Optimización en el modelado de motor de inducción mediante Algoritmos Bioinspirados

Ing. Gustavo Medina Vega

2026



Universidad Autónoma de Querétaro

Facultad de Ingeniería

Maestría en Ciencias (Mecatrónica)

Optimización en el modelado de motor de inducción mediante Algoritmos Bioinspirados

Tesis

Que como parte de los requisitos para obtener el Grado de

Maestro en Ciencias (Mecatrónica)

Presenta

Ing. Gustavo Medina Vega

Dirigido por:

Dr. Luis Morales Velázquez

San Juan del Río, Qro. a 29 de Julio del 2026

REPOSITORIO INSTITUCIONAL DGBSDI-UAQ

La presente obra está bajo la licencia:
https://creativecommons.org/licenses/by-nc-nd/4.0/deed.es

CC BY-NC-ND 4.0 DEED

Atribución-NoComercial-SinDerivadas 4.0 Internacional

Usted es libre de:

Compartir — copiar y redistribuir el material en cualquier medio o formato

La licenciante no puede revocar estas libertades en tanto usted siga los términos de la licencia

Bajo los siguientes términos:

Atribución — Usted debe dar crédito de manera adecuada , brindar un enlace a la licencia, e indicar si se han realizado cambios . Puede hacerlo en cualquier forma razonable, pero no de forma tal que sugiera que usted o su uso tienen el apoyo de la licenciante.

NoComercial — Usted no puede hacer uso del material con propósitos comerciales .

SinDerivadas — Si remezcla, transforma o crea a partir del material, no podrá distribuir el material modificado.

No hay restricciones adicionales — No puede aplicar términos legales ni medidas tecnológicas que restrinjan legalmente a otras a hacer cualquier uso permitido por la licencia.

Avisos:

No tiene que cumplir con la licencia para elementos del material en el dominio público o cuando su uso esté permitido por una excepción o limitación aplicable.

No se dan garantías. La licencia podría no darle todos los permisos que necesita para el uso que tenga previsto. Por ejemplo, otros derechos como publicidad, privacidad, o derechos morales pueden limitar la forma en que utilice el material.



Universidad Autónoma de Querétaro

Facultad de Ingeniería

Maestría en Ciencias (Mecatrónica)

Optimización en el modelado de motor de inducción mediante Algoritmos Bioinspirados

Tesis

Que como parte de los requisitos para obtener el Grado de
Maestro en Ciencias (Mecatrónica)

Presenta

Ing. Gustavo Medina Vega

Dirigido por

Dr. Luis Morales Velázquez

Dr. Luis Morales Velázquez

Presidente

Dr. Luis Alberto Morales Hernández

Secretario

Dr. Juan Pablo Amézquita Sánchez

Vocal

Dr. Arturo Yosimar Jaen Cuellar

Suplente

Dr. José A. Antonino-Daviu

Suplente

San Juan del Río, Qro. a 29 de Julio del 2026

一期一会

Agradecimientos

Al Dr. Luis, por su apoyo constante desde la licenciatura y a lo largo de toda la maestría. Su guía, paciencia y confianza fueron pieza clave para llegar hasta aquí. Al cuerpo de sinodales, por su tiempo y observaciones.

Al Dr. Roque, por sembrar la idea de estudiar una maestría; sin esa conversación, probablemente este documento no existiría.

A mi familia, por sostenerme y creer en mí en cada etapa de este camino. A Thalles, pela confiança e pelas oportunidades que permitiram avançar com menos peso durante esse processo.

A Emi, mi novia, por su paciencia, su cariño y por estar presente incluso en los momentos más pesados de este proceso. A estas alturas sigo descubriendo cosas de ti, y me gusta que sea así: nunca se termina de conocer del todo a alguien, y contigo eso no es un problema, es una promesa.

A mis amigos del cubo, Charlie, Citla y Meli, por el apoyo, las distracciones necesarias y por seguir ahí a pesar de mis ausencias. Voy a extrañar esos momentos: aunque fueron efímeros, fueron de los mejores.

Y en general, a todos los que de una forma u otra, dentro de la facultad, ayudaron a que este momento llegara.

A Josh, Lalo y Leo, por su amistad de siempre y por seguir ahí sin importar el camino que tomara. A Leo en especial, por esa plática de 2023 que, sin saberlo, me metió al mundo de los datos.

瀬戸飛鳥先生へ。日本語を教えてくださり、ありがとうございます。新しいことを学ぶのはいつも価値があると、先生のおかげで改めて実感しました。

A mis gatos, Rita, Ramona, Remi, Jacinta y Nuza, por su compañía silenciosa en las noches largas de escritura.

Y a quienes ya no están para ver esto terminado, pero que de alguna forma siguen presentes en cada página.

Amor fati.

Resumen

El modelado preciso del motor de inducción bajo fallas mecánicas requiere identificar sus parámetros a partir de señales experimentales. Los métodos convencionales de mínimos cuadrados no lineales dependen fuertemente del punto inicial y suelen converger a mínimos locales. En este trabajo se presenta un método Híbrido que combina la exploración global de la Evolución Diferencial (DE) con el refinamiento local del algoritmo L-BFGS-B para identificar los siete parámetros del modelo *d-q* del motor: las resistencias de estátor y rotor (R_s, R_r), las inductancias de dispersión de estátor y rotor (L_s, L_r), la inductancia de magnetización (L_m), el momento de inercia (J) y el coeficiente de fricción viscosa (B). La metodología se validó sobre 12 registros de arranque directo de un motor de 5.5 kW en tres condiciones: sano, desalineamiento y aflojamiento, comparándola contra los métodos LM/TRF y PSO. El método Híbrido DE+L-BFGS-B alcanzó un error cuadrático medio global de $158 \pm 76\,\mathrm{A}^2$, frente a $2520 \pm 143\,\mathrm{A}^2$ del LM/TRF (reducción del 93.7 %), y $R^2 = 0.949$ en validación cruzada. Además, los parámetros identificados variaron de forma físicamente coherente con cada falla, lo que permite emplearlos como indicador de diagnóstico a partir de la corriente de arranque.

Palabras clave: Algoritmos bioinspirados, motor de inducción, optimización paramétrica, modelado no lineal, desalineamiento, aflojamiento.

Abstract

Accurate modeling of induction motors under mechanical faults requires identifying their parameters from experimental signals. Conventional nonlinear least-squares methods strongly depend on the initial guess and often converge to local minima. This work presents a Hybrid method that combines the global exploration of Differential Evolution (DE) with the local refinement of the L-BFGS-B algorithm to identify the seven parameters of the motor *d*-*q* model: the stator and rotor resistances (R_s, R_r), the stator and rotor leakage inductances (L_s, L_r), the magnetizing inductance (L_m), the moment of inertia (J), and the viscous friction coefficient (B). The methodology was validated on 12 direct-on-line start-up records of a 5.5 kW motor under three conditions: healthy, misalignment, and looseness, and compared against the LM/TRF and PSO methods. The Hybrid DE+L-BFGS-B method reached a global mean squared error of $158 \pm 76\,\mathrm{A}^2$, versus $2520 \pm 143\,\mathrm{A}^2$ for LM/TRF (a 93.7 % reduction), and $R^2 = 0.949$ in cross-validation. Moreover, the identified parameters varied consistently with each fault, enabling their use as a diagnostic indicator based on the start-up current.

Keywords: Bio-inspired algorithms, induction motor, parametric optimization, nonlinear modeling, misalignment, looseness.

Índice

Agradecimientos **4**

Resumen **5**

Abstract **6**

Índice de cuadros **10**

Índice de figuras **11**

1 Introducción **12**

1.1 Antecedentes 12

1.1.1 Importancia y evolución de los motores en la industria 13

1.1.2 Avances en modelado y caracterización de motores de inducción 13

1.1.3 Técnicas de optimización convencionales y sus limitaciones . . 14

1.1.4 Principios de los algoritmos bioinspirados. 15

1.1.5 Aplicaciones de algoritmos bioinspirados en sistemas eléctricos 16

1.1.6 Áreas emergentes y desafíos abiertos 16

1.1.7 Estudios previos y resultados experimentales 17

1.1.8 Casos de estudio en motores de inducción 18

1.1.9 Tendencias futuras 18

1.1.10 Síntesis y oportunidades de contribución 18

1.2 Descripción del problema 21

1.3 Justificación 21

1.4 Hipótesis 22

1.5 Objetivos 22

1.5.1 Objetivo general 23

1.5.2 Objetivos específicos 23

2 Fundamentación Teórica **24**

2.1 Modelado de Motores Eléctricos 24

2.1.1 Modelos matemáticos de motores de inducción 25

2.1.2 Representación en coordenadas *d-q* 26

2.1.3 Espacio de búsqueda y parámetros nominales 28

2.1.4 Suposiciones y limitaciones de modelos lineales 29

2.2 Optimización en Sistemas de Ingeniería 31

2.2.1 Conceptos generales de optimización 31

2.2.2 Mínimos cuadrados no lineales: LM y Trust Region Reflective . 32

2.3 Algoritmos Bioinspirados 34

2.3.1 Algoritmos Genéticos (GA) 34

2.3.2 Optimización por Enjambre de Partículas (PSO) 36

2.3.3 Evolución Diferencial (DE) 38

2.3.4 Optimización por Forrajeo Bacteriano (BFO) 39

2.3.5 Criterios de convergencia y parada 40

2.3.6 Hibridación secuencial: DE + L-BFGS-B 40

2.4 Criterios de Evaluación 42

2.5 Manifestación de fallas mecánicas en el modelo d-q 44

2.5.1 Desalineamiento 44

2.5.2 Aflojamiento 44

3 Metodología **46**

3.1 Etapa 1. Adquisición y preprocesamiento de señales 48

3.2 Etapa 2. Formulación del modelo *d-q* y función objetivo 49

3.3 Etapa 3. Selección de métodos de identificación 51

3.4 Etapa 4. Implementación computacional 52

3.5 Etapa 5. Validación y análisis comparativo 53

3.6 Etapa 6. Postprocesamiento y generación de resultados 54

3.7 Software, repositorio y reproducibilidad 55

3.7.1 Entorno de software y dependencias 55

3.7.2 Scripts desarrollados 56
3.7.3 Fragmentos de código representativos 57
3.7.4 Estructura de archivos de entrada y salida 60
3.7.5 Disponibilidad del código y datos 60
3.7.6 Guía de reproducción 61

4 Resultados y Discusión 62

4.1 Etapa exploratoria sobre señal sintética 62
4.2 Comparativa global de ECM 65
4.3 Mejora porcentual relativa 66
4.4 Validación cruzada R^2 68
4.5 Análisis de variación paramétrica 69
4.6 Comparación de corriente simulada vs. experimental 72
4.7 Discusión comparativa con la literatura 73
4.8 Análisis de superioridad por archivo 75
4.9 Resumen de resultados 78

5 Conclusiones 79

Referencias 82

Apéndice A Código principal: versionFinal.py 85

Apéndice B Código exploracion sintetica: exploracion_sintetica.py 98

Apéndice C Physics-Informed Parametric Fault Signatures Beyond Black-Box Classification in Induction Motors 104

Apéndice D Análisis comparativo mediante Inteligencia de Enjambre para la Identificación Paramétrica aplicada a motores de inducción 105

Apéndice E Optimización de Carteras de Inversión Financiera mediante un Modelo Híbrido GARCH y Algoritmo Genético 106

Índice de cuadros

2.1 Espacio de búsqueda y parámetros nominales de inicialización del método LM/TRF. 29
3.1 Configuración de los tres métodos de identificación. 53
3.2 Dependencias del entorno de software. 55
3.3 Funciones del script principal `versionFinal.py`. 56
3.4 Archivos de salida generados por los scripts. 60
4.1 Resultados de la etapa exploratoria sobre señal sintética. 63
4.2 Archivos en que el Híbrido obtiene ECM estrictamente menor. 65
4.3 Prueba de Wilcoxon pareada sobre el ECM del eje α (Híbrido vs. PSO, $n = 12$). 66
4.4 Mejora porcentual del ECM del Híbrido respecto a cada referencia. 66
4.5 R^2 en el eje de identificación (α) y de validación cruzada (β) — método Híbrido. 68
4.6 Parámetros identificados por el método Híbrido DE+L-BFGS-B (media ± desviación estándar por condición). 70
4.7 ECM (A^2) por método y condición operativa (media ± desviación estándar). 72
4.8 ECM y R^2 por método para el registro `Loose_DOLstart_0` (aflojamiento, 0 % de carga nominal). 77
4.9 Comparativa global del método Híbrido frente a cada referencia (12 registros experimentales, ventana 1.6–2.0 s). 78
4.10 Métricas globales del método Híbrido por método (12 registros experimentales, ventana 1.6–2.0 s). 78

Índice de figuras

2.1 Diagrama del circuito equivalente de un motor de inducción (Autoría Propia). 24
3.1 Esquema general de las etapas metodológicas 46
3.2 Flujo detallado de la implementación computacional: modelado del sistema, identificación paramétrica mediante los tres métodos evaluados, y diagnóstico basado en el vector $\boldsymbol{\theta}^*$. 47
4.1 Etapa exploratoria, comparación de estrategias de optimización sobre señal sintética. a) ECM por método. b) R^2 por método. Señal con ruido gaussiano ($\sigma = 0.8\,\mathrm{A}$), parámetros verdaderos conocidos, DS = 8. 63
4.2 Etapa exploratoria, corriente simulada vs. referencia sintética. a) LM/TRF. b) PSO estándar. c) DE solo. d) Híbrido DE+L-BFGS-B propuesto. 64
4.3 Mejora porcentual del Híbrido frente al LM/TRF por condición operativa. Barras de error: desviación estándar sobre los cuatro registros de cada condición. Ventana 1.6–2.0 s. 68
4.4 R^2 de identificación (R^2_α, azul) y validación cruzada (R^2_β, morado) por condición operativa. Método Híbrido, ventana 1.6–2.0 s. 69
4.5 Distribución de parámetros mecánicos J y B por condición operativa (Método Híbrido). Ventana 1.6–2.0 s. 71
4.6 Distribución de parámetros eléctricos R_s, R_r, L_s, L_r y L_m por condición operativa (Método Híbrido). S = Sano, Af = Aflojamiento, De = Desalineamiento. Ventana 1.6–2.0 s. 72
4.7 ECM por método y condición operativa. Escala logarítmica. Ventana 1.6–2.0 s; DE: popsize = 10, maxiter = 15. 73
4.8 Comparación de la corriente de estátor I_α simulada contra la señal experimental de referencia `Loose_DOLstart_0` (aflojamiento, 0 % de carga nominal), ventana 1.6–2.0 s. 76

I. Introducción

Los motores de inducción constituyen el componente de mayor prevalencia en los procesos industriales modernos, representando aproximadamente el 70 % del consumo eléctrico en el sector de manufactura global (Hannan et al., 2018). La importancia de su modelado preciso se ha convertido en un desafío científico y tecnológico en creciente auge, siendo componentes fundamentales en la demanda actual de eficiencia energética.

A pesar de que los modelos convencionales otorgan aproximaciones útiles, su principal limitante es la representación de fenómenos no lineales como la saturación magnética, pérdidas en el entorno frecuencial y efectos térmicos (Bozek et al., 2023). Frente a este panorama, los algoritmos bioinspirados han ganado importancia como herramientas prometedoras para superar las limitaciones de los métodos tradicionales (Yang, 2021).

1.1 Antecedentes

Los motores de inducción son de gran importancia en la industria moderna, como se mencionó anteriormente su consumo representa una proporción mayoritaria del consumo eléctrico industrial global, con participación en cada uno de los sectores productivos. La importancia de su modelado preciso se ha convertido en un desafío científico y tecnológico en creciente auge, siendo componentes fundamentales en la demanda actual de eficiencia energética. A pesar de que los modelos convencionales otorgan aproximaciones útiles durante las últimas décadas. Su principal limitante, por su naturalidad del método, es la representación de fenómenos no lineales como la saturación magnética, problemas en el entorno frecuencial y a su vez los efectos térmicos han logrado impulsar la búsqueda de enfoques alternativos más eficientes y robustos. Referente a este panorama, los algoritmos bioinspirados, siendo estos los principales métodos para emular procesos de sistemas biológicos naturales, han ganado importancia como herramientas prometedoras con la capacidad de poder abordar la complejidad de la representación de modelos de motores de inducción, logrando ofrecer un potencial para superar las limitaciones de los métodos

tradicionales.

1.1.1 Importancia y evolución de los motores en la industria

Los motores de inducción constituyen elementos fundamentales en el sector industrial debido a su versatilidad, robustez y eficiencia operativa (Ocampo et al., 2014). Estos dispositivos representan la mayor proporción del consumo eléctrico industrial a nivel global, destacando la importancia de optimizar su rendimiento desde perspectivas económicas y ambientales (Hannan et al., 2018).

Sin embargo, la optimización del rendimiento de estos sistemas enfrenta desafíos significativos en el modelado preciso bajo condiciones operativas no ideales. Investigaciones recientes han demostrado que distorsiones armónicas y desequilibrios de voltaje pueden comprometer la eficiencia de motores de alto rendimiento, evidenciando las limitaciones de los modelos matemáticos tradicionales para predecir comportamientos no lineales como saturación magnética y efectos térmicos (Tabora et al., 2021). La gestión térmica emerge como factor crítico, ya que las pérdidas por calor constituyen un obstáculo relevante para alcanzar mayores eficiencias, requiriendo enfoques de modelado más sofisticados (Gundabattini et al., 2021). En este contexto, la implementación de metodologías avanzadas de optimización, incluyendo algoritmos bioinspirados, representa una oportunidad significativa para superar las limitaciones de los métodos lineales convencionales frente a condiciones de falla mecánica como desalineamiento y aflojamiento.

1.1.2 Avances en modelado y caracterización de motores de inducción

El modelado de motores de inducción ha evolucionado hacia modelos multifísicos, aunque los modelos tradicionales basados en circuitos eléctricos equivalentes presentan limitaciones al capturar fenómenos no lineales como la saturación magnética, pérdidas en frecuencia y efectos térmicos (Bozek et al., 2023), generando discrepancias entre predicciones y comportamiento real bajo condiciones dinámicas. Como alternativa, los modelos híbridos combinan precisión electromagnética con parámetros concentrados, logrando reducciones

del orden del 40–45 % en el error de estimación comparado con modelos convencionales sin incrementar tiempos de cálculo. La caracterización experimental ha evolucionado hacia métodos menos intrusivos basados en respuesta en frecuencia y análisis de armónicos, permitiendo identificación más precisa de parámetros críticos (Toliyat and Kliman, 2004). El desarrollo de sensores no invasivos y técnicas avanzadas de procesamiento facilita el monitoreo continuo de variables eléctricas del estátor (Tallam et al., 2007).

Revisiones recientes confirman que la identificación paramétrica mediante datos de arranque transitorio representa la tendencia dominante en el modelado de motores de inducción para aplicaciones de control y diagnóstico. Sengamalai et al. (2022) presentan una revisión sistemática de métodos de modelado dinámico y estimación de parámetros de motores de inducción trifásicos, empleando como marco de referencia el modelo d-q en el sistema rotante orientado al flujo del rotor. Asimismo, Konuhova (2025) validan experimentalmente cuatro variantes del modelo d-q sobre un motor de 3 kW, confirmando que los modelos basados en flujo concatenado capturan con mayor fidelidad los fenómenos transitorios frente a los modelos basados en corriente.

1.1.3 Técnicas de optimización convencionales y sus limitaciones

Las técnicas clásicas de optimización aplicadas al modelado y control de motores de inducción incluyen métodos analíticos, basados en gradiente y de búsqueda directa. Los enfoques analíticos, fundamentados en ecuaciones diferenciales y transformaciones matemáticas como Park y Clarke, han proporcionado soluciones elegantes para problemas específicos bajo condiciones idealizadas. Sin embargo, su aplicabilidad es limitada cuando se considera la no linealidad intrínseca del motor, llegando a desviarse del comportamiento real del sistema.

Los métodos basados en gradientes han sido implementados en estimación de parámetros y sintonización de controladores debido a su rigor matemático y eficiencia computacional. Sin embargo, presentan dependencia crítica a puntos iniciales adecuados y son susceptibles a quedar atrapados en óptimos locales. Respecto a la identificación paramétrica de motores de

inducción, el punto de inicio de dichos métodos se deriva habitualmente de los parámetros nominales obtenidos utilizando las relaciones de semejanza eléctrica a partir de los datos de placa del fabricante, como la resistencia de estátor y rotor, las inductancias de dispersión e inductancias de magnetización, como se documenta en (Krishnan, 2001). Aunque el procedimiento es sistemático y reproducible, la calidad que se obtiene en la solución final queda condicionada a qué tan lejos se encuentra del punto de partida al mínimo global, siendo una limitación que los algoritmos bioinspirados superan desde su diseño.

Las técnicas de búsqueda directa ofrecen mayor robustez en problemas con restricciones complejas al no requerir información de gradientes. No obstante, su convergencia es lenta y su rendimiento se reduce en espacios de alta dimensionalidad (Boussaïd et al., 2013). Análisis comparativos muestran que requieren mayor número de evaluaciones para alcanzar precisiones similares a técnicas avanzadas, siendo un obstáculo para aplicaciones con restricciones de tiempo o recursos computacionales.

1.1.4 Principios de los algoritmos bioinspirados.

Los algoritmos bioinspirados son métodos de optimización metaheurística que emulan procesos biológicos naturales, ganando importancia como alternativa en problemas complejos de alta dimensionalidad (Yang, 2021). Su conceptualización se basa en principios evolutivos, comportamiento colectivo y adaptación biológica, explorando eficientemente espacios de búsqueda extensos sin requerir información de gradientes (Boussaïd et al., 2013).

Los principales algoritmos incluyen: Algoritmos Genéticos (GA) basados en selección natural mediante mecanismos de selección, cruce y mutación; Optimización por Enjambre de Partículas (PSO) que emula comportamiento social de aves donde cada individuo aporta experiencia al enjambre (Kennedy and Eberhart, 1995).

Las ventajas principales incluyen capacidad para escapar de óptimos locales, manejar funciones discontinuas y adaptabilidad a restricciones cambiantes, siendo especialmente efectivos para manejar no linealidades de motores de inducción.

1.1.5 Aplicaciones de algoritmos bioinspirados en sistemas eléctricos

Los algoritmos bioinspirados han demostrado un impacto significativo en el modelado y control de motores de inducción. En identificación de parámetros, superan las limitaciones de enfoques clásicos al capturar complejidades no lineales del comportamiento electromagnético. Sakthivel y Subramanian (Sakthivel and Subramanian, 2012) compararon algoritmos genéticos (GA), optimización por enjambre de partículas (PSO) y optimización por forrajeo bacteriano (BFO) para la determinación de parámetros de circuito equivalente en motores de inducción a partir de datos de placa del fabricante, encontrando que BFO con configuración de paso quimiotáctico adaptativo obtuvo mejores soluciones en menor tiempo que GA y PSO.

Los algoritmos bioinspirados también se han aplicado al diseño y optimización de controladores, y en menor medida al diagnóstico y mantenimiento predictivo, como alternativa a las técnicas de sintonización y clasificación tradicionales. Katoch et al. (Katoch et al., 2021) documentan, en su revisión de algoritmos genéticos, cómo los operadores de selección, cruce y mutación han evolucionado hacia esquemas híbridos que amplían su aplicabilidad más allá de los problemas de optimización clásicos.

1.1.6 Áreas emergentes y desafíos abiertos

La utilización de una optimización bioinspirada en sistemas eléctricos evoluciona hacia la hibridación de algoritmos, combinando fortalezas de diferentes enfoques para superar limitaciones individuales. Boussaïd et al. (Boussaïd et al., 2013) demostraron que la combinación estratégica de métodos metaheurísticos mejora significativamente la calidad de soluciones y eficiencia computacional. En el presente marco, la Evolución Diferencial (DE), la cual fue propuesta por Storn y Price (Storn and Price, 1997), destaca como un operador de exploración global en esquemas híbridos dada su robustez ante funciones multimodales. La estrategia DE/best/1/bin, la cual utiliza 4 fases donde después de la Evolución Diferencial se utiliza al mejor vector individuo de la población actual como base para la mutación, para después utilizar una pareja de vectores (diferencial) para perturbar

el vector base y finalizar con un cruce binomial, ha demostrado una convergencia confiable en problemas donde se requiera identificar parámetros dinámicos no lineales, siendo efectiva cuando se combina el método con métodos de refinamiento local basados en gradientes los cuales explotan la región de atracción identificada en la fase global (Boussaïd et al., 2013; Storn and Price, 1997).

1.1.7 Estudios previos y resultados experimentales

Evidencia experimental confirma las ventajas teóricas de los algoritmos bioinspirados en la identificación de parámetros de motores de inducción: el estudio comparativo de Sakthivel y Subramanian (Sakthivel and Subramanian, 2012) entre BFO, PSO y GA frente al método clásico IEEE Std 112-F, usando únicamente datos de placa del fabricante, mostró que BFO obtiene soluciones de mejor calidad en menor número de evaluaciones que los otros dos algoritmos bioinspirados.

La disponibilidad de bases de datos de acceso libre representa un avance importante para el área de validación de metodologías en la identificación de parámetros en motores de inducción. Battulga et al. (Battulga et al., 2024) presentaron un sistema de monitoreo basado en un acelerómetro experimental MEMS capacitivo y sensor Hall, con el cual adquirieron una base de datos de más de 200 mediciones bajo seis condiciones de falla, en la cual incluyeron desalineamiento y aflojamiento en un motor de inducción trifásico de 380 V, 5.5 kW, cuatro polos, operando bajo 4 niveles diferentes de carga (0 %, 50 %, 75 % y 100 %) utilizando una alimentación directa en línea. Las señales de voltaje y de corriente fueron adquiridas a 10.24 kHz con un sistema de 16 bits.

Estos resultados confirman consistentemente que las ventajas teóricas de los algoritmos bioinspirados se traducen en mejoras prácticas significativas en aplicaciones reales de ingeniería eléctrica.

1.1.8 Casos de estudio en motores de inducción

Las aplicaciones exitosas de algoritmos bioinspirados en diversos aspectos de motores de inducción evidencian mejoras sustanciales frente a métodos convencionales:

- **Control adaptativo:** Passino (Passino, 2002) propuso conceptualmente el uso de BFO para ajustar los parámetros de un controlador adaptativo mediante la minimización del error de seguimiento, ilustrado mediante un ejemplo simulado, sentando las bases teóricas para su aplicación en sistemas con incertidumbre paramétrica.

Estos casos demuestran cómo los algoritmos bioinspirados pueden abordar la complejidad inherente a sistemas electromecánicos reales, proporcionando soluciones prácticas a desafíos que métodos convencionales no pueden resolver adecuadamente.

1.1.9 Tendencias futuras

El campo de optimización bioinspirada aplicada a motores de inducción continúa evolucionando hacia nuevas tendencias, cómo serían:

- **Hibridación de técnicas bioinspiradas:** Boussaïd et al. (Boussaïd et al., 2013) analizaron enfoques que combinan diferentes metaheurísticas para explotar sus fortalezas complementarias. Dichos algoritmos híbridos demuestran mejoras significativas tanto en velocidad de convergencia como en calidad final de soluciones, superando las limitaciones individuales de cada técnica.

Estas tendencias convergen hacia sistemas de optimización más sofisticados, adaptables e interpretables, con potencial para transformar fundamentalmente tanto el diseño como la operación de motores de inducción en la próxima generación de sistemas electromecánicos.

