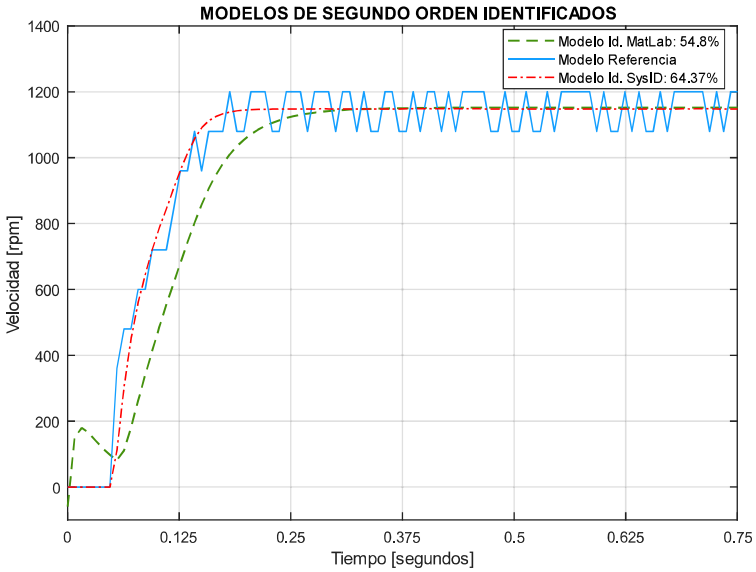


Figura 3.18. Comparación gráfica de la salida de los modelos identificados de segundo orden.

3.6 RESULTADOS DE LA IDENTIFICACIÓN DE SISTEMAS DE NO LINEALES

Como parte complementaria de la herramienta computacional, se probó su funcionamiento para la estimación de parámetros de modelos no lineales que se puedan aproximar a su modelo de orden reducido dadas sus condiciones iniciales. Los resultados obtenidos se presentan a continuación.

- Sistema Tanque de Mezclado bajo -5% de cambios a la entrada

El sistema tanque de mezclado tomado de [38] posee la propiedad de cambiar su modelo dependiendo de la variación a la cual se someta su entrada.

La Tabla 3.14 muestra los resultados obtenidos de la estimación de parámetros del sistema sometido a una variación del 10% de su entrada a un cambio de referencia negativo.

Tabla 3.14. Tabla comparativa de los parámetros optimizados

	PARÁMETROS ESTIMADOS POR SysID			RECURSOS UTILIZADOS POR CADA ALGORITMO		
	K	Épsilon	Wn	MSE	Iteraciones	Tiempo
ACO	-0.000135	0.7565	3.858	2.44e-05	43	11.178
ACO-R	-0.0001326	0.789	4.697	7.962e-08	4	0.113
PSO	-0.001348	0.7632	3.828	2.187e-08	4	0.194

Como se puede observar el algoritmo PSO es el que presenta menor índice, seguidamente del algoritmo ACO-R que lo realiza en un menor tiempo y finalmente el algoritmo ACO. Los tres algoritmos presentan aproximaciones muy similares, pero en porcentaje el algoritmo PSO es 26.4% mejor al ACO-R y 110% que el algoritmo ACO.

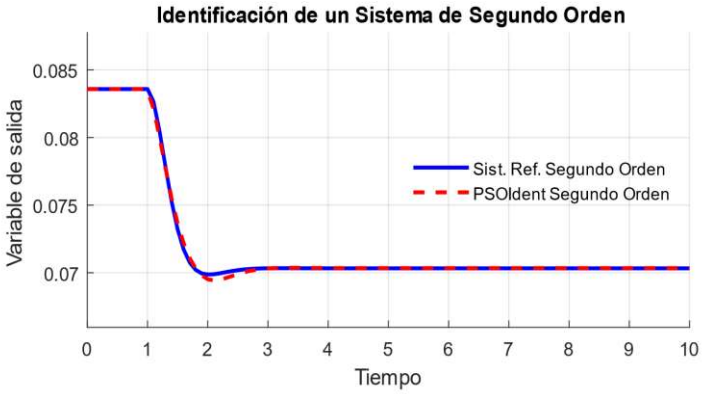


Figura 3.19. Resultado de la identificación del sistema no lineal Tanque de Mezclado.

La Figura 3.19 muestra la respuesta del sistema Tanque de Mezclado a la optimización por el algoritmo PSO y la Ecuación 3.6 presenta el modelo de sistema identificado en función de transferencia.

$$G(s) = \frac{-0.001975}{s^2 + 5.842\,s + 14.651} \tag{3.6}$$

- Sistema Tanque de Mezclado bajo 10% de cambia a la entrada

En base a lo descrito anteriormente se presenta el sistema Tanque de Mezclado a una variación del 10% en su entrada, el sistema reacciona como un de segundo orden subamortiguado. La Tabla 3.15 muestra los resultados de la identificación, así el algoritmo PSO y ACO- R iguales resultados, pero el algoritmo PSO lo realiza en un menor tiempo y en menor número de iteraciones por lo cual su resultado se considera como el mejor.

Tabla 3.15 Tabla comparativa de los parámetros optimizados

	PARÁMETROS ESTIMADOS POR SysID			**RECURSOS UTILIZADOS POR CADA ALGORITMO**		
	K	**Épsilon**	**Wn**	**ISE**	**Iteraciones**	**Tiempo**
ACO	0.0003266	0.1728	2.724	0.001095	43	0.448
ACO-R	0.000325	0.175	2.767	6.101e-7	93	0.369
PSO	0.000325	0.175	2.767	6.101e-7	74	0.267

La Figura 3.20 muestra la salida del modelo identificado con la referencia del sistema no lineal se refleja una exactitud del 97.5%. La Ecuación 3.7 presenta el sistema de segundo orden resultante de la identificación en función de transferencia.

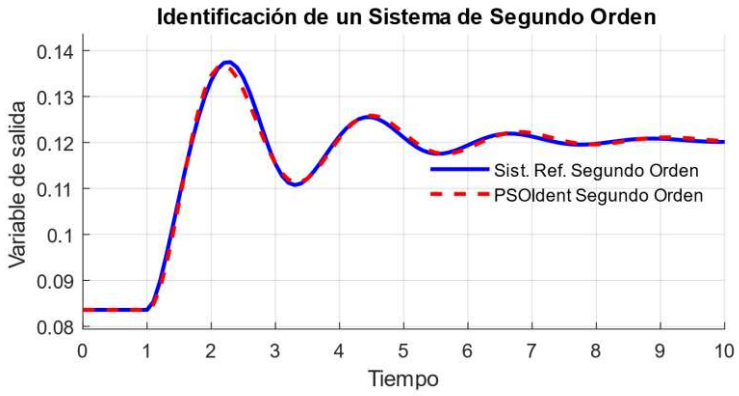


Figura 3.20. Resultado de la identificación del sistema no lineal a +10% de variación.

$$G(s) = \frac{0.002489}{s^2 + 0.968\,s + 7.657} \tag{3.7}$$

- Sistema Tanque de Mezclado al -10% de cambio en la entrada

Para este cambio de referencia el sistema reacciona como un sistema de segundo orden sobre amortiguado, pero con ganancia negativo. La Tabla 3.16 muestra los resultados de la identificación para este caso el algoritmo ACO-R se evidencia como el mejor al tener un índice de error 304,16% de que algoritmo PSO a pesar de que tenga un costo computacional más elevado que los otros dos algoritmos.

Tabla 3.16 Tabla comparativa de los parámetros optimizados

	PARÁMETROS ESTIMADOS POR SysID			RECURSOS UTILIZADOS POR CADA ALGORITMO		
	K	**ε**	**Wn**	**MSE**	**Iteraciones**	**Tiempo**
ACO	-0.0002609	0.7769	3.329	6.667e-4	42	0.395
ACO-R	-0.0002597	1.289	6.487	9.996e-8	141	0.499
PSO	-0.0002592	0.8706	3.444	4.04e-07	62	0.225

La Ecuación 3.8 y la Figura 2.24 muestran los resultados obtenidos de la identificación y estimación de parámetros mediante la herramienta SysID.

$$G(s) = \frac{-0.01093}{s^2 + 16.721s + 42.08} \tag{3.8}$$

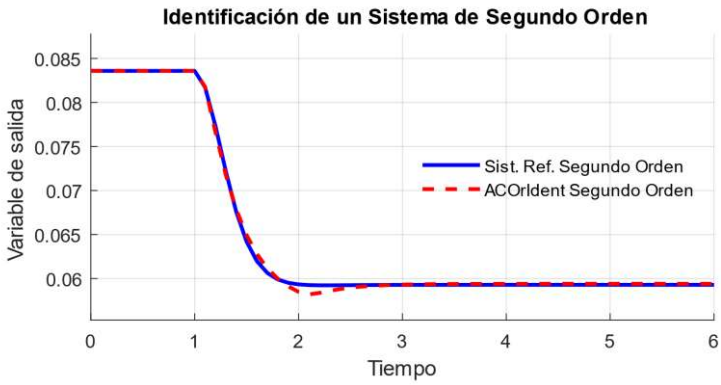


Figura 3.21. Resultado de la identificación del sistema no lineal.

- Sistema Tanque de Mezclado al +5% de cambio en la entrada

Finalmente, el sistema Tanque de Mezclado presenta una última variación cuando se somete a un cambio de referencia de +5%. La Tabla 3.17 muestra los resultados obtenidos en la cual se evidencia que el algoritmo ACO-R realiza la mejor identificación del sistema superior en el 97% a la realizada por el algoritmo PSO.

Tabla 3.17 Tabla comparativa de los parámetros optimizados

	PARÁMETROS ESTIMADOS POR SysID			**RECURSOS UTILIZADOS POR CADA ALGORITMO**		
	K	**ε**	**Wn**	**MSE**	**Iteraciones**	**Tiempo**
ACO	0.0001494	0.3474	3.487	0.0001523	44	0.457
ACO-R	0.0001507	0.3909	3.529	7.91e-08	6	0.110
PSO	0.0001537	0.4097	3.284	1.564e-07	69	0.224

La Figura 3.22 y la Ecuación 3.9 muestran los resultados de la identificación del sistema. Un punto destacable de este sistema es que su respuesta es muy pequeña lo cual hace que para la mínima variación entre sus parámetros resulta en una diferencia muy grande en el índice de error.

$$G(s) = \frac{0.001877}{s^2 + 2.758s + 12.45} \tag{3.9}$$

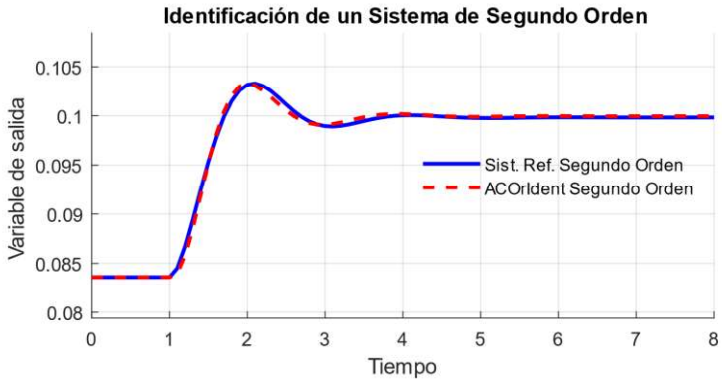


Figura 3.22. Resultado de la identificación del sistema no lineal a una variación de +5%.

- Convertidor Buck- Boost al +5% de cambio en la entrada

El último modelo no lineal identificado por la herramienta computacional desarrollada es el conversor Buck and Boost, a diferencia del otro sistema su respuesta es demasiado rápida, pero su amplitud es el orden de las decenas esto provoca que su variación en el índice de error no sea tan evidente a pesar de que los resultados de la identificación no sean tan similares entre ellos.

La Tabla 3.18 presenta el resultado de la optimización, el mejor algoritmo de identificación es el ACO-R al presentar el menor índice de error, seguido del algoritmo ACO que presenta un resultado bueno, pero a un costo computacional 400% mayor que el anterior.

