LABORATORIO NACIONAL DE INFORMÁTICA AVANZADA

TESIS DE MAESTRÍA

Hibridación de Algoritmos de Programación Matemática con Algoritmos Evolutivos en Problemas de Optimización de Diseño Mecatrónico

---

Autor:
Ing. Roides J. Cruz Lara

Supervisor:
Dr. Efrén Mezura Montes

Co-supervisor:
Dr. Edgar Alfredo Portilla Flores

Tesis presentada en cumplimiento de los requisitos para obtener el grado de Maestro en Computación Aplicada

Agosto de 2018

Resumen

Maestro en Computación Aplicada

Hibridación de Algoritmos de Programación Matemática con Algoritmos Evolutivos en Problemas de Optimización de Diseño Mecatrónico

por Ing. Roides J. Cruz Lara

La presente investigación realiza un estudio sobre la hibridación de Métodos de programación matemática con Algoritmos Evolutivos. Se resuelven seis problemas de optimización de Diseño Mecatrónico. Los primeros tres problemas son casos de estudio de la "Síntesis Óptima de un Mecanismo de Cuatro Barras", los cuales consisten en minimizar el error respecto a una trayectoria deseada. De forma similar, se procede para los dos casos de la "Síntesis Óptima de un Efector Final de Tres Dedos", donde se necesita maximizar la precisión de los dedos del efector mediante la minimización del error de la trayectoria del acoplador. El último problema resolver es la "Optimización de la Generación de Energía en una micro-red eléctrica aislada". Para este problema se considera cada hora del día como un problema de optimización donde las límites para las variables de diseño varían de acuerdo a lo acontecido en la hora anterior.

Se propone un nuevo enfoque de hibridación que plantea siete lineamientos de diseño. La distribución de diferentes instancias de un buscador local en puntos aleatorios del espacio de búsqueda, la comunicación y el balance entre los métodos de búsqueda global y local son las directrices más representativas. Siguiendo este enfoque se propusieron seis variantes del Algoritmo Híbrido basado en el Método Nelder Mead con Evolución Diferencial (HNMED). El algoritmo divide la población en NS de símpleces los cuales son operados por las instancias del Método Nelder Mead modificado. La Evolución Diferencial es aplicada a un subconjunto élite de la población en cada generación. El diseño experimental se dividió en Experimentos Preliminares y Finales. En los primeros se obtuvieron las primeras cinco variantes HNMED mediante la aplicación de pruebas de estadística descriptiva e inferencial. Teniendo en cuenta los resultados obtenidos se plantea HNMED-V6 la cual alcanza el mejor desempeño. Los experimentos finales comparan las dos ultimas variantes HNMED con la ED/rand/1/bin y C-LSHADE. Los resultados obtenidos muestran que las variantes propuestas son capaces de obtener resultados iguales o mejores a los ya reportados en la literatura utilizando un número significativamente menor de evaluaciones durante el proceso de minimización.

Agradecimientos

La realización del presente trabajo fue la cumbre formativa de dos años de aprendizaje para alcanzar el grado científico de Maestro en Computación Aplicada. Los resultados alcanzados durante todo este proceso no hubieran sido posibles sin el apoyo desinteresado y la guía de un número importante de personas

Primeramente quiero agradecer a mi padres Lourdes Lara y Roberto Cruz, por enseñarme desde pequeño el amor por la ciencia y el arte. Por inculcarme una concepción científica del mundo y la primicia de ser un individuo útil a la sociedad, comenzando por serle útil al humano más próximo. A mis hermanos de sangre Roberto y Roiber Cruz por ser constantemente mi competencia y mi ejemplo a seguir. A mis hermanos de la vida Roddy Fuentes, Joaquín Barbará, Javier Sotolongo y Andrés Gonzáles, por apoyarme incondicionalmente.

Especial agradecimiento a mis nuevos amigos de México: Fabiola Herrera y Humberto Sánchez, a mis compañeros de la MCA por todos los momentos en que me hicieron sentir como en casa.

Agradezco profundamente a mi asesor de tesis el Dr. Efrén Mezura por confiar en mi, por su guía, por los conocimientos aportados, y la rigurosidad en el trabajo que me permitieron crecer como profesional. Al Dr. Edgar Portilla por la excelente asesoría, por apoyarme durante la estancia y recibirme en su casa. A la Dra. Cora Excelente por brindarme su apoyo desde el primer día y estar siempre al pendiente de mi desempeño como estudiante, por sus consejos y su ayuda.

Agradezco al Consejo Nacional de Ciencia y Tecnología (CONACyT) por la beca que se me otorgó para realizar estudios de posgrado en México, de acuerdo a lo establecido en el Programa Nacional de Posgrados de Calidad con número de becario 604431.

Finalmente, quisiera agradecer a todo el personal docente y administrativo de LANIA, en especial al Lic. Marco Hernández Luna por todo el apoyo y la empatía brindada durante estos dos años.

Gracias a todos.

Índice general

Índice de figuras

Índice de tablas

A mis padres y mis hermanos

CAPÍTULO 1

Introducción

El desarrollo moderno industrial ha generado la necesidad de aumentar tanto la eficiencia, como la eficacia de los procesos de producción. En pos de lograr este objetivo las diferentes maquinarias y dispositivos industriales comenzaron a automatizarse para lograr sistemas de producción cada vez más ágiles, precisos y controlables. Al aumentar la complejidad de los procesos industriales se comenzaron a desarrollar además, maquinarias inteligentes, capaces de actuar efectivamente ante determinadas situaciones del proceso productivo, en lugar operar de forma programática.

La Mecatrónica surge entonces como una rama de la ingeniería capaz de lograr una integración sinérgica entre la ingeniería mecánica, la electrónica, el control automático y el cómputo inteligente [1]. Diferentes fuentes coinciden que el término Mecatrónica se utilizó por primera vez en Japón en la década de 1960 por la empresa Yasakawa, en documentos de aplicación de marca. Con el paso del tiempo, los avances en la Mecatrónica comenzaron a ser utilizados en las máquinas expendedoras, cámaras de enfoque automático y puertas automáticas. Con el advenimiento de las nuevas Tecnologías de la Información en la década de 1980, los microprocesadores se introdujeron en los sistemas mecatrónicos, mejorando el rendimiento significativamente. En la década de 1990, los avances en inteligencia computacional se aplicaron a la mecatrónica en formas que revolucionaron el campo [2]. Actualmente los sistemas mecatrónicos se utilizan en una amplia gama de la industria, donde se destacan: robots industriales, automóviles modernos, aeronaves y vehículos espaciales, sistemas de ventilación y electrodomésticos inteligentes.

Un dispositivo mecatrónico es un conjunto complejo de componentes que abarca tecnologías de diferentes áreas de la ingeniería. De forma general, se pueden identificar cuatro tipos de componentes o subsistemas principales que lo integran: Sistemas Eléctricos, Sistemas Mecánicos, Sistemas Computacionales y Sistemas de Información. Según Shetty y Kolk [3] el estudio de los sistemas mecatrónicos se puede dividir en las siguientes áreas del conocimiento:

1. **Modelado de Sistemas Físicos**: Incluye el diseño de óptimo de Sistemas Mecánicos y Eléctricos. El diseño de sistemas mecánicos se ocupa del comportamiento de la materia bajo la acción de fuerzas. Tales sistemas se categorizan como rígidos, deformables o fluidos. La cinemática y mecánica newtoniana proporcionan la base para la mayoría de los sistemas mecánicos y consta de tres conceptos independientes y absolutos: el espacio, el tiempo y la masa. Un cuarto concepto, la fuerza, también está presente, pero no es independiente de los otros tres. Los sistemas eléctricos en los sistemas mecatrónicos requieren una comprensión del análisis de circuitos de corriente continua (CC) y de corriente alterna (CA), incluyendo impedancia, potencia y dispositivos electromagnéticos así como semiconductores (tales como diodos y transistores). También se incluye el estudio de la energía en varias formas incluyendo potencial, cinética, eléctrica, calor, química, nuclear y radiante.

2. **Sensores y actuadores**: Abarca el estudio de sensores lineales y rotacionales, sensores de aceleración, fuerza, torque, presión, temperatura, flujo, sistemas de detección de luz, imágenes y visión. Dispositivos de fibra óptica, micro y nano-sensores. Incluye el estudio de actuadores electromecánicos, motores, actuadores hidráulicos y neumáticos, actuadores piezoeléctricos, micro y nano-actuadores.

3. **Señales y Sistemas**: Abarca el modelado mecatrónico y la respuesta de sistemas dinámicos. Estudia la estabilidad, capacidad de control y monitoreo de los sistemas, el diseño de control óptimo, el diseño de control adaptativo y no-linear, el control inteligente, las redes neuronales y sistemas de lógica difusa.

4. **Computadoras y Sistemas Lógicos**: Incluye el diseño de sistemas lógicos y sistemas de comunicación, la lógica secuencial sincrónica y asincrónica, la arquitectura de computadoras y microprocesadores, interfaces, controladores lógicos programables y computadoras de control integradas.

5. **Adquisición de datos y software**: Estudia los sistemas de adquisición de datos, transductores y sistemas de medición, sistemas amplificadores y acondicionadores de señales, sistemas de instrumentación basado en computadoras, ingeniería de software y grabación de datos.

Teniendo en cuenta estos elementos, Shetty y Kolk [3], definen el proceso de diseño de un sistema mecatrónico según se muestra en la Figura 1.1. Se puede observar que una tarea continua durante todo este proceso es la optimización del diseño, la cual resulta una condición necesaria para que el producto final se desempeñe eficientemente y así lograr la competitividad requerida por la industria. La optimización presenta mayor relevancia en el proceso de Modelado y Simulación. El modelado es el proceso donde se representa el comportamiento de un sistema real mediante un conjunto de ecuaciones matemáticas. El término sistema real es sinónimo de sistema físico, un sistema cuyo comportamiento se basa en la materia y la energía. Los modelos son estructuras de causa y efecto, aceptan información externa y la procesan con su lógica y ecuaciones para producir una o más salidas [4].

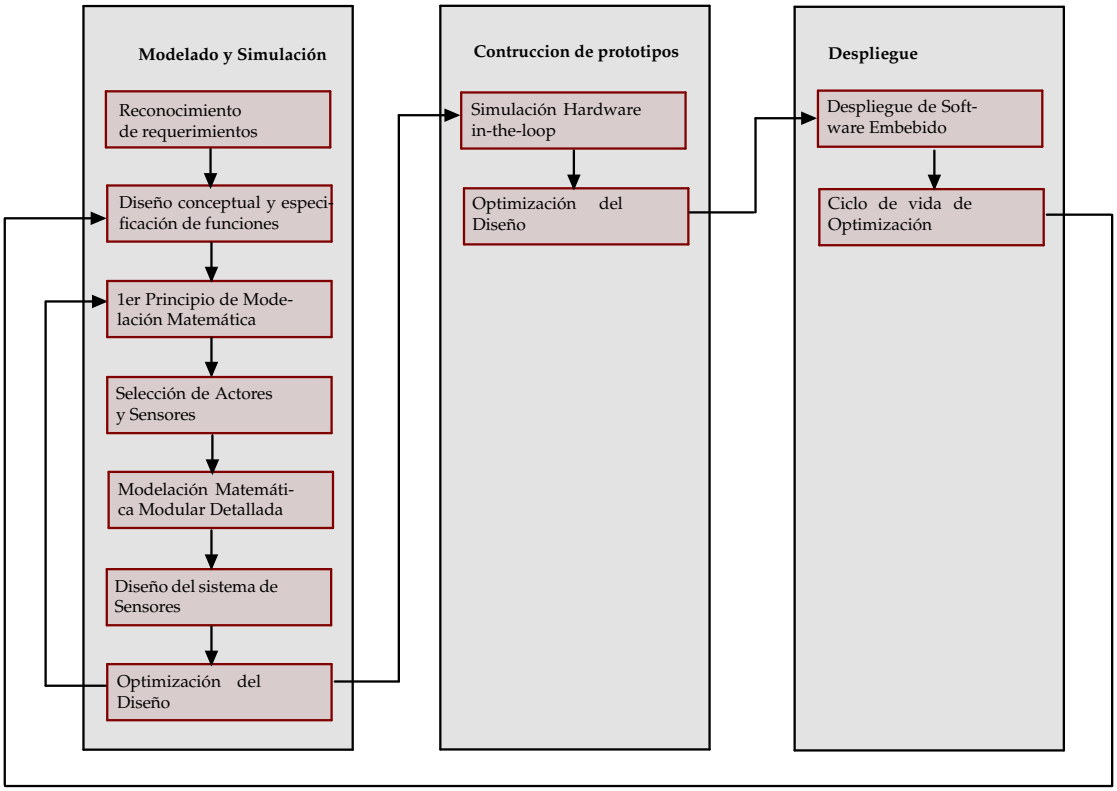


FIGURA 1.1: Proceso de Diseño Mecatrónico (Tomado de [3]).

El objetivo de la optimización es establecer una configuración óptima del sistema. Las propiedades de un sistema electromecánico pueden ser descritas matemáticamente mediante magnitudes físicas. Los valores de estas magnitudes, y por tanto, el estado del sistema pueden ser descritos mediante funciones matemáticas en correspondencia con el tipo de maquinaria y su descripción matemática. Estas funciones matemáticas constituyen las funciones objetivo de los problemas

TABLA 1.1: Resumen de características de los problemas de optimización de Diseño Mecatrónico.

Problema	Caso de estudio	Tipo de Función	n	LI	NI	LE	NE
Síntesis Óptima de un Mecanismo de Cuatro Barras	MCE1	No lineal	15	9	0	0	0
	MCE2	No lineal	6	4	0	0	0
	MCE3	No lineal	19	13	0	0	0
Síntesis Óptima de un Efector Final de Tres Dedos	GCE1	No lineal	14	8	0	0	0
	GCE2	No lineal	14	8	0	0	0
Optimización del costo de la generación de energía en una Microrred Eléctrica No Interconectable	SCE1	Cuadrática	4	3	0	1	0

de optimización en el diseño mecatrónico. Por la complejidad que presentan estos sistemas es común que se describan problemas de optimización no lineales con restricciones.

1.1. Descripción de la problemática

El presente trabajo investigativo se enfocará en la resolución de seis problemas de optimización. Primeramente se resolverán tres casos de estudio del problema de la "Síntesis Óptima de un Mecanismo de Cuatro Barras". Los problemas siguientes serán dos casos de estudio de la "Síntesis Óptima de un Efector Final de Tres Dedos". Por último, se abordará el problema de 'Optimización del costo de la generación de energía en una Microrred Eléctrica No Interconectable". Estos problemas son objeto de estudio en el Centro de Innovación y Desarrollo Tecnológico en Cómputo (CIDETEC) del Instituto Politécnico Nacional en el área de Mecatrónica. Los trabajos de investigación en los que han sido abordados servirán para validar la competitividad de la solución propuesta.

Las características de los problemas a resolver se encuentran resumidas en la Tabla 1.1, donde n, es el número de variables del problema, LI es el número de restricciones de desigualdad lineal, NI es el número de restricciones de desigualdad no lineales, LE es el número de restricciones de igualdad lineal. Además, se describe la nomenclatura a utilizar para los casos de estudio. Estos problemas de optimización han sido abordados en diferentes investigaciones, tanto por medio de algoritmos de programación matemática, como por algoritmos Bio-inspirados.

En el caso del diseño del Mecanismo de Cuatro Barras se destacan los trabajos realizados por Portilla y Mezura, quienes han aplicado métodos aproximados para la resolución del problema obteniendo resultados positivos.

En [5] se plantea el diseño del Mecanismo de Cuatro Barras como un problema de optimización dinámica mono-objetivo (PODMO) que considera los modelos

cinemático y dinámico de todo el sistema, así como un conjunto de restricciones incluyendo una restricción dinámica para lograr la síntesis de un mecanismo de cuatro barras que proporciona un movimiento simétrico en su enlace de balancín. En el artículo se describe la aplicación del Algoritmo de Colonia de Abejas Modificado donde el operador de variación utilizado por las abejas empleadas y observadoras para generar una nueva solución, incluye un mecanismo de recombinación. A partir del análisis de la simulación y los resultados obtenidos, se observó que las soluciones encontradas por el algoritmo propuesto conducen a un diseño más adecuado basado en el enfoque dinámico.

En el artículo "Síntesis Óptima de un Mecanismo de Cuatro Barras utilizando el algoritmo de Forrajeo de Bacterias Modificado", algoritmo de Forrajeo de Bacterias (BFOA por sus siglas en inglés) es adaptado para resolver el problema de optimización mediante la adición de una técnica de manipulación de restricciones capaz de incorporar un criterio de selección para los dos objetivos establecidos en el análisis cinemático del problema. Se diseñó un mecanismo de diversidad para favorecer la exploración del espacio de búsqueda y los resultados son comparados con los proporcionados por cuatro algoritmos encontrados en la literatura especializada utilizada para resolver problemas de diseño mecánico [6]. Del mismo modo, en [7] se aplica un algoritmo basado en el Forrajeo de Bacterias que utiliza dos operadores llamados Nados (TS-MBFOA por sus siglas en inglés). El algoritmo utiliza además, un mecanismo de sesgo para el conjunto inicial de bacterias, un operador de búsqueda local de segundo orden y un uso limitado de la etapa de reproducción.

La Evolución Diferencial (ED) ha presentado el mejor desempeño entre los Algoritmos Bio-inspirados que han sido aplicados a estos problemas. Lo que se evidencia en publicaciones como "Evolución Diferencial para el Ajuste Óptimo de la Ganancia de control de un Mecanismo de Cuatro Barras" donde se minimiza la variación del error de velocidad de un mecanismo de cuatro barras con fuerzas de resorte y de amortiguación utilizando la ED con un mecanismo de manejo de restricciones. Todas las ejecuciones del algoritmo convergieron al vector de la variables de diseño óptimo y la solución encontrada puede ser considerada como la global. Por otro lado, la media reportada de los tiempos de ejecución del algoritmo fue de diez minutos. Igualmente, la ED ha sido aplicada con éxito en problemas de optimización mono y multi-objetivo con restricciones de diseños mecatrónicos [8] [9].

De forma similar la ED ha obtenido resultados importantes en la resolución del problema de la "Síntesis Óptima de un Efector Final de Tres Dedos". En [10] se plantea la configuración de parámetros para un algoritmo de Evolución Diferencial con Variantes Combinadas (DECV por sus siglas en inglés) para de mejorar su desempeño. Se resuelven los dos casos de estudio del Efector Final. Se realiza una calibración de los parámetros principales del algoritmo, para posteriormente agregar un mecanismo de control para el parámetros asociado a la mutación.

Los resultados sugieren, que la aplicación del mecanismo de control de parámetros permite al algoritmo base DECV alcanzar resultados altamente competitivos con un costo computacional menor, medido por el número de evaluaciones de soluciones potenciales.

La Gestión Óptima de Generación de Energía en una MR-RNI es abordada en [11] mediante el Método del Gradiente Proyectado de Rosen, un método de programación matemática que proyecta el vector gradiente negativo en una dirección que mejora la minimización de la función objetivo, al tiempo que mantiene la factibilidad. El algoritmo implementado permitió realizar estudios de despacho económico teniendo en cuenta las restricciones del estado de carga de la batería, ante simulaciones de carga, generación eólica y fotovoltáica.

En [12] se analiza el Flujo Óptimo de Potencia (FOP) de una MR, el cual consiste en resolver ecuaciones que caracterizan un sistema eléctrico (potencia activa y reactiva de cada nodo) ajustando los valores de las variables de control (voltajes o potencias) para optimizar un parámetro específico del sistema, representado por una función objetivo. Para resolver el problema se utiliza un algoritmo matemático basado en el Método de Gradiente. El gradiente permite medir la sensibilidad de la función objetivo con respecto a los cambios en las variables de control. Originándose en una dirección opuesta al gradiente, se alcanza un punto mínimo factible con un valor de función inferior. La repetición de este proceso conduce a la solución óptima del sistema.

Finalmente en [13] se proponen dos variantes de la Evolución Diferencial: C-LSHADE y LSHADE-CV. Estas variantes están basadas en una variante competitiva de la Evolución llamada LSHADE, la cual incorpora un mecanismo de control que emplea una memoria histórica de parámetros exitosos para la configuración del factor de escala (F), la probabilidad de cruzamiento (CR) y una reducción lineal de la población. C-LSHADE agrega las adaptaciones necesarias para solucionar problemas de optimización con restricciones. Por otra parte, LSHADE-CV aplica la estrategia de LSHADE al algoritmo DECV. En este trabajo se resuelven de manera satisfactoria los problemas de diseño cinemático que serán objeto de estudio en el presente trabajo. Además, se resuelve el problema del Smart Grid donde se minimizó el costo del suministro de energía en cada hora del día. La solución encontrada aumentó la generación de energía eléctrica mediante las utilización de las Fuentes de Energía Renovable maximizando el ahorro. Todos los problemas fueron resueltos utilizando un número inferior de evaluaciones de la función objetivo respecto a la ED/rand/1/bin.

Como se puede observar, los problemas de diseño a resolver han sido atacados por los dos grupos más utilizados en problemas de optimización de diseño mecatrónico: los métodos de programación matemática y los algoritmos aproximados, entre los que se encuentran los algoritmos bio-inspirados. Los métodos de programación matemática son utilizados para encontrar mínimos locales en funciones

objetivo de una o varias variables y pueden ser clasificados de acuerdo al orden de la derivada utilizada en la aplicación del método [2]:

1. Métodos de Orden Cero (Comparativos)
    a) Métodos Comparación coordinada
    b) Métodos basados en simplex
    c) Métodos estocásticos
2. Métodos de Primer Orden (Gradiente y Cuasi-gradiente)
    a) Métodos de direcciones asociadas
    b) Métodos variable-métrica
3. Métodos de Segundo Orden (Método de Newton).

Una deficiencia sustancial de los métodos clásicos radica en que al optimizar funciones con gran cantidad de mínimos locales generalmente convergen en un mínimo cercano al punto de inicio perdiendo el mínimo global. También una característica de estos métodos, es su sensibilidad al punto de origen para encontrar la solución. Por otra parte, los algoritmos que utilizan la derivada de la función para obtener los puntos críticos, asumen como es lógico que que la función sea derivable en el espacio de búsqueda del problema. Estas características pueden restringir su aplicación, debido a que en problemas reales es común encontrar funciones que no son continuas y ni diferenciables en todo el espacio de búsqueda. A pesar de estas restricciones, los métodos de programación matemática son considerablemente menos costosos computacionalmente, presentando mayor rapidez de convergencia hacia los óptimos de la función.

Debido a que los métodos de programación matemática presentan dificultades para resolver problemas de optimización global se han aplicado, como se ha descrito anteriormente, los algoritmos bio-inspirados siendo los de tipo evolutivo los que mejores resultados presentan. Los algoritmos bio-inspirados emulan un fenómeno existente en la naturaleza mediante el uso de una metáfora biológica del comportamiento de cierto proceso natural o agente biológico. Estos algoritmos son métodos de resolución de problemas que tienen diversas aplicaciones, particularmente en la resolución de problemas de optimización global como es el caso del Diseño Mecatrónico [14]. En general los algoritmos bio-inspirados generan soluciones iniciales aleatoriamente, lo que les permite buscar de forma más eficiente en el espacio de todas las posibles soluciones. Entre los algoritmos más utilizados se pueden encontrar los siguientes:

1. Algoritmos estocásticos
2. Algoritmos de Recocido Simulado
3. Algoritmos Evolutivos

4. Algoritmos de Inteligencia de Colectiva

Una estrategia efectiva es combinar dos algoritmos para obtener un nuevo procedimiento que incorpore las mejores características de cada enfoque. A este enfoque se conoce como hibridación y será abordado con mayor profundidad en el Capítulo 6. Dentro de la clasificación de algoritmos híbridos se encuentran los Algoritmos Meméticos, este enfoque se basa en la interacción de los procesos de búsqueda global y local. Estos algoritmos son metaheurísticas basadas en población. Básicamente están constituidos por un algoritmo bio-inspirado para la búsqueda global y un conjunto de algoritmos de búsqueda local que se activan dentro del ciclo de generación del buscador global [15]. Actualmente, la aplicación de estos algoritmos se ha incrementado debido a que muestran un mejor rendimiento en espacios de búsqueda complejos.

En este ámbito es necesario destacar el trabajo realizado en [16] donde se propone un algoritmo memético que utiliza como buscador global el algoritmo Modificado de Colmena de Abejas, con la adición del método de Caminata Aleatoria. Los resultados de la simulación muestran un control de alta precisión de la trayectoria propuesta para el mecanismo de Cuatro Barras diseñado.

La principal deficiencia de los algoritmos evolutivos y sus híbridos se encuentra en que, debido a la utilización de una población y su naturaleza estocástica necesitan un número elevado de evaluaciones de la función objetivo durante la optimización. Por tanto, son considerablemente más lentos que sus contrapartes de Programación Matemática cuando se aplican a espacios de búsquedas complejos.

1.2. Planteamiento del Problema de investigación

Con base en los argumentos presentados en la sección anterior, se puede observar que la problemática existente entorno el objeto de estudio (problemas a resolver), es que, debido a la complejidad que presentan estos problemas, los algoritmos empleados hasta el momento necesitan un número elevado de evaluaciones para obtener soluciones de calidad. Teniendo en cuenta lo antes planteado se formula el siguiente problema de investigación:

¿Cómo resolver problemas de optimización no lineales de Diseño Mecatrónico disminuyendo el costo computacional, medido por el número de evaluaciones de soluciones potenciales, para obtener una solución competitiva?

1.3. Objetivo general

Para resolver el problema planteado se define como objetivo general:

Diseñar un Algoritmo Híbrido de Programación Matemática con elementos de Algoritmos Evolutivos para resolver problemas no lineales de optimización de Diseño Mecatrónico.

1.3.1. Objetivos específicos

El objetivo general se desglosa en los siguientes objetivos específicos:

1. Estudiar el comportamiento un algoritmo de programación matemática al resolver problemas de optimización de diseño mecatrónico.
2. Determinar bondades y deficiencias del método de Programación Matemática en la resolución de problemas de optimización diseño mecatrónico.
3. Diseñar un algoritmo híbrido basado en un método de programación matemática combinado con un algoritmo evolutivo para mejorar la búsqueda.
4. Aplicar el algoritmo híbrido a problemas de optimización de diseño mecatrónico.
5. Realizar pruebas de medición de desempeño mediante el análisis estadístico y medición del número de evaluaciones del algoritmo propuesto.
6. Comparar los resultados obtenidos con el desempeño de los algoritmos identificados en la literatura especializada.

1.4. Hipótesis de investigación

La propuesta de solución se basa en la siguiente hipótesis de la investigación que plantea:
Un Algoritmo Híbrido basado en un método de programación matemática y enriquecido con operadores de algoritmos evolutivos realizará menos evaluaciones, lo que reduciría el tiempo necesario para obtener soluciones iguales o mejores a las reportadas en la literatura especializada de problemas de optimización de Diseño Mecatrónico.

1.5. Justificación

La Mecatrónica es un área en constante desarrollo, cada año se generan nuevas necesidades en la industria que contribuyen al constante cambio y evolución de las tecnologías que son aplicadas para la creación de los diferentes productos [17] [18] [19] [20]. Debido a que los pasos en el proceso de diseño mecatrónico ocurren secuencialmente, el enfoque tradicional es un enfoque de ingeniería secuencial. Según [3], una encuesta de Standish Group sobre proyectos de mectrónica dependientes de softwares, existe un 222 % de rebasamiento del tiempo para proyectos terminados y solo el 16.2 % de todos los proyectos se completaron a tiempo y dentro del presupuesto. Por las deficiencias identificadas en el objeto de estudio de la presente investigación se evidencia la necesidad de definir nuevos enfoques que proporcionen eficiencia al proceso optimización de diseño mecatrónico.

En los casos de estudio a resolver, se han aplicado tres enfoques: los algoritmos bio-inspirados,los métodos de programación matemática y los algoritmos meméticos; lo que deja espacio a la investigación y análisis para nuevos enfoques de hibridación de algoritmos aplicados al objeto de estudio del presente trabajo. Además, se debe tener en cuenta que estos son problemas de optimización global y los mínimos encontrados en algunos problemas pueden no ser los óptimos globales. Teniendo en cuenta, la diversidad de algoritmos efectivos que se puede encontrar en la literatura especializada, la hibridación supone una vía factible para reducir los tiempos de ejecución al tiempo que se garantiza la convergencia a soluciones competitivas en problemas optimización de diseño ingenieril [21] [22] [23].

Por otra parte, existe un número creciente de sistemas mecatrónicos que son controlados por computadoras en tiempo, cuyo desempeño puede verse mejorado. Se destacan en este tipo de sistemas, tareas como la planeación de movimiento en tiempo real de vehículos autónomos [24], la optimización de la fuerza de agarre en tiempo real de mecanismos manipuladores de múltiples dedos [25], y la optimización de recursos en Smarts Grids [26] [27] [28]. Para estos sistemas la rapidez del algoritmo en encontrar soluciones de calidad es de suma importancia

1.6. Alcances

- El algoritmo propuesto estará destinado a resolver únicamente casos de estudios de los problemas generales planteados en el Capítulo 5.
- Los resultados alcanzados por el algoritmo propuesto deberán ser competitivos en el contexto de los problemas a resolver.

- Los resultados alcanzados por la presente investigación serán sometidos a pruebas de desempeño utilizando métricas encontradas en la literatura especializada.

1.7. Limitaciones

- La investigación se limita a la optimización del diseño y la simulación de la simulación de las soluciones obtenidas. No se implementarán físicamente los prototipos.
- Se asumirá pérdida de propiedades de los métodos clásicos al incorporar mecanismos heurísticos en ellos.

1.8. Organización del documento

A continuación se describe la estructura y contenido de las secciones que conforman este documento:

1. **Capítulo 2. Optimización**. En este capítulo se introducen los conceptos de optimización y convexidad de una función, se describen los componentes principales de un problema de optimización, su clasificación y tipos de técnicas utilizadas para resolverlos.
2. **Capítulo 3. Programación Matemática**. Se presentan los principales métodos de Programación Matemática. Se describen los métodos de acotamiento, métodos directos, así como los métodos basados en gradiente. Se seleccionan dos candidatos para la realizar la hibridación con los algoritmos evolutivos.
3. **Capítulo 4. Computación Evolutiva**. El capítulo consiste en una introducción a los conceptos básicos del Cómputo Evolutivo. Se describen los principales algoritmos en está área de la Inteligencia Artificial. Se presenta con mayor detalle el algoritmo de Evolución Diferencial.
4. **Capítulo 5. Problemas de Optimización de Diseño Mecatrónicos**. Se describen los problemas de diseño mecatrónico como problemas de optimización mono-objetivos con restricciones. Se describen tres casos de la síntesis óptima del mecanismo de cuatro barras y el despacho económico de energía en una microrred aislada.
5. **Capítulo 6. Hibridación de Métodos de Programación Matemática con Algoritmos Evolutivos**. En este capítulo se presenta la solución propuesta de

la presente investigación. Se plantea un enfoque de hibridación que contiene diferentes lineamientos de diseño. Además se describen seis variantes de un algoritmo híbrido basado en el Método Nelder Mead, y que utiliza la Evolución Diferencial como buscador global.

6. **Capítulo 7. Experimentos y Resultados**. Este capítulo describe el diseño experimental mediante el cual se diseña la propuesta de solución. Se aplican técnicas de estadística descriptiva e inferencial para el análisis del comportamiento de los algoritmos y la comparación con los métodos más competitivos encontrados en la literatura.

7. **Conclusiones y Trabajo futuro**. Se realizan observaciones finales sobre los resultados obtenidos en los problemas de optimización y consideraciones sobre el comportamiento de los algoritmos propuestos. Se describen las aportaciones más importantes de la investigación así como los trabajos futuros a realizar teniendo en cuenta los resultados alcanzados.

1.9. Conclusiones del capítulo

En el capítulo se fundamenta la presente investigación cuyo objeto de estudio es un conjunto de problemas no lineales de Optimización de Diseño Mecatrónico. Hasta el momento los enfoques más exitoso han sido los basados en metaheurísticas y sus híbridos. Sin embargo, se evidencia la necesidad de continuar investigando los problemas de optimización descritos así como diseñar y aplicar nuevos enfoques que reduzcan el costo computacional, específicamente la complejidad temporal medida en el número de evaluaciones de la función objetivo. Para lograr esto se plantea como objetivo diseñar un algoritmo híbrido que utilice un método de Programación Matemática como base, formulando la hipótesis de que este esquema reduciría el número de evaluaciones necesarias para encontrar resultados competitivos.

CAPÍTULO 2

Optimización

Como concepto general, la optimización es el proceso de obtener el mejor resultado dadas ciertas circunstancias [29]. El patrón de encontrar condiciones o estados óptimos de funcionamiento, se observa en la naturaleza tanto sistemas físicos que tienden a converger a estados mínimos de energía, como en organismos unicelulares o seres complejos como el ser humano cuando evalúan las mejores opciones como base para tomar decisiones beneficiosas [30]. Estas decisiones tienen la finalidad de minimizar el esfuerzo requerido o maximizar el beneficio deseado, lo que implica la selección de valores para una serie de variables interrelacionadas, pero centrando la atención en un único objetivo diseñado para cuantificar el desempeño y medir la calidad de la decisión. Debido a que el esfuerzo requerido o el beneficio deseado de una situación práctica pueden expresarse como una función de ciertas variables de decisión las cuales describen las características del problema, la optimización consiste en encontrar las condiciones que dan el valor máximo o mínimo de una función [29]. Antoniou y Lu [31] proporcionan un concepto que desde el punto de vista metodológico divide la optimización en teoría y práctica:

> *"La teoría de la optimización es la rama de las matemáticas que abarca el estudio cuantitativo de los óptimos y los métodos para encontrarlos. La práctica de la optimización, por otro lado, es la colección de técnicas, métodos, procedimientos y algoritmos que se pueden utilizar para encontrar el óptimo".*

El presente trabajo se enfoca en la práctica de la optimización debido a que se propondrá un algoritmo que sea capaz de resolver de forma eficiente un conjunto de problemas de optimización. Los problemas en ingeniería son generalmente problemas de programación no lineales (NLP por sus siglas en inglés). La palabra programación significa planificación en este contexto [32]. La rama de la optimización llamada Programación Matemática será objeto de análisis en el siguiente capítulo donde se abordarán sus conceptos y métodos principales que además servirán como base para la solución propuesta.

Diferentes autores coinciden en que una fase importante de la optimización, visto como proceso científico, es la modelación del problema una vez que ha sido identificado en la realidad [29] [30] [33]. El modelado es el proceso de expresar observaciones sobre un problema práctico en forma matemática, mediante funciones que realizan operaciones matemáticas básicas como suma, resta, multiplicación y división sobre las variables de decisión. Las observaciones se refieren a los datos obtenidos para el problema en cuestión, variando ciertos parámetros del problema a través de experimentos. El objetivo de los modelos matemáticos es proporcionar una predicción del comportamiento del problema para diferentes entradas. Si el modelo no arroja los resultados esperados, este puede refinarse realizando más experimentos como es el caso de los modelos de optimización en el proceso de diseño mecatrónico [34].

En el presente trabajo los modelos serán planteados como problemas de minimización. Esta norma es adoptada sobre todo en problemas de ingeniería y responde al hecho de que si un punto x^* corresponde al valor mínimo de la función $f(x^*)$, el mismo punto también corresponde al valor máximo del negativo de la función, $-f(x^*)$. Por tanto, sin pérdida de generalidad, la optimización de una función puede considerarse entonces como la minimización ya que el máximo de una función se puede encontrar buscando el mínimo del negativo de la misma función. Además, se debe destacar que las operaciones de multiplicación, división, adición o sustracción de $f(x)$ por una constante positiva c, no cambiarán la solución óptima x^* [29].

El modelo general de minimización se plantea en [33] de la siguiente forma:

$$Min \quad f(\vec{x}) \tag{2.1}$$

sujeto a:

$$h_i(\vec{x}) = 0 \quad i = 1, 2, ..., m \tag{2.2}$$

$$g_j(\vec{x}) \geq 0 \quad j = 1, 2, ..., p \tag{2.3}$$

$$\vec{x} \in S \tag{2.4}$$

Donde $\vec{x}$ es el vector de variables de decisión que pertenece a S un subespacio de $\mathbb{R}^n$. $f(\vec{x})$ es la función objetivo a optimizar, siendo $h_i(\vec{x})$ y $g_j(\vec{x})$ las restricciones

de igualdad y desigualdad respectivamente. Teniendo en cuenta que tanto $f(\vec{x})$, $h_i(\vec{x})$ como $g_j(\vec{x})$ son funciones escalares de $\vec{x}$.

2.1. Variables de diseño

Las variables en la función objetivo se denotan como las *variables de diseño* o *variables de decisión*. Si bien ambos términos son válidos, en lo adelante se denominarán variables de diseño, debido a que el campo de acción del presente trabajo radica en el área de la mecatrónica.

En problemas de ingeniería las variables de diseño pueden ser capacidades de producción o demanda en un problema de transporte, las dimensiones de una estructura o sus atributos de material, para un problema de optimización de estructura por ejemplo. A partir de consideraciones prácticas, las variables de diseño pueden tomar valores dentro de un límite inferior y un límite superior solamente. Estas pueden ser números reales o tomar valores dentro de un conjunto discreto de números, así como pueden estar definidas para valores binarios o enteros [34].

El conjunto de variables de diseño se representa mediante el vector de diseño $\vec{x} = [x_1, x_2, ..., x_n]^T$. Para la definición de $\vec{x}$ se plantea un espacio cartesiano n-dimensional llamado *espacio de diseño*. Donde el i-ésimo eje de coordenadas cartesianas constituye una variable de diseño x_i para $i = \{1, 2, ..., n\}$. Luego cada vector de diseño describe un punto en el espacio de diseño n-dimensional llamado punto de diseño el cual representa una solución factible o no al problema de optimización [29].

Desde el punto de vista de la complejidad computacional se debe tener en cuenta el orden $\vec{x}$, ya que la dimensionalidad impacta directamente la complejidad del problema, y por tanto en la eficacia de los métodos y su rapidez de convergencia.

2.2. Función Objetivo

La *función objetivo* es el criterio con respecto al cual se optimiza el diseño, expresado en función de un conjunto de variables. La elección de la función objetivo se rige por la naturaleza del problema. Los procedimientos de diseño convencionales apuntan a encontrar un diseño aceptable o adecuado que sea capaz de satisfacer los requisitos funcionales o de otro tipo del problema bajo estudio. En general, existe más de un diseño aceptable para el mismo problema, y el objetivo de la optimización es elegir el mejor de los muchos diseños aceptables y disponibles [34].

La elección de la función objetivo parece ser directa en la mayoría de los problemas de diseño. Sin embargo, existen casos donde la optimización con respecto a un criterio particular conduce a resultados que pueden no ser satisfactorios con respecto a otro criterio. Por lo tanto, la selección de la función objetivo puede ser una de las decisiones más importantes en todo el proceso de diseño óptimo.

2.3. Restricciones de Diseño

Los problemas de optimización con restricciones surgen de modelos que incluyen requisitos explícitos sobre las posibles soluciones. En estos problemas, las posibles soluciones se restringen lo cual puede ser desafiante para los algoritmos de optimización, sobre todo si se trata de un problema multivariable con varias restricciones no lineales, alcanzar un solución factible en el espacio de diseño puede resultar una tarea difícil. De forma general, se pueden encontrar restricciones de igualdad y desigualdad. Estas restricciones pueden ser límites simples a las variables dentro de ciertos intervalos, restricciones lineales generales, o desigualdades no lineales que representan relaciones complejas entre las variables.

En un problema de optimización con sólo restricciones de desigualdad $g_j(\vec{x}) \leq 0$, el conjunto de puntos X que satisfacen la ecuación $g_j(X) = 0$, forma una hipersuperficie en el espacio de diseño que se denomina *superficie de restricción*. Dicha superficie divide el espacio de diseño en dos regiones: la región cuyos puntos cumplen la condición de que $g_j(X) < 0$ y la otra en la que $g_j(X) > 0$. Tanto los puntos que se encuentran en la hipersuperficie como los puntos que se encuentran dentro de la región donde $g_j(X) < 0$ pertenecen al *espacio factible*, y son a su vez *puntos factibles*. Por el contrario, los puntos que se encuentran en la región donde $g_j(X) > 0$ son *puntos no factibles*. El conjunto de las superficies de restricción $g_j(X) = 0$, para $j = \{1, 2, ..., m\}$, que separa la región factible se denomina *superficie de restricción compuesta* [29].

Si una restricción de desigualdad $g_i(x) \leq 0$ evaluada en un punto factible x_k cumple que $g_i(x_k) = 0$, se dice que esta restricción está *activa* e *inactiva* en caso de que $g_i(x_k) < 0$. Lo planteado anteriormente puede generalizarse para las restricciones de igualdad $h_i(\vec{x}) = 0$, si se aplica la convención de que cualquier restricción de igualdad $h_i(\vec{x}) = 0$ se encuentra activa en cualquier punto factible. Las restricciones activas en un punto factible x_k restringen el espacio factible en la vecindad de x_k, mientras que las otras restricciones inactivas no tienen influencia en vecindades de x_k. Si las restricciones del problema excluyen al mínimo de la función es común que el mínimo factible se encuentre en la hipersuperficie. Por lo tanto, al estudiar las propiedades de un punto mínimo local, se debe centrar la atención en las restricciones activas. [33]. Los puntos de diseño pueden ser clasificados de acuerdo a la región a que pertenecen de la siguiente forma según [29]:

1. **Punto libre factible**: no se encuentran en ninguna superficie y es factible.
2. **Punto libre no factible**: no se encuentran en ninguna superficie y está fuera del espacio factible.
3. **Punto límite factible**: Se encuentra en una o más de una superficie de restricción y cumple con todas las restricciones.
4. **Punto límite no factible**: Se encuentra en una o más de una superficie de restricción y no cumple con todas las restricciones.

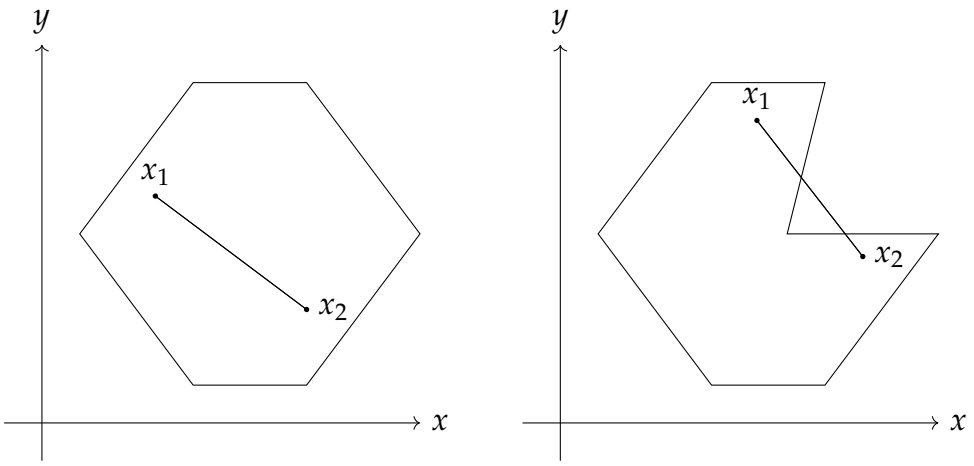
En ingeniería las restricciones pueden ser clasificadas de acuerdo a los requerimientos planteados para el sistema que se estudia. Las restricciones que se aplican colectivamente para obtener un diseño aceptable se denominan restricciones de diseño. Las restricciones que representan limitaciones en el comportamiento o el rendimiento del sistema se denominan restricciones funcionales o de comportamiento. Las restricciones que representan limitaciones físicas en las variables de diseño, como la disponibilidad, fabricabilidad y transportabilidad, se conocen como restricciones geométricas o laterales [29].

Cuando tanto la función objetivo como todas las restricciones son funciones lineales de x, se considera un problema de programación lineal. Las ciencias de la gestión y la investigación de operaciones hacen un amplio uso de los modelos lineales. Por otra parte, los problemas de programación no lineal son aquellos en los que al menos algunas de las restricciones o la función objetivo son funciones no lineales. Este tipo de problemas son más comunes en las ciencias físicas y en la ingeniería como es el caso de la Mecatrónica. [30]. En la sección 2.6 se abordan una serie de clasificaciones entre las que se encuentran este tipo de problemas.

2.4. Óptimos locales y globales

Para presentar las características de los óptimos locales y globales en una función objetivo se debe introducir primeramente el concepto de *convexidad*. Si se tienen un conjunto S de elementos y se cumple que: para todo par de elementos cualesquiera x_1 y x_2 de S, el segmento rectilíneo que une ambos se encuentra dentro de S, entonces el conjunto S es un *conjunto convexo*. De lo contrario se dice que el conjunto S, es *no convexo*, véase Figura 2.2.

El concepto de convexidad puede ser aplicado tanto a conjuntos como a funciones permitiendo analizar la función objetivo para identificar si tiene un solo mínimo. Un ejemplo ilustrativo consiste en plantear una función uni-variable $f(x)$ y dos puntos x_1 y x_2 en los cuales el valor de la función es $f(x_1)$ y $f(x_2)$ respectivamente. Si para cualquier punto x que se seleccione entre x_1 y x_2, el valor de $f(x)$ es menor que el valor de la función lineal $l(x)$ que intersecta a $f(x_1)$ y $f(x_2)$, entonces $f(x)$ es una *función convexa* y $-f(x)$ es una *función cóncava*, véase Figura 2.3.



(A) Conjunto convexo (B) Conjunto no convexo

FIGURA 2.1: Convexidad de conjuntos

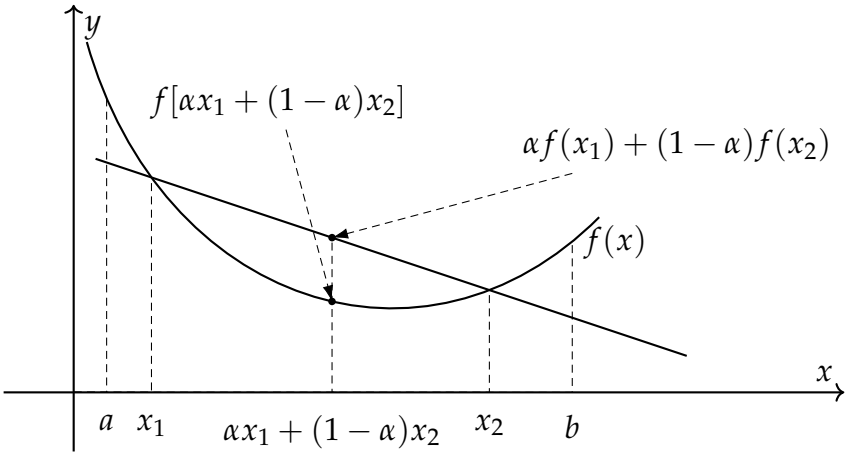


FIGURA 2.2: Función convexa.

Formalmente se plantea que una función $f(x)$ definida sobre un conjunto convexo $\mathbb{R}^n$, es convexa si para todo par de puntos x_1, x_2 y cada número real α en el rango $0 < \alpha < 1$, se cumple la desigualdad [31]:

$$f[\alpha x_1 + (1-\alpha)x_2] \leq \alpha f(x_1) + (1-\alpha)f(x_2). \tag{2.5}$$

Luego si $x_1 \neq x_2$:

$$f[\alpha x_1 + (1-\alpha)x_2] < \alpha f(x_1) + (1-\alpha)f(x_2). \tag{2.6}$$

Cuando la función objetivo presenta varios cambios de monotonía en todo su dominio, se le denomina como *función multimodal*. La condición de multimodalidad no depende ni de la continuidad ni de la convexidad de la función. En estos casos

se debe diferenciar entre un punto que representa una solución óptima global y los que representan óptimos locales. Si conoce que $S \in \mathbb{R}^n$ como el conjunto de todos puntos factibles del problema, el punto $x^* \in S$ que satisface:

$$f(x^*) \leq f(x) \quad \forall x \in S \tag{2.7}$$

se denomina *óptimo global* o *solución óptima global*. Por otra parte si un punto x' tal que existe una vecindad $Q \in S$ de x' que satisface:

$$f(x') \leq f(x) \quad \forall x \in Q \tag{2.8}$$

se denomina *óptimo local* o *solución óptima local*. Cuando el problema es convexo, o sea, cuando todas las funciones $f(x)$, $h_i(x)$ $g_i(x)$ son convexas, cualquier mínimo local es global. Se dice que un problema es multiextremal cuando tiene varios mínimos locales con diferentes valores de función objetivo. Por su propia naturaleza, la optimización global requiere métodos significativamente diferentes de los utilizados en la optimización no lineal convencional, que hasta ahora se ha centrado principalmente en la búsqueda de óptimos locales.

La esencia de un método iterativo de optimización es que dada una solución de inicial x_0 para el problema se busca una mejor. Si el método utilizado es algún procedimiento clásico de programación no lineal no existe garantía de que x_0 sea una solución global. La facilidad para quedar atascados en puntos estacionarios es la principal deficiencia de los métodos clásicos de programación no lineal y motiva la necesidad de desarrollar procedimientos de búsqueda globales. Por lo tanto, cualquier procedimiento de búsqueda global debe ser capaz de abordar la cuestión fundamental de cómo trascender una solución factible dada, que puede ser un punto estacionario y converger al mínimo global. [35].

2.5. Clasificación de Problemas de Optimización

En la literatura especializada se puede encontrar diversas clasificaciones para los problemas de optimización que abarcan desde la existencia de las restricciones y su naturaleza, la naturaleza de la función objetivo y las variables de diseño hasta las clasificaciones basadas en la complejidad del problema medido por el número de variables (dimensionalidad) y el número de restricciones [33]. La tarea de clasificar el problema de optimización puede resultar beneficiosa para la elección del algoritmo de optimización y el análisis de complejidad del problema. En las siguientes secciones se describen una serie clasificaciones descritas por Rao [29], las cuales abarcan un amplio espectro de problemas de optimización.

2.5.1. Clasificación basada en la existencia de restricciones

Los problemas de optimización se pueden clasificar teniendo en cuenta la existencia de restricciones:

1. *Problemas de optimización con restricciones*: Problemas en los que, las soluciones posibles necesitan cumplir con un grupo de requerimientos representados como funciones escalares de las variables de diseño.
2. *Problemas de optimización sin restricciones*: Surgen de aplicaciones prácticas en las cuales, aunque existen restricciones naturales sobre las variables, es posible ignorarlas debido a que no tienen ningún efecto sobre la solución óptima. Los problemas no restringidos surgen también como reformulaciones de problemas de optimización restringida, en los cuales las restricciones son reemplazadas por términos de penalización en la función objetivo que tienen el efecto de desalentar las violaciones de restricciones.

2.5.2. Clasificación basada en la naturaleza de las variables de diseño

En función de la naturaleza de las variables de diseño, los problemas de optimización se pueden clasificar en dos categorías:

1. *Problemas de optimización estática* o *problemas de optimización de parámetros*: son los problemas que consisten en encontrar valores para un conjunto de parámetros de diseño que minimizan a ciertas funciones prescritas de estos parámetros al tiempo que cumplen con las restricciones a las que están sujetas.
2. Problema de *trayectoria* o *optimización dinámica*: en este caso el objetivo es encontrar un conjunto de parámetros de diseño, que son todos funciones continuas de algún otro parámetro, y que a la vez minimiza una función objetivo sujeta a un conjunto de restricciones. En estos problemas, se deben realizar varias optimizaciones en secuencia y se puede requerir una estrategia general para lograr una solución óptima global.

2.5.3. Clasificación basada en la estructura física del problema

En esta categoría se encuentran los *problemas control óptimo* (OC por sus siglas en inglés). Esta clasificación se refiere a los problemas de programación matemática que involucran una cantidad de etapas, donde cada etapa evoluciona desde la etapa anterior de una manera prescrita. Por lo general, se describen mediante dos

tipos de variables: variables de control (diseño) y las variables de estado. Las variables de control definen el sistema y rigen la evolución del sistema de una etapa a la siguiente, y las variables de estado describen el comportamiento o el estado del sistema en cualquier etapa. El problema es encontrar un conjunto de variables de control tales que la función objetivo total (también conocida como índice de rendimiento) en todas las etapas se minimice; cumpliendo con un conjunto de restricciones sobre las variables de control y de estado.