1.1.10 Síntesis y oportunidades de contribución

La revisión sistemática de las secciones anteriores permite trazar un mapa claro del estado del arte y de sus limitaciones. Los modelos convencionales basados en circuitos equivalentes

por fase ofrecen aproximaciones útiles en estado estacionario, pero presentan discrepancias de hasta un 25 % bajo condiciones dinámicas al no capturar saturación magnética, efectos térmicos ni pérdidas dependientes de la frecuencia (Bozek et al., 2023; Sengamalai et al., 2022). Las técnicas de optimización basadas en gradiente, como Levenberg–Marquardt, resuelven eficientemente el problema de identificación paramétrica cuando el punto inicial es cercano a la solución, pero convergen sistemáticamente a mínimos locales cuando se inicializan desde datos de placa bajo condiciones de falla mecánica (Krishnan, 2001).

Los algoritmos bioinspirados revisados en las Secciones 1.1.4 y 1.1.5 superan esta limitación mediante búsqueda global, obteniendo mejores soluciones que los métodos clásicos de estimación en condiciones de estado estacionario (Sakthivel and Subramanian, 2012). Sin embargo, la mayoría de los estudios existentes evalúan un único algoritmo sobre señales sintéticas o en régimen estacionario, sin considerar el transitorio de arranque directo en línea ni condiciones simultáneas de falla mecánica. Adicionalmente, las métricas reportadas se limitan al ECM sin analizar la variación estadística de los parámetros identificados entre condiciones operativas, lo que impide explotar el potencial diagnóstico implícito del vector de parámetros $\boldsymbol{\theta}^*$ (siendo $\boldsymbol{\theta}^*$ el conjunto completo de los siete parámetros eléctricos y mecánicos del modelo *d-q* — R_s, R_r, L_s, L_r, L_m, J y B —, y no un parámetro aislado).

La hibridación secuencial de Evolución Diferencial con refinamiento local, documentada como tendencia emergente por Boussaïd et al. (Boussaïd et al., 2013) y validada recientemente en estimación de parámetros de motores (Santana et al., 2025), representa la estrategia con mayor potencial para cerrar esta brecha, al combinar la exploración global necesaria en problemas multimodales con la precisión de convergencia que los métodos basados en gradiente ofrecen en la región de atracción del óptimo.

La Facultad de Ingeniería de la UAQ, Campus San Juan del Río, cuenta con un grupo consolidado en el monitoreo de motores de inducción trifásicos. Pacheco Guerrero (Pacheco Guerrero, 2023) desarrolló metodologías para la detección de desbalance y desalineamiento mediante análisis en frecuencia (FFT), mientras que Zamudio Ramírez (Zamudio Ramírez, 2019) implementó técnicas de detección de fallas de aislamiento utilizando FPGA, con

reducción en los tiempos de ejecución respecto a una implementación por software.

En el ámbito de publicaciones internacionales del mismo grupo, Zamudio-Ramírez et al. (Zamudio-Ramírez et al., 2020a,b) propusieron un smart-sensor basado en análisis de flujo de dispersión transitorio para la detección automática de fallas electromecánicas, incluyendo desalineamiento y barras rotas, durante el arranque directo en línea, Valtierra-Rodriguez et al. (Valtierra-Rodriguez et al., 2020) aplicaron redes neuronales convolucionales sobre señales de corriente en el estado transitorio para la detección de barras rotas en el mismo campus. En trabajos más recientes, Cureño-Osornio et al. (Cureño-Osornio et al., 2024) propusieron la separación automática entre los regímenes de arranque y estado estacionario mediante estadísticos de ventana corta, utilizando PSO para la determinación de parámetros óptimos de diseño.

Estas contribuciones locales abordan el diagnóstico de fallas a través del análisis espectral y de señales en el dominio tiempo-frecuencia, sin considerar la identificación del vector paramétrico completo $\boldsymbol{\theta} = [R_s, R_r, L_s, L_r, L_m, J, B]^\top$ del modelo d-q bajo condiciones de falla mecánica en el transitorio de arranque DOL. La presente investigación se posiciona como extensión natural de esta línea de trabajo, incorporando la perspectiva de la optimización bioinspirada para conectar la identificación paramétrica con el diagnóstico implícito de condición.

Con base en este análisis, la presente investigación propone una contribución original en tres dimensiones: *(i)* el desarrollo de una metodología híbrida DE + L-BFGS-B para la identificación paramétrica del modelo d-q sobre señales reales de arranque DOL bajo condiciones de desalineamiento y aflojamiento; *(ii)* un protocolo de evaluación comparativa que incorpora validación cruzada entre ejes ortogonales independientes (α-β) para verificar la generalización del modelo sin re-optimización; y *(iii)* el análisis de la variación estadística del vector $\boldsymbol{\theta}^*$ entre condiciones operativas como indicador implícito de diagnóstico, conectando la identificación paramétrica con la detección de fallas mecánicas en motores de inducción trifásicos.

1.2 Descripción del problema

Modelar de manera precisa el comportamiento de motores de inducción es un desafío importante en la ingeniería (Bozek et al., 2023). Los enfoques convencionales se basan en aproximaciones lineales que presentan las siguientes limitaciones:

- **Incapacidad para representar fenómenos no lineales:** La saturación magnética no puede representarse con modelos lineales.
- **Baja capacidad de caracterización de pérdidas dependientes en la frecuencia:** La histéresis magnética y las corrientes parásitas se aproximan a valores constantes (Toliyat and Kliman, 2004).
- **Falta de efectos térmicos acoplados:** La variación de resistencias con la temperatura no se contempla (Gundabattini et al., 2021).
- **Susceptibilidad a óptimos locales:** Los métodos basados en gradiente tienden a estancarse en soluciones subóptimas (Boussaïd et al., 2013).

La revisión del estado del arte revela la ausencia de metodologías que evalúen el desempeño de algoritmos bioinspirados durante el estado transitorio de arranque directo en línea (DOL) bajo condiciones de falla mecánica. Adicionalmente, las métricas de comparación reportadas se restringen al ECM sin analizar la variación estadística de los parámetros entre condiciones operativas.

1.3 Justificación

La presente investigación representa un área de oportunidad con impacto multidimensional al abordar la identificación paramétrica del modelo d-q de motores de inducción bajo condiciones de desalineamiento y aflojamiento:

- **Científico:** La metodología híbrida propuesta, que combina Evolución Diferencial con refinamiento local L-BFGS-B, aborda directamente la limitación de los métodos basados en gradiente ante puntos iniciales alejados del óptimo global, escenario

recurrente cuando la inicialización se realiza únicamente con datos de placa bajo condiciones de falla mecánica.

- **Tecnológico:** El vector de parámetros identificados $\boldsymbol{\theta}^*$ permite extraer información diagnóstica implícita a partir de su variación estadística entre condiciones operativas, sin requerir modificaciones a la estructura del modelo d-q ni instrumentación adicional al motor.
- **Económico:** Al utilizar únicamente señales de arranque directo en línea (DOL) y datos de placa del fabricante como punto de partida, la metodología evita la necesidad de pruebas de laboratorio especializadas o sensores adicionales, reduciendo el costo de implementación de estrategias de mantenimiento predictivo en motores de baja tensión (1–10 kW) de uso industrial extendido.
- **Social:** La reproducibilidad del método, sustentada en una base de datos de acceso público (Battulga et al., 2024) y datos de placa disponibles para cualquier motor comercial, favorece su adopción en contextos con recursos técnicos limitados, contribuyendo a la democratización de herramientas de diagnóstico predictivo y a la reducción de fallas no detectadas que derivan en paros de planta y consumo energético ineficiente.

1.4 Hipótesis

Con la aplicación de algoritmos bioinspirados al modelado de motores de inducción con fenómenos de desalineamiento y aflojamiento, se espera obtener una mejora del 25–30 % en precisión de la identificación de la falla, medida mediante el error cuadrático medio (ECM), un incremento del 15–20 % en el coeficiente de determinación R^2 y una mejora del 35–45 % en la capacidad de representación de fenómenos no lineales respecto al desempeño obtenido con métodos convencionales basados en mínimos cuadrados no lineales.

1.5 Objetivos

1.5.1 Objetivo general

Desarrollar una metodología para la optimización del modelado de motores de inducción mediante algoritmos bioinspirados, que permitirá la detección de fenómenos de desalineamiento y aflojamiento bajo condiciones operativas variables, capturando adecuadamente fenómenos no lineales, y comparar su desempeño frente a métodos convencionales.

1.5.2 Objetivos específicos

1. Desarrollar representaciones paramétricas optimizadas del motor de inducción que incorporen explícitamente los fenómenos no lineales.
2. Diseñar un proceso sistemático de evaluación comparativa de algoritmos bioinspirados para la identificación de parámetros del modelo, estableciendo métricas que valoren precisión en la predicción de variables eléctricas y magnéticas, robustez ante condiciones operativas variables y eficiencia computacional frente a métodos convencionales.
3. Formular una metodología híbrida que potencie los algoritmos bioinspirados mediante la integración estructurada de conocimiento físico del dominio, mejorando tanto la convergencia algorítmica como la interpretabilidad física de los parámetros optimizados dentro del marco metodológico integral propuesto.
4. Implementar un protocolo de validación por simulación que evalúe sistemáticamente los modelos optimizados bajo diversos perfiles de operación.

II. Fundamentación Teórica

El modelado y la optimización de motores de inducción mediante algoritmos bioinspirados constituyen un punto clave en la unión de múltiples campos técnicos, A continuación se desarrolla el marco teórico de la presente investigación, desde los principios fundamentales en la naturaleza de los motores hasta profundizar en técnicas de optimización bioinspiradas más avanzadas.

2.1 Modelado de Motores Eléctricos

Los motores de inducción son de suma importancia en aplicaciones industriales dada su robustez y eficiencia en sistemas de control. Su representación teórica se presenta en la Figura 2.1.

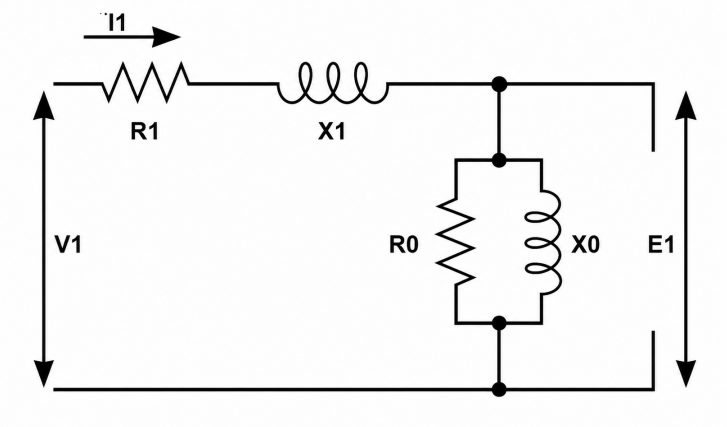


Figura 2.1: Diagrama del circuito equivalente de un motor de inducción (Autoría Propia).

Esta representación monofásica en estado estacionario es una simplificación teórica del comportamiento del motor; el modelo dinámico d-q en α-β empleado en la metodología se describe en la Sección 2.1.2.

2.1.1 Modelos matemáticos de motores de inducción

El modelado involucra ecuaciones diferenciales no lineales que describen el comportamiento electromagnético entre el estátor y el rotor. Dichas ecuaciones se expresan de manera matricial como:

$$\begin{bmatrix} \mathbf{v}_s \\ \mathbf{v}_r \end{bmatrix} = \begin{bmatrix} R_s & 0 \\ 0 & R_r \end{bmatrix} \begin{bmatrix} \mathbf{i}_s \\ \mathbf{i}_r \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \boldsymbol{\Psi}_s \\ \boldsymbol{\Psi}_r \end{bmatrix} \tag{1}$$

donde $\mathbf{v}_s$, $\mathbf{v}_r$ son los vectores de voltaje; $\mathbf{i}_s$, $\mathbf{i}_r$ los vectores de corriente; y R_s, R_r las matrices de resistencia del estátor y rotor.

El enfoque más común para su análisis ha sido el modelado del circuito equivalente en términos de ecuaciones diferenciales equivalentes por fase. La relación entre flujos magnéticos y corrientes es:

$$\begin{bmatrix} \boldsymbol{\Psi}_s \\ \boldsymbol{\Psi}_r \end{bmatrix} = \begin{bmatrix} \mathbf{L}_s & \mathbf{L}_{rs} \\ \mathbf{L}_{sr} & \mathbf{L}_r \end{bmatrix} \begin{bmatrix} \mathbf{i}_s \\ \mathbf{i}_r \end{bmatrix} \tag{2}$$

donde $\mathbf{L}_s$, $\mathbf{L}_r$ son las matrices de inductancia propia y $\mathbf{L}_{sr}$, $\mathbf{L}_{rs}$ las de inductancia mutua.

La validación experimental de modelos matemáticos de motores de inducción constituye una práctica necesaria para confirmar la fidelidad del marco teórico adoptado. En este sentido, Konuhova (2025) implementaron y validaron experimentalmente cuatro variantes del modelo d-q contra datos de un motor de 3 kW, encontrando resultados consistentes entre las distintas variantes, con la excepción de que los modelos basados en flujo concatenado capturan con mayor fidelidad los fenómenos transitorios frente a los modelos basados en corriente. De manera complementaria, Rodríguez-Blanco et al. (2022) demostraron que el modelo d-q simplificado es apto para el diagnóstico de fallas paramétricas mediante ecuaciones de paridad, estableciendo así un vínculo directo entre la identificación de parámetros y la detección de condiciones de falla en el motor.

2.1.2 Representación en coordenadas d-q

La transformada de Park permite proyectar las variables trifásicas del estátor sobre un sistema de referencia estacionario de dos ejes, denominados eje directo (d) y eje de cuadratura (q), lo que elimina la dependencia angular de las inductancias y reduce el orden del sistema. En la referencia estacionaria, también conocida como α-β, el sistema de ecuaciones que gobierna la dinámica electromagnética y mecánica del motor de inducción queda definido por cinco variables de estado: las corrientes de estátor i_{ds} e i_{qs}, los flujos del rotor $\varphi_{r\alpha}$ y $\varphi_{r\beta}$, y la velocidad angular mecánica del rotor ω_r.

Para simplificar la notación se define la **resistencia efectiva de estátor**, que agrupa la caída resistiva del estátor con la contribución resistiva del rotor referida al estátor a través del acoplamiento magnético:

$$R_{s,\text{eff}} = R_s + \frac{L_m^2\, R_r}{L_r^2} \tag{3}$$

La dinámica de la corriente de estátor en el **eje directo** se expresa como:

$$\frac{di_{ds}}{dt} = \frac{1}{\sigma L_s}\left(V_\alpha - R_{s,\text{eff}}\, i_{ds} + \frac{L_m R_r}{L_r^2}\,\varphi_{r\alpha} + P\omega_r \frac{L_m}{L_r}\,\varphi_{r\beta}\right) \tag{4}$$

La dinámica de la corriente de estátor en el **eje de cuadratura**, considerando $v_{qs} = 0$ en la referencia estacionaria, se expresa como:

$$\frac{di_{qs}}{dt} = \frac{1}{\sigma L_s}\left(V_\beta - R_{s,\text{eff}}\, i_{qs} + \frac{L_m R_r}{L_r^2}\,\varphi_{r\beta} - P\omega_r \frac{L_m}{L_r}\,\varphi_{r\alpha}\right) \tag{5}$$

La evolución temporal de los **flujos del rotor** en los ejes directo y de cuadratura queda determinada por:

$$\frac{d\varphi_{r\alpha}}{dt} = \frac{L_m R_r}{L_r}\, i_{ds} - \frac{R_r}{L_r}\, \varphi_{r\alpha} - P\omega_r\, \varphi_{r\beta} \tag{6}$$

$$\frac{d\varphi_{r\beta}}{dt} = \frac{L_m R_r}{L_r}\, i_{qs} - \frac{R_r}{L_r}\, \varphi_{r\beta} + P\omega_r\, \varphi_{r\alpha} \tag{7}$$

donde el **coeficiente de dispersión total** σ se define como:

$$\sigma = 1 - \frac{L_m^2}{L_s\, L_r} \tag{8}$$

El **par electromagnético** del motor se obtiene a partir de la interacción entre los flujos del rotor y las corrientes del estátor:

$$T_e = \frac{3}{2}\, P\, \frac{L_m}{L_r}\big(\varphi_{r\alpha}\, i_{qs} - \varphi_{r\beta}\, i_{ds}\big) \tag{9}$$

La **ecuación mecánica del eje**, que relaciona el par electromagnético con la inercia del rotor y las pérdidas por fricción viscosa, completa el sistema de ecuaciones de estado:

$$\frac{d\omega_r}{dt} = \frac{1}{J}\big(T_e - T_L - B\omega_r\big) \tag{10}$$

donde T_L representa el **par de carga**, modelado como función cuadrática de la velocidad para representar una carga de tipo ventilador o bomba centrífuga:

$$T_L = k_L\, \omega_r^2 \tag{11}$$

El **vector de parámetros** a identificar queda definido como:

$$\boldsymbol{\theta} = \big[R_s,\ R_r,\ L_s,\ L_r,\ L_m,\ J,\ B\big]^T \tag{12}$$

donde R_s y R_r son las resistencias de estátor y rotor, L_s y L_r las inductancias propias, L_m

la inductancia de magnetización, J el momento de inercia del rotor y B el coeficiente de fricción viscosa. Para garantizar la realizabilidad física del modelo, los parámetros deben satisfacer la **condición de acoplamiento magnético**:

$$L_m < \sqrt{L_s\, L_r} \tag{13}$$

2.1.3 Espacio de búsqueda y parámetros nominales

La identificación de los siete parámetros del vector requiere definir, por un lado, el dominio físicamente valido que delimita el espacio de búsqueda para los algoritmos de optimización, por otro, un punto de partida determinístico para el método convencional basado en gradiente. Ambos elementos se derivan de los datos de placa del motor experimental, 380 V (línea), 5.5 kW, cuatro polos, 60 Hz, conexión Y, y de las relaciones de semejanza eléctrica documentadas en Krishnan (Krishnan, 2001). $\boldsymbol{\theta} = [R_s, R_r, L_s, L_r, L_m, J, B]^T$ requiere definir, por un lado, el dominio físicamente valido que delimita el espacio de búsqueda para los algoritmos de optimización, por otro, un punto de partida determinístico para el método convencional basado en gradiente.

Los límites del espacio de búsqueda se derivan de rangos físicamente admisibles para motores de baja tensión (1–10 kW, 380 V). El punto inicial $\boldsymbol{\theta}_0$ para el método LM/TRF se obtiene escalando un motor de referencia (220 V, 3 hp) mediante el **factor de semejanza eléctrica** (Krishnan, 2001):

$$k_Z = \left(\frac{V_{\text{new}}}{V_{\text{ref}}}\right)^2 \cdot \frac{P_{\text{ref}}}{P_{\text{new}}} = \left(\frac{380}{220}\right)^2 \cdot \frac{2240}{5500} = 1.215 \tag{14}$$

Tabla 2.1: Espacio de búsqueda y parámetros nominales de inicialización del método LM/TRF.

Par.	**Descripción**	**Lím. inf.**	**Lím. sup.**	$\boldsymbol{\theta}_0$	**Justificación**
R_s (Ω)	Resist. estátor	1.0	15.0	2.152	Rango 1–10 kW, 380 V
R_r (Ω)	Resist. rotor	1.0	10.0	1.241	Rotor $\leq R_s$
L_s (H)	Induct. estátor	0.10	0.60	0.207	4 polos, 60 Hz
L_r (H)	Induct. rotor	0.10	0.60	0.207	Simétrico a L_s
L_m (H)	Induct. magnet.	0.10	0.55	0.201	$L_m < \sqrt{L_s L_r}$
J (kg m^2)	Inercia rotor	0.005	0.05	0.020	$J \approx 0.04(P_n/1000)^{0.75}$
B (N m s/rad)	Fricción viscosa	10^{-9}	0.01	0.002	Permite $B \approx 0$

Las resistencias e inductancias se escalan como $\theta_{new} = \theta_{ref} \cdot k_Z$, mientras que el momento de inercia se mantiene en el valor nominal de referencia, dado que su escalado requeriría información geométrica del rotor no disponible en la hoja de datos. Adicionalmente, el valor de L_m se ajusta para garantizar un margen del 3 % respecto al límite de acoplamiento magnético:

$$L_m = \text{mín}\left(L_{m,scaled},\ 0.97\sqrt{L_s L_r}\right)$$

Este procedimiento de inicialización es reproducible utilizando únicamente los datos de placa del fabricante, sin requerir pruebas de laboratorio adicionales, representando el escenario realista de un ingeniero en campo. La sensibilidad de la solución final a este punto de partida constituye, precisamente, la limitación estructural que motiva el uso de métodos de búsqueda global como PSO y el método híbrido propuesto.

2.1.4 Suposiciones y limitaciones de modelos lineales

Los modelos tradicionales presentan limitaciones importantes al representar con precisión el comportamiento real de los motores, especialmente en el régimen dinámico o condiciones de operación extremas.

1. **Saturación magnética:** Los modelos lineales asumen una relación proporcional entre corriente y flujo magnético, ignorando la saturación del núcleo ferromagnético

que ocurre a altas densidades de flujo. Esto puede generar errores significativos en la estimación de par y rendimiento, como se modela en la ecuación 15:

$$\psi = \begin{cases} L\, i, & \text{si } |i| \leq i_{\text{sat}} \\ L\, i_{\text{sat}} + L_{\text{sat}}(i - i_{\text{sat}}), & \text{si } |i| > i_{\text{sat}} \end{cases} \tag{15}$$

donde ψ es el flujo magnético, i es la corriente, i_{sat} es la corriente de saturación, L es la inductancia en la región lineal y L_{sat} es la inductancia en la región saturada.

2. **Pérdidas dependientes de frecuencia:** El capturar fenómenos que dependen de la frecuencia como la histéresis magnética y corrientes parásitas es una limitante de los modelos lineales, los cuales aproximan dichos efectos con valores constantes, introduciendo errores en aplicaciones donde la velocidad varía (Toliyat and Kliman, 2004).

3. **Efectos térmicos:** La variación térmica propia de la naturaleza de operación de los motores tiende a modificar de manera significativa parámetros de suma importancia como la resistencia y la inductancia, efectos que rara vez se contemplan en modelos convencionales (Gundabattini et al., 2021).

4. **Armónicos espaciales y temporales:** Una distribución no sinusoidal del flujo magnético en la estructura del motor y la presencia de armónicos en las corrientes eléctricas introducen un comportamiento dinámico que los modelos lineales no pueden reproducir de manera adecuada (Bozek et al., 2023).

Dichas limitaciones sirven como motivación en la búsqueda de un enfoque alternativo que permita capturar con mayor fidelidad la complejidad del comportamiento electromecánico del sistema, especialmente en aplicaciones donde se requiera una precisión crítica.

2.2 Optimización en Sistemas de Ingeniería

La complejidad del problema de identificación paramétrica radica en las no linealidades del motor, la multimodalidad de la función objetivo (múltiples mínimos locales) y el acoplamiento entre variables.

2.2.1 Conceptos generales de optimización

La optimización busca minimizar (o maximizar) una función objetivo, la cual está sujeta a un conjunto de restricciones de igualdad y desigualdad, cómo se formula en la ecuación (16):

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) \qquad \text{sujeto a:} \quad \begin{cases} g_i(\mathbf{x}) \leq 0, & i = 1, 2, \ldots, m \\ h_j(\mathbf{x}) = 0, & j = 1, 2, \ldots, p \end{cases} \tag{16}$$

Donde:

- $f(\mathbf{x})$ Es la función objetivo a minimizar.
- $\mathbf{x}$ Es el vector de variables de decisión.
- $g_i(\mathbf{x})$ Son las restricciones de desigualdad.
- $h_j(\mathbf{x})$ Son las restricciones de igualdad.

En la parte del modelado de un motor de inducción, la función objetivo típicamente busca minimizar discrepancias entre predicciones del modelo y mediciones experimentales como se observa en la ecuación (17):

$$f(\mathbf{p}) = \sum_{k=1}^{N} \sum_{j=1}^{M} w_j \left(y_j^{\text{medido}}(k) - y_j^{\text{modelo}}(k, \mathbf{p}) \right)^2 \tag{17}$$

Donde:

- $\mathbf{p}$ Es el vector de parámetros a optimizar.
- N Es el número de puntos de datos.

- M Es el número de variables medidas.
- w_j Son pesos que indican la importancia relativa de cada variable.
- $y_j^{\text{medido}}(k)$ Es el valor medido de la variable j en el punto k.
- $y_j^{\text{modelo}}(k, \mathbf{p})$ Es el valor predicho por el modelo con parámetros $\mathbf{p}$.

La complejidad inherente a este tipo de problemas radica en múltiples factores: no linealidades en las ecuaciones que rigen el comportamiento del motor, multimodalidad de la función objetivo (múltiples mínimos locales), acoplamiento entre variables y la presencia de restricciones tanto explícitas como implícitas derivadas de principios físicos fundamentales.

2.2.2 Mínimos cuadrados no lineales: LM y Trust Region Reflective

La identificación paramétrica del modelo *d-q* es posible formularla como un problema de mínimos cuadrados no lineales, tomando un vector de mediciones experimentales $\mathbf{y} \in \mathbb{R}^n$ y un modelo no lineal $\mathbf{F}(\boldsymbol{\theta})$ el cual predice las salidas del sistema, buscando el vector de parámetros $\boldsymbol{\theta}^*$ que minimiza la suma de residuos cuadráticos:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \ \frac{1}{2}\|\mathbf{r}(\boldsymbol{\theta})\|^2, \qquad \mathbf{r}(\boldsymbol{\theta}) = \mathbf{y} - \mathbf{F}(\boldsymbol{\theta}) \tag{18}$$

El algoritmo de Levenberg–Marquardt, en la implementación clásica documentada por Moré (Moré, 1978), resuelve el problema mediante una actualización iterativa del vector de parámetros que se interpola entre el método de descenso de gradiente y el método de Gauss–Newton. La dirección de actualización $\Delta\boldsymbol{\theta}$ en cada iteración se obtiene resolviendo el siguiente sistema:

$$\big(\mathbf{J}^T\mathbf{J} + \lambda\mathbf{D}\big)\Delta\boldsymbol{\theta} = -\mathbf{J}^T\mathbf{r} \tag{19}$$

donde $\mathbf{J} \in \mathbb{R}^{n\times p}$ es la matriz jacobiana de residuos ($J_{ij} = \partial r_i/\partial\theta_j$), $\lambda > 0$ es el parámetro de amortiguamiento que se ajusta de manera adaptativa, y $\mathbf{D}$ es la matriz diagonal de escalado.

La variante **Trust Region Reflective** (Branch et al., 1999), implementada en la presente investigación mediante `scipy.optimize.least_squares`, extiende LM incorporando restricciones de caja $\boldsymbol{\theta}_{lb} \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_{ub}$ de manera nativa. A diferencia de actualizar de manera incondicional la dirección $\Delta\boldsymbol{\theta}$, la actualización se restringe a una región de confianza de radio Δ:

$$\Delta\boldsymbol{\theta} = \arg \min_{\substack{\|\mathbf{p}\| \leq \Delta, \\ \boldsymbol{\theta}+\mathbf{p} \in [\boldsymbol{\theta}_{lb}, \boldsymbol{\theta}_{ub}]}} \mathbf{r}^T\mathbf{J}\mathbf{p} + \frac{1}{2}\mathbf{p}^T\mathbf{J}^T\mathbf{J}\mathbf{p} \tag{20}$$

El radio Δ se actualiza de acuerdo a la razón entre la reducción real y la reducción predicha por el modelo cuadrático local, expandiéndose o contrayéndose según la calidad de la aproximación.