Tabla 3.18 Tabla comparativa de los parámetros optimizados

	PARÁMETROS ESTIMADOS POR SysID				**RECURSOS UTILIZADOS POR CADA ALGORITMO**		
	K	**Tao**	**Coef. b**	**Coef. a**	**MSE**	**Iteraciones**	**Tiempo**
ACO	5.039	0.06586	0.7503	0.01886	0.001425	47	3.305
ACO-R	5.042	0.067	0.7462	0.006929	0.000189	91	0.753
PSO	5.091	0.0704	0.01283	0.6764	0.002205	6	0.107

La Figura 3.23 muestra el resultado gráfico de la identificación del modelo no lineal Buck and Boost y la Ecuación 3.10 presenta el modelo optimizado en función de transferencia.

$$G(s) = \frac{5.042(-0.04999\,s\ +1)}{(0.067\,s\ +\ 1)(0.0004643\,s\ +\ 1)} \tag{3.10}$$

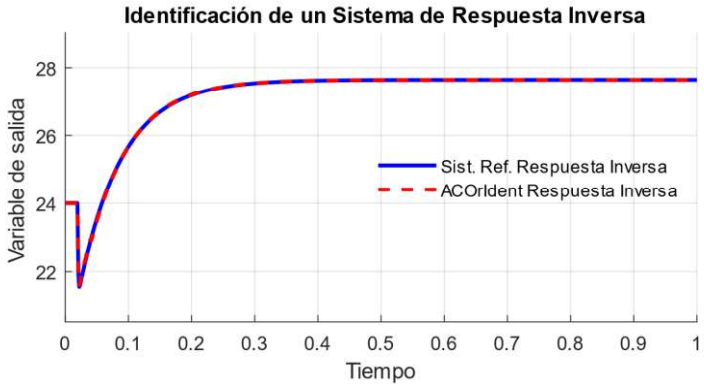


Figura 3.23. Resultado de la identificación del conversor Buck-Boost

4. CONCLUSIONES Y RECOMENDACIONES

En esta sección se exponen las conclusiones y recomendaciones que se han desencadenado de la realización del presente trabajo de titulación.

4.1 CONCLUSIONES

- De la revisión bibliográfica se concluye que debido a las diferencias heurísticas entre los algoritmos ACO y PSO no es posible una comparación ecuánime; por tanto, es imprescindible la implementación de un tercer algoritmo, ACO- R, que posea las características heurísticas de construcción y evolución.

- La distribución inicial del componente heurístico es diferente en cada algoritmo y depende del parámetro característico que se estime; así, para el parámetro tiempo de retardo siempre debe ser positiva, en el algoritmo ACO es uniforme, en el algoritmo PSO es aleatoria y para el algoritmo ACO-R se tiene una distribución normal de densidades de probabilidad basados en un kernel gaussiano.

- Los algoritmos implementados necesitan un espacio de búsqueda definido; por tal motivo se desarrolla una identificación previa para los cuatro tipos de modelos de sistemas lineales; en base a los métodos de estimación de dos puntos, ecuaciones características de los parámetros de sistemas de segundo orden subamortiguados y finalmente por el método de Balaguer para sistemas de respuesta inversa.

- La realización de una identificación previa, en la herramienta computacional desarrollada, contribuye a disminuir el tiempo de simulación y a reducir el número de iteraciones, con respecto a la ejecución de solo la herramienta computacional.

- En las pruebas simuladas, el algoritmo PSO presenta un mejor rendimiento en comparación con el algoritmo ACO-R, con el 19,96% en el tiempo de ejecución y el 20.06% en el número de iteraciones; y es también superior al algoritmo ACO con el 91,24% en tiempo de ejecución, junto al 58,93% en el número de iteraciones.

- En comparación a los modelos identificados por la herramienta System Identification de MatLab®; SysID logró, con el algoritmo ACO-R, identificar el modelo de un motor DC con una exactitud mayor en 0.98% para el modelo de primer orden y al 9,87% en el modelo de segundo orden.

- SysID cumple con la identificación de modelos de sistemas y estimación de parámetros; por tanto, debe ser considerada como un método alternativo para la identificación de parámetros de modelos de sistemas lineales y de sistemas no lineales en un punto de operación establecido.

4.2 RECOMENDACIONES

- Realizar una identificación previa para delimitar el rango de valores del espacio de búsqueda de los parámetros característicos a optimizar; esto es beneficioso porque se presenta una reducción del tiempo de ejecución del algoritmo y por tanto del costo computacional del mismo.

- Utilizar el mismo lenguaje de programación en la implementación de los algoritmos al igual que una configuración inicial semejante, como es en el espacio de búsqueda, la metodología de ajuste de parámetros y las condiciones de parada ofrecen una comparación justa entre los algoritmos optimización.

- Seleccionar el tiempo de muestreo idóneo de acuerdo con la planta real, y asegurarse de que la captura de datos sea en intervalos de tiempo exactamente iguales; para esto prestar total atención en los elementos físicos (cables) que van directo a las señales de conversión del ADC y captura de pulsos; esto con el fin de realizar una buena adquisición de datos.

- Realizar para la prueba práctica una fuente controlada de voltaje, mediante la cual se puede realizar un cambio del 10% en la entrada del sistema, después de que esta haya alcanzado su estabilidad; esto permitirá obtener una curva de reacción, la cual puede ser identificada posteriormente por la herramienta computacional desarrollada.

- Se recomienda revisar el tutorial de la aplicación System Identification del software de MatLab® antes de realizar las comparaciones con SysID, esto para facilitar su manipulación.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] V. Collazo, J. Rodríguez, A. Zambrano, y N. Troncone, "NIS: Una Herramienta Computacional para la Identificación de Sistemas Dinámicos en Lazo Cerrado Empleando Redes Neuronales Artificiales", *UNIVERSIDAD, CIENCIA Y TECNOLOGÍA*, vol. 16, núm. 64, pp. 190–202, sep. 2012.

[2] C. Duarte y J. Quiroga, "System identification of a DC motor using PSO", *Rev. Fac. Ing. Univ. Antioquia*, núm. 55, pp. 116–124, sep. 2010.

[3] María Elena López Guillén, "Identificación de sistemas. Aplicación al modelado de un motor de continua", Universidad del Cauca, Colombia, 2015.

[4] A. Bemporad, "System identification", Trento, 2011-2010.

[5] J. Sedano y J. R. Villar, "Introducción a la identificación de sistemas", *Técnica Industrial. Revista de Ingeniería e Innovación.*, vol. 256, p. 9, abr-2005.

[6] Dimitry Gorinevsky, "Model Identification", presentado en Control Engineering in Industry, Stanford University, ago-2005.

[7] K. Ogata, *Ingeniería de control moderna*, 5a ed. España: Pearson Educación, 2010.

[8] P. Balaguer, V. Alfaro, y O. Arrieta, "Second order inverse response process identification from transient step response", *ISA Trans.*, vol. 50, núm. 2, pp. 231–238, abr. 2011, doi: 10.1016/j.isatra.2010.11.005.

[9] J. Swevers, "Introduction to system identification", *Control Theory*, p. 74.

[10] K. Socha y M. Dorigo, "Ant colony optimization for continuous domains", *Eur. J. Oper. Res.*, vol. 185, núm. 3, pp. 1155–1173, mar. 2008, doi: 10.1016/j.ejor.2006.06.046.

[11] M. B. Ortiz Moctezuma, *Sistemas dinámicos en tiempo continuo: Modelado y simulación*, 1st ed., vol. 1. Universidad Politécnica de Victoria, México: OmniaScience, 2015.

[12] C. A. Smith y S. W. Campbell, *A First Course in Differential Equations, Modeling, and Simulation*. New York: CRC Press, 2014.

[13] O. Camacho y J. Martínez, *Procesos con Retardo de Tiempo Dominante, Análisis y Comparación de Estrategias de Control*. Alemania: Academia Española, 2017.

[14] C. D. Vázquez y B. del Muro Cuéllar, “Estabilización de sistemas de primer orden inestables con retardo que contienen un cero de fase mínima”, *Mem. Congr. Nac. Control Automático 2012*, pp. 238–242, oct. 2012.

[15] J. E. Normey-Rico y E. F. Camacho, *Control of dead-time processes*. London: Springer, 2007.

[16] R. D. Rojas y W. García, “Nuevo método de identificación usando la respuesta escalón para sistemas de fase no mínima de segundo orden”, *Maskana Rev. Científica*, vol. 5, pp. 23–30, ene. 2016.

[17] F. Castrillón Hernández y D. Castellanos Cárdenas, “New tuning rules for PID controllers based on IMC with minimum IAE for inverse response processes”, *DYNA*, vol. 82, núm. 194, pp. 111–118, dic. 2015, doi: 10.15446/dyna.v82n194.46744.

[18] J. Aguilar, *Introducción a los Sistemas Emergentes*, Ed. 1., vol. 1. Universidad de Los Andes, Mérida, Venezuela: Talleres Gráficos Universitarios, Mérida, 2014.

[19] M. Dorigo y G. Di Caro, “The Ant Colony Optimization Meta-Heuristic”, en *New ideas in optimization*, vol. 1, England: McGraw-Hill Ltd., 1999.

[20] M. Dorigo y L. M. Gambardella, “Ant colony system: a cooperative learning approach to the traveling salesman problem”, *IEEE Trans. Evol. Comput.*, vol. 1, núm. 1, pp. 53–66, abr. 1997, doi: 10.1109/4235.585892.

[21] T. Stützle y H. Hoos, “MAX- MIN Ant System.pdf”, *Future Gener. Comput. Syst.*, vol. 16, pp. 889–914, 2000, doi: doi.org/10.1016/S0167-739X(00)00043-1.

[22] C. Nothegger, A. Mayer, A. Chwatal, y G. R. Raidl, “Solving the post enrolment course timetabling problem by ant colony optimization”, *Ann. Oper. Res.*, vol. 194, núm. 1, pp. 325–339, abr. 2012, doi: 10.1007/s10479-012-1078-5.

[23] C. Blum y M. Sampels, “An Ant Colony Optimization Algorithm for Shop Scheduling Problems”, *J. Math. Model. Algorithms*, vol. 3, núm. 3, pp. 285–308, 2004, doi: 10.1023/B:JMMA.0000038614.39977.6f.

[24] M. Reimann, K. Doerner, y R. F. Hartl, “D-Ants: Savings Based Ants divide and conquer the vehicle routing problem”, *Comput. Oper. Res.*, vol. 31, núm. 4, pp. 563–591, abr. 2004, doi: 10.1016/S0305-0548(03)00014-5.

[25] S. U. Seçkiner, Y. Eroğlu, M. Emrullah, y T. Dereli, “Ant colony optimization for continuous functions by using novel pheromone updating”, *Appl. Math. Comput.*, vol. 219, núm. 9, pp. 4163–4175, ene. 2013, doi: 10.1016/j.amc.2012.10.097.

[26] Y. Zhao, J. Wang, y X. Xie, “Continuous Ant Colony Algorithm Based on Entity and Its Convergence”, en *2008 Second International Symposium on Intelligent Information Technology Application*, Shanghai, China, 2008, pp. 80–84, doi: 10.1109/IITA.2008.272.

[27] A. Afshar y S. Madadgar, “Ant Colony Optimization for Continuous Domains: Application to Reservoir Operation Problems”, en *2008 Eighth International Conference on Hybrid Intelligent Systems*, Barcelona, Spain, 2008, pp. 13–18, doi: 10.1109/HIS.2008.121.

[28] J. Kennedy y R. Eberhart, “Particle Swarm Optimization”, p. 7.

[29] K. D. Patiño Caiza, “OBTENCIÓN DE LAS ECUACIONES DE SINTONIZACIÓN DE UN CONTROLADOR DINÁMICO POR MODOS DESLIZANTES (DSMC) PARA SISTEMAS CON RESPUESTA INVERSA APROXIMABLES A UN MODELO DE SEGUNDO ORDEN, UTILIZANDO TÉCNICAS DE COMPUTACIÓN INTELIGENTE”, Escuela Politécnica Nacional, Quito, 2019.

[30] K. M. Hussain, R. A. R. Zepherin, M. S. Kumar, y S. M. G. Kumar, “PID Controller Tuning Methods Comparison with Particle Swarm Optimization for FOPTD System”, vol. 1, núm. 4, p. 6.

[31] National Instruments, “¿Qué es Adquisición de Datos?”, *National Instruments*, 21 01 20. [En línea]. Disponible en: https://www.ni.com/data-acquisition/what-is/esa/.

[32] MathWorks Inc., “MathWorks”, *MatLab para inteligencia artificial*, 21 20 20. [En línea]. Disponible en: https://la.mathworks.com/?s_tid=gn_logo.

[33] Arduino, “About Us”, *Arduino*, 21 01 20. [En línea]. Disponible en: https://www.arduino.cc/en/Main/AboutUs.