2.5.4. Clasificación basada en la naturaleza de las ecuaciones involucradas

Esta clasificación se basa en la naturaleza de las expresiones para la función objetivo y las restricciones. Incluye los siguientes problemas:

1. *Problema de programación no lineal*: abarca a los problemas en los que al menos una de las funciones involucradas es no lineal. Este es el problema general de programación matemática, el resto pueden considerarse como casos especiales de dicho problema. Los problemas de Diseño Cinemático se incluyen en esta categoría.

2. *Problema de programación geométrica*: es un problema de optimización en el que la función objetivo y las restricciones se expresan como posinomios en $\vec{x}$. Una función $f(\vec{x})$ se denomina posinomial si se puede expresar como la suma de los términos de potencia de cada variable $f(\vec{x}) = U_1 + U_2 + ... + U_n$ para $U_i = \{c_i {x_1}^{a_{1i}}, {x_2}^{a_{2i}}, \cdots, {x_n}^{a_{ni}}\}$; donde c_i y a_{ij} son constantes con $c_i > 0$ y $x_j > 0$.

3. *Problema de programación cuadrático*: Un problema de programación cuadrática es un problema de programación no lineal con una función objetivo cuadrática y restricciones lineales. Existen métodos de Programación Matemática destinados a resolver este tipo específico de problemas. El caso de la Micro Red Aislada se encuentra en esta categoría.

4. *Problema de programación lineal*: Si la función objetivo y todas las restricciones son funciones lineales de las variables de diseño; el problema de programación matemática se denomina problema de programación lineal (PL).

Esta clasificación resulta útil a la hora de aplicar un método efectivo para solucionar el problema ya que existen métodos especiales disponibles para la solución eficiente de una clase particular de problemas. Los métodos de programación cuadrática secuencial pueden ser aplicados a problemas cuadráticos mientras que el método simplex puede ser aplicado a un problema de programación lineal, por ejemplo. Luego, la primera tarea de un diseñador sería investigar la clase de problema y seleccionar el método correspondiente.

2.5.5. Clasificación basada en los valores permisibles de las variables de diseño

Dependiendo de los valores permitidos para las variables de diseño, los problemas de optimización se pueden clasificar como problemas de programación enteros y de valor real:

1. *Problema de programación entera*: Se refiere a problemas donde algunas o todas las variables de diseño $x_1, x_2, ..., x_n$ están restringidas para tomar solo valores enteros (o discretos).
2. *Problema de programación real*: Cuando se permite que todas las variables de diseño tengan un valor de los reales, el problema de optimización se denomina problema de programación de valor real.

2.5.6. Clasificación basada en la naturaleza determinista de las variables

Respecto a la naturaleza determinista de las variables involucradas, los problemas de optimización se pueden clasificar como:

1. *Problemas determinísticos*: En esta categoría se incluyen la mayoría de los problemas de optimización. En estos problemas se asume que si ciertas acciones son realizadas se podrá predecir con certeza tanto el resultado que se obtiene como los requerimientos para realizar estas acciones [36].
2. *Programación estocástica*: Un problema de programación estocástica es un problema de optimización en el que algunos o todos los parámetros (variables de diseño y / o parámetros preasignados) son probabilísticos (no deterministas o estocásticos).

2.5.7. Clasificación basada en la separabilidad de las funciones

Los problemas de optimización pueden clasificarse como problemas de programación *separables* y *no separables* basados en la separabilidad de la función objetivo y las restricciones. Un problema de programación separable es aquel en el que la función objetivo y las restricciones son separables. Una función $f(\vec{x})$, con $\vec{x} \in \mathbb{R}^n$ es separable si se puede expresar como la suma de n funciones de uni-variables, $f_1(x_1), f_2(x_2), \cdots, f_n(x_n)$.

2.5.8. Clasificación basada en el número de funciones objetivo

Dependiendo de la cantidad de funciones objetivo que se deben minimizar, los problemas de optimización se pueden clasificar como problemas de programación de *mono o multi objetivo*. Un problema de optimización multiobjetivo involucra múltiples funciones objetivo, por tanto, existe una posibilidad de conflicto. Una forma simple de manejar el problema es construir una función objetivo general como una combinación lineal de las funciones objetivo múltiples que entran en conflicto. Así, si $f_1(\vec{x})$ y $f_2(\vec{x})$ denotan dos funciones objetivo, se construye una nueva función objetivo global para el problema de optimización.

2.6. Conclusiones del capítulo

En el capítulo se plantean una serie de conceptos que sirven como marco teórico para la presente investigación. Se describen las características de los principales componentes de un problema de optimización: la función objetivo, las variables de diseño y las restricciones. Se presentan los conceptos de mínimos locales y globales mediante el análisis de convexidad de la función objetivo y su modalidad. Una vez planteado el problema de optimización se procede a la aplicación de un algoritmo capaz de resolverlo. Existen métodos diseñados para solucionar un tipo específico de problemas de optimización. En consecuencia, conocer la clasificación del problema a resolver resulta una tarea necesaria.

CAPÍTULO 3

Programación Matemática

La disciplina que abarca la teoría y la práctica de los métodos de optimización numérica se conoce como programación matemática [31]. El término programación se comienza a utilizar alrededor de 1940 para describir la planificación o programación de actividades dentro de una organización. Los programadores encontraron que podían representar la cantidad o el nivel de cada actividad como una variable cuyo valor debía determinarse. Luego podrían describir matemáticamente las restricciones inherentes a la planificación o problema de programación como un conjunto de ecuaciones o desigualdades que involucran las variables. Una solución a todas estas restricciones se consideraba como un plan o programa aceptable. Más tarde, se utilizaría el término matemático de programación para describir la minimización o maximización de una función objetivo de varias variables, sujeta a un conjunto de restricciones [37].

Actualmente, la programación matemática es una de las ramas de la investigación de operaciones más desarrolladas y usadas. Las técnicas de programación matemática pueden ser aplicadas para resolver problemas de optimización altamente complejos como problemas no lineales de alta dimensionalidad, y múltiples restricciones. La teoría matemática en esta área es aplicada tanto para caracterizar puntos óptimos del problema como para proporcionar la base para la mayoría de los algoritmos [30]. Por lo tanto, en las siguientes secciones se tratan los temas de condiciones de optimalidad de primer y segundo orden para funciones. Luego, se describen los métodos básicos de Programación Matemática identificando sus fortalezas y debilidades.

3.1. Información de la Derivada

El cálculo diferencial es la rama de la matemática que analiza las razones de cambio de cantidades relacionadas, estas pueden considerarse como la pendiente de una tangente a un gráfico de una función. La derivada es el objeto de estudio central del cálculo diferencial donde la idea principal detrás de este concepto es que, en una función f evaluada en un punto a, la función f se aproxima a una función lineal en vecindad de a. Una definición formal plantea lo siguiente [38]:
Definición 1. Sea A cualquier subconjunto abierto no vacío de $\mathbb{R}$, y $a \in A$. Para cualquier función $f : A \to \mathbb{R}$, la derivada de f en $a \in A$ es el límite (si existe):

$$\lim_{h \to 0, h \in U} \frac{f(a+h) - f(a)}{h} \tag{3.1}$$

donde $U = \{h \in \mathbb{R} \,|\, a + h \in A, h \neq 0\}$ este límite se denota como derivada $f'(a)$ o $\frac{\partial f}{\partial x}(a)$. Cuando la derivada existe en todo $a \in A$ se dice que la función es *diferenciable*.

La derivada de una función f' es otra función, que evaluada en a es la pendiente de la recta tangente a la curva f si esta es representada gráficamente. Este concepto es sumamente importante para la teoría de la programación matemática, ya que una recta horizontal tangente a un punto de f describe un cambio de comportamiento en la función. Los puntos cuyas rectas tangentes son horizontales se denominan *puntos estacionarios* y pueden satisfacer la condición de óptimo de la función. Esto no siempre se cumple ya que puede tratarse de un punto de inflexión. Luego, el cálculo de los puntos estacionarios de la función consiste en hallar aquellos cuya recta tangente tenga pendiente igual cero, esto es $f'(x) = 0$.Véase Figura 3.1 . Este procedimiento se conoce como la *prueba de la primera derivada*.

Los puntos estacionarios pueden ser mínimos, máximos o puntos de inflexión. Para diferenciar los puntos es necesario utilizar información de la segunda derivada. La segunda derivada de la función indica el cambio en la pendiente. Una función con $f''(x) > 0$ es convexa en x. O sea, la función se curva hacia arriba a medida que aumenta la pendiente. Por otra parte, una función con pendiente decreciente, $f''(x) < 0$ es cóncava hacia abajo en x. Entonces, la *prueba de la segunda derivada* es capaz de diferenciar mínimos y máximos de la siguiente forma:

Definición 2. Dada una función diferenciable $f : A \to \mathbb{R}$ y un punto $x \in A$ se cumple que:

1. Si $\frac{\partial f}{\partial x} = 0$ y $\frac{\partial^2 f}{\partial^2 x} > 0$, el punto x es un mínimo.
2. Si $\frac{\partial f}{\partial x} = 0$ y $\frac{\partial^2 f}{\partial^2 x} < 0$, el punto x es un máximo.

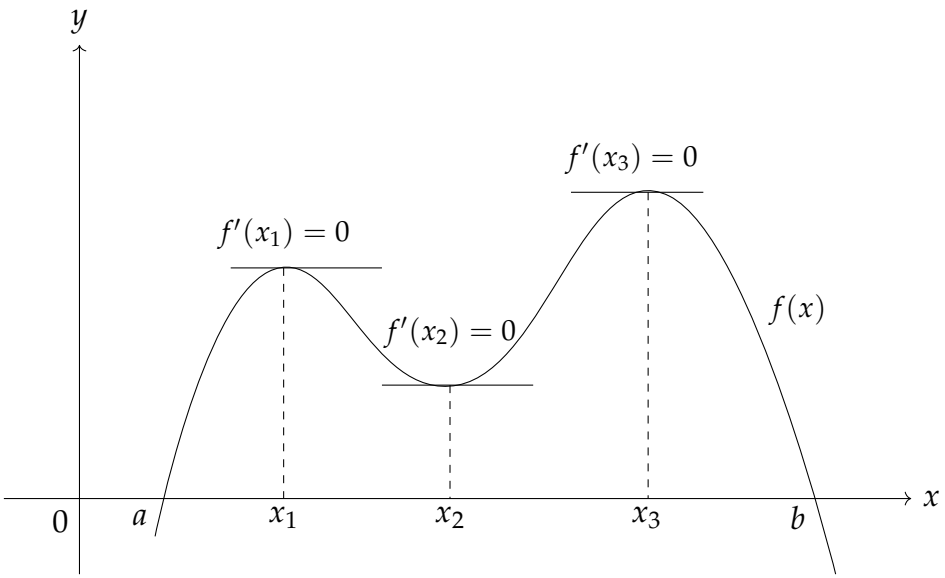


FIGURA 3.1: Puntos estacionarios y sus rectas tangentes.

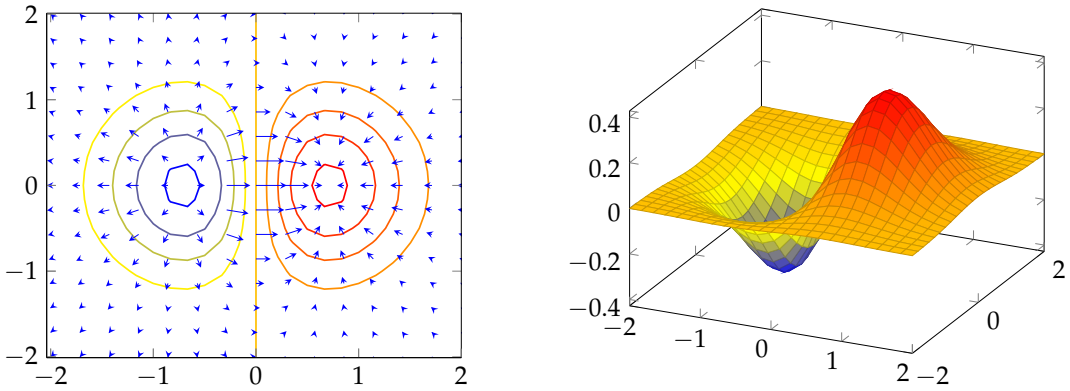
A la izquierda de un mínimo, la función decrece. A la derecha la curva se eleva. La pendiente ha cambiado de negativa a positiva. El gráfico se dobla hacia arriba y $f''(x) > 0$. Por el contrario a la derecha del máximo, la pendiente pasa de positiva a negativa en la vecindad del mínimo. En el caso excepcional, cuando $f'(x) = 0$ y también $f''(x) = 0$ la regla no se cumple y el punto puede ser cualquiera de los tres tipos de puntos estacionarios. Se debe señalar que la información de $f'(x)$ y $f''(x)$ es sólo local. Para encontrar un mínimo o máximo global, se necesita información sobre todo dominio. Además se asume que la función es continua y diferenciable en todo su dominio, lo cual no siempre se satisface [39].

3.2. Información del Gradiente

Los métodos descritos en la sección anterior solo pueden ser aplicados para funciones univariable. Para analizar el comportamiento de funciones de más de una variable se utiliza la información del gradiente de la función. Si una función multivariable $f(\vec{x})$, tiene una derivada continua de primer orden $f'(\vec{x})$ el gradiente de $f(\vec{x})$ se define como un vector compuesto por las derivadas parciales de la función objetivo respecto a cada variable:

$$\nabla f(\vec{x}) = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right]^T \tag{3.2}$$

La derivada parcial $\frac{\partial f}{\partial x_i}$ con $i = \{1, 2, \ldots, n\}$ trata las restantes $n - 1$ variables como constante y viceversa. Cada derivada parcial $\frac{\partial f}{\partial x_i}$ evaluada en un punto cualquiera denota la pendiente en la dirección x_i. Si se concibe la función como una hiper-superficie, el gradiente describe en qué dirección la superficie toma un valor máximo. Por tanto indica la dirección de ascenso, y su magnitud $|\nabla f(\vec{x})|$ indica el grado de inclinación de la superficie [39]. La Figura 3.2 muestra a la izquierda el capo vectorial de la superficie $xe^{(-x^2-y^2)}$. Se puede observar cómo la dirección del gradiente indica hacia donde la superficie asciende y su longitud es mayor en áreas con cambios de ascenso máximo. Luego en los puntos donde la función se hace plana, o llega a su valor máximo o mínimo, el gradiente tiende a desaparecer.



(A) Campo vectorial (B) Superficie

FIGURA 3.2: Función $x \exp(-x^2 - y^2)$

De lo planteado en la sección anterior se entiende que las derivadas parciales tomarán valores de cero para los puntos estacionarios de la función. Luego, el cálculo de los puntos estacionarios para una función multivariable consiste en resolver el siguiente sistema de ecuaciones [40]:

$$\frac{\partial f}{\partial x_1} = 0 \tag{3.3}$$

$$\frac{\partial f}{\partial x_2} = 0 \tag{3.4}$$

$$\vdots$$

$$\frac{\partial f}{\partial x_n} = 0 \tag{3.5}$$

Las Ecuaciones 3.3, 3.4 y 3.5 representan un sistema de n ecuaciones con n incógnitas. Luego, para determinar determinar cuáles de los puntos estacionarios son mínimos o máximos se utiliza la información proporcionada por la matriz Hessiana. La matriz Hessiana representa la segunda derivada de una función de más de una variable. Si la función f tiene segundas derivadas parciales, entonces la matriz de Hessiana se define como:

$$H(\vec{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial^2 x_1} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial^2 x_2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial^2 x_n} \end{bmatrix} \tag{3.6}$$

Si la matriz $H(\vec{x})$ es positiva definida, esto es, que su determinante sea mayor que cero; entonces $\vec{x}$ representa un mínimo de la función f, en cambio si la matriz $H(\vec{x})$ es definida negativa, entonces $f(\vec{x})$ es un máximo [31]. En próximas secciones se describe cómo la matriz Hessiana no solo puede ser utilizada para determinar condiciones de optimalidad, sino que también se utiliza para calcular direcciones de búsqueda en métodos basados en gradiente.

3.3. Métodos de Optimización Univariable

A pesar de que los problemas no restringidos, con función objetivo de una sola variable son poco comunes en situaciones prácticas, la determinación del mínimo de una función de valor real univariable, desempeña un rol activo en la optimización no lineal. Los métodos univariables son importantes primeramente desde el punto de vista teórico y demostrativo. Además, como se verá en la Sección 3.4, este tipo de procedimientos de minimización se pueden aplicar varias veces en un problema multivariable. En efecto, los algoritmos de optimización univarible forman parte de los bloques básicos de operación de los algoritmos de optimización multivariables. Estos métodos se clasifican de forma general en algoritmos basados en gradiente y no basados en gradiente. En las siguientes secciones se discuten algunos de los más populares. Los dos primeros métodos son la Búsqueda de Fibonacci y el método de la Sección Dorada, los cuales entran en la clasificación de los no basados en gradiente y además son métodos de eliminación de regiones bien conocidos. Por último se presenta el método de Newton-Raphson, el cual utiliza información de la primera y la segunda derivada para encontrar la dirección óptima de búsqueda.

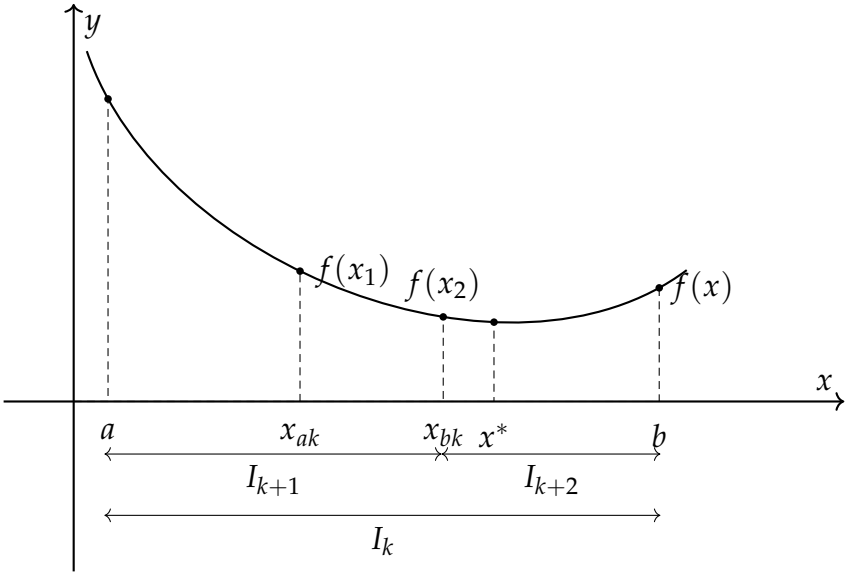


FIGURA 3.3: Reducción de intervalos en el método de Fibonacci.

3.3.1. Método de Fibonacci

El método de Fibonacci se considera como un método de eliminación de regiones pues reduce el intervalo de búsqueda de acuerdo a la serie de Fibonacci definida por la recursión $F_k = F_{k-1} + F_{k-2}$. Esta expresión genera la secuencia, $F_i = \{1, 1, 2, 3, 5, 8, 13, ..., F\} = \{F_0, F_1, F_2, F_3, F_4, F_5, F_6, \dots, F_n\}$, conocida como secuencia de Fibonacci que se produce en varias ramas de las matemáticas. Este método realiza una serie de iteraciones acotando el intervalo de acuerdo a la siguiente razón:

$$I_n = \frac{I_0}{F_n} \tag{3.7}$$

Esto es, que para la iteración n el tamaño del intervalo original será reducido al n-ésimo número de Fibonacci. La secuencia de intervalos de Fibonacci sólo se puede generar si se conoce n. Luego, si el objetivo de la optimización es encontrar x^* dentro de una tolerancia prescrita, el n requerido se puede deducir fácilmente usando Ecuación 3.7. Los métodos de eliminación de regiones asumen que $f(x)$ es estrictamente unimodal en I, donde siempre se divide el intervalo actual en dos regiones; el intervalo que contenga un punto con menor valor de la función objetivo será elegido y se convierte en el intervalo actual. Por tanto, se conoce que n será menor si el mínimo de $f(x)$ es poco profundo y alto si $f(x)$ varía rápidamente en la vecindad de la solución.

Los principios anteriores se pueden usar para implementar el método. Se comienza definiendo un intervalo I_0 y los límites iniciales inferior y superior $x_{L\,0}$ y $x_{U\,0}$ respectivamente. Donde $x_{a\,0}$ y $x_{b\,0}$ son dos puntos interiores en el intervalo I_k. Además, se conoce el valor de n, y la descripción matemática de $f(x)$. En cada

iteración se calculan intervalos sucesivos, se evalúa $f(x)$, se actualiza y selecciona el intervalo correspondiente. En la k-ésima iteración, los valores de x_{Lk}, x_{Uk}, x_{ak}, x_{bk}, $f(x_{ak})$, $f(x_{bk})$ y el intervalo I_{k+1} son conocidos. Luego, el valor de I_{k+2} se calcula mediante la siguiente expresión:

$$I_{k+2} = \frac{F_{n-k-1}}{F_{n-k}} I_{k+1} \tag{3.8}$$

La actualización de los puntos interiores y los intervalos se realiza atendiendo las siguientes condiciones:

1. Si $f(x_{ak}) > f(x_{bk})$ entonces x^* está en el intervalo $[x_{ak}, x_{Uk}]$ por tanto $x_{Lk+1} = x_{Lk}$ y $x_{Uk+1} = x_{bk}$ son los nuevos límites del intervalo I_{k+1}. Luego, los puntos interiores de este intervalo serían $x_{ak+1} = x_{bk}$ y $x_{bk+1} = x_{Lk+1} + I_{k+2}$.
2. Si $f(x_{ak}) < f(x_{bk})$ entonces x^* está en el intervalo $[x_{Lk}, x_{bk}]$ por tanto $x_{Lk+1} = x_{ak}$ y $x_{Uk+1} = x_{Uk}$. Los puntos interiores de este intervalo toman entonces los siguientes valores $x_{ak+1} = x_{Uk+1} - I_{k+2}$ y $x_{bk+1} = x_{ak}$.

Aunque es poco probable, se puede dar el caso en que $f(x_{ak}) = f(x_{bk})$. Entonces se pueden aplicar cualquiera de las asignaciones anteriores ya que x^* está contenido en ambos intervalos $[x_{Lk}, x_{bk}]$ y $[x_{ak}, x_{Uk}]$. Este procedimiento se repite hasta que $k = n - 2$ en cuyo caso $I_k + 2 = I_n$ y $x^* = x_{ak+1} = x_{bk+1}$ [31].

3.3.2. Método de la Sección Dorada

Este método tiene sus antecedentes en la antigua Grecia donde los arquitectos griegos afirmaban que un edificio con lados d y b satisficiera la siguiente relación tendría mejores propiedades:

$$\frac{d+b}{d} = \frac{d}{b} = \phi = 1.618034 \tag{3.9}$$

La proporción descrita por la ecuación anterior se denomina razón dorada. En este método, como en la búsqueda de Fibonacci, se genera una secuencia de intervalos $\{I_1, I_2, I_3, ...\}$ utilizando la razón dorada. Sin embargo, a diferencia del método de Fibonacci, no es necesario especificar el número total de iteraciones como parámetro de inicio. La regla según la cual se generan las longitudes de los intervalos sucesivos es que la relación de dos intervalos adyacentes debe ser

constante igual a ϕ, es decir:

$$\begin{aligned} I_1 &= I_2 + I_3 \\ I_2 &= I_3 + I_4 \\ &\vdots \\ \frac{I_2}{I_1} = \frac{I_3}{I_3} &= \frac{I_4}{I_3} = \ldots = \phi \end{aligned} \tag{3.10}$$

Si toma la primera ecuación a conveniencia y se sustituye $I_2 = \phi I_1$ y $I_3 = \phi I_2 = \phi^2 I_1$ se obtiene la ecuación:

$$\phi^2 + \phi - 1 = 0 \tag{3.11}$$

Cuyas raíces son $\phi = \frac{1 \pm \sqrt{5}}{2}$. La estrategia de reducción de intervalo en el algoritmo de sección dorada sigue los pasos utilizados en el algoritmo de Fibonacci, excepto que la Ecuación 3.8 se reemplaza $\frac{F_{n-k-1}}{F_{n-k}}$ por la razón áurea $\phi = \frac{1-\sqrt{5}}{2}$. El método de sección dorada es un método robusto que se puede integrar de manera eficiente con otros métodos de búsqueda multivariable [32].

3.3.3. Método de Newton-Raphson

El método fue diseñado por Joseph Raphson, basándose en los métodos de Isaac Newton para evaluar la raíz de una ecuación usando una secuencia de polinomios. El método de Newton-Raphson es una técnica de búsqueda de raíz en la que se evalúa la ecuación $f'(x) = 0$. Usando la serie de Taylor, la función $f'(x)$ se puede aproximar como:

$$f'(x_k) + f''(x_k)\Delta x \tag{3.12}$$

Donde $\Delta x = x_{k+1} - x_k$. Haciendo la ecuación anterior igual a cero, el siguiente punto de aproximación puede darse como:

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)} \tag{3.13}$$

El procedimiento realizado por el método es iterativo comenzando por un punto x_0. Luego en la k-ésima iteración se mejora de acuerdo a la ecuación anterior, hasta que la diferencia entre x_k y x_{k-1} sea menor que un tolerancia ϵ. Al igual que la mayoría de los métodos de programación matemática, la convergencia del método de Newton-Raphson es sensible al punto inicial. Además, de que la convergencia se ralentiza cuando el valor de la derivada es cercano a cero, en este método se asume que la segunda derivada de la función existe lo cual puede limitar su aplicación [29].

3.4. Métodos de optimización multivariable

Las técnicas de solución para problemas de optimización multivariable y sin restricciones se pueden agrupar en métodos basados en gradiente y métodos directos de búsqueda. Los métodos basados en gradientes requieren información de la derivada de la función para generar direcciones de búsqueda hacia el mínimo. En los métodos directos, la solución se obtiene utilizando únicamente evaluaciones de la función objetivo. El enfoque general es explorar el espacio de búsqueda mediante una estrategia sistemática con el fin de encontrar una trayectoria que conduzca progresivamente a valores reducidos de la función objetivo. Los métodos de búsqueda multidimensionales pueden ser considerados como análogos a sus contrapartes unidimensionales, y al igual que estos últimos, presentan deficiencias similares en cada grupo.

Los métodos basados en gradiente se pueden agrupar en dos clases, métodos de primer orden y segundo orden. Los métodos de primer orden se basan en la aproximación lineal de la serie de Taylor y, por lo tanto, implican sólo el gradiente de la función. Los métodos de segundo orden, por otro lado, se basan en la aproximación cuadrática de las series de Taylor por lo que utilizan además información de la matriz Hessiana. La información del gradiente, la matriz Hessiana o la combinación de ambos, permite encontrar una dirección de búsqueda donde la función tienen su mayor inclinación de descenso. Una vez que se identifica la dirección de búsqueda, se necesita evaluar cuánto se debe mover el punto actual en esa dirección para minimizar la función. Este es un problema unidimensional en cual se puede aplicar cualquiera de los métodos de optimización univariable descritos en las secciones anteriores [33].

Los métodos directos no requieren información de la primera o segunda derivada para encontrar la dirección de búsqueda. En su lugar, ésta se determina mediante las evaluaciones de función, así como por las direcciones de búsqueda calculadas a partir de iteraciones anteriores. Esta característica, permite que este tipo de métodos puedan ser aplicados a un espectro más amplio de problemas de optimización [32]. En las secciones correspondientes, los algoritmos de los métodos directos serán descritos con mayor detalle, debido a que, por sus características son los principales candidatos para la hibridación con operadores de algoritmos evolutivos.

3.4.1. Método del Descenso Inclinado

El funcionamiento básico de la mayoría de las técnicas de optimización basadas en gradiente, consiste en encontrar la tasa de cambio de una función objetivo con respecto a un parámetro λ a lo largo de una dirección determinada, S_k, que se aleja de un punto $\vec{x}_k$. Cualquier punto en la dirección S_k se puede expresar

$\vec{x}_{k+1} = \vec{x}_k + \lambda S_k$. Si el objetivo es la minimización, entonces la tarea consiste en encontrar la tasa de cambio respecto a λ en la dirección S_i que minimice la función objetivo. Si se conoce que el gradiente de una función $\nabla f(x)$ indica la dirección de ascenso, y su magnitud $|\nabla f(\vec{x})|$ el grado de inclinación de la función en esa dirección; el negativo del gradiente nos indica dirección hacia donde existe mayor descenso de la función.

La utilización del negativo del vector de gradiente como una dirección para la minimización fue realizado por primera vez por Cauchy en 1847. En este método, también conocido como método de Cauchy, se comienza desde un punto inicial $\vec{x}_0$ y se avanza iterativamente siguiendo las direcciones de descenso cada vez más inclinadas hasta encontrar el punto óptimo. El método de descenso inclinado se puede resumir en los siguientes pasos [29]:

1. Definir un punto inicial arbitrario $\vec{x}_0$ y número de iteración $k = 0$
2. Calcular la dirección de búsqueda $S_k = -\nabla f(\vec{x}_k)$
3. Determinar la longitud óptima del paso λ_k^* en la dirección S_k que minimice la función objetivo en el punto:

   $$\vec{x}_{k+1} = \vec{x}_k + \lambda_k^* S_k \tag{3.14}$$

   Esto es,

   $$\vec{x}_{k+1} = \vec{x}_k - \nabla f(\vec{x}_k)\lambda_k^* \tag{3.15}$$

4. Probar optimalidad del nuevo $\vec{x}_{k+1}$. Si $\vec{x}_{k+1}$ es óptimo, terminar el proceso. De lo contrario, ir al paso 5.
5. Actualizar el número de iteración $k = k + 1$ e ir al paso 2.

El método de descenso inclinado es el método más representativo de los basados en gradiente. Es importante destacar que para hallar λ_k^* se debe minimizar la función que describe la Ecuación 3.15, conocidos x_i y el vector gradiente evaluado en este punto. En este caso se puede aplicar cualquiera de los métodos de búsqueda unidimensional. Si bien es cierto que búsqueda unidimensional comienza en la mejor dirección, la dirección de descenso más pronunciado es una propiedad local, por tanto el método no es realmente efectivo en la mayoría de los problemas con funciones multimodales.

3.4.2. Método de Newton

El método de Newton se comporta de manera similar al de Cauchy. Sin embargo, utiliza las segundas derivadas o la matriz de Hessiana de la función objetivo para calcular la dirección de búsqueda. Este método, cuando converge, lo hace a un ritmo más rápido que los métodos de primer orden como el método de Cauchy.

La idea fundamental aquí, es construir una aproximación cuadrática a la función $f(x)$ y realizar su minimización. Por lo tanto, en un punto x_k, se construye la aproximación cuadrática de la función [32]:

$$q(\vec{x}) = f(\vec{x}_k) + \nabla f(\vec{x}_k)^T(\vec{x} - \vec{x}_k) + \frac{1}{2}(\vec{x} - \mathbf{x}_k)^T \nabla^2 f(\vec{x})(\vec{x} - \vec{x}_k) \tag{3.16}$$

El mínimo de esta ecuación se encuentra en $\nabla q(\vec{x}) = 0$, asumiendo que $\nabla^2 f(\vec{x})$ es positiva y definida, y denotamos $S_k = (\vec{x} - \vec{x}_k)$ tenemos que:

$$\left[\nabla^2 f(\vec{x})\right] S_k = -\nabla f(\vec{x}) \tag{3.17}$$

Despejando $\left[\nabla^2 f(\vec{x})\right]$ de la ecuación anterior se obtiene la dirección de búsqueda:

$$S_k = -\nabla f(\vec{x}) \left[\nabla^2 f(\vec{x})\right]^{-1} \tag{3.18}$$

Nótese que $\left[\nabla^2 f(\vec{x})\right]$ es la matriz Hessiana $H(\vec{x})$ de f y $\left[\nabla^2 f(\vec{x})\right]^{-1}$ su inversa. A partir de este punto, el método se comporta de forma similar al de Cauchy, generando una secuencia de puntos $\vec{x}_k$ en direcciones descendentes S_k. El punto límite de esta secuencia es el óptimo $\vec{x}^*$ donde $\nabla f(\vec{x}^*) = 0$.

De la Ecuación 3.16, se deriva que si la función es cuadrática con una matriz Hessiana definida, entonces el método de Newton logrará converger en la primera iteración. Sin embargo, para funciones no cuadráticas el método de Newton no converge a menos que el punto de partida sea cercano al óptimo. Si la función presenta cambios de modalidad frecuentes en su dominio, una aproximación cuadrática puede resultar inefectiva.

Por otra parte, las condiciones de suficiencia que requieren que $H(\vec{x}^*)$ sea positiva definida en $\vec{x}^*$ no imponen restricciones a la H durante el proceso iterativo en un punto $\vec{x}_k$. Entonces, si $H(\vec{x}_k)$ no es positivo o si es singular, entonces $q(\vec{x})$ puede no tener un mínimo. Por tanto, la Ecuación 3.17 puede no tener solución, o si puede resolverse, puede empeorar punto que el punto actual. Por supuesto, si f es estrictamente convexa, entonces su Hessiana es siempre definida y el método de Newton puede aplicarse de manera efectiva [32].

3.4.3. Método Nelder-Mead

Basándose en el trabajo de Spendley [41] donde se plantea una técnica para el seguimiento de condiciones operativas óptimas, mediante la evaluación de la salida de un sistema en un conjunto de puntos que forman un simplex en el espacio, Nelder y Mead propusieron un algoritmo que realiza la minimización de una función en un espacio de n variables mediante operaciones sobre un simplex que se

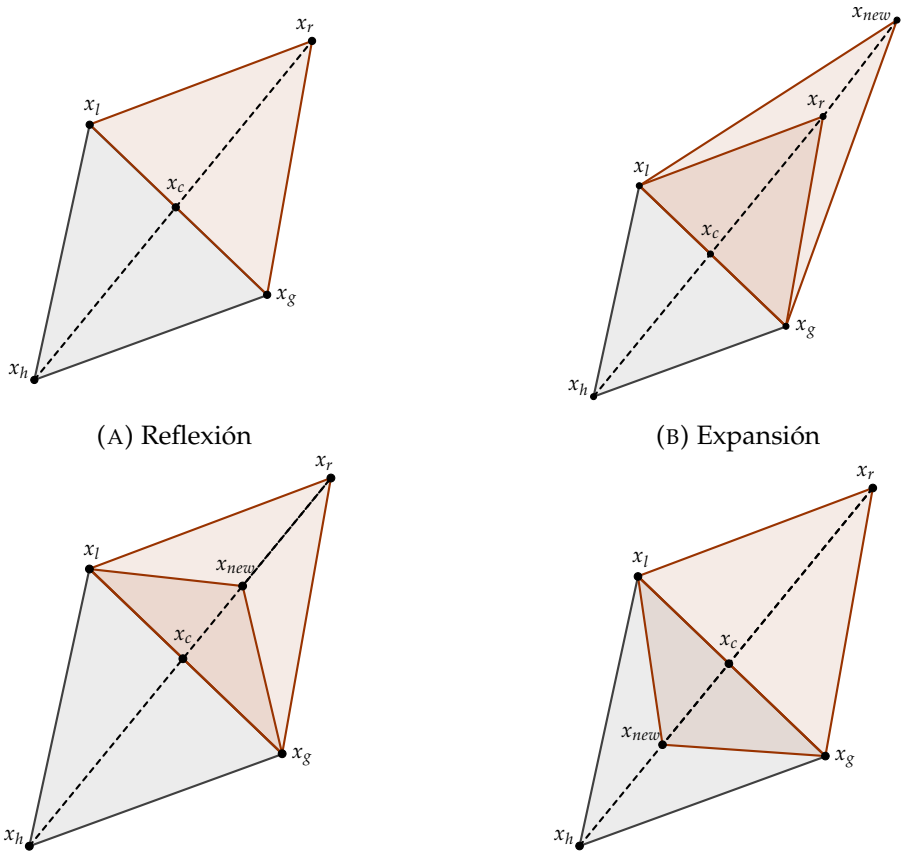
adapta al "paisaje local", alargándose por largos planos inclinados, cambiando de dirección al encontrar un descenso en un ángulo determinado y contrayéndose en la vecindad de un mínimo. El algoritmo original fue diseñado para optimización sin restricciones [42].

Algoritmo 1 Método de Nelder-Mead

1: Elegir parámetros: $\beta > 0, \gamma \in (0,1)$ y un parámetro de finalización ϵ.
2: Generar un simplex inicial.
3: **while** $\epsilon \leq \left\{ \sum_{i=1}^{N+1} \frac{((x_i)-f(xc))^2}{N+1} \right\}^{\frac{1}{2}}$ **do**
4: Elegir $\vec{x}_h$ (peor punto), $\vec{x}_l$ (mejor punto) y $\vec{x}_g$ (el segundo peor punto).
5: Calcular $\vec{x}_c = \frac{1}{N} \sum_{i=1, i \neq h}^{N+1} x_i$
6: Realizar la reflexión $\vec{x}_r = 2\vec{x}_c - \vec{x}_h$
7: **if** $f(\vec{x}_r) < f(\vec{x}_l)$ **then**
8: Hacer $\vec{x}_{new} = (1+\gamma)\vec{x}_c - \gamma\vec{x}_h$ (Expansión)
9: **else**
10: **if** $f(\vec{x}_r) \geq f(\vec{x}_h)$ **then**
11: Hacer $\vec{x}_{new} = (1-\beta)\vec{x}_c + \beta\vec{x}_h$ (Contracción adentro)
12: **end if**
13: **else**
14: **if** $f(\vec{x}_g) < f(\vec{x}_r) < f(\vec{x}_h)$ **then**
15: Hacer $\vec{x}_{new} = (1+\beta)\vec{x}_c - \beta\vec{x}_h$ (Contracción afuera)
16: **end if**
17: **end if**
18: Calcular $f(\vec{x}_{new})$ y remplazar $\vec{x}_h$ por $\vec{x}_{new}$
19: **end while**

El Algoritmo 1 es una versión ligeramente modificada de la original, planteada en [43]. La condición de parada es el error estándar entre los puntos del simplex. El desempeño del Método de Nelder-Mead (NMM por sus siglas en inglés) depende de los parámetros γ (parametro de expansión) y β (parámetro de contracción). Si los valores de γ o $\frac{1}{\beta}$ aumentan se llegará más rápido a la vecindad del óptimo pero la convergencia al mismo puede dificultarse. En cambio si valores menores de γ o $\frac{1}{\beta}$ son seleccionados se necesitarán más evaluaciones de la función objetivo pero se logra un mayor acercamiento al mínimo.

Como se muestra en la Figura 3.4 se realizan una serie de pasos, de acuerdo a determinadas condiciones. Una vez calculado el centroide $\vec{x}_c$ y la reflexión correspondiente a $\vec{x}_h$ en el punto $\vec{x}_r$, se realiza una expansión del simplex en la dirección de $\vec{x}_r$ si este mejora el valor de la función respecto a $\vec{x}_h$. Si el punto de reflexión es peor o igual a $\vec{x}_h$ se realiza una contracción hacia adentro donde el punto $\vec{x}_{new}$ resultante se encuentra entre $\vec{x}_h$ y x_c. En cambio, si $\vec{x}_r$ se encuentra entre $\vec{x}_g$ y $\vec{x}_h$ se realiza una contracción hacia afuera donde el punto $\vec{x}_{new}$ queda entre $\vec{x}_c$ y $\vec{x}_r$. En caso de que las condiciones anteriores no se cumplan $\vec{x}_{new}$ será igual a $\vec{x}_r$.



(A) Reflexión (B) Expansión

(C) Contracción hacia afuera (D) Contracción hacia dentro

FIGURA 3.4: Posibles movientos del simplex durante una iteración del Nelder Mead para $n = 2$

3.4.4. Método de Hooke-Jeeves

Al igual que el método Nelder-Mead, el método de Hooke y Jeeves, es un método directo. El patrón de búsqueda en este algoritmo genera una serie de direcciones de forma iterativa de tal manera que se cubra todo el espacio de búsqueda. Esto significa que, las direcciones generadas deben ser tales que partiendo de cualquier punto en el espacio se debe poder llegar cualquier otro punto siguiendo solo las direcciones generadas.

El método realiza dos operaciones principales: movimientos de exploración y movimientos según patrones heurísticos. La exploración se realiza en vecindad del punto actual de forma sistemática para para encontrar el punto mejor alrededor del punto actual. Tanto el punto actual como el mejor punto encontrado a su alrededor se utilizan para realizar un movimiento de patrón:

1. **Exploración**: Se denota $\vec{x}_c$ como el punto actual y $\vec{x}$ el mejor punto en la vecindad de $\vec{x}_c$, se asigna $\vec{x} = \vec{x}_c$ y el número de la iteración $i = 1$. Se realizan las siguientes operaciones:
   a) Calcular $f = f(\vec{x})$, $f^+ = f(x_i + \Delta_i)$, $f^- = f(x_i - \Delta_i)$.
   b) Encontrar $f_{min} = min(f, f^+, f^-)$. Hacer $\vec{x}_k$ corresponder a f_{min}.
   c) Si $i < N$, actualizar $i = i + 1$ e ir al paso *a*, de otra manera ir al paso *d*.
   d) Si $\vec{x} \neq \vec{x}_k$ la exploración ha tenido éxito. Sino la exploración ha fallado.

2. **Búsqueda de patrón**: Se calcula un nuevo punto $\vec{x}_{k+1}$ saltando en la dirección que conecta el punto previo $\vec{x}_{k-1}$ con el mejor punto actual $\vec{x}_k$:

$$\vec{x}_{k+1} = \vec{x}_k + (\vec{x}_k - \vec{x}_{k-1}) \tag{3.19}$$

El método continúa iterativamente realizando aplicaciones de ambos operadores exploración y saltos mediante el movimiento de patrón. Si este último no mueve el punto actual a una mejor región el movimiento de patrón no es aceptado y la extensión del rango de exploración es reducido. El Algoritmo 2 describe el método de acuerdo a lo planteado en [43]. Nótese, que una deficiencia de este método es la cantidad de evaluaciones de la función objetivo que se realizan por iteración. El operador de exploración necesita evaluar $2n$ puntos en la vecindad de x_k y la búsqueda de patrón puede ser ineficiente si cuando se encuentra un descenso, el punto x_{k-1} se encuentra cerca a x_k por lo que los saltos serán cada vez más pequeños, realizándose evaluaciones innecesarias.

Algoritmo 2 Método de Jookes-Jeeves

Elegir valores para: punto inicial $\vec{x}_0$, vector de incrementos de variables Δ, parámetro de reducción de paso $\alpha > 1$. Hacer $k = 1$, y elegir un parámetro de finalización ϵ.
while $\epsilon < ||\Delta||$ **do**
  Hacer $\vec{x} = \vec{x}_k$. Comienza la exploración
  for $i \in \{1, \ldots, N\}$ **do**
    Calcular $f = f(\vec{x})$, $f^+ = f(x_i + \Delta_i)$, $f^- = f(x_i - \Delta_i)$
    Encontrar $f_{min} = min(f, f^+, f^-)$
    Hacer x corresponder a f_{min}
  end for
  if $\vec{x} \neq \vec{x}_k$ **then**
    Hacer $\vec{x}_k = \vec{x}$, $\vec{x}_{k-1} = \vec{x}_k$, $k = k + 1$.Si la exploración es exitosa hacer movimiento de patrón.
    while $f(\vec{x}_{k+1}) < f(\vec{x}_k)$ **do**
      Calcular $\vec{x}_{k+1} = \vec{x}_k + (\vec{x}_k - \vec{x}_{k-1})$
      Hacer $\vec{x} = \vec{x}_{k+1}$. Hacer exploración.
      for $i \in \{1, \ldots, N\}$ **do**
        Calcular $f = f(\vec{x})$, $f^+ = f(x_i + \Delta_i)$, $f^- 0 f(x_i - \Delta_i)$
        Encontrar $f_{min} = min(f, f^+, f^-)$
        Hacer x corresponder a f_{min}
      end for
      Hacer $\vec{x}_{k+1} = x$ y $\vec{x}_k = \vec{x}_{k+1}$
    end while
  end if
  for $i \in \{1, \ldots, N\}$ **do**
    Hacer $\Delta_i = \Delta_i - \alpha$
  end for
end while

3.5. Técnicas de optimización con restricciones

Los métodos descritos en las secciones anteriores realizan la optimización de un problema sin tener en cuenta las restricciones. En problemas reales la ausencia de restricciones es poco probable, por esta razón se han desarrollado diferentes técnicas de programación matemática diseñadas para resolver problemas restringidos. Cuando se aplica un técnica clásica como la condición de optimalidad de primer orden $f'(x) = 0$ en la función objetivo, el mínimo se puede obtener en cierto punto x^*. Sin embargo, en presencia de una restricción que excluya a x^*, el mínimo ocurre en un punto factible x'^*. En este escenario las condiciones de optimalidad pueden no identificar el óptimo del problema. A continuación se discuten primeramente las condiciones de optimalidad con restricciones mediante el Método de Multiplicadores de Lagrange, así como diferentes técnicas para la solución de problemas con restricciones tales como los Métodos de Penalización, y la Programación Cuadrática Secuencial.

3.5.1. Método de Multiplicadores de Lagrange

Este método consiste en encontrar las raíces de la función Lagrangiana, la cual esta compuesta por la función objetivo más la sumatoria de cada restricción de igualdad h_i multiplicada por una variable λ_i denominada multiplicador de Lagrange. Este método se puede generalizar a problemas con restricciones de desigualdad si estas se transforman en restricciones de igualdad agregando variables de holgura no negativas y_j. Luego, la fórmula general de la ecuación Lagrangiana está definida por la siguiente expresión:

$$\mathcal{L}(\vec{x}, \vec{y}, \vec{\lambda}) = f(\vec{x}) + \sum_{1}^{m} \lambda_j h_j(\vec{x}, \vec{y}) \tag{3.20}$$

Donde $\vec{x}$ es el vector de variables de diseño, $\vec{y}$ el vector de variables de holgura y $\vec{\lambda}$es el vector de los multiplicadores de Lagrange. Luego los puntos estacionarios de $\mathcal{L}(\vec{x}, \vec{y}, \vec{\lambda})$ se pueden encontrar aplicando las condiciones necesarias:

$$\frac{\partial \mathcal{L}}{\partial x_i}(\vec{x}, \vec{y}, \vec{\lambda}) = \frac{\partial f}{x_i}(\vec{x}) + \sum_{1}^{m} \lambda_j \frac{\partial h_j}{\partial x_i}(\vec{x}) = 0 \tag{3.21}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda_i}(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda}) = h_j(\mathbf{x}) + y_j^2 = 0 \tag{3.22}$$

$$\frac{\partial \mathcal{L}}{\partial y_i}(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda}) = 2\lambda_j y_j = 0 \tag{3.23}$$

Dado que $\vec{x}$, $\vec{y}$ y $\vec{\lambda}$ son vectores las ecuaciones anteriores representan un sistema de $n+2m$ ecuaciones con $n+2m$ incógnitas. La solución a este sistema proporciona el vector de soluciones óptimo, $\vec{x}^*$; el vector de multiplicadores de Lagrange, $\vec{\lambda}^*$; y el vector variable de holgura, $\vec{y}^*$. El significado físico de los multiplicadores de Lagrange es que $\vec{\lambda}^*$ denota la sensibilidad (o tasa de cambio) de f, denotando qué tan estrechamente la restricción se vincula con el punto óptimo [29].

El procedimiento continua analizando las soluciones encontradas para el sistema de ecuaciones ya que sólo se tienen puntos estacionarios hasta el momento. Para caracterizar las posibles soluciones obtenidas se utiliza la matriz Hessiana ampliada de la función Lagrangiana se define como la matriz de Hessianna de función lagrangiana "bordeada" con la matriz $(n\,x\,m)$ jacobiana de las restricciones de igualdad [44]:

$$A = \begin{pmatrix} 0 & Jh(\vec{x}^*) \\ Jh(\vec{x}^*)^T & H\mathcal{L}(\vec{x}^*) \end{pmatrix} = \begin{bmatrix} 0 & \dots & 0 & \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \frac{\partial h_m}{\partial x_1} & \cdots & \frac{\partial h_m}{\partial x_n} \\ \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_m}{\partial x_1} & \frac{\partial^2 \mathcal{L}}{\partial^2 x_1} & \cdots & \frac{\partial^2 \mathcal{L}}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_1}{\partial x_m} & \cdots & \frac{\partial h_m}{\partial x_n} & \frac{\partial^2 \mathcal{L}}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 \mathcal{L}}{\partial^2 x_n} \end{bmatrix} \tag{3.24}$$

Si el determinante de la matriz Hessiana ampliada es negativo y definido significa que el punto $\vec{x}^*$ es un mínimo. Si por el contrario, es positivo definido el punto $\vec{x}^*$ es un máximo.

3.5.2. Condiciones de Optimalidad en problemas con restricciones

Análogamente a los problemas sin restricciones, existen condiciones de optimalidad para problemas restringidos. Estas condiciones también se conocen como las condiciones de Kuhn-Tucker por los matemáticos Harold W. Kuhn, y Albert W. Tucker quienes las derivaron. En problemas restringidos es evidente que el gradiente no necesita ser igual a cero en el punto óptimo. Si el óptimo se encuentra en el interior, donde todas las restricciones están inactivas, entonces la condición de gradiente cero es verdadera. Sin embargo, cuando el punto estacionario está excluido de la región factible por alguna restricción, estas condiciones de optimalidad dejan de ser válidas.

Las condiciones de optimalidad para problemas con restricciones se puede dividir en dos procedimientos fundamentales. Primero, solo se considerarán las restricciones activas a través del método de los multiplicadores de Lagrange. Luego se aplica la condición denominada calificación de restricción. Dado un punto $\vec{x}^*$ que representa una solución óptima local las condiciones de Kuhn-Tucker son representadas por las siguientes expresiones [34]:

$$\nabla \mathcal{L}(\vec{x}, \vec{\lambda}) = \nabla f(\vec{x}) + \sum_{1}^{m} \lambda_j \nabla h_j(\vec{x}) + \sum_{1}^{m} \mu_j \nabla g_i(\vec{x}) \tag{3.25}$$

Donde si una restricción $g(\vec{x})_j \leq 0$ está inactiva $\mu_j = 0$. Luego la Ecuación 3.25 puede ser expresa como:

$$-\nabla f(\vec{x}) = \sum_{1}^{m} \lambda_j \nabla h_j(\vec{x}) + \sum_{1}^{p} \mu_j \nabla g_i(\vec{x}) \tag{3.26}$$

Esta ecuación significa que el negativo del gradiente de la función objetivo se puede expresar como una combinación lineal del gradiente de las restricciones. Para cualquier punto factible $\vec{x}^*$, el conjunto de restricciones de desigualdad activas se denota por $A(\vec{x}^*) = \{i | g_i(\vec{x}^*) = 0\}$. Estas condiciones son válidas si $\vec{x}^*$ es un *punto regular*. Un punto es regular si el gradiente de todas las restricciones de desigualdad activas y todas las restricciones de igualdad son linealmente independientes. Este requisito se denomina *calificación de restricción*.

Si la calificación de la restricción se viola en el punto $\vec{x}^*$, la Ecuación 3.26 puede o no tener una solución. La calificación de la restricción es difícil verificar sin conocer x^* de antemano. Sin embargo, la calificación de la restricción siempre se satisface para los problemas que tengan cualquiera de las siguientes características [29]:

1. Todas las funciones de restricción de desigualdad e igualdad son lineales.
2. Todas las funciones de restricción de desigualdad son convexas, todas las funciones de restricción de igualdad son lineales, y existe al menos un vector factible $\vec{x}$ que se encuentra estrictamente dentro de la región factible, de modo que $g_j(\vec{x}) < 0$, con $j = \{1, 2, \ldots, m\}$ y $h_i(\vec{x}) = 0$ con $i \{= 1, 2, ..., p\}$

Es importante tener en cuenta que las condiciones Kuhn-Tucker son necesarias pero no suficientes para la optimalidad. Es decir, puede haber otros óptimos locales en los que se satisfagan las condiciones de Kuhn-Tucker.