El punto inicial $\boldsymbol{\theta}_0$ se deriva a partir de los datos de la placa del motor mediante las relaciones de semejanza eléctrica documentadas en Krishnan (Krishnan, 2001), las cuales escalan los parámetros de un motor de referencia al motor bajo prueba mediante el factor:

$$k_Z = \left(\frac{V_{\text{new}}}{V_{\text{ref}}}\right)^2 \cdot \frac{P_{\text{ref}}}{P_{\text{new}}} = \left(\frac{380}{220}\right)^2 \cdot \frac{2240}{5500} = 1.215 \tag{21}$$

Dicho procedimiento es sistemático y reproducible, pero la calidad de la solución final queda condicionada a la cercanía de $\boldsymbol{\theta}_0$ al mínimo global, lo cual constituye la limitación estructural que los algoritmos de búsqueda global superan.

Como alternativa a los métodos basados en gradiente, Véliz-Tejo et al. (2022) proponen un sistema de referencia adaptativo normalizado (N-MRAS) que estima tanto los parámetros eléctricos como los mecánicos del motor de inducción, validado experimentalmente en banco de pruebas. Dicho enfoque ilustra la necesidad de métodos que superen la sensibilidad al punto inicial que caracteriza a los métodos basados en gradiente.

2.3 Algoritmos Bioinspirados

Los algoritmos bioinspirados constituyen una clase en los métodos de optimización metaheurística, donde se emulan los procesos y comportamientos observados en sistemas biológicos naturales (Yang, 2021). Dichos algoritmos surgieron como alternativas para abordar problemas de optimización complejos donde los métodos tradicionales tienden a no tener un rendimiento aceptable, como serían problemas donde la misma física del modelo sea no lineal.

Se definen estos algoritmos como procedimientos computacionales que abstraen y formalizan principios fundamentales observados en sistemas naturales para resolver problemas complejos de optimización. Sus características distintivas incluyen:

- **Auto-organización:** Emergencia de comportamientos y estructuras complejas a partir de interacciones locales simples, sin control centralizado.
- **Inteligencia colectiva:** Capacidad de múltiples agentes simples para exhibir comportamiento coordinado que supera las capacidades individuales.
- **Adaptabilidad:** Modificación dinámica de comportamientos en respuesta a cambios en el entorno o en las condiciones del problema.
- **Balance exploración-explotación:** Equilibrio dinámico entre búsqueda global en regiones inexploradas (diversificación) y refinamiento local en regiones prometedoras (intensificación).

2.3.1 Algoritmos Genéticos (GA)

Los Algoritmos Genéticos, cuyas bases conceptuales fueron establecidas por Holland y refinadas en revisiones comprehensivas recientes como Katoch et al. (Katoch et al., 2021), se inspiran en principios de selección natural y evolución biológica. Estos algoritmos operan sobre poblaciones de soluciones candidatas codificadas como “cromosomas”, aplicando iterativamente los siguientes operadores genéticos:

- **Selección:** Identificación probabilística de individuos con mayor aptitud para reproducción.
- **Cruce:** Intercambio de información genética entre soluciones seleccionadas.
- **Mutación:** Modificaciones aleatorias para mantener diversidad genética y explorar nuevas regiones.

Como se observa en la ecuación (22), la representación cromosómica queda definida como:

$$\mathbf{x} = [x_1, x_2, \ldots, x_n] \tag{22}$$

donde $\mathbf{x}$ representa el vector de variables de decisión del problema. Para la selección de individuos se utiliza el operador de selección por ruleta, el cual se calcula de acuerdo a la ecuación (23):

$$P(x_i) = \frac{f(x_i)}{\sum_{j=1}^{N_p} f(x_j)} \tag{23}$$

El operador de cruce permite el intercambio de información genética entre soluciones seleccionadas. Como se muestra en la ecuación (24), el cruce de un punto queda definido como:

$$\begin{aligned} \text{Padres:} \quad & [x_1, x_2, \ldots, x_k \mid x_{k+1}, \ldots, x_n] \quad \text{y} \quad [y_1, y_2, \ldots, y_k \mid y_{k+1}, \ldots, y_n] \\ \text{Hijos:} \quad & [x_1, x_2, \ldots, x_k \mid y_{k+1}, \ldots, y_n] \quad \text{y} \quad [y_1, y_2, \ldots, y_k \mid x_{k+1}, \ldots, x_n] \end{aligned} \tag{24}$$

Finalmente, para mantener la diversidad genética, se aplica el operador de mutación según se establece en la ecuación (25):

$$x_i' = \begin{cases} x_i + \Delta, & \text{con probabilidad } p_m \\ x_i, & \text{con probabilidad } 1 - p_m \end{cases} \tag{25}$$

Función de aptitud normalizada: para garantizar que todos los individuos tengan probabilidades de selección apropiadas, la función de aptitud se normaliza respecto a la suma total de aptitudes de la población, como se define en la ecuación (23).

2.3.2 Optimización por Enjambre de Partículas (PSO)

La Optimización por Enjambre de Partículas, propuesta por Kennedy y Eberhart (Kennedy and Eberhart, 1995), se inspira en el comportamiento social de bandadas de aves o bancos de peces. Cada solución candidata (partícula) se mueve a través del espacio de búsqueda ajustando su velocidad basada en:

- Su mejor posición histórica (componente cognitivo).
- La mejor posición global o de su vecindario (componente social).

Este mecanismo simple pero efectivo permite un balance dinámico entre exploración global y convergencia local.

El algoritmo opera mediante dos ecuaciones fundamentales. La actualización de velocidad se calcula de acuerdo a la ecuación (26):

$$v_i^{t+1} = w\, v_i^t + c_1 r_1 \big(\text{pbest}_i - x_i^t\big) + c_2 r_2 \big(\text{gbest} - x_i^t\big) \tag{26}$$

Posteriormente, la actualización de posición se realiza según se establece en la ecuación (27):

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{27}$$

Donde:

- v_i^t Es la velocidad de la partícula (i) en la iteración (t).

- x_i^t Es la posición de la partícula (i) en la iteración (t).
- pbest_i Es la mejor posición personal encontrada por la partícula (i).
- gbest Es la mejor posición global encontrada por el enjambre.
- w Es el factor de inercia.
- c_1, c_2 Son coeficientes de aceleración.
- r_1, r_2 Son números aleatorios uniformes en $([0, 1])$.

PSO presenta ventajas en términos de implementación sencilla, pocos parámetros de ajuste y convergencia relativamente rápida.

Factor de inercia dinámico: Para mejorar la convergencia, el factor de inercia disminuye linealmente durante la ejecución:

$$w(t) = w_{\text{máx}} - \frac{(w_{\text{máx}} - w_{\text{mín}}) \times t}{t_{\text{máx}}} \tag{28}$$

Este mecanismo favorece la exploración global al inicio (w alto) y la explotación local al final (w bajo).

Criterio de convergencia: El algoritmo se detiene cuando la mejora en la mejor solución global es insignificante:

$$\left|\text{gbest}(t) - \text{gbest}(t-1)\right| < \varepsilon \tag{29}$$

donde ε es la tolerancia de convergencia, típicamente 10^{-6} para problemas de identificación paramétrica.

En el contexto específico de motores de inducción, Vukašinović et al. (2023) proponen una variante de PSO de dos etapas para la estimación de parámetros del motor de jaula de ardilla, separando la búsqueda de parámetros en operación nominal de la búsqueda de parámetros de rotor en arranque, con error absoluto medio menor al 5 % frente a datos de fabricante. Dicha arquitectura de dos etapas comparte la motivación del método híbrido

propuesto en la presente investigación, aunque difiere en el algoritmo de refinamiento empleado.

2.3.3 Evolución Diferencial (DE)

Propuesta por Storn y Price (Storn and Price, 1997), la Evolución Diferencial (DE) es un algoritmo de optimización estocástico el cual basa su población en operar directamente sobre vectores de números reales, sin tener que requerir una codificación binaria ni operadores de cruce genéticos convencionales. El mecanismo de búsqueda se fundamenta en la generación de vectores mediante perturbaciones diferenciales entre individuos poblacionales. Dicho mecanismo confiere una capacidad de exploración global robusta ante funciones objetivo multimodales y no diferenciables.

Tomando una población de N_p vectores $\mathbf{x}_{i,g}$, donde $i = 1, \ldots, N_p$ es el índice del individuo y g la generación actual, DE aplica en cada iteración tres operadores: mutación, cruce y selección.

Mutación: Para cada individuo objetivo $\mathbf{x}_{i,g}$, se crea un vector $\mathbf{v}_{i,g}$ a partir del mejor individuo de la generación actual $\mathbf{x}_{\text{best},g}$ y de la diferencia entre dos individuos seleccionados aleatoriamente $\mathbf{x}_{r_1,g}$ y $\mathbf{x}_{r_2,g}$, con $r_1 \neq r_2 \neq i$. Dicha estrategia se denomina `DE/best/1/bin` y se define como:

$$\mathbf{v}_{i,g} = \mathbf{x}_{\text{best},g} + F \cdot \left(\mathbf{x}_{r_1,g} - \mathbf{x}_{r_2,g}\right) \tag{30}$$

donde $F \in (0, 2]$ es el factor de escala de mutación, el cual controla la amplitud de las perturbaciones diferenciales.

Cruce: Para incrementar la diversidad en los vectores de prueba, se aplica un operador de cruce binomial entre el vector mutante $\mathbf{v}_{i,g}$ y el vector objetivo $\mathbf{x}_{i,g}$, generando el vector de prueba $\mathbf{u}_{i,g}$:

$$u_{i,g}^{(j)} = \begin{cases} v_{i,g}^{(j)}, & \text{si } \text{rand}_j \leq C_r \text{ o } j = j_{\text{rand}} \\ x_{i,g}^{(j)}, & \text{en otro caso} \end{cases} \tag{31}$$

donde $C_r \in [0, 1]$ se indica como la probabilidad de cruce, rand_j es un número aleatorio uniforme entre $[0, 1]$ el cual se genera para cada componente j, y j_{rand} es un índice aleatorio que garantiza que el vector de prueba difiera del objetivo en al menos un componente.

Selección: La actualización de la población se realiza mediante selección codiciosa, la cual, a diferencia de otros métodos que puede ser probabilística, en este caso solo el mejor sobrevive. El vector de prueba reemplaza al objetivo únicamente si mejora el valor de la función objetivo:

$$\mathbf{x}_{i,g+1} = \begin{cases} \mathbf{u}_{i,g}, & \text{si } f(\mathbf{u}_{i,g}) \leq f(\mathbf{x}_{i,g}) \\ \mathbf{x}_{i,g}, & \text{en otro caso} \end{cases} \tag{32}$$

La estrategia `DE/best/1/bin`, empleada en la presente investigación, favorece la convergencia hacia las regiones prometedoras del espacio de búsqueda al orientar la mutación de los candidatos desde el mejor individuo conocido, lo cual permite mantener simultáneamente la diversidad poblacional mediante perturbaciones diferenciales aleatorias en el proceso.

Trabajos recientes confirman la vigencia de la Evolución Diferencial en la estimación de parámetros de motores de inducción. Castoldi et al. (2025) presentan un método de estimación fuera de línea que combina DE con las curvas de desempeño del fabricante, eliminando la necesidad de instrumentación adicional y validando el procedimiento bajo el estándar IEEE 112, lo cual es consistente con el enfoque de bajo presupuesto de evaluaciones adoptado en la presente metodología.

2.3.4 Optimización por Forrajeo Bacteriano (BFO)

El algoritmo Optimización por Forrajeo Bacteriano (BFO), propuesto por Passino (Passino, 2002), simula el comportamiento de búsqueda de nutrientes en bacterias *E. coli* a través

de quimiotaxis, reproducción y eliminación. Su capacidad para mantener la diversidad y escapar de óptimos locales lo hace relevante como antecedente en identificación paramétrica de máquinas eléctricas (Sakthivel and Subramanian, 2012). Aunque su implementación no forma parte de la metodologia propuesta en la presente investigacion, se documenta unicamente como referencia historica dada su importancia en el campo de los algoritmos bioinspirados aplicados a maquinas electricas.

2.3.5 Criterios de convergencia y parada

El criterio de diversidad poblacional evalúa la convergencia mediante el calculo de la desviación estándar de las funciones objetivo: :

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(f_i - f_{\text{avg}})^2} \tag{33}$$

Cuando $\sigma < \sigma_{\text{mín}}$, el algoritmo se considera convergido.

2.3.6 Hibridación secuencial: DE + L-BFGS-B

La metodología híbrida propuesta combina de manera secuencial la capacidad de exploración global de la Evolución Diferencial con el refinamiento local del método L-BFGS-B (Limited-memory Broyden–Fletcher–Goldfarb–Shanno). Dicha estrategia de hibridación secuencial explota el potencial de ambos enfoques: DE identifica la región de atracción del mínimo global en el espacio de parámetros, mientras que el uso de L-BFGS-B logra converger hacia ese mínimo aprovechando la información brindada por DE.

En la primera fase, DE opera durante $G_{\text{máx}}$ generaciones sobre una población de N_p individuos, obteniendo el vector solución intermedia $\boldsymbol{\theta}^*_{DE}$:

$$\boldsymbol{\theta}^*_{DE} = \arg \min_{\boldsymbol{\theta} \in [\boldsymbol{\theta}_{lb}, \boldsymbol{\theta}_{ub}]} f(\boldsymbol{\theta}) \qquad \text{(fase global)} \tag{34}$$

En la segunda fase, $\boldsymbol{\theta}^*_{DE}$ se utiliza como punto inicial del refinamiento local mediante L-BFGS-B (Byrd et al., 1995). Dicho método aproxima el inverso del hessiano de la función objetivo mediante un número limitado de pares de vectores los cuales se encuentran almacenados en memoria; la dirección de búsqueda se actualiza como:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \alpha_k \, \mathbf{H}_k^{-1} \nabla f(\boldsymbol{\theta}_k) \tag{35}$$

donde $\mathbf{H}_k^{-1}$ es la aproximación de memoria limitada del hessiano inverso, $\nabla f(\boldsymbol{\theta}_k)$ se refiere al gradiente de la función objetivo en la iteración k, y α_k es el tamaño de paso el cual se determina por búsqueda lineal con condiciones de Wolfe. La solución final de la hibridación es:

$$\boldsymbol{\theta}^*_{\text{Hyb}} = \arg \min_{\boldsymbol{\theta} \in [\boldsymbol{\theta}_{lb}, \boldsymbol{\theta}_{ub}]} f(\boldsymbol{\theta}), \qquad \boldsymbol{\theta}_0 = \boldsymbol{\theta}^*_{DE} \qquad \text{(fase local)} \tag{36}$$

La ventaja que presenta este esquema radica en que DE, al no depender de gradientes ni de un punto inicial específico, evita lo que se conoce como problema de convergencia prematura a mínimos locales, lo que caracteriza principalmente a los métodos basados en gradientes cuando se aplican directamente al problema de identificación. De manera secuencial al aplicar el refinamiento por L-BFGS-B, se garantiza que la solución satisfaga con precisión las condiciones de optimalidad de primer orden, lo que logra una precisión final que DE por sí sola no alcanzaría con el mismo presupuesto de evaluaciones con ambos métodos.

La tendencia hacia métodos híbridos en la estimación de parámetros de motores se documenta también en trabajos recientes. Santana et al. (2025) combinan el algoritmo Sine Cosine (SCA) con PSO para la estimación rápida sobre un modelo estático, con validación posterior sobre el modelo dinámico, reportando que la hibridación reduce el tiempo de cómputo sin sacrificar precisión en estado estacionario. Esta estrategia refuerza el argumento de que la hibridación secuencial exploración global – refinamiento local representa el paradigma más eficiente para la identificación paramétrica de motores de inducción bajo restricciones de

presupuesto de evaluaciones.

2.4 Criterios de Evaluación

Para evaluar el desempeño de cualquier método de optimización, es indispensable contar con métricas las cuales cuantifiquen una relación entre las predicciones y el comportamiento real. Las principales métricas utilizadas incluyen:

Error Cuadrático Medio (ECM): Cuantifica la discrepancia promedio cuadrática entre valores observados y predichos, penalizando especialmente errores grandes. Se utiliza esta métrica como criterio primario para la evaluación comparativa de algoritmos como se observa en la ecuación (37):

$$ECM = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{37}$$

Coeficiente de Determinación (R^2): Mide la proporción de varianza explicada por el modelo, con valores cercanos a 1 indicando alto poder predictivo. Particularmente útil para evaluar la captura de comportamientos dinámicos complejos según se establece en la ecuación (38):

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} \tag{38}$$

Error Porcentual Absoluto Medio (MAPE): Proporciona una medida de error normalizada en términos porcentuales, facilitando comparaciones entre diferentes condiciones operativas, como se define en la ecuación (39):

$$MAPE = \frac{100\,\%}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{39}$$

Error Máximo Absoluto (MAE): Identifica la mayor discrepancia puntual, crítico para aplicaciones donde limitaciones de seguridad o rendimiento dependen de precisión en condiciones extremas, de acuerdo a la ecuación (40):

$$MAE = \max_{i=1,\ldots,N} |y_i - \hat{y}_i| \tag{40}$$

Donde:

- y_i son los valores observados.
- $\hat{y}_i$ son los valores predichos por el modelo.
- $\bar{y}$ es la media de los valores observados.
- N es el número de observaciones.

Las cuatro métricas anteriores son complementarias y se emplean de manera conjunta en la presente investigación. El ECM actúa como función objetivo primaria durante la identificación paramétrica, ya que su diferenciabilidad lo hace compatible con el refinamiento local de L-BFGS-B y su penalización cuadrática favorece la convergencia hacia soluciones con error distribuido uniformemente. El R^2 constituye la métrica de comparación principal entre métodos, dado que es invariante a la escala de la señal y permite interpretar directamente qué fracción de la varianza de la corriente experimental es explicada por el modelo identificado. El MAPE complementa al R^2 en la evaluación de errores relativos por punto de operación, siendo especialmente útil al comparar condiciones de carga distintas donde la amplitud de la corriente varía. Finalmente, el MAE permite identificar instantes específicos de alto error, relevantes para detectar limitaciones del modelo d-q en el régimen transitorio de arranque donde la saturación magnética no está capturada por parámetros constantes. La combinación de estas métricas sobre los ejes de identificación (α) y de validación cruzada (β) permite una evaluación integral tanto de la precisión como de la capacidad de generalización del modelo identificado.

2.5 Manifestación de fallas mecánicas en el modelo d-q

Las fallas mecánicas de desalineamiento y aflojamiento representan dos de las condiciones más frecuentes en motores de inducción industriales. Su detección mediante señales eléctricas ha ganado atención dado el carácter no invasivo de la medición de corriente de estátor (Niu et al., 2023; Hamani et al., 2025).

2.5.1 Desalineamiento

El desalineamiento ocurre cuando el eje del motor y el eje de la carga no comparten la misma línea de acción, generando fuerzas radiales y axiales cíclicas sobre el rotor. Dichas fuerzas podrían producir una variación periódica del entrehierro efectivo que modificaría la distribución del flujo magnético entre estátor y rotor, un efecto conocido del entrehierro no uniforme sobre las inductancias del motor (Hamani et al., 2025). Como hipótesis de modelado, desde la perspectiva del modelo d-q con parámetros concentrados, esta variación se manifestaría como un incremento en la inductancia de magnetización efectiva L_m y en la inductancia de estátor L_s identificadas durante el régimen cuasi estacionario. Esta hipótesis se retoma y contrasta con los resultados experimentales en la Sección 4.5.

2.5.2 Aflojamiento

El aflojamiento de la base del motor (*foot looseness*) genera vibraciones mecánicas de baja frecuencia que se transmiten al rotor como perturbaciones de par, potencialmente detectables mediante análisis de firma de corriente (Niu et al., 2023). En el contexto de identificación paramétrica, la looseness incrementa las pérdidas disipativas del sistema mecánico, lo que se refleja en un mayor valor del coeficiente de fricción viscosa identificado B. El presente trabajo confirmó este comportamiento, con B mayor en aflojamiento ($0.006\,\mathrm{N\cdot m\cdot s/rad}$) que en la condición sana ($0.005\,\mathrm{N\cdot m\cdot s/rad}$), de manera consistente con las perturbaciones de rozamiento generadas por la condición de la falla mecánica.

La capacidad del vector $\boldsymbol{\theta}^*$ identificado para capturar estos efectos de manera implícita,

sin modificar la estructura del modelo d-q, constituye la base de la función diagnóstica indirecta propuesta en la presente investigación y discutida en la Sección 4.5.

III. Metodología

La metodología propuesta se estructura en seis etapas secuenciales, diseñadas para garantizar condiciones de comparación equivalentes entre los tres métodos de identificación evaluados. La Figura 3.1 presenta el flujo general del proceso. En la primera etapa se adquieren y preprocesan las señales experimentales de voltaje y corriente provenientes de la base de datos de Battulga et al. (Battulga et al., 2024), aplicando la transformada de Clarke para obtener las componentes α-β. La segunda etapa formula el modelo d-q en referencia estacionaria y define la función objetivo basada en el ECM entre la corriente simulada y la corriente experimental medida. La tercera etapa selecciona los tres métodos de identificación a partir de una exploración preliminar sobre señal sintética con parámetros verdaderos conocidos. La cuarta etapa realiza la implementación computacional en Python con condiciones de reproducibilidad controladas. La quinta etapa evalúa y compara los métodos mediante métricas de error en los ejes de identificación (α) y de validación cruzada (β). Finalmente, la sexta etapa cuantifica la mejora porcentual relativa del método propuesto frente a cada referencia y consolida los resultados en una base de datos estructurada para el análisis estadístico posterior.

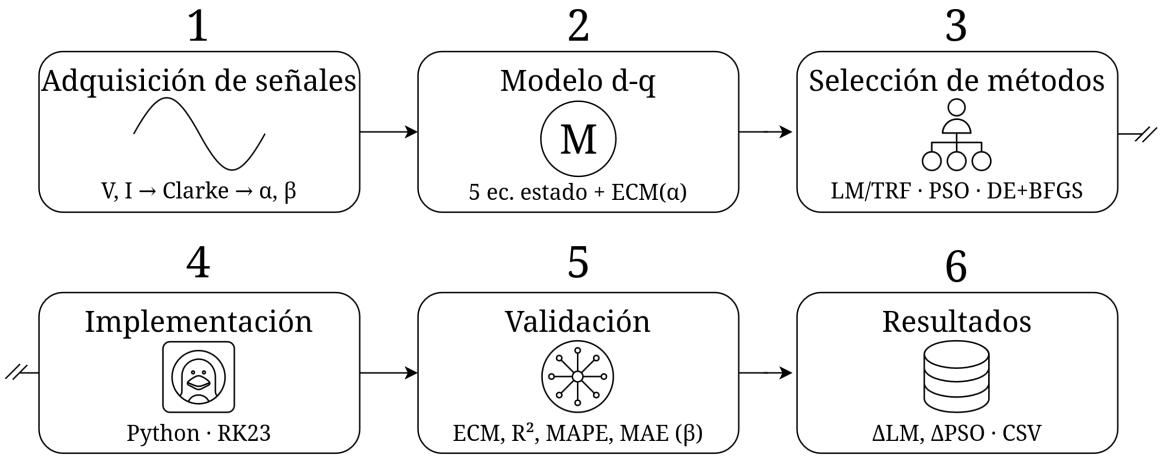


Figura 3.1: Esquema general de las etapas metodológicas

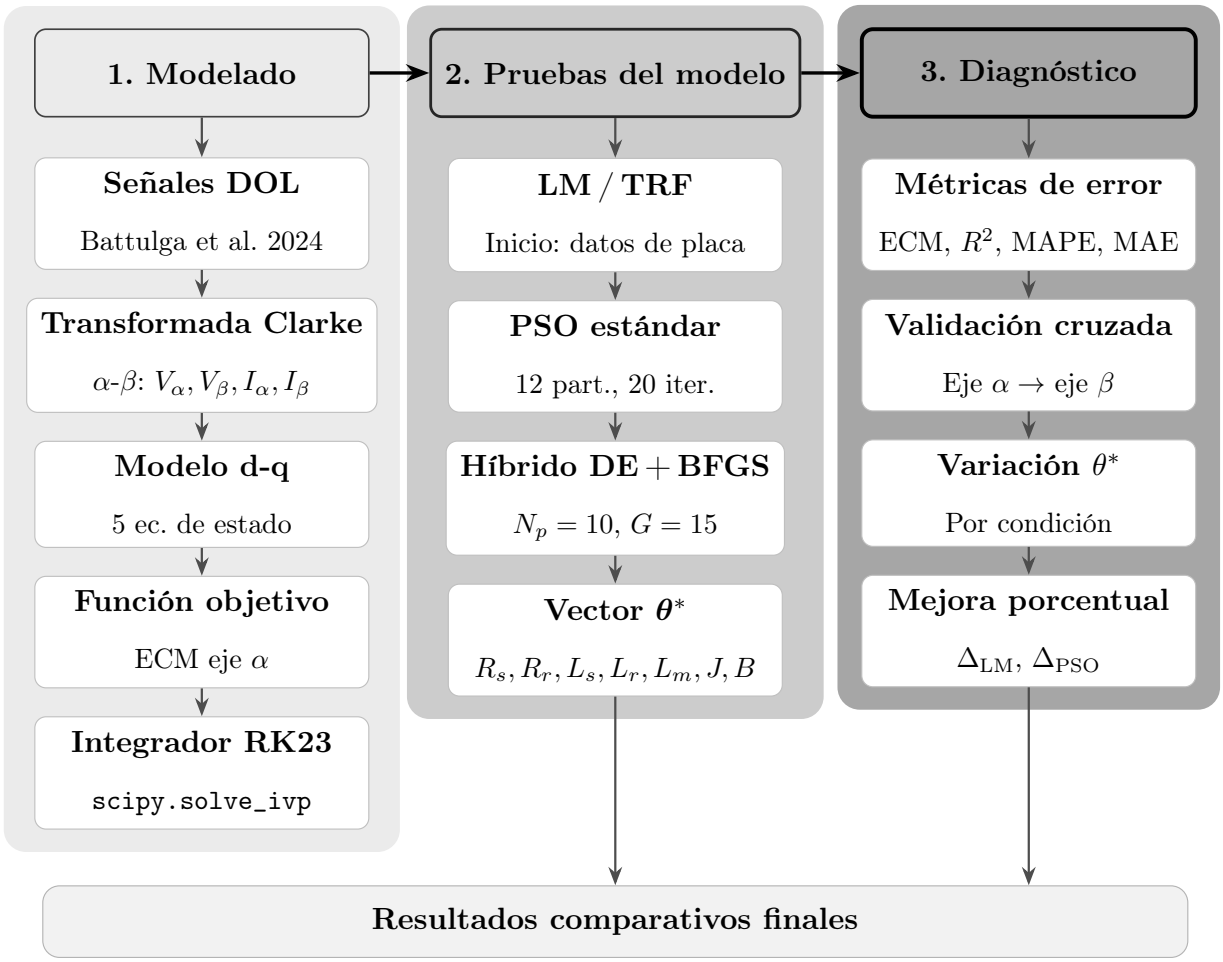


Figura 3.2: Flujo detallado de la implementación computacional: modelado del sistema, identificación paramétrica mediante los tres métodos evaluados, y diagnóstico basado en el vector $\boldsymbol{\theta}^*$.

