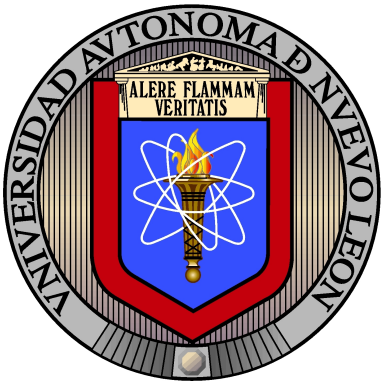
Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

División de Estudios de Posgrado



Estrategia evolutiva con caminatas aleatorias de Metrópolis para problemas de optimización global

Por

Jonás Velasco Álvarez

En opción al grado de

Doctor en Ingeniería

Con especialidad en

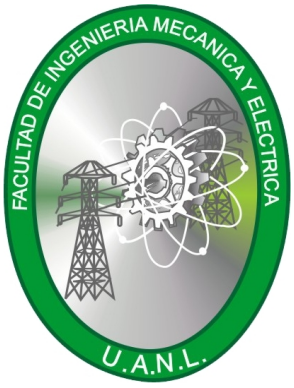
Ingeniería de Sistemas

Ciudad Universitaria, Julio de 2013

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

División de Estudios de Posgrado



Estrategia evolutiva con caminatas aleatorias de Metrópolis para problemas de optimización global

Por

Jonás Velasco Álvarez

En opción al grado de

Doctor en Ingeniería

Con especialidad en

Ingeniería de Sistemas

Ciudad Universitaria, Julio de 2013

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

División de Estudios de Posgrado

Los miembros del Comité de Tesis recomendamos que la Tesis «Estrategia evolutiva con caminatas aleatorias de Metrópolis para problemas de optimización global», realizada por el alumno Jonás Velasco Álvarez, matrícula 1437957, sea aceptada para su defensa como opción al grado de Doctor en Ingeniería con especialidad en Ingeniería de Sistemas.

El Comité de tesis

Dr. Arturo Berrones Santos
Director

Dr. Javier Almaguer Martínez
Revisor

Dr. Óscar Chacón Mondragón
Revisor

Dr. Héctor Flores Cantú
Revisor

Dr. Hugo Jair Escalante
Revisor

Vo. Bo.

Dr. Moisés Hinojosa Rivera
Subdirector
División de Estudios de Posgrado

Ciudad Universitaria, Julio de 2013

Dedicatoria

Este trabajo está dedicado a todos aquellos

a quien amo, especialmente

A mis padres, René y Rosa Martha.

A mis hermanos, René, Noé y Martha Elena.

A mi hija, Azul Zoé,

y a mi novia, Lorenia.

Agradecimientos

Quiero expresar mi especial agradecimiento a mi director de tesis el Dr. Arturo Berrones por su tiempo y apoyo dedicado a este trabajo. Por darme la oportunidad de integrarme a otros proyectos de investigación. Por su trato humano y sincero, por su paciencia y comprensión, pero, sobre todo, por confiar en mí y compartir sus conocimientos para formarme como investigador. Muchas gracias.

Al comité de tesis, el Dr. Javier Almaguer, el Dr. Óscar Chacón, el Dr. Hugo Escalante y al Dr. Hector Flores, por contribuir en la revisión de este trabajo, por sus sugerencias y comentarios que ayudaron a mejorar su calidad. Además, por su apoyo, paciencia y comprensión.

Al Dr. Javier Almaguer por brindarme su amistad, lealtad y confianza. Por compartir sus conocimientos y por su enorme apoyo en todos los aspectos. Por darme la oportunidad de integrarme a muchos de sus proyectos de investigación, pero sobre todo, por siempre estar ahí en los malos y mejor aún, los buenos momentos durante mis estudios. De igual forma, agradezco mucho al Dr. Óscar Chacón por su trato humano y sincero, por sus sabios consejos, pero sobre todo, por sus palabras de ánimo que me impulsó a seguir adelante. Agradezco al Dr. Javier Morales por sus consejos, por darme la oportundidad de trabajar en conjunto en proyectos muy interesantes, por compartir sus anécdotas, pero sobre todo, por sus palabras de ánimo, su apoyo y su amistad.

Agradezco a los profesores del Programa de Ingeniería de Sistemas que me enseñaron más allá de un libro cosas que aún hoy recuerdo y pongo en práctica. A mis compañeros del doctorado y maestría por su apoyo brindado cuando lo necesité. Gracias por su tiempo, dedicación y paciencia.

Agradezco a la Universidad Autónoma de Nuevo León y su Facultad de Ingeniería Mecánica y Eléctrica por el apoyo financiero para la realización de mis estudios.

Al Consejo Nacional de Ciencia y Tecnología por brindarme beca de manutención durante mis estudios de doctorado, además del apoyo económico proporcionado dentro del programa de Becas Mixtas que me permitió realizar la estacia de investigación en Donostia-San Sebastián, España.

Agradezco al Dr. Jose Antonio Lozano del Departamento de Ciencia de la Computación e Inteligencia Artificial de la Universidad del País Vasco, y a todo su equipo de trabajo, por el apoyo, el interés y las facilidades brindadas para la realización de mi estancia académica de investigación en Donostia-San Sebastián, España.

A mis amigos, en especial a Mario Hernandez, Josué Alcántar, Víctor Cervantes, Ramón A. Lopez, Xochitl Aquiahuatl y a Miguel A. Urbano, por confiar siempre en mí, por su apoyo y porque que sé que siempre puedo contar con ustedes.

Finalmente, agradezco profundamente a mis padres, René y Martha, a mis hermanos, René, Noé y Martha, y a mi princesa Azul Zoé, por estar siempre conmigo y por creer en mí, además por todo el amor que me han demostrado a lo largo de los años. En gran parte es gracias a ustedes que hoy puedo ver cumplida esta meta. A mi gran amor, Lorenia Robles, por su paciencia, apoyo y comprensión. Por estar ahí en los buenos y malos momentos durante este trayecto, pero sobre todo, por todo el amor y felicidad que me ha brindado.

Resumen

Este trabajo doctoral consiste en proponer una metodología novedosa y robusta para resolver problemas irrestrictos y restringidos de optimización global. La metodología consiste en integrar las ideas de los métodos de muestreo tipo Monte Carlo con las ideas escenciales de las estrategias evolutivas. La investigación versa sobre dos contribuciones principales: (1) Se propone una nueva forma de muestreo que permite explorar correctamente el espacio de búsqueda del problema mediante la distribución de salto de Cauchy. Los saltos de Cauchy aseguran que ninguna región del espacio tenga probabilidad cero de ser explorada. Además, el muestreador auto-adapta sus parámetros de acuerdo a la estructura del problema. El ajuste de estos parámetros ayudan a mantener un balance entre la exploración y explotación durante el muestreo, previniendo la convergencia hacia un óptimo local e incrementando la probabilidad de encontrar un óptimo global. (2) Se propone un novedoso método heurístico de búsqueda aleatoria para optimización global. Básicamente el método de muestreo propuesto se utiliza para generar un conjunto de soluciones candidatas. Después, las soluciones candidatas son utilizadas para extraer información acerca el espacio de búsqueda, ubicando las regiones más probables de encontrar óptimos locales o incluso el óptimo global. Finalmente, las regiones prometedoras son explotadas mediante un procedimiento de búsqueda local. La búsqueda local realiza movimientos finos en dichas zonas con el fin de mejorar localmente las soluciones candidatas encontradas.

El correcto desempeño del método de optimización propuesto se basa principalmente por tener las siguientes características:

- La capacidad de escapar de óptimos locales.
- La capacidad de saltar amplias regiones de baja probabilidad.
- La capacidad de converger rápidamente hacia la distribución de interés.
- La capacidad de explotar las regiones de alta probabildad.

El potencial del método de optimización propuesto es evaluado sobre dos conjuntos de prueba estandar para los algoritmos evolutivos, y comparado contra los algoritmos de referencia en el campo más destacados sobre dichos conjuntos de prueba. Los resultados obtenidos indican que la metodología propuesta es robusta ante un conjunto amplio de problemas y sobre las dimensiones probadas. Además, las soluciones obtenidas por la metodología son de buena calidad y muy comparable para todos los casos probados.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

Un problema de optimización global es el problema de encontrar una solución óptima global o cercana a la óptima dentro de una multiplicidad de óptimos locales, partiendo de un conjunto específico de soluciones factibles y utilizando una medida para evaluar cada solución individual. Un algoritmo que resuelve dicho problema se llama algoritmo de optimización. Los problemas de optimización global son de gran interés ya que surgen en numerosas áreas que modelan el mundo real, y se ha encontrado una amplia aplicación en el estudio de sistemas físicos, químicos, económicos y de ingeniería [1]. Por otro lado, los problemas de optimización global generalmente implican optimizar un gran número de variables de decisión, lo que los convierten en problemas retadores para los algoritmos generales de búsqueda. Inherente a esto, es necesario diseñar algoritmos robustos capaces de resolver problemas con diferentes características dentro de cada campo.

Los algoritmos de búsqueda aleatoria son uno de los pilares de la mayoría de los métodos heurísticos utilizados en optimización global. Básicamente los algortimos heurísticos de búsqueda aleatoria introducen perturbaciones estocásticas, por ejemplo el operador de mutación en los algoritmos genéticos, dicha perturbación permite explorar grandes regiones del espacio de búsqueda y potencialmente ayuda a escapar de los mínimos locales. La magnitud óptima de las perturbaciones en orden de alcanzar un balance entre la exploración

y explotación, va a depender de la estructura del problema de optimización que nos interesa resolver. En general, esta dependencia hace que la selección de los parámetros sea un factor muy importante en el diseño de algoritmos heurísticos de búsqueda.

Un algoritmo heurístico de exploración global puede construirse utilizando una analogía muy conocida entre los problemas de optimización y los sistemas físicos en equilibrio [2]. Considere que se tiene un problema de optimización con una función de costos $f(x_1, x_2, ..., x_i, ..., x_n)$. La densidad de probabilidad de un sistema físico en equilibrio bajo un potencial f esta dada por

$$p(x_1, x_2, ..., x_i, ..., x_n) = \left(\frac{1}{Z}\right)\exp(-f/kT)$$

donde T es la temperatura, k es la constante de Boltzmann y Z es la constante de normalización llamada función de partición. Se puede observar que cuando el sistema esta "caliente" y el valor de kT disminuye, la distribución de Boltzmann se concentra en los estados de menor energía, es decir, explora aquellas regiones del espacio cercanas al óptimo global. Por otro lado, si kT se aproxima a cero en donde el sistema está "congelado", únicamente los estados con mínima energía tienen una probabilidad de ocurrencia diferente de cero, lo que implicaría explotar demasiado el espacio de búsqueda quedando atrapado en óptimos locales. Es evidente que existe la necesidad de una cuidadosa selección de parámetros que rija el grado de perturbación adecuado al proceso de búsqueda aleatoria. Un conjunto de parámetros de temperatura para un cierto problema de optimización podría no ser adecuado para otro problema dado.