3.5.3. Método de Penalización de Restricciones

Como su nombre indica, este método penaliza la función objetivo en caso de que se violen las restricciones. La motivación del método de función de penalización es resolver el problema de optimización restringida utilizando algoritmos para problemas no restringidos, donde en cada iteración se modifica el parámetro de penalización R. La función objetivo modificada con términos de penalización se escribe como:

$$P(\vec{x}, R) = f(x) + \Omega(R, g(\vec{x}), h(\vec{x})) \tag{3.27}$$

Donde Ω es una función llamada *término de penalización*. Estas funciones pueden ser clasificadas en términos de penalización exteriores e interiores. Los términos de penalización exteriores son aquellos que penalizan solo los puntos que se encuentran fuera de la región factible. En cambio, los términos de penalización interior penalizan aquellos puntos factibles que se acercan a la activación de la restricción o aquellos que la activan. A continuación se presentan algunos de los términos más utilizados [43]:

1. **Penalización parabólica**: Se aplica sólo a restricciones de igualdad. Es un término de penalización exterior ya que todos los puntos no factibles son penalizados. El valor de R aumenta gradualmente en cada iteración (Ecuación 3.28).

$$\Omega = R\{h(\mathbf{x})\}^2 \tag{3.28}$$

2. **Penalización de barrera infinita**: Se utiliza para controlar restricciones de desigualdad $g(\vec{x})$. Aquí se aplica una penalización proporcional a la suma del conjunto J de las restricciones violadas. Es un término de penalización exterior (Ecuación 3.29).

$$\Omega = R \sum_{j \in J} g_j(\vec{x}) \tag{3.29}$$

3. **Penalización logarítmica**: Se aplica para controlar restricciones de desigualdad. Este término solo se aplica a puntos factibles. Mientras más cercanos esté un punto de violar la restricción mayor el valor del término. Por tanto, es un término de penalización interior (Ecuación 3.30).

$$\Omega = -R \ln g_j(\vec{x}) \tag{3.30}$$

4. **Penalización inversa**: Se utiliza en restricciones de desigualdad. Se comienza con un valor alto de R y se reduce gradualmente. Este es un término de penalización interior (Ecuación 3.31).

$$\Omega = -R \frac{1}{g_j(\vec{x})} \tag{3.31}$$

5. **Penalización con operador de paréntesis**: Se asignan valores positivos a puntos no factibles y cero a los factibles. Se comienza con un valor pequeño de R que se aumenta gradualmente. Es un término de penalización exterior (Ecuación 3.32).

$$\Omega = R\langle h(\vec{x})\rangle^2 \tag{3.32}$$

Con este mecanismo para el manejo de restricciones el minimizador puede iniciar desde un punto no factible. Además, esta técnica puede ser integrada directamente en métodos de optimización sin restricciones como los descritos anteriormente. No obstante es importante considerar que la función objetivo se transforma a medida que aumenta el valor de los términos de penalización. Debido a cambios abruptos en el valor de la función, el valor del gradiente puede ser grande y el algoritmo puede mostrar divergencia. Dado que este método no satisface exactamente las restricciones, no es adecuado para problemas de optimización donde la viabilidad debe garantizarse en todas las iteraciones [34].

3.5.4. Programación Cuadrática Secuencial

El método de Programación Cuadrática Secuencial (SQP por sus siglas en inglés) es uno de los métodos de Programación Matemática más recientes y efectivos sobre todo en problemas convexos. Su base teórica se relaciona primeramente con la solución de un conjunto de ecuaciones no lineales utilizando el método de Newton y luego con la derivación de ecuaciones simultaneas utilizando las condiciones de Kunn-Tucker. El procedimiento de este método consiste en aproximar la función objetivo a una forma cuadrática y trasforma a lineales las restricciones en cada iteración. El problema de programación cuadrática se resuelve para obtener $\Delta\vec{x}$:

$$Q = \Delta\vec{x}^T \nabla f(x) + \frac{1}{2}\Delta\vec{x}^T \nabla^2 \mathcal{L} \Delta\vec{x} \tag{3.33}$$

sujeto a

$$h_j(\vec{x}) + \nabla h_j(\vec{x})^T \Delta\vec{x} = 0 \tag{3.34}$$

$$g_i(\vec{x}) + \nabla g_i(\vec{x})^T \Delta\vec{x} = 0 \tag{3.35}$$

El valor de $\vec{x}$ se actualiza con $\Delta\vec{x}$ haciendo $\vec{x} = \vec{x} + \Delta\vec{x}$. De nuevo, la función objetivo se aproxima con una función cuadrática y las restricciones se hacen lineales con el nuevo valor de x. Las iteraciones se repiten hasta que no se puedan realizar mejoras al valor de la función objetivo.

3.6. Conclusiones del capítulo

El presente capítulo describe el panorama general de las principales técnicas de Programación Matemática. Primeramente se describen las condiciones necesarias para la optimalidad (máxima o mínima) que plantean que el gradiente desaparece en los puntos estacionarios de la función. Luego, si la segunda derivada de la función objetivo evaluada en un punto estacionario es positiva, se identifica a un mínimo y si la segunda derivada es negativa, es el caso de un máximo. El gradiente proporciona información sobre la velocidad de cambio instantánea de una función en una dirección en particular. La matriz Hessiana $H(\vec{x})$ representa la segunda derivada de una función multivariable. Las condiciones de segundo orden para funciones multivariables plantean que la Hessiana debe ser definida positiva en el punto mínimo de la función. Esto es, que su determinante es positivo.

Se analizaron los métodos de optimización para funciones de una sola variable en problemas sin restricciones, los cuales son utilizados por los métodos de optimización multivariable. Estos últimos pueden ser divididos en métodos basados en gradiente y métodos directos. Los métodos basados en gradiente como el de Cauchy y Newton asumen que la función objetivo es derivable. Incluso cuando esta condición se cumple estos métodos presentan dificultades desde el punto de vista de complejidad computacional debido al costo de operaciones matriciales. Por ejemplo, aunque el método de Newton es conocido por converger en una iteración para una función cuadrática, el cálculo de la matriz de Hessiana es computacionalmente costoso. Si la función no es cuadrática el método resulta ineficiente. Se debe destacar además, que estos métodos son buscadores locales generalmente sensibles al punto de inicio. Por último las técnicas para el manejo de restricciones constituyen las herramientas finales para la resolución de problemas reales, ya que estas pueden ser incorporadas a los métodos de optimización sin restricciones.

En contraste con los métodos basados en gradiente, los algoritmos de búsqueda directa como el Nelder-Mead y Jookes-Jeeves utilizan solo el valor de la función objetivo. El método de Nelder-Mead puede ser utilizado para solucionar problemas de optimización global al introducir cierta aleatoriedad [32] y realiza pocas evaluaciones de la función objetivo por iteración. Por otra parte, el método Hookes-Jeeves es un eficiente buscador local, aunque realiza más evaluaciones de $f(x)$ por iteración. Estos dos métodos directos serán los candidatos iniciales a utilizar en la propuesta solución debido a que tanto sus operadores como su forma de evaluar una solución se alienan con las técnicas evolutivas en mayor grado que otras métodos de programación matemática. Además, como se describe en el Capítulo 5, algunos de los problemas a resolver, presentan funciones objetivo que no están definidas en los reales para todo su dominio. Esta característica, hace que los métodos basados en gradientes no puedan a ser aplicados.

CAPÍTULO 4

Computación Evolutiva

La computación evolutiva es una subárea dentro de las técnicas bio-inspiradas, que abarca los métodos de resolución de problemas basados en aspectos fundamentales de la teoría de la evolución de Charles Darwin. Específicamente, los componentes de un algoritmo evolutivo, se basan el concepto de sistema evolutivo darwiniano los cuales presentan, entre otras, las siguientes características [45]:

1. El sistema está conformado por una o más poblaciones de individuos que compiten por recursos limitados.
2. Las poblaciones son dinámicamente cambiantes debido al nacimiento y muerte de individuos.
3. La aptitud es una métrica que refleja la capacidad de un individuo para sobrevivir y reproducirse.
4. La herencia variacional: define que los descendientes se parecen a sus padres, pero no son idénticos.

Estas características permiten entender un sistema evolutivo como un proceso que dado un estado inicial (las condiciones iniciales particulares), realiza una trayectoria a través de un complejo espacio de estados que evolucionan en el tiempo. Este proceso puede ser utilizado como base teórica para el diseño de algoritmos evolutivos. Los modelos de algoritmos evolutivos más simples se centran en la evolución a lo largo del tiempo de una única población de individuos de tamaño fijo que es modificada a través de mecanismos de reproducción y herencia.

Dependiendo del problema y la estrategia, varios componentes del algoritmo pueden ser fijos o estar sujetos a presiones evolutivas. Algunos de ellos, como el tamaño inicial de la población, el número de puntos de cruce, las probabilidades para la mutación o el cruce pueden ser ajustados de forma adaptativa [46]. Desde el punto de vista computacional, los algoritmos evolutivos se pueden estudiar en diversos aspectos, como sus propiedades de convergencia, su sensibilidad a las condiciones iniciales, o su comportamiento transitorio.

A pesar de que la teoría de la evolución se había concebido como metáfora para procesos computacionales durante la primera mitad del siglo XX, su consolidación sucede durante la década de 1960 debido a la creciente disponibilidad de computadoras digitales de bajo costo para su uso como una herramienta de simulación y modelado por parte de la comunidad científica. Varios grupos científicos comenzaron aplicar los modelos evolutivos simples, los cuales podían expresarse computacionalmente, para resolver problemas informáticos complejos como la planificación de rutas, programación de listas y optimización de diseños [47] [48].

De forma general, los algoritmos evolutivos exploran simultáneamente el espacio de búsqueda con una población de soluciones candidatas llamadas *individuos*, a los cuales se les asocia un valor para medir su calidad. Este valor es denominado valor de aptitud (fitness en inglés), y se obtiene teniendo cuenta función objetivo evaluada en el individuo y otras características del medio ambiente. A continuación, se seleccionan entre la población aquellas soluciones con mayor aptitud, dando lugar a la siguiente generación que consiste en réplicas de las soluciones más adecuadas que han sido genéticamente mutadas y cruzadas. Estos mecanismos perturban las variables de decisión de manera que, o bien heredarán algunas características de sus padres, o cambian de manera aleatoria.

A pesar de su simplicidad, los algoritmos evolutivos pueden producir un rango amplio de soluciones durante todo el proceso evolutivo como resultado de complejas interacciones de las condiciones iniciales, las elecciones particulares hechas para los mecanismos de reproducción y herencia, y las características de la función de aptitud. Durante las últimas décadas, el éxito de los algoritmos evolutivos en la resolución de problemas reales de optimización los ha convertido en un método popular para resolver problemas de optimización complejos mono y multiobjetivo [49].

4.1. Representación

Basándose en la metáfora de la evolución natural, el diseño general de un algoritmo evolutivo puede dividirse en dos capas de abstracción [50]. En la capa superior se representa el contexto del problema a resolver. Una posible solución $x \in S$, es tratada como un *fenotipo*, su calidad como individuo o aptitud para la

supervivencia se evalúa mediante la función objetivo o una función de aptitud determinada, sujetas a optimización para dirigir la evolución de los individuos hacia un objetivo específico. En este nivel se aplican mecanismos de selección como los descritos en la sección 4.3. Por otra parte, en el nivel inferior, los fenotipos son representados mediante genotipos, al igual que en la naturaleza, cada ser vivo es una instancia de su genoma $g \in G$, donde G es el espacio generado por todas combinaciones de los valores posibles en las variables de diseño. Los genotipos son son estructuras de datos que pueden manipularse para producir variaciones en las posibles soluciones. En este nivel se aplican los mecanismos de reproducción (Véase sección 4.4). La relación genotipo-fenotipo gpm enlaza las dos capas de abstracción y puede plantearse como $x = gpm(g)$ [48]. Para la representación de genotipos las siguientes estructuras de datos pueden ser utilizadas:

1. **Representación binaria**: Es la codificación natural para representar genotipos. Esta representación discretiza el espacio de búsqueda de fenotipos continuos, donde el genotipo es una cadena binaria de longitud l que genera 2^l puntos en todo el espacio de búsqueda. Idealmente, todos estos puntos deberían estar dispersos de manera uniforme para evitar cualquier sesgo hacia una región en particular del espacio de búsqueda [51].

2. **Representación entera**: Consiste en representar a los individuos como arreglos de números enteros. Este enfoque resulta efectivo en problemas de optimización combinatoria como el problema del agente viajero; donde una solución al problema se plantea como un vector de enteros $\vec{x} = (i_1, i_2, ..., i_n)$ que representa un recorrido desde i_1 hasta i_2, y así sucesivamente terminando con i_{n-1} hasta n y de regreso a i_1. Luego, los operadores cruzamiento y mutación resultan sencillos de definir en comparación con una representación binaria para este problema. La evaluación de los individuos consiste en sumar los costos de viajar de la cuidad i_1,i_2,...,i_n [52].

3. **Representación real**: Se representan los individuos como vectores de valor real. Esta representación es utilizada para problemas de optimización continua donde las variables de decisión son dimensiones, masas o potencias [43].

4. **Representación híbrida**: El genotipo de los individuos se conforma mediante una representación binaria, una representación real, y una bandera que indican la representación se va a usar en cada generación. Se deben realizar procedimientos de sincronización entre ambas representaciones para que luego de aplicar los operadores correspondientes, ambas mantengan el mismo valor [53].

5. **Representación mediante árboles**: Esta representación se utiliza para manipular programas de computadora como datos, que luego puedan compilar, enlazar y ejecutar otros programas o apoyar a un intérprete para ejecutar los

nuevos programas. Se aplica en la programación de compiladores para manipular el árbol de análisis sintáctico de un programa y en la compilación de expresiones [54]

4.2. Función de aptitud

La función de aptitud (*fitness* en inglés) no solo refleja la calidad o jerarquía de un individuo con respecto a otros en la población, sino que también puede incorporar información de densidad hacía una región determinada del espacio de búsqueda. De esta manera, no sólo se considera qué tan buena es una solución candidata, sino también la diversidad general de la población. Esto puede mejorar la posibilidad de encontrar el óptimo global así como el rendimiento del algoritmo de optimización de manera significativa. La aptitud no solo puede depender de la solución candidata en sí misma, sino de toda la población del algoritmo evolutivo y en caso de que se conozca de las soluciones óptimas [48].

4.3. Selección

En el proceso de selección se elige cierto número de individuos en la población de acuerdo a su valor de aptitud. Al igual que en la naturaleza, aquellos individuos con mayor capacidad para adaptarse al medio y sobrevivir, prevalecerán para reproducirse. A mayor valor de aptitud un individuo tendrá más probabilidades de reproducirse y de pasar sus genes a la próxima generación.

Los mecanismos de selección pueden ser agrupados en dos categorías: mecanismos de selección con reemplazo y sin reemplazo [48]. En un mecanismo de selección sin reemplazo, cada individuo de la población se toma en consideración para la reproducción a lo sumo una vez y por lo tanto también aparecerá en conjunto de individuos seleccionados para la reproducción una vez como máximo. Por otra parte, los mecanismos con reemplazo pueden generar conjunto padres en los que se repitan individuos. Esto se debe al hecho natural de que un individuo puede tener múltiples descendientes.

Los mecanismos de selección tienen un gran impacto en el rendimiento de los algoritmos evolutivos, por lo que su comportamiento ha sido objeto de varios estudios, destacándose los trabajos Goldberg y Deb [55], Sarma y De Jong [56] , y Zhong [57]. En [58] se propone un mecanismo para el manejo de restricciones utilizando reglas lógicas. Estas realizan la selección basándose primeramente en la suma de violaciones de restricciones y luego en valor de la función objetivo. En la literatura especializada se destacan los siguientes mecanismos de selección:

- **Selección determinista**: También llamada selección de umbral, devuelve los mejores k individuos que participarán en la cruza. Estos elementos se copian tantas veces como sea necesario hasta que se alcanza cantidad de individuos deseada en el grupo padres. El procedimiento de forma general consiste en ordenar ascendentemente la población de acuerdo a su valor de aptitud y seleccionar los primeros k individuos de la población ordenada. Como pueden existir padres repetidos, es común que los algoritmos evolutivos sean combinados con un mecanismo de asignación de aptitud que incorpore información de diversidad de la población para evitar la convergencia prematura [59].
- **Selección proporcional al valor de aptitud**: Es el mecanismo aplicado en los algoritmos genéticos originales introducidos por Holland en [47], donde la probabilidad $P(x_i)$ del individuo $x_i \in X$ de ser seleccionado para la reproducción es proporcional a su aptitud $fitness(x_i)$ cuyo valor es sujeto a maximización. La proporción se establece entre el valor de aptitud del individuo y la sumatoria de las aptitudes del resto de los individuos en la población. Esta relación en su forma original se define en la ecuación a continuación:

$$P(x_i) = \frac{fitness(x_i)}{\sum_{j \neq i} fitness(x_j)} \tag{4.1}$$

  Existe una amplia variedad de estrategias de selección que realizan distribuciones de probabilidad como la Selección del Resto Estocástico y Selección Estocástica Universal. Un método ampliamente utilizado es la selección de la Ruleta de Monte Carlo por De Jong [60], donde de manera imaginaria los individuos son posicionados en una rueda de ruleta. Los individuos con mayor valor de aptitud ocupan mayor área por lo que será más probable que la ruleta se detenga sobre ellos. Este procedimiento se repite hasta que se hayan seleccionado los k individuos padres.
- **Selección por torneo**: La selección mediante torneos, es uno de los esquemas de selección más utilizados y efectivos. En la selección por torneo, k elementos se seleccionan de la población mediante la realización de comparaciones entre sí en un torneo. El ganador de cada competencia es seleccionado para la reproducción. El individuo con mayor valor de aptitud en la población ganará todos los concursos en los que participe y, por lo tanto se repetirá más en el conjunto de padres. El peor individuo en la población perderá todos sus desafíos frente a otros candidatos [48].

4.4. Reproducción

El siguiente paso en la estructura general de un algoritmo evolutivo es la reproducción. Los operadores de reproducción utilizan la información obtenida en la generación actual para generar nuevas soluciones candidatas que se evaluarán en la siguiente generación. A continuación se describen los operadores básicos de reproducción [48]:

1. **Creación**: Consiste en crear un nuevo genotipo sin antepasados o herencia. Por lo tanto, se puede comparar con la aparición de las primeras células vivas a partir de la mezcla de ciertos elementos químicos. La operación de creación se usa para generar nuevos genotipos con una configuración aleatoria ya que en la generación cero, aún no se ha obtenido información sobre el espacio de búsqueda.
2. **Duplicación**: Se asemeja a la división celular, lo que resulta en dos individuos similares a uno de los padres. La operación de duplicación se usa para crear una copia exacta de un genotipo existente.
3. **Cruza**: Al igual que en la reproducción sexual, la cruza o recombinación combina dos genotipos parentales dando lugar a un nuevo genotipo que incluye rasgos de ambos ancestros.
4. **Mutación**: Realiza variaciones pequeñas y aleatorias en el genotipo de un individuo, al igual que su contraparte natural. La cantidad de variación puede ser controlada especificando cuántos genes se van a modificar y la manera en que se modificarán.

4.5. Paradigmas de la Computación Evolutiva

Desde su surgimiento la computación evolutiva ha sido un área de investigación fértil, dando lugar a una gran cantidad de modelos evolutivos para la resolución de problemas. No obstante, existen tres paradigmas principales que pueden ser analizados independientemente pues se distinguen desde el punto de vista conceptual y de diseño.

4.5.1. Programación Evolutiva

Las bases de la Programación Evolutiva (PE) fueron propuestos Fogel en su tesis doctoral en 1964, donde fueron utilizadas para solucionar problemas de control [61]. En general, es difícil distinguir la Programación Evolutiva de los Algoritmos

Genéticos y las Estrategias Evolutivas. Sin embargo existen ciertas diferencias que distinguen la programación evolutiva del resto de los paradigmas.

Según [48], a pesar de que no existe una especificación clara o variante algorítmica para la programación evolutiva; se puede identificar una diferencia semántica con respecto a las demás clases de algoritmos evolutivos. Mientras que la mayoría de los algoritmos evolutivos, los individuos de una especie son la metáfora biológica de las soluciones candidatas, en la programación evolutiva, se considera que una solución candidata es una especie. Por lo tanto, la mutación y la selección son los únicos operadores utilizado en la PE y la recombinación generalmente no se aplica. El esquema de selección utilizado en la PE es normalmente bastante similar al método clásico. En determinados aspectos, el enfoque de la programación evolutiva se ha fusionado con estas otras áreas de investigación.

4.5.2. Estrategias evolutivas

Las Estrategias Evolutivas fueron propuestas por Rechenberg en [62] como una técnica de optimización experimental basadas en los conceptos de adaptación y evolución. Este tipo de algoritmo evolutivo presenta las siguientes características que las diferencian de los demás paradigmas evolutivos [63] [48]:

1. Generalmente utilizan vectores reales de tamaño fijo como representación de soluciones candidatas.
2. La mutación y selección son los operadores principales.
3. La mutación modifica los componentes de un vector x_i utilizando números aleatorios de una distribución normal.
4. Generalmente utiliza selección de truncamiento con una estrategia de población variable $(\mu + \lambda)$ donde μ es el número de padres y λ es el número de hijos.
5. La regla de $\frac{1}{5}$ definida por Rechenberg plantea que el cociente de dividir el número de mutaciones exitosas entre el número total de mutaciones debe ser de aproximadamente igual a $\frac{1}{5}$.

4.5.3. Algoritmos Genéticos

Los algoritmos genéticos fueron reconocidos formalmente como un nuevo enfoque para la resolución de problemas gracias a los trabajos de Holland [64] [65] [47] en la década de 1960. Los algoritmos genéticos (AG) son una subclase de algoritmos evolutivos que se distingue por el uso de representación binaria para los genotipos. En un algoritmo genético los elementos del espacio de búsqueda son

cadenas binarias o matrices de otros tipos elementales. Los genotipos se usan en las operaciones de reproducción mientras que los valores de la función objetivo se calculan con base en los fenotipos que se obtienen a través del mapeo genotipo-fenotipo [48].

Los AG también implementan la noción biológica de la aptitud de una forma diferente a las estrategias vistas anteriormente. Los algoritmos basados en paradigmas como Estrategias de Evolución o Programación Evolutiva usan la aptitud para determinar qué descendientes sobreviven hasta la edad adulta (es decir, cuáles son los que llegarán a ser padres). Luego, los individuos que conformar el conjunto de padres tienen las mismas oportunidades de reproducirse. Ocasionalmente, esto podría permitir que ciertos individuos sobrevivan hasta la edad adulta y luego pasen intactos a la nueva generación. Sin embargo, tales sistemas no son biológicamente probables ni computacionalmente deseables ya que permitir que los individuos sobrevivan y se reproduzcan indefinidamente puede resultar en una pérdida significativa de la diversidad en la población y puede aumentar la probabilidad de que el algoritmo quede atrapado en un mínimo local. Los AG atacan este problema de varias formas. Una estrategia válida es permitir (con poca frecuencia) que nuevos individuos reemplacen a las individuos en la próxima generación a pesar de que estos últimos presente un mejor valor de aptitud. Otra forma de atacar este problema es mediante un modelo generacional en el cual los padres sobreviven exactamente una generación y son completamente reemplazados por su descendencia [45].

4.6. Evolución Diferencial

Desarrollada por Storn y Price en 1996, la Evolución Diferencial (ED) se ha convertido en uno de los algoritmos más eficientes en la resolución de problemas no lineales de optimización numérica, como es el caso de los problemas de optimización de diseño mecatrónico abordados en el presente trabajo de tesis.

En el artículo *."Evolución diferencial- un esquema de adaptación simple y eficiente para la optimización global en espacios continuos"* Storn y Price definen la ED como método de búsqueda directa paralela que utiliza un conjunto de NP vectores de D dimensiones $x_{i,G}$ con $i = \{1; 2; ...; NP\}$ como población para cada generación G. Donde NP es el tamaño de la población el cual se mantiene constante durante el proceso de minimización. La población inicial de vectores se elige de forma aleatoria siguiendo una distribución de probabilidad uniforme.

Una de las características más significativas de la ED es su operador de mutación definido en la ecuación 4.2. El vector mutado o vector ruido $v_{i,G+1}$ se genera a partir tres vectores seleccionados aleatoriamente, tal que los indices aleatorios cumplan que $i \neq r_1 \neq r_2 \neq r_3$. Luego $v_{i,G+1}$ se calcula sumando la diferencia

ponderada entre dos vectores de los vectores a un tercer vector.

$$v_{i,G+1} = x_{r_1,G} + F(x_{r_2,G} - x_{r_3,G}) \tag{4.2}$$

Donde $F > 0$ es una constante real qué determina que tan amplia será la variación diferencial. Una vez obtenido el vector ruido se procede a realizar la cruza. Como se describe en la ecuación 4.3, esta operación consiste recorrer cada variable j generando un número aleatorio $randb(j)$ si este es menor que la probabilidad de cruza CR la variable $u_{ji,G+1}$ tomará el valor de $v_{ji,G+1}$.

$$u_{ij,G+1} = \begin{cases} v_{ij,G+1}, & \text{si}(randb(j) \leq CR) \text{ o } j = rnbr(i) \\ x_{ij,G}, & \text{en otro caso} \end{cases} \tag{4.3}$$

Nótese que en la ecuación anterior $rnbr(i)$ es número entero aleatorio entre 1 y D que garantiza que $u_{i,G+1}$ herede al menos uno de los valores de $v_{i,G+1}$. En el artículo original el mecanismo selección se basa en un criterio voraz. El vector $u_{i,G+1}$ sustituye al padre $x_{ji,G}$ si tiene un menor valor de función objetivo, asumiendo minimización; en caso contrario, $x_{ji,G}$ se conserva para la próxima generación. Gracias a la flexibilidad de este algoritmo otros mecanismos de selección pueden ser utilizados con éxito.

En este sentido se destaca el trabajo de Mezura en [66] donde se incorpora un mecanismo simple para el manejo de restricciones que prioriza las soluciones factibles o aquellas que violan menor cantidad de restricciones. Este enfoque de manejo de restricciones no agrega ningún parámetro adicional definido por el usuario al algoritmo de Evolución Diferencial, obteniéndose además resultados competitivos con respecto a tres técnicas representativas del estado del arte en problemas de optimización con restricciones.

Como se describió en el capitulo introductorio las variantes de la ED han obtenido los resultados más competitivos en la resolución de los problemas de optimización de Diseño Mecatrónico a resolver. Esto, sumado a su flexibilidad, simplicidad de su procedimiento y su eficiencia como buscador global, hacen de la ED el candidato principal para la realización de la hibridación. El pseudo-código de la ED se describe en el Algoritmo 3, donde G_{MAX} es la cantidad de generaciones a obtener, y NP es el tamaño de la población.

Algoritmo 3 Evolución Diferencial

Generar aleatoriamente una población inicial $x_{i,G}$; $\forall i, i = 1, 2, ..., NP$.
Evaluar la población inicial $f(x_{i,G})$; $\forall i, i = 1, 2, ..., NP$.
for $G \in \{1, \ldots, G_{MAX}\}$ **do**
  for $i \in \{1, \ldots, NP\}$ **do**
    Seleccionar aleaoriamente r_1, r_2,r_3, tal que $r_1 \neq r_2 \neq r_3 \neq i$
    Generar aleatoriamente el índice $randb(j) = (1, D)$
    for $j \in \{1, \ldots, D\}$ **do**
      if $(randb(j) \leq CR)$ **or** $(j == randb(j))$ **then**
        $u_{ij,G+1} = x_{r_1 j,G} + F(x_{r_2 j,G} - x_{r_3 j,G})$
      else
        $u_{ij,G+1} = x_{ij,G}$
      end if
    end for
    if $f(u_{i,G+1}) \leq f(x_{i,G})$ **then**
      $x_{i,G+1} = u_{i,G+1}$
    else
      $x_{i,G+1} = x_{i,G}$
    end if
  end for
end for

4.7. Conclusiones del capítulo

Los algoritmos evolutivos son un método eficiente para resolver problemas de optimización que presenten espacios de búsqueda complejos, y pueden ser considerados algoritmos de optimización global. En las últimas décadas se ha desarrollado una gran diversidad de variantes de algoritmos evolutivos con aplicaciones en ramas como la optimización numérica y combinatoria [67] [68], la robótica [69] [70], el diseño ingenieril [71] [72], entre otros. Además de la simplicidad y flexibilidad ante la incorporación de nuevos operadores o mecanismos para el manejo de restricciones; la ventaja principal de los Algoritmos Evolutivos en comparación con otros métodos de optimización es que requieren una comprensión menor de la naturaleza del espacio búsqueda (no necesitan información de la derivada) para la construcción de una heurística eficaz. Por lo tanto, los Algoritmos Evolutivos pueden ser aplicados a un diferentes categorías de problemas de optimización obteniendo resultados positivos.

CAPÍTULO 5

Problemas de Optimización de Diseño de Mecatrónico

En el capítulo introductorio del presente trabajo se abordaron algunos conceptos fundamentales sobre el Diseño Mecatrónico:

1. El modelado de un sistema mecatrónico tiene como objetivo representar el comportamiento de un sistema real mediante un conjunto de ecuaciones matemáticas.
2. Los sistemas reales son sistemas físicos, cuyo comportamiento se basa en la materia y la energía.
3. Los modelos utilizados en el diseño mecatrónico representan estructuras de causa y efecto, aceptan información externa y la procesan con su lógica y ecuaciones para producir una o más salidas.
4. El objetivo de la optimización es, dado un modelo de un sistema determinado físico, encontrar una configuración óptima para este sistema.

Estos conceptos son la base para el desarrollo de la siguientes secciones. En adelante se describen primeramente tres casos de estudio de la "Síntesis Óptima de un Mecanismo de Cuatro Barras", luego se presentan dos casos de la "Síntesis Óptima de un Efector Final de Tres Dedos". Por último, se presenta el problema de "Optimización del costo de generación de energía en una Micro-red Eléctrica no Interconectable". Estos problemas han sido formulados y resueltos por investigadores del departamento de mecatrónica del Centro de Innovación y Desarrollo

Tecnológico en Cómputo (CIDETEC) del Instituto Politécnico Nacional (IPN), lo que brinda un marco de referencia tanto desde el punto de vista del problema (conocimiento de óptimos) como en el conocimiento del desempeño de los algoritmos utilizados (número de evaluaciones, rapidez de convergencia, resultados de pruebas estadísticas).

5.1. Diseño cinemático de Mecanismos

Las máquinas son dispositivos utilizados para alterar, transmitir y dirigir las fuerzas para lograr un objetivo específico. Un mecanismo es la parte mecánica de una máquina que tiene la función de transferir movimiento y fuerzas desde una fuente de alimentación a una salida. Los mecanismos se pueden considerar como piezas rígidas que están dispuestas y conectadas para que produzcan el movimiento deseado de la máquina. Los mecanismos de enlace son aquellos compuestos por diferentes eslabones (barras) conectados entre sí para formar una cadena. Los eslabones son las partes individuales del mecanismo y se consideran cuerpos rígidos que se encadenan para transmitir movimiento y fuerza. Teóricamente, un cuerpo rígido no cambia de forma durante el movimiento. A pesar de que no existe un verdadero cuerpo rígido, pues los componentes del mecanismo siempre están sujetos a ciertas deformaciones, en lo adelante se consideran estos componentes como cuerpos rígidos, atendiendo solo al análisis cinemático. En los mecanismos de enlace un eslabón se designa como marco (frame en inglés) de referencia para el movimiento de todas las otras partes. El marco de referencia es típicamente un eslabón que no realiza movimiento. Entre dos eslabones existe una conexión movible que permite el movimiento relativo entre estos, llamada *junta*. Las articulaciones básicas pueden ser clasificadas en giratorias y deslizantes. La articulación giratoria permite la rotación pura entre los dos enlaces que conecta. La junta deslizante también llamada pistón o junta prismática, permite el deslizamiento lineal entre los enlaces conectados. Además existen otros tipos de juntas como la junta de leva que permite tanto la rotación como el deslizamiento entre los dos enlaces que conecta. La junta de engranaje también permite la rotación y el deslizamiento entre dos engranajes. Ambos tipos de juntas son consideradas articulaciones de orden superior debido a que permiten movimientos más complejos [73].

La síntesis es el proceso de desarrollar un mecanismo para satisfacer un conjunto de requisitos de rendimiento para una máquina. De forma general, la cinemática estudia el movimiento de los cuerpos sólidos. Aplicada al diseño de mecanismos, se utiliza para analizar la geometría del movimiento, lo que implica el análisis de la posición, el desplazamiento, la rotación, la velocidad, y la aceleración de los componentes. Por tanto, el análisis cinemático asegura que el mecanismo mostrará un movimiento que cumplirá con determinado conjunto de requisitos. Para

lograr la síntesis óptima de un mecanismo es necesario recurrir a técnicas analíticas avanzadas que implican el modelado mediante funciones matemáticas. Las técnicas analíticas combinan las teorías de geometría, trigonometría y análisis de mecanismos gráficos. La síntesis óptima comprende la modelación de estos mecanismos matemáticamente, en forma de problemas de optimización numérica. Estos problemas pueden ser resueltos mediante algunas de las técnicas abordadas en capítulos anteriores [73].

5.1.1. Análisis cinemático de un Mecanismo de Cuatro Barras

El mecanismo de cuatro barras es una combinación de cuatro eslabones, que están conectado por cuatro juntas de giratorias. Este mecanismo tiene una gran cantidad de aplicaciones por lo que se encuentra frecuentemente en la literatura.

El esquema general del mecanismo de cuatro barras que se muestra en la Figura 5.1. Está formado por una barra de referencia r_1, una barra de entrada r_2 (manivela), un acoplador r_3 y una barra de salida r_4 (basculante). Para analizar este mecanismo se establecen dos sistemas de coordenadas: un sistema que se fija al mundo real (OXY) y otro para la auto-referencia [7].

Para el análisis de la posición, velocidad y aceleración se comienza por plantear las ecuaciones que describen el hecho de que las barras del mecanismo conforman un lazo cerrado:

$$\vec{r_1} + \vec{r_4} = \vec{r_2} + \vec{r_3} \tag{5.1}$$

En la Ecuación 5.1 las barras son consideras vectores. Como cualquier vector puede ser expresado como un número complejo utilizando notación polar, donde la parte real corresponde al eje de las abscisas y la parte imaginaria pertenece al eje de las coordenadas. Luego, la Ecuación 5.1 puede ser expresada en coordenadas polares de la siguiente forma:

$$r_1 e^{j\theta_1} + r_4 e^{j\theta_4} = r_2 e^{j\theta_2} + r_3 e^{j\theta_3} \tag{5.2}$$

Donde $e^{j\theta_1}$ es la entidad de Euler y θ_i es el ángulo comprendido entre cada barra y el eje de coordenadas. La identidad de Euler plantea la siguiente relación donde j es la unidad imaginaria [74]:

$$e^{jx} = \cos x + j \sin x \tag{5.3}$$

Luego sustituyendo la Ecuación 5.3 en 5.2 se obtiene la expresión:

$$r_1(\cos\theta_1 + j\sin\theta_1) + r_4(\cos\theta_4 + j\sin\theta_1) = r_2(\cos\theta_2 + j\sin\theta_2) + r_3(\cos\theta_3 + j\sin\theta_3) \tag{5.4}$$

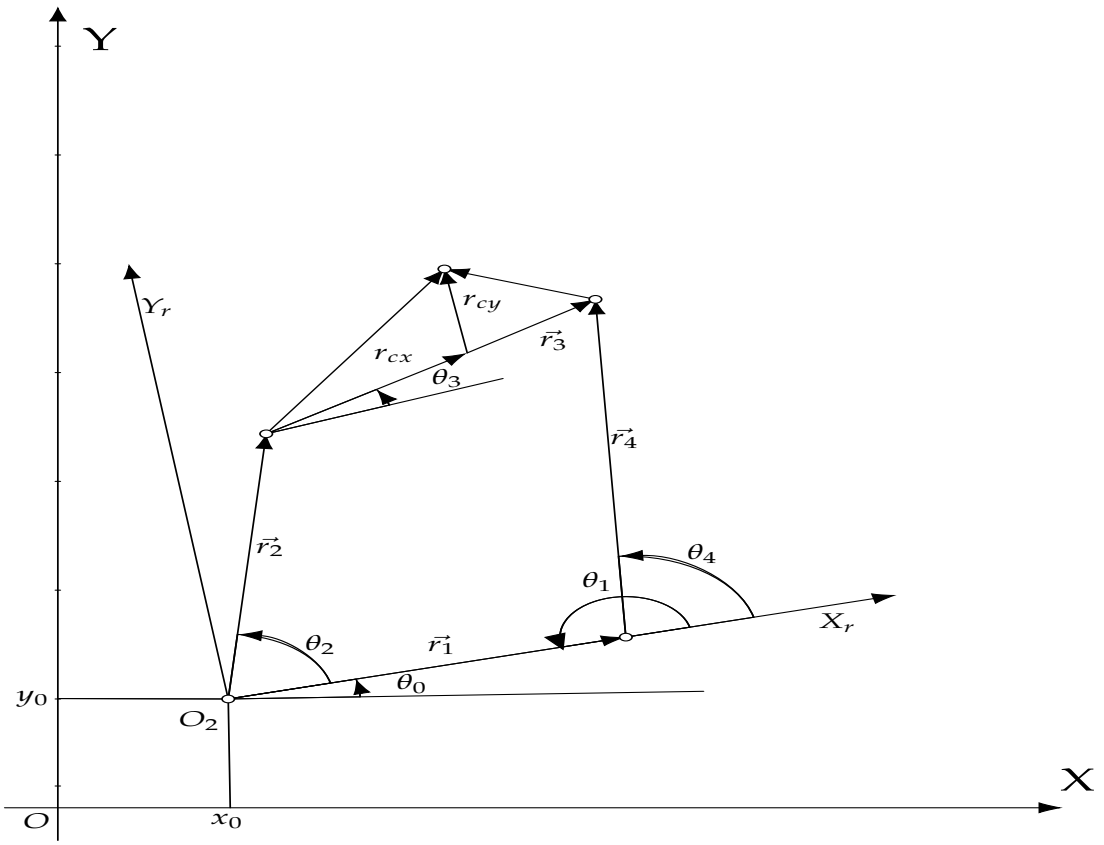


FIGURA 5.1: Mecanismo de cuatro barras.

Resolviendo:

$$r_1 \cos\theta_1 + r_1 j \sin\theta_1 + r_4 \cos\theta_4 + r_4 j \sin\theta_4 = r_2 \cos\theta_2 + r_2 j \sin\theta_2 + r_3 \cos\theta_3 + r_3 j \sin\theta_3 \tag{5.5}$$

Agrupando la real en el miembro izquierdo y la parte imaginaria en el miembro derecho:

$$r_1 \cos\theta_1 + r_4 \cos\theta_4 - r_2 \cos\theta_2 - r_3 \cos\theta_3 = r_2 j \sin\theta_2 + r_3 j \sin\theta_3 - r_1 j \sin\theta_1 - r_4 j \sin\theta_4 \tag{5.6}$$

Separando la parte real y la parte imaginaria en un sistema de ecuaciones se obtiene:

$$r_1 \cos\theta_1 + r_4 \cos\theta_4 = r_2 \cos\theta_2 + r_3 \cos\theta_3 \tag{5.7}$$
$$r_1 j \sin\theta_1 + r_4 j \sin\theta_4 = r_2 j \sin\theta_2 + r_3 j \sin\theta_3 \tag{5.8}$$

Dividiendo por la unidad imaginaria en la ecuación 5.8:

$$r_1 \cos\theta_1 + r_4 \cos\theta_4 = r_2 \cos\theta_2 + r_3 \cos\theta_3 \tag{5.9}$$
$$r_1 \sin\theta_1 + r_4 \sin\theta_4 = r_2 \sin\theta_2 + r_3 \sin\theta_3 \tag{5.10}$$

Configuración	θ_3	θ_4
abierto	$+\surd$	$-\surd$
cerrado	$-\surd$	$+\surd$

TABLA 5.1: Signo del radical según el tipo de mecanismo.

Este sistema de ecuaciones se expresa en función de θ_4 para obtener:

$$r_4 \cos\theta_4 = r_2 \cos\theta_2 + r_3 \cos\theta_3 - r_1 \cos\theta_1 \quad (5.11)$$
$$r_4 \sin\theta_4 = r_2 \sin\theta_2 + r_3 \sin\theta_3 - r_1 \sin\theta_1 \quad (5.12)$$

Se obtiene la ecuación compacta de Freudenstein elevando al cuadrado y sumando los términos del sistema de ecuaciones anterior:

$$A_1 \cos\theta_3 + B_1 \sin\theta_3 + C_1 = 0 \quad (5.13)$$

Donde:

$$A_1 = 2r_3(r_2 \cos\theta_2 - r_1 \cos\theta_1) \quad (5.14)$$
$$B_1 = 2r_3(r_2 \sin\theta_2 - r_1 \sin\theta_1) \quad (5.15)$$
$$C_1 = r_1^2 + r_2^2 + r_3^2 - r_4^2 - 2r_1 r_2 \cos(\theta_1 - \theta_2) \quad (5.16)$$

El ángulo θ_3 se puede obtener como función de A_1, B_1, C_1 y θ_2, si sen θ_3 y $\cos\theta_3$ se expresan en términos de $\tan(\frac{\theta_3}{2})$:

$$\sin\theta_3 = \frac{2tan(\frac{\theta_3}{2})}{1 + tan^2(\frac{\theta_3}{2})} \quad , \quad \cos\theta_3 = \frac{1 - tan^2(\frac{\theta_3}{2})}{1 + tan^2(\frac{\theta_3}{2})} \quad (5.17)$$

Sustituyendo la Ecuación 5.17 en la Ecuación 5.13 se obtiene una ecuación no lineal de segundo orden:

$$[C_1 - A_1] = tan^2\frac{\theta_3}{2} + [2B_1] \tan(\frac{\theta_3}{2}) + A_1 + C1 = 0 \quad (5.18)$$

Resolviendo la ecuación 5.18 el ángulo θ_3 puede ser expresado de la siguiente forma:

$$\theta_3 = 2 \arctan\left[\frac{-B_1 \pm \sqrt{B_1^2 + A_1^2 - C_1^2}}{C_1 - A_1}\right] \quad (5.19)$$

El ángulo θ_4 se obtiene de forma similar, seleccionando el signo del radical para los ángulos θ_3 y θ_4 se seleccionan de acuerdo a la configuración del mecanismo según la Tabla 5.1.

Debido a que el punto de interés del acoplador es C, se establece la siguiente ecuación para determinar su posición en el sistema $O_2X_rY_r$:

$$C_{xr} = r_2\cos\theta_2 + r_{cx}\cos\theta_3 - r_{cy}\sin\theta_3 \tag{5.20}$$
$$C_{yr} = r_2\sin\theta_2 + r_{cy}\sin\theta_3 + r_{cy}\cos\theta_3 \tag{5.21}$$

En el sistema global de coordenadas, este punto puede expresarse como:

$$\begin{bmatrix} C_x \\ C_y \end{bmatrix} = \begin{bmatrix} \cos\theta_0 & -\sin\theta_0 \\ \sin\theta_0 & \cos\theta_0 \end{bmatrix} \begin{bmatrix} C_{xr} \\ C_{xy} \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \tag{5.22}$$

Las ecuaciones anteriores permiten calcular la trayectoria del punto C durante el movimiento de las barras del mecanismo. Teniendo en cuenta este análisis cinemático se puede encontrar una configuración para los componentes del mecanismo tal que el punto C recorra una trayectoria deseada.

5.1.2. Planteamiento del Problema de Optimización para casos de estudio del Mecanismo de Cuatro Barras

Con base en el análisis cinemático abordado en la Sección 5.1.1 se plantea un problema de optimización numérico que tiene como objetivo minimizar el error entre el punto C del acoplador y ciertos puntos de precisión para describir una trayectoria determinada. Los componentes del mecanismo como las barras y los ángulos de rotación están sujetas ciertas restricciones para garantizar el movimiento.

Función Objetivo

La síntesis óptima del mecanismo de cuatro barras persigue hallar una configuración que permita la máxima precisión del mecanismo al realizar una trayectoria deseada. Esto significa que se deben encontrar las longitudes de las barras, el ángulo de rotación respecto al eje OXY, la distancia entre los dos sistemas de referencias, los valores para el conjunto de ángulos para la barra de entrada que permitirán generar la trayectoria deseada. Esta trayectoria puede ser lograda definiendo un conjunto de puntos de precisión. En el sistema global de coordenadas el punto de precisión i esta dado por:

$$C_d^i = \left[C_x^i, C_y^i\right], \tag{5.23}$$

El conjunto de los N puntos se define entonces como:

$$\Omega = \left\{ C_d^i | i \in N \right\} \tag{5.24}$$

Partiendo de las análisis cinemático descrito en la sección 5.1.1 se conoce que dado un conjunto de valores de las barras del mecanismo y sus parámetros x_0, y_0, θ_0, la posición del acoplador puede ser expresada como una función de la posición de la barra de entrada:

$$C^i = \left[C_x\left(\theta_2\right), C_y\left(\theta_2\right) \right], \tag{5.25}$$

Luego, para maximizar la precisión del mecanismo es necesario minimizar la distancia entre los puntos de precisión C_d^i y los puntos calculados C^i. La función propuesta para dicho cálculo es:

$$f(\theta_2^i) = \sum_{i=1}^{N} \left[\left(C_{xd}^i - C_x^i\right)^2 + \left(C_{yd}^i - C_y^i\right)^2 \right] \tag{5.26}$$

Restricciones de diseño

Las restricciones aplicadas al mecanismo de cuatro barras están dirigidas a garantizar la movilidad del mecanismo y a cumplir con especificaciones de dimensionamiento. las restricciones relacionadas con la movilidad del mecanismo se derivan de las *Ley de Grashoff* las cuales establecen el tipo de movimiento del mecanismo de cuatro barras atendiendo a la longitud de sus elementos. Shigley define lo siguiente en [75]:

> *"En un eslabonamiento plano de cuatro barras, la suma de las longitudes más corta y más larga de los eslabones no puede ser mayor que la suma de las longitudes de los dos eslabones restantes, sí se desea que exista una rotación relativa continua entre dos elementos."*

Si se aplica la convención de que el eslabón más largo tiene la longitud l, la del más corto es s y los otros dos tienen las longitudes p y q. La ley de Grashof especifica que uno de los eslabones, en particular el más pequeño, podrá girar continuamente en relación con los otros tres sólo cuando se satisfaga la siguiente desigualdad:

$$s + l \leq p + q \tag{5.27}$$

De forma análoga se platea la ley de Grashoff para el mecanismo de cuatro barras en cuestión:

$$r_1 + r_2 \leq r_3 + r_4 \tag{5.28}$$

Además de estas condiciones, se deben tomar en cuenta las siguientes restricciones para asegurar que el mecanismo cumpla con la ley de Grashof:

$$r_2 < r_3; \quad r_3 < r_4; \quad r_4 < r_1 \tag{5.29}$$

Debido a que el problema de la síntesis es de generación de trayectoria sin sincronizan prescrita, los valores que toman los ángulos de la manivela deben estar ordenados secuencialmente. Si el ángulo del punto de precisión i se denota como θ_2^i , se debe cumplir que:

$$\theta_2^1 < \theta_2^2 < ... < \theta_2^N \tag{5.30}$$

5.1.3. Caso de estudio 1: seguimiento de una trayectoria lineal vertical, sin sincronización previa - MCE1

El presente caso de estudio es tomado de [16], en el cual el acoplador debe pasar por seis puntos de precisión indicados en la Ecuación 5.31, los cuales se encuentran alineados verticalmente.

$$\Omega = \{(20,20),(20,25),(20,30),(20,35),(20,40)(20,45)\} \tag{5.31}$$

El vector de variables para este caso está constituido por 15 componentes:

$$\begin{aligned} \vec{p} &= \{p_1, p_2, p_3, ..., p_{15}\} && (5.32)\\ &= \{r_1, r_2, r_3, r_4, r_{cx}, r_{cy}, \theta_0, x_0, y_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6\} && (5.33) \end{aligned}$$

Las primeras cuatro variables en el vector de diseño corresponde a los valores de la longitud de las barras, las dos variables siguientes r_{cx} y r_{cy} describen la posición del acoplador, θ_0, x_0 y y_0 indican la la posición del sistema de referencia $O_2X_rY_r$ con respecto al sistema OXY , y las variables restantes corresponden a la secuencia de ángulos de la barra de entrada r_2.

Los límites superior e inferior de las variables de diseño se encuentran definidos por:

$$\begin{aligned} r_1, r_2, r_3, r_4 &\in [0, 60] && (5.34)\\ r_{cx}, r_{cy}, \theta_0, x_0, y_0 &\in [-60, 60] && (5.35)\\ \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6 &\in [0, 2\pi] && (5.36)\\ & && (5.37) \end{aligned}$$

El problema de optimización mono-objetivo para el este caso de estudio se define como:

$$\begin{aligned} min \quad & f(\vec{p}) = \sum_{i=1}^{N} \left[\left(C_{xd}^i - C_x^i\right)^2 + \left(C_{yd}^i - C_y^i\right)^2 \right] \\ & \vec{p} \in \mathbb{R}^{15} \end{aligned} \tag{5.38}$$

s.a:

$$g_1(\vec{p}) = p_1 + p_2 - p_3 - p_4 \leq 0, \tag{5.39}$$
$$g_2(\vec{p}) = p_2 - p_3 \leq 0, \tag{5.40}$$
$$g_3(\vec{p}) = p_3 - p_4 \leq 0, \tag{5.41}$$
$$g_4(\vec{p}) = p_4 - p_1 \leq 0, \tag{5.42}$$
$$g_5(\vec{p}) = p_{10} - p_{11} \leq 0, \tag{5.43}$$
$$g_6(\vec{p}) = p_{11} - p_{12} \leq 0, \tag{5.44}$$
$$g_7(\vec{p}) = p_{12} - p_{13} \leq 0, \tag{5.45}$$
$$g_8(\vec{p}) = p_{13} - p_{14} \leq 0, \tag{5.46}$$
$$g_9(\vec{p}) = p_{14} - p_{15} \leq 0 \tag{5.47}$$

5.1.4. Caso de estudio 2: seguimiento de una trayectoria no alineada, con sincronizacion previa - MCE2

En el segundo caso de estudio se plantea una trayectoria no alineada con una sincronización prescrita. El conjunto de puntos de precisión que conforma la trayectoria deseada esta constituido por los siguientes elementos:

$$\Omega = \{(3,3), (2.759, 3.363), (2.372, 3.663), (1.890, 3.862), (1.355, 3.943)\} \tag{5.48}$$

Para la sincronizacion prescrita se plantean la siguiente secuencia de ángulos:

$$\Omega = \left\{ \frac{2\pi}{12}, \frac{3\pi}{12}, \frac{4\pi}{12}, \frac{5\pi}{12}, \frac{6\pi}{12} \right\} \tag{5.49}$$

Debido a la sincronizan prescrita se necesita calcular sólo la longitud de las barras. Por tanto, vector de diseño se define como:

$$\vec{p} = \{p_1, p_2, p_3, p_4, p_5, p_6\} \tag{5.50}$$
$$= \{r_1, r_2, r_3, r_4, r_{cx}, r_{cy}\} \tag{5.51}$$

Los límites superior e inferior de las variables de diseño se encuentran definidos por:

$$\begin{aligned} r_1, r_2, r_3, r_4 &\in [0, 50] && (5.52)\\ r_{cx}, r_{cy} &\in [-50, 50] && (5.53)\\ & && (5.54) \end{aligned}$$

Luego, se define el problema de optimización mono-objetivo para el segundo caso de estudio según la Ecuación 5.55 a la Ecuación 5.59:

$$\begin{aligned} min \quad f(\vec{p}) = \sum_{i=1}^{N} \left[\left(C_{xd}^i - C_x^i\right)^2 + \left(C_{yd}^i - C_y^i\right)^2 \right] \\ \vec{p} \in \mathbb{R}^6 \end{aligned} \tag{5.55}$$

s.a:

$$\begin{aligned} g_1(\vec{p}) &= p_1 + p_2 - p_3 - p_4 \leq 0, && (5.56)\\ g_2(\vec{p}) &= p_2 - p_3 \leq 0, && (5.57)\\ g_3(\vec{p}) &= p_3 - p_4 \leq 0, && (5.58)\\ g_4(\vec{p}) &= p_4 - p_1 \leq 0 && (5.59) \end{aligned}$$

5.1.5. Caso de estudio 3: generación de movimiento delimitado por un conjunto de pares de puntos - MCE3

En el tercer caso de estudio, tomado de [76], se plantean diez pares de puntos de precisión $(K = 10)$, dados por las coordenadas de la Tabla 5.2. El vector de variables de diseño define entonces como:

$$\begin{aligned} \vec{p} &= \{p_1, p_2, p_3, ..., p_{19}\} && (5.60)\\ &= \{r_1, r_2, r_3, r_4, r_{cx}, r_{cy}, \theta_0, x_0, y_0, \theta_2^1, ..., \theta_2^{10}\} && (5.61) \end{aligned}$$

Los límites superior e inferior para las variables de diseño son definidos como:

$$\begin{aligned} r_1, r_2, r_3, r_4 &\in [0, 60] && (5.62)\\ r_{cx}, r_{cy}, \theta_0, x_0, y_0 &\in [-60, 60] && (5.63)\\ \theta_0, \theta_2^1, ..., \theta_2^{10} &\in [0, 2\pi] && (5.64)\\ & && (5.65) \end{aligned}$$

En este caso se necesita minimizar la distancia entre los puntos de la trayectoria de C en el acoplador y los K pares de puntos de precisión. Básicamente, el punto C debe pasar entre cada par de puntos de precisión de manera que la sumatoria

Par	C_{1d}	C_{2d}
1	(1.768, 2.3311)	(1.9592, 2.44973)
2	(1.947, 2.6271)	(2.168, 2.675)
3	(1.595, 2.7951)	(1.821, 2.804)
4	(1.019, 2.7241)	(1.244, 2.720)
5	(0.479, 2.4281)	(0.705, 2.437)
6	(0.126, 2.0521)	(0.346, 2.104)
7	(-0.001, 1.720)	(0.195, 1.833)
8	(0.103, 1.514)	(0.356, 1.680)
9	(0.442, 1.549)	(0.558, 1.742)
10	(1.055, 1.905)	(1.186, 2.088)

TABLA 5.2: Pares de puntos de precisión para el Caso de Estudio 3 del Mecanismo de Cuatro Barras.

de las distancias entre cada punto del par i y C sea mínima. Por tanto, se plantea una función diferente y trece restricciones como se describe desde la Ecuación 5.66 a la Ecuación 5.79:

$$min \quad f(\vec{p}) = \sum_{i=1}^{K} \left[\left(C_{1xd}^i - C_x^i\right)^2 + \left(C_{2yd}^i - C_y^i\right)^2 \right] + \sum_{i=1}^{K} \left[\left(C_{2xd}^i - C_x^i\right)^2 + \left(C_{2yd}^i - C_y^i\right)^2 \right] \qquad \vec{p} \in \mathbb{R}^{19} \tag{5.66}$$

s.a:

$$g_1(\vec{p}) = p_1 + p_2 - p_3 - p_4 \leq 0, \tag{5.67}$$
$$g_2(\vec{p}) = p_2 - p_3 \leq 0, \tag{5.68}$$
$$g_3(\vec{p}) = p_3 - p_4 \leq 0, \tag{5.69}$$
$$g_4(\vec{p}) = p_4 - p_1 \leq 0, \tag{5.70}$$
$$g_5(\vec{p}) = p_{10} - p_{11} \leq 0, \tag{5.71}$$
$$g_6(\vec{p}) = p_{11} - p_{12} \leq 0, \tag{5.72}$$
$$g_7(\vec{p}) = p_{12} - p_{13} \leq 0, \tag{5.73}$$
$$g_8(\vec{p}) = p_{13} - p_{14} \leq 0, \tag{5.74}$$
$$g_9(\vec{p}) = p_{14} - p_{15} \leq 0, \tag{5.75}$$
$$g_{10}(\vec{p}) = p_{15} - p_{16} \leq 0, \tag{5.76}$$
$$g_{11}(\vec{p}) = p_{16} - p_{17} \leq 0, \tag{5.77}$$
$$g_{12}(\vec{p}) = p_{17} - p_{18} \leq 0, \tag{5.78}$$
$$g_{13}(\vec{p}) = p_{18} - p_{19} \leq 0 \tag{5.79}$$

5.1.6. Diseño de un Efector Final de Tres Dedos

De forma similar a los casos de estudio planteados en el problema del mecanismo de cuatro barras, en este problema se desea obtener la síntesis dimensional de generación de trayectoria. El mecanismo en este caso es un efector final o gripper, el cual está compuesto de tres dedos. Cada dedo esta diseñado con un mecanismo de seis barras (Figura 5.2) cuya configuración permite un agarre esférico. En este mecanismo los dedos son simétricos por lo que es suficiente obtener la síntesis de uno de ellos para construir el efector final.