La Figura 3.2 desglosa la Etapa 4 (implementación computacional) de la Figura 3.1 en sus tres componentes funcionales. El grupo *Modelado* integra las señales experimentales mediante la transformada de Clarke y las propaga a través del sistema de cinco ecuaciones de estado del modelo d-q hasta obtener la corriente simulada. El grupo *Pruebas del modelo* aplica los tres métodos de identificación (LM/TRF, PSO estándar e Híbrido DE + L-BFGS-B) sobre dicho modelo para obtener el vector de parámetros $\boldsymbol{\theta}^*$. Finalmente, el grupo *Diagnóstico* evalúa este vector mediante métricas de error, validación cruzada y análisis de variación paramétrica, consolidando los resultados comparativos finales de la Etapa 5.

3.1 Etapa 1. Adquisición y preprocesamiento de señales

Los datos experimentales empleados en el presente proyecto provienen de la base de datos de acceso público descrita en Battulga et al. (Battulga et al., 2024), la cual fue adquirida sobre un motor de inducción trifásico de 380 V, 5.5 kW, cuatro polos, 60 Hz, conexión Y. Se utilizaron exclusivamente las señales de voltaje y corriente de fase registradas durante el transitorio de arranque directo en línea (DOL); las señales fueron muestreadas a $f_s = 10{,}240$ Hz con una resolución de 16 bits.

Las condiciones operativas son: motor sano (*Healthy*), desalineamiento (*Misalignment*) y aflojamiento (*Looseness*), con niveles de carga de 0 %, 50 %, 75 % y 100 % de la carga nominal, resultando en un total de 12 registros de arranque, distribuidos uniformemente en cuatro archivos por condición.

Para cada registro se cargan las cuatro señales de forma pareada: el voltaje de línea A-B (`vol1`), el voltaje de línea B-C (`vol2`), la corriente de fase A (`cur1`) y la corriente de fase B (`cur2`). Antes de cualquier procesamiento, cada señal se escala por un factor de 100 para recuperar las unidades físicas en volts y amperes respectivamente, y se sustrae el valor medio para eliminar el componente de corriente continua, como se documenta en Battulga et al. (Battulga et al., 2024).

A partir de las cuatro señales adquiridas se obtienen las componentes en referencia estacionaria α-β mediante la **transformada de Clarke**, de acuerdo con las expresiones:

$$V_\alpha = \frac{2V_{ab} + V_{bc}}{3}, \qquad V_\beta = \frac{V_{bc}}{\sqrt{3}} \tag{41}$$

$$I_\alpha = I_a, \qquad I_\beta = \frac{I_a + 2I_b}{\sqrt{3}} \tag{42}$$

donde V_{ab} y V_{bc} son los voltajes de línea medidos, e I_a, I_b son las corrientes de fase A y B respectivamente. Esta transformación proyecta el sistema trifásico sobre dos ejes ortogonales estacionarios, lo que proporciona las señales de excitación V_α y V_β para el modelo *d*-*q* y las

corrientes de referencia I_α e I_β para la evaluación de la función objetivo y la validación cruzada. Para cada registro se extrae un segmento de 1.5 segundos desde el instante inicial del archivo, lo que corresponde a $n = 15{,}360$ muestras; en dicho intervalo se captura el transitorio completo del arranque DOL.

3.2 Etapa 2. Formulación del modelo d-q y función objetivo

El comportamiento dinámico del motor se puede representar mediante el modelo *d-q* (α-β), el cual se aplica de manera uniforme a las tres condiciones operativas (sano, desalineamiento y aflojamiento). Bajo este enfoque, el comportamiento electromecánico asociado a cada condición queda capturado implícitamente en el vector de parámetros identificados $\boldsymbol{\theta}^* = [R_s, R_r, L_s, L_r, L_m, J, B]^T$, el conjunto completo de los siete parámetros eléctricos y mecánicos del modelo *d-q*, y no un parámetro aislado, siendo de interés la varianza estadística de estos parámetros entre condiciones.

El sistema de ecuaciones de estado implementado incorpora una **resistencia efectiva de estátor** $R_{s,\text{eff}}$, definida como:

$$R_{s,\text{eff}} = R_s + \frac{L_m^2 R_r}{L_r^2} \tag{43}$$

$R_{s,\text{eff}}$ agrupa la caída resistiva del estátor con la contribución resistiva del rotor referida al estátor a través del acoplamiento magnético. Las cinco ecuaciones de estado del modelo son:

$$\frac{di_{ds}}{dt} = \frac{1}{\sigma L_s}\left(V_\alpha - R_{s,\text{eff}}\, i_{ds} + \frac{L_m R_r}{L_r^2}\,\varphi_{r\alpha} + P\omega_r \frac{L_m}{L_r}\,\varphi_{r\beta}\right) \tag{44}$$

$$\frac{di_{qs}}{dt} = \frac{1}{\sigma L_s}\left(V_\beta - R_{s,\text{eff}}\, i_{qs} + \frac{L_m R_r}{L_r^2}\,\varphi_{r\beta} - P\omega_r \frac{L_m}{L_r}\,\varphi_{r\alpha}\right) \tag{45}$$

$$\frac{d\varphi_{r\alpha}}{dt} = \frac{L_m R_r}{L_r}\, i_{ds} - \frac{R_r}{L_r}\,\varphi_{r\alpha} - P\omega_r\,\varphi_{r\beta} \tag{46}$$

$$\frac{d\varphi_{r\beta}}{dt} = \frac{L_m R_r}{L_r}\, i_{qs} - \frac{R_r}{L_r}\,\varphi_{r\beta} + P\omega_r\,\varphi_{r\alpha} \tag{47}$$

$$\frac{d\omega_r}{dt} = \frac{1}{J}\big(T_e - T_L - B\omega_r\big) \tag{48}$$

donde $\sigma = 1 - L_m^2/(L_s L_r)$ es el factor de dispersión total, $T_e = \frac{3}{2}P\frac{L_m}{L_r}(\varphi_{r\alpha}\, i_{qs} - \varphi_{r\beta}\, i_{ds})$ es el par electromagnético, $T_L = 0.0005\,\omega_r^2$ es el par de carga cuadrático y $P = 2$ es el número de pares de polos. Los signos de los términos de acoplamiento en los flujos del rotor ($-P\omega_r\varphi_{r\beta}$ en la ecuación de $\varphi_{r\alpha}$ y $+P\omega_r\varphi_{r\alpha}$ en la de $\varphi_{r\beta}$) se verificaron experimentalmente como la combinación que produce rotación positiva del rotor consistente con los datos medidos.

La función objetivo a minimizar es el **Error Cuadrático Medio (ECM)** entre la corriente de estátor simulada en el eje α, $\hat{\imath}_{ds}(t_k;\boldsymbol{\theta})$, obtenida integrando numéricamente el sistema anterior, y la corriente experimental medida $I_\alpha(t_k)$:

$$f(\boldsymbol{\theta}) = \frac{1}{n}\sum_{k=1}^{n}\Big(\hat{\imath}_{ds}(t_k;\boldsymbol{\theta}) - I_\alpha(t_k)\Big)^2 \tag{49}$$

La integración numérica se realiza mediante el método de Runge-Kutta de orden 2-3 (RK23), implementado en `scipy.integrate.solve_ivp`, utilizando condiciones iniciales nulas $\mathbf{x}(0) = \mathbf{0}$ y las señales V_α, V_β obtenidas en la Etapa 1 como excitación del sistema. El espacio de búsqueda queda acotado por restricciones de caja $\boldsymbol{\theta} \in [\boldsymbol{\theta}_{lb}, \boldsymbol{\theta}_{ub}]$ derivadas de rangos físicamente posibles para motores de inducción de baja tensión, con la restricción adicional de acoplamiento magnético $L_m < \sqrt{L_s L_r}$, la cual se verifica en cada evaluación de la función objetivo.

Después de lograr identificar el vector $\boldsymbol{\theta}^*$ que minimiza el ECM en el eje α, se realiza una **validación cruzada** sobre el eje β: utilizando el mismo modelo el cual es integrado con los parámetros identificados usando V_β como excitación, y la corriente simulada resultante $\hat{\imath}_{qs}$ se compara con la corriente experimental I_β mediante las métricas ECM, R^2, MAE y MAPE. Este procedimiento evalúa la capacidad de generalización del modelo identificado sobre un eje de excitación independiente, sin volver a optimizar parámetros.

3.3 Etapa 3. Selección de métodos de identificación

La selección de los tres métodos de identificación implementados en esta investigación se realizó mediante una etapa exploratoria con señales sintéticas generadas a partir de parámetros nominales conocidos. Dicho enfoque permitió evaluar la capacidad de recuperación paramétrica de distintas estrategias de optimización bajo condiciones controladas donde la solución verdadera era conocida.

En dicha etapa se evaluaron métodos basados en gradiente, algoritmos bioinspirados de enjambre, incluyendo PSO y variantes con factor de inercia adaptativo, y estrategias híbridas que combinan exploración global con refinamiento local. Los resultados de esta exploración mostraron que los métodos basados exclusivamente en gradiente presentaban alta sensibilidad al punto inicial, convergiendo frecuentemente a mínimos locales alejados de la solución verdadera. El PSO estándar demostró mayor robustez ante esta limitación, aunque con alta variabilidad entre ejecuciones. Las estrategias híbridas que combinaron un algoritmo de búsqueda global libre de gradientes con un refinamiento local de alta precisión presentaron la mejor recuperación paramétrica en términos tanto de exactitud como de consistencia entre ejecuciones. Con base en estos resultados, se seleccionó la hibridación secuencial de Evolución Diferencial con estrategia `DE/best/1/bin` como fase de exploración global, seguida de refinamiento local usando L-BFGS-B, el PSO estándar como representante del estado del arte en métodos bioinspirados y el LM/TRF con inicialización por datos de placa como método convencional de referencia.

El **método de referencia convencional** corresponde al algoritmo Trust Region Reflective

(LM/TRF), la cual es una variante acotada del clásico Levenberg–Marquardt, implementada mediante `scipy.optimize.least_squares` con tolerancias `ftol = xtol = gtol` $= 10^{-10}$ y un máximo de 500 evaluaciones de la función objetivo (`max_nfev = 500`). Su punto de partida $\boldsymbol{\theta}_0$ se calculó analíticamente desde los datos de placa del motor mediante las relaciones de semejanza eléctrica (Krishnan, 2001); dichos parámetros se escalaron al motor de referencia de 220 V / 2240 W con el factor $k_Z = (380/220)^2 \cdot (2240/5500) = 1.215$, lo que produce $R_s = 2.152\,\Omega$, $R_r = 1.241\,\Omega$, $L_s = L_r = 0.207\,\mathrm{H}$, $L_m = 0.201\,\mathrm{H}$, $J = 0.020\,\mathrm{kg\,m^2}$ y $B = 0.002\,\mathrm{N/rad}$. Dicho procedimiento es reproducible únicamente con los datos de placa del fabricante, lo que permite replicar el método de manera convencional en el campo.

El **método referente al estado del arte** corresponde al algoritmo de Optimización por Enjambre de Partículas (PSO) estándar (Kennedy and Eberhart, 1995), con un factor de inercia fijo de $w = 0.5$ y coeficientes de aceleración de $c_1 = c_2 = 1.5$, operando sobre una población de 12 partículas durante 20 iteraciones, dando un presupuesto total de 240 evaluaciones de la función objetivo por registro.

El **método propuesto** corresponde a la hibridación secuencial DE + L-BFGS-B: Evolución Diferencial con estrategia `DE/best/1/bin` (Storn and Price, 1997) como fase de exploración global, seguida de refinamiento local mediante L-BFGS-B (Byrd et al., 1995). DE opera con una población de 10 individuos durante 15 generaciones; la fase local se inicializa en la mejor solución $\boldsymbol{\theta}^*_{DE}$ encontrada por DE y converge hacia el mínimo local más próximo con precisión de gradiente.

3.4 Etapa 4. Implementación computacional

La implementación se realizó íntegramente utilizando Python 3, empleando las bibliotecas NumPy y SciPy para el cómputo numérico. Los tres métodos comparten la misma función objetivo, el mismo espacio de búsqueda y el mismo conjunto de señales de entrada, lo que garantiza las condiciones de comparación equivalentes. El procesamiento de los 12 registros se ejecutó de manera paralela mediante el uso de `multiprocessing.Pool` con un proceso por núcleo del procesador; este tipo de ejecución permite reducir el tiempo total de cómputo

en hardware convencional. Para asegurar la reproducibilidad exacta de los resultados, se utilizó una semilla aleatoria global `RANDOM_SEED = 42`, con semillas individuales por archivo las cuales se calculan como `seed = 42 + indice_archivo` para PSO y DE.

Para reducir el costo computacional de las evaluaciones de la función objetivo sin comprometer la fidelidad en el modelado, la señal de identificación se construyó mediante submuestreo de la señal completa por un factor de 8, lo que reduce el vector de evaluaciones de 15 360 a 1 920 puntos. El integrador ODE opera con el método Runge-Kutta de orden 2-3 (RK23) con paso máximo $\Delta t_{\text{máx}} = 2 \times 10^{-3}$ s; a su vez el interpolador de voltaje opera sobre la señal completa en todo momento, lo que logra que la excitación del modelado no pierda resolución temporal. Los resultados finales, el vector de parámetros identificado por cada método y las métricas del error en ambos ejes, se guardan en una base de datos en formato CSV con 12 filas y 58 columnas para un análisis estadístico posterior.

La Tabla 3.1 resume la configuración de los tres métodos implementados.

Tabla 3.1: Configuración de los tres métodos de identificación.

Parámetro	**LM/TRF**	**PSO**	**DE+L-BFGS-B**
Punto inicio	Datos de placa	Aleatorio uniforme	Aleatorio (DE)
Par. clave	ftol=xtol=10^{-10}	$w=0.5$, $c_1=c_2=1.5$	$F=0.8$, $C_r=0.9$
	max_eval = 500	12 part., 20 iter.	$N_p=10$, $G=15$
Evaluaciones	≤ 500	240 (12×20)	≈240–300
Semilla	N/A (determ.)	42 + índice	42 + índice
Librería	`least_squares`	NumPy vectorizado	DE propio + `minimize`

3.5 Etapa 5. Validación y análisis comparativo

Para la evaluacion del desempeno de los tres metodos, se calcula para cada registro experimental el ECM entre la corriente simulada en el eje α con los parametros identificacios y la corriente experimental I_α medida. Las metricas de comparacion adicional como R^2, MAPE y MAE se calculan sobre los resultados del CSV para ambos ejes, α (identificación) y

β (validación cruzada). El analisis comparativo se estructura en tres niveles: La comparacion por archivo individual, comparacion por condicion operativa (media $\pm$ desviación estándar del ECM sobre los cuatro archivos de cada condición) y el analisis de la variacion parametrica del vector $\boldsymbol{\theta}^*$ entre condiciones. Todo esto con el objetivo de evaluar la capacidad diagnostica implicita del modelo d-q,α con los parametros identificacios y la corriente experimental I_α medida.

3.6 Etapa 6. Postprocesamiento y generación de resultados

Una vez concluida la etapa de identificación paramétrica, se realiza una etapa de post-procesamiento la cual está orientada a cuantificar la mejora relativa del método Híbrido DE+L-BFGS-B frente a los dos métodos de referencia.

La mejora porcentual del método Híbrido DE+L-BFGS-B respecto a cada referencia se calcula para cada archivo individual mediante las expresiones:

$$\Delta_{PSO} = \frac{ECM_{PSO} - ECM_{\text{Híbrido}}}{ECM_{PSO}} \times 100\,\% \tag{50}$$

$$\Delta_{LM} = \frac{ECM_{LM} - ECM_{\text{Híbrido}}}{ECM_{LM}} \times 100\,\% \tag{51}$$

A diferencia de formulaciones previas en la literatura, que recortan los valores negativos a cero cuando el método de referencia supera al propuesto, en la presente investigación se reportan los valores reales de Δ_{LM} y Δ_{PSO}, incluyendo valores negativos en los casos donde el método Híbrido DE+L-BFGS-B no obtiene la mejor solución, esto con el objetivo de mantener la transparencia en la comparación y reflejar fielmente el comportamiento de cada método en todos los escenarios evaluados.

Los resultados completos, como el vector de parámetros identificados por cada método, métricas del error en ambos ejes y porcentajes de mejora, se consolidan en la base de datos CSV de 12 filas y 58 columnas descrita en la Etapa 4. Dicha base de datos constituye la

fuente primaria para todos los análisis estadísticos y figuras reportadas en la sección de Resultados y Discusión.

3.7 Software, repositorio y reproducibilidad

Los resultados reportados en el Capítulo 4 fueron generados con dos scripts de Python de desarrollo propio. La Figura 3.2 resume el flujo de dicha implementación, desde el modelado del sistema hasta el diagnóstico basado en el vector de parámetros identificados $\boldsymbol{\theta}^*$. Esta sección documenta el entorno de software, los repositorios de código y datos, los fragmentos de código más representativos y el procedimiento paso a paso para reproducir los resultados de manera exacta.

3.7.1 Entorno de software y dependencias

La implementación se realizó en Python 3.9.25 sobre Windows 10 de 64 bits. La Tabla 3.2 lista las bibliotecas requeridas con sus versiones.

Tabla 3.2: Dependencias del entorno de software.

Biblioteca	**Versión**	**Uso principal**
Python	3.9.25	Intérprete base
NumPy	1.24.x	Operaciones vectoriales, PSO
SciPy	1.11.x	ODE (RK23), LM/TRF, DE, L-BFGS-B
Pandas	2.0.x	Exportación CSV de resultados
Matplotlib	3.7.x	Generación de figuras
Seaborn	0.13.x	Diagramas de caja
tqdm	4.66.x	Barra de progreso en procesamiento paralelo

El entorno puede recrearse ejecutando:

Listing 1: Instalación de dependencias.

```
pip install numpy==1.24.4 scipy==1.11.4 pandas==2.0.3 \
            matplotlib==3.7.5 seaborn==0.13.2 tqdm==4.66.4
```

3.7.2 Scripts desarrollados

La implementación se distribuye en dos scripts independientes, cuya correspondencia con las etapas metodológicas se describe a continuación.

`versionFinal.py` Script principal que ejecuta los tres métodos de identificación paramétrica sobre los 12 registros experimentales. La Tabla 3.3 resume cada función y su rol dentro de la metodología.

Tabla 3.3: Funciones del script principal `versionFinal.py`.

Función	Descripción
`_find_base_path()`	Localiza el directorio de datos `EECS/CommonSensor/StartingData/` recorriendo el árbol de carpetas desde el script.
`get_nominal_params()`	Calcula $\boldsymbol{\theta}_0$ mediante semejanza eléctrica desde los datos de placa (Krishnan, 2001).
`load_specific_pair()`	Carga los cuatro archivos `.mat` de un registro, aplica el factor $\times 100$, centra en DC, recorta la ventana 1.6–2.0 s y ejecuta la transformada de Clarke.
`motor_ode()`	Sistema de cinco EDO del modelo d-q con $R_{s,\text{eff}} = R_s + L_m^2 R_r / L_r^2$.
`_simulate()`	Integra `motor_ode` con RK23 ($\Delta t_{\text{máx}} = 2 \times 10^{-3}$ s).
`objective_alpha()`	Calcula el ECM entre $\hat{\imath}_{ds}$ e I_α.
`residuals_alpha()`	Vector de residuos en eje α para LM/TRF.
`run_levenberg_marquardt()`	Método convencional LM/TRF desde $\boldsymbol{\theta}_0$.
`run_simple_pso()`	PSO estándar ($N_p = 12$, $T = 20$, $w = 0.5$, $c_1 = c_2 = 1.5$).
`compute_metrics()`	ECM, R^2, MAE y MAPE para los ejes α (identificación) y β (validación cruzada).
`process_file()`	Unidad de trabajo paralelo: ejecuta los tres métodos sobre un archivo y devuelve el diccionario de resultados.
`main()`	Orquesta `multiprocessing.Pool` y exporta el CSV de 12×58 columnas.

`exploracion_sintetica.py` Script auxiliar que reproduce la Etapa 3 (selección de métodos, Sección 3.3) sobre una señal sintética con parámetros verdaderos conocidos ($R_s = 2.5\,\Omega$, $R_r = 1.8\,\Omega$, $L_s = 0.22\,\text{H}$, $L_r = 0.35\,\text{H}$, $L_m = 0.28\,\text{H}$, $J = 0.035\,\text{kg m}^2$, $B = 0.006\,\text{N m s/rad}$)

y ruido gaussiano $\sigma = 0.8\,\text{A}$. La excitación sintética empleada en este script es una señal bifásica simplificada (V_a, V_b con desfase de $\pi/2$), suficiente para evaluar la capacidad de recuperación paramétrica en esta etapa exploratoria de selección de métodos; la validación real de la metodología se realiza sobre las señales experimentales completas de `versionFinal.py`, con la transformada de Clarke aplicada a los tres registros de fase.

3.7.3 Fragmentos de código representativos

Preprocesamiento y transformada de Clarke.

Listing 2: Centrado DC y transformada de Clarke (`load_specific_pair`).

```
fs = 10_240

# Factor x100: recupera unidades fisicas (V, A)
Vab = _load(vol1_path) * 100;  Vbc = _load(vol2_path) * 100
Ia  = _load(cur1_path) * 100;  Ib  = _load(cur2_path) * 100

# Centrado DC
Vab -= np.mean(Vab);  Vbc -= np.mean(Vbc)
Ia  -= np.mean(Ia);   Ib  -= np.mean(Ib)

# Ventana 1.6-2.0 s (regimen cuasi-estacionario, post-inrush)
idx_start = int(1.60 * fs)   # muestra 16 384
idx_end   = int(2.00 * fs)   # muestra 20 480
Vab = Vab[idx_start:idx_end];  Vbc = Vbc[idx_start:idx_end]
Ia  = Ia [idx_start:idx_end];  Ib  = Ib [idx_start:idx_end]

# Transformada de Clarke (Ecs. 41-42)
V_alpha = (2.0 * Vab + Vbc) / 3.0
V_beta  = Vbc / np.sqrt(3.0)
I_alpha = Ia
I_beta  = (Ia + 2.0 * Ib) / np.sqrt(3.0)
```

Sistema ODE del modelo d-q.

Listing 3: Modelo d-q en referencia estacionaria α-β (`motor_ode`).

```
def motor_ode(t, y, params, vd_interp, vq_interp):
    Rs, Rr, Ls, Lr, Lm, J, B = params
    sigma  = max(1.0 - (Lm**2 / (Ls * Lr)), 1e-4)
    ids, iqs, phird, phirq, wr = y
    P      = 2
    Rs_eff = Rs + (Lm**2 * Rr) / Lr**2          # Ec. (3)

    dids = (1.0/(sigma*Ls)) * (vd_interp(t) - Rs_eff*ids
           + (Lm*Rr/Lr**2)*phird + P*wr*(Lm/Lr)*phirq)  # Ec. (4)
    diqs = (1.0/(sigma*Ls)) * (vq_interp(t) - Rs_eff*iqs
           + (Lm*Rr/Lr**2)*phirq - P*wr*(Lm/Lr)*phird)  # Ec. (5)

    dphird = (Lm*Rr/Lr)*ids - (Rr/Lr)*phird - P*wr*phirq # Ec. (6)
    dphirq = (Lm*Rr/Lr)*iqs - (Rr/Lr)*phirq + P*wr*phird # Ec. (7)

    Te  = 1.5*P*(Lm/Lr)*(phird*iqs - phirq*ids)          # Ec. (9)
    TL  = 0.0005 * wr**2
    dwr = (1.0/J) * (Te - TL - B*wr)                     # Ec.(10)
    return [dids, diqs, dphird, dphirq, dwr]
```

Método Híbrido DE + L-BFGS-B.

Listing 4: Hibridación secuencial DE + L-BFGS-B dentro de `process_file`.

```
# Semilla individual garantiza reproducibilidad exacta
seed = RANDOM_SEED + file_index   # 42, 43, ..., 53

# Fase 1 -- Exploracion global DE/best/1/bin
# Configuracion Tabla 3.1: Np=10, G=15, F=0.8, Cr=0.9
res_de = differential_evolution(
    objective_alpha, BOUNDS,
    args=(t_ds, I_alpha_ds, vd_func, vq_func),
    strategy="best1bin",
    maxiter=15,         # G  = 15 generaciones
    popsize=10,         # Np = 10 individuos
    mutation=0.8,       # F  = 0.8
    recombination=0.9,  # Cr = 0.9
    polish=False,
    seed=seed)

```

```
# Fase 2 -- Refinamiento local L-BFGS-B
res_hybrid = minimize(
    objective_alpha, res_de.x,
    args=(t_ds, I_alpha_ds, vd_func, vq_func),
    method="L-BFGS-B",
    bounds=BOUNDS)

params_hybrid = res_hybrid.x
mse_hybrid    = float(res_hybrid.fun)
```

Validación cruzada.

Listing 5: Cálculo de ECM, R^2, MAE y MAPE en ambos ejes (`compute_metrics`).

```
def compute_metrics(params, t_eval, I_alpha_ref,
                    I_beta_ref, vd_func, vq_func):
    sol = _simulate(params, t_eval, vd_func, vq_func)
    if sol is None:
        return None

    metrics = {}
    for axis, sim, ref in [("alpha", sol.y[0], I_alpha_ref),
                           ("beta",  sol.y[1], I_beta_ref)]:
        n    = min(len(sim), len(ref))
        mse = float(np.mean((sim[:n] - ref[:n])**2))
        mae = float(np.mean(np.abs(sim[:n] - ref[:n])))
        ss_res = np.sum((sim[:n] - ref[:n])**2)
        ss_tot = np.sum((ref[:n] - np.mean(ref[:n]))**2)
        r2     = 1.0 - ss_res/ss_tot if ss_tot > 0 else 0.0
        mask   = np.abs(ref[:n]) > 0.5
        mape   = (float(np.mean(np.abs(
                    (sim[:n][mask]-ref[:n][mask])
                    /ref[:n][mask])))*100
                  if np.any(mask) else 0.0)
        metrics[f"ECM_{axis}"]  = mse
        metrics[f"R2_{axis}"]   = r2
        metrics[f"MAE_{axis}"]  = mae
        metrics[f"MAPE_{axis}"] = mape
    return metrics
```

3.7.4 Estructura de archivos de entrada y salida

Datos de entrada. Los archivos `.mat` deben organizarse con la siguiente estructura relativa al script:

Listing 6: Estructura de directorios de entrada.

```
Datos/EECS/CommonSensor/StartingData/
    Healthy/        (4 registros)
    Looseness/      (4 registros)
    Misalignment/   (4 registros)
```

Archivos de salida. La Tabla 3.4 describe los archivos generados por ambos scripts y su correspondencia con los resultados del Capítulo 4.

Tabla 3.4: Archivos de salida generados por los scripts.

Archivo	Corresponde a
`resultados/resultados_completos.csv`	Base de datos primaria (12×58); fuente de las Tablas 4.7 a 4.6.
`resultados/graficas/control_ecm_alpha.svg`	Fig. 4.7 (diagramas de caja ECM).
`resultados/graficas/validacion_cruzada_r2.svg`	Fig. 4.4 (R^2 por condición).
`resultados_exploracion/Fig_Exploracion_*.pdf`	Figs. 4.1 y 4.2.

3.7.5 Disponibilidad del código y datos

Datos experimentales. Los 12 registros de arranque DOL provienen de la base de datos pública de Battulga et al. (Battulga et al., 2024), disponible en *IEEE Sensors Journal*.[1] Las señales de voltaje y corriente de fase fueron adquiridas a $f_s = 10{,}240\,\text{Hz}$ con resolución de 16 bits.