[34] B. P. Cajamarca Echeverría, “DISEÑO E IMPLEMENTACIÓN DE CONTROLADORES CLÁSICOS Y ROBUSTOS EN UNA TARJETA EMBEBIDA, APLICADOS A UN MODELO SIMULADO DE UN

CONVERTIDOR DC/DC DE TOPOLOGÍA BUCK-BOOST DE FASE NO MÍNIMA", Escuela Politécnica Nacional, Quito, 2019.

[35] Universidad Politécnica de Madrid, "Sistema Muestreados". [En línea]. Disponible en: http://www.ieef.upm.es/webantigua/spain/Asignaturas/Servos/practicas/prac10_sis_mestr.pdf.

[36] J. Aguilar y M. Cerrada, "Genetic Programming-Based Approach for System Identification", *Adv. Fuzzy Syst. Evol. Algorithms*, p. 7, mar. 2001.

[37] D. E. Seborg, T. F. Edgar, y D. A. Mellichamp, Eds., *Process dynamics and control*, 3rd ed. Hoboken, N.J.: Wiley, 2011.

[38] O. Camacho y C. A. Smith, "Sliding mode control: an approach to regulate nonlinear chemical processes", *ISA Trans.*, vol. 39, núm. 2, pp. 205–218, abr. 2000, doi: 10.1016/S0019-0578(99)00043-9.

[39] Michael Hawrylo, "ISA101 Human Machine Interfaces", oct-2015.

[40] Atmel, "Datasheet Arduino Mega 2560".

[41] C. J. Mejía Aguirre, "DISEÑO E IMPLEMENTACIÓN DE CUATRO ESQUEMAS DE CONTROL MODIFICADOS BASADOS EN EL PREDICTOR DE SMITH EN UNA TARJETA EMBEBIDA, APLICADOS A DOS MODELOS SIMULADOS QUE PRESENTAN RETARDO: UN TANQUE DE MEZCLADO Y UN REACTOR DE AGITACIÓN CONTINUA (CSTR)", Escuela Politécnica Nacional, Quito, 2019.

[42] M. Wright, "The interior- point revolution in optimization: history, recent developments, and lasting consequences", *Bulletin of the American Mathematical Society*, vol. 42, núm. 1, pp. 39–58, 09-2004.

[43] C. Lyzell, *Initialization methods for system identification*. Linköping: Department of Electrical Engineering, Linköping University, 2009.

6. ANEXOS

ANEXO A. Manual de usuario

ANEXO B. Reportes de la identificación de sistemas

ANEXO C. Resultados de las pruebas realizadas

ANEXO D. Resultados de la identificación por MatLab® System Identification Toolbox

ANEXO A

MANUAL DE USUARIO

A.1. INTRODUCCIÓN

El documento descrito a continuación tiene el propósito de guiar en la manipulación de la herramienta computacional System Identifier (SysID), desarrollada en MatLab R2019a, con su interfaz en App Designer 2019a y con Simulink.

A.2. PRESENTACIÓN DE LA INTERFAZ

La Figura A.1 muestra la interfaz gráfica desarrollada para la herramienta computacional SysID, en esta se encuentran los siguientes componentes:

1. PRESENTACIÓN: se inicializa por default en la herramienta, contiene la información general del proyecto de titulación como se muestra Figura A.1.
2. IDENTF. ACO: permite la identificación de sistemas y estimación de parámetros mediante el algoritmo de optimización por colonia de hormigas.
3. IDENTF. ACO-R: se accede a la identificación de sistemas y estimación de parámetros mediante el algoritmo de optimización colonia de hormigas para sistemas de dominio continuo.
4. IDENTF. PSO: despliega los recursos necesarios para la identificación de sistemas y estimación de parámetros por enjambre de partículas.

Figura A.6.1 Pantalla de presentación de la interfaz gráfica de usuario.

La Figura A.2 presenta el diseño tipo de interfaz gráfica para cada pestaña, su estructura se describe a continuación:

1. Botón- INFORMACIÓN: despliega una ventana adicional donde se muestra un resumen de la utilidad de la herramienta computacional. Se describe brevemente cada uno de los algoritmos implementados.
2. Botón- PREGUNTAS: despliega las respuestas a las preguntas más frecuentes en una ventana adicional.
3. Botón- CARGAR: el botón 3a. es para guardar los valores de referencia ingresados según la selección del MODELO DE REFERENCIA, a la vez que el botón 3b. es para simular archivos desde Simulink y cargar los valores de referencia (entrada, salida y tiempo de simulación) para la identificación del sistema y estimación de parámetros.
4. Botón- ENCONTRAR SISTEMA: inicia la búsqueda del mejor modelo a identificarse por medio del algoritmo de búsqueda que se señala en la pestaña.
5. Botón- GUARDAR RESULTADOS: hace una captura de los PARÁMETROS ENCONTRADOS, además guarda en. fig y en formato imagen .svg la gráfica de la identificación del sistema.
6. Botón- CONECTAR/ ADQUIRIR DATOS: el botón 6a. establece la comunicación serial con la tarjeta embebida. El botón 6b. inicia la adquisición de datos.
7. Selector- SELECCIONAR: permite escoger el tipo de sistema que se desea estimar sus parámetros característicos, esto cuando se conoce la respuesta del sistema.

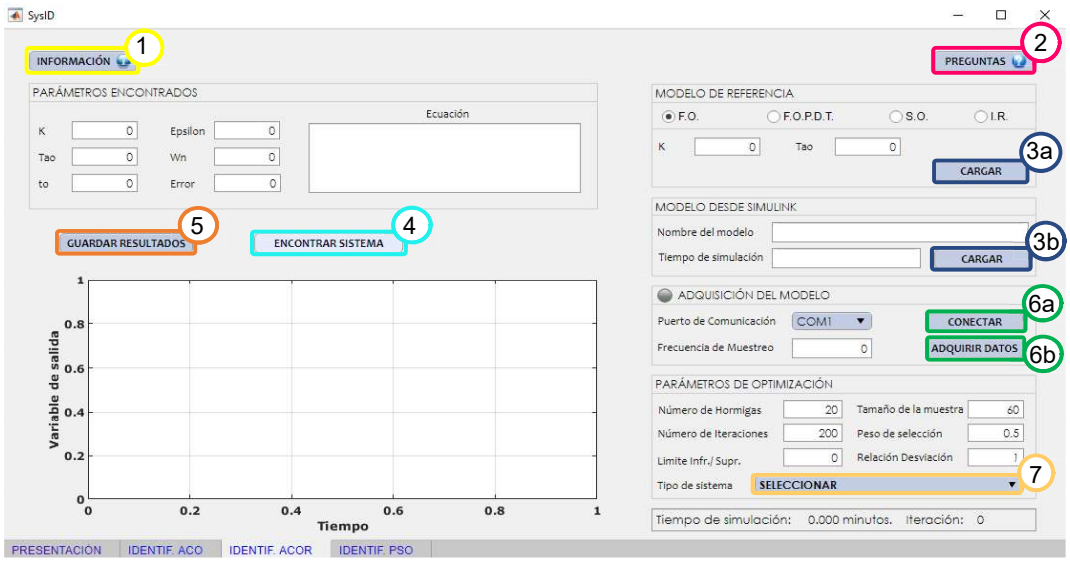


Figura A.2 Ventana tipo para los tres casos de algoritmos de optimización.

A.3. MANIPULACIÓN DE COMPONENTES

En la Figura A.3 se muestran los ambientes que contiene la interfaz gráfica, los mismo que se describen a continuación:

1. PARÁMETROS ENCONTRADOS: espacio de visualización de resultados a la estimación e identificación de sistemas, también se muestra el error y la función de transferencia a la identificación del modelo de sistema.

2. En globa las tres formas de adquisición de referencias para la identificación y estimación de sistemas. El MODELO DE REFERENCIA donde el usuario escoge el tipo de sistema que quiere identificar e ingresa los volares manualmente. El MODELO DESDE SIMULINK es para la identificación de sistemas desde archivos .slx. La ADQUISICIÓN DEL MODELO con ayuda de una tarjeta embebida.

3. VISUALIZACIÓN DE RESULTADOS: sirve tanto para la presentación del sistema referencia como para la visualización del del sistema identificado vs. el sistema referencia y para la visualización de la evolución del error por iteración.

4. PARÁMETROS DE OPTIMIZACIÓN: este espacio consta de las principales características de los algoritmos que pueden ser modificados según la necesidad del problema a identificarse. Los campos varían de acuerdo con el algoritmo a utilizarse.

5. En este espacio se muestra el tiempo de simulación y el número de iteraciones necesarias para la identificación del sistema.

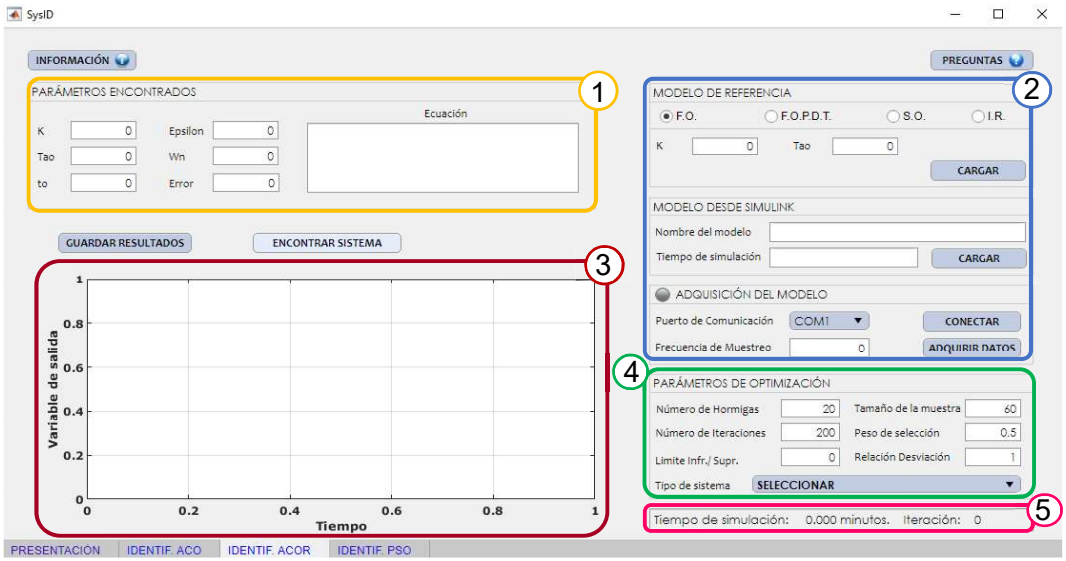


Figura A.3 Presentación de ambientes en la interfaz gráfica.

A.4. MANIPULACIÓN DE SYSID

El procedimiento de manipulación de SysID, en la identificación de sistemas cuando se ingresa los valores de referencia es el siguiente:

1. Como se muestra en la Figura A.4, en la sección **MODELO DE REFERENCIA** se escoge el tipo de sistema y se completa los valores. Se presiona el botón **CARGAR** y en la interfaz se muestra la función de transferencia y la representación gráfica del sistema.

Figura A.4 Ingreso de datos de referencia en la interfaz gráfica.

2. SysID automáticamente en **PARÁMETROS DE OPTIMIZACIÓN**, según el modelo de referencia que se haya seleccionado, selecciona el tipo de sistema a optimizar como la vez posee valores predefinidos para los parámetros del algoritmo. Si se desea aproximar a otro tipo de sistema se debe seleccionar "***Encontrar el mejor***" opción inmersa en el selector.

Figura A.5 Selección del sistema de modelo a identificar.

3. Una vez establecido el sistema de referencia, el modelo del sistema a cuál se desea aproximar y establecidos los parámetros adecuados para el algoritmo de optimización. Se procede a presionar en **ENCONTRAR SISTEMA**, el algoritmo comienza con la ejecución de sus rutinas y se despliega una barra de progreso que muestra el avance del algoritmo en encontrar el modelo más próximo al óptimo.

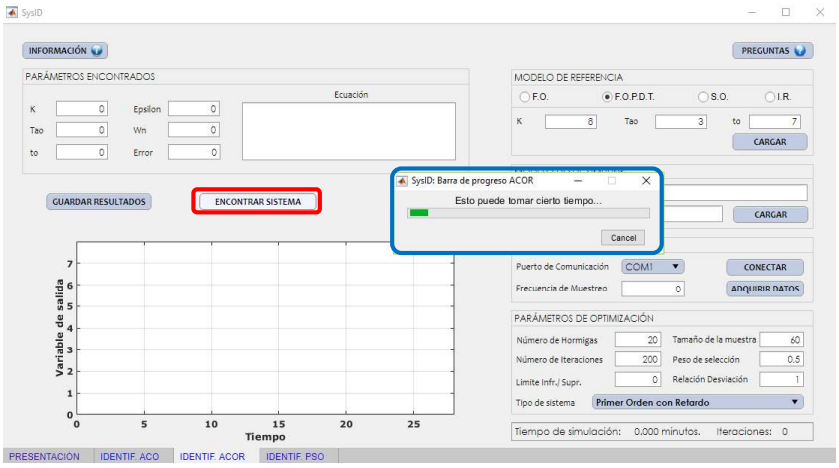


Figura A.6 Ejecución de la identificación de sistemas.