Los métodos de cadenas de Markov Monte Carlo (MCMC por sus siglas en inglés, Markov Chain Monte Carlo) han adquirido una importancia considerable como un muy general y potente enfoque para extraer muestras de una distribución de interés $p(\mathbf{x})$ mediante la realización de una caminata aleatoria sobre el vector $\mathbf{x} = (x_1, x_2, ..., x_i, ..., x_n)$. Las muestras extraídas representan las posibles soluciones al problema de optimización. Para

problemas que implican espacios de búsqueda con alta dimensionalidad, es decir, $n >> 1$, la distribución de interés $p(\mathbf{x})$ asociada a ese problema también tendrá una alta dimensionalidad. Para generar muestras en problemas de alta dimensión, el uso del muestreo de Gibbs es conveniente. El muestreo de Gibbs es un simple y ampliamente aplicable método MCMC, que puede ser visto como un caso especial del algoritmo de Metrópolis. El muestreo de Gibbs es un método para extraer muestras de una distribución de interés multivariada $p(\mathbf{x})$ en términos de las distribuciones condicionales $p(x_i|\mathbf{x}_{-i})$. La razón principal para utilizar el muestreo de Gibbs es que no se requiere ajustar algún parámetro de perturbación, es decir, un tamaño de paso en la caminata aleatoria. Además, el proceso de búsqueda generado por el muestreo Gibbs es capaz de saltar grandes regiones de baja probabilidad [3, 4]. El principal problema con elegir el tamaño de paso de la caminata aleatoria, como es el caso del muestreo de Metrópolis, es que si se eligen tamaños de pasos muy pequeños, la caminata aleatoria tomará mucho tiempo para llegar a cualquier lado de la distribución que se desea explorar. Lo anterior implica que el muestreo de Metrópolis tendrá una convergencia lenta para obtener una muestra aleatoria independiente. La convergencia a la distribución correcta $p(\mathbf{x})$ en el muestreo de Gibbs, es geométrica bajo ciertas condiciones generales [5, 6].

En este trabajo de tesis, se introduce una nueva forma de muestreo de Gibbs capaz de adaptar el tamaño de los pasos en la caminata aleatoria, explorando de manera eficiente el espacio de búsqueda en problemas de optimización con estructuras muy complejas. El algoritmo de muestreo propuesto se basa en el algoritmo de Metrópolis dentro de Gibbs, de manera que la caminata aleatoria combina el algoritmo unidimensional de Metrópolis con el multidimensional muestreador de Gibbs. Por otro lado, el algoritmo de muestreo propuesto en combinación con un procedimiento de búsqueda local, da como resultado un nuevo algoritmo heurístico de optimización global llamado, Estrategia Evolutiva con Caminatas Aleatorias de Metrópolis (RWM-ES por su siglas en inglés, *Random Walk Metropolis Evolution Strategy*).

1.1. Objetivo de la tesis

El objetivo principal de la tesis consiste en presentar un nuevo enfoque heurístico de búsqueda aleatoria, abordando la aplicabilidad de las técnicas de muestreo basados en métodos de Monte Carlo dentro del campo de la computación evolutiva para resolver diferentes tipos de problemas de optimización. El objetivo principal de la tesis puede ser divido en 4 partes:

1. Estudiar los métodos de cadenas de Markov Monte Carlo (MCMC) y los algoritmos de estrategias evolutivas.

2. Proponer y evaluar un nuevo método de muestreo para generar un conjunto de soluciones candidatas que exploren de forma eficiente el espacio de búsqueda de un problema de optimización.

3. Diseñar e implementar un nuevo enfoque heurístico de búsqueda aleatoria para optimización dentro del campo de la computación evolutiva.

4. Evaluar el desempeño del algoritmo propuesto en un rango amplio de problemas de optimización y comparar los resultados obtenidos con otros enfoques heurísticos dentro del mismo campo en el estado del arte.

1.2. Justificación del estudio

En este trabajo se busca brindar un nuevo enfoque heurístico de búsqueda aleatoria dentro del campo de la computación evolutiva para resolver diferentes tipos de problemas de optimización global. Los problemas de optimización global son de gran interés ya que surgen en numerosas áreas que modelan el mundo real, y se ha encontrado una amplia aplicación en el estudio de sistemas físicos, químicos, económicos y de ingeniería [1].

El teorema de No Free Lunch (NFL) establece que todos los algoritmos que buscan optimizar una función de costos, en promedio tienen un comportamiento igual ante un conjunto de todos los problemas de optimización posibles [7, 8]. Además, el teorema de NFL nos dice que es imposible tener un algoritmo general de optimización, y que la única manera de que un algoritmo sea mejor que otro es si el algoritmo está diseñado especialmente para la estructura específica del problema a optimizar [9].

Dada la dificultad de contar con un método de óptimización de proposito general, es de gran interés establecer un nuevo algoritmo heurístico de búsqueda global que brinde soluciones de buena calidad y que se comporte bien ante un amplio rango de problemas de optimización en los cuales el conocimiento sobre la estructura del problema es insuficiente.

1.3. Hipótesis

La hipótesis principal de este trabajo es que se tiene un método de muestreo eficiente que puede extraer muestras de una distribución de interés, la cual contiene toda la información de la estructura del problema que se desea resolver. El método de muestreo propuesto mantiene un balance entre la exploración y explotación del espacio de búsqueda mediante la adaptación de sus parámetros. El método de muestreo propuesto en combinación con un procedimiento de búsqueda local tiene como consecuencia un nuevo algoritmo heurístico de búsqueda global, que es capaz de encontrar soluciones de buena calidad y robustas a problemas de optimización con estructuras muy complejas.

1.4. Contribución científica

Las principales contribuciones del trabajo de tesis doctoral se centran en dos aspectos: el método de muestreo y un algoritmo heurístico de optimización.

Muestreador de Gibbs-Cauchy Se contribuye en este trabajo, con un método de mues-

treo que es capaz de adaptar el tamaño de los pasos de la caminata aleatoria, explorando de manera eficiente el espacio de búsqueda en problemas de optimización con estructuras muy complejas. El método de muestreo propuesto tiene dos características muy importantes:

1. El nivel de perturbación en la caminata aleatoria es controlado adaptativamente de acuerdo a la estructura del problema de optimización.
2. Los valores candidatos para cada variable son generados por una distribución de salto de Cauchy. La distribución de Cauchy permite dar saltos largos durante el proceso de búsqueda ya que tiene varianza infinita y, como resultado, permite escapar de óptimos locales y explorar otras regiones del espacio.

Algoritmo RWM-ES El segundo aspecto en que se centra la contribución de este trabajo consiste en el diseño e implementación de un nuevo método heurístico de búsqueda aleatoria para optimización llamado, Estrategia Evolutiva con Caminatas Aleatorias de Metrópolis (RWM-ES por su siglas en inglés, Random Walk Metropolis Evolution Strategy). Un resultado colateral de esta contribución es la implementación de un paquete computacional del método RWM-ES propuesto, escrito en el lenguaje de programación R.

1.5. Estructura de la tesis

El presente trabajo de tesis de encuentra estructurado de la siguiente manera. En el capítulo 2 se presenta el marco téorico y los conceptos necesarios de las diferentes áreas que convergen en el desarrollo de este trabajo como lo es la teoría de las cadenas de Markov, los métodos básicos de cadenas de Markov Monte Carlo y la teoría de los algoritmos evolutivos. El capítulo 3 se presenta el método de muestreo propuesto y el algoritmo heurístico de optimización global. En el capítulo 4 se presenta la implementación

computacional del método de optimización propuesto, asi como un ejemplo de uso del paquete computacional sobre un problema clásico de optimización. En el capítulo 5 se presenta la experimentación computacional que prueba el desempeño de la metodología propuesta sobre un rango amplio de problemas irrestrictos y restringidos de optimización con estructuras muy complejas. Específicamente, los bancos de problemas de optimización que se utilizan en este trabajo son los que se sometieron en las competencias 2005 y 2006 del congreso de computación evolutiva (CEC por sus siglas en ingles, *Congress on Evolutionary Computation*) de la IEEE. Además, se analizan los resultados obtenidos por el método propuesto con otros métodos heurísticos en el estado del arte. Por último, en el capítulo 6 se establecen las conclusiones del trabajo y se comenta brevemente el trabajo que queda por hacer como consecuencia de los resultados obtenidos.

Capítulo 2

Marco teórico

El propósito de este capítulo es presentar las definiciones y los conceptos básicos necesarios para abordar el presente trabajo. Estos conceptos serán útiles para la comprensión del cuerpo principal de la tesis doctoral. En la sección 2.1 se describe qué son las caminatas aleatorias. La sección 2.2 se define el problema de muestreo y las dificultades subyacentes de muestrear una función de distribución arbitraria. La sección 2.3 define las propiedades de las cadenas de Markov, particulamente se definen las propiedades de convergencia de las cadenas de Markov a una distribución de interés. Los principales métodos de muestreo basados en cadenas de Markov Monte Carlo son presentados en la sección 2.4. Después, la sección 2.5 presentan los conceptos básicos de los algoritmos evolutivos, así como un par de métodos de esta clase son descritos en la misma sección. Por último, en la sección 2.6 se define lo que es un problema de optimización y algunas de las propiedades estructurales de las funciones a optimizar.

2.1. Caminatas Aleatorias

Una caminata aleatoria es un proceso en el cual el "caminante" realiza sucesivos movimientos (o pasos) al azar, vagando fuera del lugar de donde empezó. La trayectoria

generada por dichos movimientos al azar se le conoce como caminata aleatoria. Digamos que X_m define una trayectoria en la posición x después de $0, 1, ..., m$ pasos. Una caminata aleatoria se modela matemáticamente mediante la siguiente expresión:

$$X_m = X_{m-1} + \varepsilon_m \tag{2.1}$$

donde ε es una perturbación estocástica que rigue la longitud y dirección del siguiente paso. La caminata aleatoria descrita por la ecuación (2.1) nos dice que el valor actual del m-ésimo paso de la caminata está dado por el valor pasado en el paso $m-1$ más un término de perturbación. Se puede observar que un proceso estocástico definido de esta manera cumple la propiedad de las cadenas de Markov, en la cual las transiciones entre los estados, sólo puede producirse entre estados vecinos.