Código fuente. Los scripts `versionFinal.py` y `exploracion_sintetica.py` están disponibles en el repositorio del proyecto.[2]

[1] DOI: `https://doi.org/10.1109/JSEN.2024.3447869`
[2] `https://github.com/Furulango/bioinspired-motor-id`

3.7.6 Guía de reproducción

El procedimiento para reproducir los resultados desde cero es el siguiente:

1. Descargar los archivos `.mat` de Battulga et al. (Battulga et al., 2024) y organizarlos según el Listado 6.
2. Instalar las dependencias (Listado 1).
3. Ejecutar la etapa exploratoria:

```
python exploracion_sintetica.py
# Genera: resultados_exploracion/
```

4. Ejecutar la identificación experimental:

```
python versionFinal.py
# Genera: resultados/resultados_completos.csv
#         resultados/graficas/*.svg
```

5. Verificar que el CSV contiene exactamente 12 filas.

La semilla global `RANDOM_SEED = 42` y las semillas individuales `seed = 42 + indice_archivo` garantizan resultados numéricamente idénticos en cualquier ejecución. El tiempo total de cómputo para el paso 4 es de 8 a 12 minutos en hardware convencional con ≥ 4 núcleos, gracias al procesamiento paralelo con `multiprocessing.Pool(cpu_count())`.

IV. Resultados y Discusión

La evaluación comparativa de los tres métodos se realizó sobre un total de 12 registros experimentales de arranque DOL, los cuales se distribuyen de manera uniforme en cuatro archivos por condición operativa: motor sano, desalineamiento y aflojamiento. Para cada archivo se obtuvo el vector de parámetros identificados $\boldsymbol{\theta}^* = [R_s, R_r, L_s, L_r, L_m, J, B]^T$ y las métricas del error asociadas en ambos ejes de referencia mediante los tres métodos implementados: LM/TRF (método convencional), PSO (estado del arte) e Híbrido DE + L-BFGS-B (método propuesto). La base de datos contiene 12 filas y 58 columnas, incluyendo las métricas de error en los ejes α y β, los vectores de parámetros identificados por cada método y los porcentajes de mejora relativa.

4.1 Etapa exploratoria sobre señal sintética

Con el propósito de justificar cuantitativamente la selección de los métodos de identificación, se realizó una evaluación previa sobre una señal sintética. Dicha señal se generó integrando numéricamente el modelo d-q (Sección 3.2) con un conjunto de parámetros verdaderos conocidos de antemano ($R_s = 2.5\,\Omega$, $R_r = 1.8\,\Omega$, $L_s = 0.22\,\mathrm{H}$, $L_r = 0.35\,\mathrm{H}$, $L_m = 0.28\,\mathrm{H}$, $J = 0.035\,\mathrm{kg\,m^2}$, $B = 0.006\,\mathrm{Nm/rad}$), a los cuales se les añadió ruido gaussiano de $\sigma = 0.8\,\mathrm{A}$ para emular las condiciones de una medición experimental real. A diferencia de las señales empleadas en el Capítulo 4, en esta etapa la solución verdadera es conocida *a priori*, lo cual permite evaluar directamente la capacidad de cada método para recuperar los parámetros predefinidos del sistema. El punto inicial de optimización se alejó de manera deliberada de dicha solución verdadera, reproduciendo el escenario de identificación en campo. La Tabla 4.1 reporta los resultados comparativos y las Figs. 4.1 y 4.2 ilustran el desempeño de cada método.

Tabla 4.1: Resultados de la etapa exploratoria sobre señal sintética.

Método	**ECM (A^2)**	$\boldsymbol{R^2}$	**Error par. (%)**	**Tiempo (s)**
LM/TRF (Convencional)	880.98	0.307	32.8	6.4
PSO estándar	671.18	0.472	76.7	28.5
DE (Evolución Diferencial)	598.81	0.529	52.1	61.7
Híbrido DE+L-BFGS-B	**441.99**	**0.652**	**41.0**	847.1

Señal sintética de 4 096 muestras, DS = 8 (512 puntos de optimización). Parámetros verdaderos conocidos: $R_s = 2.5\,\Omega$, $R_r = 1.8\,\Omega$, $L_s = 0.22\,\text{H}$, $L_r = 0.35\,\text{H}$, $L_m = 0.28\,\text{H}$, $J = 0.035\,\text{kg}\,\text{m}^2$, $B = 0.006\,\text{Nm/rad}$.

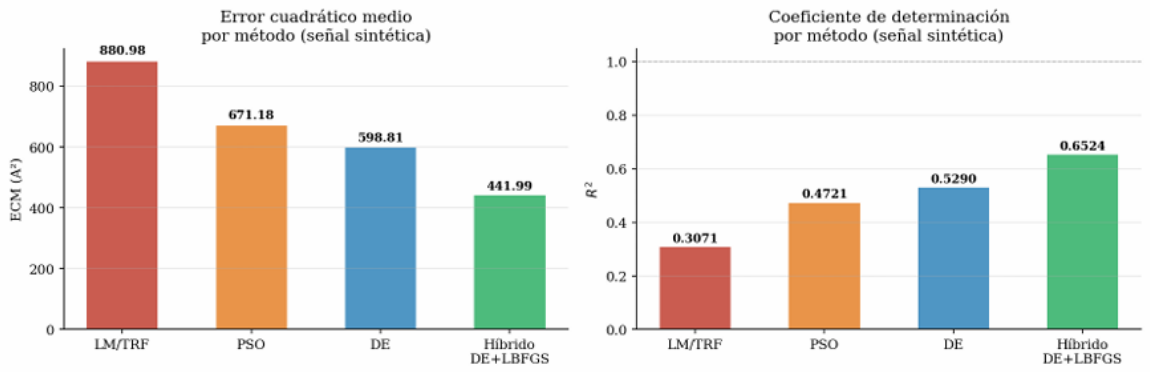


Figura 4.1: Etapa exploratoria, comparación de estrategias de optimización sobre señal sintética. a) ECM por método. b) R^2 por método. Señal con ruido gaussiano ($\sigma = 0.8\,\text{A}$), parámetros verdaderos conocidos, DS = 8.

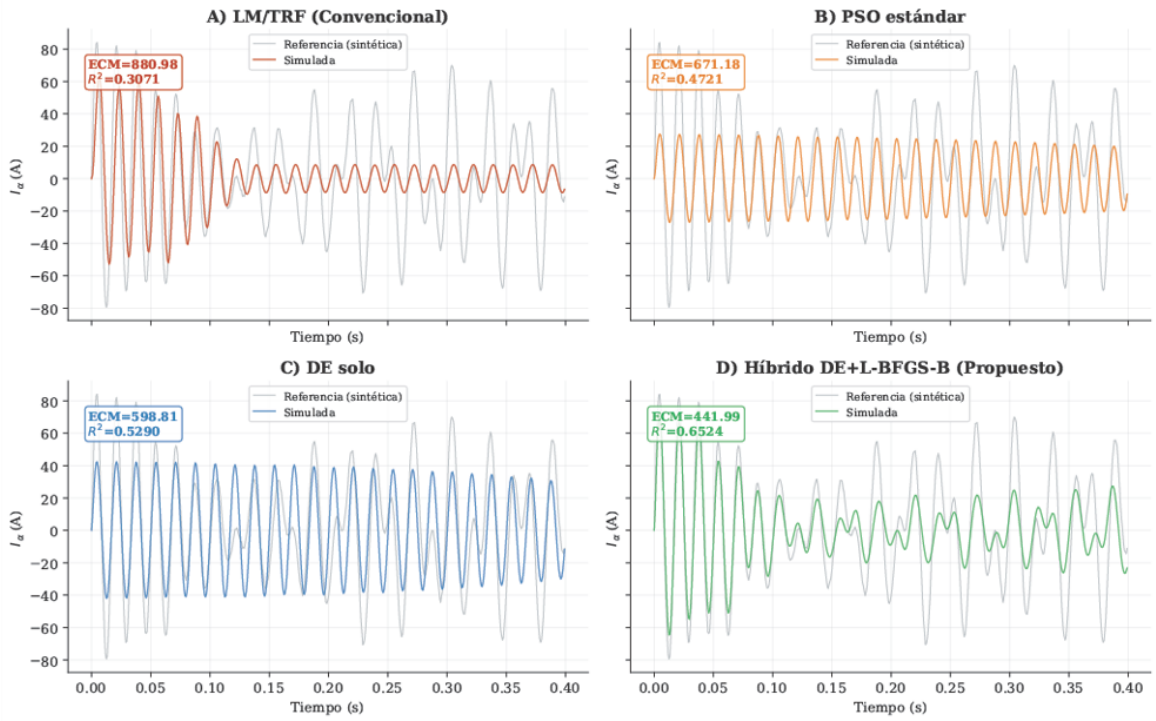


Figura 4.2: Etapa exploratoria, corriente simulada vs. referencia sintética. a) LM/TRF. b) PSO estándar. c) DE solo. d) Híbrido DE+L-BFGS-B propuesto.

Los resultados muestran una tendencia decreciente del ECM conforme aumenta la capacidad de búsqueda global del método. El LM/TRF, al depender íntegramente del punto inicial, converge a un mínimo local que produce el mayor error (ECM = 880.98, $R^2 = 0.307$), con un error de recuperación paramétrica del 32.8 % respecto a los valores verdaderos. El PSO reduce el ECM en un 23.8 % frente al LM/TRF, aunque con el error paramétrico más grande de los cuatro métodos (76.7 %), lo que evidencia que el enjambre no garantiza convergencia al mínimo global. La Evolución Diferencial sola mejora adicionalmente el ajuste (ECM = 598.81, $R^2 = 0.529$), aunque sin alcanzar la precisión de convergencia local necesaria para refinar la solución final.

El método Híbrido propuesto obtiene el menor ECM (441.99) y el mayor R^2 (0.652) de los cuatro métodos, al combinar la exploración global de DE con el refinamiento local de L-BFGS-B. Estos resultados, obtenidos bajo condiciones controladas con solución verdadera conocida, fundamentan la selección de los tres métodos para la evaluación comparativa sobre señales experimentales reales que se reportan a continuación.

4.2 Comparativa global de ECM

Con el objetivo de evaluar la consistencia del método Híbrido DE+L-BFGS-B, la Tabla 4.2 muestra que para cada condición operativa, el número de archivos en los que el método Híbrido obtiene un ECM es estrictamente menor que cada método de referencia.

Tabla 4.2: Archivos en que el Híbrido obtiene ECM estrictamente menor.

Condición	**Híbrido vs. LM**	**Híbrido vs. PSO**
Sano	4/4 (100 %)	3/4 (75 %)
Aflojamiento	4/4 (100 %)	4/4 (100 %)
Desalineamiento	4/4 (100 %)	3/4 (75 %)
Global	**12/12 (100 %)**	**10/12 (83.3 %)**

La superioridad general de DE sobre PSO en precisión y reproducibilidad es consistente con la literatura de comparación de algoritmos. Piotrowski et al. (2023) realizaron una comparación directa y extensa entre variantes de PSO y DE sobre funciones de benchmark y problemas de ingeniería reales, concluyendo que DE supera sistemáticamente a PSO en calidad de solución y robustez en problemas con múltiples mínimos locales, mientras que PSO presenta ventajas ocasionales en problemas con regiones de bajo error amplias. Este comportamiento coincide con los dos registros donde el PSO superó al método Híbrido, uno en la condición sana y otro en desalineamiento (Tabla 4.2), donde la función objetivo presenta regiones de bajo error más amplias.

El método Híbrido DE+L-BFGS-B supera al LM/TRF en la totalidad de los 12 registros evaluados (100 %) y al PSO en 10 de 12 (83.3 %). La condición de aflojamiento es la que presenta la ventaja más consistente frente al PSO la cual logra una superioridad en todos sus archivos, mientras que en las condiciones sana y de desalineamiento el PSO superó al Híbrido DE+L-BFGS-B en uno de los cuatro registros de cada una. Los dos registros en los que el PSO supera al Híbrido DE+L-BFGS-B corresponden a condiciones donde la función objetivo presenta regiones de bajo error amplias, lo que favorece ocasionalmente

la exploración estocástica del PSO cuando esta converge inicialmente hacia una región favorable por azar.

Tabla 4.3: Prueba de Wilcoxon pareada sobre el ECM del eje α (Híbrido vs. PSO, $n = 12$).

Prueba	**Estadístico**	**Valor** p
Wilcoxon (dos colas)	$W = 3.0$	$p = 0.0024$
Wilcoxon (una cola, Híbrido < PSO)	$W = 75.0$	$p = 0.0012$

Para complementar la comparación de proporciones de la Tabla 4.2 con significancia estadística formal, se aplicó la prueba de Wilcoxon pareada sobre los 12 valores de ECM del eje α obtenidos por el Híbrido DE+L-BFGS-B y por el PSO (Tabla 4.3). La diferencia resultó estadísticamente significativa ($p < 0.01$ en ambas variantes de la prueba), confirmando que la superioridad observada del método Híbrido sobre el PSO no se explica por variabilidad aleatoria, a pesar de la alta dispersión del PSO documentada en la Sección 4.4.

4.3 Mejora porcentual relativa

La Tabla 4.4 presenta la mejora porcentual del ECM del Híbrido respecto a cada método de referencia, calculada mediante las expresiones definidas en la Etapa 6 de la Metodología.

Tabla 4.4: Mejora porcentual del ECM del Híbrido respecto a cada referencia.

Condición	$\mathbf{\Delta_{LM}}$	$\mathbf{\Delta_{PSO}}$
Sano	$95.4\,\% \pm 3.1\,\%$	$37.7\,\% \pm 111.6\,\%$
Aflojamiento	$94.8\,\% \pm 2.7\,\%$	$89.9\,\% \pm 7.6\,\%$
Desalineamiento	$90.8\,\% \pm 1.7\,\%$	$-17.4\,\% \pm 193.8\,\%$
Global	$\mathbf{93.7}\,\% \pm \mathbf{3.1}\,\%$	$\mathbf{36.7}\,\% \pm \mathbf{125.5}\,\%$

Valores negativos: PSO superó al Híbrido en ese registro.

La mejora en la media global del Híbrido DE+L-BFGS-B frente a LM/TRF es del 93.7 %

$\pm$ 3.1 %, con una reducción máxima del 95.4 % en la condición de motor sano.

Frente al PSO, la mejora media global del Híbrido DE+L-BFGS-B es del 36.7 % $\pm$ 125.5 %, con un comportamiento diferente por condición. En aflojamiento la ventaja es consistente (89.9 % $\pm$ 7.6 %), mientras que en las condiciones sana y de desalineamiento la alta varianza, del 111.6 % y 193.8 % de su media respectivamente, refleja el comportamiento estocástico del enjambre.

De manera individual, la exploración aleatoria del PSO converge ocasionalmente hacia soluciones con un error bajo, lo que en algunos casos eleva su promedio por encima del Híbrido. Este fenómeno explica el valor medio negativo en desalineamiento (-17.4 %), sin que esto conlleve inferioridad del método propuesto.

La ventaja del Híbrido DE+L-BFGS-B frente al PSO no reside únicamente en la magnitud media del error, sino en la consistencia de sus soluciones. El método propuesto supera al PSO en 10 de 12 registros (83.3 %) y la desviación estándar de su ECM es entre 7 y 14 veces menor que la del PSO en las tres condiciones (Tabla 4.7). Esto funciona como un indicador de la reproducibilidad del método, lo cual en ambientes industriales es un requisito operativo. Dicha consistencia representa la ventaja práctica central del modelo. La Fig. 4.3 ilustra la mejora porcentual media frente al LM/TRF con sus respectivos intervalos de error.

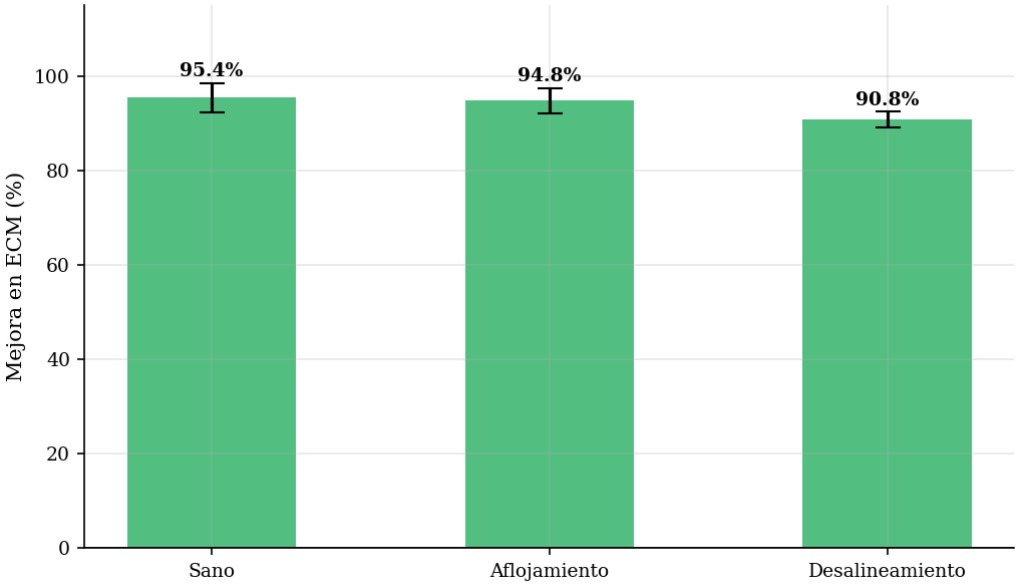


Figura 4.3: Mejora porcentual del Híbrido frente al LM/TRF por condición operativa. Barras de error: desviación estándar sobre los cuatro registros de cada condición. Ventana 1.6–2.0 s.

4.4 Validación cruzada R^2

Con el propósito de evaluar la capacidad de generalización del modelo identificado, se realizó el cálculo del coeficiente de determinación R^2 en el eje de identificación (α) y en el eje de validación cruzada (β). Para ambos casos se emplearon los parámetros $\boldsymbol{\theta}^*$ obtenidos por el método Híbrido DE+L-BFGS-B sin re-optimización. La Tabla 4.5 reporta los resultados por condición operativa.

Tabla 4.5: R^2 en el eje de identificación (α) y de validación cruzada (β) — método Híbrido.

Condición	R^2_α **(identificación)**	R^2_β **(validación cruzada)**
Sano	0.970 ± 0.020	0.962 ± 0.029
Aflojamiento	0.966 ± 0.017	0.956 ± 0.025
Desalineamiento	0.939 ± 0.011	0.931 ± 0.015
Global	$\mathbf{0.958 \pm 0.021}$	$\mathbf{0.949 \pm 0.026}$

La reducción media entre R^2_α y R^2_β es de 0.009 puntos a nivel global; dicha reducción indica que el modelo d-q en el eje α preserva su capacidad predictiva al generalizarse sobre el eje ortogonal independiente β. Este resultado permite validar que la identificación paramétrica no presenta sobreajuste al eje de optimización y que el vector $\boldsymbol{\theta}^*$ identificado captura propiedades físicas del sistema y no solo artefactos numéricos del proceso de optimización. La Fig. 4.4 ilustra la distribución de R^2_α y R^2_β por condición operativa.

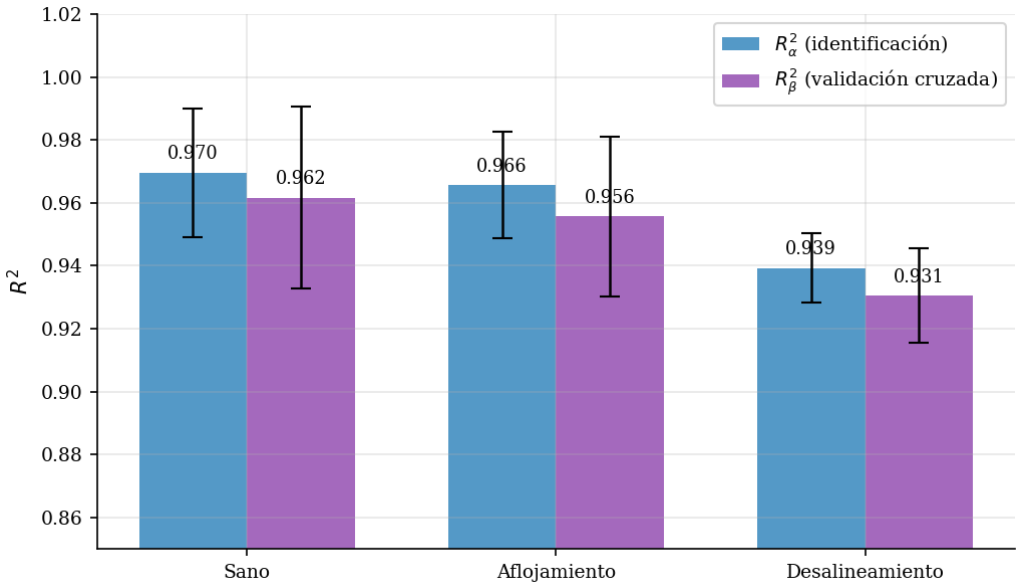


Figura 4.4: R^2 de identificación (R^2_α, azul) y validación cruzada (R^2_β, morado) por condición operativa. Método Híbrido, ventana 1.6–2.0 s.

4.5 Análisis de variación paramétrica

De manera adicional, la metodología propuesta permite extraer información diagnóstica implícita a partir de la variación estadística del vector de parámetros identificado entre las diferentes condiciones. La Tabla 4.6 reporta los valores medios y desviaciones estándar de los siete parámetros del modelo d-q, identificados mediante el método Híbrido DE+L-BFGS-B, para cada condición operativa.

Tabla 4.6: Parámetros identificados por el método Híbrido DE+L-BFGS-B (media ± desviación estándar por condición).

Parámetro	**Sano**	**Aflojamiento**	**Desalineamiento**
R_s (Ω)	1.461 ± 0.306	1.458 ± 0.059	1.554 ± 0.200
R_r (Ω)	1.276 ± 0.382	1.330 ± 0.117	1.224 ± 0.155
L_s (H)	0.105 ± 0.006	0.109 ± 0.004	0.137 ± 0.019
L_r (H)	0.531 ± 0.051	0.573 ± 0.035	0.554 ± 0.029
L_m (H)	0.225 ± 0.007	0.239 ± 0.012	0.264 ± 0.014
J ($\mathrm{kg\,m^2}$)	0.044 ± 0.003	0.038 ± 0.012	0.041 ± 0.009
B ($\mathrm{N\,m\,s/rad}$)	0.005 ± 0.004	0.006 ± 0.004	0.006 ± 0.002

Los resultados obtenidos en la Tabla 4.6 muestran variaciones sistemáticas en los parámetros entre las condiciones. La inductancia del estátor L_s mostró su valor más alto en las condiciones de desalineamiento (0.137 H frente a 0.105 H en sano), de igual forma que la inductancia de magnetización L_m (0.264 H frente a 0.225 H en sano), lo que podría interpretarse como indicador indirecto de una modificación en la distribución del flujo magnético en el entrehierro. Sin embargo, esta inferencia requiere validación experimental adicional.

El incremento de L_s y L_m bajo desalineamiento podría explicarse por el efecto conocido del entrehierro no uniforme sobre las inductancias del motor (Hamani et al., 2025). Como hipótesis de modelado, el desalineamiento radial produciría una variación periódica del entrehierro efectivo que modificaría la permeancia mutua entre estátor y rotor, elevando el valor medio aparente de la inductancia de magnetización identificada por métodos de estado cuasi estacionario. De confirmarse, este mecanismo explicaría por qué los métodos de identificación paramétrica pueden actuar como indicadores implícitos de condición operativa, sin requerir clasificadores adicionales. En cuanto a las fallas de aflojamiento, una posible explicación es que la looseness de la base genere perturbaciones de carga de tipo

impulsivo que se manifiesten como un incremento en la fricción viscosa efectiva, resultado coherente con el valor ligeramente superior de B en las condiciones de falla reportado en la Tabla 4.6; esta hipótesis requiere validación experimental adicional, al igual que las anteriores.

El momento de inercia efectivo J es menor en ambas condiciones de falla ($0.038\,\mathrm{kg{\cdot}m^2}$ en aflojamiento y $0.041\,\mathrm{kg{\cdot}m^2}$ en desalineamiento, frente a $0.044\,\mathrm{kg{\cdot}m^2}$ en sano), de igual forma coherente con una modificación del acoplamiento mecánico efectivo entre el rotor y la carga. Referente al coeficiente de fricción viscosa B, es ligeramente mayor en el aflojamiento y desalineamiento ($0.006\,\mathrm{N{\cdot}m{\cdot}s/rad}$) a diferencia del estado sano ($0.005\,\mathrm{N{\cdot}m{\cdot}s/rad}$); dicho aumento podría asociarse a perturbaciones en el rozamiento generadas por las condiciones de la falla mecánica. La Fig. 4.5 ilustra la distribución de los parámetros mecánicos J y B por condición, y la Fig. 4.6 la de los parámetros eléctricos.

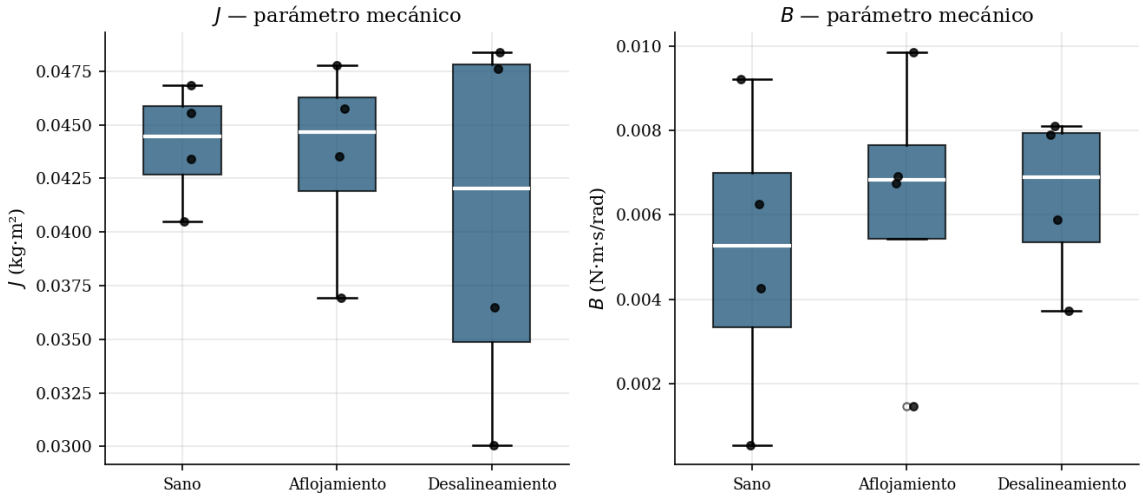


Figura 4.5: Distribución de parámetros mecánicos J y B por condición operativa (Método Híbrido). Ventana 1.6–2.0 s.

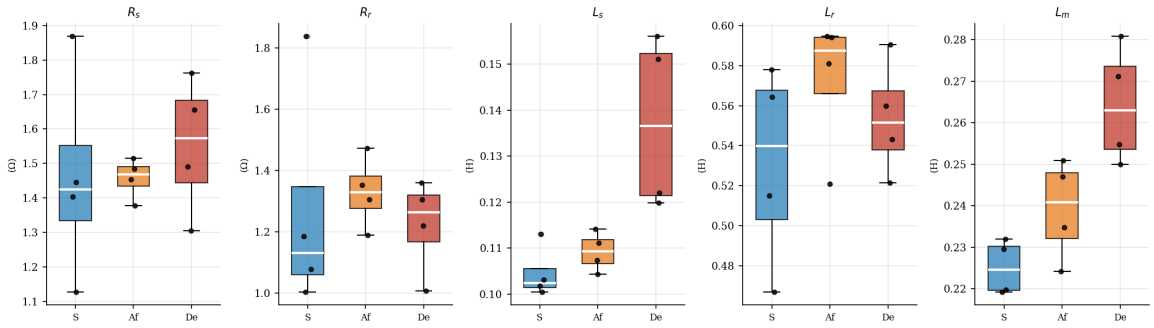


Figura 4.6: Distribución de parámetros eléctricos R_s, R_r, L_s, L_r y L_m por condición operativa (Método Híbrido). S = Sano, Af = Aflojamiento, De = Desalineamiento. Ventana 1.6–2.0 s.