4. Una vez que la ejecución el algoritmo ha finalizado, se despliega en **PARÁMETROS ENCONTRADOS** los parámetros estimados, el ISE, la función de transferencia, las gráficas del modelo referencia vs. modelo identificado, el número de iteraciones necesario y el tiempo de ejecución del algoritmo. Al mismo tiempo se habilita el botón **GRÁFICA DEL ERROR** que al presionarlo se muestra el gráfico de la evolución del error por iteración como se muestra en la Figura A.8. Para volver a la gráfica de identificación se necesita presionar nuevamente en el botón **GRÁFICA DEL ERROR** para que se habilite el botón de **ENCONTRAR SISTEMA.**

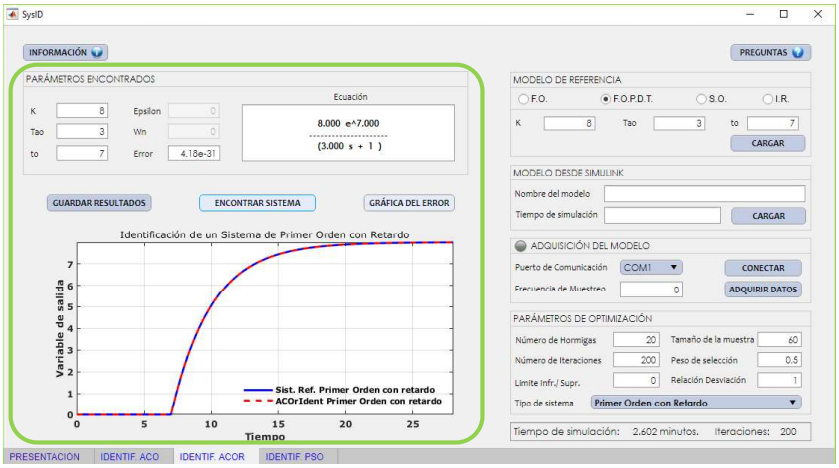


Figura A.7 Presentación de resultados del proceso de identificación de sistemas.

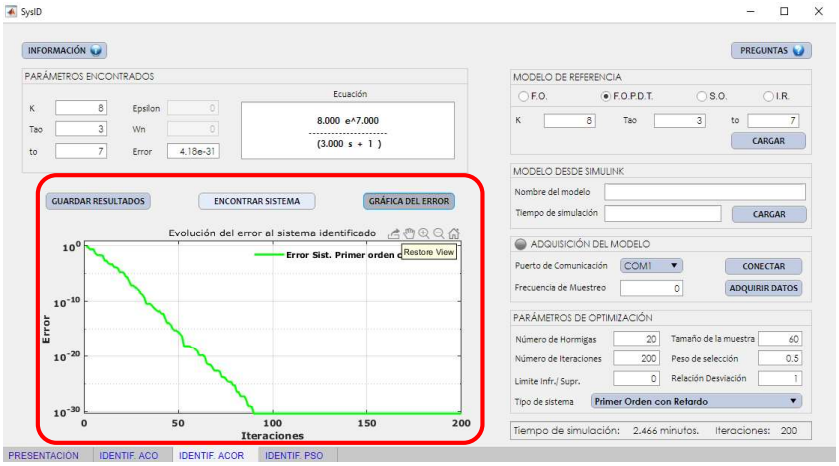


Figura A.8 Presentación de la gráfica del error por número de iteraciones.

5. Para guardar los resultados de la identificación se utiliza el botón **GUARDAR RESULTADOS**, esta función permite guardar una captura .svg de los resultados obtenidos del proceso de identificación.

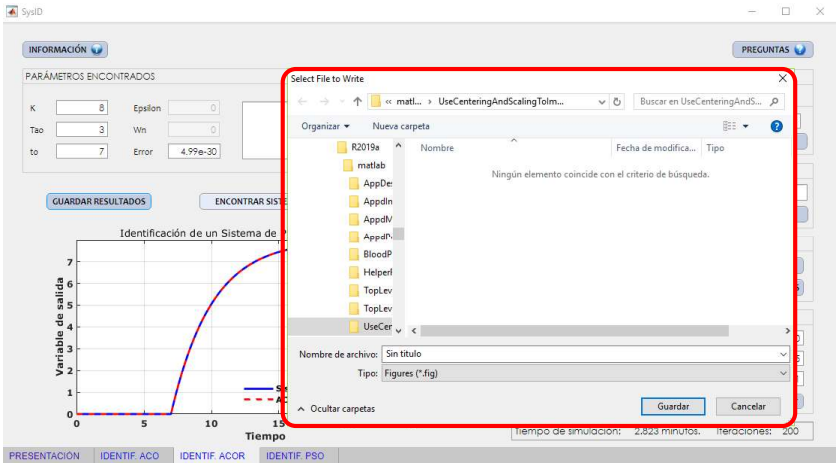


Figura A.9 Selección de la carpeta para guardar los archivos .fig y .svg.

La herramienta SysID también incorpora en su interfaz de usuario la opción de ingresar modelos de simulación desde el software Simulink ™. A continuación, se describe las consideraciones que deben tomarse al crear cada modelo de Simulink, y el procedimiento adecuado para el buen uso de este.

1. Cada archivo .slx debe cumplir específicamente con el nombre de las variables que se especifican en la Tabla A.1, esto asegura que SysID funcione adecuadamente, caso contrario saldrá un mensaje de error y no se efectuará la identificación del sistema.

Tabla A.6.1 Variables a especificarse en los archivos de Simulink.

Bloque de Simulink	Nombre en SysID	Función
Step	Step	Señal de entrada del sistema
Output1	ENTRADA_REF	Etiqueta de la variable que va al workspace de los datos de la señal de entrada del modelo.
Output2	TIEMPO_REF	Etiqueta de la variable que va al workspace del vector del tiempo de simulación del modelo.
Output3	SALIDA_REF	Etiqueta de la variable que va al workspace de los datos de la señal de salida del modelo.

De igual forma los modelos .slx deben ser de *Fixed Step Solver,* de preferencia el *ode 3 (Bogacki- Shampine)* u *ode4 (Runge- Kutta)*, la Figura A.10 muestra un modelo tipo de Simulink.

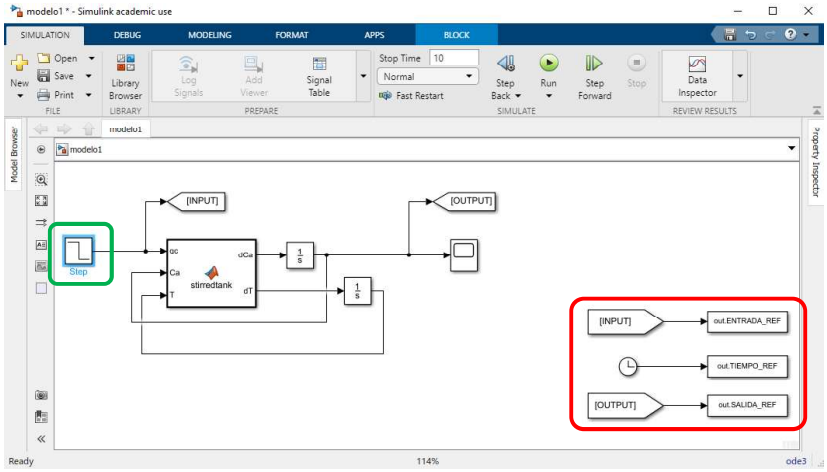


Figura A.10 Presentación de la estructura de un modelo de Simulink tipo.

2. En **MODELO DESDE SIMULINK** se ingresa el nombre del archivo de .slx y el tiempo de simulación. Se presiona el botón **CARGAR,** aparecerá un cuadro de dialogo de éxito si el archivo se ha cargado completamente y se mostrará el modelo referencia en la parte de gráficas.

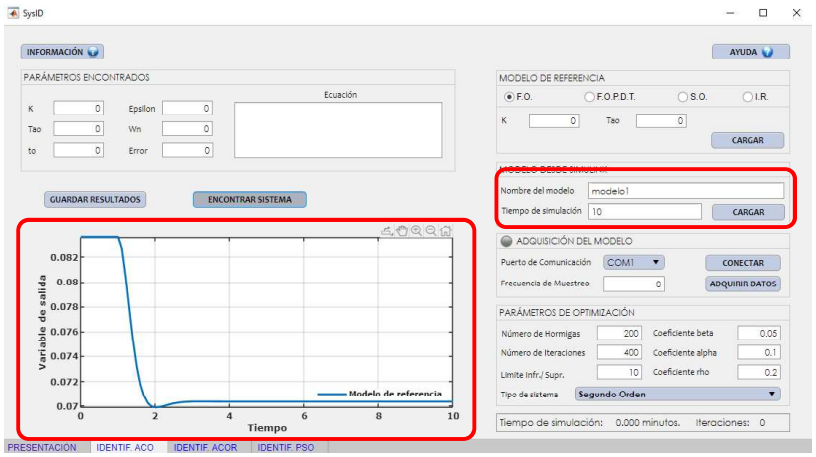


Figura A.11 Presentación del modelo referencia en SysID.

3. Una vez cargado el sistema referencia se puede escoger entre las distintas opciones de modelos de sistemas, siempre acorde a los modelos de sistemas al cual se desea aproximar la identificación o se puede utilizar la opción ***"Encontrar el mejor"***. Se presiona el botón **ENCONTRAR SISTEMA**, para identificar el modelo del sistema.

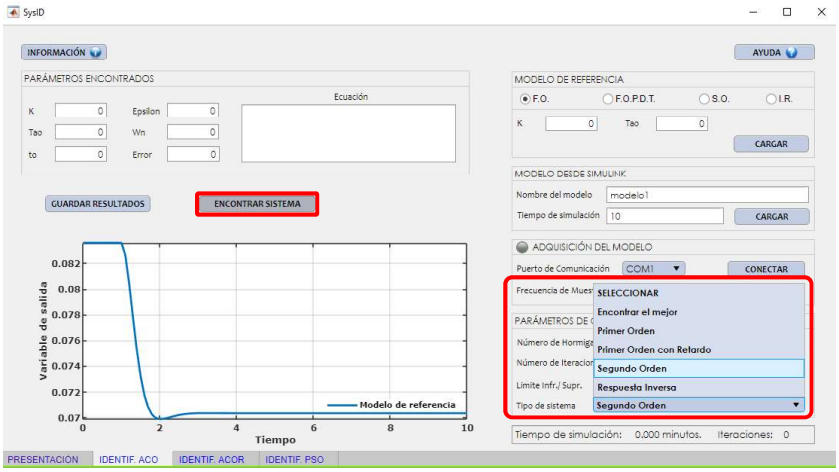


Figura A.11 Selección del modelo de sistema a identificar.

4. Al terminar la identificación y estimación de los parámetros de los modelos, el resultado se muestra igual que en el caso anterior. Asimismo, la opción de guardar está habilitada como en los casos anteriores.

En SysID también cuenta con una pequeña parte de adquisición de datos, el mismo que se usa de la siguiente manera.

1. Se configura los puertos de comunicación, se escoge el puerto adecuado, verificado del Administrador de dispositivos de Windows.
2. Se presiona el botón **CONECTAR**, para iniciar la comunicación serial entre el Arduino y la interfaz de usuario de SysID, se presiona nuevamente el botón para **DESCONECTAR** y deshabilitar la comunicación serial entre el Arduino y la interfaz de usuario de SysID, se borran los datos del puerto serial y se cierra el puerto serial, de esta forma se asegura que está listo para una nueva comunicación
3. Se presiona el botón **ENVIAR** para el enviar el tiempo de muestreo, este botón se debe aplastar seguidamente del botón anterior, siempre y cuando no haya una adquisición en ejecución.
4. Se adquiere los datos referencia los mismos que se muestran en la parte de gráfica de la herramienta computacional.
5. El botón **ACTUALIZAR** permite el reconocimiento de los puertos seriales conectados al equipo y su selección, como también al existir alguna falla de

comunicación reinicia la comunicación serial, al habilitar y deshabilitar el puerto serial seleccionado.