El caso más simple para representar una caminata aleatoria es en el que el "caminante" visita los sitios de una red discreta unidimensional. Los sitios de la red están igualmente espaciados una distancia unitaria. En cada paso, partiendo del origen, el "caminante" con probabilidad q se mueve (o salta) al sitio de la derecha, o bien al de la izquierda con probabilidad $1-q$. En la Figura 2.1 se muestra una caminata aleatoria simple simétrica con $q = 1/2$. La Figura 2.2 muestra las trayectorias generadas por cinco caminatas aleatorias simples con m= 100 pasos para cada una.

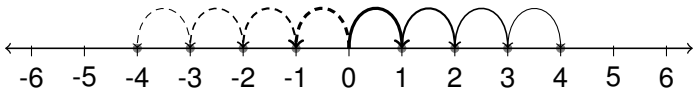


Figura 2.1 – En cada paso el "caminante" solo va al sitio más cercano con igual probabilidad.

La caminata aleatoria unidimensional descrita anteriormente, puede ser vista como un ejemplo tradicional de una caminata aleatoria que genera una persona borracha moviéndose al azar un paso hacia adelante o un paso hacia atrás. Otro ejemplo clásico es el de la "ruina del jugador". Supongamos que un jugador con un cierto monto de dinero, juega una serie

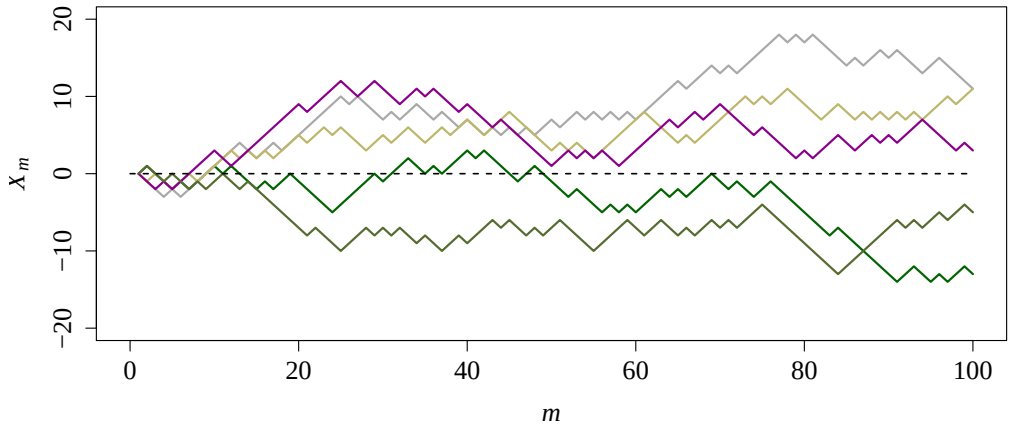


Figura 2.2 – Trayectorias generadas por cinco caminatas aleatorias simples con m=100 pasos cada una.

de partidas de azar en contra de un adversario infinitamente rico. En cada juego, el jugador puede ganar una unidad de dinero con probabilidad q y puede perder una unidad de dinero con probabilidad $1 - q$. Cuando el dinero del jugador se acaba (es cero), entonces éste se retira. La trayectoria X_m generada representa la cantidad de dinero que tiene el jugador después de jugar m partidas.

2.2. Muestreo

El término de muestreo se refiere al problema de generar un conjunto de posiciones aleatorias $\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, ..., \mathbf{x_m}$, llamadas muestras, a partir de una distribución de probabilidad dada $p(\mathbf{x})$ mediante la realización de una caminata aleatoria sobre el vector $\mathbf{x} = (x_1, x_2, ..., x_n)$. La distribución de probabilidad $p(\mathbf{x})$ se le conoce como distribución de interés.

Supongamos que queremos generar muestras a partir de una distribución de interés que se puede expresar de la forma $p(\mathbf{x}) = f(\mathbf{x})/Z$ donde

$$f(x_1, x_2) = 0.5e^{-90(x_1-0.5)^2-45(x_2+0.1)^4} + e^{-45(x_1+0.4)^2-60(x_2-0.5)^2}$$

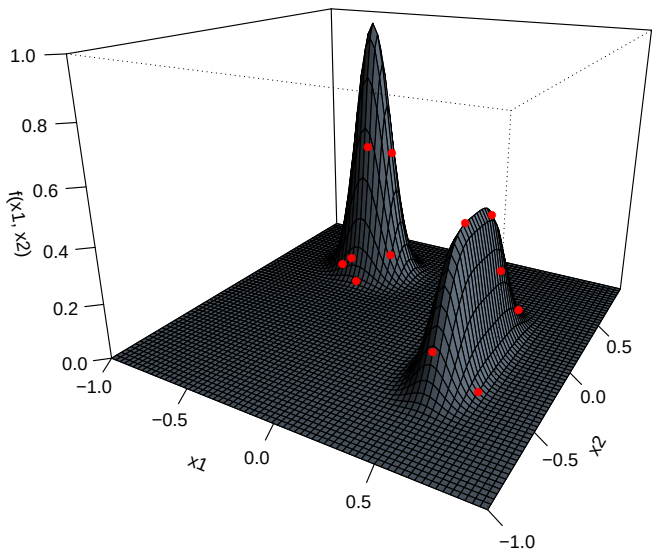


Figura 2.3 – Muestras extraídas (puntos rojos) a partir de la distribución de interés $p(\mathbf{x})$.

y Z es la constante de normalización. En la Figura 2.3 se puede observar las muestras con puntos de color rojo sobre el perfil de la función de interés $f(\mathbf{x})$. La densidad de estos puntos se aproximará a la distribución de interés $p(\mathbf{x})$; cuanto mayor sea el número de muestras, mejor será la aproximación.

Inherente al muestreo existen dos dificultades para extraer muestras a partir de la distribución $p(\mathbf{x})$. La primer dificultad es que típicamente no se conoce la constante de normalización

$$Z = \int f(\mathbf{x}) d^n \mathbf{x}$$

La segunda dificultad es que, aun conociendo a la constante de normalización Z, el problema de extraer muestras a partir de $p(\mathbf{x})$ sigue siendo muy complicado, especialmente en espacios de alta dimensión ($n >> 1$), ya que no hay una manera de muestrear a $p(\mathbf{x})$ sin tener que visitar cada punto en el espacio. Supongamos que queremos calcular Z.

El costo para evaluar la constante Z, teniendo un sistema de dimensión $n = 100$ y 50 puntos del espacio en cada dimensión, es de 50^{100} evaluaciones de $f(\mathbf{x})$, siendo esto computacionalmente costoso.

Una vez establecido que extraer muestras de una distribución de alta dimensión en la forma $p(\mathbf{x}) = f(\mathbf{x})/Z$ es difícil, incluso si $f(\mathbf{x})$ es fácil de evaluar en cualquier valor dado de $\mathbf{x}$, es necesario recurrir al estudio de los métodos de muestreo basados en cadenas de Markov Monte Carlo, ya que no requieren de un conocimiento de la constante de normalización Z y escalan bien con la dimensionalidad del espacio. Los métodos de muestreo basados en cadenas de Markov Monte Carlo son discutidos más adelante.

2.3. Cadenas de Markov

Antes de discutir a detalle los métodos de cadenas de Markov Monte Carlo, es necesario estudiar algunas propiedades de las cadenas de Markov. En particular, nos interesa saber bajo qué circunstancia una cadena de Markov convergerá a la distribución de interés.

Una cadena de Markov es un proceso estocástico que se define como una serie de variables aleatorias $X_0, X_1, X_2, X_3, ..., X_m$ que corresponden a los estados de un cierto sistema, de tal manera que cualquier estado futuro sólo depende del estado actual. Con lo anterior, la distribución de probabilidad de transición (de un paso) se define de la siguiente manera,

$$p(X_{m+1}|X_0, X_1, X_2, X_3, ..., X_m) = p(X_{m+1}|X_m)$$

Podemos observar que las cadenas de Markov tienen la capacidad de "recordar" el último evento, condicionando las probabilidades de eventos futuros. Una cadena de Markov se dice que es homogénea (o estacionaria) si las probabilidades de transición de ir de un estado a otro, en un paso, son las mismas para toda m, es decir, independientes del tiempo

en el que se encuentra la cadena, esto es:

$$p(X_1|X_0) = p(X_2|X_1) = p(X_3|X_2) = \cdots = p(X_{m+1}|X_m)$$

en donde $p(\cdot|\cdot) \geq 0$. De aquí en adelante las probabilidades de transición $p(X_{m+1} = y|X_m = x)$ se escribirán simplemente como $q(x, y)$, y satisface que $\sum_y q(x, y) = 1$ para todo x.

La distribución de probabilidad marginal para una variable en particular se puede expresar en términos de la distribución marginal de la varible anterior de la cadena en la forma

$$p(y) = \sum_x q(x, y)p(x) \tag{2.2}$$

La distribución de probabilidad $p(y)$ se dice que es invariante (o estacionaria) con respecto a la cadena de Markov si cumple con la ecuación (2.2) para todo y. Una distribución se dice que es invariante con respecto a una cadena de Markov si en cada paso en la cadena deja esa distribución invariante.

Una condición para garantizar que la distribución requerida $p(\cdot)$ sea invariante, es elegir las probabilidades de transición que satisfacen la propiedad de balance detallado, esto es, la cadena de Markov tiene la misma probabilidad de iniciar en x y saltar a y como iniciar en y y saltar a x, y está definida por

$$q(x, y)p(x) = q(y, x)p(y)$$

para la distribución particular $p(y)$. Se observa fácilmente que una probabilidad de transición que satisface el balance detallado con respecto a una distribución particular dejará esa distribución invariante, porque

$$\sum_x p(x)q(x, y) = \sum_x p(y)q(y, x) = p(y) \sum_x q(y, x) = p(y)$$

Una cadena de Markov que satisface la propiedad de balance detallado se dice que es reversible. Una cadena de Markov reversible tiene las mismas leyes de ir hacia adelante o hacia atrás en cada paso.