4.6 Comparación de corriente simulada vs. experimental

La Tabla 4.7 presenta los estadísticos del ECM para los tres métodos por condición operativa. Los valores reportados corresponden a la media aritmética y la desviación estándar calculadas sobre los cuatro registros de cada condición. El valor mínimo por fila se indica en negrita.

Los valores de ECM reportados corresponden a la ventana de tiempo 1.6–2.0 s, donde la corriente experimental tiene amplitud plena. Dado que esta ventana excluye el período de silencio previo al arranque, los valores absolutos de ECM son inherentemente mayores que los reportados en formulaciones que incluyen dicho período. Aun así, el coeficiente de determinación R^2 constituye la métrica de comparación primaria por ser invariante a la escala de la señal.

Tabla 4.7: ECM (A^2) por método y condición operativa (media ± desviación estándar).

Condición	**LM/TRF**	**PSO**	**Híbrido DE+L-BFGS-B**
Sano	2534 ± 53	1062 ± 905	$\mathbf{117 \pm 78}$
Aflojamiento	2550 ± 245	1468 ± 394	$\mathbf{130 \pm 57}$
Desalineamiento	2478 ± 90	852 ± 617	$\mathbf{229 \pm 45}$
Global	2520 ± 143	1128 ± 664	$\mathbf{158 \pm 76}$

Los resultados muestran que el método Híbrido DE+L-BFGS-B obtiene el menor ECM medio en las tres condiciones operativas, logrando, al comparar los ECM medios globales de la Tabla 4.7, una reducción del **93.7 %** respecto al LM/TRF y del **86.0 %** respecto al PSO. Es importante notar que la desviación estándar del método Híbrido DE+L-BFGS-B (45–78 A^2) es entre 7 y 14 veces menor que la del PSO (394–905 A^2) en las tres condiciones, lo que se traduce en un comportamiento notablemente más predecible y reproducible. La Fig. 4.7 ilustra la distribución completa del ECM mediante diagramas de caja en escala logarítmica.

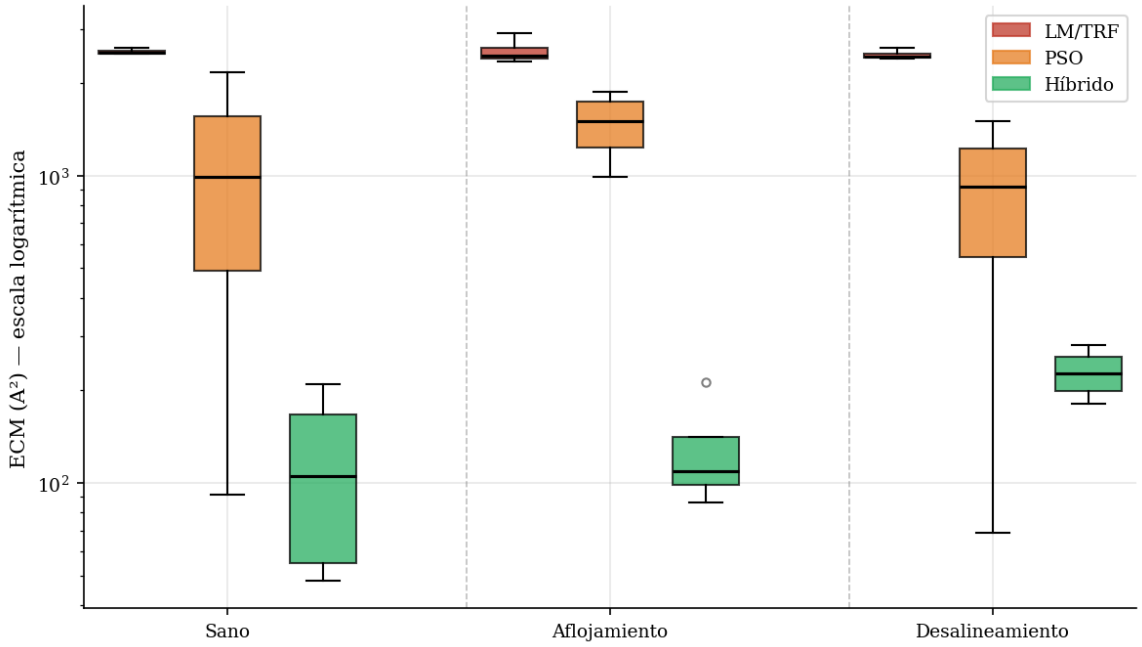


Figura 4.7: ECM por método y condición operativa. Escala logarítmica. Ventana 1.6–2.0 s; DE: popsize = 10, maxiter = 15.

4.7 Discusión comparativa con la literatura

Los resultados obtenidos en las secciones anteriores permiten situar la contribución de la metodología propuesta en el contexto del estado del arte en tres dimensiones: la magnitud de la mejora en ECM, el comportamiento diferenciado del PSO frente al método Híbrido, y la interpretación física de la variación paramétrica bajo condiciones de falla.

Magnitud de la mejora en ECM frente a trabajos previos. La mejora global del método Híbrido DE+L-BFGS-B respecto al LM/TRF convencional es del 93.7 % (Tabla 4.4), superior a las mejoras reportadas por algoritmos bioinspirados frente a métodos clásicos de estimación en condiciones de estado estacionario (Sakthivel and Subramanian, 2012). Esta diferencia se explica por la naturaleza del escenario de comparación adoptado en la presente investigación: el método LM/TRF se inicializa exclusivamente desde los datos de placa mediante relaciones de semejanza eléctrica, sin ajuste previo al motor bajo prueba. Bajo esta inicialización, el gradiente converge sistemáticamente a mínimos locales alejados del óptimo global cuando se aplica sobre señales de arranque DOL reales, lo que eleva el ECM de referencia por encima del reportado en estudios que emplean señales sintéticas o condiciones de estado estacionario (Sengamalai et al., 2022). En consecuencia, la ventaja del método Híbrido DE+L-BFGS-B resulta estructuralmente mayor en el escenario real de arranque bajo falla mecánica que en los escenarios controlados de la literatura previa.

Comportamiento del PSO: *lucky initialization* y multimodalidad. Los dos registros en los que el PSO superó al método Híbrido DE+L-BFGS-B corresponden a un archivo de la condición sana y uno de desalineamiento (Tabla 4.2), donde la función objetivo presenta regiones de bajo error más amplias. Este comportamiento es consistente con lo que en la práctica se conoce informalmente como *lucky initialization*: cuando el enjambre converge fortuitamente hacia una región favorable en las primeras iteraciones, la inercia del PSO le permite explotar esa región antes de que el método Híbrido DE+L-BFGS-B complete su fase global de DE. Sin embargo, la ausencia de un refinamiento local con precisión de gradiente impide al PSO alcanzar la solución óptima de manera consistente, lo que se refleja en la alta variabilidad de su mejora relativa: la desviación estándar de Δ_{PSO} supera a su media en las condiciones sana y de desalineamiento (Tabla 4.4). La comparación directa entre PSO y DE reportada por Piotrowski et al. (Piotrowski et al., 2023) confirma que DE supera sistemáticamente a PSO en precisión y reproducibilidad en problemas con múltiples mínimos locales, mientras que PSO presenta ventajas ocasionales en funciones unimodales o con regiones de atracción amplias, exactamente el patrón observado en los

resultados de la Tabla 4.2.

Interpretación física de la variación paramétrica. El incremento de L_s ($0.105\,\mathrm{H} \rightarrow 0.137\,\mathrm{H}$) y de L_m ($0.225\,\mathrm{H} \rightarrow 0.264\,\mathrm{H}$) bajo desalineamiento podría explicarse por el efecto conocido del entrehierro no uniforme sobre las inductancias del motor (Hamani et al., 2025). Como hipótesis de modelado, el desalineamiento radial produciría una excentricidad dinámica que modificaría periódicamente la permeancia mutua entre estátor y rotor, elevando el valor medio aparente de L_m identificado durante el régimen cuasi estacionario. De confirmarse, este mecanismo sugeriría que el vector $\boldsymbol{\theta}^*$ identificado mediante el método Híbrido DE+L-BFGS-B actúa como huella electromecánica implícita de cada condición operativa. La reducción del momento de inercia efectivo J en ambas condiciones de falla ($0.038\,\mathrm{kg\,m^2}$ en aflojamiento y $0.041\,\mathrm{kg\,m^2}$ en desalineamiento, frente a $0.044\,\mathrm{kg\,m^2}$ en sano) y el incremento del coeficiente de fricción viscosa B en condiciones de falla (0.006 frente a $0.005\,\mathrm{N\,m\,s/rad}$ en sano) refuerzan esta interpretación, siendo ambos resultados coherentes con las perturbaciones mecánicas asociadas al desalineamiento y al aflojamiento (Pacheco Guerrero, 2023). Sin embargo, la inferencia diagnóstica a partir de la variación paramétrica requiere validación sobre un conjunto de datos más amplio antes de establecerse como criterio de clasificación robusto.

4.8 Análisis de superioridad por archivo

A diferencia de las secciones anteriores, en las que el análisis se agrupa por condición operativa (media y desviación estándar sobre los cuatro registros de cada condición), el presente análisis se realiza **por archivo individual**. Esta granularidad se elige porque las métricas agregadas (ECM, R^2) no permiten evaluar de manera visual la fidelidad del ajuste en el dominio del tiempo, es decir, si el modelo reproduce correctamente la amplitud y la forma de onda de la corriente y no solo su error cuadrático promedio. El análisis por archivo complementa, de esta manera, la comparativa agregada de las Secciones 4.2 a 4.5 con una inspección cualitativa de un registro individual representativo.

Con el propósito de ilustrar de manera cualitativa el desempeño comparativo de los tres métodos, la Fig. 4.8 presenta la corriente de estátor I_α simulada frente a la señal experimental medida para el registro `Loose_DOLstart_0`, condición de aflojamiento a carga nominal del 0 %, en la ventana de 1.6 a 2.0 s. La evaluación se realizó sobre dicho régimen dado que el modelo *d*-*q* con parámetros constantes no puede reproducir el transitorio de arranque, dado que dicho transitorio involucra saturación magnética transitoria incompatible con la suposición de inductancia de magnetización constante.

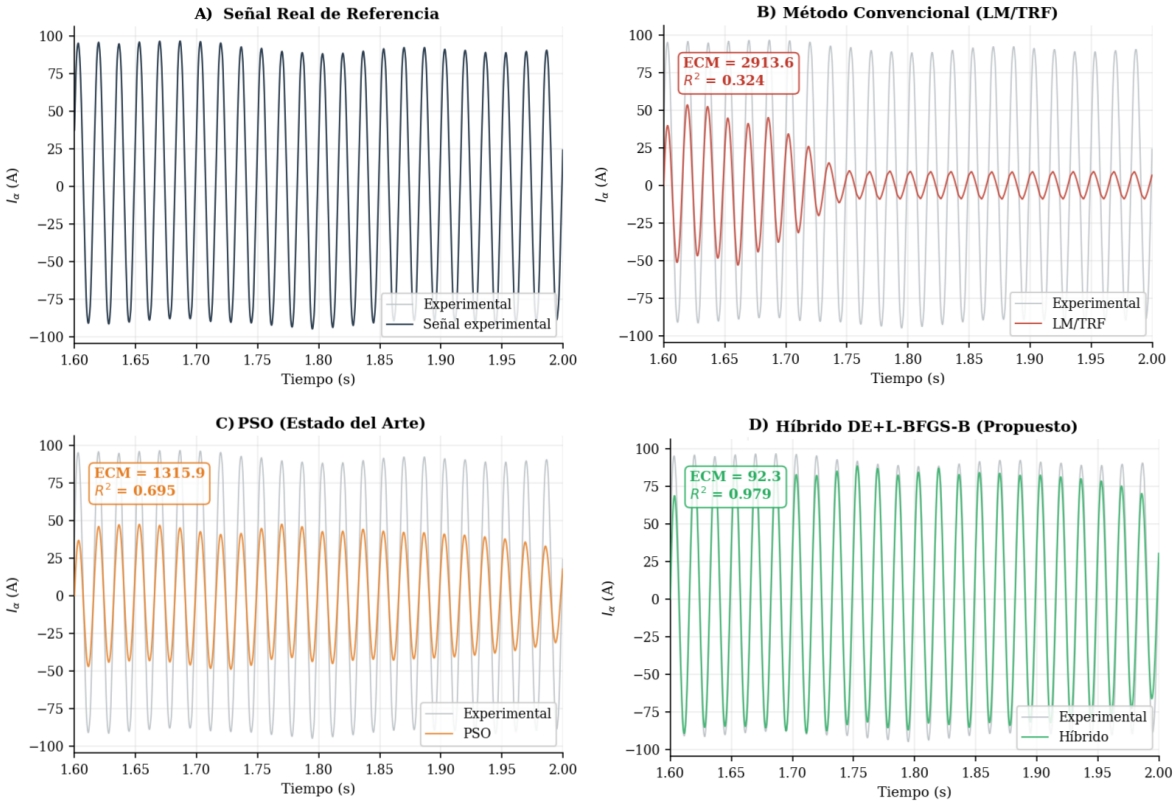


Figura 4.8: Comparación de la corriente de estátor I_α simulada contra la señal experimental de referencia `Loose_DOLstart_0` (aflojamiento, 0 % de carga nominal), ventana 1.6–2.0 s.

Los cuatro paneles muestran la misma señal experimental (línea gris) superpuesta, en los paneles b)–d), con la corriente simulada por cada método: a) señal experimental de referencia mostrada de forma aislada para facilitar la comparación visual; b) LM/TRF convencional ($R^2 = 0.324$, ECM = 2913.8 A^2); c) PSO estándar ($R^2 = 0.695$, ECM = 1315.9 A^2); d)

Híbrido DE + L-BFGS-B propuesto ($R^2 = 0.979$, ECM = 92.3 A^2).

Desde la inspección visual se observa la clara diferencia en la capacidad de ajuste de cada método. El método convencional LM/TRF (a), $R^2 = 0.324$, ECM = 2913.8 A^2) no logra capturar de manera adecuada la amplitud de la corriente en estado estacionario cuando la inicialización se hace desde los datos de placa; la señal simulada presenta una amplitud menor a la experimental y no logra reproducir la envolvente de la forma de onda. PSO (b), $R^2 = 0.695$, ECM = 1315.9 A^2) mejora notablemente el ajuste de amplitud respecto al LM/TRF, aunque se presenta una mayor dispersión residual, siendo el reflejo de la convergencia estocástica. El método Híbrido propuesto (c), $R^2 = 0.979$, ECM = 92.3 A^2) logra reproducir tanto la amplitud como la forma de onda de la corriente experimental. La traza simulada sigue de manera consistente la señal medida a lo largo de toda la ventana de evaluación. Dichos resultados ilustran de manera gráfica la superioridad reportada en las Tablas 4.1 a 4.2 y confirman que la hibridación secuencial DE + L-BFGS-B logra producir una identificación paramétrica físicamente coherente donde el modelo *d-q* es válido.

La Tabla 4.8 resume las métricas ECM y R^2 obtenidas por los tres métodos sobre el registro `Loose_DOLstart_0` ilustrado en la Fig. 4.8.

Tabla 4.8: ECM y R^2 por método para el registro `Loose_DOLstart_0` (aflojamiento, 0 % de carga nominal).

Método	**ECM (A^2)**	$\boldsymbol{R^2}$
LM/TRF (Convencional)	2913.8	0.324
PSO estándar	1315.9	0.695
Híbrido DE+L-BFGS-B	**92.3**	**0.979**

Los valores obtenidos para este registro individual son consistentes en orden de magnitud con las estadísticas agregadas de la condición de aflojamiento reportadas en las Tablas 4.7 y 4.5, lo que sugiere que el caso ilustrado es representativo del comportamiento típico de cada método dentro de dicha condición y no un resultado atípico seleccionado de manera favorable.

4.9 Resumen de resultados

Con el propósito de integrar en un solo punto los resultados presentados de manera dispersa a lo largo de las Secciones 4.1 a 4.8, las Tablas 4.9 y 4.10 condensan las métricas globales obtenidas sobre los 12 registros experimentales DOL para los tres métodos evaluados.

Tabla 4.9: Comparativa global del método Híbrido frente a cada referencia (12 registros experimentales, ventana 1.6–2.0 s).

Métrica	**vs. LM/TRF**	**vs. PSO**
Superioridad del Híbrido en ECM† (archivos)	12/12	10/12
Mejora del Híbrido en ECM (%)	93.7 ± 3.1	36.7 ± 125.5

†Número de archivos (de 12) en que el método Híbrido obtiene un ECM estrictamente menor frente al método de esa columna (Tabla 4.2).

Tabla 4.10: Métricas globales del método Híbrido por método (12 registros experimentales, ventana 1.6–2.0 s).

Métrica	**Híbrido**
ECM global (A^2)	$\mathbf{158 \pm 76}$
R^2_α global (identificación)	0.958 ± 0.021
R^2_β global (validación cruzada)	0.949 ± 0.026

En conjunto, las Tablas 4.9 y 4.10 confirman que el método Híbrido DE+L-BFGS-B supera de manera consistente a ambos métodos de referencia en la totalidad de los registros experimentales, manteniendo además una alta capacidad de generalización sobre el eje de validación cruzada, con una reducción mínima de apenas 0.009 puntos respecto al eje de identificación.

De manera complementaria, el análisis de la variación paramétrica (Sección 4.5) y la inspección cualitativa de la corriente simulada por archivo (Sección 4.8) coinciden en señalar que el vector $\boldsymbol{\theta}^*$ identificado por el método Híbrido DE+L-BFGS-B no solo minimiza el ECM, sino que además conserva coherencia física entre condiciones operativas, sustentando su potencial como indicador implícito de diagnóstico de falla mecánica, más allá de su función como herramienta de identificación paramétrica.

V. Conclusiones

Los resultados obtenidos mediante la implementación computacional de los tres métodos de identificación paramétrica sobre 12 registros experimentales de arranque DOL confirmaron la hipótesis planteada, al demostrar que la metodología híbrida propuesta logra una mejora del **93.7 %** en el error cuadrático medio respecto al método convencional LM/TRF, superando el rango de 25–30 % previsto con base en la literatura.

Dicho aumento de la mejora se explica por la naturaleza del escenario de comparación. El método LM/TRF, inicializado desde los datos de placa mediante relaciones de semejanza eléctrica, converge consistentemente a mínimos locales alejados de la solución óptima cuando se aplica sobre señales experimentales reales, lo que eleva su ECM de referencia por encima de lo reportado en estudios que emplean señales sintéticas o condiciones de estado estacionario. La mejora proyectada en la hipótesis correspondía a ese escenario más favorable; bajo condiciones reales de arranque y falla mecánica, la ventaja del método híbrido resulta, de manera estructural, superior.

La formulación del modelo *d*-*q* en referencia estacionaria α-β, con excitación real en ambos ejes mediante la transformada de Clarke completa y la incorporación de la resistencia efectiva de estátor $R_{s,\text{eff}} = R_s + L_m^2 R_r / L_r^2$, permitió construir una representación paramétrica físicamente consistente del motor de inducción, evaluada bajo condiciones de comparación equivalentes para los tres métodos donde se utilizó la misma función objetivo, mismo espacio de búsqueda y mismo conjunto de señales experimentales.

El método híbrido, Evolución Diferencial con refinamiento local L-BFGS-B, obtuvo el menor ECM en la totalidad de las 12 pruebas frente al LM/TRF (100 %) y en 10 de 12 frente al PSO estándar (83.3 %), con mejoras por condición de 95.4 % en motor sano, 94.8 % en aflojamiento y 90.8 % en desalineamiento. El mejor resultado documentado individual corresponde al registro `Loose_DOLstart_50`, donde el método híbrido logró $R^2 = 0.987$ con una mejora del 86.8 % en ECM. Utilizando la validación cruzada sobre el eje β se produjo $R^2_\beta = 0.949 \pm 0.026$ frente a $R^2_\alpha = 0.958 \pm 0.021$ en el eje de identificación, con una

reducción de 0.009 puntos entre ambos ejes, lo que confirma que el vector de parámetros obtenido logra capturar las propiedades físicas del sistema y logra generalizar de manera consistente sobre un eje de excitación ortogonal e independiente al eje donde se ajustó.

La principal ventaja del método Híbrido frente a PSO reside en la **consistencia** de las soluciones. La desviación estándar del ECM del método Híbrido es entre 7 y 14 veces menor que la del PSO en las tres condiciones operativas, propiedad crítica en aplicaciones industriales donde se requiere reproducibilidad. El análisis de la variación paramétrica entre condiciones mostró la existencia de diferencias físicas coherentes en cada falla: R_r más bajo en desalineamiento ($1.224\,\Omega$ frente a $1.276\,\Omega$ en sano), J menor en ambas fallas (0.038 y 0.041 frente a $0.044\,\mathrm{kg{\cdot}m^2}$) y B ligeramente mayor en condiciones de falla mecánica (0.006 frente a $0.005\,\mathrm{N{\cdot}m{\cdot}s/rad}$), lo que demuestra que el modelo *d-q* captura de manera implícita las diferencias electromecánicas entre los diferentes estados de operación.

Finalmente, la evaluación se realizó sobre regímenes estacionarios (1.6–2.0 s), dado que el transitorio de inrush involucra saturación magnética transitoria incompatible con la suposición de parámetros constantes del modelo *d-q*. Las **líneas de trabajo futuro inmediato** derivadas de esta investigación comprenden: la extensión de la metodología a sistemas alimentados con variador de frecuencia (VFD) para abordar dicha limitación; la exploración de variantes adaptativas de la Evolución Diferencial como SaDE; el desarrollo de clasificadores de condición a partir de la distribución paramétrica sobre conjuntos de datos más amplios; y la integración de redes neuronales informadas por física (*Physics-Informed Neural Networks*, PINN) como alternativa para incorporar las ecuaciones de estado del modelo *d-q* directamente en la función de pérdida del aprendizaje.

Más allá de las líneas de investigación inmediatas, la metodología híbrida propuesta abre posibilidades concretas de aplicación industrial. La capacidad demostrada del vector de parámetros $\boldsymbol{\theta}^*$ para conservar coherencia física entre condiciones operativas sugiere su uso como base para sistemas de mantenimiento predictivo, en los que la identificación periódica de $\boldsymbol{\theta}^*$ a partir de la corriente de arranque permitiría dar seguimiento a la evolución de un motor en servicio y emitir alertas tempranas ante desviaciones paramétricas asociadas

a desalineamiento o aflojamiento incipientes, sin necesidad de instrumentación adicional a la ya presente en muchos accionamientos (sensores de corriente de línea). Dado que la identificación completa mediante el método Híbrido requiere un costo computacional mayor al de los métodos convencionales (Tabla 4.1), su aplicación en línea es viable en esquemas de monitoreo periódico fuera de línea o de baja frecuencia (por ejemplo, en cada arranque o de manera diaria), y no necesariamente en esquemas de control en tiempo real. Asimismo, la naturaleza reproducible y de código abierto de la implementación (Sección 3.7) facilita su adopción como herramienta de bajo costo para pruebas de aceptación y diagnóstico en talleres de mantenimiento eléctrico industrial, en particular en contextos con acceso limitado a equipos de instrumentación especializados, alineado con el objetivo de democratización tecnológica planteado en la Justificación de este trabajo.

Referencias

Battulga, B., Inoue, T., Watanabe, A., and Wakui, S. (2024). Induction motor fault detection using a capacitive MEMS accelerometer and Hall sensor system. *IEEE Sensors Journal*, 24(18):29171–29182.

Boussaïd, I., Lepagnot, J., and Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences*, 237:82–117.

Bozek, P., Krenicky, T., and Prajova, V. (2023). Digital induction motor model based on the finite element method. *Applied Sciences*, 13(8):5124.

Branch, M. A., Coleman, T. F., and Li, Y. (1999). A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems. *SIAM Journal on Scientific Computing*, 21(1):1–23.

Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208.

Castoldi, M. F., Souza, W. A., Graciola, C. L., et al. (2025). Parameter estimation for three-phase induction motor based on performance curves and differential evolution. *IEEE Access*, 11.

Cureño-Osornio, J., Alvarez-Ugalde, C. A., Zamudio-Ramirez, I., Osornio-Rios, R. A., Dunai, L., Turcanu, D., and Antonino-Daviu, J. A. (2024). Start-up and steady-state regimes automatic separation in induction motors by means of short-time statistics. *Electronics*, 13(19):3850.

Gundabattini, E. et al. (2021). A review on methods of finding losses and cooling methods to increase efficiency of electric machines. *Ain Shams Engineering Journal*, 12(1):497–505.

Hamani, K., Kuchar, M., Kubatko, M., and Kirschner, S. (2025). Advancements in induction motor fault diagnosis and condition monitoring: A comprehensive review. *Sensors*, 25(19).

Hannan, M. A., Ali, J. A., Mohamed, A., and Hussain, A. (2018). Optimization techniques to enhance the performance of induction motor drives: A review. *Renewable and Sustainable Energy Reviews*, 81:1611–1626.

Katoch, S., Chauhan, S. S., and Kumar, V. (2021). A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, 80(6):8091–8126.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. *Proceedings of the IEEE International Conference on Neural Networks*, 4:1942–1948.

Konuhova, M. (2025). Induction motor dynamics regimes: A comprehensive study of mathematical models and validation. *Applied Sciences*, 15(3):1527.

Krishnan, R. (2001). *Electric Motor Drives: Modeling, Analysis, and Control*. Prentice Hall.

Moré, J. J. (1978). The Levenberg–Marquardt algorithm: Implementation and theory. In Watson, G. A., editor, *Numerical Analysis*, pages 105–116. Springer.

Niu, G., Dong, X., and Chen, Y. (2023). Motor fault diagnostics based on current signatures: A review. *IEEE Transactions on Instrumentation and Measurement*, 72:1–19.

Ocampo, A., Asiain, T. I., and Ruiz-Vega, D. (2014). Validación experimental del modelo matemático del motor de inducción. *ResearchGate*.

Pacheco Guerrero, J. P. (2023). Detección de desbalance y desalineamiento en motores de inducción bajo cargas oscilantes

usando corriente y transformada rápida de Fourier. Tesis de licenciatura, Universidad Autónoma de Querétaro, Facultad de Ingeniería.

Passino, K. M. (2002). Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control Systems Magazine*, 22(3):52–67.

Piotrowski, A. P., Napiorkowski, J. J., and Piotrowska, A. E. (2023). Particle swarm optimization or differential evolution—A comparison. *Engineering Applications of Artificial Intelligence*, 121:106008.

Rodríguez-Blanco, M. A., Golikov, V., Osorio-Sánchez, R., Samovarov, O., Ortiz-Torres, G., et al. (2022). Fault diagnosis of induction motor using D-Q simplified model and parity equations. *Energies*, 15(22):8372.

Sakthivel, V. P. and Subramanian, S. (2012). Bio-inspired optimization algorithms for parameter determination of three-phase induction motors. *COMPEL*, 31(2):528–551.