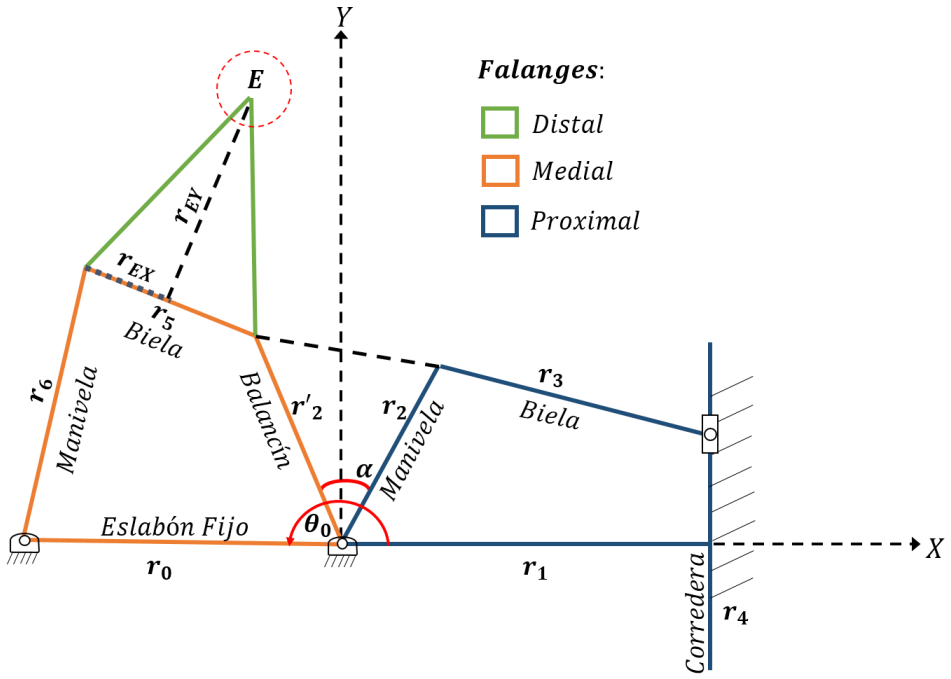


FIGURA 5.2: Mecanismo de seis barras para el diseño de un dedo del Efector Final. Tomado de [13]

En la Figura 5.2, se muestra el diseño de un dedo conformado por dos sub-sistemas que simulan las falanges del mismo. El primer sub-sistema corresponde a un mecanismo manivela-biela-corredera (MBC) que se usa en la entrada (r_2, r_3, r_4), y el segundo es un mecanismo de cuatro barras en la salida (r_0, r'_2, r_5, r_6). Ambos mecanismos conforman el sistema, en donde el impulsor es la corredera y el punto E en el acoplador es la salida. El análisis de este problema fue tomado del trabajo realizado por Capistrán, Portilla y Mezura en [77] y se presenta en la siguiente sección.

5.1.7. Cinemática del Efector Final de Tres Dedos

Para llegar al planteamiento final de este problema de optimización se debe realizar primeramente el análisis de los dos sub-sistemas mencionados en la sección anterior individualmente. Luego, las ecuaciones que describen el comportamiento cinemático de ambos sistemas son integrados para conformar el problema de optimización. Primeramente se analiza el mecanismo MBC (Figura 5.3) respecto al eje X según la posición de la corredera r_4, en donde se pueden presentar tres casos posibles: $r_4 > 0$, $r_4 < 0$ y $r_4 = 0$. En cada caso la variable de interés que describe al sub-sistema es θ_3. A continuación se presenta el modelo de cada caso [77]:

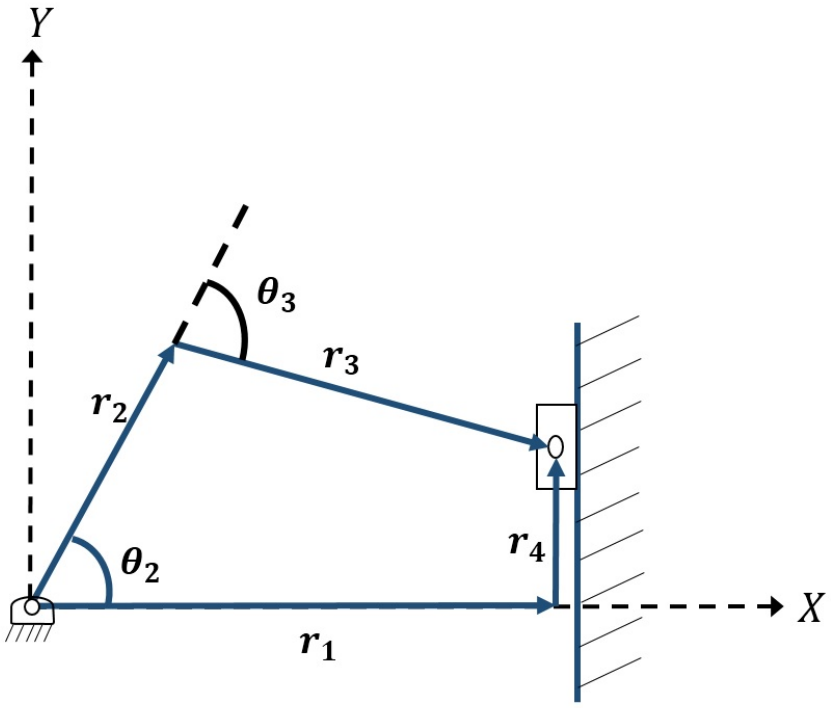


FIGURA 5.3: Mecanismo manivela-biela-corredera en un dedo del Efector Final. Tomado de [13]

1. **Si** $r_4 > 0$ **entonces**:

   Se calcula el valor del ángulo θ_3 mediante la sumatoria de los ángulos θ_2, φ y π (igual a 180). En la Figura 5.4 se muestran los ángulos requeridos para efectuar los cálculos necesarios en este caso.

   Para calcular θ_2 se procede de la siguiente forma:

$$\theta_2 = \beta + \gamma \tag{5.80}$$

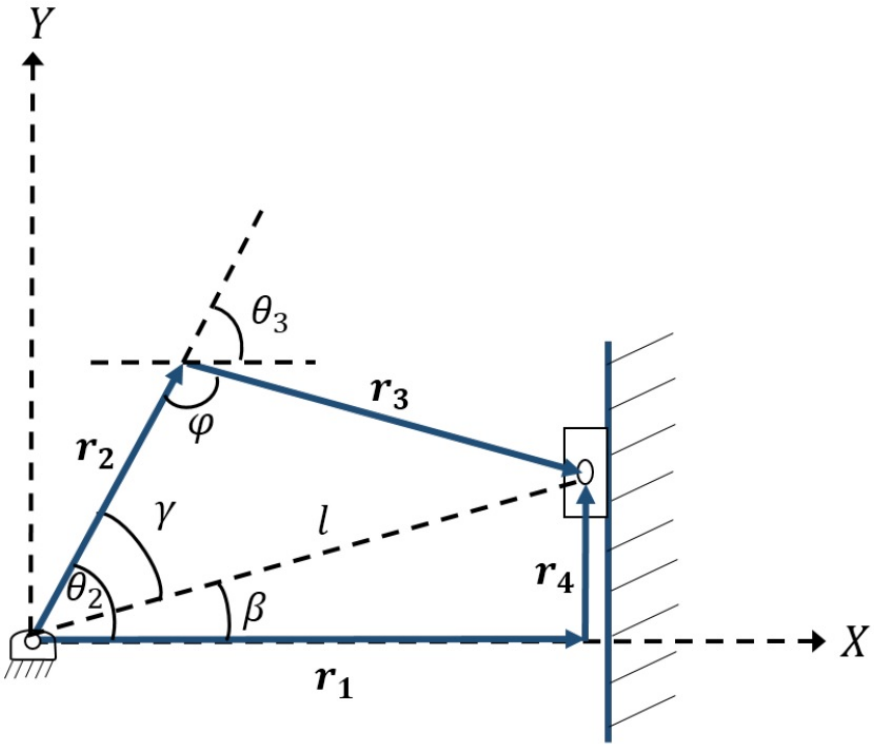


FIGURA 5.4: Mecanismo manivela-biela-corredera para $r_4 > 0$ un dedo del el Efector Final. Tomado de [13]

Para obtener γ debe calcular l mediante el teorema de pitágoras:

$$l = \sqrt{r_1^2 + r_4^2} \tag{5.81}$$

Aplicando la ley de cosenos y despejando se obtiene la Ecuación 5.82

$$\gamma = \cos^{-1}\left[\frac{r_2^2 + l^2 - r_3^2}{2r_2 l}\right] \tag{5.82}$$

Para cálculo del ángulo se plantea que:

$$\beta = \tan^{-1}\left[\frac{|r_4|}{r_1}\right] \tag{5.83}$$

Sustituyendo las Ecuaciones 5.82 y 5.83 en la Ecuación 5.80 se obtiene:

$$\theta_2 = \cos^{-1}\left[\frac{r_2^2 + l^2 - r_3^2}{2r_2 l}\right] + \tan^{-1}\left[\frac{|r_4|}{r_1}\right] \tag{5.84}$$

El valor de φ se calcula aplicando ley de cosenos y se despejando:

$$\varphi = \cos^{-1}\left[\frac{r_2^2 + r_3^2 - l^2}{2r_2r_3}\right] \tag{5.85}$$

Luego, el valor de θ_3 se calcula por la sumatoria expresada en la Ecuación 5.86

$$\theta_3 = \theta_2 + \varphi + \pi \tag{5.86}$$

Esto es:

$$\theta_3 = \cos^{-1}\left[\frac{r_2^2 + l^2 - r_3^2}{2r_2l}\right] + \tan^{-1}\left[\frac{|r_4|}{r_1}\right] + \cos^{-1}\left[\frac{r_2^2 + r_3^2 - l^2}{2r_2r_3}\right] + \pi \tag{5.87}$$

2. **Si** $r_4 < 0$ **entonces**:

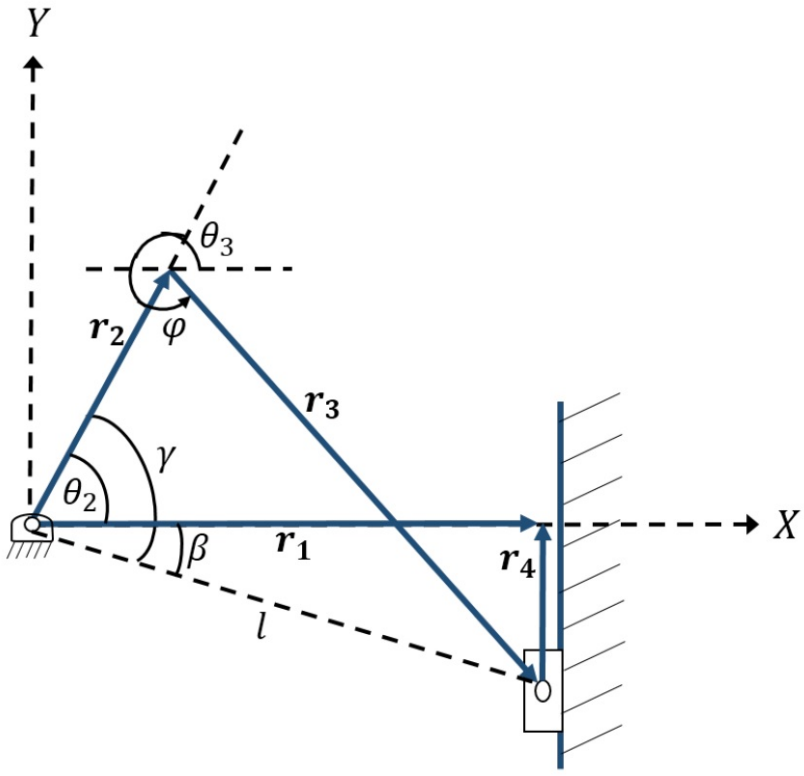


FIGURA 5.5: Mecanismo manivela-biela-corredera para $r_4 < 0$ en un dedo del Efector Final. Tomado de [13]

De forma similar al caso anterior el análisis matemático es similar al del caso anterior, se debe calcular el valor del ángulo θ_3 solo que, en este caso el

ángulo θ_2 es la diferencia entre los ángulos β y γ:

$$\theta_2 = \beta - \gamma \tag{5.88}$$

Sustituyendo las Ecuaciones 5.82 y 5.83 en la Ecuación 5.88 se obtiene:

$$\theta_2 = \cos^{-1}\left[\frac{r_2^2 + l^2 - r_3^2}{2r_2 l}\right] - \tan^{-1}\left[\frac{|r_4|}{r_1}\right] \tag{5.89}$$

Luego, el valor de θ_3 se obtiene sustituyendo este valor de θ_2 y el valor de φ de la Ecuación 5.85, en la Ecuación 5.86:

$$\theta_3 = \cos^{-1}\left[\frac{r_2^2 + l^2 - r_3^2}{2r_2 l}\right] - \tan^{-1}\left[\frac{|r_4|}{r_1}\right] + \cos^{-1}\left[\frac{r_2^2 + r_3^2 - l^2}{2r_2 r_3}\right] + \pi \tag{5.90}$$

3. **Si** $r_4 = 0$ **entonces**:

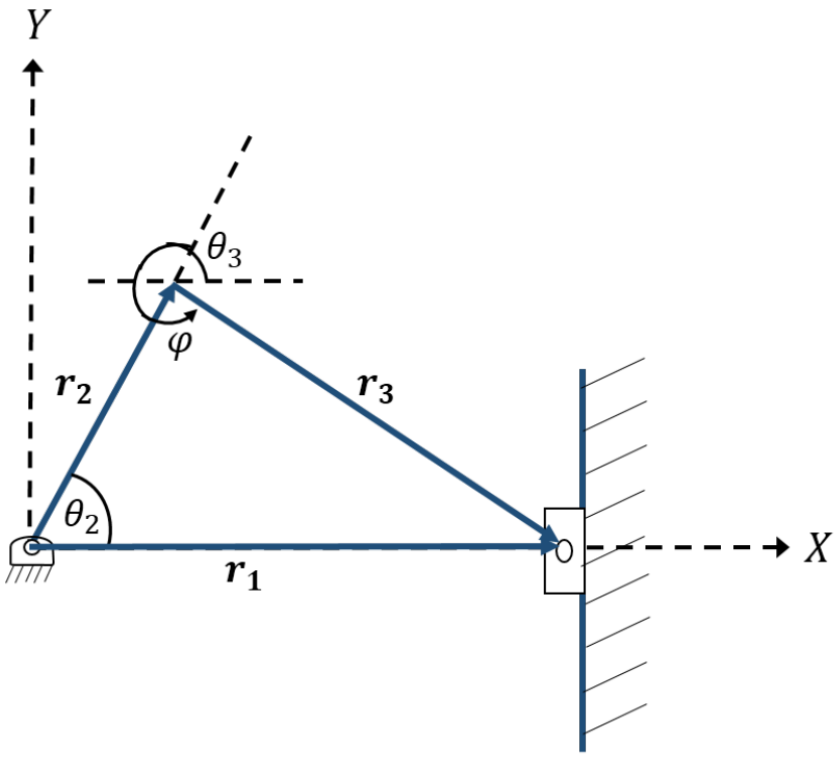


FIGURA 5.6: Mecanismo manivela-biela-corredera para $r_4 = 0$ en un dedo del Efector Final. Tomado de [13]

Debido que $r_4 = 0$ corredera del mecanismo se alinea con el eje x. Al igual que en los casos anteriores el valor del ángulo θ_3 es igual a la suma de los ángulos θ_2, φ y π . Para el calculo de θ_2y φ se aplica la ley de cosenos y se despeja, obteniendo las Ecuaciones 5.91 y 5.92

$$\theta_2 = \cos^{-1}\left[\frac{r_1^2 + r_2^2 - r_3^2}{2r_1r_2}\right] \tag{5.91}$$

$$\varphi = \cos^{-1}\left[\frac{r_2^2 + r_3^2 - r_1^2}{2r_2r_3}\right] \tag{5.92}$$

Finalmente, se sustituyen los ángulos θ_2 y φ en la Ecuación 5.86:

$$\theta_3 = \cos^{-1}\left[\frac{r_1^2 + r_2^2 - r_3^2}{2r_1r_2}\right] + \cos^{-1}\left[\frac{r_2^2 + r_3^2 - r_1^2}{2r_2r_3}\right] + \pi \tag{5.93}$$

El análisis cinemático presentado hasta el momento corresponde al sub-sistema MBC del mecanismo de seis barras. A continuación se procede a desarrollar el modelo cinemático del mecanismo de cuatro barras conformado por los eslabones r_0, r_2', r_5 y r_6. En la Figura 5.7 se muestra dicho mecanismo sin el acoplador. Nótese, que la barra r_2 del sub-sistema MBC también forma parte del mecanismo de cuatro barras fungiendo como su impulsor y formando junto con r_2' un único eslabón.

El análisis cinemático del sub-sistema de cuatro barras es similar al del problema descrito en la Sección 5.1.1. Para iniciar se establece la ecuación de lazo cerrado:

$$\vec{r_2'} + \vec{r_5} = \vec{r_0} + \vec{r_6} \tag{5.94}$$

Utilizando la notación polar se tiene que:

$$r_2' e^{j\theta_2'} + r_5 e^{j\theta_5} = r_0 e^{j\theta_0} + r_6 e^{j\theta_6} \tag{5.95}$$

Aplicando la ecuación de Euler y separando la parte real de la parte imaginaria se obtiene:

$$r_2' \cos\theta_2' + r_5 \cos\theta_4 = r_0 \cos\theta_2 + r_6 \cos\theta_6 \tag{5.96}$$

$$r_2' \sin\theta_2' + r_5 \sin\theta_4 = r_0 \sin\theta_2 + r_6 \sin\theta_6 \tag{5.97}$$

en donde:

$$\theta_2' = \theta_2 + \alpha \tag{5.98}$$

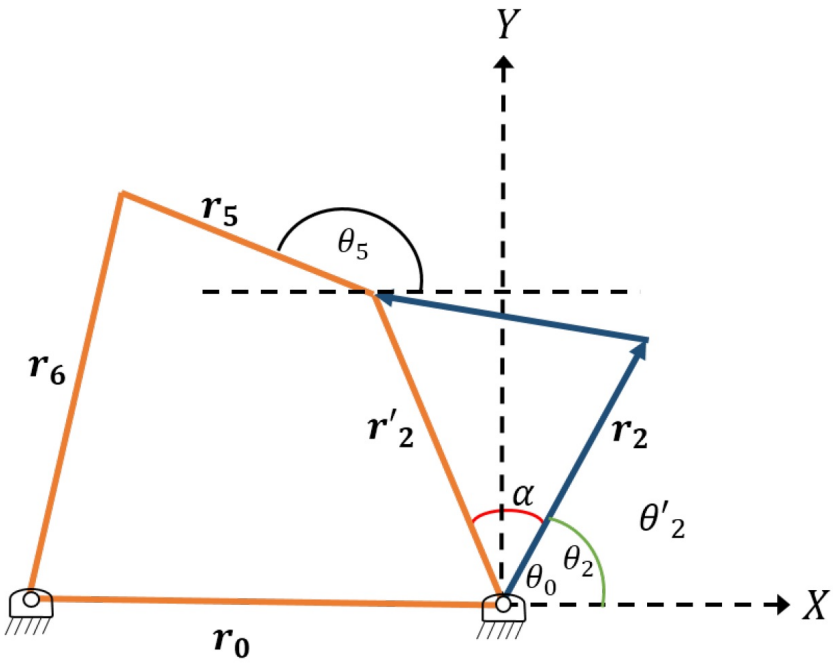


FIGURA 5.7: Mecanismo de cuatro barras de un dedo del Efector Final. Tomado de [13]

El ángulo θ_2 se obtiene del análisis cinemático del sub-sistema MBC. Luego para calcular θ_5, el sistema de ecuaciones 5.96 se expresa en función de θ_4 para obtener:

$$r_6 \cos\theta_6 = r'_2 \cos\theta'_2 + r_5 \cos\theta_5 - r_0 \cos\theta_0 \quad (5.99)$$

$$r_6 \sin\theta_6 = r'_2 \sin\theta'_2 - r_5 \sin\theta_5 r_0 \cos\theta_0 \quad (5.100)$$

Se obtiene la ecuación compacta de Freudenstein elevando al cuadrado los términos del sistema de ecuaciones anterior y sumando sus términos:

$$A\cos\theta_3 + B\sin\theta_3 + C = 0 \quad (5.101)$$

Donde:

$$A = 2r_5(r'_2\cos\theta'_2 - r_0\cos\theta_0) \quad (5.102)$$

$$B = 2r_5(r'_2\sin\theta'_2 - r_0\sin\theta_0) \quad (5.103)$$

$$C = r_0^2 + r'_2 + r_5^2 - r_6^2 - 2r_0 r'_2\cos(\theta_0 - \theta'_2) \quad (5.104)$$

El ángulo θ_5 se puede obtener como función de A, B y C, si $\operatorname{sen}\theta_5$ y $\cos\theta_5$ se expresan en términos de $\tan(\frac{\theta_5}{2})$:

$$\sin\theta_5 = \frac{2tan(\frac{\theta_5}{2})}{1+tan^2(\frac{\theta_5}{2})} \quad , \quad \cos\theta_5 = \frac{1-tan^2(\frac{\theta_5}{2})}{1+tan^2(\frac{\theta_5}{2})} \tag{5.105}$$

Sustituyendo la Ecuación 5.105 en la Ecuación 5.101 se obtiene una ecuación no lineal de segundo orden:

$$[C-A] = tan^2\frac{\theta_3}{2} + [2B]\tan(\frac{\theta_3}{2}) + A + C = 0 \tag{5.106}$$

Resolviendo la ecuación 5.106 el ángulo θ_3 puede ser expresado de la siguiente forma:

$$\theta_5 = 2\arctan\left[\frac{-B \pm \sqrt{B^2+A^2-C^2}}{C-A}\right] \tag{5.107}$$

El ángulo θ_6 se obtiene de forma similar, siendo la ecuación compacta de Freudenstein:

$$D\cos\theta_6 + E\sin\theta_6 + F = 0 \tag{5.108}$$

Donde:

$$D = 2r_6(r_0\cos\theta_0 - r_2'\cos\theta_2') \tag{5.109}$$
$$E = 2r_6(r_0\sin\theta_0 - r_2'\cos\theta_2') \tag{5.110}$$
$$F = r_0^2 + r_2'^2 + r_5^2 - r_6^2 - 2r_0r_2'\cos(\theta_0-\theta_2') \tag{5.111}$$

Luego, θ_6 se define como:

$$\theta_6 = 2\arctan\left[\frac{-E \pm \sqrt{E^2+D^2-F^2}}{F-D}\right] \tag{5.112}$$

Las Ecuaciones 5.107 y 5.112 determinan los ángulos de la biela y el balancín del mecanismo de cuatro barras. El signo del radical para los ángulos θ_5 y θ_6 también se seleccionan de acuerdo a la la Tabla 5.1 de la Sección 5.1.1. Para analizar, se debe determinar la posición del punto E del acoplador (Figura 5.8) a través de la siguiente relación:

$$E = r_2' + r_{EX} + r_{EY} \tag{5.113}$$

En términos de componentes tenemos que:

$$E_x = r_2'\cos\theta_2' + r_{EX}\cos\theta_5 + r_{EY}\sin\theta_{EY} \tag{5.114}$$
$$E_y = r_2'\cos\theta_2' + r_{EY}\sin\theta_5 + r_{EY}\cos\theta_{EY} \tag{5.115}$$

en donde:

$$\theta_2' = \theta_2 - \alpha \tag{5.116}$$
$$\theta_{EY} = \theta_5 - 90 \tag{5.117}$$

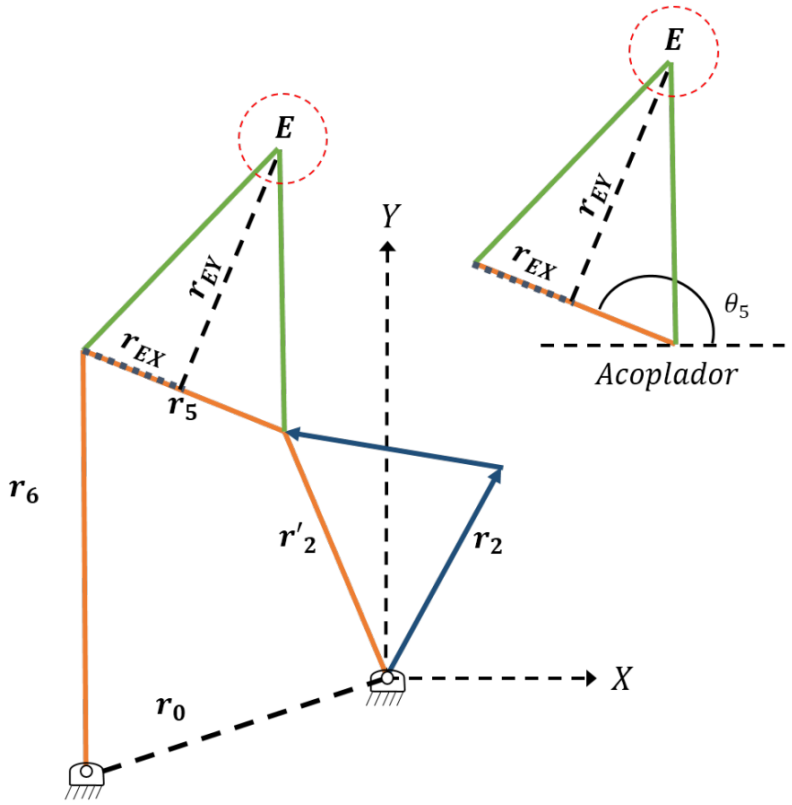


FIGURA 5.8: Acoplador del mecanismo de cuatro barras en un dedo del Efector Final. Tomado de [13]

5.1.8. Declaración de Problemas de Optimización para Casos de Estudio del Efector

Para la optimización del diseño correspondiente a un dedo del efector final, se deben encontrar las dimensiones de todas los eslabones del mecanismo de seis barras de tal forma que el punto E del acoplador siga una trayectoria esferica. Los puntos de la trayectoria se encuentran definidos por la Ecuación 5.118 y fueron tomados de [77].

$$\Omega = \{(10, 160), (10, 170), (40, 165)\} \tag{5.118}$$

El vector de variables para este caso está constituido por 14 componentes:

$$\vec{p} = \{p_1, p_2, p_3, ..., p_{14}\} \quad (5.119)$$
$$= \{r_1, r_2, r_3, r_4^1, r_4^2, r_4^3, r_0, r_2', r_5, r_6, r_{EX}, r_{EY}, \alpha, \theta_0\} \quad (5.120)$$

En donde r_1, r_2, r_3, r_0, r_2', r_5 y r_6 son las dimensiones de las barras. Las variables r_4^1,r_4^2,r_4^3, corresponden a las posiciones de la corredera y se emplean para calcular los puntos en los que se ubica el sistema, r_{EX} y r_{EY} son las dimensiones de las barras del acoplador, mientras que α es el ángulo entre r_2 y r_2' , y θ_0 el ángulo de r_0.

Función objetivo

Para el primer caso del efector final se plantea la función de error

$$min \quad f(r_4^i) = \sum_{i=1}^{N} \left[\left(C_{1d}^i - C^i\right)^2 = \sum_{i=1}^{N} \left[\left(C_{1xd}^i - C_x^i\right)^2 + \left(C_{yd}^i - C_y^i\right)^2 \right] \right. \quad (5.121)$$

En la Ecuación 5.121 se calcula la distancia (a minimizar) de los puntos de precisión C_d^i y los puntos calculados C_i a partir de r_4^i. Donde C_x^i y C^i son las coordenadas (x, y) del punto E del acoplador.

Restricciones de diseño

Para el correcto funcionamiento del gripper se deben tener en cuenta las siguientes restricciones de diseño:

- *Ley de Grashof:* De forma similar al mecanismo de 4 barras, debe cumplirse la ley de Grashof para el mecanismo de seis barras:

  $$r_5 + r_6 \leq r_0 + r_2' \quad (5.122)$$

  Adicionalmente debe cumplirse que:

  $$r_0 < r_6, \quad r_2' < r_6, \quad r_5 < r_6 \quad (5.123)$$

- *Secuencia de posición de la corredera*: Las posiciones de la corredera deben estar ordenadas de tal forma que cada posición preceda a la siguiente de acuerdo a su magnitud:

  $$r_4^1 > r_4^2 > r_4^3 \quad (5.124)$$

- *Consideraciones sobre barras*: Esta restricción tiene como objetivo garantizar un alto nivel que la estética del acoplador. Para lograr esto r_{EX} debe ser lo más parecida posible a r_5, por lo que:

$$r_{EX} - r_5 \leq 0 \tag{5.125}$$

Además, para el sub-sistema MBC, la relación recomendada entre biela y manivela se encuentra entre $\frac{1}{3}$ y $\frac{1}{5}$, esto es:

$$\frac{1}{5} \leq \frac{r_2}{r_3} \leq \frac{1}{3} \tag{5.126}$$

Luego:

$$r_2 - \frac{r_3}{3} \leq 0 \quad , \quad \frac{r_3}{5} - r_2 \leq 0 \tag{5.127}$$

Finalmente, se formulan en las siguientes secciones dos casos de estudio para el diseño óptimo del gripper, ambos comparten el mismo vector de variables y el conjunto de puntos de precisión.

5.1.9. Caso de estudio 1: Diseño sin normalización de barras - GCE1

El problema de optimización mono-objetivo para el este caso de estudio se define como:

$$\begin{aligned} min \quad & f(\vec{p}) = \sum_{i=1}^{3} \left[\left(C_{xd}^i - C_x^i\right)^2 + \left(C_{yd}^i - C_y^i\right)^2 \right] \\ & \vec{p} \in \mathbb{R}^{15} \end{aligned} \tag{5.128}$$

s.a:

$$g_1(\vec{p}) = p_{10} + p_9 - p_8 - p_7 \leq 0, \tag{5.129}$$
$$g_2(\vec{p}) = p_7 - p_{10} \leq 0, \tag{5.130}$$
$$g_3(\vec{p}) = p_8 - p_{10} \leq 0, \tag{5.131}$$
$$g_4(\vec{p}) = p_9 - p_{10} \leq 0, \tag{5.132}$$
$$g_5(\vec{p}) = p_{11} - p_9 \leq 0, \tag{5.133}$$
$$g_6(\vec{p}) = |p_4| - |p_5| \leq 0, \tag{5.134}$$
$$g_7(\vec{p}) = |p_5| - |p_6| \leq 0, \tag{5.135}$$
$$g_8(\vec{p}) = p_2 - \frac{p_3}{3} \leq 0, \tag{5.136}$$
$$g_9(\vec{p}) = \frac{p_3}{3} - p_2 \leq 0 \tag{5.137}$$

Los límites superior e inferior de cada variable de diseño son los siguientes:

$$p_1, p_2, p_3 \in [0, 100] \tag{5.138}$$
$$p_4, p_5, p_6 \in [-50, 0] \tag{5.139}$$
$$p_7, p_8, p_{10}, p_{11}, p_{12} \in [0, 150] \tag{5.140}$$
$$p_9 \in [0, 50] \tag{5.141}$$
$$p_{13} \in \left[\frac{\pi}{12}, \pi\right] \tag{5.142}$$
$$p_{14} \in \left[\frac{3\pi}{4}, \frac{5\pi}{4}\right] \tag{5.143}$$

5.1.10. Caso de estudio 2: Diseño con normalización de barras - GCE2

En el segundo caso de estudio se agrega una función de normalización ($f(\vec{x})$) para las barras del mecanismo, con el fin de obtener un diseño estético y antropomórfico de las falanges del gripper. La función objetivo queda definida de la siguiente forma:

$$f_r = w_1 f(\vec{p}) + w_2 f(\vec{x}) \tag{5.144}$$

esto es:

$$min \quad f(\vec{p}) = \sum_{i=1}^{3} \left[\left(C_{xd}^i - C_x^i\right)^2 + \left(C_{yd}^i - C_y^i\right)^2 \right] \qquad \vec{p} \in \mathbb{R}^{14} \tag{5.145}$$

$$min \quad f(\vec{x}) = \frac{\sqrt{\sum_{i=1}^{6} \sum_{j=1} (x_j - x_i)^2}}{\sqrt{\sum_{i=1}^{6} x_i^2}} \tag{5.146}$$

s.a:

$$g_1(\vec{p}) = p_{10} + p_9 - p_8 - p_7 \leq 0, \tag{5.147}$$
$$g_2(\vec{p}) = p_7 - p_{10} \leq 0, \tag{5.148}$$
$$g_3(\vec{p}) = p_8 - p_{10} \leq 0, \tag{5.149}$$
$$g_4(\vec{p}) = p_9 - p_{10} \leq 0, \tag{5.150}$$
$$g_5(\vec{p}) = p_{11} - p_9 \leq 0, \tag{5.151}$$
$$g_6(\vec{p}) = |p_4| - |p_5| \leq 0, \tag{5.152}$$
$$g_7(\vec{p}) = |p_5| - |p_6| \leq 0, \tag{5.153}$$
$$g_8(\vec{p}) = p_2 - \frac{p_3}{3} \leq 0, \tag{5.154}$$
$$g_9(\vec{p}) = \frac{p_3}{3} - p_2 \leq 0 \tag{5.155}$$

Los límites superior e inferior para las variables de diseño son:

$$p_1, p_2, p_3 \in [0, 100] \tag{5.156}$$
$$p_4, p_5, p_6 \in [-50, 0] \tag{5.157}$$
$$p_7, p_8, p_{10}, p_{11}, p_{12} \in [0, 150] \tag{5.158}$$
$$p_9 \in [0, 50] \tag{5.159}$$
$$p_{13} \in \left[\frac{\pi}{12}, \pi\right] \tag{5.160}$$
$$p_{14} \in \left[\frac{3\pi}{4}, \frac{5\pi}{4}\right] \tag{5.161}$$

En la Ecuación 5.144, w_1 y $w2$, son pesos que controlan la precisión de la trayectoria y la estética del mecanismo, respectivamente. La suma de ambos pesos debe ser igual 1, por lo que, para este caso se tomarán los valores de $w_1 = 0.9$ y $w_2 = 0.1$. Las variables x_i y x_j de la Ecuación 5.146 están relacionadas con las coordenadas de cada barra del sistema.

5.2. Gestión de Generación de Energía en una Microrred Eléctrica

En la actualidad los sistemas de generación y distribución de energía convencionales enfrentan diversos problemas como el agotamiento gradual de los recursos de combustibles fósiles, la escasa eficiencia energética y la contaminación ambiental. Estos problemas han impulsado la nueva tendencia de generar energía localmente mediante el uso de *Fuentes de Energía Renovables* (FER) como gas natural, biogás, energía eólica, panales solares fotovoltaicos, sistemas combinados de calor y energía, y su integración en la red de distribución energía de uso público. Este tipo de generación de energía se denomina *generación distribuida* (GD) y las fuentes de energía se denominan *recursos de energía distribuida* (RED). El término generación distribuida se utiliza para distinguir este concepto de generación convencional centralizada. Debido al aumento tanto de la demanda de energía eléctrica como del uso las FER, la utilización de microrredes eléctricas ha tenido un aumento en las ultimas décadas. Lasseter plantea lo siguiente en [78]:

> *"Las microrredes son redes de suministro de energía de pequeña escala, diseñadas para suministrar cargas eléctricas o térmicas a una comunidad pequeña, como una urbanización o una localidad suburbana. Una Microrred es esencialmente una red de distribución ya que es un conglomerado de sistemas GD con diferentes cargas a nivel de voltaje de distribución. Los generadores o micro-fuentes empleados en una Microrred suelen ser de Generación Distribuida renovable o no convencional."*

Desde el punto de vista operativo, las micro-fuentes deben estar equipadas con interfaces electrónicas de potencia (IEP) y controles para proporcionar la flexibilidad requerida y garantizar así el funcionamiento como sistema único. Esta flexibilidad de control permite que la Micro-red se presente al sistema de energía de la red principal como una sola unidad controlada que satisface las necesidades locales de energía. La principal ventaja de una Microrred es su facilidad de control, así como el cumplimiento de las regulaciones de la red sin obstaculizar la confiabilidad y seguridad del servicio eléctrico. Desde el punto de vista de los clientes, las microrredes son beneficiosas para satisfacer localmente sus requisitos eléctricos. Pueden suministrar energía ininterrumpible, mejorar la confiabilidad local y proporcionar soporte de voltaje local. Desde el punto de vista ambiental, las microrredes reducen la contaminación ambiental y el calentamiento global mediante la utilización de tecnología que produce bajas emisiones de carbono [78].

Sin embargo, para lograr una operación estable y segura mediante micro-redes, es necesario resolver una serie de problemas técnicos, normativos y económicos. Ciertas características de las microrredes pueden resultar problemáticas. Algunas intrínsecas, como la naturaleza intermitente y dependiente del clima de la generación mediante ciertos tipos de FER como la fotovoltaica y la eólica; u otras como el bajo contenido de energía de los combustibles no fósiles, la falta de estándares y regulaciones para operar las Microrredes de forma factible. El estudio de estas cuestiones requiere un amplio análisis en tiempo real y en simulaciones, que ha sido objeto de diferentes investigaciones.

5.2.1. Optimización del costo de la generación de energía en una Microrred Eléctrica No Interconectable - SCE1

En el presente problema el objetivo es minimizar el costo de operación de una Microrred Eléctrica en un lugar remoto no interconectable (MR-RNI), al tiempo que se garantiza el equilibrio entre la generación y la demanda. La generación fiable de energía se logra mediante la generación híbrida de potencia en donde se emplean Fuentes de Energía Renovables (FER), generación convencional con diésel y Sistemas de Almacenamiento de Energía (SAE). Las variables de diseño representan entonces, las potencias que serán suministradas por los Generadores de Diésel (GD), energía fotovoltáica (GFV), energía eólica (GE) y el de la MR-RNI en función del tiempo dentro de un intervalo de 24 horas. Se define un modelo de optimización basado en el despacho económico, el cual intenta obtener el menor costo posible derivado del uso del sistema, mientras los generadores entregan potencia a la carga total. El cálculo de minimización del costo de la generación de energía eléctrica en la MR-RNI se debe realizar para un periodo de 24, donde cada hora del día representa un problema de optimización con límites para las variables de diseño diferentes en cada hora. Esto se debe a que los recursos para

la generación diaria son variables. Específicamente debe tomarse en cuenta los siguientes escenarios [13]:

1. Las FER no pueden entregar energía las 24 horas.
2. La demanda total de energía no puede ser entregada solo por las FER.
3. Los costos más representativos son los de combustible,y por tanto la potencia entregada por el generador de diésel es la más costosa.
4. La operación de los sistemas de generación híbrida de potencia tiene un costo generalmente no lineal.

Costo de Generación Diésel

La GD presenta una función de costo asociada a la potencia del generador. Dicha función se encuentra dada por la Ecuación 5.162, en donde i es la i-ésima fuente de generación , P_i y F_i son la salida de potencia eléctrica y el costo de operación de la fuente i, respectivamente, mientras que α, β y γ son los cocientes de costo.

$$f(P_i) = \alpha_i + \beta_i P_i + \gamma P_i^2 \tag{5.162}$$

Para $\alpha = 14.88$, $\beta = 0.3$ y $\gamma = 0.000435$ según se plantea en [13]. Luego la función de costo correspondiente a la generación de diésel se define entonces como:

$$f_1(P_1) = 14.88 + 0.3P_1 + 0.000435P_1^2 \tag{5.163}$$

Costo de Generación Solar Fotovoltaica

Para el generador de energía fotovoltaica se define la siguiente función de costo de generación:

$$f_3(P_3) = \alpha I^p P_3 + G^E P_3 \tag{5.164}$$

$$\alpha = \frac{r}{[1 + (1 + r)^{-N}]} \tag{5.165}$$

En este caso P_3 es la potencia de salida de GFV, α es la tasa de retorno de inversión y r corresponde a la tasa de interés (se considera de 0.09 para el caso base), N es la vida útil del generador, que ha sido establecida en 20 años, I^p indica el costo de inversión por unidad de potencia ($5000=kW) y G^E es el costo de operación y mantenimiento por unidad de potencia generada. La función de costo por generación fotovoltaica queda definida entonces como:

$$f_3(P_3) = 545.016P_3 \tag{5.166}$$

Costo de Generación de Energía Eólica

Según lo planteado en [13] se toman los mismos valores de I^p y G^E, la función de costo para la generación eólica queda como:

$$f_4(P_4) = 152.616P_4 \tag{5.167}$$

Costo de Sistema de Almacenamiento de Energía

Para el SAE se considera un banco de baterías de 2kW. En [13], el costo I^p por unidad de almacenamiento instalada es de \$1000/kW, mientras que el G^E es igual a ¢1.6/kW y representa el costo de operación y mantenimiento por unidad de potencia generada. El uso del SAE describe la siguiente función de costo:

$$f_2(P_2) = 119P_2 \tag{5.168}$$

5.2.2. Planteamiento del Problema de Optimización

El despacho económico es un método eficiente para obtener un menor costo del sistema al tiempo que se satisface la carga o demanda de energía mediante el despacho de las fuentes de generación. Simultáneamente, las restricciones de las fuentes de generación deben ser satisfechas. En el presente caso de estudio se deben calcular las potencias de cada uno de los generadores en la MR-RNI en cada hora del día, minimizando el costo. El vector de variables de diseño se plantea en la Ecuación 5.169:

$$\vec{p} = [P_1, P_2, P_3, P_4]^T \tag{5.169}$$

La formulación clásica del despacho económico se plantea en [79] de la siguiente forma:

$$min \quad F = \sum_{i=1}^{NG} F_i P_i \tag{5.170}$$

Donde NG es el número de fuentes; P_i es el generador de fuente i; F_i es el costo total de producción de la fuente i. Luego la función objetivo considerada para el caso de estudio en cuestión tiene en cuenta la variabilidad de la carga en cada hora τ del día según la Ecuación 5.171

$$min \quad f(\vec{p}) = w_1 C_f f(P_1(\tau)) + w_2 f(P_2(\tau)) + w_3 f(P_3(\tau)) + w_4 f(P_4(\tau)) \tag{5.171}$$

Donde w_1,w_2,w_3,w_4 son los pesos asignados a los generadores, cumpliéndose que $w_1 + w_2 + w_3 + w_4 = 1$ y se establece que $w_1 = w_2 = w_3 = w_4 = 0.25$. C_f es el costo del combustible de diésel que se asignó en 1 USD.

Teniendo en cuenta que el costo de generación de energía debe ser calculado en cada hora del día la función objetivo para las 24 horas del día se define según la Ecuación 5.172:

$$min \quad F = \sum_{\tau=1}^{24} f(\vec{p}_\tau) \quad \vec{p}_\tau \in \mathbb{R}^4 \tag{5.172}$$

Restricciones de Diseño

La gestión de generación de energía en la MR-RNI está sujeta a las siguientes restricciones de diseño:

- *Balance de potencia*: La suma de las potencias suministradas por los cuatro generadores debe ser igual a la carga P_L del sistema :

$$P_1 + P_2 + P_3 + P_4 = P_L \tag{5.173}$$

- *Modelo del banco de baterías*: El poder de salida del generador PV y la carga demandada en cierta hora t, determinan el poder de carga y descarga dentro y fuera del banco de baterías. El estado de carga (SOC por sus siglas en inglés) del banco de baterías en cualquier hora, $SOC(t)$,depende del SOC en la hora previa $SOC(t-1)$. Luego, para el flujo de energía de la hora $t-1$ a t, se debe considerar la siguiente relación [80]:

$$SOC(t) = SOC(t-1) - \alpha_D P_2(t) + \alpha_C P_3(t) + \alpha_C P_4(t) \tag{5.174}$$

  Donde $\alpha_D = \frac{\eta_D}{Bc_{max}}$, $\alpha_C = \frac{\eta_C}{Bc_{max}}$ y η_D y η_D son las eficiencias de carga y descarga del banco de baterías respectivamente. Luego la expresión que describe la dinámica de la batería se define como [80]:

$$SOC(t-1) = SOC(0) - \alpha_D \sum_{\tau=1}^{t} P_2(\tau) + \alpha_C \sum_{\tau=1}^{t} P_3(\tau) + \alpha_C \sum_{\tau=1}^{t} P_4(\tau) \tag{5.175}$$

  donde se tiene que $SOC(0)$ como el SOC inicial de la batería y $\alpha_C \sum_{\tau=1}^{t} P_3(\tau) + \alpha_C \sum_{\tau=1}^{t} P_4(\tau)$ es el poder aceptado por la batería en la hora t y $\alpha_D \sum_{\tau=1}^{t} P_2(\tau)$ es el poder de descarga de la batería en el tiempo t. Además se debe tener en cuenta que la capacidad disponible de la batería no debe ser menor que la capacidad mínima permitida ni mayor que la capacidad máxima permitida [80], es decir:

$$SOC^{min} - SOC(t) \leq 0 \quad , \quad SOC(t) - SOC^{max} - \leq 0 \tag{5.176}$$

  Los parámetros del banco de baterías se muestran en la Tabla 5.3

Parámetros	%
Eficiencia de carga (η_C)	85
Eficiencia de descarga (η_D)	100
Estado máximo de carga (SOC_{max})	95
Estado mínimo de carga (SOC_{min})	40

TABLA 5.3: Parámetros del Sistemas de Almacenamiento de Energía en la Micro-Red No Interconectable.

Teniendo en cuenta las secciones anteriores, se plantea el problema de optimización mono-objetivo para el caso de estudio de la MR-RNI descrito por las Ecuaciones de la 5.177 a la 5.184:

$$min \quad F = \sum_{\tau=1}^{24} f(\vec{p}_\tau) \quad \vec{p}_\tau \in \mathbb{R}^4 \tag{5.177}$$

s.a:

$$g_1(\vec{p}_\tau) = P_1(\tau) + P_2(\tau) + P_3(\tau) + P_4(\tau) = P_L(\tau) \tag{5.178}$$
$$g_2(\vec{p}_\tau) = SOC^{min} - SOC(t) \leq 0 \tag{5.179}$$
$$g_3(\vec{p}_\tau) = SOC(t) - SOC^{max} - \leq 0 \tag{5.180}$$

La carga $P_L(\tau)$ fue obtenida de [81] y los límites superior e inferior de cada variable se encuentran definidos como:

$$0 \leq P_1(\tau) \leq GD_{nominal} \tag{5.181}$$
$$0 \leq P_2(\tau) \leq SOC(0) \times Bc_{max} - SOC^{min} \times Bc_{max} \tag{5.182}$$
$$0 \leq P_3(\tau) \leq P_{pv}(\tau) \tag{5.183}$$
$$0 \leq P_4(\tau) \leq P_{wind}(\tau) \tag{5.184}$$

donde $GD_{nominal}$ es la capacidad nominal del generador diesel, con un valor de 5000 VA (5 kVA) para cada hora del día en este caso de estudio. El límite superior para $P_2(\tau)$ depende de la capacidad máxima de la batería (Bc_{max}) que es de 2000 VA (2 kVA) y el estado inicial de la misma. Los datos de $P_{pv}(\tau)$ y $P_{wind}(\tau)$ fueron tomados de [79]. El estado de carga $SOC(0)$ de la batería se generó de forma aleatoria entre un rango de 40 % y 95 %. La potencia de P_3 es equivalente a siete módulos fotovoltaicos.

5.3. Conclusiones del capítulo

En el presente capítulo se plantea la formulación matemática de los problemas a resolver por la propuesta de solución. Es necesario primeramente destacar la complejidad de los diferentes casos de diseño cinemático de mecanismos, donde el análisis realizado para obtener la trayectoria deseada por el punto del acoplador, concibe una función compleja no lineal que incluye operaciones mediante funciones trigonométricas y de potencias sobre las variables de diseño. Se puede observar que la función objetivo de estos problemas puede tener valores imaginarios (Ecuación 5.4) para vectores reales. Al no estar definida en los reales para todo su dominio, los métodos de optimización basados en información del gradiente suelen no ser aplicables, surgiendo la necesidad de aplicar métodos directos. Además, las restricciones de diseño para garantizar el cumplimiento de las leyes de Grashoff y la secuencia de ángulos acotan el espacio factible aumentando la dificultad del problema.