Nuestro objetivo es utilizar las cadenas de Markov para extraer muestras de una distribución arbitraria dada. Esto se puede lograr si creamos una cadena de Markov de tal manera que la distribución de interés es invariante. Sin embargo, también debemos exigir que para $m \to \infty$, la distribución $p(X_m = x)$ converge a la distribución invariante requerida $p(y)$, independientemente de la elección de la distribución inicial de la cadena $p(X_0 = x_0)$. Esta propiedad se llama ergodicidad, y la distribución invariante entonces se llama distribución en equilibrio. Una cadena de Markov ergódica tiene solo una distribución en equilibrio.

2.4. Métodos de Cadenas de Markov Monte Carlo

Los métodos de cadenas de Markov Monte Carlo (MCMC por sus siglas en inglés, Markov Chain Monte Carlo), son métodos genéricos que generan posiciones aleatorias, llamadas muestras, que se aproximan a las de una distribución arbitraria. La idea principal es generar una cadena de Markov cuya distribución converge a la distribución de interés. La cadena de Markov se construye de tal modo que la cadena pasa más tiempo en las regiones más importantes.

Los métodos MCMC tiene sus origenes en la física [10] y se utilizó por primera vez para investigar las propiedades de equilibrio de sistemas de gran escala de partículas [2]. Para más detalles sobre los métodos de cadenas de Markov Monte Carlo, teoremas y demostraciones, así como una lista completa de referencias, se remite al lector a [11, 12, 13].

En esta sección, se describen los dos métodos MCMC más simples y destacados: las caminatas aleatorias de Metrópolis y el muestreador de Gibbs. Además, se presenta un método MCMC híbrido llamado caminatas aleatorias de Metrópolis dentro de Gibbs.

Este último método es la base para el desarrollo del método de muestreo propuesto que se encuentra descrito en la sección 3.1.

2.4.1. Caminatas aleatorias de Metrópolis

El algoritmo de caminatas aleatorias de Metrópolis (RWM por sus siglas en inglés, *Random Walk Metropolis*), fue propuesto por Metrópolis en 1953 [2], y tomó su forma general en el artículo de Hastings de 1970 [14]. Las caminatas aleatorias de Metrópolis extraen muestras a partir de alguna distribución de interés. La distribución de interés se puede expresar de la forma $p(\cdot) = f(\cdot)/Z$, en donde $f(\cdot)$ puede ser evaluada fácilmente para cualquier valor dado, aunque el valor de la constante de normalización Z sea desconocida. El algoritmo RWM procede de la siguiente manera. Dado un valor actual de la cadena de markov m-dimensional, $X_m = x$, un valor candidato y para la siguiente posición de la cadena X_{m+1}, se obtiene de la distribución propuesta $\beta(y, x)$. El valor candidato y se acepta de acuerdo a la probabilidad de aceptación,

$$\alpha(x, y) = \text{mín}\left\{1, \frac{p(y)\beta(y, x)}{p(x)\beta(x, y)}\right\} \tag{2.3}$$

si no rechazar con probabilidad $1 - \alpha(x, y)$. Si el valor candidato se acepta, se convierte en el siguiente valor actual $X_{m+1} = y$; En caso contrario, el valor candidato se deja sin cambio, $X_{m+1} = x$.

En cada ciclo del algoritmo RWM se genera una muestra candidata $\mathbf{x}$ a partir de la distribución propuesta $\beta(y, x)$. La distribución propuesta se elige lo suficientemente simple para poder extraer muestras directamente de ellas. Además, se requiere también que la distribución propuesta sea una función de probabilidad simétrica, es decir $\beta(y, x) = \beta(x, y)$, por ejemplo una distribución Gaussiana centrada en el estado actual de la cadena x. Con lo

anterior, la probabilidad de aceptación de la ecuación (2.3) queda redefinida como,

$$\alpha(x, y) = \text{mín}\left\{1, \frac{p(y)}{p(x)}\right\}. \tag{2.4}$$

La probabilidad de aceptación $\alpha(x, y)$ se elige de modo que la cadena de Markov sea reversible con respecto a la distribución de interés $p(\cdot)$. Una cadena de Markov que es reversible implica que satisface la propiedad de balance detallado. Para verificar esto, primero se va a definir la probabilidad de transición como $q(x, y) = \beta(x, y)\alpha(x, y)$, entonces

$$\begin{aligned} p(x)q(x, y) &= p(x)\beta(x, y)\alpha(x, y) \\ &= \beta(x, y)\,\text{mín}(p(x), p(y)) \\ &= \beta(y, x)\,\text{mín}(p(y), p(x)) \\ &= p(y)\beta(y, x)\alpha(y, x) \\ &= p(y)q(y, x) \end{aligned}$$

Con lo anterior, se puede observar que la distribución de interés $p(\cdot)$ es una distribución invariante (estacionaria) para la cadena [15]. El conjunto de muestras $\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, ..., \mathbf{x_m}$, generadas por el algoritmo RWM se distribuyen aproximadamente de acuerdo a la distribución de interés $p(\mathbf{x})$, para m grandes.

2.4.2. Muestreador de Gibbs

El muestreador de Gibbs puede ser visto como un caso especial del algoritmo de caminatas aleatorias de Metrópolis. Este método fue introducido por los hermanos Geman [16] con el fin de resolver problemas de reconstrucción en el tratamiento de imágenes. El muestreador de Gibbs es particulamente útil para generar vectores aleatorios n-dimensionales, $\mathbf{x} = (x_1, x_2, ..., x_i, ..., x_n)$. La característica distintiva del muestreo de Gibbs es que la cadena de Markov se construye a partir de una secuencia de distribuciones condicionales de la

distribución conjunta $p(\mathbf{x})$. Esto es, asumiendo que es muy complicado generar muestras a partir de la distribución $p(\mathbf{x})$, pero las distribuciones condicionales $p(x_i|p(\mathbf{x}_{-i})$ son tratables para trabajar.

Consideremos la distribución $p(\mathbf{x}) = p(x_1, ..., x_i, ..., x_n)$ a partir de la cual queremos extraer muestras y suponemos que se escoge un estado inicial de la cadena de Markov. Cada paso del algoritmo $1, 2, ..., m$, consiste en reemplazar el valor de una de las variables por un valor extraído de la distribución de esa variable condicionada a los valores de las variables restantes. Para un sistema de n variables, un simple ciclo de Gibbs involucra muestrear una variable a la vez:

$$\begin{aligned}
x_1^{m+1} &\sim p(x_1|x_2^m, x_3^m, ..., x_i^m, ..., x_n^m) \\
x_2^{m+1} &\sim p(x_2|x_1^m, x_3^m, ..., x_i^m, ..., x_n^m) \\
&\vdots \\
x_i^{m+1} &\sim p(x_i|x_1^m, x_2^m, ..., x_{i-1}^m, x_{i+1}^m, ..., x_n^m) \\
&\vdots \\
x_n^{m+1} &\sim p(x_n|x_1^m, x_2^m, ..., x_i^m, ..., x_{n-1}^m)
\end{aligned}$$

Lo anterior nos lleva a un nuevo estado y se completa el ciclo. Después de m ciclos, el muestreo de Gibbs generá una secuencia de muestras $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_m}$ que se aproximan a las de una distribución $p(\mathbf{x})$. Este procedimiento se puede repetir, ya sea de forma ciclica a través de las variables en un orden particular, o mediante la elección de la variable que desea actualiza en cada paso al azar.

Por otro lado, se puede observar que cuando se muestrea a partir de $p(x_i|p(\mathbf{x}_{-i})$, la distribución marginal $p(\mathbf{x}_{-i})$ es claramente invariante debido a que el valor de $\mathbf{x}_{-i}$ no es modificado. Con lo anterior, se puede notar que la distribución $p(\mathbf{x})$ es invariante en cada uno de los pasos individuales del muestreador de Gibbs y por lo tanto, de toda la cadena de Markov. Otro punto que se tiene que satisfacer para generar muestras a partir de la distribución correcta es que sea ergódica. Una condición suficiente para la ergodicidad, es que las distribuciones condicionales sean distintas de cero. Esto garantiza que existe una

probabilidad no nula para que la cadena pase de cualquier estado a cualquier otro estado en varios pasos, involucrando una actualización de cada una de las variables [15].

2.4.3. Caminatas aleatorias de Metrópolis dentro de Gibbs

Un método MCMC híbrido se puede construir mediante la combinación de una caminata aleatoria unidimensional de Metrópolis con el multidimensional muestreador de Gibbs. El método MCMC híbrido resultante es llamado caminatas aleatorias de Metrópolis dentro de Gibbs y procede de la siguiente manera. Consideremos una distribución de interés multivariada $p(\mathbf{x}) = p(x_1, ..., x_i, ..., x_n)$ a partir de la cual queremos extraer muestras mediante la realización de una caminata aleatoria sobre el vector $\mathbf{x} = (x_1, ..., x_i, ..., x_n)$, actualizando una de las componentes x_i del vector a la vez, en un ciclo de longitud n. Iniciando con un vector inicial arbitrario $\mathbf{x}^o = (x_1^o, ..., x_i^o, ..., x_n^o)$, suponer que en el m-ésimo paso de la caminata aleatoria se ha muestreado un vector $\mathbf{x}^m = (x_1^m, ..., x_i^m, ..., x_n^m)$. Para generar un vector $\mathbf{x}^{m+1} = (x_1^{m+1}, ..., x_i^{m+1}, ..., x_n^{m+1})$ se actualizará una componente x_i^m generando un valor candidato y a partir de distribución propuesta $q(y|\mathbf{x}_{-i}^m)$, donde $\mathbf{x}_{-i}^m = (x_1^{m+1}, ..., x_{i-1}^{m+1}, x_{i+1}^m, ..., x_n^m)$ y aceptar este valor, $x_i^{m+1} = y$, de acuerdo con la probabilidad de aceptación de la ecuación (2.4) definida como,

$$\alpha(x_i^m, y) = \text{mín}\left\{1, \frac{p(y|\mathbf{x}_{-i}^m)}{p(x_i^m|\mathbf{x}_{-i}^m)}\right\} \tag{2.5}$$

si no rechazar con probabilidad $1 - \alpha(x_i^m, y)$, dejando el valor de la variable sin cambio, esto es $x_i^{m+1} = x_i^m$. Después de realizar m ciclos, el muestreo generá una secuencia de muestras $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_m}$, de manera que las muestras de cada ciclo se distribuyen como la distribución de interés $p(\mathbf{x})$.