Santana, N. H. B., Yahyaoui, I., Oliveira, F. D. C., Amorim, A. E. A., Simonetti, D. S. L., and Rocha, H. R. O. (2025). Dynamic to static model comparison and hybrid metaheuristic optimization in induction motor parameter estimation. *Electronics*, 14(3):524.

Sengamalai, U., Anbazhagan, G., Thamizh Thentral, T. M., Vishnuram, P., Khurshaid, T., and Kamel, S. (2022). Three phase induction motor drive: A systematic review on dynamic modeling, parameter estimation, and control schemes. *Energies*, 15(21):8260.

Storn, R. and Price, K. (1997). Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359.

Tabora, J. M. et al. (2021). Assessing energy efficiency and power quality impacts due to high-efficiency motors operating under nonideal energy supply. *IEEE Access*, 9:121871–121882.

Tallam, R. M. et al. (2007). A survey of methods for detection of stator-related faults in induction machines. *IEEE Transactions on Industry Applications*, 43(4):920–933.

Toliyat, H. A. and Kliman, G. B. (2004). *Handbook of Electric Motors*. CRC Press, 2nd edition.

Valtierra-Rodriguez, M., Rivera-Guillen, J. R., Basurto-Hurtado, J. A., De-Santiago-Perez, J. J., Granados-Lieberman, D., and Amezquita-Sanchez, J. P. (2020). Convolutional neural network and motor current signature analysis during the transient state for detection of broken rotor bars in induction motors. *Sensors*, 20(13):3721.

Vukašinović, J., Štatkić, S., Milovanović, M., and Perović, B. (2023). Combined method for the cage induction motor parameters estimation using two-stage PSO algorithm. *Electrical Engineering*, 105(5):2703–2715.

Véliz-Tejo, A., Travieso-Torres, J. C., Peters, A. A., Mora, A., and Leiva-Silva, F. (2022). Normalized-model reference system for parameter estimation of induction motors. *Energies*, 15(13):4542.

Yang, X.-S. (2021). *Nature-Inspired Optimization Algorithms*. Academic Press, 2nd edition.

Zamudio-Ramírez, I., Osornio-Ríos, R. A., Antonino-Daviu, J. A., and Quijano-Lopez, A. (2020a). Smart-sensor for the automatic detection of electromechanical faults in induction motors based on the transient stray flux analysis. *Sensors*, 20(5):1477.

Zamudio-Ramírez, I., Ramirez-Nunez, J. A., Antonino-Daviu, J. A., Osornio-Rios, R. A., Quijano-López, A., Razik, H., and Romero-Troncoso, R. d. J. (2020b). Automatic diagnosis of electromechanical faults in induction motors based on the transient analysis of the stray flux via MUSIC methods. *IEEE Transactions on Industry Applications*, 56(4):3604–3613.

Zamudio Ramírez, I. (2019). Metodología para detección de falla de aislamiento en motores de inducción mediante FPGA. Tesis de maestría, Universidad Autónoma de Querétaro, Facultad de Ingeniería.

I. Código principal: versionFinal.py

El siguiente script en Python implementa la identificación paramétrica del motor de inducción utilizando el modelo d-q.

```
"""
Identificación paramétrica del motor de inducción -- modelo d-q
en referencia estacionaria (α-β) con excitación real en ambos ejes.

Tres métodos comparados:
  1. LM / TRF (Tradicional)
  2. PSO estándar (Kennedy & Eberhart, 1995)
  3. Híbrido DE + L-BFGS-B (Propuesto)

Motor: 380 V, 5.5 kW, 4 polos, 60 Hz, Y. (Battulga et al., 2024).

Metodología:
  - Transformada de Clarke completa con las 4 señales experimentales:
      vol1 (Vab) + vol2 (Vbc) -> Vα, Vβ
      cur1 (Ia)  + cur2 (Ib)  -> Iα, Iβ
  - Modelo ODE de 5 estados (ids, iqs, ψrα, ψrβ, ωr) excitado con
    Vα y Vβ reales simultáneamente.
  - Identificación (fit): minimización del ECM en el eje α (Iα).
  - Validación cruzada: predicción de Iβ con los mismos parámetros
    identificados, sin re-optimización.
  - Centrado DC en señales crudas antes de Clarke.
  - DS=8 para función objetivo; interpoladores a resolución completa.
  - Ecuaciones con Rs_eff = Rs + Lm²Rr/Lr², signos flujo {-, +}.

Resultados en carpeta "resultados".
"""

import numpy as np
import scipy.io as sio
import pandas as pd
from scipy.integrate import solve_ivp
from scipy.optimize import differential_evolution, minimize, least_squares
import matplotlib.pyplot as plt
import seaborn as sns
from pathlib import Path
import time
import warnings
from tqdm import tqdm
from multiprocessing import Pool, cpu_count
```

```
40
41 warnings.filterwarnings("ignore")
42
43 # 1. CONFIGURACIÓN
44 RANDOM_SEED = 42
45 DS          = 8
46
47 def _find_base_path(start: Path) -> Path:
48     for parent in [start, *start.parents]:
49         candidate = parent / "Datos" / "EECS" / "CommonSensor" / "StartingData"
50         if candidate.exists():
51             return candidate
52     raise FileNotFoundError(
53         f"No se encontró '/EECS/CommonSensor/StartingData/' desde {start}.")
54
55 BASE_PATH = _find_base_path(Path(__file__).resolve().parent)
56
57 FOLDERS = {
58     "Healthy":      "Sano",
59     "Looseness":    "Aflojamiento",
60     "Misalignment": "Desalineamiento",
61 }
62
63 RESULTS_DIR      = Path("resultados")
64 RESULTS_FILE     = RESULTS_DIR / "resultados_completos.csv"
65 CONTROL_PLOT_DIR = Path("resultados") / "graficas"
66
67 BOUNDS = [
68     (1.0,   15.0),   # Rs (Ω)
69     (1.0,   10.0),   # Rr (Ω)
70     (0.1,    0.6),   # Ls (H)
71     (0.1,    0.6),   # Lr (H)
72     (0.1,   0.55),   # Lm (H)
73     (0.005, 0.05),   # J  (kg·m²)
74     (1e-9,  0.01),   # B  (N·m·s/rad)
75 ]
76
77 # 2. PARÁMETROS NOMINALES (Krishnan, 2001)
78
79 def get_nominal_params() -> list:
80     Rs      = 2.152
81     Rr      = 1.241
82     Ls      = 0.207
83     Lr      = 0.207
```

```
    Lm_raw = 0.166 * 1.215
    lim_Lm = np.sqrt(Ls * Lr) * 0.999
    Lm     = min(Lm_raw, lim_Lm * 0.97)
    J      = 0.020
    B      = 0.002
    return [Rs, Rr, Ls, Lr, Lm, J, B]

# 3. CARGA DE DATOS + CLARKE

def load_specific_pair(vol1_path: Path):
    """
    Carga las 4 señales (vol1, vol2, cur1, cur2), aplica centrado DC y
    transformada de Clarke.

    v4: usa la señal COMPLETA -- sin truncar a 1.5 s.
    El número de muestras queda determinado por el mínimo de longitud
    entre los cuatro archivos del mismo registro.

    vol1 = Vab (Line Voltage A-B)   -> Vα = (2Vab + Vbc) / 3
    vol2 = Vbc (Line Voltage B-C)   -> Vβ = Vbc /sqrt3
    cur1 = Ia  (Phase Current A)    -> Iα = Ia
    cur2 = Ib  (Phase Current B)    -> Iβ = (Ia + 2Ib) /sqrt3
    """
    stem   = vol1_path.stem
    base   = stem.replace("vol1", "")
    parent = vol1_path.parent

    vol2_path = parent / (base + "vol2.mat")
    cur1_path = parent / (base + "cur1.mat")
    cur2_path = parent / (base + "cur2.mat")

    for p in [vol2_path, cur1_path, cur2_path]:
        if not p.exists():
            return (None,) * 6

    try:
        def _load(path: Path) -> np.ndarray:
            mat = sio.loadmat(path)
            key = [k for k in mat.keys() if not k.startswith("__")][0]
            return mat[key].flatten().astype(float)

        Vab = _load(vol1_path) * 100
        Vbc = _load(vol2_path) * 100
        Ia  = _load(cur1_path) * 100
```

```
        Ib  = _load(cur2_path) * 100

        fs = 10_240

        # -- Centrado DC --
        Vab -= np.mean(Vab)
        Vbc -= np.mean(Vbc)
        Ia  -= np.mean(Ia)
        Ib  -= np.mean(Ib)

        # -- Ventana 1.6-2.0 s (estado cuasi-estacionario, post-inrush) -
        # El arranque DOL ocurre típicamente en aprox. 0.9-1.5 s por archivo.
        # El período de silencio + inrush (~120 A pico) no puede reproducirse
        # con el modelo d-q de parámetros constantes. La ventana 1.6-2.0 s
        # captura el régimen cuasi-estacionario donde el modelo es válido.
        idx_start = int(1.60 * fs)
        idx_end   = int(2.00 * fs)
        n_max     = min(len(Vab), len(Vbc), len(Ia), len(Ib))
        idx_end   = min(idx_end, n_max)

        if idx_end <= idx_start:
            print(f"    [AVISO] {vol1_path.name}: señal demasiado corta para ventana 1.6-
                2.0 s")
            return (None,) * 6

        Vab = Vab[idx_start:idx_end]
        Vbc = Vbc[idx_start:idx_end]
        Ia  = Ia[idx_start:idx_end]
        Ib  = Ib[idx_start:idx_end]
        n   = len(Vab)

        # -- Transformada de Clarke --
        V_alpha = (2.0 * Vab + Vbc) / 3.0
        V_beta  = Vbc / np.sqrt(3.0)
        I_alpha = Ia
        I_beta  = (Ia + 2.0 * Ib) / np.sqrt(3.0)

        t = np.arange(n) / fs   # tiempo (0 -> 0.4 s)
        print(f"    [{vol1_path.name}] ventana 1.6-2.0 s: {n} muestras ({n/fs:.3f} s)")
        return t, V_alpha, V_beta, I_alpha, I_beta, fs

    except Exception as e:
        print(f"    ERROR cargando {vol1_path.name}: {e}")
        return (None,) * 6
```

```
171
172 # 4. MODELO ODE + FUNCIONES DE COSTO
173
174 def motor_ode(t, y, params, vd_interp, vq_interp):
175     """
176     Modelo d-q completo en referencia estacionaria (α-β).
177     Rs_eff = Rs + Lm²Rr/Lr². Flujos: -Pωr·ψrβ (α), +Pωr·ψrα (β).
178     """
179     Rs, Rr, Ls, Lr, Lm, J, B = params
180     sigma = max(1.0 - (Lm**2 / (Ls * Lr)), 1e-4)
181     ids, iqs, phird, phirq, wr = y
182     vds, vqs = vd_interp(t), vq_interp(t)
183     P = 2
184     Rs_eff = Rs + (Lm**2 * Rr) / (Lr**2)
185
186     dids = (1.0 / (sigma * Ls)) * (
187         vds - Rs_eff * ids
188         + (Lm * Rr / Lr**2) * phird
189         + P * wr * (Lm / Lr) * phirq)
190     diqs = (1.0 / (sigma * Ls)) * (
191         vqs - Rs_eff * iqs
192         + (Lm * Rr / Lr**2) * phirq
193         - P * wr * (Lm / Lr) * phird)
194
195     dphird = (Lm * Rr / Lr) * ids - (Rr / Lr) * phird - P * wr * phirq
196     dphirq = (Lm * Rr / Lr) * iqs - (Rr / Lr) * phirq + P * wr * phird
197
198     Te  = 1.5 * P * (Lm / Lr) * (phird * iqs - phirq * ids)
199     TL  = 0.0005 * wr**2
200     dwr = (1.0 / J) * (Te - TL - B * wr)
201     return [dids, diqs, dphird, dphirq, dwr]
202
203
204 def _simulate(params, t_eval, vd_func, vq_func):
205     try:
206         sol = solve_ivp(
207             motor_ode, [t_eval[0], t_eval[-1]], [0, 0, 0, 0, 0],
208             args=(params, vd_func, vq_func),
209             t_eval=t_eval, method="RK23", max_step=2e-3)
210         return sol if sol.status == 0 else None
211     except Exception:
212         return None
213
214
```

```python
def objective_alpha(params, t_eval, I_alpha_ref, vd_func, vq_func):
    """ECM solo en eje α (identificación)."""
    Rs, Rr, Ls, Lr, Lm, J, B = params
    if any(p <= 0 for p in [Rs, Rr, Ls, Lr, Lm, J]) or B < 0:
        return 1e6
    if Lm >= np.sqrt(Ls * Lr) * 0.999:
        return 1e6
    sol = _simulate(params, t_eval, vd_func, vq_func)
    if sol is None:
        return 1e6
    n = min(sol.y.shape[1], len(I_alpha_ref))
    return float(np.mean((sol.y[0][:n] - I_alpha_ref[:n])**2))


def residuals_alpha(params, t_eval, I_alpha_ref, vd_func, vq_func):
    """Residuos en eje α para LM/TRF."""
    Rs, Rr, Ls, Lr, Lm, J, B = params
    N   = len(I_alpha_ref)
    bad = np.ones(N) * 1e3
    if any(p <= 0 for p in [Rs, Rr, Ls, Lr, Lm, J]) or B < 0:
        return bad
    if Lm >= np.sqrt(Ls * Lr) * 0.999:
        return bad
    sol = _simulate(params, t_eval, vd_func, vq_func)
    if sol is None:
        return bad
    n   = min(sol.y.shape[1], N)
    res = sol.y[0][:n] - I_alpha_ref[:n]
    if n < N:
        res = np.concatenate([res, np.zeros(N - n)])
    return res

# 5-6. MÉTODOS DE OPTIMIZACIÓN

def run_levenberg_marquardt(t_eval, I_alpha, vd_func, vq_func):
    x0 = get_nominal_params()
    lb = [b[0] for b in BOUNDS]
    ub = [b[1] for b in BOUNDS]
    result = least_squares(
        residuals_alpha, x0,
        args=(t_eval, I_alpha, vd_func, vq_func),
        method="trf", bounds=(lb, ub),
        max_nfev=500, ftol=1e-10, xtol=1e-10, gtol=1e-10, verbose=0)
    mse = objective_alpha(result.x, t_eval, I_alpha, vd_func, vq_func)
```

```
    return result.x, mse


def run_simple_pso(func, bounds, args=(), popsize=12, maxiter=20, seed=None):
    rng = np.random.default_rng(seed)
    dim = len(bounds)
    lb  = np.array([b[0] for b in bounds])
    ub  = np.array([b[1] for b in bounds])
    X     = lb + (ub - lb) * rng.random((popsize, dim))
    V_vel = np.zeros((popsize, dim))
    P_pos = X.copy()
    P_fit = np.array([func(x, *args) for x in X])
    G     = P_pos[np.argmin(P_fit)].copy()
    G_fit = float(np.min(P_fit))
    w = 0.5; c1 = 1.5; c2 = 1.5
    for _ in range(maxiter):
        r1 = rng.random((popsize, dim))
        r2 = rng.random((popsize, dim))
        V_vel = w*V_vel + c1*r1*(P_pos - X) + c2*r2*(G - X)
        X = np.clip(X + V_vel, lb, ub)
        fits = np.array([func(x, *args) for x in X])
        mask = fits < P_fit
        P_pos[mask] = X[mask]
        P_fit[mask] = fits[mask]
        best_idx = int(np.argmin(fits))
        if fits[best_idx] < G_fit:
            G     = X[best_idx].copy()
            G_fit = float(fits[best_idx])
    return G, G_fit

# 7. MÉTRICAS DE VALIDACIÓN

def compute_metrics(params, t_eval, I_alpha_ref, I_beta_ref, vd_func, vq_func):
    """
    Calcula métricas de fit (eje α) y validación cruzada (eje β).
    Retorna dict con ECM, R², MAPE y MAE para ambos ejes.
    """
    sol = _simulate(params, t_eval, vd_func, vq_func)
    if sol is None:
        return None

    n = min(sol.y.shape[1], len(I_alpha_ref), len(I_beta_ref))
    ids_sim = sol.y[0][:n]
    iqs_sim = sol.y[1][:n]
```

```
    ia_ref  = I_alpha_ref[:n]
    ib_ref  = I_beta_ref[:n]

    metrics = {}

    for axis, sim, ref in [("alpha", ids_sim, ia_ref),
                           ("beta",  iqs_sim, ib_ref)]:
        mse    = float(np.mean((sim - ref)**2))
        mae    = float(np.mean(np.abs(sim - ref)))
        ss_res = np.sum((sim - ref)**2)
        ss_tot = np.sum((ref - np.mean(ref))**2)
        r2     = 1.0 - ss_res / ss_tot if ss_tot > 0 else 0.0
        mask   = np.abs(ref) > 0.5
        mape   = float(np.mean(np.abs((sim[mask] - ref[mask]) / ref[mask]))) * 100 \
                 if np.any(mask) else 0.0

        metrics[f"ECM_{axis}"]  = mse
        metrics[f"R2_{axis}"]   = r2
        metrics[f"MAE_{axis}"]  = mae
        metrics[f"MAPE_{axis}"] = mape

    metrics["wr_final"] = float(sol.y[4][-1])
    return metrics

# 8. WORKER
class _Interp:
    def __init__(self, arr, fs):
        self.arr, self.fs, self.n = arr, fs, len(arr)
    def __call__(self, t_val):
        idx = int(t_val * self.fs)
        return float(self.arr[min(idx, self.n - 1)])


def process_file(args):
    fpath, label, file_index = args

    t, V_alpha, V_beta, I_alpha, I_beta, fs = load_specific_pair(fpath)
    if t is None:
        return None
    if np.max(np.abs(I_alpha)) < 0.1:
        return None

    vd_func = _Interp(V_alpha, fs)
    vq_func = _Interp(V_beta,  fs)
```

```
347
348     t_ds       = t[::DS]
349     I_alpha_ds = I_alpha[::DS]
350     I_beta_ds  = I_beta[::DS]
351
352     seed = RANDOM_SEED + file_index
353
354     # -- Método 1: LM / TRF --
355     params_lm, mse_lm = run_levenberg_marquardt(
356         t_ds, I_alpha_ds, vd_func, vq_func)
357
358     # -- Método 2: PSO --
359     params_pso, mse_pso = run_simple_pso(
360         objective_alpha, BOUNDS,
361         args=(t_ds, I_alpha_ds, vd_func, vq_func),
362         popsize=12, maxiter=20, seed=seed)
363
364     # -- Método 3: Híbrido (DE + L-BFGS-B) --
365     # Configuración según Tabla 3.1: Np=10, G=15, F=0.8, Cr=0.9
366     res_de = differential_evolution(
367         objective_alpha, BOUNDS,
368         args=(t_ds, I_alpha_ds, vd_func, vq_func),
369         strategy="best1bin", maxiter=15, popsize=10,
370         mutation=0.8, recombination=0.9,
371         polish=False, seed=seed)
372     res_hybrid = minimize(
373         objective_alpha, res_de.x,
374         args=(t_ds, I_alpha_ds, vd_func, vq_func),
375         method="L-BFGS-B", bounds=BOUNDS)
376     mse_hybrid    = float(res_hybrid.fun)
377     params_hybrid = res_hybrid.x
378
379     # -- Métricas completas (fit α + validación β) --
380     met_lm  = compute_metrics(params_lm,     t_ds, I_alpha_ds, I_beta_ds, vd_func,
        vq_func)
381     met_pso = compute_metrics(params_pso,    t_ds, I_alpha_ds, I_beta_ds, vd_func,
        vq_func)
382     met_hyb = compute_metrics(params_hybrid, t_ds, I_alpha_ds, I_beta_ds, vd_func,
        vq_func)
383
384     mejora_vs_lm  = (mse_lm  - mse_hybrid) / mse_lm  * 100 if mse_lm  > 0 else 0.0
385     mejora_vs_pso = (mse_pso - mse_hybrid) / mse_pso * 100 if mse_pso > 0 else 0.0
386
387     n_muestras = len(t)
```

```
    duracion_s = t[-1]

    print(
        f"-> {fpath.name[:28]:28s} | n={n_muestras} ({duracion_s:.1f}s) "
        f"| LM:{mse_lm:.2e} | PSO:{mse_pso:.2e} | Hyb:{mse_hybrid:.2e} "
        f"| ΔLM:{mejora_vs_lm:+.1f}%  ΔPSO:{mejora_vs_pso:+.1f}%"
        f"  R²α:{met_hyb['R2_alpha']:.3f}  R²β:{met_hyb['R2_beta']:.3f}")

    nom = get_nominal_params()

    row = {
        "Condicion":      label,
        "Archivo":        fpath.name,
        "N_muestras":     n_muestras,
        "Duracion_s":     round(duracion_s, 3),
        "ECM_Tradicional": mse_lm,
        "ECM_PSO":         mse_pso,
        "ECM_Hibrido":     mse_hybrid,
        "Mejora_vs_Trad":  mejora_vs_lm,
        "Mejora_vs_PSO":   mejora_vs_pso,
    }

    for prefix, met in [("Trad", met_lm), ("PSO", met_pso), ("Hyb", met_hyb)]:
        if met:
            for k, v in met.items():
                row[f"{prefix}_{k}"] = v

    for prefix, params in [("Hyb", params_hybrid), ("PSO", params_pso), ("Trad",
        params_lm)]:
        for name, val in zip(["Rs","Rr","Ls","Lr","Lm","J","B"], params):
            row[f"{prefix}_{name}"] = val

    row["Nom_Rs"] = nom[0]
    row["Nom_Rr"] = nom[1]
    row["Nom_Ls"] = nom[2]

    return row

# 9. MAIN
if __name__ == "__main__":
    np.random.seed(RANDOM_SEED)
    start_time = time.time()

    RESULTS_DIR.mkdir(parents=True, exist_ok=True)
```

```
    CONTROL_PLOT_DIR.mkdir(parents=True, exist_ok=True)

    all_tasks = []
    for folder_name, label in FOLDERS.items():
        folder_path = BASE_PATH / folder_name
        if folder_path.exists():
            for f in sorted(folder_path.glob("*_vol1.mat")):
                all_tasks.append((f, label, len(all_tasks)))

    if not all_tasks:
        print("[AVISO] No se encontraron archivos *_vol1.mat.")
        raise SystemExit(1)

    print(f"=== Identificación paramétrica v4 -- señal completa ===")
    print(f"    {len(all_tasks)} archivos | Clarke: vol1+vol2+cur1+cur2")
    print(f"    Fit: eje α (Iα)  |  Validación cruzada: eje β (Iβ)")
    print(f"    DS={DS} | RK23, max_step=2e-3")
    print(f"    LM: nfev<=500 | PSO: 12×20 | DE: Np=10×G=15, F=0.8, Cr=0.9 + L-BFGS-B")
    print(f"    Ventana: 1.6-2.0 s por registro (4,096 muestras, post-inrush)")
    print()

    NCPU = cpu_count()
    print(f"    Paralelismo: {NCPU} cores\n")

    with Pool(processes=NCPU) as pool:
        results_raw = list(tqdm(
            pool.imap_unordered(process_file, all_tasks),
            total=len(all_tasks),
            desc="Procesando", unit="archivo",
            colour="green", dynamic_ncols=True,
        ))
    results = [r for r in results_raw if r is not None]

    if not results:
        print("[AVISO] Ningún archivo produjo resultados válidos.")
        raise SystemExit(1)

    df = pd.DataFrame(results)
    df.to_csv(RESULTS_FILE, index=False)
    print(f"\nCSV: {RESULTS_FILE}  ({df.shape[0]}×{df.shape[1]})")

    # -- Resumen ------
    print("\n-- Duración promedio de señales por condición --")
    for cond in ["Sano", "Aflojamiento", "Desalineamiento"]:
```

```
        sub = df[df["Condicion"] == cond]
        if sub.empty: continue
        print(f"  {cond}: {sub['Duracion_s'].mean():.2f} s promedio "
              f"({sub['N_muestras'].mean():.0f} muestras)")

    print("\n-- ECM eje α (identificación) por condición --")
    for cond in ["Sano", "Aflojamiento", "Desalineamiento"]:
        sub = df[df["Condicion"] == cond]
        if sub.empty: continue
        print(f"  {cond} (n={len(sub)})")
        for col, lbl in [("ECM_Tradicional","LM "),("ECM_PSO","PSO"),("ECM_Hibrido","Hyb")]:
            print(f"    {lbl}: media={sub[col].mean():.3e}  std={sub[col].std():.3e}")

    print("\n-- R² Híbrido (α = fit, β = validación cruzada) --")
    for cond in ["Sano", "Aflojamiento", "Desalineamiento"]:
        sub = df[df["Condicion"] == cond]
        if sub.empty: continue
        print(f"  {cond}: R²_α={sub['Hyb_R2_alpha'].mean():.3f}  "
              f"R²_β={sub['Hyb_R2_beta'].mean():.3f}")

    print(f"\n-- Mejora global --")
    print(f"  Híbrido vs LM:  {df['Mejora_vs_Trad'].mean():.1f}%")
    print(f"  Híbrido vs PSO: {df['Mejora_vs_PSO'].mean():.1f}%")
    print(f"\n=== Fin ({time.time()-start_time:.1f} s) ===")

    # -- Gráfica ECM --
    rename = {"ECM_Tradicional":"LM (Trad.)","ECM_PSO":"PSO","ECM_Hibrido":"Híbrido"}
    df_m = df.melt(id_vars=["Condicion"], value_vars=list(rename.keys()),
                   var_name="Metodo", value_name="ECM")
    df_m["Metodo"] = df_m["Metodo"].map(rename)
    colores = {"LM (Trad.)":"#e74c3c","PSO":"#f39c12","Híbrido":"#2ecc71"}

    sns.set_style("whitegrid")
    fig, ax = plt.subplots(figsize=(10, 5))
    sns.boxplot(x="Condicion", y="ECM", hue="Metodo", data=df_m,
                palette=colores, ax=ax)
    ax.set_yscale("log")
    ax.set_title("v4 -- ventana 1.6-2.0 s | ECM eje α (identificación)")
    ax.set_xlabel("Condición operativa")
    ax.set_ylabel("ECM eje α (log)")
    fig.savefig(CONTROL_PLOT_DIR / "control_ecm_alpha.svg", bbox_inches="tight")
    plt.close(fig)

```

```
    # -- Gráfica R² validación cruzada --
    fig2, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 5))
    for ax, col, title in [(ax1, "Hyb_R2_alpha", "R² eje α (fit)"),
                           (ax2, "Hyb_R2_beta",  "R² eje β (validación)")]:
        sns.barplot(x="Condicion", y=col, data=df,
                    palette=["#2ecc71","#f39c12","#e74c3c"], ax=ax)
        ax.set_title(title)
        ax.set_ylim(0, 1)
        ax.set_ylabel("R²")
        ax.set_xlabel("")
    fig2.suptitle("v4 -- ventana 1.6-2.0 s | Validación cruzada Híbrido", fontsize=13)
    fig2.tight_layout()
    fig2.savefig(CONTROL_PLOT_DIR / "validacion_cruzada_r2.svg", bbox_inches="tight")
    plt.close(fig2)
    print(f"Gráficas guardadas en {CONTROL_PLOT_DIR}")
```