A diferencia de los casos de diseño de mecanismos, la función objetivo a optimizar en el problema de la MR-RNI es una función cuadrática. Sin embargo, en este problema se presentan límites diferentes en cada hora del día para las variables de diseño. Para la potencia generada por el sistema de almacenamiento de energía, este límite depende del estado de la carga en la hora anterior lo que representa un reto para el algoritmo de optimización. Por otra parte, el balance de carga implica una restricción de igualdad, las cuales son generalmente más difíciles de satisfacer.

CAPÍTULO 6

Hibridación de Métodos de Programación Matemática con Algoritmos Evolutivos

En los Capítulos 3 y 4 se presentan dos esquemas importantes en cuanto al diseño de algoritmos en el área de la optimización, específicamente en la optimización de problemas con espacios de búsqueda continuos. En el primer esquema se tienen los algoritmos clásicos de programación matemática los cuales son eficientes buscadores locales, y en el segundo se encuentran algoritmos evolutivos, los cuales utilizan una población de individuos (soluciones candidatas) que les permite obtener información global sobre el espacio de búsqueda. Teniendo en cuenta estos dos esquemas, este capítulo describe la propuesta de solución diseñada durante la presente investigación.

La calidad del desempeño de un algoritmo de optimización depende en gran medida del balance entre las operaciones de exploración y explotación del espacio de búsqueda que sean concebidas en su diseño. Cuando los operadores del algoritmo están orientados a la explotación, este puede quedar estancado en mínimos locales o bien ser sensible al punto de inicio debido a que sólo se conoce una región limitada del espacio de posibles soluciones. Sin embargo, los algoritmos con estas características presentan una mayor velocidad de convergencia al mínimo para espacios de búsqueda convexos. Por otra parte, un algoritmo con operadores mayormente exploratorios presentará generalmente una convergencia lenta y tiempos de ejecución altos; pero no será sensible al punto de inicio y la probabilidad de quedar estancado en mínimos locales será menor.

Debido a la diversidad de problemas en el área de la optimización este balance es difícil de lograr. Algunos problemas requerirán mayor explotación en regiones prometedoras para agilizar la búsqueda, y en otros se necesitará información global del espacio de búsqueda (operaciones de exploración) para encontrar la solución óptima. Esta cuestión es abordada en los teoremas de No Free Lunch para la optimización, donde se plantea que si un algoritmo se desempeña particularmente bien en promedio para una clase de problemas, entonces debe hacerlo peor en promedio con respecto a los problemas restantes. En particular, si un algoritmo tiene mejor rendimiento que la búsqueda aleatoria en una clase de problemas, entonces debe tener un rendimiento peor que la búsqueda aleatoria en las clases de problemas restantes [82]. En este sentido la hibridación puede resultar un enfoque efectivo para diseño de algoritmos más eficientes y que puedan resolver eficazmente un amplio espectro de problemas.

En el contexto de las metaheurísticas, la hibridación se refiere principalmente al proceso de combinar o integrar las mejores características de dos o más algoritmos, para formar uno nuevo que supere las partes originales en la resolución de un problema específico o un conjunto de problemas de referencia general [83]. Dragoi en [84] considera tres clasificaciones de hibridación de acuerdo a diferentes aspectos.

1. Según los tipos de algoritmos que participan en la hibridación:
   a) Metaheurísticas con metaheurísticas.
   b) Metaheurísticas con algoritmos específicos del problema.
   c) Metaheurísticas con métodos de programación matemática o inteligencia artificial (que, a su vez, pueden ser: técnicas exactas u otras heurísticas)
2. Con respecto al nivel de hibridación, se encuentran dos casos:
   a) Acoplamiento débil de alto nivel (los algoritmos conservan sus propias identidades)
   b) Acoplamiento fuerte de bajo nivel (se intercambian componentes individuales)
3. Cuando se toma en cuenta el orden de ejecución, la hibridación puede ser:
   a) Secuencial.
   b) Intercalada.
   c) Paralela

De las siguientes clasificaciones se infiere la diversidad de investigaciones en el área de la hibridación. A continuación se presentan algunos trabajos de hibridación desde el enfoque que concibe una metaheurística como buscador global y

un método directo de programación matemática como buscador local. Específicamente se destacarán los trabajos previos que se presentan los dos algoritmos utilizados en la propuesta de solución la Evolución Diferencial (buscador global) y el método de Nelder Mead como buscador local.

6.1. Hibridación de la Evolución Diferencial con Métodos de Búsqueda Local

En el capítulo introductorio del presente trabajo se abordó el concepto de algoritmo memético (AM): un esquema que utiliza una metaheurística como algoritmo base de búsqueda global y que aplica métodos de búsqueda local para mejorar el rendimiento y refinar a los individuos en la población. Este tipo de hibridación es un enfoque integrador (coercitivo), ya que el algoritmo de búsqueda local se considera un subordinado y está incorporado en otro algoritmo [85]. Diversos estudios muestran que, para algunos problemas, los AM son más eficientes y más efectivos que los AE tradicionales [86]. Los trabajos de hibridación más destacados que combinan la ED y un método de programación matemática se encuentran dentro de este tipo de esquema y sirvieron como base para el enfoque propuesto.

En el artículo "Algoritmo Integrado de Estrategias de Evolución Diferencial con Búsqueda Local" (ISDE-L), se aplica un procedimiento de búsqueda local para mejorar el rendimiento [87]. En en cada k generaciones, se ordena la población según su valor de aptitud y se selecciona un individuo al azar del 25 % de la población. A este individuo se le aplica una técnica de búsqueda local con un número máximo de evaluaciones. La búsqueda local consiste en elegir aleatoriamente una variable del vector seleccionado, y luego se le suma o resta un número gaussiano aleatorio, como un tamaño de paso, tomándose la mejor dirección de acuerdo con la función de aptitud. Si el paso es exitoso y mejora el individuo seleccionado, entonces el valor se actualiza. Este proceso continúa hasta que se seleccionan todas las variables o se alcanza el número máximo de evaluaciones. El algoritmo ISDE-L es aplicado al conjunto de problemas de prueba presentados en congreso CEC 2010. Los resultados obtenidos fueron mejores o iguales a los alcanzados por algoritmos de vanguardia para estos problemas prueba.

Liao en [88], combina una versión de la ED modificada (MDE) con el método de Caminata Aleatoria con Dirección de Explotación (RWDE por sus siglas en inglés) como el operador de búsqueda local para mejorar los vectores x_{r_1},x_{r_2} y x_{r_3} en la mutación diferencial. RWDE es un método iterativo de optimización estocástica que genera una secuencia de aproximaciones al óptimo al asumir un vector aleatorio como una dirección de búsqueda. El algoritmo híbrido resultante es llamado MDE-LS (del inglés Modified Differential Evolution- Local Search) y es considerado por el autor como un algoritmo memético. Se seleccionan un

total de catorce problemas de diseño de ingeniería encontrados en literatura en diferentes campos de ingeniería. En siete de los catorce problemas, el algoritmo MDE-LS encuentra el óptimo global dentro del error 10E-6 en treinta ejecuciones. El algoritmo MDE-LS produce tasas de éxito más altas en 8 problemas pero menor en 2, en comparación con las del algoritmo MDE. En general, el MDE-LS mejora la tasa de éxito promedio en un 10 % sobre el algoritmo MDE.

En [89], se presenta una algoritmo memético que utiliza como buscador global la ED y dos algoritmos de búsqueda local, el Algoritmo Hooke-Jeeves (HJ) y el Buscador Local Estocástico (SLS por sus siglas en inglés). Los métodos de búsqueda local ayudan al marco evolutivo (ED) al ofrecer perspectivas exploratorias alternativas. El objetivo de ambos algoritmos es mejorar localmente una solución inicial explorando su vecindad. El espacio a explorar por los buscadores locales tiene un radio inicial. Luego, cuando no es posible realizar una mejora, se genera una vecindad del espacio más pequeña reduciendo el radio. Para ambos algoritmos de búsqueda local, se establece un presupuesto igual a 300 evaluaciones de la función de aptitud. Los resultados estadísticos muestran los MA presentados son competitivos con la ED y no requieren un ajuste de parámetros que podría terminar siendo costoso en términos de recursos de cómputo.

Rogalsky y Derksen en [90] proponen un algoritmo híbrido que combina el método el descenso simple (DS) con la ED para de acelerar la convergencia del algoritmo, sin que este quede atrapado en los mínimos locales. DS es un método de búsqueda local basado en simplex que utiliza operadores de reflexión, expansión y contracción. La versión híbrida, denotada HDE utiliza los tres mecanismos. De cada generación resultante de la aplicación de la ED, se eligen $n + 1$ individuos para formar un simplex. Esto puede realizarse de diferentes formas ya sea seleccionando los $n + 1$ mejores, peores o aleatoriamente. A través de los operadores de DS, el símplex se modifica hasta que uno (o varios) individuos se mejoran. Todos los vectores mejorados se eligen para pasar a la siguiente generación.

Wang en [91], combina la búsqueda realizada por DE con el metodo Nedler-Mead en un algoritmo híbrido que es aplicado a la identificación de parámetros de varios sistemas caóticos. En cada generación se realizan cuatro pasos principales: primero los P individuos de la población son ordenados de acuerdo su valor de aptitud. Luego, los primeros Q individuos se usan para calcular el centroide del simplex. En el tercer paso, se realiza por cada uno de los restantes P-Q individuos, una iteración del NM con el centroide calculado mediante los Q mejores individuos. Aquí, la información de los mejores individuos es utilizada para guiar la búsqueda local del algoritmo. Por lo tanto, se modifica la búsqueda clásica de NM simplex, que se utiliza para guiar la búsqueda hacia regiones prometedoras. Luego se aplica la ED a toda la población P. Las simulaciones numéricas basadas en varios sistemas caóticos típicos y las comparaciones con otros enfoques existentes demostraron la efectividad, eficiencia y robustez del algoritmo híbrido propuesto.

6.2. Enfoque propuesto

Como se puede observar en las investigaciones previas, la mayoría de los enfoques consisten en aplicar, bajo ciertas condiciones y con un número determinado de evaluaciones, un método de búsqueda local a determinados individuos de la población para mejorar su valor de aptitud. La evolución diferencial siempre se aplica sobre toda la población y en la mayoría de las propuestas los métodos de búsqueda local no conocen información global en el momento de su aplicación. Sólo el caso de [91] difiere en este sentido, ya que que el NM utiliza un centroide calculado con los mejores individuos. Sin embargo, esta propuesta realiza un número de evaluaciones por generación considerablemente mayor que la ED tradicional ya que se realizarán en cada generación $2(P - Q)$ evaluaciones de la función de aptitud debido a la aplicación de NM (el cual realiza dos evaluaciones en la versión sin operador de encogimiento) para luego realizar las P evaluaciones de la ED.

Teniendo en cuenta los trabajos de hibridación encontrados en la literatura especializada, la hipótesis y objetivos de la presente investigación, se diseña un nuevo enfoque de hibridación que pretende aumentar el equilibrio y la sinergia entre los buscadores global y local, así como la reducción del número de evaluaciones para disminuir la complejidad temporal. Los algoritmos híbridos presentados en próximas secciones fueron diseñados de acuerdo a los siguientes lineamientos de diseño:

1. Se distribuyen aleatoriamente K instancias de un buscador local que cubrirán y explotarán el espacio de búsqueda.
2. Una instancia de un buscador global se utiliza para coordinar y compartir información entre las instancias del buscador local.
3. Las instancias del buscador local pueden incorporar información global o histórica en sus operadores.
4. El buscador local se aplica a un subconjunto del total de individuos de la población en cada generación.
5. El buscador global se aplica a cierto subconjunto del total de la población en cada generación.
6. La información de todos individuos en la población es utilizada en cada generación.
7. El trabajo debe ser balanceado: el número de evaluaciones destinadas a los buscadores local y global no debe ser desproporcionado para garantizar un balance entre exploración y explotación.

La novedad del enfoque propuesto radica en la dinámica establecida entre ambos buscadores. A diferencia de los enfoques identificados en la literatura, este enfoque delega mayor participación al buscador local, aunque no existe subordinación de ningún algoritmo. Al distribuir diferentes instancias del buscador local se garantiza una mayor exploración del espacio de búsqueda en las primeras generaciones y una explotación exhaustiva de la región más prometedora en las generaciones finales.

El hecho de que los buscadores locales utilicen información global o histórica en sus operadores (mejor o peor individuo, individuos seleccionados aleatoriamente de la población, historial de mejores o peores individuos, etcétera), reduce la probabilidad de estancamiento o de convergencia prematura de las instancias. Por tanto, los buscadores locales deberán ser adecuadamente modificados para garantizar esta característica.

Por otra parte, tanto las instancias de los buscadores locales como el buscador global no se aplicarán a todos los individuos de la población sino un subconjunto de la misma, lo que reduce el número de evaluaciones realizadas en cada generación. Tanto la característica anterior como la directriz del trabajo balanceado, implica que el buscador local a utilizar debe realizar un número relativamente pequeño de evaluaciones por iteración en comparación con el buscador global. De lo contrario, el número de instancias del buscador local deberá ser reducido, lo implicaría explotar menos regiones del espacio de búsqueda.

En las siguientes secciones se describen las variantes híbridas que constituyen las propuestas de solución del presente trabajo de investigación. Estos algoritmos integran la Evolución Diferencial como buscador global y el método de Nelder-Mead buscador local bajo este enfoque de hibridación. El método NM fue modificado mediante el diseño experimental para lograr la hibridación. Nuevos operadores son agregados, los cuales conciben aleatoriedad e insertan información sobre regiones del espacio de búsqueda que se encuentran fuera de la vecindad en que trabaja la instancia el buscador local.

6.2.1. Método de Nelder Mead con Expansión de Longitud Aleatoria: NMELA

Se selecciona el método Nelder Mead para llevar a cabo la hibridación por tres razones principales. La primera se debe a la capacidad del Nelder Mead para recolectar información de un área local del espacio de búsqueda realizando pocas evaluaciones de la función objetivo. El nuevo simplex en la i-ésima iteración contiene sólo un nuevo vértice: el punto aceptado que reemplaza al peor vértice en el simplex anterior. Este punto es generado aparir del punto reflejado por el peor, por lo tanto en cada iteración el método Nelder Mead realiza dos evaluaciones

de la función objetivo. En el caso de las variantes planteadas a continuación, se realizan cuatro evaluaciones de la función objetivo como máximo por iteración.

La segunda razón es la complejidad temporal del método NM original. Debido al hecho de que, el simplex en la i-ésima iteración fue objeto de ordenamientos en iteraciones anteriores, la complejidad temporal del ordenamiento raras veces llega al peor caso. Por tanto, el orden de los vértices se puede realizar en tiempo lineal (a lo sumo n comparaciones) en un paso de ordenación de inserción por ejemplo. El nuevo centroide también se calcula actualizando el anterior en una operación de sumatoria con complejidad de $O(n)$, con poco espacio adicional de almacenamiento en memoria [92].

La tercera razón se debe a los resultados experimentales obtenidos por las versiones aleatorizadas descritas en el Capítulo 7, Sección 7.3.1. Al aumentar la capacidad de exploración mediante expansiones con longitud aleatoria (las cuales pueden llegar a ser más largas que la expansión original), se evidencia un aumento significativo del desempeño con respecto al método original para todos los problemas de diseño cinemático.

Con el objetivo de explorar cuáles serían los beneficios de agregar cierta diversidad en los movimientos del simplex se realizaron tres versiones del Nelder Mead original (Véase Algoritmo 1). Las versiones consisten en aplicar operadores basados en el operador de expansión original pero con longitudes aleatorias que permiten diversificar la búsqueda, expandiendo un simplex pequeño para lograr el escape de mínimos locales. Se realizaron modificaciones para el manejo de restricciones aplicando las reglas de Deb para comparar los puntos del simplex (Véase [58]), y una regla de acotamiento para las variables de diseño. El simplex inicial fue generado de forma aleatoria, y se utilizaron los parámetros γ y β de acuerdo a las características del problema. La condición de parada fue cambiada por el número de evaluaciones con la finalidad de medir el rendimiento.

Operador de Expansión con Longitud Aleatoria

El primer operador propuesto se aplica cuando ninguno de los operadores del método original mejoran, de acuerdo a las reglas de Deb, al peor punto x_h . Entonces, se realiza una expansión donde el factor γ es sustituido por dos números aleatorios para cada miembro del operador. Como coeficiente de x_c, se emplea un número aleatorio determinado dado por la ecuación $C = 1 + c_1(N + 1)$ donde $c_1 \in (0, 1)$ es un número aleatorio. Como coeficiente del punto x_h se emplea un número aleatorio $c_2 \in (0, 1)$. Se puede observar que el tamaño de la longitud de la expansión será proporcional a la dimensionalidad del problema. La Ecuación 6.1 refleja el operador de Expansión de Longitud Aleatoria (ELA). El método NM modificado con el operador ELA se describe en el Algoritmo 4. Donde NE es el número máximo de evaluaciones a realizar, E es el contador de evaluaciones, la

función $es\_mejor(x_1, x_2)$ retorna verdadero si el primer argumento es mejor que el segundo de acuerdo a las reglas de Deb.

$$x_{new} = (1 + c_1(N + 1))x_c - c_2 x_h \tag{6.1}$$

Operador de Expansión Inversa con Longitud Aleatoria

El segundo operador propuesto se aplica de forma similar al operador ELA. Cuando ninguno de los operadores del algoritmo original mejoran el peor punto se realiza una expansión con longitud aleatoria, sólo que en este caso se toma al mejor punto x_l para expandir en lugar de x_h.

El objetivo de realizar esta operación es hacer un movimiento drástico para escapar del estancamiento probando direcciones contrarias al mejor que se asume como mínimo local. En caso de que el punto encontrado no mejore al peor punto, será utilizado como peor en las iteraciones siguientes y mejorado con los operadores clásicos del método, ya sea por reflexión o expansión. En cualquier caso, el operador permite probar nuevas direcciones en el espacio de búsqueda y aumentar la diversidad del simplex, una vez que este se encuentra "cerrado" sobre un mínimo local. En la Ecuación 6.2 se describe el operador de Expansión Inversa de Longitud Aleatoria (EILA). El Algoritmo 5 describe la modificación del Nelder Mead original con el operador en cuestión.

$$x_{new} = (1 + c_1(N + 1))x_c - c_2 x_l \tag{6.2}$$

Combinación de los operadores

Una versión final del NM se realizó combinando los dos operadores. Al igual que en las versiones anteriores, si los operadores originales no logran mejorar el peor punto, primero se realiza el operador descrito en la Ecuación 6.1 , y si este no supera al peor punto se aplica el operador que refleja el mejor punto descrito en el Ecuación 6.2. Esta combinación de operadores se realiza con el propósito de probar más direcciones de búsqueda y alcanzar mayor capacidad de exploración en el método NM. El Algoritmo 6 describe la versión del método NM con dos Operadores de Expansión con Longitud Aleatoria (NM2ELA).

Algoritmo 4 Método de Nelder-Mead con Expansión de Longitud Aleatoria

Elegir parámetros: $\beta > 0, \gamma \in (0, 1)$
Generar un simplex inicial de forma aleatoria.
Evaluar simplex inicial
Hacer $E = N + 1$.
while $E < NE$ **do**
  Ordenar los puntos del simplex según $es\_mejor(x_1, x_2)$.
  Elegir x_h (peor punto), x_l (mejor punto) y x_g (el segundo peor punto).
  Calcular $x_c = \frac{1}{N} \sum_{i=1, i \neq h}^{N+1} x_i$
  Realizar la reflexión $x_r = 2x_c - x_h$
  Acotar x_r
  Evaluar $f(x_r)$
  Hacer $E = E + 1$
  if $es\_mejor(x_r, x_l)$ **then**
    Hacer $x_{new} = (1 + \gamma)x_c - \gamma x_h$ (Expansión)
  else
    if $es\_mejor(x_h, x_r)$ **then**
      Hacer $x_{new} = (1 - \beta)x_c + \beta x_h$ (Contracción adentro)
    end if
  else
    if $es\_mejor(x_r, x_g)$ **and** $es\_mejor(x_h, x_r)$ **then**
      Hacer $x_{new} = (1 + \beta)x_c - \beta x_h$ (Contracción afuera)
    end if
  end if
  Acotar x_{new}
  Evaluar $f(x_{new})$
  Hacer $E = E + 1$
  if $es\_mejor(x_h, x_{new})$ **then**
    Calcular $x_{new} = (1 + c_1(N + 1))x_c - c_2(x_h)$ (ELA)
    Acotar x_{new}
    Evaluar $f(x_{new})$
    Hacer $E = E + 1$
  end if
end while

Algoritmo 5 Método de Nelder-Mead con Expansión Inversa de Longitud Aleatoria

Elegir parámetros: $\beta > 0, \gamma \in (0,1)$
Generar un simplex inicial de forma aleatoria.
Evaluar simplex inicial
Hacer $E = N + 1$.
while $I < NE$ **do**
  Ordenar los puntos del simplex.
  Elegir x_h (peor punto), x_l (mejor punto) y x_g (el segundo peor punto).
  Calcular $x_c = \frac{1}{N} \sum_{i=1, i \neq h}^{N+1} x_i$
  Realizar la reflexión $x_r = 2x_c - x_h$
  if $es\_mejor(x_r, x_l)$ **then**
    Hacer $x_{new} = (1 + \gamma)x_c - \gamma x_h$ (Expansión)
  else
    if $es\_mejor(x_h, x_r)$ **then**
      Hacer $x_{new} = (1 - \beta)x_c + \beta x_h$ (Contracción adentro)
    end if
  else
    if $es\_mejor(x_r, x_g)$ **and** $es\_mejor(x_h, x_r)$ **then**
      Hacer $x_{new} = (1 + \beta)x_c - \beta x_h$ (Contracción afuera)
    end if
  end if
  Acotar x_{new}
  Evaluar $f(x_{new})$
  Hacer $E = E + 1$
  if $es\_mejor(x_h, x_{new})$ **then**
    Calcular $x_{new} = (1 + c_1(N + 1))x_c - c_2(x_l)$ (EILA)
    Acotar x_{new}
    Evaluar $f(x_{new})$
    Hacer $E = E + 1$
  end if
end while

Algoritmo 6 Método de Nelder-Mead con dos operadores de Expansión con Longitud Aleatoria

Elegir parámetros: $\beta > 0, \gamma \in (0,1)$.
Generar un simplex inicial de forma aleatoria.
Evaluar el simplex inicial
Hacer $E = N + 1$
while $E < NE$ **do**
  Ordenar los puntos de acuerdo a las reglas de Deb.
  Elegir x_h (peor punto), x_l (mejor punto) y x_g (el segundo peor punto).
  Calcular $x_c = \frac{1}{N} \sum_{i=1, i \neq h}^{N+1} x_i$
  Realizar la reflexión $x_r = 2x_c - x_h$
  if $es\_mejor(x_r, x_l)$ **then**
    Hacer $x_{new} = (1 + \gamma)x_c - \gamma x_h$ (Expansión)
  else
    if $es\_mejor(x_h, x_r)$ **then**
      Hacer $x_{new} = (1 - \beta)x_c + \beta x_h$ (Contracción adentro)
    end if
  else
    if $es\_mejor(x_r, x_g)$ **and** $es\_mejor(x_h, x_r)$ **then**
      Hacer $x_{new} = (1 + \beta)x_c - \beta x_h$ (Contracción afuera)
    end if
  end if
  if $es\_mejor(x_h, x_{new})$ **then**
    Calcular $x_{new} = (1 + c_1(N+1))x_c - c_2(x_h)$ (ELA)
  end if
  Acotar x_{new}
  Evaluar $f(x_{new})$
  Evaluar $E = E + 1$
  if $es\_mejor(x_h, x_{new})$ **then**
    Calcular $x_{new} = (1 + c_1(N+1))x_c - c_2(x_l)$ (EILA)
    Acotar x_{new}
    Evaluar $f(x_{new})$
    Evaluar $E = E + 1$
  end if
end while

6.2.2. Observaciones Preliminares

Los resultados obtenidos durante el diseño experimental (Véase Sección 7.3.1) muestran tanto potencialidades como desventajas del NM. Aumentando la probabilidad de expansiones más largas tanto por la reflexión del peor punto como por la reflexión del mejor, se logra aumentar la eficiencia del método superando para MCE2, los resultados obtenidos por la ED (2.6280E-03) en el mejor punto con **3.8126E-04** para NM2ELA. Es importante destacar que el nuevo mínimo esta dado por un vector distante al encontrado por la ED y representa un mecanismo completamente diferente, lo que indica un aumento en la capacidad de exploración del método.

Otra de las bondades de estos operadores es que aumentan la robustez (sobre todo para baja dimensionalidad), con respecto al NM clásico. Lo queda demostrado en las medidas de tendencia central y de dispersión, siempre superiores al método original. El número de evaluaciones es significativamente menor al utilizado por la ED pues debido a la rápida convergencia del NM no se encontraron diferencias de desempeño al realizar mayor cantidad de evaluaciones.

Además, las pruebas de Friedman y Bonferroni (Véase Figura 7.1), establecen que las variantes propuestas presentan un desempeño significativamente superior con respecto a dos versiones deterministas del NM, para los 5 primeros problemas de optimización correspondientes al diseño cinemático.

De forma general, se puede observar que la dimensionalidad del espacio vectorial impacta significativamente en el desempeño del método, característica que ya ha sido reportada en trabajos previos y que se confirma en el presente [93]. Para los problemas con $N > 10$ el algoritmo muestra un comportamiento más errático en los resultados de sus ejecuciones, las cuales comienzan a diferir unas de otras.

Las variantes aleatorizadas pueden encontrar valores cercanos, iguales o mejores a los mínimos ya reportados, a diferencia de las versiones deterministas. Sin embargo, las medidas de tendencia central son distantes a las obtenidas por las variantes de la ED, sobre todo para los problemas de mayor dimensión. Por tanto, se evidencia que a pesar de las mejoras en el rendimiento, las versiones aleatorizadas aún son sensibles al punto de inicio (simplex inicial) y continúan siendo métodos de búsqueda local. Esta deficiencia puede ser atacada mediante un enfoque poblacional, de acuerdo a los lineamientos propuestos en la Sección 6.2 para lograr un mayor cubrimiento del espacio de búsqueda en las iteraciones tempranas.

6.2.3. Algoritmo Híbrido basado en el Método Nelder Mead con Evolución Diferencial: HNMED. Variante I

Las etapas iniciales del diseño experimental resultaron en el diseño e implementación del Algoritmo Híbrido basado en el Método Nelder Mead con Evolución Diferencial: HNMED. El procedimiento general consiste en dividir una población X generada de forma aleatoria en exactamente NS símpleces o sub-poblaciones de tamaño $N+1$. En la generación G, cada instancia $k = \{1, 2, ..., NS\}$ del NM realizará una búsqueda local utilizando el simplex S_k. Luego, se aplica la ED utilizando como individuo padre el mejor punto x_l^k del simplex S_k. Los individuos seleccionados aleatoriamente para realizar la mutación diferencial (x_{r_1},x_{r_2},x_{r_1}), se eligen de toda la población X. Se selecciona como buscador global la ED/rand/1/bin debido a que el operador sólo utiliza información global para realizar la mutación diferencial. En la primera variante se utiliza como buscador local la versión aleatorizada del NM con el operador NMEILA descrito en la Sección 6.2.1.

El diseño se basa en el enfoque de hibridación planteado al inicio del presente capítulo. Al dividir la población en varios símpleces, se garantiza una explotación de diferentes regiones del espacio de búsqueda mediante varias instancias del buscador local. El método NM resulta una opción adecuada ya que al utilizar el simplex para realizar sus operaciones, es capaz recolectar información de un área limitada del espacio del búsqueda mediante el cálculo del centroide. Por tanto, para cada ejecución de una instancia del NM se utiliza información de todo el subconjunto de individuos a su disposición pero sólo se opera sobre el peor y se realizan tres evaluaciones de la función objetivo. Esto implica que en cada generación se realizarán $3NS$ evaluaciones en total, por parte de los buscadores locales.

La ED también opera siguiendo las reglas del enfoque propuesto. En cada generación, la ED se aplica a una sub-población de tamaño NS integrada por los mejores individuos de cada simplex: $X_l = \{x_l^1, x_l^2, ..., x_l^{NS}\}$. Sin embargo, para realizar la mutación diferencial se utiliza información de la población, seleccionando aleatoriamente los tres individuos para generar el vector ruido de toda la población. De esta forma, la ED realiza NS evaluaciones de la función objetivo por cada generación.

En resumen, en una generación de HNMED se realizan un total $3NS + NS = 4NS$ evaluaciones entre las NS instancias del NM y la ED/rand/1/bin. La proporción entre las evaluaciones realizadas por el buscador local y global se encuentra dada por la relación $\frac{E_{ED}}{E_{NM}} = \frac{1}{4}$. El buscador local tiene mayor carga ya que constituye el método base que garantizará mayor rapidez. Sin embargo, su presupuesto de evaluaciones por generación no es excesivamente superior, de modo que se cumple el lineamiento del trabajo balanceado.

Teniendo en cuenta que el tamaño de la población es $NP = (N+1)NS$, y que un algoritmo ED convencional realiza NP evaluaciones por generación, la proporción entre las cantidad evaluaciones es:

$$\frac{E_{HNMED}}{E_{ED}} = \frac{4NS}{(N+1)NS} = \frac{4}{N+1} \tag{6.3}$$

Nótese que, a medida que aumenta la dimesionalidad, el número de evaluaciones en cada generación realizadas por HNMED representarán un porciento cada vez menor con respecto a las realizadas por la ED. Por ejemplo, para el problema MCE2 ($N = 6$) las evaluaciones por iteración de HNMED representan el 57 % ($\frac{4}{6+1} = 0.57$) de las realizadas por la ED; para los casos GCE1 y GCE2 con $N = 14$, el 26.7 %($\frac{4}{14+1} = 0.267$); para el problema MCE1 el 25 % ($\frac{4}{15+1} = 0.25$), el 20 % ($\frac{4}{19+1} = 0.20$) para MCE3 y así sucesivamente.

La Figura 6.1 describe el esquema general de las variantes híbridas propuestas. Se pueden observar los diferentes símpleces que cubren diferentes áreas del espacio de búsqueda. Los mejores puntos de cada simplex son mejorados por la ED con información global acercándolos a regiones prometedoras en las primeras generaciones y reuniéndolos sobre el mínimo global en las iteraciones finales.

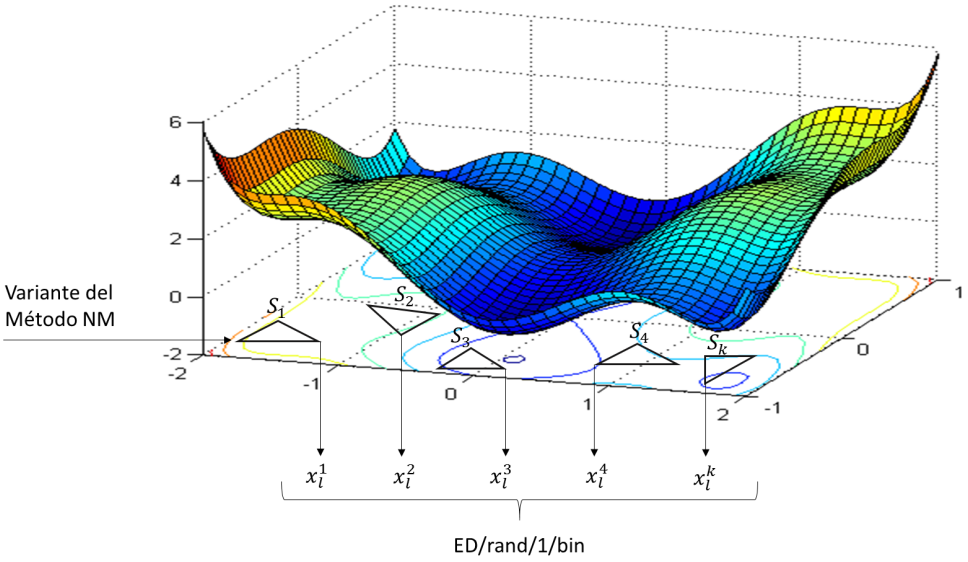


FIGURA 6.1: Esquema de las variantes HNMED.

El Algoritmo 7 describe el pseudocódigo para las variantes HNMED. Las flechas rojas en las lineas desde la 26 a la 28 indican el fragmento de código que varía en cada una de las versiones. Donde NE es el número máximo de evaluaciones, NP es el tamaño de la población e $I_{x_l^k}$ el índice correspondiente al punto x_l^k de S_k en la población X. En las siguientes secciones se presentan cuatro variantes híbridas

bajo el mismo enfoque, las cuales sólo difieren en el operador con aleatoriedad que incluye el método NM. Por tanto, solo se describirán dichos operadores; para mayor detalle sobre la implementación de estas variantes consultar el Anexo A.

HNMED. Variantes II y III

La segunda variante difiere en la versión del NM utilizada. Si los operadores del Nelder Mead no mejoran al peor vector este se sustituye por un vector que tiene como base el mejor vector local x_l^k y va hacia la dirección del mejor individuo en la población *best* según describe la Ecuación 6.4.

$$x_{new} = x_l^k + F(best - x_h^k) \tag{6.4}$$

Este operador está basado en el cálculo del vector ruido de la mutación diferencial, nótese que F es el mismo factor de escala utilizado en la mutación de la ED. En este caso, las instancias del Nelder Mead tienen un operador que utiliza información global: el mejor individuo en la población. El objetivo aquí es acercar el peor punto en el simplex S_k al mejor punto a nivel global de forma paulatina.

La tercera variante procede de la siguiente forma: en dependencia de cual dirección sea mejor, el vector ruido se dirigirá hacia el mejor punto a nivel global (Ecuación 6.4) o se alejará del peor a nivel local con un tamaño de paso proporcional a la diferencia absoluta entre *best* y x_h^k. Esta versión contiene el operador de la Variante II y agrega un segundo operador (Ecuación 6.5) que calcula un vector que tiene como base al mejor individuo local x_l^k y que va en sentido contrario al peor vector local x_h^k.

$$x_{new} = x_l^k - F(x_h^k - best) \tag{6.5}$$

De forma similar a la estrategia seguida en NM2ELA el objetivo de utilizar dos operadores que presentan direcciones opuestas es aumentar la capacidad de exploración del buscador local y aumentar la diversidad en simplex para evitar convergencia prematura o estancamiento de las diferentes instancias.

Algoritmo 7 Algoritmo HNMED

Elegir parámetros: $\beta > 0, \gamma \in (0,1)$
Generar y evaluar una población aleatoria X inicial.
Hacer $E = NP$.
while $E < NE$ **do**
  Generar $F = (0.3, 0.9)$ y $CR = (0, 0.8)$
  for $k \in \{1, \ldots, NS\}$ **do**
    Seleccionar S_k de X
    Ordenar los puntos en S_k de acuerdo a las reglas de Deb.
    Elegir x_h^k, x_l^k y x_g^k.
    Calcular $x_c^k = \frac{1}{N} \sum_{i=1, i \neq h}^{N+1} x_i^k$
    Realizar la reflexión $x_r^k = 2x_c^k - x_h^k$
    Acotar x_r^k y evaluar $f(x_r^k)$
    if $es\_mejor(x_r^k, x_l^k)$ **then**
      Hacer $x_{new} = (1+\gamma)x_c^k - \gamma x_h^k$ (Expansión)
    else
      if $es\_mejor(x_h^k, x_r^k)$ **then**
        Hacer $x_{new}^k = (1-\beta)x_c^k + \beta x_h^k$ (Contracción adentro)
      end if
    else
      if $es\_mejor(x_r^k, x_g^k)$ **and** $es\_mejor(x_h^k, x_r^k)$ **then**
        Hacer $x_{new} = (1+\beta)x_c^k - \beta x_h^k$ (Contracción afuera)
      end if
    end if
    Acotar x_{new}^k y evaluar $f(x_{new}^k)$
    if $es\_mejor(x_h, x_{new})$ **then**
      $\Rightarrow$ Calcular x_{new}^k según la variante
      $\Rightarrow$ Acotar x_{new}^k y evaluar $f(x_{new}^k)$
      $\Rightarrow$ Hacer $E = E + 1$
    end if
    Seleccionar aleatoriamente índices $r_1 \neq r_2 \neq r_3 \neq I_{x_l^k}$
    Generar aleatoriamente el índice $randb(j) = (1, N)$
    for $j \in \{1, \ldots, N\}$ **do**
      if $(randb(j) \leq CR)$ **or** $(j == randb(j))$ **then**
        $u_j^k = x_{r_1} + F(x_{r_2} - x_{r_3})$
      else
        $u_j^k = x_{l_j}^k$
      end if
    end for
    Acotar u^k y evaluar $f(u^k)$
    if es_mejor$(f(u^k), x_l^k)$ **then**
      $x_l^k = u^k$
    end if
    Hacer $E = E + 3$
  end for
end while

HNMED. Variantes IV y V

Las últimas dos variantes están basadas en los operadores utilizados por la versión DE/current-to-best/1 para calcular el vector ruido. El propósito de estas variantes es evitar las dos evaluaciones de la variante III, incorporando más información global en el operador. Los vectores x_{r_1},x_{r_2} y x_{r_3} son seleccionado aleatoriamente de toda la población y cumplen la condición de $I_{x_l^k} \neq r_1 \neq r_2 \neq r_3$. Nótese que en el tercer miembro de la Ecuación 6.6 el factor de escala es $1 - F$ el cual es el complemento de la probabilidad de F. Esta modificación se aplica con el objetivo de mantener un balance entre el miembro elitista y el aleatorio en el operador.

$$x_{new} = x_l^k + F(best - x_{r_1}) + (1 - F)(x_{r_2} - x_{r_3}) \tag{6.6}$$

Los operadores propuestos hasta el momento utilizan como vector base un punto local (x_l^k). En la quinta variante, el operador utiliza como vector base uno de los vectores elegidos aleatoriamente, una vez más en la búsqueda de aumentar la exploración del espacio de búsqueda y mantener la diversidad de la población. El operador utilizado en la variante V se describe en la ecuación 6.7

$$x_{new} = x_{r_1} + F(best - x_{r_1}) + (1 - F)(x_{r_2} - x_{r_3}) \tag{6.7}$$

Diferentes variantes y combinaciones de estos operadores fueron implementadas. No obstante, se presentan las propuestas que mostraron un mejor rendimiento promedio para los problemas a resolver en la presente investigación.

HNMED. Variante VI

En el diseño inicial de las variantes HNMED se concibió la inicialización de la población de forma aleatoria de acuerdo al procedimiento del enfoque evolutivo. Sin embargo, se pasa por alto la naturaleza geométrica de los operadores utilizados por el Nelder Mead. En [41], Spendley plantea que un simplex inicial regular puede aumentar el desempeño de un método basado en simplex y plantea las fórmulas para generar un simplex regular de longitud d tomando como centro del simplex un punto c determinado. Por otra parte, Han y Neumann en [93] comprueban que la tasa de convergencia ρ del NM tiende a 1 a medida que aumenta respecto al número de variables n, esto es $\lim_{n\to\infty} \rho(S_0, n) = 1$. Incluso,

los resultados experimentales muestran que para $\rho(S_0, 32) = 0.9912$, lo que indica que este efecto sucede rápidamente. Además, algunas consideraciones sobre el comportamiento del método NM original son presentadas: las operaciones de expansión y contracción producen una mayor reducción de la función que las operaciones de reflexión, y un simplex inicial de bordes más largos contribuye a una mejor minimización de la función objetivo.

Teniendo en cuenta lo planteado en los párrafos anteriores y los resultados obtenidos durante la experimentación con las primeras variantes, se diseña un última versión (Véase Algoritmo 8) que tiene como objetivo alcanzar una mayor capacidad de exploración en problemas de mayor dimensión. Dos modificaciones principales son llevadas a cabo respecto a las versiones anteriores:

1. Se reemplaza la inicialización aleatoria de la población por el siguiente procedimiento: cada punto x_i con $i = \{1, 2, ..., N+1\}$ en el simplex S_k se genera de acuerdo a la siguiente formula:

$$x_j^i = \begin{cases} \frac{u_j - l_j}{2} + p_j(\frac{u_j - l_j}{2}), & \text{si } i == j. \\ u_j + q_k(\frac{u_j - l_j}{2}) + r_j, & \text{en caso contrario.} \end{cases} \tag{6.8}$$

   Donde u_j y l_j son los limites superior e inferior en la dimensión j. Los valores de p_j, q_k y r_j son números aleatorios de una distribución uniforme en los intervalos siguientes:

$$p_j \in \begin{cases} (0.05; 5n/100), & \text{si } 5n/100 < 0.95. \\ (0.05; 0.95), & \text{en caso contrario.} \end{cases} \tag{6.9}$$

$$q_k \in \begin{cases} (0.5; 3n/100), & \text{si } 3n/100 < 0.95. \\ (0.05; 0.95), & \text{en caso contrario.} \end{cases} \tag{6.10}$$

$$r_j \in (-NP/100, +NP/100) \tag{6.11}$$

   La Ecuación 6.8 esta basada en la presentada por Spendley en [41] para la generación de un simplex regular de longitud d, la cual fue concebida para un algoritmo que utiliza un solo simplex y no contempla la diversidad en las dimensiones requerida por la parte evolutiva para un desempeño correcto. Aquí se garantiza una distancia mínima entre los vértices del simplex los cuales se ubicarán en vecindades cercanas a los límites del espacio de búsqueda. Por ejemplo, el punto x_1 del simplex S_k estará desplazado en su primera componente hacia región de longitud $(\frac{5n}{100} - 0.05)\frac{u_1 - l_1}{2}$ en la vecindad del límite superior de la primera dimensión. El punto x_2 desplazado hacia el límite superior de la segunda dimensión y así sucesivamente. Finalmente, el punto x_{n+1} se ubicará en la vecindad $(\frac{3n}{100} - 0.05)\frac{u_j - l_j}{2}$ de los

límites inferiores en todas las dimensiones. El valor aleatorio r_j se utiliza para garantizar una variación de los valores en la dimensión j para el simplex S_k. El objetivo de esta inicialización de los símpleces es garantizar tetraedros más regulares y de mayor tamaño.

2. Las versiones iniciales de HNMED tenían un enfoque elitista. En el caso de HNMED-V4 y HNMED-V5 se utiliza la dirección del mejor para mejorar el peor punto de simplex y la ED sólo se aplicaba al los mejores puntos de cada simplex. Para garantizar que mayor cantidad de individuos reciban información global, los operadores son aplicados a una posición del simplex en cada generación ascendentemente. Esto es, que para la generación 1 la ED se aplica al individuo en la posición 1 en cada simplex, en la segunda generación al individuo en la posición 2 y así sucesivamente hasta llegar a la posición $NS + 1$ donde se comienza de nuevo por la primera posición.

3. El operador de mutación diferencial utilizado es basado en DE/current-to-best/1:
$$v_m = x_m + F(x_l^r - x_{r_1}) + (1 - F)(x_{r_2} - x_{r_3}) \tag{6.12}$$
Donde el individuo elitista x_l^r es seleccionado aleatoriamente del conjunto de los mejores individuos de cada simplex X_l.

6.3. Conclusiones del capítulo

En el presente capítulo se propone un nuevo enfoque de hibridación que tiene como objetivo aumentar la eficiencia de la búsqueda usando como base un buscador local. Mediante la distribución de varias instancias en el espacio de búsqueda, el buscador local se potencia con la capacidad de exploración. El buscador global se utiliza como mecanismo de coordinación entre las diferentes instancias del método de búsqueda local. El número de evaluaciones destinadas al buscador local y global no debe ser desproporcionada para garantizar un balance entre la explotación y la exploración. Además, se plantea la idea de que, aunque cada buscador podrá informarse del resto de la población, sólo se aplicarán a un subconjunto de ella por generación, lo que implica una reducción considerable del número de evaluaciones.

Bajo las directrices de este enfoque se diseñaron e implementaron seis variantes híbridas. Se utilizó como buscador local variantes del método Nelder Mead con operadores que incluyen aleatoriedad e información global. El método NM fue seleccionado por sus características en cuanto a número de evaluaciones, complejidad temporal y los resultados alcanzados en el diseño experimental de las versiones con Expansión de Longitud Aleatoria. La ED/rand/1/bin se selecciona por utilizar vectores seleccionados aleatoriamente de toda la población para la creación del vector ruido por lo que su búsqueda brinda mayor información

global. El número de evaluaciones por iteración en comparación con la ED, se reduce a medida que aumenta la dimensionalidad del problema en proporción de $\frac{4}{N+1}$.

Una última versión es diseñada de acuerdo a los resultados del diseño experimental. En la versión seis, se tiene en cuenta la naturaleza geométrica del método Nelder Mead. La generación inicial de los símpleces de forma aleatoria, implicaba la irregularidad y el traslape de estos. Para controlar esta deficiencia se aplica una estrategia de inicialización de los símpleces que garantiza figuras de mayor tamaño y regularidad.

Algoritmo 8 HNMED. Variante IV

Elegir parámetros: $\beta > 0, \gamma \in (0,1)$
$\Rightarrow$ Generar símpleces según Equación 6.8 y evaluar X.
Hacer $E = NP$.
while $E < NE$ **do**
  Generar $F = (0.3, 0.9)$ y $CR = (0, 0.8)$
  for $k \in \{1, \ldots, NS\}$ **do**
    Seleccionar S_k de X
    Ordenar los puntos en S_k de acuerdo a las reglas de Deb.
    Elegir x_h^k, x_l^k y x_g^k.
    Calcular $x_c^k = \frac{1}{N} \sum_{i=1, i \neq h}^{N+1} x_i^k$
    Realizar la reflexión $x_r^k = 2x_c^k - x_h^k$
    Acotar x_r^k y evaluar $f(x_r^k)$
    if $es\_mejor(x_r^k, x_l^k)$ **then**
      Hacer $x_{new} = (1+\gamma)x_c^k - \gamma x_h^k$ (Expansión)
    else
      if $es\_mejor(x_h^k, x_r^k)$ **then**
        Hacer $x_{new}^k = (1-\beta)x_c^k + \beta x_h^k$ (Contracción adentro)
      end if
    else
      if $es\_mejor(x_r^k, x_g^k)$ **and** $es\_mejor(x_h^k, x_r^k)$ **then**
        Hacer $x_{new} = (1+\beta)x_c^k - \beta x_h^k$ (Contracción afuera)
      end if
    end if
    Acotar x_{new}^k y evaluar $f(x_{new}^k)$
    if $es\_mejor(x_h, x_{new})$ **then**
      $\Rightarrow$ Calcular $x_{new} = x_{r_1} + F(best - x_{r_1}) + (1-F)(x_{r_2} - x_{r_3})$
      Acotar x_{new}^k y evaluar $f(x_{new}^k)$
      Hacer $E = E + 1$
    end if
    $\Rightarrow$ Seleccionar aleatoriamente índices r =(1,N+1) y $r_1 \neq r_2 \neq r_3 \neq I_{x_l^k}$
    Generar aleatoriamente el índice $randb(j) = (1, N)$
    for $j \in \{1, \ldots, N\}$ **do**
      if $(randb(j) \leq CR)$ **or** $(j == randb(j))$ **then**
        $\Rightarrow u_j^k = x_{mj}^k + F(x_l^r - x_{r_1}) + (1-F)(x_{r_2} - x_{r_3})$
      else
        $\Rightarrow u_j^k = x_{mj}^k$
      end if
    end for
    Acotar u^k y evaluar $f(u^k)$
    if es_mejor$(f(u^k), x_l^k)$ **then**
      $\Rightarrow x_m^k = u^k$
    end if
    Hacer $E = E + 3$
  end for
  $\Rightarrow$ Hacer $m = m + 1$
end while

CAPÍTULO 7

Experimentos y Resultados

La investigación de algoritmos metaheurísticos y especialmente la computación evolutiva (CE) se basa principalmente en dos pilares: la teoría y la práctica. Sin embargo, la mayor parte del trabajo publicado en esta área se dedica a la aplicación de la CE y los métodos relacionados a problemas reales o de referencia. En la teoría de análisis de algoritmos la medición del desempeño de un método determinado se enfoca fundamentalmente en conocer cómo este funciona para el peor caso posible. Por otra parte, las investigaciones experimentales rara vez se preocupan por los peores casos, sobre todo si el problema tiene un trasfondo del mundo real, donde la observación del peor caso puede ser poco probable. Por lo tanto, el mayor interés se centra en el promedio de cierta cantidad los casos.

Los trabajos experimentales han tenido dos motivaciones principales: resolver problemas del mundo real o al menos demostrar que pueden ser resueltos mediante un algoritmo evolutivo y demostrar la eficacia de un algoritmo (preferiblemente nuevo) en una clase de problemas. La realización de experimentos en ciencias de la computación puede requerir las siguientes tareas [94]:

1. Encontrar los mejores parámetros o algoritmos dados k conjuntos de números aleatorios que representan el resultado de ciertos experimentos.
2. Encontrar la mejor asignación para un conjunto de variables reales (dentro de un rango dado) que representan los parámetros del algoritmo para una clase de problema.

3. Dadas las posibles formas de modificar un algoritmo A (por ejemplo, mediante el uso de operadores adicionales) encontrar la mejor combinación para una clase de problema.

El presente capítulo se enfoca fundamentalmente en esta última tarea. Sin embargo, las muestras que se pueden obtener para realizar las comparaciones en un número determinado de aspectos como: operadores, problemas de optimización, parámetros, número de evaluaciones entre otros; no permiten necesariamente obtener conclusiones generales sobre los algoritmos comparados. Por tanto, la pregunta de investigación más adecuada es bajo qué condiciones un algoritmo A es mejor que A' y por qué.

En este caso, se trata de comparar los algoritmos propuestos con aquellos que mejores resultados han obtenido en el conjunto específico de problemas planteados. Para realizar estas comparaciones se realizarán mediciones que permitirán además describir el comportamiento de los algoritmos propuestos e implementar ciertas mejoras. El proceso de experimentación debe ser guiado por una metodología que estructure el diseño experimental y garantice su rigurosidad.

7.1. Diseño Experimental

El diseño experimental de la propuesta de solución se estructuran de acuerdo a la metodología planteada a continuación. El objetivo de especificar una metodología es organizar el proceso de experimentación y garantizar la calidad de los resultados. Basado en lo presentado en [94], se plantea una metodología que contempla los siguientes pasos:

1. **Planteamiento objetivos del experimento**: para cada tipo experimento se debe indicar si se realizan para:
   a) Demostrar el rendimiento de un algoritmo.
   b) Verificar la robustez de un algoritmo en varias instancias de problema.
   c) Comparar dos (o varios) algoritmos conocidos.
   d) Explicar o comprender un comportamiento observado y detectar novedades.
2. **Definición de medidas y pruebas de desempeño**: Con base en los objetivos del experimento se debe definir cómo realizar las mediciones:
   a) El rendimiento será medido en términos del número de evaluaciones de la función objetivo.

b) Para medir la robustez del algoritmo se utilizará la estadística descriptiva: medidas de tendencia central y de dispersión de una muestra de tamaño n ejecuciones del algoritmo A.

c) La comparación de los algoritmos se realizará mediante pruebas de estadística inferencial: Pruebas de Friedman, Bonferroni-Dunn, prueba no paramétrica de suma de rangos de Wilcoxon.

d) Se utilizarán medidas como tasa de éxito de los operadores, medidas de diversidad y gráficas de convergencia para explicar o comprender comportamientos y detectar novedades.

3. **Planificación pre-experimental y configuración**: Este paso abarca la selección del número máximo permitido de evaluaciones de la función objetivo, el conjunto de problemas a resolver, número de ejecuciones para obtener las muestras y número de comparaciones a realizar. Se debe implementar y describir todo lo necesario para replicar el experimento. Esto consiste en la implementación y documentación de los algoritmos, problemas, parámetros y condiciones externas importantes como detalles del hardware empleado.
4. **Realización de los experimentos**: Los experimentos deberán ser realizados de forma automática. Las implementaciones deberán estar debidamente comentadas y los resultados deben ser presentados de forma organizada.
5. **Resultados y Visualización** : Se resumen y presentan los resultados numéricos, pruebas de hipótesis, así como las tablas o figuras obtenidas a partir de los experimentos.
6. **Observaciones**: Reportar el comportamiento inesperado o notable detectado mediante la revisión de los resultados, sin interpretación. Se debe discutir la relevancia estadística. Si las hipótesis estadísticas son aceptadas o rechazadas. Discutir el significado científico de lo observado: explicaciones de los resultados del experimento en contexto con otros hallazgos científicos. Este paso está destinado a contener declaraciones subjetivas que pueden conducir a nuevas hipótesis o preguntas de investigación basadas en los resultados de los experimentos actuales.