2.5. Algoritmos Evolutivos

Los algoritmos evolutivos (EA por sus siglas en inglés, *Evolutionary Algorithm*), se refieren a cualquier marco heurístico que está inspirado por el proceso de la evolución y adaptación natural. Los algoritmos evolutivos son métodos de optimización y búsqueda de soluciones basados en la evolución de una población, comunicando e intercambiando información entre los individuos de la misma. La población evoluciona de generación en generación en dos etapas. En la primer etapa, algún mecanismo de selección es aplicado con el fin de crear una nueva población. En la segunda etapa, algún mecanismo de alteración se aplica a la población recientemente creada. Los mecanismos de alteración son aquellos que permiten el intercambio de información entre los individuos de la población, asi como también algunas perturbaciones aleatorias dentro de la misma. Las perturbaciones también son conocidas como mutaciones.

El objetivo es crear una población de individuos con cualidades superiores con respecto a alguna medida de aptitud en la población. El ejemplo más simple es donde $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_m)$ es un conjunto de soluciones n-dimensionales y el objetivo es minimizar o maximizar alguna función de costos $f(\mathbf{x})$, o bien, también llamada función de aptitud. En este caso, la evaluación de $\mathbf{X}$ corresponde al calculo de $f(\mathbf{x})$ para todo $\mathbf{x} \in \mathbf{X}$. Por lo general esta información se utiliza en la etapa de selección, por ejemplo, permitiendo que solamente el 10 % de la población participe en el intercambio de información para la creación de la nueva generación. El marco general de un algoritmo evolutivo se resume a continuación.

Algoritmo 1 Esquema básico de un algoritmo evolutivo.

1: Definir $t = 1$. Inicializar la población $\mathbf{X}^t$. Evaluar $\mathbf{X}^t$.
2: Seleccionar una nueva población $\mathbf{X}^{t+1}$ a partir de $\mathbf{X}^t$.
3: Alterar la población $\mathbf{X}^{t+1}$.
4: Evaluar $\mathbf{X}^{t+1}$.
5: Si un criterio de paro es alcanzado, parar; en otro caso, definir $t = t + 1$ y repetir desde el paso 2.

Los algoritmos evolutivos lleva a cabo el proceso de optimización mediante la exploración del espacio de búsqueda de forma aleatoria. La naturaleza estacástica de los algoritmos evolutivos implica que los resultados varíen de una ejecución a otra, por lo que no se puede garantizar que siempre encuentren la mejor solución sobre un problema dado. Sin embargo, el desempeño promedio de estos métodos heurísticos suelen ser buenos en problemas de optimización con estructuras de la función de costos muy complejas y con tiempos de cómputo razonable.

En las siguientes secciones, se describen brevemente dos paradigmas más conocidos de los algoritmos evolutivos que se encuentran en la literatura: Los algoritmos genéticos y las estrategias evolutivas. El marco general de las estrategias evolutivas nos servirá como base para el desarrollo del método de optimización propuesto descrito en la sección 3.2.

2.5.1. Algoritmos genéticos

Los algoritmos genéticos (GA por sus siglas en inglés, *Genetic Algorithms*), son una clase de algoritmos evolutivos muy populares y fueron desarrollados por Holland en 1970 para entender los procesos de adaptación de los sistemas naturales. Después, en 1980 se aplicaron a la optimización y a el aprendizaje automático [17].

Las algoritmos genéticos son métodos heurísticos que están basados en una población de individuos que evolucionan de generación en generación. Los individuos de la población pueden ser representados como posibles soluciones a un problema de optimización. La evolución de la población puede ser dividida en dos etapas. Para la primer etapa, los algoritmos genéticos utilizan una selección probabilística que es originalmente la selección proporcional, es decir, los individuos (padres) de la población que tenga una mejor función de aptitud (costos), tienen mayor probabilidad de reproducirse, mientras que los peores individuos tienen baja probabilidad de generar descendientes (hijos). Durante la segunda etapa, llamada la etapa reproductiva, generalmente se aplica un mecanismo de recombi-

nación de individuos padres que desempeña un papel importante en el proceso evolutivo para producir descendientes, además de un mecanismo de mutación que modifica de forma aleatoria el contenido genético de los individuos hijos de la población para promover la diversidad. Básicamente el mecanismo de mutación asegura que ninguna región del espacio de búsqueda tenga probabilidad cero de ser explorada. El reemplazo (selección del superviviente) es generacional, es decir, los padres se sustituyen sistemáticamente por los hijos. De esta forma se genera una nueva población que remplaza a la población anterior y comienza otra generación.

El intercambio de información entre los padres dentro del mecanismo de recombinación y los pequeños cambios (mutación) en la descendencia promueven la búsqueda hacia mejores individuos. Combinando estos dos factores, la población será cada vez más y más apta hasta que se haya encontrado una solución óptima o cercana a ella. El marco general de los algoritmos genéticos se resume a continuación.

Algoritmo 2 Esquema básico de un algoritmo genético.

1: Definir $t = 1$. Inicializar una población $\mathbf{X}^t$ de forma aleatoria. Evaluar $\mathbf{X}^t$.
2: Seleccionar a los padres de $\mathbf{X}^t$.
3: Recombinar a los padres seleccionados.
4: Mutar a los individuos obtenidos de la recombinación.
5: Reemplazar a los padres por su descendencia para producir una nueva población $\mathbf{X}^{t+1}$.
6: Evaluar $\mathbf{X}^{t+1}$.
7: Si un criterio de paro es alcanzado, parar; en otro caso, definir $t = t + 1$ y repetir desde el paso 2.

2.5.2. Estrategias Evolutivas

Las estrategias evolutivas (ES por sus siglas en inglés, *Evolution Strategies*), son una subclase de los algoritmos evolutivos, tales como los algoritmos genéticos. Fueron desarrolladas originalmente por Rechenberg y Schwefel en 1964. La motivación original de las estrategias evolutivas fue resolver problemas muy complejos de hidrodinámica experimental relacionados con optimización de formas [18]. Dichos problemas eran intratables

con métodos clásicos de optimización de esa época.

Existen muchas variantes de las estrategias evolutivas de acuerdo al esquema de selección que utilizan. La propuesta original (1+1)-ES de Rechenberg, usaba un solo padre del cual se generaba un solo hijo, es decir, la población estaba compuesta por dos individuos. El hijo se mantenía si era mejor que el padre, según una función de aptitud (costos), o de lo contrario se eliminaba. En 1973, Rechenberg introduce el concepto de población al proponer la estrategia (μ+1)-ES, en la cual a partir de los μ padres se genera un solo hijo, el cual remplaza al peor padre de la población. En la estrategia (1+λ)-ES, el mejor individuo de los λ hijos se convierte en el padre de la siguiente población, mientras que el padre actual siempre se elimina. Por otro lado, Schwefel en 1975 introdujo las estrategias (μ+λ)-ES y (μ,λ)-ES [19]. La estrategia (μ+λ)-ES combina a los μ padres y λ hijos y solo los mejores μ individuos forma una nueva población. En la estrategia (μ,λ)-ES, los μ padres se utilizan para generar λ hijos y se seleccionan solo los mejores μ individuos dentro de los λ hijos para formar una nueva población para la siguiente generación. En el enfoque anterior se requiere de que $\lambda \geq \mu$.

El mecanismo de mutación dentro de las estrategias evolutivas juegan un papel importate en la exploración del espacio de búsqueda. Se define a un padre como un vector de variables $\mathbf{x} = (x_1, ..., x_i, ..., x_n)$, y al hijo como $\mathbf{x}' = (x'_1, ..., x'_i, ..., x'_n)$. Entonces, los hijos se generan a partir de los padres de la siguiente manera,

$$x'_i = x_i + \sigma_i \cdot \mathcal{N}_i(0, 1) = \mathcal{N}(x_i, \sigma_i)$$

donde $\mathcal{N}(x_i, \sigma_i)$ es una perturbación aleatoria que sigue una distribución Gaussiana centrada en x_i y una desviación estandar σ_i como parámetro de escala (tamaño de paso de la mutación). El índice i indica que la perturbación es generada de nuevo para cada variable. Los parámetros de escala σ son muy importantes en la mutación Gaussiana ya que afecta de forma importante el desempeño de la estrategia evolutiva. El valor óptimo de este parámetro

depende del problema y su dimensionalidad. Existe una regla formulada por Rechenberg [20] para ajustar σ durante el proceso evolutivo para garantizar la convergencia de la estrategia (1+1)-ES. La regla es conocida como *la regla del éxito 1/5*. La regla dice que, la razón entre las mutaciones exitosas y el total de mutaciones debe de ser 1/5. Si es mayor, entonces debe de aumentarse σ. Si es menor, entonces debe reducirse σ. Con lo anterior, la regla del éxito 1/5 ajusta σ después de cada G iteraciones de acuerdo a los siguientes pasos:

1. Contar el número Gs mutaciones exitosas durante ese periodo.

2. Determinar la tasa de éxito Ps por

$$Ps = Gs/G$$

3. Cambiar σ de acuerdo a:

$$\sigma = \begin{cases} \sigma/c & \text{si} \quad Ps > 1/5 \\ \sigma \cdot c & \text{si} \quad Ps < 1/5 \\ \sigma & \text{si} \quad Ps = 1/5 \end{cases}$$

donde c es un valor entre 0.8 y 1. Por otro lado, Schwefel en 1977 sugiere la regla auto-adaptiva de los parámetros de escala (estratégicos) de la siguiente forma,

$$\sigma'_i = \sigma_i \cdot \exp\left(\tau_0 \cdot \mathcal{N}(0,1) + \tau \cdot \mathcal{N}_i(0,1)\right)$$

en donde $\mathcal{N}(0,1)$ es un valor aleatorio que sigue una distribución normal estandar y los parámetros de aprendizaje son

$$\tau_0 = \frac{1}{\sqrt{2n}} \quad \text{y} \quad \tau = \frac{1}{\sqrt{2\sqrt{n}}}$$

Las estrategias evolutivas están basadas en una población de individuos, y utilizan principalmente los mecanismos de mutación y selección en la búsqueda de la mejor solución. Los mecanismos de recombinación son raramente usados. El marco general de las estrategia evolutiva se resume a continuación.

Algoritmo 3 Esquema básico de una estrategia evolutiva.

1: Inicializar una población de μ individuos padres.
2: Evaluar los μ individuos padres.
3: Generar λ hijos a partir de los μ padres.
4: Evaluar los λ individuos hijos.
5: Seleccionar a los μ individuos que formarán la nueva población.
6: Si un criterio de paro es alcanzado, parar; en otro caso, repetir desde el paso 3.