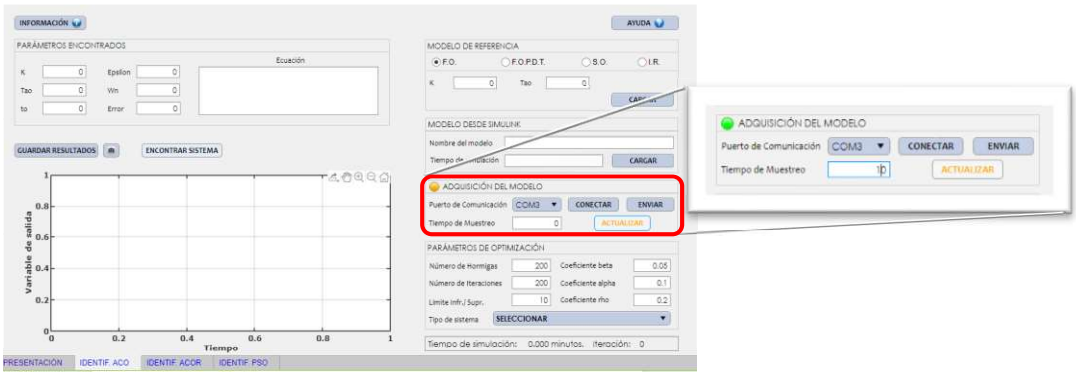


Figura A.12 Conexión de la comunicación serial.

A.5. AYUDA E INFORMACIÓN

La herramienta computacional posee varias ventanas emergentes de ayuda e información para guiar al usuario, estos son:

1. Al ingresar valores incorrectos en **MODELO DE REFERENCIA** se despliega un cuadro de aviso que indica la falta de valores de referencia o que los caracteres ingresados no son los permitidos.

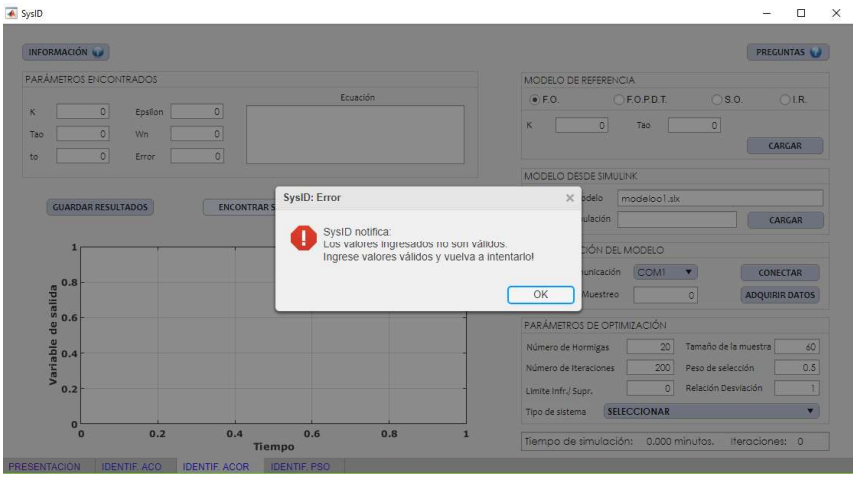


Figura A.13 Cuadro de dialogo del error.

2. En **MODELO DESDE SIMULINK**, si no se encuentra el archivo .slx se muestra un cuadro de dialogo donde señala que el nombre del archivo es incorrecto, si no se especifica el tiempo de simulación también se presenta un cuadro de aviso con el propósito de que el usuario rectifique y vuelva a cargar el archivo. De lo contrario, si la carga es correcta se muestra la ventada de dialogo de éxito.

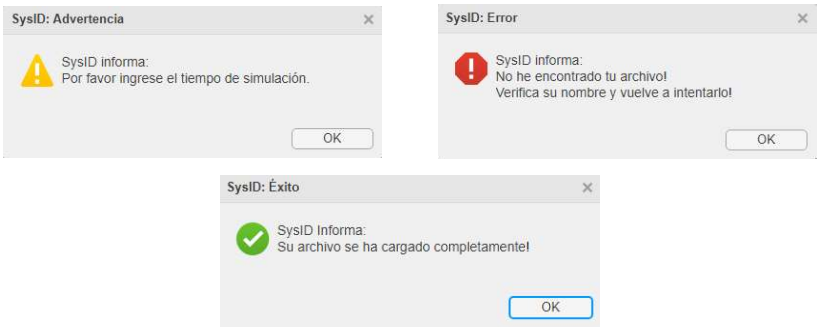


Figura A.14 Cuadros de dialogo de Advertencia y Éxito.

3. Al guardar el modelo identificado se muestra una ventana informativa que indica la ruta en donde se ha almacenado el archivo .fig y la captura .svg.

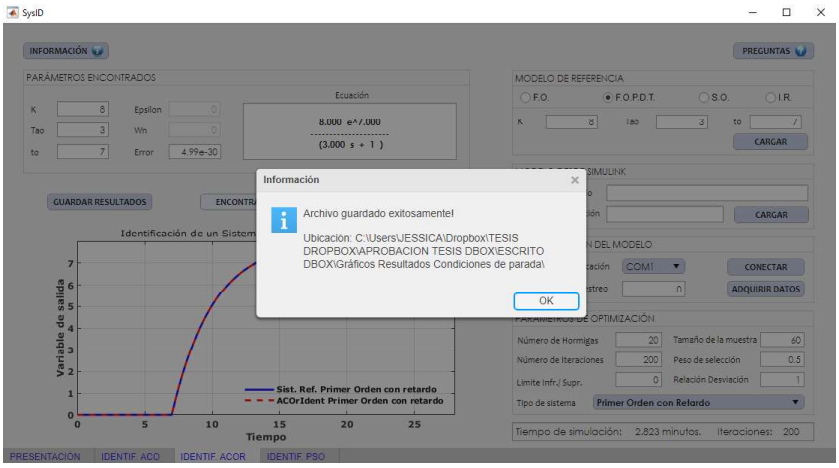


Figura A.15 Cuadro de dialogo de la ubicación del archivo.

ANEXO B

REPORTES DE LA IDENTIFICACIÓN

B.1. SISTEMA TIPO DE PRIMER ORDEN

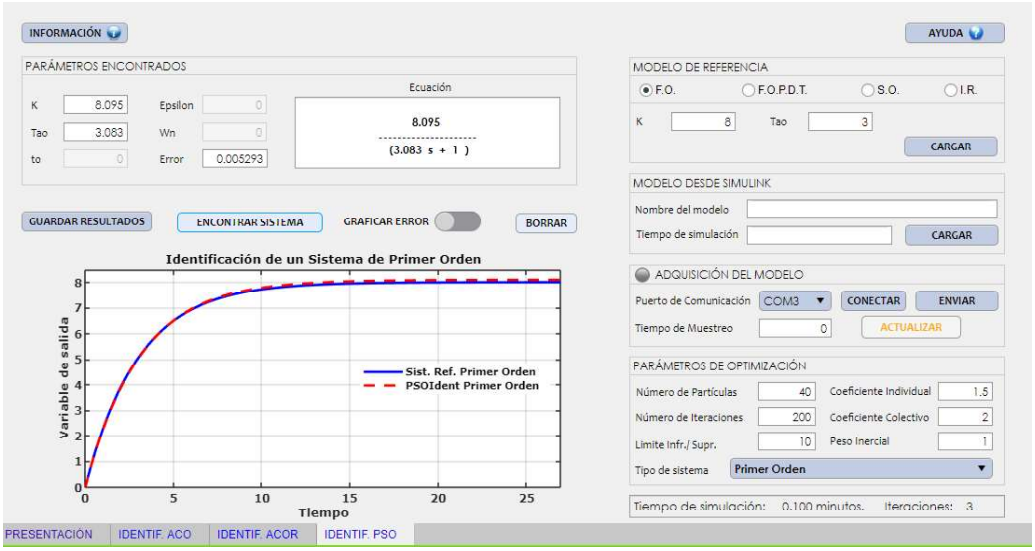


Figura B.1. Captura de la identificación del sistema tipo de primer orden.

B.2. SISTEMA TIPO DE PRIMER ORDEN CON RETARDO

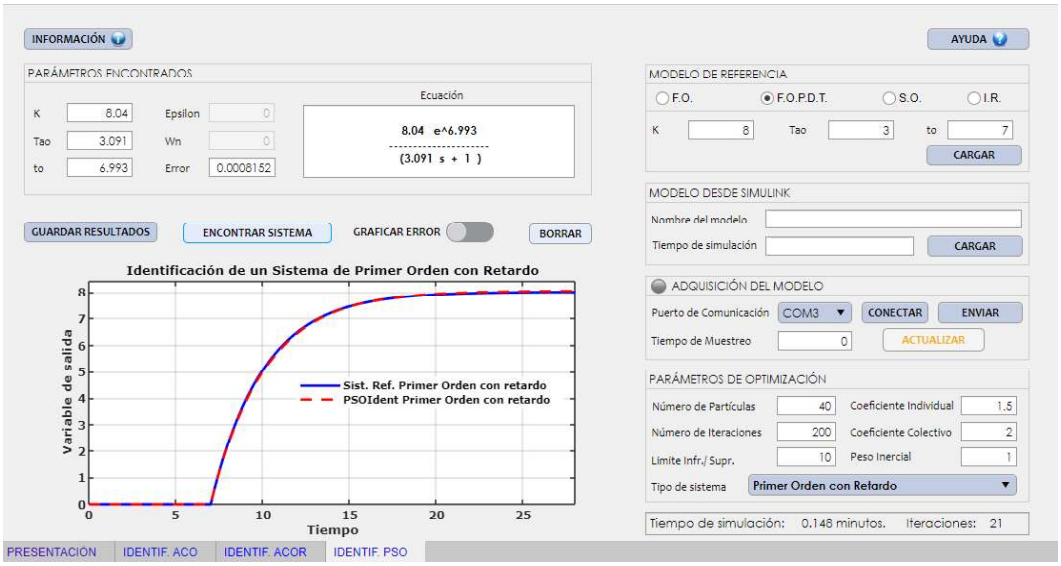


Figura B.2. Captura de la identificación del sistema tipo de primer orden con retardo

B.3. SISTEMA TIPO DE SEGUNDO ORDEN

Figura B.3. Captura de la identificación del sistema tipo de segundo orden.

B.4. SISTEMA TIPO DE RESPUESTA INVERSA

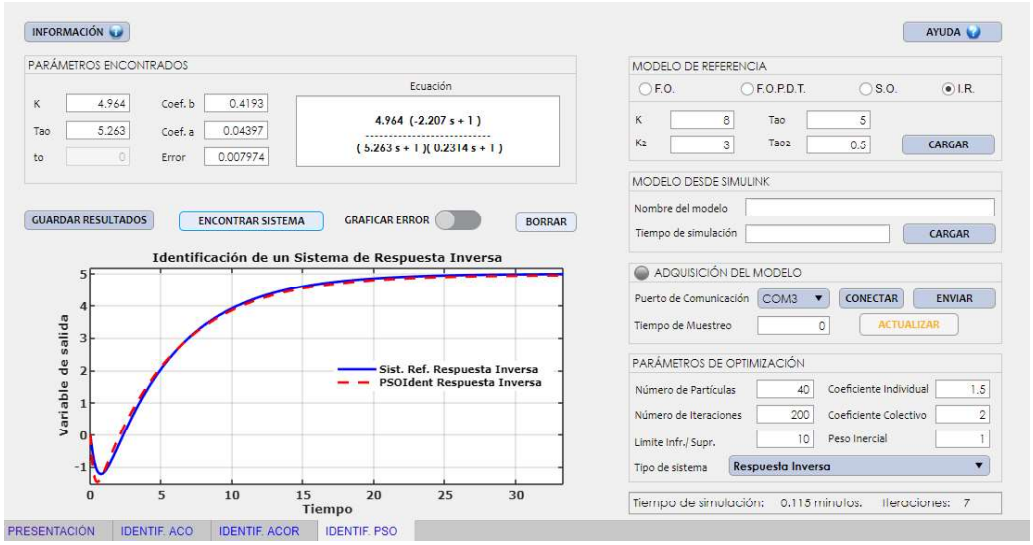


Figura B.4. Captura de la identificación del sistema tipo de respuesta inversa.