7.2. Estadística inferencial: Pruebas estadísticas no paramétricas

Para la realización de las comparaciones entre los algoritmos propuestos y los de mejor desempeño encontrados en la literatura se utilizarán medidas de estadística descriptiva e inferencial. A diferencia de la estadística descriptiva, cuyos

cálculos son sencillos y de conocimiento general; las estadística inferencial, que abarca las pruebas estadísticas no paramétrica, presentan procedimientos más complejos y se debe explicar claramente para qué se utiliza cada una de estas pruebas.

La tarea esencial de la estadística inferencial es determinar conclusiones sobre un dominio completo de fenómenos (una población), a partir del análisis de una muestra limitada de instancias extraídas de ese dominio. Para lograr esto, se formulan dos hipótesis sobre el fenómeno. La hipótesis nula H_0 enuncia que no existen diferencias entre los fenómenos observados y la hipótesis alternativa H_1 el caso contrario. Sin embargo, la visión del fenómeno proporcionada por la muestra puede representar o no la realidad objetiva; esta última posibilidad se deriva del hecho de que los fenómenos en naturaleza pueden presentar variabilidad aleatoria. En cualquier caso, hasta que se evalúe rigurosamente esa posibilidad, no se pueden extraer conclusiones razonables de una muestra. La significación estadística es el mecanismo lógico y matemático mediante el cual se lleva a cabo esa evaluación. [95].

En las pruebas no paramétricas, el estadístico p es calculado en lugar de definir un valor de significancia a priori. El valor de p determina el nivel de significancia resultante de rechazar H_0 y provee información sobre si la prueba estadística es significativa o no. A menor valor de p, resulta mayor la evidencia en contra de H_0. Luego el rechazo o aceptación de cada hipótesis se realiza de la siguiente forma:

$$\text{Si } p \leq \alpha, \quad \text{se acepta } H_1. \tag{7.1}$$

$$\text{Si } p > \alpha, \quad \text{se acepta } H_0. \tag{7.2}$$

Donde α es un valor que determina el nivel de significancia mínimo necesario para aceptar H_0. Las pruebas no paramétricas en su concepto inicial se ejecutan sobre datos nominales. Sin embargo, pueden ser aplicadas a datos continuos mediante la jerarquización de los datos de entrada. Existen dos clases de análisis: comparaciones por pares y comparaciones múltiples. Los procedimientos estadísticos por pares realizan comparaciones individuales entre dos muestras, obteniendo en cada aplicación un valor p independiente de otro. Por lo tanto, para llevar a cabo una comparación que involucre más de dos muestras, se deben usar pruebas de comparaciones múltiples. Ambas clases de análisis pueden ser utilizados para comparar el desempeño de dos algoritmos, siendo las observaciones en las muestras el resultado de las ejecuciones de los métodos y la población el desempeño general (número infinito de ejecuciones) del algoritmo en un problema determinado. Entonces, la hipótesis H_0 plantea que no existen diferencias significativas en el desempeño general de dos algoritmos y H_1 afirma lo contrario [96].

7.2.1. Prueba de suma de jerarquías de Wilcoxon

La prueba de la suma de jerarquías de Wilcoxon es una prueba no paramétrica que utiliza datos muestrales jerarquizados de dos poblaciones independientes. La prueba de suma de jerarquías de Wilcoxon a menudo se describe como la versión no paramétrica de la prueba t de dos muestras. A diferencia de la prueba t, que asume la distribución normal de las poblaciones, aquí no se asume que existe una distribución conocida. La tasa de eficiencia de prueba de esta prueba es de 0.95 en comparación con la prueba paramétrica t. La hipótesis nula H_0 plantea que las muestras independientes provienen de poblaciones con medianas iguales. La hipótesis alternativa H_1 es la aseveración de que las dos poblaciones tienen medianas diferentes. Dadas dos muestras M_1 y M_2 de tamaño n_1 y n_2 respectivamente, el procedimiento a seguir el siguiente:

1. Combinar temporalmente las dos muestras en una muestra de tamaño M'.
2. Reemplazar cada valor muestral por su jerarquía. Esto es, remplazar el mejor valor por 1 y el peor por 2 para cada par de valores en la muestra. En caso de empate asignar el promedio de las jerarquías a ambos elementos.
3. Separar las muestras y calcular la suma de los rangos R_1 y R_2 de las dos muestras.
4. Calcular el valor del estadígrafo de prueba:

$$z = \frac{R - \mu_R}{\sigma_R} \tag{7.3}$$

   Donde:

$$\mu_R = \frac{n_1(n_1 + n_2 + 1)}{2} \tag{7.4}$$

$$\sigma_R = \sqrt{\frac{n_1 n_2(n_1 + n_2 + 1)}{12}} \tag{7.5}$$

   El valor de R corresponde a la suma de rangos obtenida en la muestra M. Cualquiera de las dos muestras se puede utilizar para calcular el estadígrafo.
5. La prueba de hipótesis se realiza con un nivel de significancia $\alpha = 0.05$ que corresponde a un valor crítico de ± 1.96. Luego:

$$\text{Si } z > 1.96 \text{ o' } z < -1.96, \quad \text{se acepta } H_1 \tag{7.6}$$

$$\text{En caso contrario,} \quad \text{se acepta } H_0 \tag{7.7}$$

7.2.2. Pruebas de Friedman

La prueba de Friedman es una prueba de comparaciones múltiples que puede ser utilizada para detectar diferencias significativas entre el comportamiento de dos o más algoritmos. De forma general, la prueba de Friedman dictamina si para un conjunto de $k \geq 2$ muestras, al menos dos de estas representan poblaciones con medianas diferentes. Estas pruebas son análogas a las pruebas ANOVA de medidas repetidas en procedimientos estadísticos no paramétricos.

De forma similar a otras pruebas de estadística inferencial la hipótesis nula H_0 para la prueba de Friedman plantea la igualdad entre las medias de las poblaciones representadas por las muestras y la hipótesis alternativa H_1 la negación de H_0. El procedimiento general para llevar a cabo esta prueba es el siguiente:

1. Conformar una matriz de k columnas (muestras) y n filas (tamaño de las muestras).
2. Para cada la fila i de la muestra asignar las jerarquías correspondientes (el mejor resultado de la fila toma jerarquía 1, el segundo mejor 2 y así sucesivamente). En caso de empates asignar el promedio de las jerarquías a cada elemento.
3. Calcular por cada la columna los promedios de jerarquías $R_j = \frac{1}{n} \sum_i r_i$
4. Calcular es estadígrafo:

$$F_f = \frac{12n}{k(k+1)} \left[\sum_j R_j - \frac{k(k+1)^2}{4} \right] \tag{7.8}$$

   Donde F_f se distribuye de acuerdo a χ^2 con el que obtiene el valor de p
5. Realizar la prueba de hipótesis con $\alpha = 0.05$:

$$\text{Si } p \leq \alpha, \quad \text{se acepta } H_1. \tag{7.9}$$
$$\text{Si } p > \alpha, \quad \text{se acepta } H_0. \tag{7.10}$$

7.2.3. Pruebas Post-hoc: Bonferroni-Dunn

El principal inconveniente de las pruebas de Friedman, es que sólo pueden detectar diferencias significativas sobre la comparación múltiple completa, y no pueden establecer comparaciones adecuadas entre algunos de los algoritmos considerados. Cuando el objetivo de la aplicación de las pruebas múltiples es realizar una comparación considerando un método de control y un conjunto de algoritmos, se puede definir una familia de hipótesis, todas relacionadas con el método de control. Entonces, la aplicación de una prueba post-hoc puede conducir a

la obtención de un valor p que determina el grado de rechazo de cada hipótesis. Una familia de hipótesis es un conjunto de hipótesis de comparaciones lógicamente interrelacionadas que, en comparaciones $1xN$, compara los algoritmos $k-1$ del estudio (excluyendo el control) con el método de control, mientras que en las comparaciones NxN consideran las $\frac{(k-1)}{2}$ posibles comparaciones entre algoritmos. Por lo tanto, la familia de hipótesis estará compuesta de $k-1$ o $\frac{(k-1)}{2}$ hipótesis, respectivamente, que se pueden ordenar por su valor ρ, de menor a mayor.

El valor ρ de cada hipótesis en la familia se puede obtener a través de la conversión de las clasificaciones calculadas por cada prueba usando una aproximación normal. Las pruebas Post hoc se realizan teniendo en cuenta los resultados obtenidos por una prueba de comparación múltiple como la de Friedman. Por tanto, el estadístico de prueba z para comparar el algoritmo i-ésimo y el algoritmo j-ésimo, depende del principal procedimiento no paramétrico utilizado. En caso de realizar una prueba post-hoc a partir de la prueba de Friedman se utilizará el siguiente estadígrafo:

$$z = \frac{R_i - R_j}{\sqrt{\frac{k(k+1)}{6n}}} \tag{7.11}$$

En las pruebas Post Hoc en lugar de calcular un valor de p para probar cada hipótesis, se utilizan valores de p ajustados (APV por sus siglas en inglés). Estos valores tienen en cuenta el error acumulado al realizar múltiples pruebas. Además, los APV se pueden comparar directamente con cualquier nivel de significancia α elegido. Por lo tanto, se recomienda su uso ya que brindan más información en un análisis estadístico. Para la prueba de Bonferroni-Dunn el APV para la hipótesis i se calcula mediante un solo paso multiplicando el número de comparaciones por el valor p_i obtenido.

$$APV_i = min\{v, 1\}, \text{ donde } v = (k-1)p_i \tag{7.12}$$

7.3. Experimentos preliminares

Los experimentos preliminares se realizan con el objetivo de implementar las primeras variantes de la propuesta de solución y demostrar la competitividad del enfoque propuesto. Por competitividad del enfoque se entiende que los algoritmos diseñados de acuerdo al enfoque puedan obtener resultados iguales o mejores a los encontrados en la literatura especializada de acuerdo a diferentes pruebas de desempeño. Los resultados obtenidos serán medidos de acuerdo a los siguientes aspectos:

1. **Robustez**: dada por las medidas de tendencia central y de dispersión en una muestra de n ejecuciones. La estadística descriptiva permite conocer si los algoritmos logran encontrar los mínimos conocidos de la función y con qué frecuencia lo hacen.
2. **Eficiencia**: dada por la rapidez de convergencia de los algoritmos implementados a partir del enfoque. Para métodos deterministas se utiliza la taza de convergencia la cual determina rapidez con que este se aproxima al mínimo en cada iteración. En el caso de una metaheurística este valor varía en cada ejecución, por lo que generalmente se grafica el valor de función objetivo del mejor individuo, o el promedio de los valores de $f(x)$ de todos los individuos; para una ejecución representativa de una muestra de ejecuciones como es la mediana. Debido a que la operación con mayor complejidad temporal es generalmente la función objetivo, es común que se grafique el valor de f(x) en función del número de evaluaciones para tener una medida la rapidez del método independientemente de plataforma (hardware y software) en que se ejecuta.
3. **Nivel de significación**: las pruebas de estadística inferencial se aplicarán para determinar si existen diferencias significativas entre el desempeño (ya sea mejor o peor) de algoritmos propuestos y los encontrados en la literatura especializada.

Para esto se someten a prueba todas las variantes descritas en la sección 6.2.3. Se realizan tres experimentos preliminares. El experimento A consiste en obtener la estadística descriptiva e inferencial para tomar decisiones de diseño sobre el método de programación matemática a utilizar.

El experimento B consiste en realizar una comparación múltiple entre las variantes HNMED propuestas y la ED/rand/1/bin mediante las pruebas de Friedman y Bonferroni-Dunn, manteniendo el mismo número de evaluaciones para los 6 problemas de optimización a resolver.

En el experimento C se reduce el número de evaluaciones de las variantes híbridas teniendo en cuenta las gráficas de convergencia de cada algoritmo. Se realiza un ajuste de parámetros para cada variante y se mide el desempeño tanto desde el punto de vista estadístico descriptivo como inferencial.

7.3.1. Experimento A: Pruebas de estadísticas sobre el Método Nelder Mead con Expansión de Longitud Aleatoria

Debido a que la hibridación bajo el enfoque propuesto plantea la interacción entre los métodos de búsqueda local y global, el primer experimento se realiza para responder preguntas de investigación, relacionadas con el comportamiento del Nelder Mead al introducir aleatoriedad o información global en sus operadores:

1. ¿ Cómo introducir aleatoriedad en los operadores del NM para aumentar su capacidad de exploración?
2. ¿ Introducir operadores que exploren puntos distantes de la vecindad actual puede aumentar el desempeño en los problemas de optimización a resolver?

Para responder estas interrogantes se plantearon las variantes NMELA descritas en las Sección 6.2.1.

Definición de medidas

Se obtendrá para cada algoritmo una muestra de n ejecuciones. Cada elemento de la muestra es el valor de la función objetivo del mejor individuo factible encontrado en la ejecución i para $i = \{1, 2, ..., n\}$. Para cada muestra se obtendrá:

1. **Mejor**: valor mínimo de función objetivo registrado en la muestra.
2. **Peor**: valor máximo registrado en la muestra.
3. **Mediana**: valor que se encuentra en el centro de la muestra ordenada. Esta medida es fundamental para evaluar el desempeño ya que el promedio puede ser afectado por valores anómalos.
4. **Promedio**: valor característico o tendencia central la muestra.
5. **Desviación estándar**: grado de dispersión de las soluciones registradas en la muestra con respecto al promedio.
6. **Prueba de Friedman**: con nivel significancia $\alpha = 0.05$ para determinar si existen diferencias significativas entre el desempeño de al menos dos de los algoritmos comparados.
7. **Prueba Bonferroni-Dunn**: con nivel significancia $\alpha = 0.05$. Teniendo en cuenta el promedio jerarquías obtenidas en la prueba de Friedman se determina entre cuáles de los algoritmos comparados existe una diferencia significativa de desempeño.

Planificación pre-experimental y configuración

Para el presente experimento se generaron 5 muestras por cada problema, correspondientes a los algoritmos NMELA, NMEILA, NM2ELA, las versiones deterministas NM y el método Nelder Mead con operador de encogimiento. Cada muestra tiene un tamaño tamaño $n = 31$. Las pruebas se realizaron sobre los resultados obtenidos en los 5 primeros problemas de optimización correspondientes al diseño cinemático. Las ejecuciones se llevaron a cabo en una computadora

TABLA 7.1: Evaluaciones utilizadas por problema de optimización: Experimento A.

Problema	**Evaluaciones**
MCE1	400000
MCE2	20000
MCE3	225000
GCE1	125000
GCE2	200000

con procesador AMD Athlon II X2, memoria RAM 2GB, sistema operativo Windows 7 con arquitectura de 64 bits y como entorno de programación se utilizó MatLab R2016b.

Para este experimento se utilizó un número fijo de evaluaciones para todos los algoritmos en comparación en cada problema de optimización según se describe en la Tabla 7.1:

La configuración de parámetros utilizada por las variantes HNMED y la ED es la siguiente:

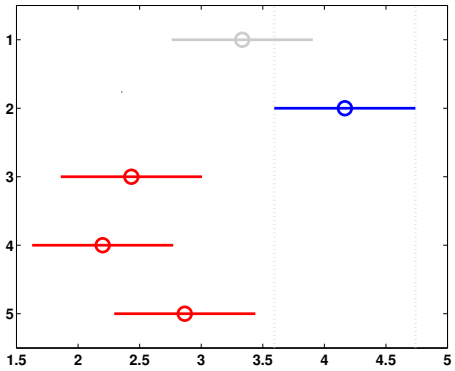
1. Para las variantes NMELA se establecieron los parámetros de reflexión $\alpha = 2$, expansión $\gamma = 1.05$ $\beta = 0.5$ para todos los problemas.
2. Para las variantes determistas se establecieron los parámetros de reflexión $\alpha = 2$, expansión $\gamma = 1.5$ $\beta = 0.5$ para todos los problemas.

Presentación de resultados

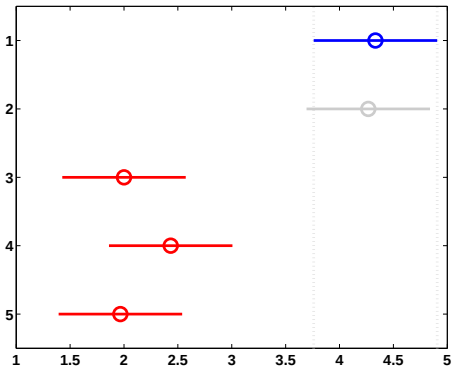
En la Tabla 7.2 se observan los resultados obtenidos para las medidas de estadística descriptivas definidas para el experimento. Las primeras dos columnas muestran los resultados correspondientes a las variantes deterministas NM(versión descrita en sección 3.4.3) y el NM con operador de encogimiento (NMS). La Figura 7.1 muestra los resultados de las pruebas post hoc de Bonferroni-Dunn utilizando los promedios de jerarquías obtenidos en la prueba de Friedman para los 5 problemas de diseño cinemático. Las etiquetas 1 y 2, corresponden al NM y NMS, y las etiquetas 3, 4 y 5 corresponden a NMELA, NMEILA y NM2ELA respectivamente. Las lineas son los intervalos de confianza calculados. En caso de que dos intervalos se traslapen indica que la sub-hipótesis nula se cumple para ambos algoritmos.

TABLA 7.2: Resultados estadísticos obtenidos por las versiones aleatorizadas NMELA en el Experimento A. Los mejores resultados se resaltan en negritas.

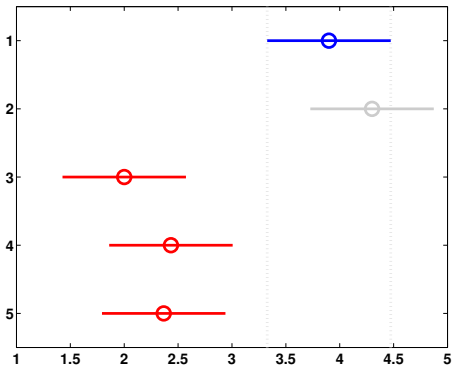
Problema	Estadística	NM	NMS	NMELA	NMEILA	NM2ELA
MCE1	Mejor	6.7163E-06	1.3720E-01	4.9641E-26	3.1550E-25	**3.0569E-26**
	Peor	4.3821E+02	4.5506E+02	**4.3750E+02**	4.3750E+02	9.7062E+03
	Mediana	9.1467E-01	4.3196E+01	**2.0672E-02**	9.3146E-04	2.0673E-02
	Promedio	6.6808E+01	9.6835E+01	7.4217E+01	**4.5667E+01**	7.5356E+02
	Desv. Est.	1.4200E+02	1.4305E+02	1.5845E+01	**1.1869E+01**	2.1343E+03
	Evaluaciones			**400000**		
MCE2	Mejor	2.6590E-03	7.2367E-04	3.8155E-04	2.6280E-03	**3.8126E-04**
	Peor	7.7302E+03	1.2757E-01	3.9431E+00	**1.4211E-02**	3.2957E-01
	Mediana	3.6442E-03	3.7092E-03	**2.6281E-03**	2.6288E-03	**2.6281E-03**
	Promedio	2.7292E+02	9.5048E-03	1.3391E-01	**3.0235E-03**	1.3837E-02
	Desv. Est.	1.4109E+03	2.3509E-02	7.1944E-01	**2.1132E-02**	5.9671E-02
	Evaluaciones			**20000**		
MCE3	Mejor	1.9620E+00	2.9152E+01	2.7496E-01	**2.7496E-01**	2.7496E-01
	Peor	2.5350E+03	4.2133E+04	**1.3508E+01**	1.4420E+01	1.3508E+01
	Mediana	1.5008E+01	6.3121E+01	4.7321E-01	**1.1410E+00**	1.2710E+00
	Promedio	4.3987E+02	1.9493E+03	**4.4381E+00**	5.0760E+00	5.0759E+00
	Desv. Est.	8.3627E+02	7.7310E+03	**6.0582E+00**	6.2441E+00	6.0650E+00
	Evaluaciones			**225000**		
GCE1	Mejor	4.4681E-27	9.0876E-28	1.0602E-27	**3.1554E-30**	8.7090E-28
	Peor	3.3175E+05	1.5295E+05	**1.2863E+05**	3.3973E+05	1.1731E+05
	Mediana	1.4012E+04	1.7275E+02	4.4855E-01	**5.6447E-15**	5.5696E+02
	Promedio	5.4149E+04	2.5236E+04	**7.6006E+03**	1.8545E+04	7.7064E+03
	Desv. Est.	8.3973E+04	4.4103E+04	2.4672E+04	6.2803E+04	**2.3619E+04**
	Evaluaciones			**125000**		
GCE2	Mejor	1.1872E-01	1.3937E-01	1.1399E-01	1.1391E-01	**1.1389E-01**
	Peor	2.8728E+05	2.5495E+05	**6.8139E+04**	7.2095E+04	1.5666E+05
	Mediana	7.6415E+03	1.1269E+02	1.0466E+00	**1.7267E-01**	5.1727E-01
	Promedio	5.3444E+04	1.5327E+04	1.1382E+04	**4.6427E+03**	1.0443E+04
	Desv. Est.	8.3696E+04	5.0724E+04	2.2244E+04	**1.5319E+04**	3.3321E+04
	Evaluaciones			**200000**		



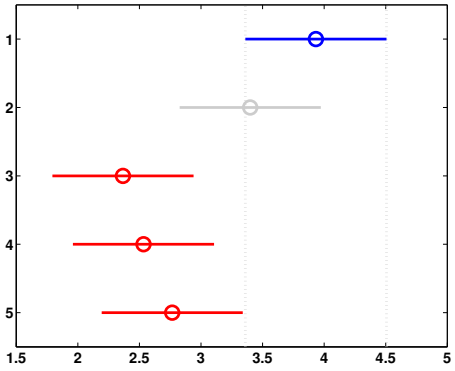
(A) MCE1



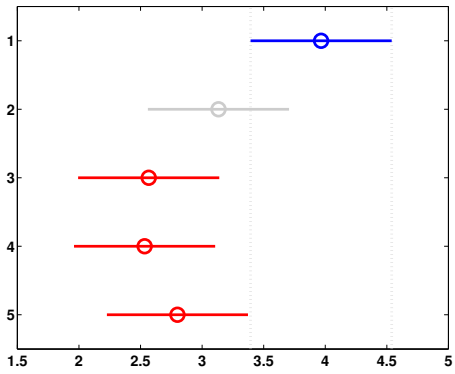
(B) MCE2



(C) MCE3



(D) GCE1



(E) GCE2

FIGURA 7.1: Resultados de las pruebas de Bonferroni-Dunn para las variantes NMELA y NM en Experimento A. Se etiquetan en el eje vertical del 1 al 5 los algoritmos NM, NMS, NMELA, NMEILA, NM2ELA respectivamente.

Observaciones

En la Tabla 7.2 se pueden observar que las variantes NMLEA obtienen los mejores resultados en todas las medidas de estadística descriptiva y son capaces de acercase, igualar o superar en el mejor valor de la muestra a los mínimos encontrados para estos problemas. Sin embargo las medidas de tendencia central y dispersión se encuentran lejos de estos mínimos en la mayoría de los problemas, lo que indica la sensibilidad de estos algoritmos a los puntos iniciales. El resultado de las pruebas de Friedman y Bonferroni-Dunn indican en todos los problemas existen diferencias significativas entre al menos una variante determinista y una variante aleatorizada. Para los problemas MCE2 y MCE3 existen diferencias significativas entre el grupo determinista marcado en azul y las versiones propuestas que se encuentran a la izquierda en el grupo de color rojo con las mejores jerarquías. La capacidad del método Nelder-Mead para utilizar eficazmente información proveniente de regiones lejanas al área donde se encuentra operando, es la característica fundamental observada de forma experimental, por la cual se elige como método de búsqueda local para el diseño de los algoritmos híbridos HNMED.

7.3.2. Experimento B: Comparación de desempeño entre Variantes HNMED vs ED/rand/1/bin

El primer experimento se realiza para responder preguntas de investigación básicas sobre el enfoque de hibridación propuesto:

1. ¿ Son los algoritmos diseñados de acuerdo al enfoque propuesto capaces de hallar valores iguales o cercanos a los mínimos conocidos para los problemas planteados ?
2. ¿ Con qué frecuencia los algoritmos diseñados de acuerdo al enfoque propuesto son capaces de hallar valores iguales o cercanos a los mínimos conocidos para los problemas planteados ?
3. ¿ Con qué rapidez los algoritmos diseñados de acuerdo al enfoque propuesto son capaces converger a los mínimos conocidos para los problemas planteados ?
4. ¿ Presentan los algoritmos propuestos en conjunto un desempeño significativamente mejor a la Evolución Diferencial en los problemas de optimización a resolver?
5. ¿ Cuáles de los algoritmos propuestos presentan mejor desempeño?

Definición de medidas

Se obtendrá para cada algoritmo una muestra de n ejecuciones. Cada elemento de la muestra es el valor de la función objetivo del mejor individuo factible encontrado en la ejecución i para $i = \{1, 2, ..., n\}$. Para cada muestra se obtendrá:

1. **Mejor**: valor mínimo de función objetivo registrado en la muestra.
2. **Peor**: valor máximo registrado en la muestra.
3. **Mediana**: valor que se encuentra en el centro de la muestra ordenada. Esta medida es fundamental para evaluar el desempeño ya que el promedio puede ser afectado por valores anómalos.
4. **Promedio**: valor característico o tendencia central la muestra.
5. **Desviación estándar**: grado de dispersión de las soluciones registradas en la muestra con respecto al promedio.
6. **Gráfica de convergencia**: Presenta el valor de la función objetivo del mejor individuo en la población en cada evaluación para la ejecución correspondiente a la mediana de la muestra. La graficación de estos valores permite observar la rapidez de convergencia de los métodos.
7. **Prueba de Friedman**: con nivel significancia $\alpha = 0.05$ para determinar si existen diferencias significativas entre el desempeño de al menos dos de los algoritmos comparados.
8. **Prueba Bonferroni-Dunn**: con nivel significancia $\alpha = 0.05$. Teniendo en cuenta el promedio jerarquías obtenidas en la prueba de Friedman se determina entre cuáles de los algoritmos comparados existe una diferencia significativa de desempeño.

Planificación pre-experimental y configuración

Para el presente experimento se generaron 6 muestras correspondientes a las 5 variantes propuestas y la ED/rand/1/bin de tamaño $n = 31$. Las ejecuciones se llevaron a cabo en una computadora con procesador AMD Athlon II X2, memoria RAM 2GB, sistema operativo Windows 7 con arquitectura de 64 bits y como entorno de programación se utilizó MatLab R2016b.

Para este experimento se utilizó un número fijo de evaluaciones (las utilizadas por la ED) para todos los algoritmos en comparación en cada problema de optimización según se describe en la tabla 7.3:

La configuración de parámetros utilizada por las variantes HNMED y la ED es la siguiente:

TABLA 7.3: Evaluaciones utilizadas por problema de optimización: Experimento B.

Problema	Evaluaciones
MCE1	750030
MCE2	20000
MCE3	450018
GCE1	750030
GCE2	750030
SCE1	288000

TABLA 7.4: Tamaño de población utilizados para cada problema en el experimento B.

	ED/rand/1/bin	HNMED	
Problema	**NP**	**NS**	**NP**
MCE1	138	9	144
MCE2	50	4	28
MCE3	138	7	140
GCE1	138	9	135
GCE2	138	8	135
SCE1	50	7	35

1. Se establece la probabilidad de cruza $CR = \{0.8, 1\}$ para ED y HNMED para todos los problemas.
2. Se establece el factor de escala $F = \{0.3, 0.9\}$ para ED y HNMED para todos los problemas.
3. Para las variantes HNMED se establecieron los parámetros de reflexión $\alpha = 2$, expansión $\gamma = 1.5$ $\beta = 0.5$ para todos los problemas.
4. El tamaño de población se estableció según la Tabla 7.4. Donde el valor NS se refiere a la cantidad de símpleces utilizados los cuales determinan la población para cada problema $NP = NS(N+1)$ donde N es la dimensión del problema.

Presentación de resultados

En la Tabla 7.5 se muestran los resultados obtenidos en la estadísticas descriptivas para las variantes propuestas y la ED/rand/1/bin en los 6 problemas de optimización. Los resultados ganadores se resaltan en letra negrita. A pesar de que existe el análisis teórico para determinar la velocidad de convergencia, los experimentos realizados se limitaron a la graficación del valor de $f(x)$ presentado por el mejor individuo factible en la población, en función del número de

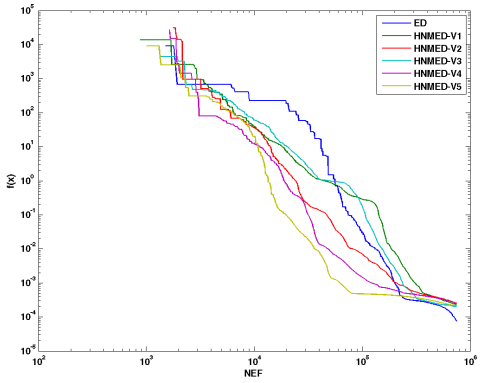
TABLA 7.5: Resultados estadísticos obtenidos por las variantes HNMED y DE/rand/1/bin en el Experimento B para los seis problemas de optimización. Se marcan en negritas los mejores valores de cada medida.

Problema	Estadística	HNMED-V1	HNMED-V2	HNMED-V3	HNMED-V4	HNMED-V5	ED/rand/1/bin
MCE1	Mejor	5.55358E-28	5.31513E-07	**0**	1.26218E-29	5.04871E-29	1.7670E-28
	Peor	2.73815E-02	1.6149E+00	2.5685E-02	**2.2528E-02**	2.5749E-02	2.7649E-02
	Mediana	2.2312E-04	2.5478E-04	2.0521E-04	2.4414E-04	2.1680E-04	**4.2746E-06**
	Promedio	2.6670E-03	5.4682E-02	**1.8612E-03**	3.6684E-03	2.5208E-03	2.9850E-01
	Desv. Est.	7.3917E-03	2.8965E-01	**6.2715E-03**	7.9230E-03	7.1453E-03	8.1906E-03
MCE2	Mejor	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03
	Peor	2.6290E-03	**2.6280E-03**	**2.6280E-03**	**2.6280E-03**	**2.6280E-03**	2.6426E-03
	Mediana	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03
	Promedio	2.6281E-03	**2.6280E-03**	**2.6280E-03**	**2.6280E-03**	**2.6280E-03**	2.6288E-03
	Desv. Est.	2.3345E-07	**17600E-18**	1.0776E-15	3.2328E-15	**1.7600E-18**	2.7205E-06
MCE3	Mejor	2.7496E-01	2.7496E-01	2.7496E-01	2.7496E-01	2.7496E-01	2.7496E-01
	Peor	1.3508E+01	1.3508E+01	1.3508E+01	1.3508E+01	1.3508E+01	1.3508E+01
	Mediana	4.3436E-01	1.6290E+00	9.1663E-01	6.6851E-01	4.2346E-01	**2.7563E-01**
	Promedio	4.2838E+00	6.4180E+00	4.7519E+00	5.1492E+00	5.5448E+00	**1.2313E+00**
	Desv. Est.	6.0160E+00	6.5415E+00	5.9814E+00	6.3172E+00	6.4445E+00	**3.2919E+00**
GCE1	Mejor	2.8398E-29	**0**	**0**	**0**	**0**	6.7147E-27
	Peor	3.1333E-27	1.7670E-27	6.8157E-28	6.3109E-29	**5.0487E-29**	5.9926E-20
	Mediana	1.0302E-28	3.1554E-30	3.1554E-30	1.5777E-30	**0**	3.9485E-23
	Promedio	1.2272E-27	8.7721E-29	4.3439E-29	1.1359E-29	**7.9937E-30**	3.4957E-21
	Desv. Est.	97794E-28	3.2234E-28	1.2753E-28	**1.6873E-29**	1.7728E-29	1.2077E-20
GCE2	Mejor	**1.1385E-01**	**1.1385E-01**	**1.1385E-01**	**1.1385E-01**	**1.1385E-01**	1.1388E-01
	Peor	1.9561E-01	1.5875E-01	1.5965E-01	1.5976E-01	**1.5751E-01**	1.6075E-01
	Mediana	1.1995E-01	1.1440E-01	1.5199E-01	1.1410E-01	**1.1385E-01**	1.1447E-01
	Promedio	1.3600E-01	1.3166E-01	1.3561E-01	1.2979E-01	1.2940E01	**1.2080E-01**
	Desv. Est.	2.4766E-02	2.0980E-02	2.1283E-02	1.9838E-01	1.9926E-02	**1.4713E-02**
SCE1	Mejor	-5.7049E+05	**-5.7741E+05**	-5.6195E+05	-5.4519E+05	-5.3238E+05	-5.3206E+05
	Peor	-5.1016E+05	-5.2146E+05	-5.2772E+05	-5.3071E+05	**-5.3198E+05**	-5.3204E+05
	Mediana	-5.3059E+05	-5.3158E+05	-5.3135E+05	-5.3184E+05	**-5.3209E+05**	-5.3205E+05
	Promedio	-5.3174E+05	**-5.3339E+05**	-5.3308E+05	-5.3236E+05	-5.3214E+05	-5.3205E+05
	Desv. Est.	1.1452E+04	9.2013E+03	7.1985E+03	2.4432E+03	1.0096E+02	**3.3010E+00**

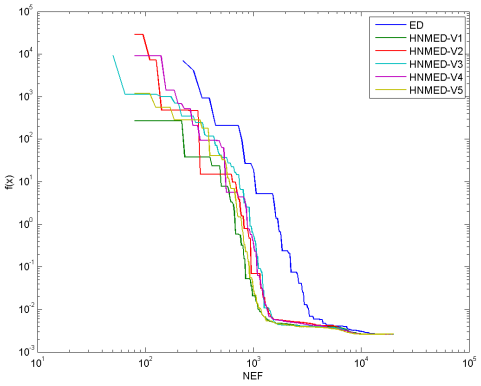
evaluaciones de $f(x)$ denominado NEF. En la Figura 7.2 se muestran las gráficas de convergencia de la ejecución correspondiente a la mediana de la muestra. En caso del problema de la micro-red eléctrica Se grafican los valores de función objetivo correspondientes a la hora 12.

En la Tabla 7.6 se presentan las jerarquías promedios obtenidas por los algoritmos, los valores p y la aceptación o rechazo de la hipótesis obtenidas en la prueba de Firedman en cada uno de los problemas de optimización. Se resaltan en letra negrita las promedios de jerarquías ganadores.

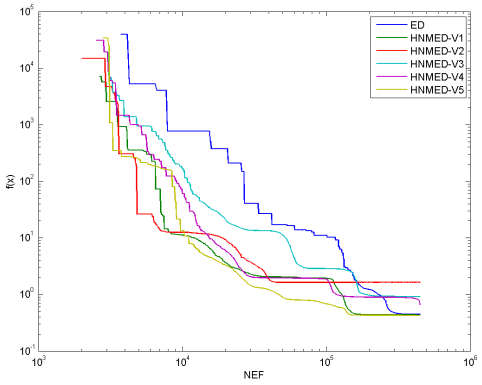
Teniendo en cuenta los resultados de la prueba de Friedman, la Figura 7.3 la prueba post hoc de Bonferroni-Dunn para los 6 problemas. En las gráficas, el eje vertical presenta las etiquetas dadas a cada algoritmo y el horizontal las jerarquías promedios. Las etiquetas están asignadas de la siguiente forma: a la ED le corresponde el número 1 y a las variantes HNMED desde V1 a V5 les corresponden los números del 2 al 6 respectivamente. Los círculos corresponden a la asignación de la etiqueta L_j con el promedio jerarquía R_j. Las lineas son los intervalos de



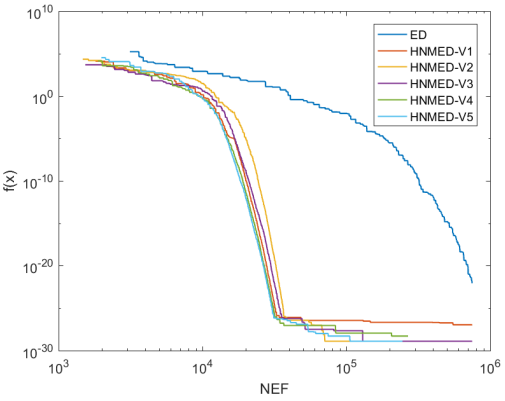
(A) MCE1



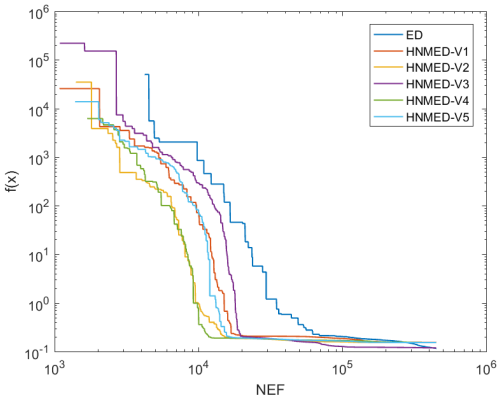
(B) MCE2



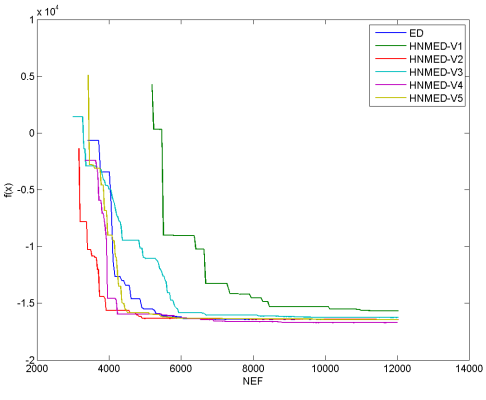
(C) MCE3



(D) GCE1



(E) GCE2



(F) SCE2

FIGURA 7.2: Gráficas de convergencia obtenidas por las variantes HNMED y ED/rand/1/bin en el Experimento B. El eje horizontal presenta el número de evaluaciones de la función objetivo (NEF) y el eje vertical el valor de $f(x)$.

confianza calculados por la prueba.

Cuando los intervalos de confianza de dos muestras no se traslapan, se rechaza la sub-hipótesis nula y por lo tanto se afirma que existen diferencias significativas entre los algoritmos que describen dichas muestras. En las figuras presentadas los promedios de jerarquía e intervalos de confianza entre los cuales existen diferencias significativas se dibujan con colores diferentes (azul y rojo) para hacer más legible la gráfica.

Observaciones

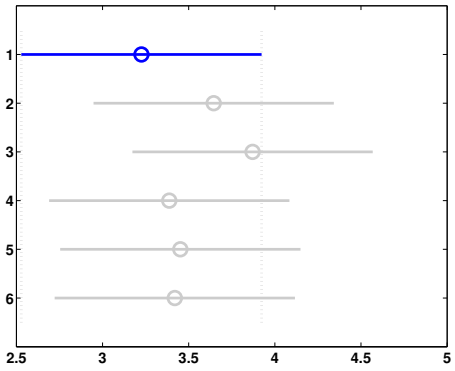
Los resultados obtenidos en la estadística descriptiva responden la primera pregunta de investigación planteada para el experimento B; ya que en efecto, las variantes propuestas son capaces de hallar valores iguales o mejores a los mínimos conocidos para todos los problemas de optimización a resolver. Las observaciones por cada problema son las siguientes:

1. **MCE1**: Las variantes HNMED superan a la ED en las medidas de mejor y peor solución, promedio y desviación estándar. Ninguna logra superar el valor de la mediana de la ED. La variante HNMED con mejores resultados es V3.
2. **MCE2**: Las variantes HNMED superan a la ED en todas las medidas. La variante con mejores resultados es V5.
3. **MCE3**: Se iguala a la ED en la mejor y peor observación de la muestra. La ED supera a las variantes HNMED en los valores de mediana, promedio y desviación estándar.La variante con mejores resultados es la V3.
4. **GCE1**: Las variantes supera a la ED en todas las medidas. HNMED-V5 presenta los mejores resultados.
5. **GCE2**: Todas las variantes superan a la ED en su mejor y peor solución, excepto HNMED-V1. Sólo HNMED-V5 (variante con mejores resultados) supera la mediana de la ED la cual prevalece en las demás medidas (promedio y desviación estándar).
6. **SCE1**: Todas las variantes superan a la ED en el mejor resultado, siendo HNMED-V5 la que logra superar a la ED en las medidas de promedio y mediana.

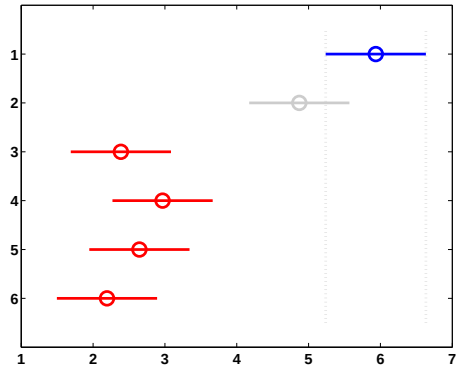
De forma general, las variantes propuestas presentan mejores valores en las medidas de tendencia central para la mayoría de los problemas y valores modestos de desviación estándar lo que indica cierta sensibilidad a la distribución inicial de la población. Se destacan además las medida de la mejor solución de la muestra en la cual las variantes siempre obtienen resultados iguales (para MCE2 y MCE3)

TABLA 7.6: Resultados de la prueba de Friedman obtenidos por las variantes HNMED y DE/rand/1/bin en el Experimento B para los seis problemas de diseño mecatrónico. Se resaltan en negrita los promedios de jerarquía ganadores.

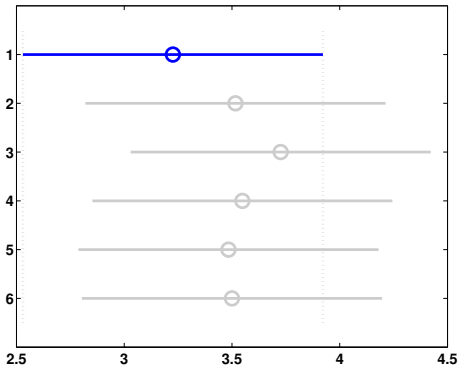
Problema	Algoritmo	Jerarquía promedio	Valor de p	H_0	H_1
MCE1	HNMED-V1	3.6451E+00	8.1173E-01	Aceptada	Rechazada
	HNMED-V2	3.8709E+00			
	HNMED-V3	3.3870E+00			
	HNMED-V4	3.4516E+00			
	HNMED-V5	3.4193E+00			
	ED/rand/1/bin	**3.2258E+00**			
MCE2	HNMED-V1	3.5322E+00	1.3277E-26	Rechazada	Aceptada
	HNMED-V2	**2.8387E+00**			
	HNMED-V3	2.9193E+00			
	HNMED-V4	2.9032E+00			
	HNMED-V5	**2.8387E+00**			
	ED/rand/1/bin	5.9677E+00			
MCE3	HNMED-V1	3.5161E+00	9.4985E-01	Aceptada	Rechazada
	HNMED-V2	3.7258E+00			
	HNMED-V3	3.5483E+00			
	HNMED-V4	3.5483E+00			
	HNMED-V5	3.5800E+00			
	ED/rand/1/bin	**3.2258E+00**			
GCE1	HNMED-V1	4.9000E+00	1.3365E-22	Rechazada	Aceptada
	HNMED-V2	2.7333E+00			
	HNMED-V3	2.7333E+00			
	HNMED-V4	2.5000E+00			
	HNMED-V5	**2.1333E+00**			
	ED/rand/1/bin	6.0000E+00			
GCE2	HNMED-V1	4.0967E+00	1.0989E-01	Aceptada	Rechazada
	HNMED-V2	3.4032E+00			
	HNMED-V3	3.8387E+00			
	HNMED-V4	3.0483E+00			
	HNMED-V5	**2.9677E+00**			
	ED/rand/1/bin	3.0322E+00			
SCE1	HNMED-V1	3.8709E+00	1.7205E-03	Rechazada	Aceptada
	HNMED-V2	3.7741E+00			
	HNMED-V3	4.3225E+00			
	HNMED-V4	3.4516E+00			
	HNMED-V5	**2.4193E+00**			
	ED/rand/1/bin	3.1612E+00			



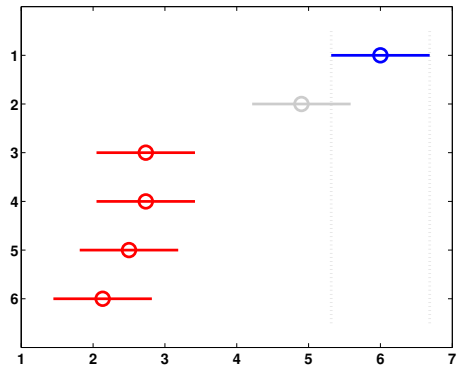
(A) MCE1



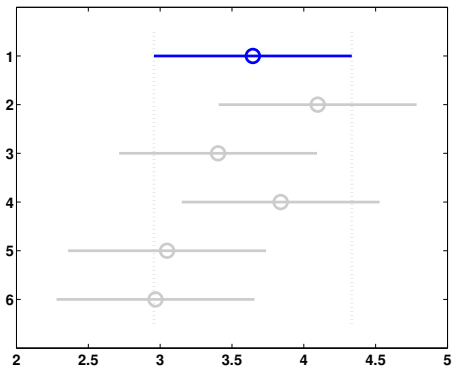
(B) MCE2



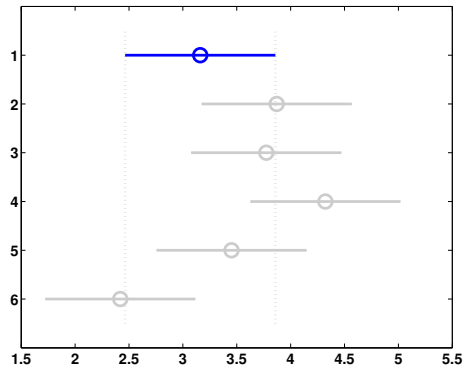
(C) MCE3



(D) GCE1



(E) GCE2



(F) SCE1

FIGURA 7.3: Resultados de las pruebas de Bonferroni-Dunn obtenidos por las variantes HNMED vs ED/rand/1/bin en el Experimento B. Se etiquetan en el eje vertical del 1 al 6 los algoritmos ED/rand/1/bin y HNMEDV1 a HNMEDV5 respectivamente.

o mejores (MCE1,GCE1,GCE2,SCE1) a la ED. Esto se debe a la capacidad de explotación de las instancias del método Nelder Mead.

En la Figura 7.2 se observa que, para todos los problemas las curvas descritas por las variantes HNMED se encuentran por debajo a las de ED/rand/1/bin. Esto significa que todas las versiones pueden obtener resultados competitivos en un menor número de evaluaciones. También se puede observar que las curvas dadas por los algoritmos propuestos se encuentran ligeramente corridas a la izquierda, lo que indica que son capaces de encontrar individuos factibles en iteraciones tempranas. Este comportamiento se debe a la capacidad del simplex de muestrear el espacio de búsqueda con pocas evaluaciones. Es importante destacar que con los resultados obtenidos se demuestran dos aspectos principales robustez y eficiencia del enfoque propuesto.

En la Tabla 7.6 se observan los resultados obtenidos en la prueba de Friedman para cada problema:

1. **MCE1**: El mejor promedio de jerarquía obtenido corresponde a la ED. Sin embargo, se acepta la hipótesis nula con un valor de $p = 8.1173E-01$. Esto significa que a pesar de que la ED obtiene mejores resultados en este problema, no se puede afirmar que son significativamente mejores y por lo tanto su desempeño es similar al obtenido por las variantes HNMED. La mejor jerarquía obtenida por las variantes HNMED corresponden a V5.

2. **MCE2**: El algoritmo HNMED-V5 obtiene el mejor promedio de jerarquía. Además, todas las versiones HNMED obtienen una mejor jerarquía promedio con respecto a la ED. En este caso se rechaza la hipótesis nula con valor de $p = 1.3277E-26$. En este problema se evidencia un desempeño significativamente mejor por parte de las 5 variantes propuestas en comparación con la ED.

3. **MCE3**: La ED/rand/1/bin alcanza el mejor promedio de jerarquía. No obstante, se acepta la hipótesis nula con valor de $p = 9.4985E-01$. Al igual que en la prueba para el problema MCE1 se concluye que existe un desempeño similar entre todos los algoritmos comparados.

4. **GCE1**: De forma similar al problema MCE2 todas las variantes HNMED obtienen mejor jerarquía respecto a la ED, siendo la V5 el algoritmo ganador. La hipótesis nula se rechaza concluyendo que las versiones V5 se comportan significativamente mejor que la ED.

5. **GCE2**: EL algoritmo ganador es HNMED-V5. Todas las versiones HNMED obtienen mejores jerarquías que la ED a excepción de la V1. En este caso la hipótesis nula se acepta por lo tanto no se confirma que existen diferencias significativas entre el desempeño de las variantes ganadoras y la ED.

6. **SCE1**: En este caso se rechaza la hipótesis nula con valor de $p = 1.7205E-03$, siendo HNMED-V5 el algoritmo con mejor promedio de jerarquía. Sin embargo, como se puede observar en la Subfigura F de la Figura **??**, las diferencias significativas se encuentran entre HNMED-V5 y las variantes V1 y V2.

De forma general se observa un desempeño competitivo de las variantes propuestas, obteniéndose resultados significativamente mejores en 2 de los problemas y desempeño similar el resto con respecto a la ED/rand/1/bin. La variante HNMED-V5 consigue el mejor desempeño de forma general la cual obtiene mejor promedio de jerarquías en 4 de los 6 problemas de optimización. Los peores promedios de jerarquía se obtienen en los problemas MCE1,MCE3 y GCE2 los cuales presentan mayor dimensión, restricciones y complejidad de la función. A pesar de que las variantes propuestas ganan en rapidez de convergencia, heredan en cierta medida las deficiencias del método de búsqueda local cuyas operaciones están destinadas a la explotación. Por tanto, se presenta un decremento del desempeño para problemas de mayor dimensionalidad aunque este no es significativo.

7.3.3. Experimentos C : Comparación de Variantes HNMED vs ED con reducción del número de evaluaciones

El experimento C se realiza para conocer de forma exacta el número de evaluaciones en las que, las variantes propuestas presentan un desempeño competitivo en comparación con la ED/rand/1/bin. Con la realización de estos experimentos se reúne información sobre las variables contempladas en la hipótesis de investigación (calidad de resultados y rapidez) que permitirá junto con los experimentos finales una aceptación o rechazo de la misma.

Definición de medidas

Se obtiene para cada algoritmo una muestra de $n = 31$ ejecuciones. Para cada problema las observaciones en la muestra corresponden al valor de la función objetivo del mejor individuo factible encontrado por el algoritmo $A_i$en la ejecución j para $j = \{1, 2, ..., n\}$. Se aplicarán al conjunto de muestras las siguientes pruebas de estadística descriptiva e inferencial:

1. **Estadística descriptiva**: Se obtiene la la mejor y peor solución, mediana, promedio y desviación estándar de las muestras
2. **Gráfica de convergencia**: Se presenta la gráfica de convergencia correspondiente a la mediana de la muestra para cada algoritmo en comparación.

TABLA 7.7: Evaluaciones utilizadas por problema de optimización: Experimento C.

Problema	ED	HNMED-V1	HNMED-V2	HNMED-V3	HNMED-V4	HNMED-V5
MCE1	750030	500000	400000	400000	400000	400000
MCE2	20000	15000	15000	15000	15000	15000
MCE3	450018	35000	25000	25000	225000	225000
GCE1	750030	125000	125000	125000	125000	125000
GCE2	750030	225000	225000	225000	225000	225000
SCE1	288000	240000	240000	240000	216000	216000

3. **Prueba de Friedman**: con nivel significancia $\alpha = 0.05$ para determinar si existen diferencias significativas entre el desempeño de al menos dos de los algoritmos comparados.
4. **Prueba Bonferroni-Dunn**: con nivel significancia $\alpha = 0.05$ para determina entre cuáles de los algoritmos comparados existe una diferencia significativa de desempeño.