II. Código exploracion sintetica: exploracion_sintetica.py

El siguiente script en Python implementa la exploracion sintetica de los diferentes modelos.

```
"""
Etapa exploratoria -- comparación de estrategias de optimización
sobre señal sintética con parámetros verdaderos conocidos.

Métodos evaluados:
  1. LM/TRF  -- mínimos cuadrados no lineales (punto inicial lejano)
  2. PSO     -- enjambre de partículas estándar
  3. DE      -- Evolución Diferencial sola
  4. Híbrido  -- DE + L-BFGS-B (propuesto)

Genera:
  - Tabla CSV con ECM y R² por método
  - Figura comparativa de barras
  - Figura de señal simulada vs referencia por método
"""

import numpy as np
from scipy.integrate import solve_ivp
from scipy.optimize import differential_evolution, minimize, least_squares
import matplotlib.pyplot as plt
import pandas as pd
import time
from pathlib import Path

# -- Configuración --
RANDOM_SEED = 42
DS          = 8       # downsampling para función objetivo
T_SIM       = 0.4     # duración señal sintética (s)
RUIDO_STD   = 0.8     # desviación estándar del ruido gaussiano (A)
P           = 2       # pares de polos

# Parámetros VERDADEROS (conocidos) -- motor 380V 5.5kW
PARAMS_TRUE = dict(Rs=2.5, Rr=1.8, Ls=0.22, Lr=0.35,
                   Lm=0.28, J=0.035, B=0.006)

# Punto inicial LM -- datos de placa (alejado de la solución)
X0_LM = [2.152, 1.241, 0.207, 0.207, 0.201, 0.020, 0.002]

# Espacio de búsqueda
BOUNDS = [(1.0, 15.0), (1.0, 10.0), (0.1, 0.6),
```

```
          (0.1,  0.6), (0.1, 0.55), (0.005, 0.05), (1e-9, 0.01)]

RESULTS_DIR = Path("resultados_exploracion")
RESULTS_DIR.mkdir(exist_ok=True)

# -- Señal de excitación sintética --
fs = 10_240
t  = np.arange(int(T_SIM * fs)) / fs
Va = 220 * np.sqrt(2) * np.sin(2 * np.pi * 60 * t)
Vb = 220 * np.sqrt(2) * np.sin(2 * np.pi * 60 * t - np.pi / 2)

class _Interp:
    def __init__(self, arr, fs_):
        self.arr = arr; self.fs = fs_; self.n = len(arr)
    def __call__(self, t_val):
        idx = int(t_val * self.fs)
        return float(self.arr[min(idx, self.n - 1)])

iVa = _Interp(Va, fs); iVb = _Interp(Vb, fs)

# -- Modelo ODE d-q --
def simulate(params, t_eval):
    Rs, Rr, Ls, Lr, Lm, J, B = params
    sigma  = max(1.0 - Lm**2 / (Ls * Lr), 1e-4)
    Rs_eff = Rs + Lm**2 * Rr / Lr**2
    def ode(t, x):
        ids, iqs, pra, prb, wr = x
        Va_ = iVa(t); Vb_ = iVb(t)
        dids = (1/(sigma*Ls))*(Va_-Rs_eff*ids+(Lm*Rr/Lr**2)*pra+P*wr*(Lm/Lr)*prb)
        diqs = (1/(sigma*Ls))*(Vb_-Rs_eff*iqs+(Lm*Rr/Lr**2)*prb-P*wr*(Lm/Lr)*pra)
        dpra = (Lm*Rr/Lr)*ids-(Rr/Lr)*pra-P*wr*prb
        dprb = (Lm*Rr/Lr)*iqs-(Rr/Lr)*prb+P*wr*pra
        Te   = 1.5*P*(Lm/Lr)*(pra*iqs-prb*ids)
        dwr  = (1/J)*(Te-0.0005*wr**2-B*wr)
        return [dids, diqs, dpra, dprb, dwr]
    sol = solve_ivp(ode, [t_eval[0], t_eval[-1]], [0,0,0,0,0],
                    method='RK23', max_step=2e-3, t_eval=t_eval)
    return sol.y[0] if sol.status == 0 else np.zeros_like(t_eval)

# Generar señal de referencia con ruido
t_ds   = t[::DS]
I_true = simulate(list(PARAMS_TRUE.values()), t_ds)
rng    = np.random.default_rng(RANDOM_SEED)
I_ref  = I_true + rng.normal(0, RUIDO_STD, len(t_ds))
```

```
85
86 print(f"Señal sintética: {len(t)} muestras | {len(t_ds)} puntos de optimización (DS={DS})
       ")
87 print(f"Parámetros verdaderos: {PARAMS_TRUE}\n")
88
89 # -- Funciones de evaluación --
90 def obj(params):
91     Rs, Rr, Ls, Lr, Lm, J, B = params
92     if any(p <= 0 for p in [Rs,Rr,Ls,Lr,Lm,J]) or B < 0: return 1e6
93     if Lm >= np.sqrt(Ls * Lr) * 0.999: return 1e6
94     s = simulate(list(params), t_ds)
95     return float(np.mean((s - I_ref)**2))
96
97 def compute_r2(params):
98     s = simulate(list(params), t_ds)
99     ss_r = np.sum((s - I_ref)**2)
100    ss_t = np.sum((I_ref - np.mean(I_ref))**2)
101    return 1.0 - ss_r / ss_t if ss_t > 0 else 0.0
102
103 def error_recuperacion(params):
104    """Error relativo promedio en recuperación de parámetros"""
105    p_hat = np.array(list(params))
106    p_true = np.array(list(PARAMS_TRUE.values()))
107    return float(np.mean(np.abs(p_hat - p_true) / np.abs(p_true)) * 100)
108
109 results = []
110
111 # -- Método 1: LM/TRF --
112 print("Método 1: LM/TRF...", end=' ', flush=True)
113 t0 = time.time()
114 res_lm = least_squares(
115     lambda p: simulate(list(p), t_ds) - I_ref, X0_LM,
116     method='trf',
117     bounds=([b[0] for b in BOUNDS], [b[1] for b in BOUNDS]),
118     max_nfev=500, ftol=1e-10, xtol=1e-10, gtol=1e-10)
119 e_lm = obj(res_lm.x); r_lm = compute_r2(res_lm.x)
120 err_lm = error_recuperacion(res_lm.x); dt_lm = time.time()-t0
121 print(f"ECM={e_lm:.2f}  R²={r_lm:.4f}  Err.param={err_lm:.1f}%  ({dt_lm:.0f}s)")
122 results.append({'Método':'LM/TRF (Convencional)', 'ECM':e_lm,
123                 'R²':r_lm, 'Error param. (%)':err_lm, 'Tiempo (s)':dt_lm})
124
125 # -- Método 2: PSO --
126 print("Método 2: PSO...", end=' ', flush=True)
127 t0 = time.time()
```

```
rng2 = np.random.default_rng(RANDOM_SEED)
lb = np.array([b[0] for b in BOUNDS]); ub = np.array([b[1] for b in BOUNDS])
pop = 12; X = lb+(ub-lb)*rng2.random((pop,7)); V_ = np.zeros((pop,7))
Pp = X.copy(); Pf = np.array([obj(x) for x in X])
G  = Pp[np.argmin(Pf)].copy(); Gf = float(np.min(Pf))
for _ in range(20):
    r1=rng2.random((pop,7)); r2_=rng2.random((pop,7))
    V_ = 0.5*V_+1.5*r1*(Pp-X)+1.5*r2_*(G-X)
    X  = np.clip(X+V_, lb, ub)
    fs_ = np.array([obj(x) for x in X])
    mask = fs_<Pf; Pp[mask]=X[mask]; Pf[mask]=fs_[mask]
    bi = int(np.argmin(fs_))
    if fs_[bi] < Gf: G=X[bi].copy(); Gf=float(fs_[bi])
e_pso=obj(G); r_pso=compute_r2(G); err_pso=error_recuperacion(G); dt_pso=time.time()-t0
print(f"ECM={e_pso:.2f}  R²={r_pso:.4f}  Err.param={err_pso:.1f}%  ({dt_pso:.0f}s)")
results.append({'Método':'PSO estándar', 'ECM':e_pso,
                'R²':r_pso, 'Error param. (%)':err_pso, 'Tiempo (s)':dt_pso})

# -- Método 3: DE --
print("Método 3: DE solo...", end=' ', flush=True)
t0 = time.time()
res_de = differential_evolution(obj, BOUNDS, strategy='best1bin',
    maxiter=15, popsize=10, seed=RANDOM_SEED, polish=False)
e_de=obj(res_de.x); r_de=compute_r2(res_de.x); err_de=error_recuperacion(res_de.x)
dt_de=time.time()-t0
print(f"ECM={e_de:.2f}  R²={r_de:.4f}  Err.param={err_de:.1f}%  ({dt_de:.0f}s)")
results.append({'Método':'DE (Evolución Diferencial)', 'ECM':e_de,
                'R²':r_de, 'Error param. (%)':err_de, 'Tiempo (s)':dt_de})

# -- Método 4: Híbrido DE + L-BFGS-B --
print("Método 4: Híbrido DE+L-BFGS-B...", end=' ', flush=True)
t0 = time.time()
res_de2 = differential_evolution(
    obj, BOUNDS,
    strategy='best1bin',
    maxiter=15,           # G = 15 generaciones (Tabla 3.1)
    popsize=10,           # Np = 10 individuos  (Tabla 3.1)
    mutation=0.8,         # F  = 0.8            (Tabla 3.1)
    recombination=0.9,    # Cr = 0.9            (Tabla 3.1)
    seed=RANDOM_SEED,
    polish=False          # refinamiento manual con L-BFGS-B
)
res_hyb = minimize(obj, res_de2.x, method='L-BFGS-B', bounds=BOUNDS)
e_hyb=obj(res_hyb.x); r_hyb=compute_r2(res_hyb.x)
```

```
err_hyb=error_recuperacion(res_hyb.x); dt_hyb=time.time()-t0
print(f"ECM={e_hyb:.2f}  R²={r_hyb:.4f}  Err.param={err_hyb:.1f}%  ({dt_hyb:.0f}s)")
results.append({'Método':'Híbrido DE+L-BFGS-B (Propuesto)', 'ECM':e_hyb,
                'R²':r_hyb, 'Error param. (%)':err_hyb, 'Tiempo (s)':dt_hyb})

# -- Tabla --
df = pd.DataFrame(results)
df['ECM']    = df['ECM'].round(3)
df['R²']     = df['R²'].round(4)
df['Error param. (%)'] = df['Error param. (%)'].round(1)
df['Tiempo (s)'] = df['Tiempo (s)'].round(1)
print("\n=== TABLA COMPARATIVA ===")
print(df.to_string(index=False))
df.to_csv(RESULTS_DIR / 'tabla_exploracion.csv', index=False)

# -- Figura 1: Barras ECM y R² --
plt.rcParams.update({'font.family':'serif','font.size':10,
                     'axes.spines.top':False,'axes.spines.right':False})
fig, axes = plt.subplots(1, 2, figsize=(12, 4.5))
colores = ['#C0392B','#E67E22','#2980B9','#27AE60']
etiq    = ['LM/TRF','PSO','DE','Híbrido\nDE+LBFGS']
ecms    = [e_lm, e_pso, e_de, e_hyb]
r2s     = [r_lm, r_pso, r_de, r_hyb]

ax = axes[0]
bars = ax.bar(etiq, ecms, color=colores, alpha=0.82, width=0.55)
ax.set_ylabel('ECM (A²)'); ax.grid(axis='y', alpha=0.3)
ax.set_title('Error cuadrático medio\npor método (señal sintética)')
for bar, val in zip(bars, ecms):
    ax.text(bar.get_x()+bar.get_width()/2, bar.get_height()*1.02,
            f'{val:.2f}', ha='center', va='bottom', fontsize=9, fontweight='bold')

ax = axes[1]
bars = ax.bar(etiq, r2s, color=colores, alpha=0.82, width=0.55)
ax.set_ylim(0, 1.05); ax.set_ylabel('$R^2$'); ax.grid(axis='y', alpha=0.3)
ax.set_title('Coeficiente de determinación\npor método (señal sintética)')
ax.axhline(1.0, color='gray', linestyle='--', linewidth=0.8, alpha=0.5)
for bar, val in zip(bars, r2s):
    ax.text(bar.get_x()+bar.get_width()/2, bar.get_height()+0.01,
            f'{val:.4f}', ha='center', va='bottom', fontsize=9, fontweight='bold')

fig.suptitle('Etapa exploratoria -- comparación de estrategias de optimización\n'
             f'Señal sintética con ruido gaussiano (σ={RUIDO_STD} A) | '
             f'Parámetros verdaderos conocidos | DS={DS}',
```

```
            fontsize=10, fontweight='bold')
plt.tight_layout()
plt.savefig(RESULTS_DIR / 'Fig_Exploracion_Barras.pdf', bbox_inches='tight')
plt.savefig(RESULTS_DIR / 'Fig_Exploracion_Barras.png', bbox_inches='tight', dpi=150)
plt.close(); print("\nFig barras guardada.")

# -- Figura 2: Señal simulada vs referencia --
fig, axes = plt.subplots(2, 2, figsize=(12, 8), sharex=True, sharey=True)
configs = [
    (axes[0,0], 'A) LM/TRF (Convencional)',        res_lm.x, '#C0392B', e_lm,  r_lm),
    (axes[0,1], 'B) PSO estándar',                  G,        '#E67E22', e_pso, r_pso),
    (axes[1,0], 'C) DE solo',                       res_de.x, '#2980B9', e_de,  r_de),
    (axes[1,1], 'D) Híbrido DE+L-BFGS-B (Propuesto)', res_hyb.x,'#27AE60',e_hyb,r_hyb),
]
t_full = t[::DS]
for ax, title, params, color, e, r in configs:
    i_sim = simulate(list(params), t_full)
    ax.plot(t_full, I_ref,  color='#BDC3C7', lw=0.8, label='Referencia (sintética)')
    ax.plot(t_full, i_sim,  color=color,     lw=1.0, alpha=0.85, label='Simulada')
    ax.text(0.04, 0.90, f'ECM={e:.2f}\n$R^2$={r:.4f}',
            transform=ax.transAxes, fontsize=9, color=color, fontweight='bold',
            bbox=dict(boxstyle='round,pad=0.3', facecolor='white',
                      edgecolor=color, alpha=0.85), va='top')
    ax.set_title(title, fontweight='bold')
    ax.set_xlabel('Tiempo (s)'); ax.set_ylabel('$I_\\alpha$ (A)')
    ax.legend(fontsize=8, framealpha=0.8)
    ax.grid(True, alpha=0.2)
    ax.spines['top'].set_visible(False); ax.spines['right'].set_visible(False)

fig.suptitle('Etapa exploratoria -- corriente simulada vs. referencia sintética\n'
             'Parámetros verdaderos conocidos | Punto inicial lejano a la solución',
             fontsize=10, fontweight='bold')
plt.tight_layout()
plt.savefig(RESULTS_DIR / 'Fig_Exploracion_Corrientes.pdf', bbox_inches='tight')
plt.savefig(RESULTS_DIR / 'Fig_Exploracion_Corrientes.png', bbox_inches='tight', dpi=150)
plt.close(); print("Fig corrientes guardada.")

print(f"\nResultados en: {RESULTS_DIR}/")
print("Archivos: tabla_exploracion.csv | Fig_Exploracion_Barras.pdf |
    Fig_Exploracion_Corrientes.pdf")
```

III. Physics-Informed Parametric Fault Signatures Beyond Black-Box Classification in Induction Motors

Physics-Informed Parametric Fault Signatures Beyond Black-Box Classification in Induction Motors

1st Gustavo Medina Vega
Facultad de Ingeniería
Universidad Autónoma de Querétaro
San Juan del Río, QRO, Mexico
0009-0002-6621-1188

2nd Israel Zamudio-Ramirez
Facultad de Ingeniería
Universidad Autónoma de Querétaro
San Juan del Río, QRO, Mexico
0000-0002-8499-3948

3rd Jose A. Antonino-Daviu
Instituto Tecnológico de la Energía
Universitat Politècnica de València
Valencia, Spain
0000-0003-1898-2228

4th Luis Morales-Velazquez
Facultad de Ingeniería
Universidad Autónoma de Querétaro
San Juan del Río, QRO, Mexico
0000-0003-1130-7131

***Abstract*—A Physics-Informed Neural Network (PINN) based on the Park dq model is proposed for the interpretable diagnosis of faults in induction motors. The goal is to replace black-box classification with a fault signature expressed directly in physical parameters carrying electrical and mechanical meaning: the PINN estimates seven such parameters from steady-state voltage and current measurements, so that each diagnosis is tied to a physical quantity rather than to an abstract learned feature. The methodology is evaluated on a laboratory-bench dataset with 24 operating conditions (healthy, unbalanced, and misaligned states across four drive frequencies and two mechanical load regimes) yielding 768 converged windows. The loss-function curvature analysis reveals that the identification problem is intrinsically better conditioned for the electromagnetic inductances than for the mechanical and resistive parameters: the former are two to three orders of magnitude more identifiable, while the latter are practically unobservable in steady state. The PINN thus reproduces, automatically and without explicit penalties, the model's structural identifiability constraints. Consistently with this ordering, of the parametric shifts that the physics predicts for each fault, 20 of 24 are identified in the correct direction, evidencing that the better-conditioned parameters carry a physically coherent fault signature.**

***Index Terms*—physics-informed neural networks, induction motor, fault diagnosis, dq model, parameter identification**

I. Introduction

Electric motors account for nearly 72% of the industrial sector's electricity consumption [1], [2], and induction motors (IMs) make up the majority of them, driving the pumps, fans, compressors, and conveyor belts that lie on the critical path of most production lines. As a result, an unexpected motor failure halts the entire process it drives, and the resulting downtime and lost production typically dominate the cost of the failure, making early and physically interpretable diagnosis of direct economic value. Current physics-based diagnostic methods offer interpretability but struggle with modeling uncertainty [3]; data-driven methods, on the other hand, achieve high accuracy but operate as black-box classifiers [4], [5].

PINNs [6], [7] embed the differential equations describing the system into the loss function, thereby combining both approaches. However, little has been explored regarding how the 7 IM parameters ($R_s, R_r, L_s, L_r, L_m, J, B$) shift under different fault or operating conditions when identified through a physics-informed framework [4]. This work closes that gap by embedding the Park-transformed dq circuit and the electromechanical model directly into the PINN loss function, and validating the identified parameters against physics-based fault predictions.

Within the physics-based family, model-based diagnosis relies on an explicit electrical model of the machine. Approaches built on the dq or coupled-circuit representation detect faults by monitoring deviations in model residuals or parameters [3], and winding-function models have been used to predict how air-gap eccentricity alters the machine inductances [8]–[10]. These methods are interpretable because every indicator maps to a physical quantity, but their accuracy hinges on how faithfully the assumed model matches the real machine, and unmodeled effects degrade the diagnosis.

Data-driven diagnosis, by contrast, learns fault patterns directly from measured signals. Motor Current Signature Analysis extracts fault-related harmonics from the stator current [4], while machine-learning classifiers achieve high accuracy on vibration and current features [5]. Their weakness is interpretability: the decision rests on abstract learned features rather than on physical quantities, which limits trust and diagnostic

This work was partially supported by the grant PID2024-155729OB-I00 funded by MICIU/AEI/10.13039/501100011033 and the European Union ERDF. The authors thank J. P. Pacheco Guerrero for providing the experimental dataset and the supporting code for its interpretation.

IV. Análisis comparativo mediante Inteligencia de Enjambre para la Identificación Paramétrica aplicada a motores de inducción

Análisis comparativo mediante Inteligencia de Enjambre para la Identificación Paramétrica aplicada a motores de inducción.

Comparative Analysis using Swarm Intelligence for Parameter Identification applied to Induction Motors.

Medina Vega, Gustavo y Morales Velázquez, Luis**
*Facultad de Ingeniería, Universidad Autónoma de Querétaro, Querétaro, México**
gmedina08@alumnos.uaq.mx; ORCID: https://orcid.org/0009-0002-6621-1188
luis.moralesv@uaq.mx; ORCID: https://orcid.org/0000-0003-1130-7131

Resumen: La eficiencia energética en sistemas industriales es un pilar fundamental para la sostenibilidad, en el cual los motores de inducción cumplen el papel como la principal fuerza motriz. Lograr una operación óptima de estos motores y una implementación de estrategias avanzadas de control dependen significativamente de la precisión con la que se cuente el modelo físico del motor. El presente trabajo presenta un análisis comparativo entre la Optimización por Enjambre de Partículas (PSO) y el Optimizador del Lobo Gris (GWO) para la identificación paramétrica de un modelo dinámico de quinto orden de un motor de inducción. Se utilizaron datos transitorios con ruido para emular condiciones realistas. Los resultados demuestran de manera cuantitativa la superioridad de GWO, el cual logro una precisión aproximadamente cinco veces mayor que PSO bajo las mismas condiciones. Se concluye que la selección de un algoritmo de optimización como GWO es un paso importante para futuras aplicaciones como la implementación de gemelos digitales para el monitoreo y diagnóstico.

Palabras clave: MEC; Motores de Inducción; Identificación de Parámetros; Inteligencia de Enjambre; Optimización Metaheurística; Optimizador del Lobo Gris

Abstract: Energy efficiency in industrial systems is a fundamental pillar for sustainability, in which induction motors (IM) fulfill the role as the main driving force. Achieving optimal operation of these motors and the implementation of advanced control strategies depend significantly on the accuracy of the motor's physical model. This work presents a comparative analysis between Particle Swarm Optimization (PSO) and the Grey Wolf Optimizer (GWO) for the parametric identification of a fifth-order dynamic model of an induction motor. Transient data with noise were used to emulate realistic conditions. The results quantitatively demonstrate the superiority of GWO, which achieved approximately five times greater accuracy than PSO under the same conditions. It is concluded that the selection of an optimization algorithm like GWO is an important step for accurate parameter identification in induction motors, laying the groundwork for future applications such as the implementation of digital twins for monitoring and diagnostics.

Keywords: MEC; Induction Motors; Parameter Identification; Swarm Intelligence; Metaheuristic Optimization; Grey Wolf Optimizer.

1. Introducción

Los Motores de Inducción (MI) formal la estructura central de la industria moderna, desde su importancia siendo pivotes en un amplio número de aplicaciones debido a su versatilidad, robustez y fiabilidad. Sin embargo, dichas ventajas han logrado una creciente demanda en la eficiencia energética y una búsqueda de rendimiento dinámico lo que ha impulsado a transicionar de un control convencional a técnicas avanzadas. Por ejemplo, el Control Orientado por Campo (FOC, por sus siglas en ingles Field Oriented control) y el Control Predictivo Basado en Modelo (MPC por sus siglas en ingles Model Predictive Control) [1]. La eficacia y aplicación de estos métodos depende de gran manera en la disponibilidad de un modelo matemático que pueda representar con precisión el comportamiento dinámico del motor. Una estimación deficiente/inexacta de los parámetros impacta directamente de manera negativa en el rendimiento del controlador [2].

Para resolver el problema de identificación paramétrica de motores de inducción actualmente es dominado por algoritmos de optimización

V. Optimización de Carteras de Inversión Financiera mediante un Modelo Híbrido GARCH y Algoritmo Genético

Congreso Internacional de Mecatrónica Control e Inteligencia Artificial (CIMCIA), UNAM, FESC, Estado de México, 2025

Optimización de Carteras de Inversión Financiera mediante un Modelo Híbrido GARCH y Algoritmo Genético

Gustavo Medina Vega
Facultad de Ingeniería
Universidad Autónoma de Querétaro
Querétaro, México
gmedina08@alumnos.uaq.mx

Luis Morales Velázquez
Facultad de Ingeniería
Universidad Autónoma de Querétaro
Querétaro, México
luis.moralesv@uaq.mx

***Resumen*—La optimización de carteras de inversión en mercados financieros presenta un desafío significativo debido a la volatilidad no estacionaria y a dinámicas complejas. Este trabajo propone y valida un sistema híbrido que aborda estas limitaciones mediante la integración de un modelo GARCH (Generalized Autoregressive Conditional Heteroskedasticity) para el pronóstico de la volatilidad condicional y un Algoritmo Genético (GA) para la optimización de la asignación de activos. El objetivo del sistema es maximizar el Ratio de Sortino, enfocándose explícitamente en la mitigación del riesgo de caída (downside risk). Para evaluar su eficacia, se implementó un riguroso protocolo de backtesting con una metodología de ventana móvil (walk-forward), asegurando la robustez de los resultados. Aplicado a una cartera de acciones de alta capitalización, el modelo GARCH-GA alcanzó un retorno anualizado del 38.51 %. La principal contribución es la validación empírica, mediante un riguroso protocolo de backtesting, de una arquitectura sinérgica que combina predicción econométrica y optimización evolutiva, ofreciendo un marco superior para la gestión activa de activos en entornos de mercado dinámicos.**

***Palabras clave*—Machine Learning, algoritmos genéticos, modelo GARCH, inteligencia computacional, optimización, finanzas cuantitativas, gestión de activos.**

I. Introducción

El problema central en finanzas cuantitativas consiste en construir carteras de inversión que maximicen la rentabilidad ajustada al riesgo en un entorno de mercado inherentemente volátil y no lineal. Los modelos estáticos tradicionales resultan insuficientes en este entorno, lo que exige el desarrollo de sistemas dinámicos que se adapten a las condiciones cambiantes. Este artículo aborda dicho problema proponiendo un sistema híbrido que integra técnicas econométricas y de inteligencia computacional para gestionar activamente el riesgo de caída (*downside risk*), un factor crucial para la preservación del capital a largo plazo.

El estado del arte ha evolucionado significativamente desde la Teoría Moderna de Carteras de Markowitz [1], cuyas limitaciones impulsaron la adopción de métodos más sofisticados. Actualmente, la inteligencia computacional es protagonista, con algoritmos genéticos (GA) para explorar espacios de soluciones complejas [2], [3] y modelos econométricos como GARCH, que se han consolidado para capturar la dinámica de clústeres de volatilidad que los modelos clásicos ignoran [4]. Investigaciones recientes se centran en hibridar estos enfoques, utilizando predicciones de modelos avanzados como insumos para optimizadores evolutivos, buscando superar a los métodos tradicionales [5].

A pesar del notable avance en el campo, un análisis de los enfoques existentes demuestra que muchas implementaciones utilizan métricas de volatilidad histórica que reaccionan con lentitud a los shocks del mercado o se centran en el Ratio de Sharpe [6], el cual penaliza por igual la volatilidad deseable (al alza) y la indeseable (a la baja) [7]. Esto define el problema central: la necesidad de un sistema integrado que utilice predicciones de volatilidad dinámicas y orientadas al futuro como insumo para un algoritmo de optimización diseñado específicamente para maximizar los rendimientos ajustados por el riesgo de caída, reflejando así de manera más fiel la aversión a las pérdidas del inversor.

Este trabajo propone un modelo híbrido GARCH-GA para la optimización dinámica de carteras. Se postula que esta estrategia, al optimizar el Ratio de Sortino, generará un rendimiento ajustado al riesgo superior al de benchmarks pasivos. El objetivo de la presente investigación es diseñar, implementar y validar un sistema de backtesting para dicha estrategia, empleando una ventana móvil para evitar sesgos. La principal aportación es la validación de una arquitectura cohesiva donde la predicción econométrica de la volatilidad y la optimización evolutiva funcionan de manera integrada, ofreciendo un enfoque robusto y práctico para la gestión de activos en mercados financieros modernos.

II. Metodología

La metodología propuesta consiste en un sistema híbrido que utiliza modelos GARCH para la predicción de la volatilidad y un algoritmo genético (GA) para la optimización de los pesos de la cartera. Este enfoque busca adaptarse dinámicamente a las condiciones cambiantes del mercado para maximizar los rendimientos ajustados al riesgo.