B.5. SISTEMA REAL MOTOR DC

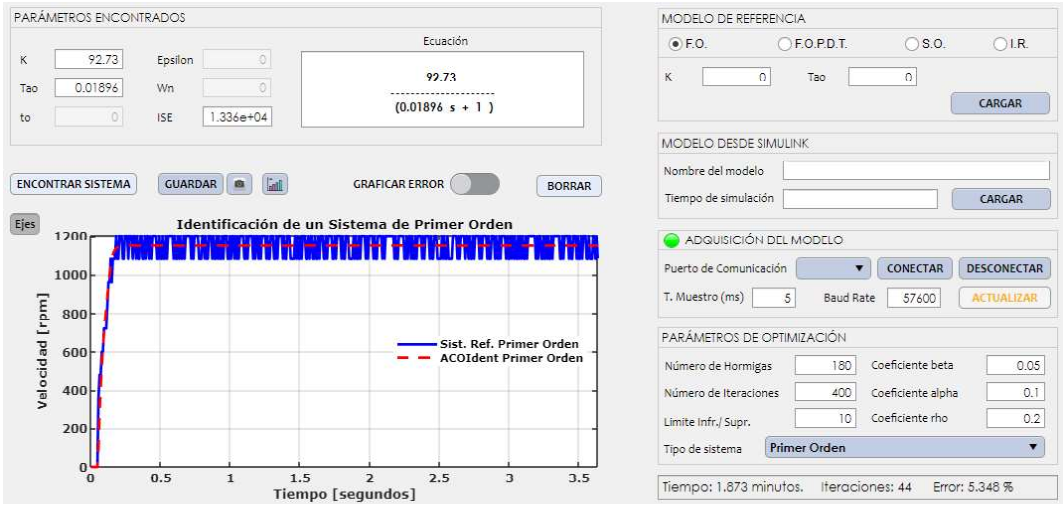


Figura B.5. Captura de la identificación del motor DC por el algoritmo ACO.

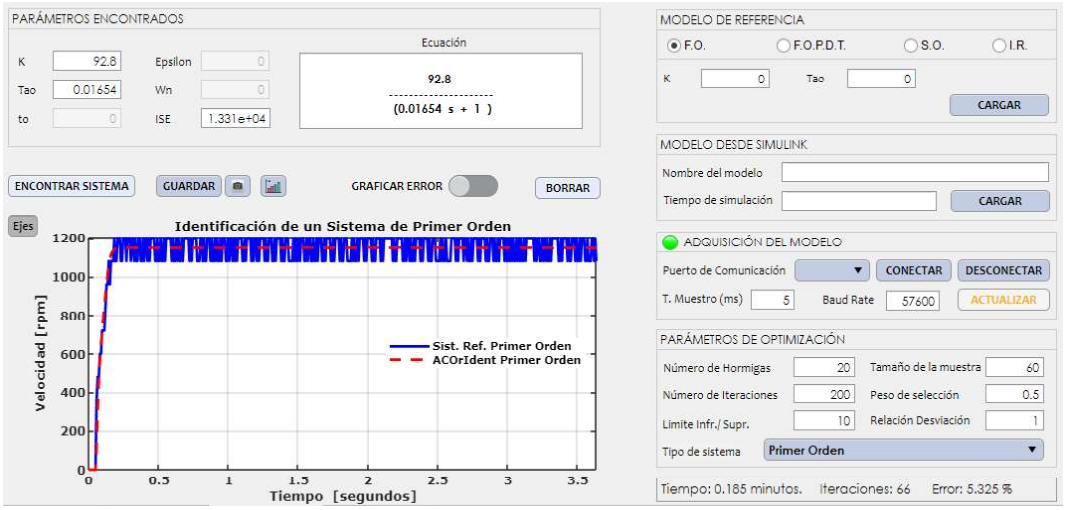


Figura B.6. Captura de la identificación del motor DC por el algoritmo ACOR.

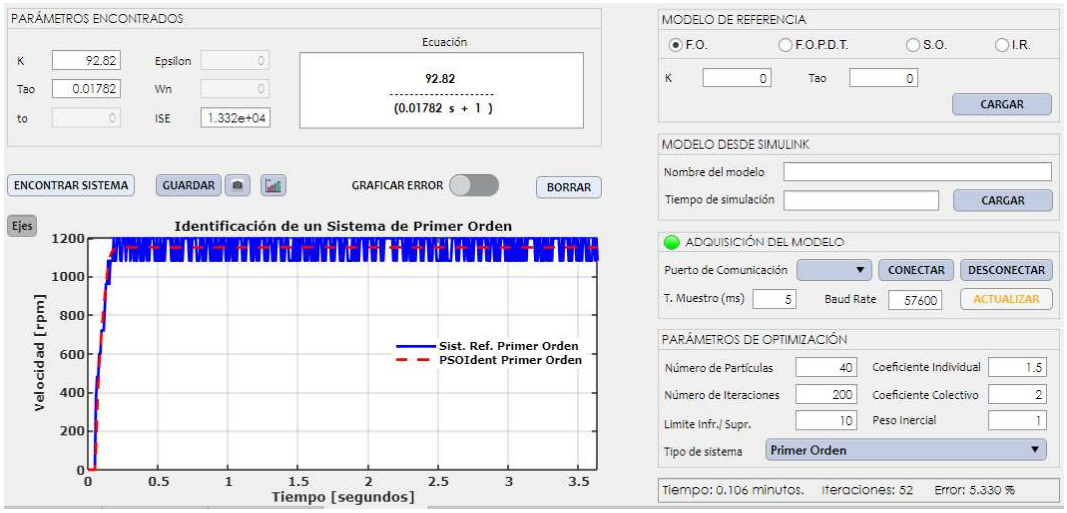


Figura B.7. Captura de la identificación del motor DC por el algoritmo PSO.

B.6. SISTEMAS NO LINEALES DESDE SIMULINK

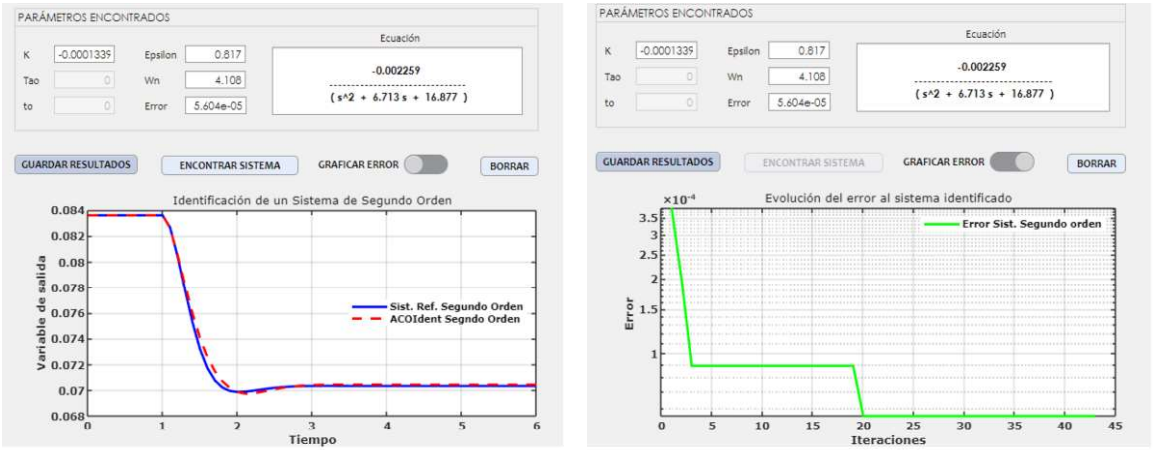


Figura B.10. Captura de la identificación del sistema no lineal por el algoritmo ACO.

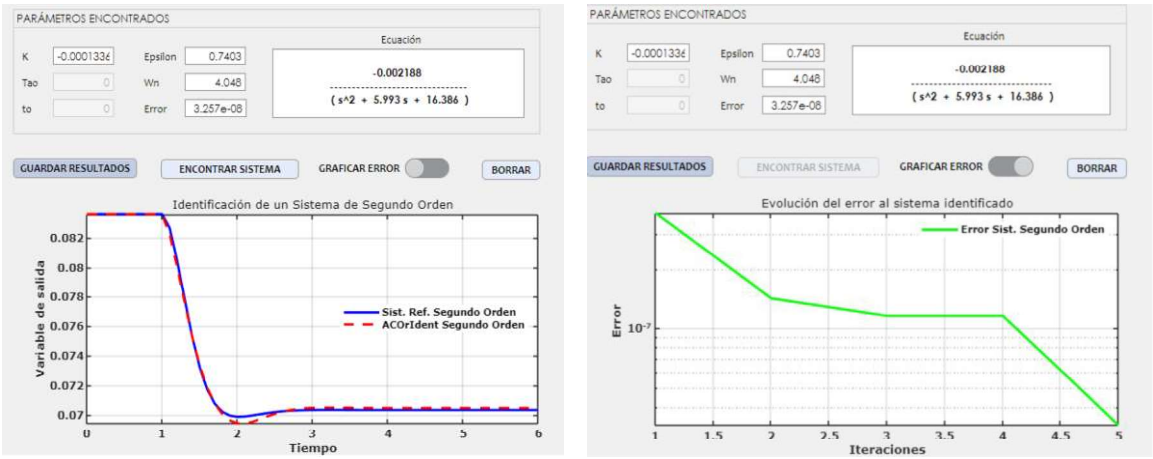


Figura B.11. Captura de la identificación del sistema no lineal por el algoritmo ACO-R.

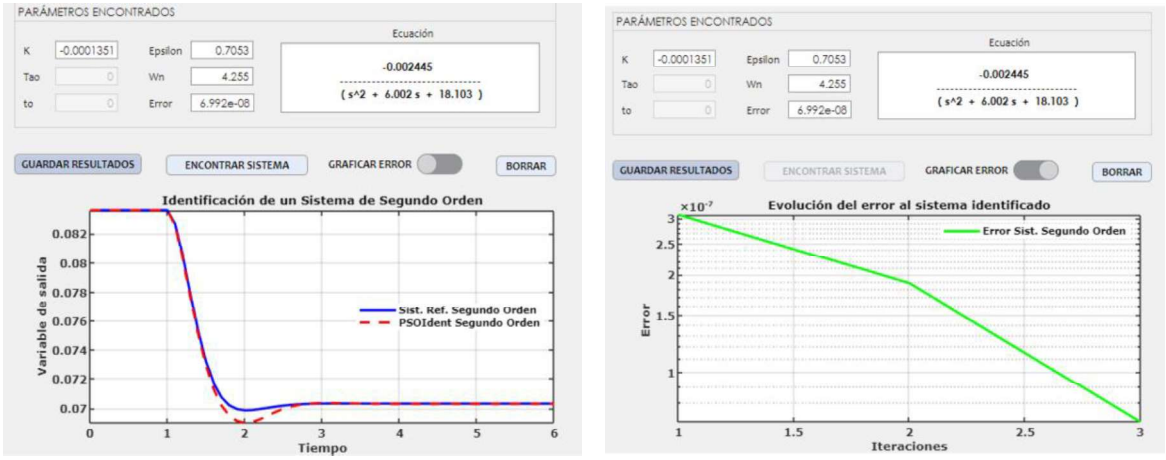


Figura B.12. Captura de la identificación del sistema no lineal por el algoritmo PSO.

ANEXO C

RESULTADOS DE LAS PRUEBAS REALIZADAS

- **IDENTIFICACIÓN DEL SISTEMA TIPO DE PRIMER ORDEN**

Tabla C.1 Estimación de parámetros al modelo tipo primer orden mediante ACO.

	CONDICIONES DE PARADA								
	Índice de error < 0.01					Máximo de 200 iteraciones			
	K	Tao	ISE	Tiempo [min]	Nro. Iter.	K	Tao	ISE	Tiempo [min]
1	7.939	2.908	0.00233	4.064	2	8.017	3.081	0.000919	7.249
2	7.9	2.972	0.00729	0.204	3	8	2.994	7.901e-6	5.219
3	7.959	3.124	0.00657	0.115	3	8.024	3.042	0.00037	4.636
4	7.899	2.787	0.00823	0.148	2	8.045	3.107	0.001783	5.270
5	8.026	3.039	0.00042	0.168	4	7.963	2.923	0.001079	4.793
6	7.988	3.147	0.00503	0.117	2	8.017	3.081	0.000919	5.371
7	8.083	3.068	0.00407	0.112	4	8	2.994	7.901e-6	4.529
8	8.011	3.069	0.00069	0.183	3	8.024	3.042	0.000371	4.380
9	7.984	2.886	0.00203	0.117	2	8.017	3.081	0.000919	6.692
10	7.987	3.139	0.00463	0.108	2	8	2.994	7.901e-6	5.264
Promedio			0.00413	0.5336	2.7	**Promedio**		0.000638	5.3403

Tabla C.2 Estimación de parámetros al modelo tipo de primer orden mediante ACO- R.