Planificación pre-experimental y configuración

Para este experimento se mantuvo el número de evaluaciones realizadas por la ED según se describe en la literatura. El número de evaluaciones fue reducido para cada uno de los algoritmos en comparación en cada problema de optimización según se describe en la Tabla 7.7:

La configuración de parámetros utilizada por las variantes HNMED y la ED es la siguiente:

1. Se establece la probabilidad de cruza $CR = \{0.8, 1\}$ para ED y HNMED para todos los problemas.
2. Se establece el factor de escala $F = \{0.3, 0.9\}$ para ED y HNMED para todos los problemas.
3. Para las variantes HNMED se establecieron los parámetros de reflexión $\alpha = 2$, expansión $\gamma = 1.05$ $\beta = 0.5$ para todos los problemas.
4. El tamaño de población utilizado se ajusta para cada algoritmo según la Tabla 7.8. Donde el valor NS se refiere a la cantidad de símpleces utilizados los cuales determinan la población para cada problema $NP = NS(N + 1)$ donde N es la dimensión del problema.

TABLA 7.8: Tamaños de población utilizados para cada problema en el Experimento C.

	ED/rand/1/bin	HNMED-V1		HNMED-V2		HNMED-V3		HNMED-V4		HNMED-V5	
Problema	**NP**	**NS**	**NP**	**NS**	**NP**	**NS**	**NP**	**NS**	**NP**	**NS**	**NP**
MCE1	138	9	144	8	128	8	128	7	112	7	112
MCE2	50	3	21	4	28	4	28	3	21	3	21
MCE3	138	7	140	7	140	7	140	7	140	7	140
GCE1	138	9	135	9	135	9	135	9	135	9	135
GCE2	138	9	135	9	135	9	135	9	135	9	135
SCE1	50	7	35	7	35	7	35	7	35	7	35

Presentación de resultados

La Tabla 7.9 muestra los resultados estadísticos obtenidos para cada problema indicando la reducción de evaluaciones en porciento. En la Figura 7.4 se muestra las gráficas de convergencias. La Tabla 7.10 presenta los resultados de la prueba de Friedman y en la Figura 7.5 contiene las pruebas post hoc de Bonferroni-Dunn para los 6 problemas de optimización.

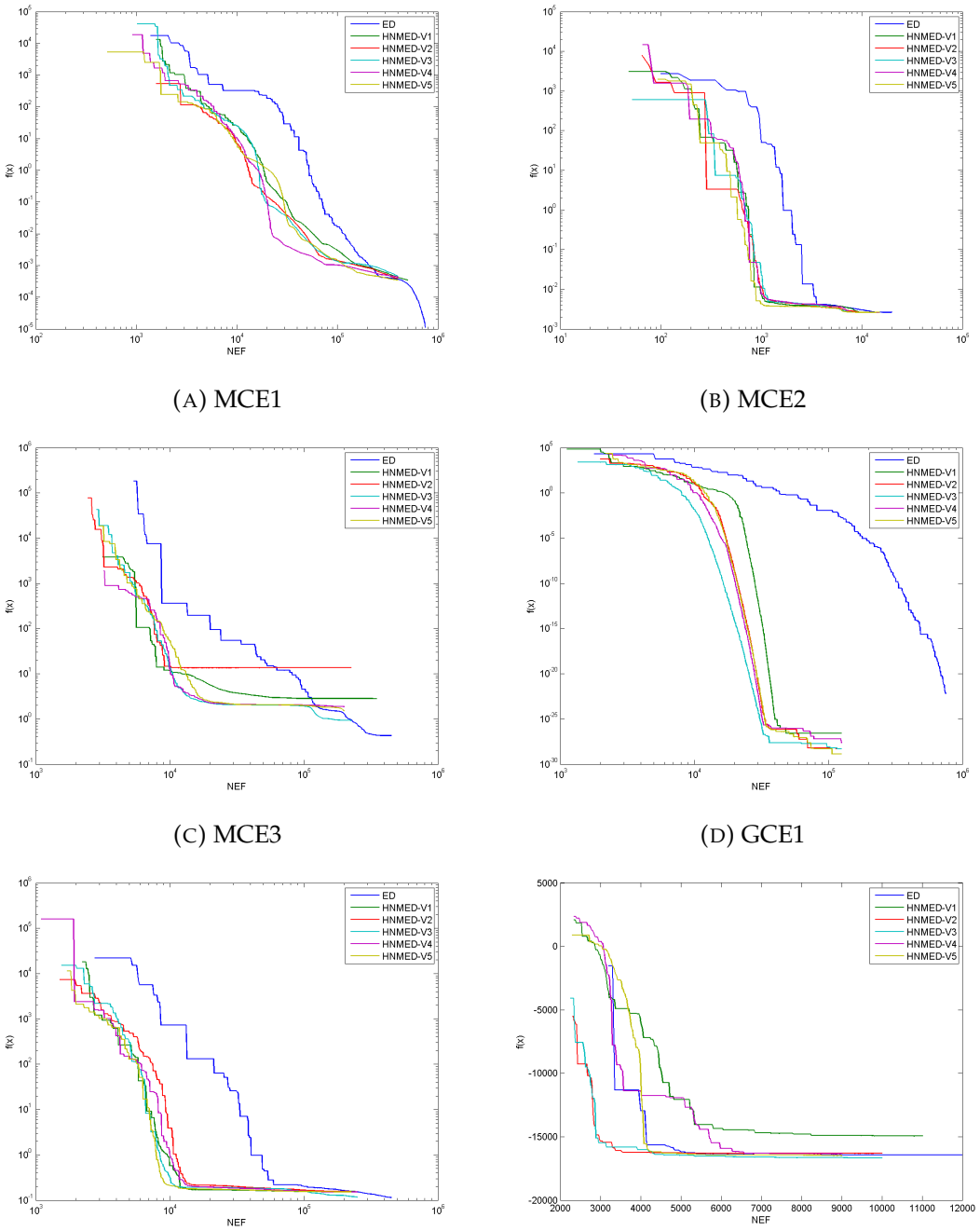
Observaciones

Teniendo en cuenta los resultados obtenidos en ambos experimentos se pueden realizar las siguientes observaciones:

1. Las variantes HNMED son capaces de encontrar resultados iguales o mejores para la mayoría de los problemas de optimización utilizando un número de evaluaciones menor, el cual representa un ahorro de más del 44 % en la mayoría de los problemas con excepción de MCE2 (-33.3 %) y SCE1 (-25.3 %) para las variante HNMED-V4 y V5).
2. De forma similar a los resultados obtenidos en el experimento A, las curvas de convergencia que describen las versiones de HNMED correspondientes a la mediana de las muestras se encuentran por debajo de la curva de la ED/ran/1/bin para la todos los problemas. Esto se logra reduciendo el número de símpleces y por tanto el tamaño de la población, esto implica un ahorro de evaluaciones por iteración y permite realizar más operaciones de explotación. El hecho de que se alcancen los mínimos conocidos y valores de tendencia central competitivos descarta la posibilidad de que HNMED presente una tendencia a la convergencia prematura o estancamiento para los problemas presentados.
3. Al reducir el número de evaluaciones algunas versiones comienzan a presentar un desempeño significativamente inferior a la ED como es el caso de la V1. La variante HNMED-V5 es la que presenta mejor promedio de

TABLA 7.9: Resultados estadísticos obtenidos por las variantes HNMED y DE/rand/1/bin en el Experimento C para los seis problemas de optimización. Se marcan en negritas los mejores valores de cada medida.

Problema	Estadística	HNMED-V1	HNMED-V2	HNMED-V3	HNMED-V4	HNMED-V5	ED/rand/1/bin
MCE1	Mejor	5.80602E-28	**1.26218E-29**	5.04871E-29	6.31089E-29	4.9224E-29	1.7670E-28
	Peor	3.1713E-02	2.7676E-02	2.9936E-02	**2.7424E-02**	2.7432E-02	2.7649E-02
	Mediana	3.4667E-04	3.9921E-04	4.5762E-04	3.9547E-04	3.4607E-04	**4.2746E-06**
	Promedio	**3.1573E-03**	4.7172E-03	7.1235E-03	4.5629E-03	6.9630E-03	2.9850E-01
	Desv. Est.	8.7501E-03	1.0145E-02	**1.1201E-03**	4.9555E-03	1.1579E-03	8.1906E-03
	Evaluaciones	500000(-33 %)	400000 (-46.66 %)	400000 (-46.66 %)	400000 (-46.66 %)	400000 (-46.66 %)	750030
MCE2	Mejor	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03
	Peor	2.8317E-03	**2.6280E-03**	**2.6280E-03**	**2.6280E-03**	**2.6280E-03**	2.6426E-03
	Mediana	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03	2.6280E-03
	Promedio	2.6349E-03	**2.6280E-03**	**2.6280E-03**	**2.6280E-03**	**2.6280E-03**	2.6288E-03
	Desv. Est.	4.6798e-03	**8.3771E-10**	2.2862E-09	6.8497E-08	2.8769E-09	2.7205E-06
	Evaluaciones	**15000**(-33.3 %)	**15000**(-33.3 %)	**15000**(-33.3 %)	**15000**(-33.3 %)	**15000**(-33.3 %)	20000
MCE3	Mejor	2.7496E-01	2.7527E-01	2.7496E-01	2.7496E-01	2.7496E-01	2.7496E-01
	Peor	1.3508E+01	1.3508E+01	1.3508E+01	1.3508E+01	1.3508E+01	1.3508E+01
	Mediana	7.3675E-01	7.5977E+01	1.6433E+00	5.4040E-01	8.8976E-01	**2.7563E-01**
	Promedio	5.1387E+00	5.6636E+00	5.79209E+00	4.6850E+00	5.6552E+00	**1.2313E+00**
	Desv. Est.	6.3164E+00	6.6190E+00	6.2649E+00	6.1952E+00	6.3652E+00	**8.1906E-03**
	Evaluaciones	350000(-22.2 %)	250000(-44.4 %)	250000(-44.4 %)	225000(-50.0 %)	**225000**(-50.0 %)	450018
GCE1	Mejor	6.3108E-29	**0**	**0**	**0**	**0**	6.7147E-27
	Peor	1.7951E-26	1.9090E-27	4.7773E-27	2.0699E-27	**9.0876E-28**	5.9926E-20
	Mediana	2.47387E-27	5.0487E-29	5.04871-29	5.6797E-29	**1.2621E-29**	3.9485E-23
	Promedio	3.80302E-27	2.3248E-28	4.8268E-28	2.4801E-28	**6.2396E-29**	3.4957E-21
	Desv. Est.	4.36121E-27	4.8191E-28	1.1600E-27	4.4014E-28	**1.6612E-28**	1.2077E-20
	Evaluaciones	**125000**(-83.3 %)	**125000**(-83.3 %)	**125000**(-83.3 %)	**125000**(-83.3 %)	**125000**(-83.3 %)	750030
GCE2	Mejor	**1.1385E-01**	1.4713E-02	**1.1385E-01**	**1.1385E-01**	**1.1385E-01**	1.1388E-01
	Peor	1.89116E-01	2.3328E-01	1.7400E-01	**1.5932E-01**	1.6559E-01	1.6075E-01
	Mediana	1.5513E-01	1.5417E-01	1.1746E-01	1.5202E-01	1.5294E-01	**1.1447E-01**
	Promedio	1.4773E-01	1.4407E-01	1.3132E-01	1.3611E-01	1.3934E-01	**1.2080E-01**
	Desv. Est.	2.1528E-02	2.6887E-02	2.06472E-02	2.0660E-02	2.0410E-02	**1.4713E-02**
	Evaluaciones	**250000**(-44.4 %)	**250000**(-44.4 %)	**250000**(-44.4 %)	**250000**(-44.4 %)	**225000**(-50.0 %)	450018
SCE1	Mejor	-5.7452E+05	-5.7725E+05	**-5.8217E+05**	-5.7364E+05	-5.3652E+05	-5.3206E+05
	Peor	-4.9379E+05	-4.9669E+o5	-5.2250E+05	-5.2442E+05	-5.3080E+05	**-5.3204E+05**
	Mediana	-5.3667E+05	-5.3083E+05	-5.3188E+05	-5.3175E+05	**-5.3217E+05**	-5.3205E+05
	Promedio	-5.3690E+05	-5.33062E+05	-5.4196E+05	-5.3292E+05	**-5.3230E+05**	-5.3205E+05
	Desv. Est.	1.9677E+04	1.7853E+04	1.9516E+04	7.9197E+03	1.0945E+03	**3.3010E+00**
	Evaluaciones	240000 (-8.3 %)	240000 (-16.6 %)	240000 (-16.6 %)	**216000** (-25.0 %)	**216000** (-25.0 %)	288000



(A) MCE1

(B) MCE2

(C) MCE3

(D) GCE1

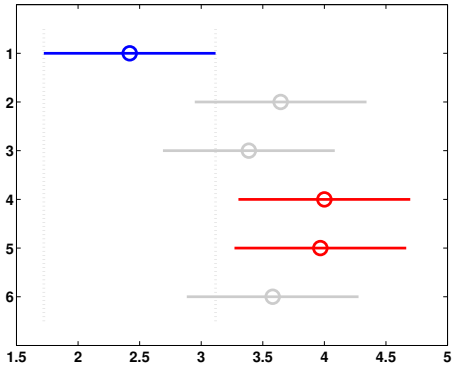
(E) GCE2

(F) SCE2

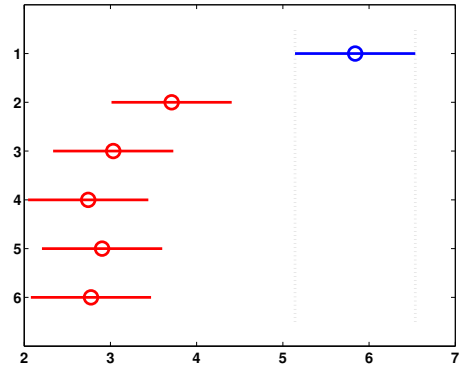
FIGURA 7.4: Gráficas de convergencia obtenidas por las variantes HNMED y ED/rand/1/bin en el Experimento C. El eje horizontal presenta el número de evaluaciones de la función objetivo (NEF) y el eje vertical el valor de $f(x)$.

TABLA 7.10: Resultados de la prueba de Friedman obtenidos por las variantes HNMED y DE/rand/1/bin en el experimento C para los seis problemas de diseño mecatrónico. Se resaltan en negritas los promedios de jerarquías ganadores

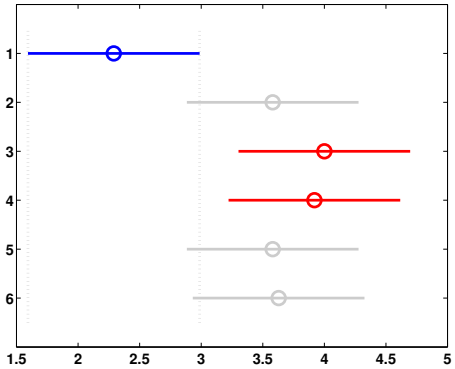
Problema	Algoritmo	Jerarquía promedio	Valor de p	H_0	H_1
MCE1	HNMED-V1	3.6451E+00	1.1011E-02	Rechazada	Aceptada
	HNMED-V2	3.3870E+00			
	HNMED-V3	4.0000E+00			
	HNMED-V4	3.9677E+00			
	HNMED-V5	3.5806E+00			
	ED/rand/1/bin	**2.4193E+00**			
MCE2	HNMED-V1	3.7096E+00	2.1026E-12	Rechazada	Aceptada
	HNMED-V2	3.0322E+00			
	HNMED-V3	2.74193E+00			
	HNMED-V4	2.9032E+00			
	HNMED-V5	**2.7741E+00**			
	ED/rand/1/bin	5.8387E+00			
MCE3	HNMED-V1	3.580645	4.4788E-03	Rechazada	Aceptada
	HNMED-V2	4.0000E+00			
	HNMED-V3	3.9193E+00			
	HNMED-V4	3.5806E+00			
	HNMED-V5	3.6290E+00			
	ED/rand/1/bin	**2.2903E+00**			
GCE1	HNMED-V1	4.6451E+00	1.3274E-20	Rechazada	Aceptada
	HNMED-V2	2.500E+00			
	HNMED-V3	2.6290E+00			
	HNMED-V4	3.4354E+00			
	HNMED-V5	**1.8870E+00**			
	ED/rand/1/bin	5.9032E+00			
GCE2	HNMED-V1	4.5483E+00	3.9377E-03	Rechazada	Aceptada
	HNMED-V2	3.8709E+00			
	HNMED-V3	**3.0322E+00**			
	HNMED-V4	3.2580E+00			
	HNMED-V5	3.7741E+00			
	ED/rand/1/bin	2.5161E+00			
SCE1	HNMED-V1	3.1935E+00	3.2643E-02	Rechazada	Aceptada
	HNMED-V2	4.3225E+00			
	HNMED-V3	3.3870E+00			
	HNMED-V4	3.9032E+00			
	HNMED-V5	**2.8709E+00**			
	ED/rand/1/bin	3.3225E+00			



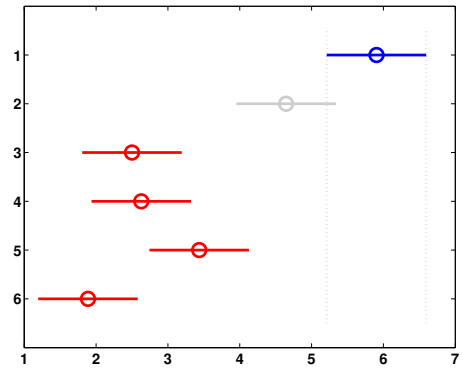
(A) MCE1



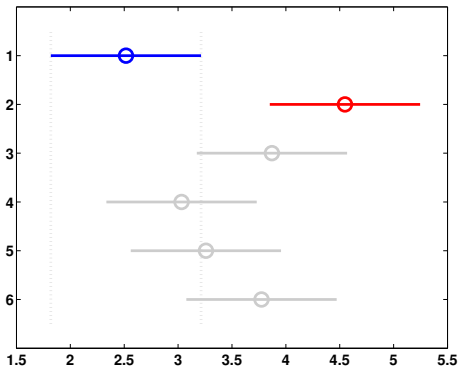
(B) MCE2



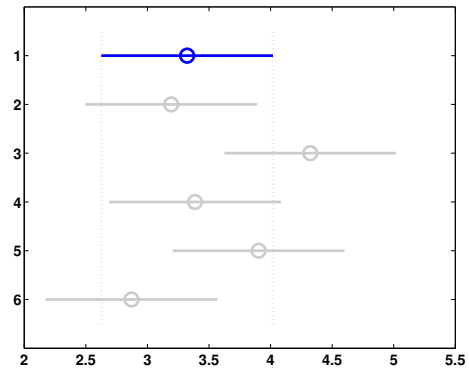
(C) MCE3



(D) GCE1



(E) GCE2



(F) SCE1

FIGURA 7.5: Resultados de las pruebas de Bonferroni-Dunn obtenidos por las variantes HNMED vs ED/rand/1/bin en el Experimento C. Se etiquetan en el eje vertical del 1 al 6 los algoritmos ED/rand/1/bin y HNMEDV1 a HNMEDV5 respectivamente.

jerarquía en la mayoría de los problemas obteniendo resultados significativamente mejores o similares a la ED.

7.4. Experimentos finales

Los experimentos iniciales permitieron observar el comportamiento de las variantes HNMED diseñadas bajo el enfoque de hibridación propuesto en la Sección 6.2. Se evidenció que a pesar de la rapidez de estos algoritmos, se presentaron dificultades en los problemas MCE1, MCE3 y GCE2. Estos problemas presentan un espacio de búsqueda más complejo, ya sea por contar con mayor número de restricciones, dimesionalidad, complejidad de la función objetivo o la cualquier combinación de estas características.

Las variantes propuestas heredan, en cierta medida, algunas deficiencias del método de búsqueda local utilizado como la reducción del desempeño ante un aumento de la dimesionalidad del problema y la sensibilidad a los puntos iniciales. Por tanto, una mayor capacidad de exploración es requerida en orden de alcanzar mejores resultados. Con este objetivo trazado, se diseñó la HNMED-V6 cuyo desempeño es medido en los experimentos finales junto a la variante HNMED-V5.

7.4.1. Experimentos D y E : Comparación de Variantes HNMED-V5 y HNMED-V6 con Algoritmo C-LSHADE

El objetivo de ambos experimentos es comprobar la competitividad de las dos mejores versiones HNMED. Por tanto, se realizan dos comparaciones finales por medio de estadística descriptiva e inferencial: el algoritmo HNMED-V5 y HNMED-V6 con el algoritmo C-LSHADE propuesto por Zapata en [13] una adaptación del algoritmo L-SHADE para problemas con restricciones. L-SHADE está basado en la Evolución Diferencial y se caracteriza por realizar una adaptación de los parámetros F y CR utilizando una memoria histórica de parámetros, donde se almacenan aquellos valores de F y CR que han tenido éxito en el mejoramiento de los individuos. Su procedimiento también incorpora un mecanismo de reducción lineal del tamaño de población. L-SHADE fue ganador de la Sesión Especial y Competencia de Optimización Mono-Objetivo con Parámetros Reales CEC-2014 [97].

TABLA 7.11: Evaluaciones utilizadas por problema de optimización: Experimentos D y E.

Problema	C-LCHADE	HNMED-V5	HNMED-V6
MCE1	400000	400000	350000
MCE2	15000	15000	10000
MCE3	200000	25000	175000
GCE1	325000	125000	100000
GCE2	150000	225000	125000
SCE1	240000	210000	210000

Definición de medidas

Se obtiene para cada algoritmo una muestra de $n = 31$ ejecuciones. Para cada problema las observaciones en la muestra corresponden al valor de la función objetivo del mejor individuo factible encontrado por el algoritmo A_i en la ejecución j para $j = \{1, 2, ..., n\}$. Se aplicarán al conjunto de muestras las siguientes pruebas de estadística descriptiva e inferencial:

1. **Estadística descriptiva**: Se obtiene la la mejor y peor solución, mediana, promedio y desviación estándar de las muestras
2. **Gráfica de convergencia**: Se presenta la gráfica de convergencia correspondiente a la mediana de la muestra para cada uno de los algoritmos en comparación.
3. **Prueba suma de Jerarquías de Wilcoxon**: con nivel significancia $\alpha = 0.05$ para determinar si existen diferencias significativas entre el desempeño los dos algoritmos comparados.

Planificación pre-experimental y configuración

Para este experimento se mantuvo el número de evaluaciones realizadas por C-LSHADE según se describe en [13]. En la Tabla 7.11 se describe el número de evaluaciones realizadas por cada algoritmo comparado.

Se mantiene la configuración de HNMED-V5 y C-LSHADE. La configuración de parámetros utilizada por la variante HNMED-V6:

1. Se establece la probabilidad de cruza $CR = \{0.8, 1\}$ para todos los problemas.
2. Se establece el factor de escala $F = \{0.3, 0.9\}$ para todos los problemas.
3. Establecieron los parámetros de reflexión $\alpha = 2$, expansión $\gamma = 1.7$ $\beta = 0.3$ para todos los problemas.

TABLA 7.12: Tamaños de población utilizados para cada problema en los experimentos finales.

	C-LSHADE	**HNMED-V5**		**HNMED-V6**
Problema	**NP**	**NS**	**NP**	**NS**	**NP**
MCE1	138	9	144	8	128
MCE2	50	3	21	4	28
MCE3	138	7	140	5	75
GCE1	138	9	135	7	105
GCE2	138	9	135	7	105
SCE1	50	7	35	2	10

4. El tamaño de población utilizado se ajusta para cada algoritmo según la Tabla 7.12.

Resultados del Experimento D

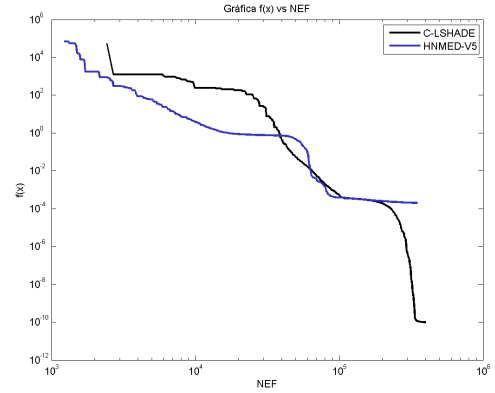
La Tabla 7.13 muestra los resultados estadísticos obtenidos por HNMED-V5 y los obtenidos por C-LSHADE en [13]. En la Figura 7.6 se presentan las gráficas de convergencias correspondientes al Experimento D para los seis problemas. Finalmente, la Tabla 7.14 contiene los resultados de la prueba de Suma de Jerarquías de Wilcoxon para los seis problemas de optimización con un nivel de significancia de 95 %.

Resultados del Experimento E

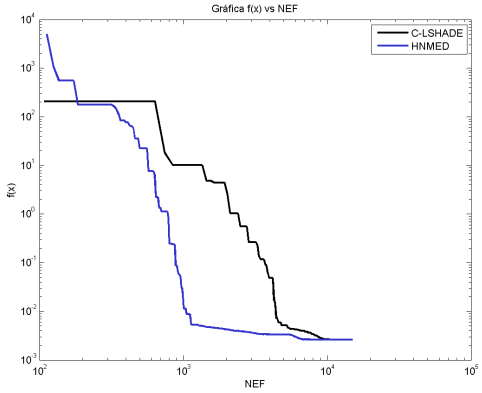
En las Tablas 7.15 y 7.16 se muestran los resultados estadísticos obtenidos por la HNMED-V6 frente a la ED/rand/1/bin y C-LSHADE respectivamente. Se agrega la comparación estadística de HNMED-V6 con la ED/rand/1/bin para evidenciar la reducción del número evaluaciones respecto a esta variante. Se realizó la comparación directa con C-LSHADE ya que este supera en desempeño a la ED/rand/1/bin [13]. En la Figura 7.7 se muestra las gráficas de convergencia correspondientes al Experimento E. Por último, en la Tabla 7.17 se presenta para la comparación de HNMED-V6 y C-LSHADE mediante prueba de Suma de Jerarquías de Wilcoxon para lo seis problemas de optimización de diseño mecatrónico con un nivel de significancia del 95 %.

TABLA 7.13: Resultados estadísticos obtenidos por HNMED-V5 y C-LSHADE en el Experimento D para los seis problemas de diseño mecatrónico. Se marcan en negritas los mejores valores de cada medida.

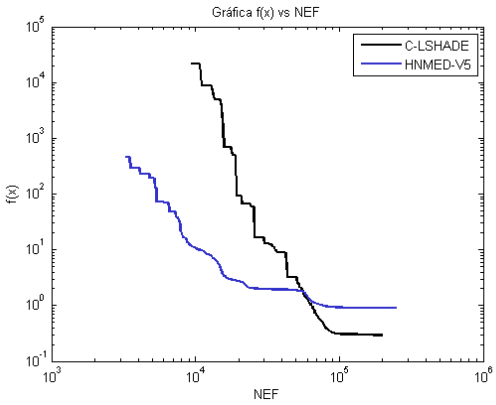
Problema	Estadística	HNMED-V5	C-LSHADE
MCE1	Mejor	6.3108E-29	**0**
	Peor	2.7323E-02	**6.4277E-04**
	Mediana	3.2843E-04	**3.6143E-08**
	Promedio	2.5730E-03	**2.4021E-04**
	Desv. Est.	6.9657E-03	**2.7680E-04**
	Evaluaciones	400000	400000
MCE2	Mejor	2.6280E-03	2.6280E-03
	Peor	**2.6280E-03**	2.6281E-03
	Mediana	2.6280E-03	2.6280E-03
	Promedio	2.6280E-03	2.6280E-03
	Desv. Est.	**2.1708E-10**	8.5532E-09
	Evaluaciones	15000	15000
MCE3	Mejor	2.7577E-01	**2.7496E-01**
	Peor	1.3508E+01	1.3508E+01
	Mediana	5.0946E-01	**2.8886E-01**
	Promedio	4.8813E+00	**1.6068E+00**
	Desv. Est.	6.2202E+00	**3.9629E+00**
	Evaluaciones	225000	**200000** (-12.5 %)
GCE1	Mejor	0	0
	Peor	7.6992E-28	**3.0292E-28**
	Mediana	1.8932E-29	**1.2621E-29**
	Promedio	1.0886E-28	**2.7991E-28**
	Desv. Est.	1.7494E-28	**5.6901E-29**
	Evaluaciones	**125000**(-61.5 %)	325000
GCE2	Mejor	1.1385E-01	1.1385E-01
	Peor	1.6276E-01	**1.5783E-01**
	Mediana	1.1631E-01	**1.1392E-01**
	Promedio	1.2887E-01	**1.2066E-01**
	Desv. Est.	1.9809E-02	**1.5457E-02**
	Evaluaciones	250000	**150000**(-40.0 %)
SCE1	Mejor	**-5.3652E+05**	-5.3205E+05
	Peor	-5.3080E+05	-5.3205E+05
	Mediana	**-5.3217E+05**	-5.3205E+05
	Promedio	**5.3230E+05**	-5.3205E+05
	Desv. Est.	1.094E+03	**1.0515E-04**
	Evaluaciones	**216000**(-10.0 %)	240000



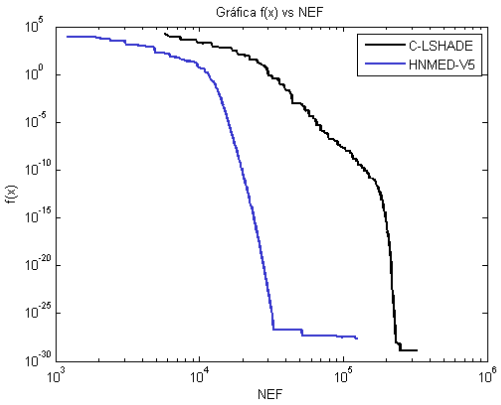
(A) MCE1



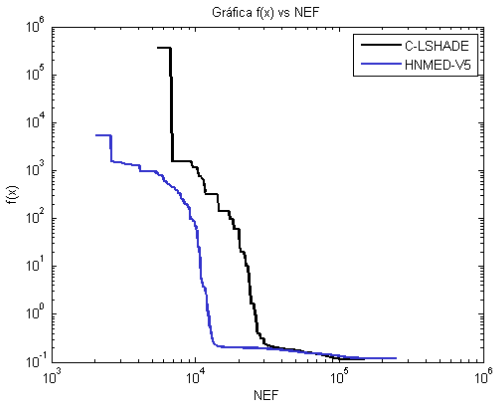
(B) MCE2



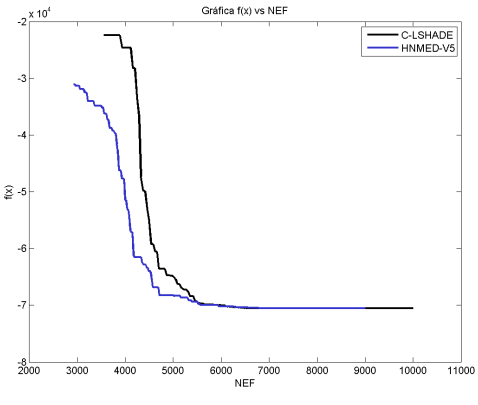
(C) MCE3



(D) GCE1

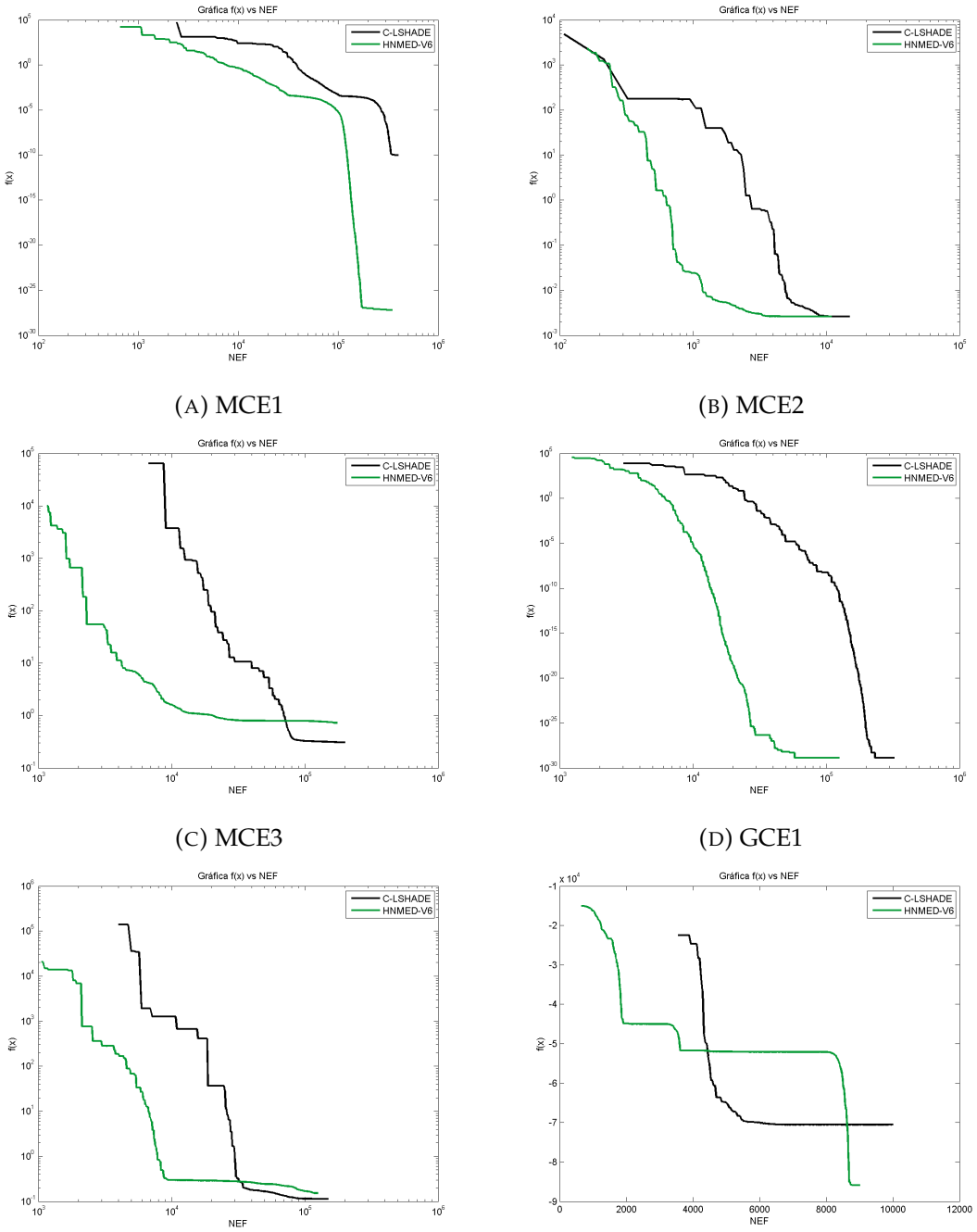


(E) GCE2



(F) SCE2

FIGURA 7.6: Gráficas de convergencia obtenidas por la variante HNMED y C-LSHADE en el Experimento D. El eje horizontal presenta el número de evaluaciones de la función objetivo (NEF) y el eje vertical el valor de $f(x)$.



(A) MCE1

(B) MCE2

(C) MCE3

(D) GCE1

(E) GCE2

(F) SCE2

FIGURA 7.7: Gráficas de convergencia obtenidas por la variante HNMED-V6 y C-LSHADE en el Experimento E. El eje horizontal presenta el número de evaluaciones de la función objetivo (NEF) y el eje vertical el valor de $f(x)$.

TABLA 7.14: Comparación de C-LSHADE y HNMED-V5 en los seis problemas de optimización mediante la prueba de Suma de Jerarquías de Wilcoxon. Los símbolos +, - y ≈ señalan que HNMED-V5 obtuvo un desempeño significativamente mejor (+), significativamente peor (-) o no significativamente diferente (≈) comparado contra C-LSHADE usando la prueba de suma de jerarquías de Wilcoxon con un 95 % de confianza

vs C-LSHADE	**Problemas**						**Total**
	MCE1	MCE2	MCE3	GCE1	GCE2	SCE1	
		+				+	2
HNMED-V5	−		−		−		3
				≈			1

Observaciones Finales

Los resultados de los experimentos E y D permiten plantear las siguientes observaciones finales:

1. La versión con peor desempeño en comparación a C-LSHADE resultó ser V5, la cual obtiene un desempeño significativamente inferior en los problemas MCE1, MCE3 y GCE2 y sólo alcanza mejores resultados para los problemas MCE2 y SCE1 (los de menor dimensión). A pesar de que HNMED-V5 es capaz de utilizar un número menor de evaluaciones para los problemas GCE1 y SCE1; para los casos MCE3 y GCE2 el algoritmo C-LSHADE utiliza menos evaluaciones de la función objetivo. Las gráficas de convergencia muestran que HNMED-V5 presenta mayor rapidez de convergencia respecto C-LSHADE, sin embargo el punto al que converge no siempre es mejor que el encontrado por C-LSHADE para todos los problemas de optimización.

2. La versión más competitiva es HNMED-V6, lo que se evidencia tanto en las pruebas de estadística descriptiva como inferencial. HNMED-V6 obtienen resultados de mediana y promedio mejores que C-LSHADE en los problemas MCE1, MCE2, GCE1 y SCE1. Para el problema MCE2 se encuentra un nuevo mínimo $(f(\vec{x}) = 2.5643E-04)$ el cual supera al obtenido tanto por la ED/rand/1/bin como por C-LSHADE con valor de $(f(\vec{x}) = 2.6280E-03)$. En la sección de visualización de resultados se puede observar que éste mínimo se encuentra en un punto del espacio de búsqueda distante de la solución encontrada hasta el momento en la literatura. Para el problema de la micro-red aislada se logra encontrar un ahorro total significativamente mayor con un valor de $\sum_1^{24} f(\vec{x}) = -5.5147$, en este caso las curvas de costo en tiempo son similares a las descritas por la soluciones dadas por C-LSHADE

TABLA 7.15: Resultados estadísticos obtenidos por HNMED-V6 vs ED/rand/1/bin en el experimento E para los seis problemas de diseño mecatrónico. Se marcan en negritas los mejores valores de cada medida.

Problema	Estadística	HNMED-V5	ED/rand/1/bin
MCE1	Mejor	**3.7865E-29**	1.7670E-28
	Peor	**3.6342E-03**	2.7649E-02
	Mediana	**6.5633E-28**	4.2746E-06
	Promedio	**1.6110E-04**	2.9850E-01
	Desv. Est.	**6.4886E-04**	8.1906E-03
	Evaluaciones	**350000**(-53.53 %)	750030
MCE2	Mejor	**2.5643E-04**	2.6280E-03
	Peor	**2.6280E-03**	2.6426E-03
	Mediana	2.6280E-03	2.6280E-03
	Promedio	**2.3985E-03**	2.6288E-03
	Desv. Est.	7.1276E-04	**2.7205E-06**
	Evaluaciones	**10000**(50.0 %)	20000
MCE3	Mejor	2.7496E-01	2.7496E-01
	Peor	1.3508E+01	1.3508E+01
	Mediana	3.0867E-01	**2.7563E-01**
	Promedio	2.4412E+00	**1.2313E+00**
	Desv. Est.	4.7466E+00	**3.2919E+00**
	Evaluaciones	**175000**(-61.1 %)	450018
GCE1	Mejor	**0**	6.7147E-27
	Peor	**5.3327E-28**	5.9926E-20
	Mediana	**1.2621E-29**	3.9485E-23
	Promedio	**5.8223E-29**	3.4957E-21
	Desv. Est.	**1.1714E-28**	1.2077E-20
	Evaluaciones	**100000**(-86.6 %)	750030
GCE2	Mejor	1.1385E-01	1.1385E-01
	Peor	2.3331E-01	**1.6075E-01**
	Mediana	1.5210E-01	**1.1447E-01**
	Promedio	1.4351E-01	**1.2080E-01**
	Desv. Est.	3.3531E-02	**1.4713E-02**
	Evaluaciones	**125000**(-72.2 %)	450018
SCE1	Mejor	**-5.5147E+05**	-5.3206E+05
	Peor	**-5.5147E+05**	-5.3204E+05
	Mediana	**-5.5147E+05**	-5.3205E+05
	Promedio	**-5.5147E+05**	-5.3205E+05
	Desv. Est.	**2.3667E-10**	3.3010E+00
	Evaluaciones	**216000**(-25.0 %)	288000

TABLA 7.16: Resultados estadísticos obtenidos por HNMED-V6 vs C-LSHADE en los seis problemas de diseño mecatrónico. Se marcan en negritas los mejores valores de cada medida.

Problema	Estadística	HNMED-V6	C-LSHADE
MCE1	Mejor	3.7865E-29	**0**
	Peor	3.6342E-03	**6.4277E-04**
	Mediana	**6.56332E-28**	3.6143E-08
	Promedio	**1.6110E-04**	2.4021E-04
	Desv. Est.	6.4886E-04	**2.7680E-04**
	Evaluaciones	**350000**(-12.5 %)	400000
MCE2	Mejor	**2.5643E-04**	2.6280E-03
	Peor	**2.6280E-03**	2.6281E-03
	Mediana	2.6280E-03	2.6280E-03
	Promedio	**2.3985E-03**	2.6280E-03
	Desv. Est.	7.1276E-04	**8.5532E-09**
	Evaluaciones	**10000**(30.0 %)	15000
MCE3	Mejor	2.7496E-01	2.7496E-01
	Peor	1.3508E+01	1.3508E+01
	Mediana	3.0867E-01	**2.8886E-01**
	Promedio	2.4412E+00	**1.6068E+00**
	Desv. Est.	4.7466E+00	**3.9629E+00**
	Evaluaciones	**175000**(-22.2 %)	200000
GCE1	Mejor	0	0
	Peor	5.3327E-28	**3.0292E-28**
	Mediana	1.2621E-29	1.2621E-29
	Promedio	**5.8223E-29**	2.7991E-28
	Desv. Est.	1.1714E-28	**5.6901E-29**
	Evaluaciones	**100000**(-61.5 %)	325000
GCE2	Mejor	1.1385E-01	1.1385E-01
	Peor	2.3331E-01	**1.5783E-01**
	Mediana	1.5210E-01	**1.1392E-01**
	Promedio	1.4351E-01	**1.2066E-01**
	Desv. Est.	3.3531E-02	**1.5457E-02**
	Evaluaciones	**125000**(-16.6 %)	150000
SCE1	Mejor	**-5.5147E+05**	-5.3205E+05
	Peor	**-5.5147E+05**	-5.3205E+05
	Mediana	**-5.5147E+05**	-5.3205E+05
	Promedio	**-5.5147E+05**	-5.3205E+05
	Desv. Est.	**2.3667E-10**	1.0515E-04
	Evaluaciones	**216000**(-12.5 %)	240000

TABLA 7.17: Comparación de C-LSHADE y HNMED-V6 en los seis problemas de optimización mediante la prueba de Suma de Jerarquías de Wilcoxon. Los símbolos +, - y $\approx$ señalan que HNMED-V6 obtuvo un desempeño significativamente mejor (+), significativamente peor (-) o no significativamente diferente ($\approx$) comparado contra LSHADE-CV usando la prueba de suma de jerarquías de Wilcoxon con un 95 % de confianza

vs C-LSHADE	**Problemas**						**Total**
	MCE1	MCE2	MCE3	GCE1	GCE2	SCE1	
		+				+	2
HNMED-V6			−		−		2
	$\approx$			$\approx$			2

y ED/rand/1/bin. Esto significa que se mejoran los resultados tanto por capacidad de exploración como por explotación. Al igual que la variante anterior, se presentan los peores resultados para los problemas MCE3 y GCE1 con respecto C-LSHADE.

3. Un aspecto importante sobre el desempeño de la variante HNMED-V6 es su rápida convergencia para todos los problemas de optimización, lo que resulta en reducción del número de evaluaciones frente a las utilizadas por C-LSHADE y el algoritmo ED/rand/1/bin. Respecto a este último, HNMED-V6 logra reducir el número de evaluaciones en más del 50 % para 5 de los 6 problemas de optimización. Por otra parte, se puede observar en la Tabla 7.12, que el número de símpleces y por tanto los tamaños de población utilizados son inferiores, lo que constituye un ahorro de espacio en memoria. Esto se debe a la estrategia de inicialización de la población que genera símpleces de mayor tamaño garantizando un cubrimiento efectivo del espacio de búsqueda en etapas tempranas del proceso de minimización. Al utilizar menor número de símpleces también se reducen los ordenamientos y las evaluaciones realizadas por generación, permitiendo el ahorro de evaluaciones en general y la rapidez de convergencia alcanzados.

4. Finalmente, se concluye que la hipótesis de la presente investigación es **aceptada**, debido a que los algoritmos propuestos, los cuales están basados en un método de programación matemática y utilizan operadores de un algoritmo evolutivo son capaces de encontrar resultados iguales o mejores que los ya reportados en la literatura especializada en los problemas de optimización de diseño mecatrónico, utilizando un menor número de evaluaciones.

7.5. Visualización de Resultados

En las secciones anteriores se evidenció que los mejores resultados fueron obtenidos por HNMED-V6. A continuación se presentan los vectores de diseño obtenidos por esta variante de HNMED para los seis problemas de optimización. Se realizan además, ciertas comparaciones en aquellos casos en los que se logró superar los resultados obtenidos en la literatura especializada, así como en aquellos donde se encuentran diferentes vectores de diseño con igual valor de función objetivo. Las simulaciones de los problemas de diseño cinemático fueron implementadas y ejecutadas en el software Geobegra 2.9.

7.5.1. Mecanismo de Cuatro Barras

En la Tabla 7.18 se muestran los resultados obtenidos por HNMED-V6 y se incorporan además los vectores obtenidos mediante C-LSHADE. Se puede observar para el caso del MEC2, que el vector obtenido por HNMED-V6 es un punto del espacio de búsqueda distante del encontrado por C-LSHADE y ED/rand/1/bin. En los demás casos se encontraron vectores similares.

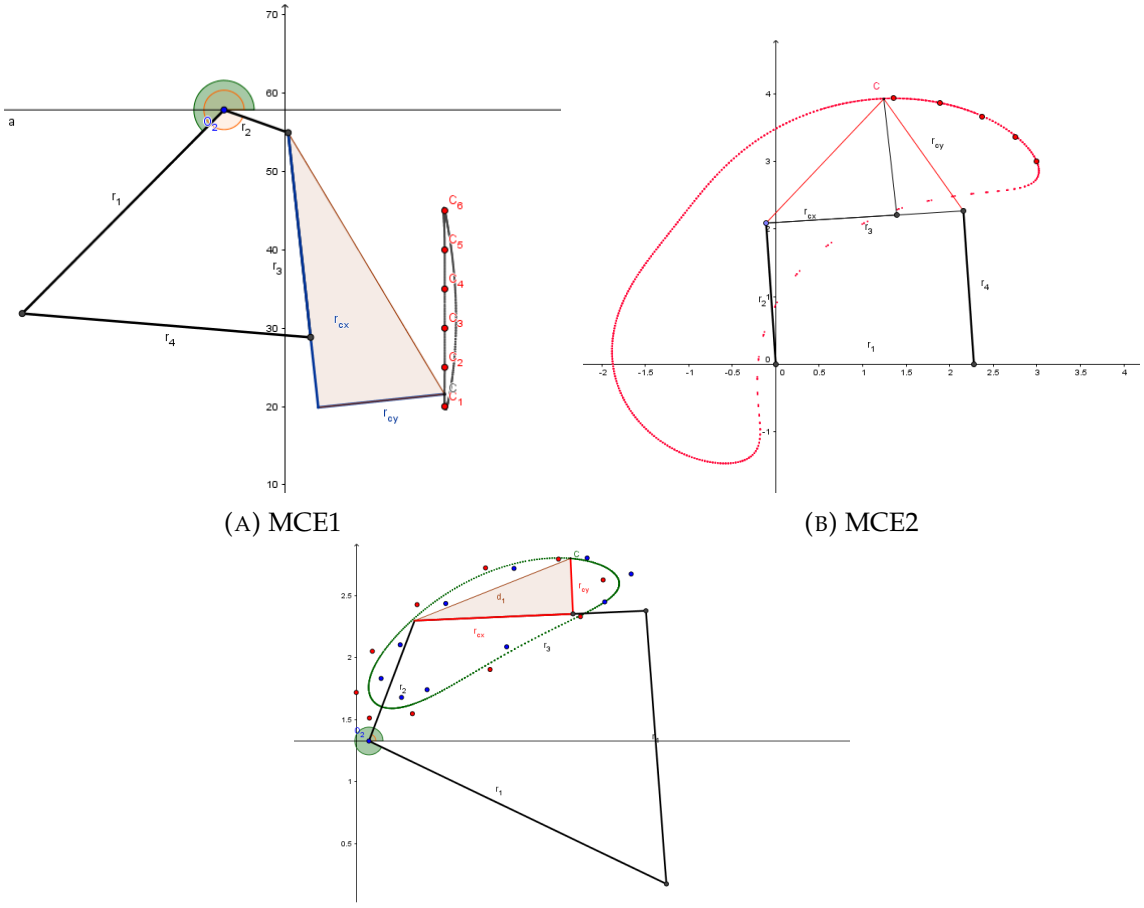
A partir de los vectores obtenidos por HNMED-V6 se realizaron las simulaciones correspondientes, cuyas graficas se presentan en la Figura 7.8. Se obtienen tres mecanismos diferentes para cada caso de estudio. EN el problema MCE1 el vector de diseño describe un mecanismo que sigue una trayectoria lineal vertical con una precisión cercan a cero. Para el caso MCE2 se obtiene un nuevo mecanismo que es capaz de seguir una trayectoria no alineada con mayor precisión ($f(\vec{x}) = \mathbf{2.5644E-04}$). Por último, se obtiene para el caso 3 un mecanismo similar a los ya encontrados, capaz de trasladar el acoplador entre los diez pares de puntos de precisión.

7.5.2. Efector Final de Tres Dedos

Los vectores de diseño obtenidos mediante para los dos casos de estudio de la Síntesis Óptima del Efector Final de Tres Dedos se muestran en la Tabla 7.19 . A partir de estos vectores se realizaron las simulaciones correspondientes cuyas gráficas se presentan en la Figura 7.9. Los resultados encontrados para el caso 1 confirman la multimodalidad del función objetivo de este problema, ya que se obtuvieron vectores diferentes con el mismo valor de $F(\vec{x})$. En la Tabla 7.19 se muestran en la columna correspondiente a HNMED-V6 dos de las soluciones óptimas encontradas (de izquierda a derecha $\vec{x}_1$ y $\vec{x}_2$). Se puede observar en las subfiguras A) y B) (Figura 7.9) la diferencia entre las longitudes de las barras de ambos efectores que alcanzan el valor de $F(\vec{x}) = 0$. Es importante destacar que

TABLA 7.18: Vectores de diseño obtenidos por HNMED-V6 y C-LSHADE para los tres casos de estudio del mecanismo de cuatro barras, correspondientes a la mejor observación de una muestra de ejecuciones independientes.

Variables	**MCE1**		**MCE2**		**MCE3**	
	HNMED-V6	**C-LSHADE**	**HNMED-V6**	**C-LSHADE**	**HNMED-V6**	**C-LSHADE**
r_1	36.2098	38.4576	2.2753	14.3145	2.6145	2.6154
r_2	8.5122	8.5384	2.08576	2.2111	1.0347	1.0349
r_3	26.2861	28.1572	2.2753	14.3145	1.8268	1.8275
r_4	36.1337	38.4020	2.2753	14.3145	2.2078	2.2081
r_{cx}	35.2850	37.8397	1.5039	2.1743	1.2509	1.2514
r_{cy}	15.8562	16.6131	1.7259	0.0222	0.4473	0.4468
θ_0	3.94255	3.9498	NA	NA	5.8267	5.8270
x_0	-7.5517	-9.4736	NA	NA	0.0991	0.0984
y_0	57.8675	59.4502	NA	NA	1.3287	1.3289
θ_2^1	1.6656	1.7537	NA	NA	0.4102	0.4098
θ_2^2	2.4331	2.4668	NA	NA	1.0393	1.0390
θ_2^3	2.9521	2.9766	NA	NA	1.6500	1.6495
θ_2^4	3.4537	3.4721	NA	NA	2.2600	2.2594
θ_2^5	4.0026	4.0196	NA	NA	2.8660	2.8652
θ_2^6	5.3124	5.1246	NA	NA	3.4902	3.4893
θ_2^7	NA	NA	NA	NA	4.1639	4.1628
θ_2^8	NA	NA	NA	NA	4.9055	4.9047
θ_2^9	NA	NA	NA	NA	5.4165	5.4157
θ_2^{10}	NA	NA	NA	NA	6.0676	6.0670
FO	3.7865E-29	0	2.5644E-04	2.6280E-03	2.7496E-01	2.7496E-01



(A) MCE1

(B) MCE2

(C) MCE3

FIGURA 7.8: Mecanismos correspondientes a las mejores soluciones encontradas por el algoritmo HNMED-V6 en los tres casos de estudio de la Síntesis Óptima del Mecanismo de Cuatro Barras.