Para una completa introducción a las estrategias evolutivas consultar [21]. Un enfoque similar al de la estrategias evolutiva descrita aquí, es utilizado en el desarrollo del método heurístico de optimización propuesto en este trabajo.

2.5.3. Conceptos comunes para los algoritmos evolutivos

Los principales componentes de búsqueda para el diseño de un algoritmo heurístico evolutivo son los siguientes:

1. **Inicialización:** La inicialización es un mecanismo para generar de forma aleatoria una población inicial, es decir, un conjunto de soluciones que representan las posibles soluciones del problema de interés. Es importante que la inicialización se realize de forma aleatoria, o bien, que se garantice que dentro de la población inicial, se tiene la diversidad estructural de estas soluciones para tener una representación de la mayor parte de la población posible. La diversidad de las soluciones podría evitar la convergencia prematura de algún algoritmo evolutivo.

2. **Tamaño de la población:** Entre más grande es el tamaño de la población en un algoritmo evolutivo, mejor es la convergencia hacia las “buenas” soluciones. Sin

embargo, la complejidad de tiempo en los algoritmos evolutivos crece linealmente con el tamaño de la población. Por lo tanto, se debe de encontrar un compromiso entre la calidad de las soluciones obtenidas y el tiempo de búsqueda del algoritmo.

3. **Función de costos:** La función de costos $f(x_1, ..., x_n)$ se relaciona con el objetivo a alcanzar. Es la guía hacia la búsqueda de "buenas" soluciones del espacio de búsqueda. Se asocia a cada solución del espacio de búsqueda que describe la calidad o la aptitud de la solución. La función de costos también se le conoce en la literatura como función objetivo, función de utilidad y función de evaluación.

4. **Estrategia de selección:** La estrategia de selección se refiere a la siguiente pregunta: ¿Qué soluciones de la población (padres) se eligen para la siguiente generación con un sesgo hacia una mejor aptitud?. El principio fundamental de los métodos de selección es, «individuos con una mejor aptitud, mayor es su probabilidad de ser padres». Dicha presión de selección conducirá a la población hacia mejores soluciones. Sin embargo, los peores individuos no deben ser descartados, ya que cuentan con alguna posibilidad de ser elegidos.

5. **Estrategia de reproducción:** La estrategia de reproducción consiste en el diseño de un mecanismo de mutación adecuado y un mecanismo de recombinación para generar nuevos individuos (descendencia).

   Algunos puntos importantes que deben tenerse en consideración en el diseño o el uso de un mecanismo de mutación son los siguientes:

   - **Ergodicidad:** El mecanismo de mutación debe permitir que cualquier solución del espacio de búsqueda sea alcanzado.
   - **Validez:** El mecanismo de mutación debe producir soluciones válidas. Esto no siempre es posible para los problemas de optimización con restricciones.

- **Localidad:** La mutación debe producir cambios mínimos. El tamaño de la mutación es importante y debe ser controlable.

Algunos puntos importantes que deben tenerse en consideración en el diseño o el uso de un mecanismo de recombinación son los siguientes:

- **Heredabilidad:** El mecanismo de recombinación debe heredar un material genético de ambos padres. Un mecanismo de recombinación puro (fuerte heredabilidad) es que dos individuos idénticos generen descendientes idénticos.
- **Validez:** El mecanismo de recombinación debe producir soluciones válidas. Esto no siempre es posible para los problemas de optimización con restricciones.

6. **Estrategia de reemplazo:** Los nuevos individuos descendientes (hijos) compiten con los individuos de la generación pasada para tomar su lugar en la próxima generación (supervivencia del más apto).

7. **Criterio de paro:** Se pueden utilizar muchos criterios de parada basados en la evolución de una población. Estos criterios se pueden dividir en dos tipos:

   - **Procedimiento estático:** En un procedimiento estático, puede conocerce a priori al final de la búsqueda. Por ejemplo, se puede utilizar un número fijo de iteraciones (generaciones), un límite de tiempo de cómputo, o máximo número de evaluaciones de la función de costos.
   - **Procedimiento adaptativo:** En un procedimiento adaptativo, no puede conocerce a priori al final de la búsqueda. Por ejemplo, puede utilizarse un número fijo de iteraciones (generaciones) sin mejora, o bien, cuando se alcanza un óptimo o una solución satisfactoria (por ejemplo, un error que mide la distancia del mejor valor con respecto al valor óptimo).

2.6. Optimización

El término de optimización se refiere a elegir la mejor decisión posible dentro de un conjunto de alternativas posibles. Los problemas de optimización están relacionados con la búsqueda de soluciones mínimas o máximas de una función de costos $f(x_1, x_2, ..., x_i, ..., x_n)$ dentro de un conjunto permitido Ω:

$$\min_{\mathbf{x}\in\Omega} f(\mathbf{x}) \quad \text{o} \quad \max_{\mathbf{x}\in\Omega} f(\mathbf{x})$$

Un problema de maximización se puede transformar fácilmente en un problema de minimización a través de la equivalencia, $\max_{\mathbf{x}} f(\mathbf{x}) \equiv -\min_{\mathbf{x}} -f(\mathbf{x})$. El domino de Ω es conocido como el espacio de búsqueda y los elementos $\mathbf{x} \in \Omega$ se les conoce como soluciones candidatas. Si Ω es un conjuto no numerable tales como $\Re^n$ (valores reales) y f toma valores en un conjunto no numerable, entoces se dice que se tiene un problema de optimización continua.

2.6.1. Optimización global

Los problemas de optimización global consisten en encontrar el mejor mínimo o máximo dentro de una multitud de mínimos o máximos locales de acuerdo a una función de costos. En este trabajo nos centraremos en los problemas de minimización. Un mínimo local de $f(x_1, x_2, ..., x_i, ..., x_n)$ es una solución $\mathbf{x}^* \in \Omega$ que satisface $f(\mathbf{x}^*) \leq f(\mathbf{x})$ para todo $\mathbf{x}$ en alguna vecindad de $\mathbf{x}^*$. Si se satisface que $f(\mathbf{x}^*) \leq f(\mathbf{x})$ para todo $\mathbf{x} \in \Omega$, entonces el valor de $\mathbf{x}^*$ se llama mínimo global, o también conocido como óptimo global. En otras palabras, un óptimo global puede verse como el mejor valor posible dentro de todo el espacio de búsqueda, mientras que un óptimo local es un valor que es mejor que todos los valores que están cercanos a él. Nótese que una solución óptima global también es un óptimo local, pero no viceversa. La Figura 2.4 muestra una función de costos unidimensional con

múltiples óptimos.

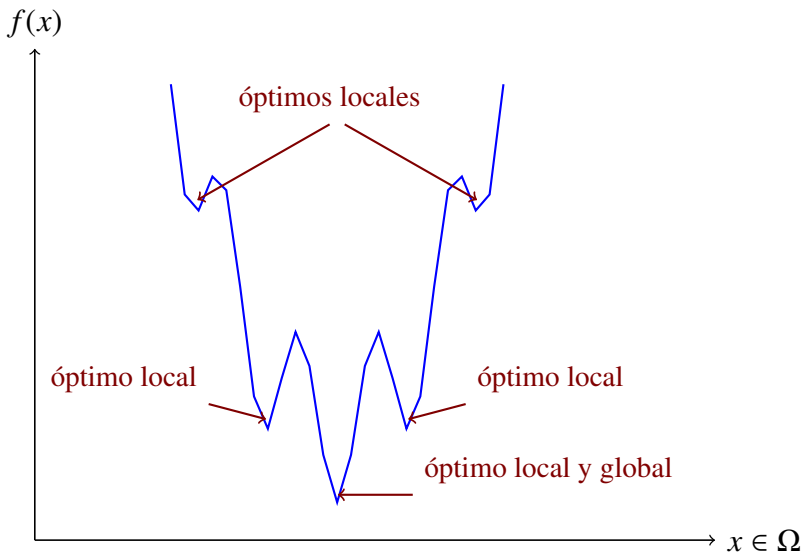


Figura 2.4 – Una función de costos con un óptimo global y varios óptimos locales.

Por otro lado, el espacio de búsqueda Ω también puede ser definido por medio de restricciones. La forma general de los problemas de optimización con restricciones se puede ilustrar de la siguiente manera:

$$\begin{aligned} \text{mín}_{\mathbf{x}\in\Omega} \quad & f(\mathbf{x}) \\ \text{sujeto a:} \quad & h_j(\mathbf{x}) = 0, \qquad j = 1, ..., J \\ & g_r(\mathbf{x}) \leq 0, \qquad r = 1, ..., R \\ & L_i \leq x_i \leq U_i, \quad i = 1, ..., n \end{aligned}$$

donde $h_j(\mathbf{x})$ son las restricciones de igualdad y $g_r(\mathbf{x})$ son las restricciones de desigualdad. Ambos tipos de restricciones pueden ser lineales como no lineales. L_i y U_i son los límites inferior y superior de las variables x_i, respectivamente, que forman el espacio de búsqueda. La región donde la función de costos $f(\mathbf{x})$ es definida y donde todas las restricciones impuestas se satisfacen, se llama región factible. Por el contrario, la región donde no se satisfacen las restricciones se llama región infactible. Una solución $\mathbf{x}^* \in \Omega$ de $f(\mathbf{x})$ sujeto a las restricciones $h_j(\mathbf{x})$ y $g_r(\mathbf{x})$ se dice que es un óptimo global, si la solución $\mathbf{x}^*$ satisface las

restricciones y satisface que $f(\mathbf{x}^*) \leq f(\mathbf{x})$ para todo $\mathbf{x} \in \Omega$.

Debido a la posible forma compleja de las restricciones, la relación entre la región factible y el espacio de búsqueda podría complicarse. La Figura 2.5 es un ejemplo.