	CONDICIONES DE PARADA								
	Índice de error < 0.01					Máximo de 200 iteraciones			
	K	Tao	ISE	Tiempo [min]	Nro. Iter.	K	Tao	ISE	Tiempo [min]
1	7.951	2.916	0.00159	0.189	2	8	3	1.11e-30	1.308
2	7.967	3.043	0.00189	0.142	4	8	3	1.11e-30	1.209
3	8.118	3.153	0.00806	0.118	3	8	3	1.11e-30	1.207
4	8.014	3.001	0.00017	0.117	3	8	3	1.11e-30	1.217
5	7.935	3.069	0.00632	0.114	3	8	3	1.11e-30	1.302
6	8	2.808	0.00754	0.121	3	8	3	1.11e-30	1.235
7	8.052	3.056	0.00159	0.121	4	8	3	1.11e-30	1.211
8	7.955	2.961	0.00119	0.144	5	8	3	1.11e-30	1.360
9	7.952	3.144	0.00886	0.111	3	8	3	1.11e-30	1.241
10	7.982	2.998	0.00026	0.117	4	8	3	1.11e-30	1.236
Promedio			0.003747	0.1294	3.4	**Promedio**		1.11e-30	1.2526

Tabla C.3 Estimación de parámetros al modelo tipo de primer orden mediante PSO.

	CONDICIONES DE PARADA								
	Índice de error < 0.01					Máximo de 200 iteraciones			
	K	Tao	ISE	Tiempo [min]	Nro. Iter.	K	Tao	ISE	Tiempo [min]
1	7.928	3.03	0.00539	0.110	3	8	3	1.11e-30	0.898
2	8.059	2.95	0.00476	0.113	4	8	3	1.11e-30	0.870
3	7.964	2.891	0.00174	0.103	2	8	3	1.11e-30	0.870
4	7.887	2.787	0.00924	0.101	2	8	3	1.11e-30	0.867
5	7.93	3.085	0.00808	0.106	3	8	3	1.11e-30	0.867
6	8.1	3.148	0.00599	0.119	6	8	3	1.11e-30	0.867
7	7.965	2.988	0.00088	0.110	2	8	3	1.11e-30	0.866
8	8.103	3.234	0.00858	0.138	5	8	3	1.11e-30	0.868
9	7.956	2.801	0.00576	0.102	2	8	3	1.11e-30	0.885
10	7.95	3.077	0.00489	0.133	4	8	3	1.11e-30	0.899
Promedio			0.0055	0.1135	3.3	**Promedio**		1.11e-30	0.8757

- **IDENTIFICACIÓN DEL SISTEMA TIPO DE PRIMER ORDEN CON RETARDO**

Tabla C.4 Estimación de parámetros al modelo tipo de primer orden con retardo mediante ACO.

	CONDICIONES DE PARADA										
	Índice de error < 0.01						Máximo de 200 iteraciones				
	K	Tao	T_0	ISE	Tiempo [min]	Número iteración	K	Tao	T_0	ISE	Tiempo [min]
1	8.064	3.382	6.78	0.0093	9.795	230	8.01	2.92	7.16	0.0063	6.733
2	8.141	3.244	6.804	0.0124	7.126	230	7.83	2.79	7.16	0.0152	6.625
3	7.858	2.735	7.07	0.0073	2.333	230	8.14	3.43	6.92	0.0121	6.525
4	7.942	3.395	6.707	0.0231	7.226	230	8.00	3.28	6.81	0.0078	6.464
5	8.06	2.937	7.042	0.0101	7.259	230	8.03	3.49	6.77	0.0169	6.684
6	8.064	3.382	6.78	0.0093	13.10	230	7.93	2.87	7.05	0.0016	6.690
7	7.858	2.735	7.07	0.0073	7.314	230	8.05	3.23	6.93	0.0032	6.711
8	7.942	3.395	6.707	0.0231	7.143	230	7.81	2.92	7.01	0.0150	6.973
9	8.141	3.244	6.804	0.0124	7.388	230	7.96	2.90	7.08	0.0014	6.651
10	8.116	2.937	7.042	0.0101	7.464	230	7.82	2.65	7.15	0.0126	6.753
Promedio				0.0124	7.6148	230	**Promedio**			0.0092	6.681

Tabla C.5 Estimación de parámetros al modelo tipo de primer orden con retardo mediante ACO- R.

	CONDICIONES DE PARADA										
	Índice de error < 0.01%						Máximo de 200 iteraciones				
	K	Tao	T_0	ISE	Tiempo [min]	Nro. Iter.	K	Tao	T_0	ISE	Tiempo [min]
1	7.962	3.055	6.985	0.0019	0.164	7	8	3	7	4.18e-31	2.055
2	8.064	3.039	6.968	0.0022	0.140	5	8	3	7	4.18e-31	2.028
3	8.03	3.156	6.979	0.0022	0.220	9	8	3	7	4.99e-31	2.048
4	8.103	3.271	6.946	0.0052	0.220	13	8	3	7	4.18e-31	2.116
5	7.973	3.232	6.802	0.0087	0.161	7	8	3	7	4.18e-31	2.012
6	8.096	3.024	7.085	0.0057	0.210	8	8	3	7	4.18e-31	2.011
7	8.024	3.076	7.073	0.0042	0.229	10	8	3	7	4.99e-30	2.055
8	8.112	3.362	6.852	0.0071	0.278	17	8	3	7	4.18e-31	2.013
9	8.13	3.13	6.882	0.0098	0.165	6	8	3	7	4.99e-30	2.026
10	8.073	3.376	6.801	0.0083	0.202	11	8	3	7	4.18e-31	2.031
Promedio				0.00553	0.1989	9.3	**Promedio**			1.34E-30	2.0395

Tabla C.6 Estimación de parámetros al modelo tipo de primer orden con retardo mediante PSO.

	CONDICIONES DE PARADA										
	Índice de error < 0.01%						Máximo de 200 iteraciones				
	K	Tao	T_0	ISE	Tiempo [min]	Nro. iter.	K	Tao	T_0	ISE	Tiempo [min]
1	7.972	3.034	6.89	0.0034	0.179	12	8	3	7	4.18e-31	1.443
2	7.909	2.967	7.04	0.0049	0.159	9	8	3	7	4.99e-30	1.411
3	8.024	2.952	7.052	0.0009	0.145	4	8	3	7	4.18e-31	1.468
4	7.975	2.866	7.1	0.0018	0.214	6	8	3	7	4.99e-30	1.624
5	7.979	3.281	6.869	0.0092	0.152	7	8	3	7	4.18e-31	1.441
6	8.051	3.225	6.885	0.0030	0.217	9	8	3	7	4.18e-31	1.523
7	8.004	3.054	6.95	0.0005	0.151	8	8	3	7	4.18e-31	1.404
8	7.87	2.627	7.118	0.0097	0.236	9	8	3	7	4.18e-31	1.406
9	7.965	2.849	7.026	0.0019	0.139	6	8	3	7	4.99e-30	1.405
10	8.147	3.187	6.971	0.0069	0.230	9	8	3	7	4.99e-30	1.484
Promedio				0.00422	0.1822	7.9	**Promedio**			2.25E-30	1.4609

- **IDENTIFICACIÓN DEL SISTEMA TIPO DE SEGUNDO ORDEN**

Tabla C.7 Estimación de parámetros al modelo tipo de segundo orden mediante ACO.

	CONDICIONES DE PARADA										
	Índice de error < 0.01%						**Máximo de 200 iteraciones**				
	K	ξ	ω_n	**ISE**	**Tiempo [min]**	**Número iteración**	**K**	ξ	ω_n	**ISE**	**Tiempo [min]**
1	8.673	0.27	1.00	0.504	9.443	230	8.67	0.28	1.00	0.5036	8.570
2	7.827	0.211	0.96	0.099	6.339	230	7.83	0.21	0.96	0.0989	5.368
3	7.827	0.211	0.96	0.098	5.697	230	7.83	0.21	0.96	0.0989	9.849
4	8.346	0.218	0.93	0.329	5.634	230	8.29	0.37	1.25	0.7165	5.517
5	7.724	0.240	1.06	0.210	6.352	230	7.71	0.35	1.11	0.5362	4.927
6	8.039	0.149	1.00	0.167	5.956	230	8.67	0.28	1.00	0.5036	5.574
7	7.707	0.348	1.11	0.536	5.722	231	7.83	0.21	0.96	0.0989	4.949
8	7.731	0.272	1.07	0.274	5.602	231	8.35	0.22	0.93	0.3292	4.871
9	8.199	0.133	1.07	0.741	5.620	230	7.72	0.24	1.06	0.2104	4.905
10	7.816	0.361	1.11	0.499	5.566	230	8.039	0.15	1.01	0.1674	4.927
Promedio				0.3457	6.1931	230.2	**Promedio**			0.3264	5.9457

Tabla C.8 Estimación de parámetros al modelo tipo segundo orden mediante ACO- R.

	CONDICIONES DE PARADA										
	Índice de error < 0.01%						**Máximo de 200 iteraciones**				
	K	ξ	ω_n	**ISE**	**Tiempo [min]**	**Nro. iter.**	**K**	ξ	ω_n	**ISE**	**Tiempo [min]**
1	8.14	0.19	1.01	0.008	1.190	146	8	0.2	1	5.497e-30	1.58
2	8.11	0.21	1.01	0.005	0.317	33	8	0.2	1	5.497e-30	1.32
3	8.08	0.208	1.01	0.007	0.293	22	8	0.2	1	5.497e-30	1.45
4	8.08	0.214	1.01	0.008	0.944	72	8	0.2	1	5.497e-30	1.358
5	8.07	0.202	1.01	0.002	0.289	30	8	0.2	1	5.497e-30	1.564
6	8.17	0.209	1.013	0.009	0.908	133	8	0.2	1	5.497e-30	1.322
7	8.04	0.197	0.987	0.007	0.402	37	8	0.2	1	5.497e-30	1.285
8	7.95	0.211	1.00	0.008	0.336	34	8	0.2	1	5.497e-30	1.336
9	8.00	0.207	1.015	0.007	0.574	67	8	0.2	1	5.497e-30	1.319
10	8.04	0.187	0.996	0.009	0.441	46	8	0.2	1	5.497e-30	1.307
Promedio				0.007	0.5694	62	**Promedio**			5.50E-30	1.3841

Tabla C.9 Estimación de parámetros al modelo tipo de segundo orden mediante PSO.

	CONDICIONES DE PARADA										
	Índice de error < 0.01%						Máximo de 200 iteraciones				
	K	ξ	ω_n	ISE	Tiempo [min]	Nro. Iter.	K	ξ	ω_n	ISE	Tiempo [min]
1	8.032	0.207	0.995	0.0035	0.362	59	8	0.2	1	5.497e-30	1.062
2	8.011	0.211	0.995	0.0056	0.224	23	8	0.2	1	5.497e-30	0.931
3	7.926	0.201	0.998	0.0060	0.339	47	8	0.2	1	5.497e-30	1.001
4	8.033	0.208	0.989	0.0082	0.288	41	8	0.2	1	5.497e-30	1.317
5	8.003	0.199	0.989	0.0045	0.262	36	8	0.2	1	5.497e-30	1.009
6	8.051	0.204	0.998	0.0029	0.257	36	8	0.2	1	5.497e-30	0.962
7	7.947	0.204	1.005	0.0046	0.317	51	8	0.2	1	5.497e-30	1.377
8	8.017	0.212	1.007	0.0054	0.224	31	8	0.2	1	5.497e-30	0.933
9	7.943	0.189	1.007	0.0092	0.269	42	8	0.2	1	5.497e-30	0.910
10	7.935	0.202	0.997	0.0052	0.355	55	8	0.2	1	5.497e-30	0.968
Promedio				0.00551	0.2897	42.1	**Promedio**			5.50E-30	1.05

- **IDENTIFICACIÓN DEL SISTEMA TIPO DE RESPUESTA INVERSA**

Tabla C.10 Estimación de parámetros al modelo tipo de respuesta inversa mediante ACO.