TABLA 7.19: Vectores de diseño por HNMED-V6 y C-LSHADE para los dos casos de estudio del efector final de tres dedos, correspondientes al la mejor observación de una muestra de ejecuciones independientes.

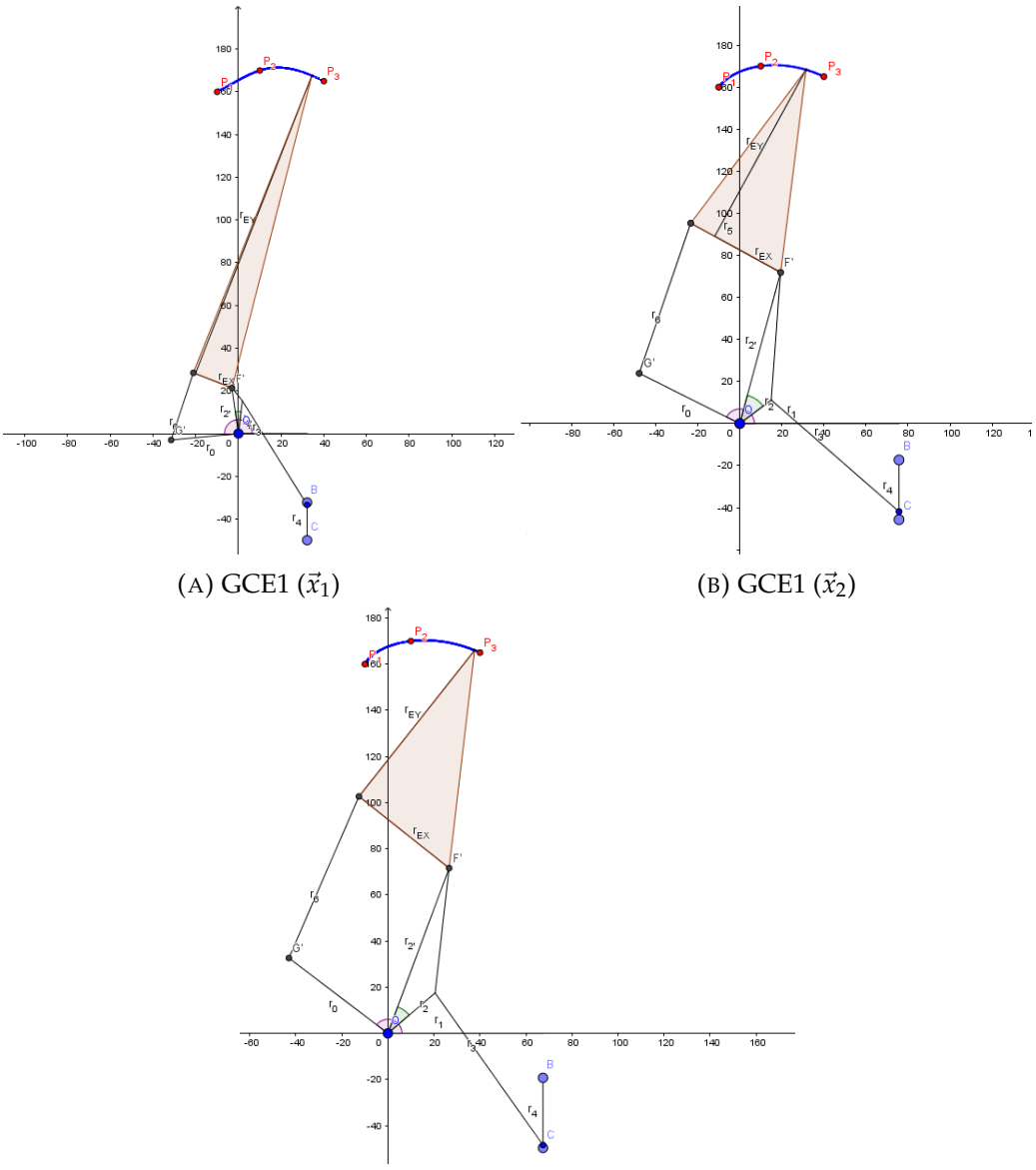
Variables	**GCE1**			**GCE2**	
	HNMED-V6		**C-LSHADE**	**HNMED-V6**	**C-LSHADE**
r_1	32.1665	75.7315	93.3515	67.3809	67.3809
r_2	15.4153	18.7233	24.8445	26.9523	26.9523
r_3	57.3403	80.6528	91.1652	80.8570	80.857
r_4^1	-49.9221	-17.3087	-10.7608	-19.3905	-8.5852
r_4^2	-41.8768	-31.5388	-26.3809	-34.0277	-25.1694
r_4^3	-32.2552	-45.6307	-41.9142	-49.6243	-41.3845
r_0	31.6542	53.4419	61.6546	53.9841	53.9841
r_2'	21.5820	74.4829	117.6764	76.3699	76.3699
r_5	19.3986	48.7177	37.8306	50	50
r_6	33.1513	75.4862	128.4622	76.3699	76.3699
r_{EX}	18.3572	35.7005	12.18	50	50
r_{EY}	149.8560	90.3028	52.5227	80.8429	80.8429
α	0.26237	0.6646	0.6639	0.5093	0.2618
θ_0	3.2378	2.6777	3.0013	2.4927	2.4927
FO	0	0	0	0.11385	0.11385

aunque sólo se muestran dos vectores diferentes, en el experimento F por ejemplo, se observaron 11 vectores de diseño diferentes con el valor de $F(\vec{x}) = 0$. Estos resultados experimentales aportan nuevos conocimientos sobre la naturaleza de estas funciones.

Además, la multimodalidad permite obtener diferentes soluciones óptimas que proporcionan un espectro más amplio para la toma de decisiones por parte de los diseñadores. En el problema GCE2, se obtiene un mecanismo resultante de aplicar la fórmula de normalización de barras, la cual concibe la estética del mecanismo y logra un diseño más antropomorfo que en GCE1. Esto limita al mecanismo para alcanzar una precisión mayor, a diferencia del caso 1 donde se alcanza la precisión máxima con error igual cero.

7.5.3. Microrred Aislada

En el problema de la Micro-red Eléctrica Aislada se logró minimizar el costo de generación de energía eléctrica durante cada hora del día, alcanzándose un costo total de **-5.5147E+05**, superior al alcanzado por C-LSHADE, LSHADE-CV y ED/-rand/1/bin. El valor negativo significa el ahorro monetario alcanzado, debido a



(A) GCE1 ($\vec{x}_1$)

(B) GCE1 ($\vec{x}_2$)

(C) GCE2

FIGURA 7.9: Efectores finales correspondientes a las mejores soluciones encontradas por el algoritmo HNMED-V6 en los dos casos de estudio de la Síntesis Óptima de un Efector Final de Tres Dedos.

TABLA 7.20: Vectores de diseño obtenidos por HNMED-V6 para la microrred aislada en las 24 horas del día, correspondientes al la mejor observación de una muestra de ejecuciones independientes.

t	P1(Diesel)	P2(ESS)	P3(Solar)	P4 (Eólica)	FO_t (HNMED-V6)	FO_t (LSHADE-CV)
0	2496.2999	1.7	1	1	744.7866	744.7866
1	1755.8416	341.2007	1.1127E-03	402.9565	-4753.1500	-6167.8006
2	1704.1952	394.8047	0.9999	749.9999	-16558.9336	-21918.5478
3	2232.9547	116.0452	1	600	-18862.8782	-13517.0085
4	1964.1355	344.2916	0.9995	540.5732	-9947.9769	-15310.5979
5	1939.3028	319.8534	0.07404	240.7696	877.3945	6780.1853
6	1363.3840	437.2674	0.9937	348.3547	-109.7044	-2731.4501
7	1983.9986	1.7474E-04	265.9997	1.4022E-03	-35662.9968	-35701.6365
8	2228.9763	2.3807E-02	69.9998	0.9999	-8864.0039	-8864.7210
9	1786.2962	165.0891	368.6145	6.4863E-10	-44829.1086	-30445.4635
10	2283.5514	45.4304	21.0178	2.2889E-04	-770.1488	-16260.0739
11	1897.9986	826.2181	125.9999	99.7831	4142.6902	-9881.8426
12	884.9261	500.9353	692.8439	171.2945	-85880.2563	-70753.5073
13	525.7193	1098.3992	693.2158	2.6656	-61804.5468	-27202.5115
14	2.6908E-02	1099.7761	606.4617	643.7351	-74471.8475	-85010.2124
15	210.1557	1019.5926	559.9858	560.2657	-67319.5302	-71276.2124
16	61.6439	1040.7287	405.8161	941.8111	-60257.4986	-56471.1139
17	921.2967	1023.9955	62.9982	1141.7095	-21515.5577	-22707.6927
18	915.8894	1099.9959	0.9997	1293.1147	-16585.2094	-16585.2296
19	2398.1511	850.8494	0.9993	999.9999	-12168.3814	-12168.4505
20	3747.7705	1.2300	0.9997	499.9996	-17364.3302	-17399.8776
21	2409.9582	40.0413	3.1178E-04	550	-18977.4352	-19017.7949
22	2449.4718	208.8747	1.9987E-02	291.6334	-4075.7627	-4318.2607
23	1304.2459	1100	0.9999	244.7540	23536.9265	23676.9774
Total					-5.5147E+05	-5.3250E+05

la mayor utilización de las Fuentes de Energía Renovable(FER) durante el despacho en el día. La Tabla 7.20 se muestra los valores encontrados para las variables de diseño en cada hora del día. A modo de comparación se agregaron los valores obtenidos por el algoritmo LSHADE-CV el cual obtuvo mejores resultados que C-LSHADE para este problema en [13]. Se puede observar que en la hora 12 HNMED-V6 obtiene un mayor ahorro reduciendo la generación mediante diésel y ESS y aumentando la generación por medio de energía solar y eólica respecto a la hora anterior.

A partir de estos vectores se generó la Figura 7.10 en la que se muestra la potencia que alcanza cada uno de los generadores en respuesta a la demanda (Carga del sistema). Como se puede observar, el sumunistro mediante el Generador de Díesel (P1) tiene la mayor asignación de carga durante las primeras y últimas horas del día, sin embargo a partir de la hora 12 a la 18, el sumunistro es asumido mayormente por las FER permitiendo alcanzar el un ahorro significativo en comparación con el alcanzado por LSHADE-CV,C-LSHADE y ED/rand/1/bin.

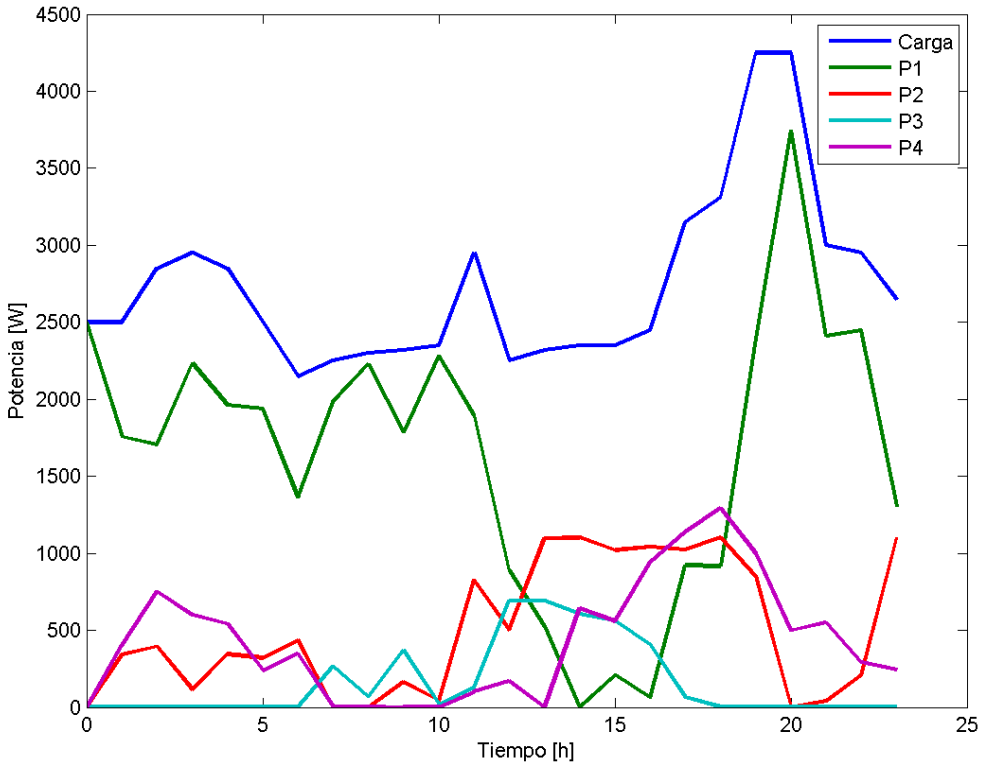


FIGURA 7.10: Visualización del suministro de energía durante las 24 horas del día para el funcionamiento de la microrred aislada, obtenido por el algoritmo HNMED-V6.

7.6. Conclusiones del capítulo

En el presente capítulo se describió el diseño experimental llevado a cabo para implementar la propuesta de solución. Este proceso de experimentación se llevo a cabo en dos etapas: una fase de experimentos preliminares que tuvo como objetivo validar la competitividad del enfoque de hibridación propuesto y evaluar el comportamiento de las primeras 5 variantes presentadas. Los resultados obtenidos en experimentos preliminares sirvieron para guiar el diseño del algoritmo HNMED-V6 el cual incluye una estrategia de inicialización de los símpleces y el buscador global se aplica a un subconjunto élite mayor durante el proceso de minimización. Los experimentos finales muestran un desempeño competitivo de esta versión con respecto a los algoritmos ED/rand/1/bin y C-LSHADE. Sin embargo HNMED-V6 utiliza un número inferior de evaluaciones de la función objetivo.

CAPÍTULO 8

Conclusiones y trabajos futuros

En el presente trabajo de investigación se propuso un nuevo enfoque de hibridación que tiene como objetivo aumentar la sinergia entre los buscadores local y global así como garantizar un mayor balance entre la operaciones de exploración y la explotación del espacio de búsqueda. El enfoque comprende siete lineamientos de diseño que pueden servir como marco de trabajo para futuros trabajos de hibridación. El lineamiento básico del enfoque plantea realizar una distribución de varias instancias de un buscador local en puntos aleatorios del espacio de búsqueda para garantizar la exploración. Sin embargo los operadores de los métodos de búsqueda local sólo explotan una vecindad relativamente pequeña del punto de inicio; por lo tanto se propone que un buscador global se encargue indicar a las diferentes instancias del buscador local, hacia dónde se encuentran las regiones más prometedoras. Estos lineamientos están motivados por las deficiencias encontradas en los trabajos de hibridación encontrados en la literatura especializada.

Teniendo en cuenta el enfoque de hibridación propuesto se procedió a realizar el diseño experimental de la propuesta de solución, donde se seleccionó como método de programación matemática el algoritmo de Nelder-Mead y como buscador global el algoritmo de Evolución Diferencial. Los primeros experimentos consistieron en introducir operadores que incluían aleatoriedad en el procedimiento original del Nelder-Mead. Así, se diseñaron tres variantes que realizan expansiones de longitud aleatoria (NM-ELA) cuando ninguno de los operadores originales de reflexión, expansión y contracción mejoran al peor punto del simplex. Los resultados de las pruebas de Friedman y Bonferroni-Dunn indicaron un

aumento significativo del desempeño del NM con los operadores ELA para los 5 problemas de diseño cinemático.

Finalmente, se diseñaron cinco variantes iniciales que toman como algoritmo de búsqueda local al método Nelder-Mead. El procedimiento general consiste en dividir una población inicial generada aleatoriamente en NS símpleces sobre los cuales cada instancia del Nelder Mead realizara una iteración por generación. El buscador global en este caso es el algoritmo de Evolución diferencial, el cual trabaja sobre el mejor punto de cada símplex. Los experimentos preliminares sobre estas variantes mostraron dificultades en los problemas de mayor dimensión. Por lo tanto, se diseñó la variante 6 que presenta una mayor capacidad de exploración y un mejor desempeño en forma general. Esto quedó evidenciado en los experimentos finales donde muestra un desempeño competitivo ante el algoritmo C-LSHADE y la ED/rand/1/bin.

Cada variante fue aplicada a seis problemas de optimización de diseño mecatrónico obteniendo resultados competitivos utilizando un número inferior de evaluaciones respecto a las variantes basadas en ED. Se obtuvieron nuevas soluciones que difieren en cuanto a forma y calidad de las ya reportadas en la literatura especializada. A continuación se describen las principales observaciones sobre el presente trabajo de investigación.

8.1. Observaciones Finales

1. El enfoque propuesto permite diseñar algoritmos con mayor capacidad de exploración de diferentes regiones del espacio de búsqueda las cuales son localmente minimizadas por las instancias de los buscadores locales.
2. A diferencia de los enfoques mayormente utilizados en el estado del arte, el buscador local tiene un papel preponderante en el proceso de minimización.
3. El buscador global debe ser aplicado a un subconjunto élite de la población para garantizar mayor rapidez de convergencia.
4. Las variantes propuestas utilizan diferentes instancias del método Nelder-Mead modificado con un operador que concibe información global del espacio de búsqueda. La Evolución Diferencial es utilizada como buscador global, actuando sobre los mejores puntos de cada símplex.
5. La variante HNMED-V6 aplica una estrategia de inicialización de los símpleces, ubicando los símpleces iniciales en la vecindad de los límites del espacio de búsqueda. Esta variante contempla la naturaleza geométrica del método Nelder-Mead, por lo que genera símpleces iniciales más regulares y de mayor tamaño.

6. Se obtuvieron resultados competitivos en todos los problemas de optimización de diseño mecatrónico utilizando un número significativamente inferior a los ya reportados en la literatura para los seis problemas de optimización. En el caso de HNMED-V6 se utilizan menos símpleces iniciales ya que la los vértices iniciales se encuentran a mayor distancia por lo tanto son capaces de muestrear mayores áreas del espacio de búsqueda. Esto constituye una disminución de la complejidad temporal como de espacio en memoria.
7. Finalmente se obtuvieron nuevas soluciones para tres de los seis problemas de optimización. En el caso MCE2 se encontró una nueva solución que describe un mecanismo diferente que alcanza una trayectoria con menor error respecto a los puntos de precisión. Para el caso 1 del gripper se obtuvieron vectores distantes que alcanzan el valor de $F(\vec{x}) = 0$, lo que demuestra la naturaleza multimodal de la función objetivo obtenida durante el diseño cinemático del mecanismo. Estos resultados demuestran la complejidad de estos problemas, sobre todo los de diseño cinemático.

8.2. Objetivos cumplidos

El objetivo general de la presente investigación fue diseñar un Algoritmo Híbrido de Programación Matemática con elementos de Algoritmos Evolutivos para resolver problemas no lineales de optimización de Diseño Mecatrónico. El cual fue cumplido, ya que bajo el enfoque de hibridación propuesto se generaron seis algoritmos competitivos que utilizan un número inferior de evaluaciones de la función objetivo respecto a los encontrados en la literatura especializada. Para cumplir el objetivo general se satisficieron los siguientes objetivos específicos:

1. Estudiar el comportamiento de al menos dos algoritmos de programación matemática al resolver problemas de diseño mecatrónico.
2. Determinar bondades y deficiencias de los métodos de programación matemática.
3. Diseñar un Algoritmo Híbrido de programación matemática con elementos de algoritmos evolutivos para mejorar la búsqueda.
4. Aplicar el Algoritmo Híbrido a problemas de optimización de diseño mecatrónico.
5. Realizar pruebas de medición de desempeño mediante el análisis estadístico y medición del número de evaluaciones del algoritmo propuesto.
6. Comparar los resultados obtenidos con el desempeño de los algoritmos identificados en la literatura especializada.

8.3. Contribuciones

Las contribuciones del presente trabajo se dividen en dos vertientes fundamentales. La primera es proveer a la comunidad científica de un nuevo enfoque de diseño para la hibridación entre métodos de búsqueda local y global, que permite la implementación de algoritmos eficientes para problemas de optimización no lineales con restricciones. Por otra parte, los algoritmos propuestos permiten una optimización eficaz y eficiente de un conjunto diverso de problemas mecatrónicos complejos; por lo que quedan a disposición de los investigadores dedicados al diseño mecatrónico para futuros problemas de este tipo.

8.4. Trabajo Futuro

De acuerdo a los resultados obtenidos durante la investigación se contemplan los siguientes trabajos futuros:

1. Aplicar el enfoque de hibridación en el diseño de algoritmos que combinen otros métodos de búsqueda local y metaheurísticas para la búsqueda global.
2. Continuar investigando el presente conjunto de problemas de optimización de diseño mecatrónico para alcanzar mejores resultados en los problemas MCE2, GCE2 y SCE1.
3. Aplicar el enfoque de hibridación para la resolución de problemas de optimización no lineales con restricciones que permitan una validación general del mismo.
4. Aplicar las variantes propuestas a nuevos problemas de optimización no lineales de diseño mecatrónico.
5. Comprobar el desempeño de las variantes con otros mecanismos para el manejo de restricciones y reglas de acotamiento para las variables de diseño.
6. Realizar un análisis detallado que incluya estadísticas sobre las operaciones realizadas por cada buscador durante la minimización como conteo de aplicación de operadores, mejora de la función objetivo por operador, evaluación de la diversidad de la población entre otras.
7. Realizar un estudio sobre la configuración de parámetros del algoritmo HNMED-V6.

APÉNDICE A

Implementación de variantes de HNMED en Matlab

Teniendo en cuenta que las variantes con mejor desempeño fueron HNMED-V5 y HNMED-V6 se describirá a detalle su codificación en Matlab. La implentación de todos los algoritmos esta concebida para que estos sean ejecutados dentro de un entorno, en el cual se abstrae la implementación del algoritmo del problema de optimización que se resuelve. En la URL: `https://github.com/roydes/HNMED-Implementacion-y-Pruebas`, se encuentra la implementación de los algoritmos y los experimentos realizados. Se puede observar que el entorno donde se ejecuta el algoritmo configura el valor de N, los límites *bounds* y número de evaluaciones NE antes de la ejecución del algoritmo para el i-ésimo problema de optimización. Las funciones $get\_problem\_n()$ y $get\_problem\_bound()$ acceden a variables globales de Ns (arreglo de número de variables por problema) y *bounds* (límites de las variables por problema).

La función $evaluate\_x\_i(x, n)$ llama a la función $f(x)$ para el problema actual y asigna a las posiciones $n + 1$ de x el valor de función objetivo y en la posición $n + 2$ de x la suma de violación de restricciones obtenidas. La función f(x) accede al número del problema actual y selecciona la función correspondiente. Para agregar nuevas funciones o alterar el orden de los problemas se debe modificar la estructura de control *switch* de la función.

La función de ordenamiento $sort\_by\_rules(X)$ utiliza el método de ordenación de inserción. Sin embargo, otros métodos pueden ser utilizados. Toma un arreglo de

vectores como parámetro y devuelve el arreglo ordenado de acuerdo a las reglas de Deb, las cuales fueron codificadas en la función boleana $compare\_by\_rules(x_1, x_2)$ descrita en la Sección A.3.

La variable E_F es un arreglo de contiene tuplas $[E, F(best)]$). La función $update\_E\_F(E, E, best)$ agrega en la última posición del arreglo una tupla con el valor actual de E y $F(best)$ actuales. El arreglo es devuelto por la función $HNMED$para generar la gráfica de convergencia de la j-ésima ejecución.

Por último las funciones auxiliares más importantes son descritas secciones posteriores. Para más detalles sobre la ejecución de los algoritmos de forma individual y de los experimentos descritos en el Capítulo 7, leer el archivo README.txt que se encuentra en la carpeta raíz del repositorio.

A.1. Codificación de HNMED-V5

```
function [best,E_F]= HNMEDV5(NE,NS)
% Inciando generador de números aleatorios.
rng('shuffle','twister');
% Seleccionando parámetros de expansión,
% contracción y reflexión.
gamma=1.05;
beta=0.5;
alpha=2;
% Asignando la n para el problema actual.
n= get_problem_n();
% Asignando los límites de las variables.
bounds=get_problem_bound();
% Calculando el tamaño de la población.
NP=(n+1)*NS;
X=zeros(NP,n+2);
% Generando y evaluando la población.
X=generate_random_pop(NP,n,bounds);
X=evaluate_pop(X,NP,n);
best=X(1,:);
E=NP;
XL=zeros(NS,n+2);
E_F=[];
% Comienza el ciclo hasta llegar al número
% máximo de evaluaciones
while E<NE
    k=1;
    w=1;
```

```
28      %Se generan los factores
29      %de escala y crizamiento
30      [F,CR]=generate_F_and_CR();
31      while w<NP
32          %NELDER MEAD SONBRE SIMPLEX SK
33          %Seleccionar Sk de X
34          Sk=X(w:(w+n),:);
35          %Ordenea ascendentemente de acuerdo
36          %a las reglas de Deb
37          Sk=sort_by_rules(Sk);
38          %Seleccionar el mejor punto
39          xl=Sk(1,:);
40          %Seleccionar el segundo peor punto
41          xg=Sk(n,:);
42          %Seleccionar el peor punto
43          xh=Sk(n+1,:);
44          %Calcular el centroide
45          xc= sum(Sk(1:n,:))/(n);
46          %_______Reflexión_____________________
47          xr=alpha*xc(1:n)-xh(1:n);
48          xr=limit(xr(1:n),bounds);
49          xr=evaluate_x_i(xr,n);
50          xnew=xr;
51          E=E+1;
52          if compare_by_rules(xr,xl)
53          %______Expansión______________________
54              xnew= (1+gamma)*xc(1:n)-gamma*xh(1:n);
55          elseif ~compare_by_rules(xr,xh)
56          %______Contracción hacia fuera______
57              xnew= (1-beta)*xc(1:n)+beta*xh(1:n);
58          elseif compare_by_rules(xg,xr) && compare_by_rules(
              xr,xh)
59          %______Contracción hacia dentro______
60              xnew= (1+beta)*xc(1:n)-beta*xh(1:n);
61          end
62          xnew(1:n)=limit(xnew(1:n),bounds);
63          xnew=evaluate_x_i(xnew(1:n),n);
64          E=E+1;
65          if compare_by_rules(xnew,xh)
66             Sk(n+1,:)=xnew;
67          else
68          %Si ninguno de los operadores originales
69          %mejora a xnew se aplica el operador de V5
```

```
70              [xr1,xr2,xr3]=generate_xr1_xr2_xr3(w,X);
71              xnew=xr1(1:n)+F*(best(1:n)-xr1(1:n))+(1-F)*(xr2
                    (1:n)-xr3(1:n));
72              xnew=limit(xnew(1:n),bounds);
73              xnew=evaluate_x_i(xnew,n);
74              E=E+1;
75              Sk(n+1,:)=xnew;
76          end
77          if compare_by_rules(xnew,xl)
78              xl=xnew;
79          end
80          XL(k,:)=xl;
81          if compare_by_rules(xnew,best)
82              best=xnew;
83          end
84          %EVOLUCIÓN DIFERENCIAL SE APLICA A XL(K)
85          [xr1,xr2,xr3]=generate_xr1_xr2_xr3(w,X);
86          parent=XL(k,:);
87          v=xr1(1:n)+F*(xr2(1:n)-xr3(1:n));
88          v=limit(v,bounds);
89          u=zeros();
90          jrand=randi(n,1);
91          for j=1:n
92              randj=rand();
93              if(randj<CR || jrand==j)
94                  u(j)=v(j);
95              else
96                  u(j)=parent(j);
97              end
98          end
99          u= evaluate_x_i(u,n);
100         E=E+1;
101         if compare_by_rules(u,parent)
102             Sk(1,:)=u;
103             XL(k,:)=u;
104         end
105         if compare_by_rules(u,best)
106             best=u;
107         end
108         %SE ACTUALIZA POBLACIÓN con Sk
109         X(w:w+n,:)=Sk;
110         k=k+1;
111         w=w+n+1;
```

```
112         end
113         %Se almacena el valor de la F(x) de best y E para
114         %Gráficas de convergencia
115         [E_F]=uptdate_E_F(E,E_F,best);
116     end
117
118     end
```

A.2. Codificación de HNMED-V6

```
1   function [best,E_F]= HNMEDV6(NE,NS)
2   %Se seleccionan los parámetros de expansión,
3   %contracción y reflexión.
4   gamma=1.7;
5   beta=0.3;
6   alpha=2;
7   %Asignando la n para el problema actual.
8   n= get_problem_n();
9   %Asignando los límites de las variables.
10  bounds=get_problem_bound();
11  %Calculando el tamaño de la población.
12  NP=(n+1)*NS;
13  X=zeros(NP,n+2);
14  %Generando y evaluando la población.
15  X=generate_border_simplex(NS);
16  %Inciando generador de números aleatorios.
17  rng('shuffle','twister');
18  best=X(1,:);
19  E=NP;
20  XL=zeros(NS,n+2);
21  E_F=[];
22  m=NS+1;
23  %Comienza el ciclo hasta llegar al número
24  %máximo de evaluaciones
25  while E<NE
26      k=1;
27      w=1;
28      %m={1,2,...,NS}
29      if m>NS
30          m=1;
31      end
32      %Se generan los factores
33      %de escala y crizamiento
```

```
34      [F,CR]=generate_F_and_CR();
35      while w<NP
36          %NELDER MEAD SONBRE SIMPLEX SK
37          %Seleccionar Sk de X
38          Sk=X(w:(w+n),:);
39          %Ordenea ascendentemente de acuerdo
40          %a las reglas de Deb
41          Sk=sort_by_rules(Sk);
42        %Seleccionar el mejor punto
43          xl=Sk(1,:);
44          %Seleccionar el segundo peor punto
45          xg=Sk(n,:);
46          %Seleccionar el peor punto
47          xh=Sk(n+1,:);
48          %Calcular el centroide
49          xc= sum(Sk(1:n,:))/(n);
50          %_______Reflexión_____________________
51          xr=alpha*xc(1:n)-xh(1:n);
52          xr=limit(xr,bounds);
53          xr=evaluate_x_i(xr,n);
54          xnew=xr;
55          E=E+1;
56          if compare_by_rules(xr,xl)
57          %______Expansión_____________________
58              xnew= (1+gamma)*xc(1:n)-gamma*xh(1:n);
59          elseif ~compare_by_rules(xr,xh)
60          %______Contracción hacia fuera______
61              xnew= (1-beta)*xc(1:n)+beta*xh(1:n);
62          elseif compare_by_rules(xg,xr) && compare_by_rules(
              xr,xh)
63          %______Contracción hacia dentro______
64              xnew= (1+beta)*xc(1:n)-beta*xh(1:n);
65          end
66          xnew(1:n)=limit(xnew(1:n),bounds);
67          xnew=evaluate_x_i(xnew(1:n),n);
68          E=E+1;
69          if compare_by_rules(xnew,xh)
70              Sk(n+1,:)=xnew;
71          else
72          %Si ninguno de los operadores originales
73          %mejora a xnew se aplica el operador de V5
74              [xr1,xr2,xr3]=generate_xr1_xr2_xr3(w,X);
```

```
75          xnew=xr1(1:n)+F*(best(1:n)-xr1(1:n))+(1-F)*(xr2
                (1:n)-xr3(1:n));
76          xnew=limit(xnew(1:n),bounds);
77          xnew=evaluate_x_i(xnew,n);
78          E=E+1;
79          Sk(n+1,:)=xnew;
80        end
81        if compare_by_rules(xnew,xl)
82          xl=xnew;
83        end
84        XL(k,:)=xl;
85        X(w:w+n,:)=Sk;
86        if compare_by_rules(xnew,best)
87          best=xnew;
88        end
89        %EVOLUCIÓN DIFERENCIAL SE APLICA A S(m)
90        [xr1,xr2,xr3]=generate_xr1_xr2_xr3(w,X);
91        parent=Sk(m,:);
92        %Se selecciona un invidiuo aleatoriamente
93        %de XL (los mejores puntosde cada Simplex)
94        rk=randi(NS);
95        xlk=XL(rk,:);
96        v=parent(1:n)+F*(xlk(1:n)-xr1(1:n))+(1-F)*(xr2(1:n)
            -xr3(1:n));
97        v=limit(v,bounds);
98        u=zeros();
99        jrand=randi(n,1);
100       for j=1:n
101         randj=rand();
102         if(randj<CR || jrand==j)
103           u(j)=v(j);
104         else
105           u(j)=parent(j);
106         end
107       end
108       u= evaluate_x_i(u,n);
109       E=E+1;
110       if compare_by_rules(u,parent)
111         Sk(m,:)=u;
112       end
113      if compare_by_rules(u,xl)
114         XL(k,:)=u;
115       end
```

```
            if compare_by_rules(u,best)
                best=u;
            end
            X(w:w+n,:)=Sk;
            k=k+1;
            w=w+n+1;

        end
        %Se incrementa m por generación
        m=m+1;
        %Se almacena el valor de la F(x) de best y E para
        %Gráficas de convergencia
        [E_F]=uptdate_E_F(E,E_F,best);
    end
end
```

A.3. Codificación de las reglas de Deb

```
function [selected]=compare_by_rules(x1,x2)
n=get_problem_n();
selected=false;
%Si ambos violan restricciones
%seleccionar el que tenga menor suma de SVR
if x1(n+2)~=0 && x2(n+2)~=0
    if x1(n+2) < x2(n+2)    %
        selected=true;
    end

else
%Si ninguno viola restricciones
%seleccionar al que tiene menor valor f(x)
    if x1(n+2)==0 && x2(n+2)==0
        if x1(n+1) < x2(n+1)
            selected=true;
        end
    end
end
%si un inviduo viola restricciones
%y el otro no seleccionar este último
if x2(n+2)>0 && x1(n+2)==0
    selected=true;
end
end
```

A.4. Codificación de la Estrategia de Inicialización de los Símplecess

```
function [X]=generate_border_simplex(NS)
n=get_problem_n();
bound=get_problem_bound();
%Generador aleatorio con distribucón uniforme
rng(0,'v5uniform');
%Límites para q_k
a=0.05;
b=3*n/100;
if b>0.95
   b=0.95;
end
%Límites para p_j
c=0.05;
d=5*n/100;
if d>0.95
   d=0.95;
end
v=1;
k=1;
S=zeros(n+1,n+2);
X=zeros(NS*(n+1),n+2);
while k<=NS
for i=1:n+1
%Generar q_k para cada simplex.
        q_k=a+(b-a)*rand(1);
%Situar para cada vertice i en la vecindad
%del límite supieror de la dimensión i
for j=1:n
    if i==j
        p_j=c+(d-c)*rand(1);
        S(i,j)=((bound(j,2)-bound(j,1))/2+((bound(j,2)-bound
            (j,1))/2)*p_j);
    else
        if rand()<0.5
            r=(NS*(n+1)/100*rand());
            S(i,j)=bound(j,1)+(((bound(j,2)-bound(j,1))/2)*
                q_k)+r;
        else
            r=(NS*(n+1)/100*rand());
```

```
38              S(i,j)=bound(j,1)+(((bound(j,2)-bound(j,1))/2)*
                    q_k) -r;
39          end
40      end
41  end
42  % Limitar y evaluar el vértice.
43  S(i,:)=limit(S(i,:),bound);
44  S(i,:)=evaluate_x_i(S(i,:),n);
45  end
46  % Agregar el simplex a la población
47  X(v:v+n,:)=S;
48  v=v+n+1;
49  k=k+1;
50  end
51  end
```

Referencias

[1] Clarence W. de Silva. *Mechatronics: A Foundation Course*. CRC Press. Google-Books-ID: aI7LBQAAQBAJ.

[2] Robert H. Bishop. *The Mechatronics Handbook, Second Edition - 2 Volume Set*. CRC Press. Google-Books-ID: GIEqBgAAQBAJ.

[3] Devdas Shetty and Richard A. Kolk. *Mechatronics System Design, SI Version*. Cengage Learning. Google-Books-ID: 1UsIAAAAQBAJ.

[4] Peter Fritzson. Basic Concepts. In *Introduction to Modeling and Simulation of Technical and Physical Systems with Modelica*, pages 1–28. John Wiley & Sons, October 2011.

[5] Edgar A. Portilla-Flores, Maria B. Calva-Yáñez, Miguel G. Villarreal, and Cervantes, Miguel G. V. Dynamic approach to Optimum Synthesis Of A Four-Bar Mechanism using a Swarm Intelligence Algorithm. *Kybernetika*, 5:786 – 803.

[6] Efrén Mezura-Montes, Edgar A. Portilla-Flores, and Betania Hernández-Ocaña. Optimum synthesis of a four-bar mechanism using the modified bacterial foraging algorithm. *International Journal of Systems Science*, 45(5):1080–1100.

[7] Betania Hernández-Ocaña, Ma Del Pilar Pozos-Parra, Efrén Mezura-Montes, Edgar Alfredo Portilla-Flores, Eduardo Vega-Alvarado, and Maria Bárbara Calva-Yánez. Two-Swim Operators in the Modified Bacterial Foraging Algorithm for the Optimal Synthesis of Four-Bar Mechanisms, 2016. DOI: 10.1155/2016/4525294.

[8] Edgar A. Portilla-Flores, Efrén Mezura-Montes, Jaime Álvarez Gallegos, Carlos A. Coello-Coello, and Carlos A. Cruz-Villar. Integration of structure and control using an evolutionary approach: An application to the optimal concurrent design of a CVT. *Int. J. Numer. Meth. Engng.*, 71(8):883–901.

[9] Miguel G. Villarreal-Cervantes and Efren Mezura-Montes. Control adaptable basado en evolucion diferencial para un motor de CD.

[10] Efren Mezura-Montes, Edgar-Alfredo Portilla-Flores, and Edith Capistran-Gumersindo. Dynamic Parameter Control in Differential Evolution with Combined Variants to Optimize a Three-Finger End Effector.

[11] A.D.J. Chica Leal, C.L. Trujillo Rodríguez, and F. Santamaría Piedrahita. Optimización de la generación de energía en una Microrred aislada. September 2015.

[12] F. A. Heredia-Ramírez, E. Rivas-Trujillo, and J. A. Hernández-Mora. Optimal power flow in electrical microgrids. In *2014 IEEE Central America and Panama Convention (CONCAPAN XXXIV)*, pages 1–6, November 2014.

[13] Maury Faney Zapata Zapata. Control de Parámetros del Algoritmo Evolución Diferencial con Variantes Combinadas para la Solución de Problemas de Optimización en Mecatrónica. Master's thesis, Laboratorio Nacional de Informática Avanzada, Xalapa. Veracruz, 2017.

[14] Xin-She Yang, Zhihua Cui, Renbin Xiao, Amir Hossein Gandomi, and Mehmet Karamanoglu. *Swarm Intelligence and Bio-Inspired Computation: Theory and Applications*. Newnes. Google-Books-ID: J0VcBQxtcwsC.

[15] Neri Ferrante, Cotta Carlos, and Moscato Pablo. *Handbook of Memetic Algorithms*. Springer-Verlag, 2012.

[16] Eduardo Vega-Alvarado, Edgar Flores, Efrén Mezura-Montes, Leticia Flores Pulido, and Maria Bárbara Calva-Yáñez. A memetic algorithm applied to the optimal design of a planar mechanism for trajectory tracking. 53:83–90, 01 2016.

[17] Tanya M. Anandan. Our Autonomous Future with Service Robots.

[18] Devendra Parulekar. Why does your company need Mechatronics? *EY Library*.

[19] James R. Hagerty. Meet the New Generation of Robots for Manufacturing. *Wall Street Journal*, 2015-06-03T03:08:00.000Z.

[20] UBM Electronics. Design News - Mechatronics - Latest.

[21] Wagner F. Sacco, Hermes Alves Filho, Nélio Henderson, and Cassiano R. E. de Oliveira. A Metropolis algorithm combined with Nelder–Mead Simplex

applied to nuclear reactor core design. *Annals of Nuclear Energy*, 35(5):861–867, May 2008.

[22] Patrick Koch, Oliver Kramer, Günter Rudolph, and Nicola Beume. On the Hybridization of SMS-EMOA and Local Search for Continuous Multiobjective Optimization. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, GECCO '09, pages 603–610, New York, NY, USA, 2009. ACM.

[23] Raka Jovanovic, Sabre Kais, and Fahhad H. Alharbi. Cuckoo Search Inspired Hybridization of the Nelder-Mead Simplex Algorithm Applied to Optimization of Photovoltaic Cells. *Applied Mathematics & Information Sciences*, 10(3):961–973, May 2016. arXiv: 1411.0217.

[24] Emilio Frazzoli, Munther A. Dahleh, and Eric Feron. Real-Time Motion Planning for Agile Autonomous Vehicles. *Journal of Guidance, Control, and Dynamics*, 25(1):116–129.

[25] Guanfeng Liu and Zexiang Li. Real-time grasping-force optimization for multifingered manipulation: Theory and experiments. *IEEE/ASME Transactions on Mechatronics*, 9(1):65–77.

[26] OE. What is the Smart Grid?, 13/2/ 2014.

[27] Murat Ahat, Soufian Ben Amor, Marc Bui, Alain Bui, Guillaume Guérard, and Coralie Petermann. Smart Grid and Optimization. *American Journal of Operations Research*, pages 196–206.

[28] OpusOne. GridOS.

[29] Singiresu S. Rao and S. S. Rao. *Engineering Optimization: Theory and Practice*. John Wiley & Sons, July 2009. Google-Books-ID: YNt34dvnQLEC.

[30] Jorge Nocedal and Stephen J Wright. Numerical optimization 2nd, 2006.

[31] Andreas Antoniou and Wu-Sheng Lu. *Practical Optimization: Algorithms and Engineering Applications*. Springer Science & Business Media, December 2007. Google-Books-ID: Ek5SaNDlMn0C.

[32] Ashok D. Belegundu and Tirupathi R. Chandrupatla. *Optimization Concepts and Applications in Engineering*. Cambridge University Press, March 2011. Google-Books-ID: HZssYlzpxOEC.

[33] David G. Luenberger and Yinyu Ye. *Linear and Nonlinear Programming*. Springer. Google-Books-ID: _VwICgAAQBAJ.

[34] Rajesh Kumar Arora. *Optimization: Algorithms and Applications*. CRC Press, May 2015. Google-Books-ID: peVyCQAAQBAJ.

[35] Hoang Tuy. *Convex Analysis and Global Optimization | Hoang Tuy | Springer*. September 2015.

[36] George Dantzig. *Linear Programming and Extensions*. Princeton University Press, August 2016. Google-Books-ID: hUWPDAAAQBAJ.

[37] Samer Takriti. Ampl: A modeling language for mathematical programming, 1994.

[38] Gallier Jean and Quaintance Jocelyn. *Algebra, Topology, Differential Calculus, and Optimization Theory For Computer Science and Engineering*. University of Pennsylvania, 2017.

[39] Strang Gilbert. *Calculus*. Wellesley-Cambridge Press, 2nd edition edition, 2010.

[40] Soliman Abdel-Hady Soliman and Abdel-Aal Hassan Mantawy. Mathematical Optimization Techniques. In *Modern Optimization Techniques with Applications in Electric Power Systems*, Energy Systems, pages 23–81. Springer, New York, NY, 2012.

[41] W. Spendley, G. R. Hext, and F. R. Himsworth. Sequential Application of Simplex Designs in Optimisation and Evolutionary Operation. *Technometrics*, 4(4):441–461, November 1962.

[42] J. A. Nelder and R. Mead. A Simplex Method for Function Minimization. *The Computer Journal*, 7(4):308–313, January 1965.

[43] Kalyanmoy Deb. *Optimization for Engineering Design: Algorithms and Examples*. Prentice-Hall of India, February 2004. Google-Books-ID: JypoXt5hHrkC.

[44] Lawrence Blume and Carl P Simon. Mathematics for economists. *New York, London*, 1994.

[45] Kenneth A De Jong. *Evolutionary computation: a unified approach*. MIT press, 2006.

[46] Peter J Angeline. Adaptive and self-adaptive evolutionary computations. In *Computational intelligence: a dynamic systems perspective*. Citeseer, 1995.

[47] John Henry Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[48] Thomas Weise. Global optimization algorithms-theory and application. *Self-published*, 2, 2009.

[49] Kalyanmoy Deb. *Multi-objective optimization using evolutionary algorithms*, volume 16. John Wiley & Sons, 2001.

[50] Agoston E Eiben and Jim Smith. From evolutionary computation to the evolution of things. *Nature*, 521(7553):476, 2015.

[51] S. C. Chiam, C. K. Goh, and K. C. Tan. Issues of Binary Representation in Evolutionary Algorithms. In *2006 IEEE Conference on Cybernetics and Intelligent Systems*, pages 1–8, June 2006.

[52] Zbigniew Michalewicz. *Genetic Algorithms + Data Structures = Evolution*. Springer, 3rd edition, 1995.

[53] T. Okabe, Y. Jin, and B. Sendhoff. Evolutionary multi-objective optimisation with a hybrid representation. In *The 2003 Congress on Evolutionary Computation, 2003. CEC '03*, volume 4, pages 2262–2269 Vol.4, December 2003.

[54] John R. Koza. Genetic programming as a means for programming computers by natural selection. *Stat. Comput. (UK*, 4:191–198, 1994.

[55] David E. Goldberg and Kalyanmoy Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms*, pages 69–93. Morgan Kaufmann, 1991.

[56] Jayshree Sarma and Kenneth A De Jong. An analysis of local selection algorithms in a spatially structured evolutionary algorithm. In *ICGA*, pages 181–187. Citeseer, 1997.

[57] Jinghui Zhong, Xiaomin Hu, Min Gu, and Jun Zhang (corresponding. Comparison of performance between different selection strategies on simple genetic algorithms.

[58] Kalyanmoy Deb. An Efficient Constraint Handling Method for Genetic Algorithms. pages 311–338, 1998.

[59] Joerg Laessig, Karl Heinz Hoffmann, and Mihaela Enachescu. Threshold selecting: Best possible probability distribution for crossover selection in genetic algorithms. pages 2181–2186, January 2008.

[60] Kenneth Alan De Jong. Analysis of the behavior of a class of genetic adaptive systems. 1975.

[61] David B Fogel. Applying evolutionary programming to selected traveling salesman problems. *Cybernetics and systems*, 24(1):27–36, 1993.

[62] Ingo Rechenberg. Evolution strategy: Optimization of technical systems by means of biological evolution. *Fromman-Holzboog, Stuttgart*, 104:15–16, 1973.

[63] Hans-Paul Schwefel and Günter Rudolph. Contemporary evolution strategies. In *European conference on artificial life*, pages 891–907. Springer, 1995.

[64] John H Holland. Outline for a logical theory of adaptive systems. *Journal of the ACM (JACM)*, 9(3):297–314, 1962.

[65] John H Holland. Adaptive plans optimal for payoff-only environments. Technical report, MICHIGAN UNIV ANN ARBOR LOGIC OF COMPUTERS GROUP, 1969.

[66] Efrén Mezura-Montes, Carlos A Coello Coello, and Edy I Tun-Morales. Simple feasibility rules and differential evolution for constrained optimization. In *Mexican International Conference on Artificial Intelligence*, pages 707–716. Springer, 2004.

[67] Alden H Wright. Genetic algorithms for real parameter optimization. In *Foundations of genetic algorithms*, volume 1, pages 205–218. Elsevier, 1991.

[68] Kuk-Hyun Han and Jong-Hwan Kim. Genetic quantum algorithm and its application to combinatorial optimization problem. In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, volume 2, pages 1354–1360. IEEE, 2000.

[69] Stefano Nolfi, Josh C Bongard, Phil Husbands, and Dario Floreano. Evolutionary robotics., 2016.

[70] Xuewu Wang, Yingpan Shi, Dongyan Ding, and Xingsheng Gu. Double global optimum genetic algorithm–particle swarm optimization-based welding robot path planning. *Engineering Optimization*, 48(2):299–316, 2016.

[71] Ali R Yildiz. Comparison of evolutionary-based optimization algorithms for structural design optimization. *Engineering applications of artificial intelligence*, 26(1):327–333, 2013.

[72] Dipankar Dasgupta and Zbigniew Michalewicz. *Evolutionary algorithms in engineering applications*. Springer Science & Business Media, 2013.

[73] David H Myszka. *Machines and mechanisms*. Prentice Hall, 2004.

[74] Eric W. Weisstein. Euler Formula.

[75] Joseph Edward Shigley, John Joseph Uicker, José H Pérez, and Hortensia Corona de Contín. *Teoría de máquinas y mecanismos*. Number TJ145. S54 1983. McGraw-Hill México;, 1983.

[76] Á. Sánchez-Márquez, E. Vega-Alvarado, E. A. Portilla-Flores, and E. Mezura-Montes. Synthesis of a planar four-bar mechanism for position control using the harmony search algorithm. In *2014 11th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, pages 1–6, Sept 2014.

[77] Edith Capistran Gumersindo. Computo inspirado en la naturaleza para la optimización mono-objetivo de un efector final de tres dedos. Tesis de Maestría en Computación Aplicada.

[78] Robert H Lasseter. Microgrids. In *Power Engineering Society Winter Meeting, 2002. IEEE*, volume 1, pages 305–308. IEEE, 2002.

[79] S. Ramabhotla, S. Bayne, and M. Giesselmann. Economic dispatch optimization of microgrid in islanded mode. In *International Energy and Sustainability Conference 2014*, pages 1–5, Oct 2014.

[80] Henerica Tazvinga, Xiaohua Xia, and Jiangfeng Zhang. Minimum cost solution of photovoltaic–diesel–battery hybrid power systems for remote consumers. *Solar Energy*, 96:292–299, 2013.

[81] Henerica Tazvinga, Bing Zhu, and Xiaohua Xia. Energy dispatch strategy for a photovoltaic–wind–diesel–battery hybrid power system. *Solar Energy*, 108:412–420, 2014.

[82] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997.

[83] Swagatam Das and Ponnuthurai Suganthan. Differential evolution: A survey of the state-of-the-art. 15:4–31, 01 2011.

[84] Elena Dragoi and Vlad Dafinescu. Parameter control and hybridization techniques in differential evolution: a survey. *Artificial Intelligence Review*, 45, 12 2015.

[85] Günther R Raidl. A unified view on hybrid metaheuristics. In *International Workshop on Hybrid Metaheuristics*, pages 1–12. Springer, 2006.

[86] Natalio Krasnogor and James Smith. A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *IEEE Transactions on Evolutionary Computation*, 9(5):474–488, 2005.

[87] Saber M Elsayed, Ruhul A Sarker, and Daryl L Essam. Integrated strategies differential evolution algorithm with a local search for constrained optimization. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pages 2618–2625. IEEE, 2011.

[88] T Warren Liao. Two hybrid differential evolution algorithms for engineering design optimization. *Applied Soft Computing*, 10(4):1188–1199, 2010.

[89] Ferrante Neri and Ville Tirronen. On memetic differential evolution frameworks: a study of advantages and limitations in hybridization. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 2135–2142. IEEE, 2008.

[90] T Rogalsky and RW Derksen. Hybridization of differential evolution for aerodynamic design. In *Proceedings of the 8th annual conference of the computational fluid dynamics society of Canada*, pages 729–736, 2000.

[91] Ling Wang, Ye Xu, and Lingpo Li. Parameter identification of chaotic systems by hybrid nelder–mead simplex search and differential evolution algorithm. *Expert Systems with Applications*, 38(4):3238–3245, 2011.

[92] Sanja Singer and Saša Singer. Complexity analysis of nelder-mead search iterations. 06 2018.

[93] Lixing Han and Michael Neumann. Effect of dimensionality on the Nelder–Mead simplex method. *Optimization Methods and Software*, 21(1):1–16, February 2006.

[94] Thomas Bartz-Beielstein and Mike Preuss. Experimental analysis of optimization algorithms: Tuning and beyond. In *Theory and Principled Methods for the Design of Metaheuristics*, 2014.

[95] Richard Lowry. A first glance at the question of statistical significance. In *Concepts and Applications of Inferential Statistics*, chapter 4. 2014.

[96] Joaquín Derrac, Salvador García, Daniel Molina, and Francisco Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3–18, 2011.

[97] Ryoji Tanabe and Alex S Fukunaga. Improving the search performance of shade using linear population size reduction. In *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pages 1658–1665. IEEE, 2014.