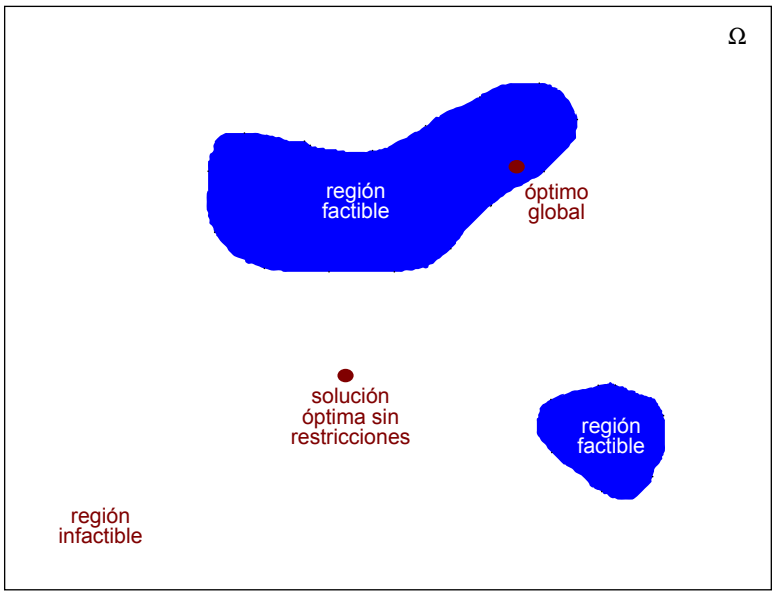


Figura 2.5 – El espacio de búsqueda y su región factible.

2.6.2. Optimización unimodal y multimodal

Los algoritmos de optimización global son aquellos algoritmos de optimización que emplean medidas que impiden la convergencia a un óptimo local, es decir, que evitan quedar atrapados en un óptimo local, y que aumentan la probabilidad de encontrar un óptimo global mediante sus mecanismos de exploración. Dichos mecanismos de exploración nos van a permitir un desplazamiento eficiente a través del espacio de búsqueda. El desempeño de los algoritmos de optimización global depende de la facilidad o dificultad para desplazarse por el espacio de búsqueda. La dificultad asociada al espacio de búsqueda está relacionado con la rugosidad o la no linealidad de la función de costos, es decir, que se tengan múltiples óptimos locales (modos) en los que se corre el riesgo de quedar atrapado, impidiendo encontrar el óptimo global. Entonces, encontrar buenas soluciones va depender de la

función de costos y la geometría del espacio de búsqueda asociada. Esto se puede ilustrar con la Figura 2.6 donde se muestra una función de costos altamente no lineal con un espacio de búsqueda muy rugoso, mientras que la Figura 2.7 ilustra una función de costos suave con un solo óptimo en el espacio de búsqueda. Las Figuras 2.8 y 2.9 presentan funciones de costos suaves con dos y tres óptimos locales (modos) en el espacio de búsqueda, respectivamente.

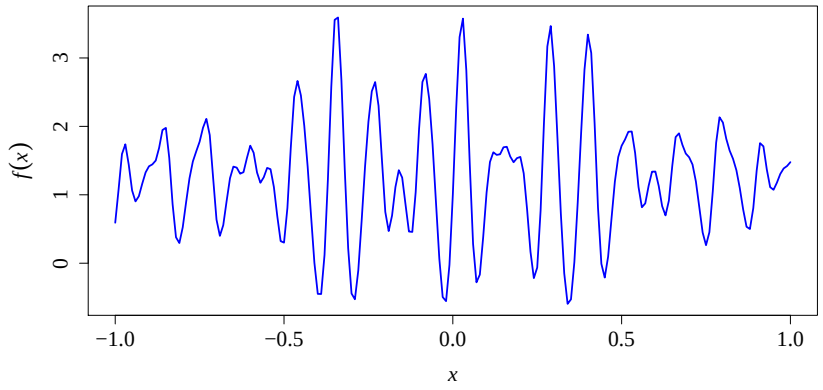


Figura 2.6 – Representación del espacio de búsqueda para $f(x) = \exp(\text{sen}(50x)) + \text{sen}(70\,\text{sen}(x))$.

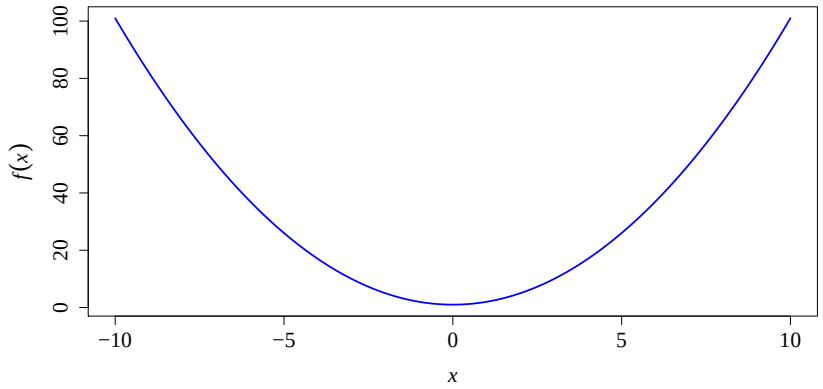


Figura 2.7 – Representación del espacio de búsqueda para $f(x) = x^2 - 1$.

El diseño adecuado de un mecanismo de exploración, por ejemplo la mutación en los algoritmos evolutivos, nos permite saltar de una región óptima local hacia otra región óptima local, desplazandose a través de todo el espacio de búsqueda, con la esperanza de encontrar una solución óptima global.

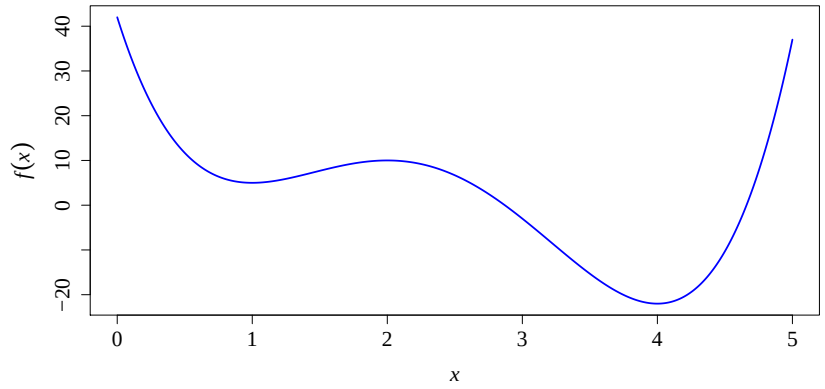


Figura 2.8 – Representación del espacio de búsqueda para $f(x) = 3x^4 - 28x^3 + 84x^2 - 96x + 42$.

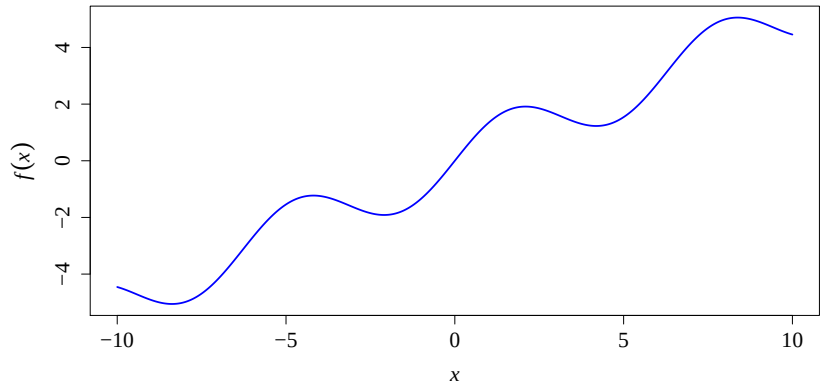


Figura 2.9 – Representación del espacio de búsqueda para $f(x) = \text{sen}(x) + 0.5x$.

La optimización unimodal está interesada en una sola solución óptima global. En la optimización multimodal, el objetivo es encontrar un conjunto de soluciones óptimas locales o globales. Por ejemplo, es posible que se quiera encontrar todas las soluciones óptimas locales. Por lo general, se está interesado en encontrar solo soluciones óptimas globales, pero también se puede estar interesado en encontrar soluciones que están cercanas al óptimo global; Por lo que el diseño de alguna estrategia especializada en lograr ésta tarea de búsqueda es de gran importancia.

Capítulo 3

Metodología propuesta

En este capítulo se presenta la metodología propuesta que permitió desarrollar el presente trabajo de tesis. Este capítulo se compone principalmente de dos partes, un método de muestreo y un algoritmo heurístico de búsqueda aleatoria para optimización global. En la sección 3.1 se describe el método de muestreo propuesto y se presenta evidencia experimental de las ventajas del método de muestreo sobre funciones que presentan un espacio de búsqueda multimodal, es decir, que presentan más de un óptimo local, o bien un óptimo global y múltiples óptimos locales. La sección 3.2 presenta un nuevo enfoque heurístico de búsqueda aleatoria dentro del campo de la computación evolutiva que tiene el potencial de encontrar soluciones buena calidad y que son robustas sobre distintas clases de problemas de optimización global. Básicamente el método de optimización propuesto integra el mecanismo de muestreo propuesto con un procedimento de búsqueda local, con el fin de explotar en las regiones prometedoras.

3.1. Muestreador de Gibbs-Cauchy

El método de muestreo propuesto se basa en el método MCMC híbrido descrito en la sección 2.4.3, llamado caminatas aleatorias de Metrópolis dentro de Gibbs. Las caminatas

aleatorias de Metrópolis dentro de Gibbs son el resultado de combinar la caminata aleatoria unidimensional de Metrópolis con el multidimensional muestreador de Gibbs. En este trabajo se proponen dos cambios importantes:

1. Se elige como la distribución propuesta $q(\cdot, \cdot)$ una distribución simétrica de Cauchy. La distribución de Cauchy ocasionalmente permite dar saltos largos de una región a otra, asegurando que ninguna región del espacio de búsqueda tenga probabilidad cero de ser explorada. Otra distribución simétrica, por ejemplo la distribución Gaussiana, no comparte dicha propiedad, ya que las colas de la campana de la distribución de Cauchy son mucho más gruesas (más probabilidad hacia fuera) en comparación con las de la distribución normal que decaen exponencialmente (ver Figura 3.1).

2. El tamaño de los pasos (parámetros de escala) de la caminata aleatoria se auto-adaptan de acuerdo a la información obtenida durante el proceso de muestreo. La auto-adaptación de los parámetros de escala permite controlar la exploración de acuerdo a la estructura del espacio de búsqueda, permitiendo un desplazamiento eficiente sobre todo el espacio, explorando y explotando cada región.