	CONDICIONES DE PARADA										
	Índice de error < 0.01%						Máximo de 200 iteraciones				
	Tao	b	a	ISE	Tiempo [min]	Nro. Iter.	Tao	b	a	ISE	Tiempo [min]
1	0.77	2.68	6.32	0.0043	7.23	26	0.63	3.33	7.82	0.0018	17.42
2	0.65	2.89	7.74	0.0055	0.465	3	0.62	3.33	7.82	0.0014	17.972
3	0.58	3.83	7.93	0.0033	0.397	3	0.625	3.33	7.82	0.0013	23.41
4	0.78	2.44	5.96	0.0058	7.601	84	0.724	2.73	6.75	0.0035	17.57
5	0.63	3.33	7.82	0.0014	7.826	81	0.503	4.4	9.67	0.0005	17.418
6	5.18	0.42	0.13	0.0093	1.144	12	0.62	3.33	7.82	0.0014	18.81
7	0.66	2.71	7.64	0.0079	0.901	10	0.72	2.73	6.78	0.0035	16.02
8	0.65	2.89	7.74	0.0055	0.270	3	0.50	4.4	9.67	0.0005	17.01
9	0.57	3.83	7.93	0.0033	0.353	3	0.64	3.39	7.55	0.0014	15.86
10	0.77	2.68	6.32	0.0043	3.234	26	0.58	3.75	7.76	0.0035	16.06
Promedio				0.00506	2.9421	25.1	**Promedio**			0.00188	17.755

Parámetro KRI = 4.964 para toda la identificación.

Tabla C.11 Estimación de parámetros al modelo tipo de respuesta inversa mediante ACO- R.

	CONDICIONES DE PARADA										
	Índice de error < 0.01%						Máximo de 200 iteraciones				
	Tao	b	a	ISE	Tiempo [min]	Nro. Iter.	Tao	b	a	ISE	Tiempo [min]
1	0.578	3.90	8.05	0.00175	0.213	8	0.69	3.08	6.82	0.00159	3.169
2	0.833	2.47	5.44	0.00395	0.341	14	0.56	3.90	8.64	0.00053	3.169
3	0.876	1.99	5.27	0.00987	0.325	14	0.66	3.25	7.19	0.00123	3.149
4	0.615	3.13	7.64	0.00613	0.140	2	0.69	3.05	6.78	0.00164	3.246
5	0.922	2.46	4.59	0.00951	0.136	3	0.66	3.26	7.24	0.00119	3.185
6	1.047	2016	4.16	0.00899	0.180	5	0.59	3.65	8.09	0.00068	3.235
7	1.113	1.74	3.99	0.00976	0.232	9	0.59	3.67	8.11	0.00068	3.179
8	0.576	3.16	8.65	0.00750	0.215	7	0.64	3.35	7.43	0.00105	3.130
9	0.423	5.27	11	0.00592	0.153	3	0.76	2.77	6.09	0.00257	3.451
10	0.904	2.15	5.35	0.00804	0.256	8	0.76	2.74	6.09	0.00259	3.342
Promedio				0.007142	0.2191	7.3	**Promedio**			0.001375	3.2255

Parámetro KRI = 4.964 para toda la identificación.

Tabla C.12 Estimación de parámetros al modelo tipo de respuesta inversa mediante PSO.

	CONDICIONES DE PARADA										
	Índice de error < 0.01%						Máximo de 200 iteraciones				
	Tao	a	b	ISE	Tiempo [min]	Nro. Iter.	Tao	b	a	ISE	Tiempo [min]
1	0.941	2.05	4.87	0.00627	0.324	9	0.528	4.163	9.22	0.00047	2.122
2	0.804	2.71	5.88	0.00495	0.241	7	0.528	4.166	9.226	0.00048	2.343
3	0.646	3.83	7.38	0.00824	0.177	5	0.527	4.167	9.229	0.00048	2.108
4	1.01	1.89	4.54	0.00786	0.196	4	0.527	4.168	9.23	0.00047	2.120
5	0.812	2.27	5.65	0.00740	0.135	3	0.528	4.171	9.237	0.00048	2.118
6	0.539	3.47	9.00	0.00776	0.131	3	0.529	4.157	9.206	0.00048	2.179
7	0.598	3.09	8.02	0.00777	0.162	3	0.528	4.169	9.234	0.00048	4.407
8	0.771	2.95	5.84	0.00459	0.215	3	0.529	4.158	9.211	0.00048	2.124
9	0.681	3.02	6.54	0.00643	0.220	9	4.872	0.452	0.108	0.00048	2.144
10	0.772	3.04	5.98	0.00669	0.170	5	0.528	4.137	9.228	0.00047	2.115
Promedio				0.00679	0.1971	5.1	**Promedio**			0.00048	2.378

Parámetro KRI = 4.964 para toda la identificación.

ANEXO D

RESULTADOS DE LA IDENTIFICACIÓN POR MATLAB®

- **MODELO DE PRIMER ORDEN DEL SISTEMA MOTOR DC**

```
Transfer Function Identification
Estimation data: Time domain data motor1
Data has 1 outputs, 1 inputs and 460 samples.
Number of poles: 1, Number of zeros: 0
Initialization Method: "iv"


Estimation Progress
Estimating delays as multiples of sample time ...
done.
Initializing model parameters...
Initializing using 'iv' method...
done.

Initialization complete.

Algorithm: Interior-Point
-----------------------------------------------------------------
                              Norm of     First-order
Iteration     Cost               step        optimality
-----------------------------------------------------------------
    0         3864.56           -              -
-----------------------------------------------------------------
Estimating parameter covariance...
done.
Result
Termination condition: Change in parameters was less than the specified tolerance..
Number of iterations: 1, Number of function evaluations: 13

Status: Estimated using TFEST
Fit to estimation data: 63.48, FPE: 3915.3
```

Figura D.1 Reporte de la identificación por System Identification Toolbox de MatLab®.

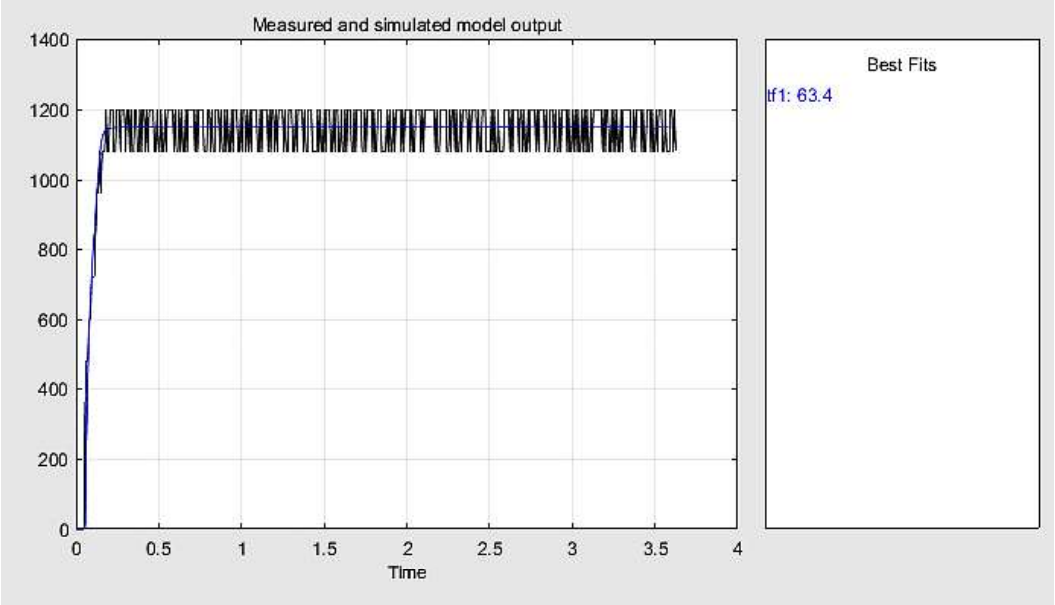


Figura D.2 Identificación de un sistema de primer orden.

```
  From input "Voltaje" to output "Velocidad":
     7011 (+/- 828)
  ---------------------
  s + 75.55 (+/- 8.942)

Name: tf1
Continuous-time identified transfer function.

Parameterization:
   Number of poles: 1   Number of zeros: 0
   Number of free coefficients: 2
   Use "tfdata", "getpvec", "getcov" for parameters and their uncertainties.

Status:
Termination condition: Change in parameters was less than the specified tolerance..
Number of iterations: 1, Number of function evaluations: 13

Estimated using TFEST on time domain data "motor1".
Fit to estimation data: 63.4% (stability enforced)
FPE: 3915, MSE: 3865
More information in model's "Report" property.
```

Figura D.3 Función de transferencia y datos de un sistema de primer orden identificado.

- **MODELO DE SEGUNDO ORDEN DEL SISTEMA MOTOR DC**

```
Transfer Function Identification
Estimation data: Time domain data motor
Data has 1 outputs, 1 inputs and 460 samples.
Number of poles: 2, Number of zeros: 0
Initialization Method: "iv"


Estimation Progress
Estimating delays as multiples of sample time ...
done.
Initializing model parameters...
Initializing using 'iv' method...
done.

Initialization complete.

Algorithm: Interior-Point
-----------------------------------------------------------------
                              Norm of     First-order
Iteration     Cost                step        optimality
-----------------------------------------------------------------
    0      5.0832e+06             -               -
    1     2.36668e+06     3.08e+03             257
    2     2.29876e+06          281             227
    3     2.07683e+06     1.24e+03             140

Result
Termination condition: Divergent gradient calculation..
Number of iterations: 68, Number of function evaluations: 120

Status: Estimated using TFEST
Fit to estimation data: 54.88%, FPE: 6023.62
```

Figura D.4 Reporte de la identificación de un sistema de segundo orden por System Identification de MatLab®.

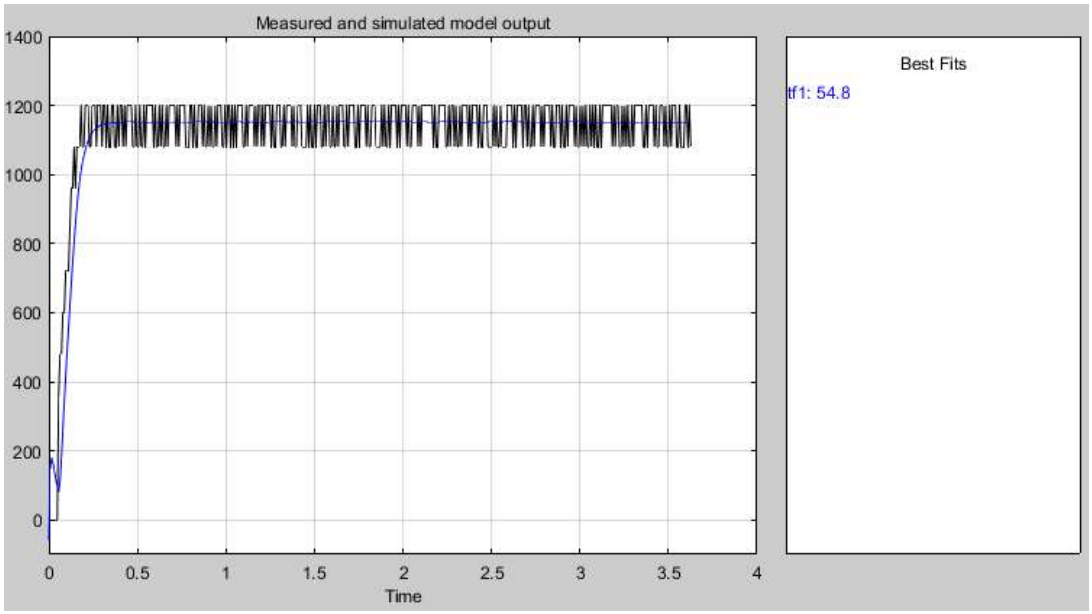


Figura D.5 Identificación de un sistema de segundo orden.

```
  From input "Voltaje" to output "Velocidad":
            3.59e05 (+/- 6.522e04)
  -----------------------------------------
  s^2 + 186.3 (+/- 31.67) s + 3856 (+/- 701)

Name: tf1
Continuous-time identified transfer function.

Parameterization:
   Number of poles: 2   Number of zeros: 0
   Number of free coefficients: 3
   Use "tfdata", "getpvec", "getcov" for parameters and their uncertainties.

Status:
Termination condition: Divergent gradient calculation..
Number of iterations: 68, Number of function evaluations: 120

Estimated using TFEST on time domain data "motor".
Fit to estimation data: 54.8% (stability enforced)
FPE: 6024, MSE: 5894
More information in model's "Report" property.
```

Figura D.6 Función de transferencia y datos de un sistema de segundo orden identificado.

ORDEN DE EMPASTADO