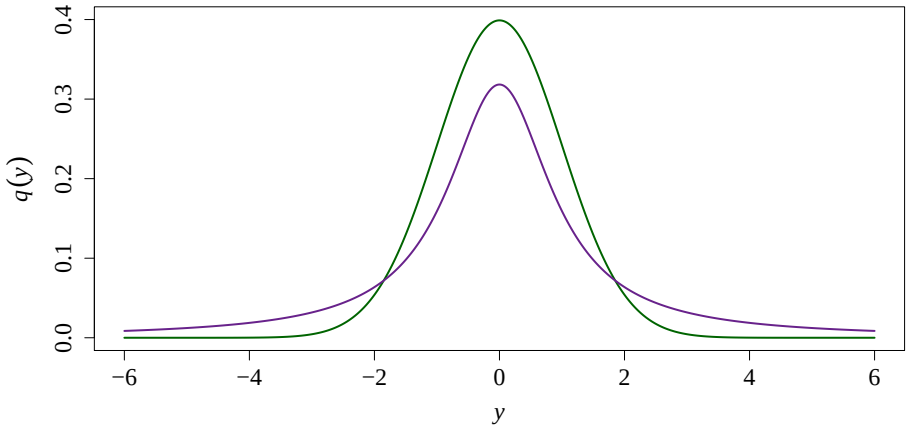


Figura 3.1 – La función de densidad de la distribución estandar de Cauchy (línea morada) y de la distribución normal estandar (línea verde).

El muestreador de Gibbs-Cauchy genera una secuencia de muestras $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, ..., \mathbf{x}_m$, a partir de la distribución de interés que se puede expresar de la forma $p(\mathbf{x}) = f(\mathbf{x})/Z$, de la siguiente manera:

Dado un vector inicial arbitrario $\mathbf{x}^o = (x_1, ..., x_i, ..., x_n)$ y un vector inicial de parámetros de escala $\sigma = (\sigma_1, ..., \sigma_i, ..., \sigma_n)$, un nuevo vector $\mathbf{x}$ puede construirse a partir de la actualización de cada componente x_i mediante la generación de un valor candidato por

$$y = x_i + \sigma_i \cdot C(0, 1) = C(x_i, \sigma_i) \tag{3.1}$$

donde $C(x_i, \sigma_i)$ es una perturbación aleatoria que sigue una distribución de Cauchy centrada en x_i (mutación de Cauchy). Esto es, el valor candidato y obedece a la función de densidad

$$q(y; x_i, \sigma_i) = \frac{1}{\pi\sigma_i \left\{1 + \left(\frac{y-x_i}{\sigma_i}\right)^2\right\}}. \tag{3.2}$$

Por lo tanto, el valor candidato $y|\mathbf{x}_{-i}$, donde $\mathbf{x}_{-i} = (x'_1, ..., x'_{i-1}, x_{i+1}, ..., x_n)$, se acepta como $x'_i = y$, de acuerdo con la probabilidad de aceptación de la ecuación (2.4) definida como,

$$\alpha(x_i, y) = \text{mín}\left\{1, \frac{p(y|\mathbf{x}_{-i})}{p(x_i|\mathbf{x}_{-i})}\right\} \tag{3.3}$$

si no rechazar con probabilidad $1 - \alpha(x_i, y)$, dejando el valor de la variable sin cambio, esto es $x'_i = x_i$. Cuando la distribución $p(\mathbf{x})$ es definida como la distribución canónica (distribución de Boltzmann) con respecto a alguna función de energía $f(\mathbf{x})$, y tomamos la temperatura como 1, la distribución canónica es

$$p(\mathbf{x}) = \left(\frac{1}{Z}\right)\exp(-f(\mathbf{x}))$$

por lo tanto la función de aceptación de la ecuación (3.3) se reescribe como,

$$\alpha(x_i, y) = \min\{1, \exp(-[f(y|\mathbf{x}_{-i}) - f(x_i|\mathbf{x}_{-i})])\} \tag{3.4}$$

Note que la evaluación de la probabilidad de aceptación no requiere del conocimiento de la constante de normalización Z.

Por otro lado, la dinámica de la ecuación (3.4) genera un vector de tasas de aceptación $\theta = (\theta_1, ..., \theta_i, ..., \theta_n)$, que esta dado por la razón entre los cambios aceptados y el total de cambios propuestos para cada variable. Entonces, después de generar m muestras con los pasos anteriores, $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_m$, el muestreador de Gibbs-Cauchy ajusta los parámetros de escala mediante la siguiente regla:

$$\sigma_i = \begin{cases} \sigma_i \cdot \exp\left(\tau_0 \cdot \mathcal{N}(0,1) + \tau \cdot \mathcal{N}_i(0,1)\right) & \text{si} \quad \theta_i < \epsilon_1 \\ \sigma_i & \text{si} \quad \epsilon_1 \leq \theta_i \leq \epsilon_2 \\ \sigma_i / \exp\left(\tau_0 \cdot \mathcal{N}(0,1) + \tau \cdot \mathcal{N}_i(0,1)\right) & \text{si} \quad \theta_i > \epsilon_2 \end{cases}$$

donde $\epsilon_1, \epsilon_2 \in [0, 1]$ y $\mathcal{N}(0, 1)$ es un valor aleatorio que sigue una distribución normal estandar con media 0 y desviación estandar 1 (ver Figura 3.1). Los parámetros de aprendizaje son

$$\tau_0 = \frac{1}{\sqrt{2n}} \quad \text{y} \quad \tau = \frac{1}{\sqrt{2\sqrt{n}}}$$

Una vez que los parámetros de escala σ se ajustaron, se elige un vector $\mathbf{x}^o = (x_1, ..., x_i, ..., x_n)$ como punto de partida para la siguiente iteración tomando en cuenta la siguiente regla:

$$\mathbf{x}^o = \begin{cases} \mathbf{x} \quad \text{arbitrario} & \text{si} \quad \frac{\sum_{i=1}^{n} \theta_i}{n} > \epsilon_3 \\ \mathbf{x}_k \quad \text{en caso contrario} \end{cases}$$

Se puede observar que cuando en promedio las tasas de aceptación son mayores a un ϵ_3, donde $\epsilon_3 \in [0.7, 0.99]$, indica que se están aceptando muchos movimientos, por lo tanto solo se está explorando una región del espacio de búsqueda. Para evitar quedar atrapado, se reinicia el punto de partida arbitrariamente y se explora en una nueva región. En caso contrario, el muestreo continua en el último punto que exploró.

Los pasos descritos anteriormente se realizan un número G de iteraciones. Al finalizar todas las iteraciones, el muestreo de Gibbs-Cauchy obtiene un conjunto de muestras

$$\mathbf{X} = \left\{(\mathbf{x}_1, ..., \mathbf{x}_m)^{(1)}, (\mathbf{x}_1, ..., \mathbf{x}_m)^{(2)}, ..., (\mathbf{x}_1, ..., \mathbf{x}_m)^{(G)})\right\}$$

de manera que las muestras generadas en cada iteración del muestreo se aproximan a la distribución de interés $p(\mathbf{x})$.

3.1.1. Parámetros de escala óptimos con propuestas de Cauchy

Consideremos el comportamiento del muestreo de Gibbs-Cauchy en función de los parámetros de escala σ en la ecuación (3.1). Si la mayoría de los saltos propuestos son pequeños comparados con alguna medida de variabilidad de la distribución de interés, entonces, aunque a menudo se aceptarán estos saltos, la cadena se moverá lentamente y la exploración de la distribución de interés será relativamente ineficiente. Si los saltos propuestos son relativamente grandes en comparación con la escala de la distribución de interés, entonces muchos cambios no serán aceptados, y la cadena rara vez se mueve, y de nuevo la exploración de la distribución de interés se realizará ineficientemente. Esto sugiere que dada la forma particular de la distribución de salto propuesta, puede existir un parámetro de escala "moderado", entre esos dos extremos, donde el muestreo explore eficientemente la distribución de interés.

Neal y Roberts [22] realizaron un estudio para optimizar los parámetros de escala de una caminata aleatoria de Metrópolis con distribuciones propuestas de Cauchy. En el

estudio realizado se sugiere que las tasas de aceptación óptimas para propuestas de Cauchy de la forma (3.2), es de $\exp(-1) = 0.368$. Además, se concluye que la tasa de aceptación óptima es robusta a la elección de $f(\mathbf{x})$.

En este trabajo vamos a definir como los parámetros de escala adecuados, aquellos que mantienen a las tasas de aceptación dentro de un intervalo alrededor del valor de la tasa de aceptación óptima igual a 0.368, esto es, en el intervalo $[0.3, 0.4]$, lo que equivale a aceptar entre el 30 % y el 40 % de los candidatos propuestos. Con el fin de alcanzar el intervalo óptimo propuesto para las tasas de aceptación, los parámetros de escala se ajustarán de acuerdo a la siguiente regla,

$$\sigma_i = \begin{cases} \sigma_i \cdot \exp\left(\tau_0 \cdot \mathcal{N}(0,1) + \tau \cdot \mathcal{N}_i(0,1)\right) & \text{si} \quad \theta_i < 0.3 \\ \sigma_i & \text{si} \quad 0.3 \leq \theta_i \leq 0.4 \\ \sigma_i / \exp\left(\tau_0 \cdot \mathcal{N}(0,1) + \tau \cdot \mathcal{N}_i(0,1)\right) & \text{si} \quad \theta_i > 0.4 \end{cases}$$

La idea de la regla es asegurar que el nivel de perturbación se mantenga dentro de un intervalo adecuado, con el fin de realizar una exploración eficiente de la distribución de interés.

3.1.2. Muestreo de Gibbs-Cauchy sobre distribuciones multimodales

Es importante que el método de muestreo Gibbs-Cauchy se pueda mover entre los modos de la distribución de interés. Esto da soporte a la utilización de distribuciones propuesta de colas pesadas (heavy tailed) como las de la distribución de Cauchy, la cual puede desplazarse más fácilmente entre los modos distantes de una función de interés. Las muestras que se generan por el método propuesto representan correctamente la estructura del espacio de búsqueda de la función de interes. Para ilustrar lo anterior, en la Figura 3.2 y 3.3 se muestran los resultados una caminata aleatoria de Metrópolis con 10000 pasos usando propuestas de la distribución Gaussiana y propuestas de la distribución de Cauchy,

respectivamente. La distribución de interés trimodal es $p(x) \propto \exp\{-(x-0.1)^2/2(0.02)^2\} + \exp\{-(x-0.6)^2/2(0.02)^2\} + \exp\{-(x-0.8)^2/2(0.02)^2\}$. La Figura 3.2 se observa que el muestreo no representa correctamente a la distribución de interés, ya que no se explora uno de los modos de la distribución. Por otro lado, en la Figura 3.3 se observa que el muestreo representa correctamente a la distribución de interés, saltando de un modo a otro, explorando adecuadamente el espacio.

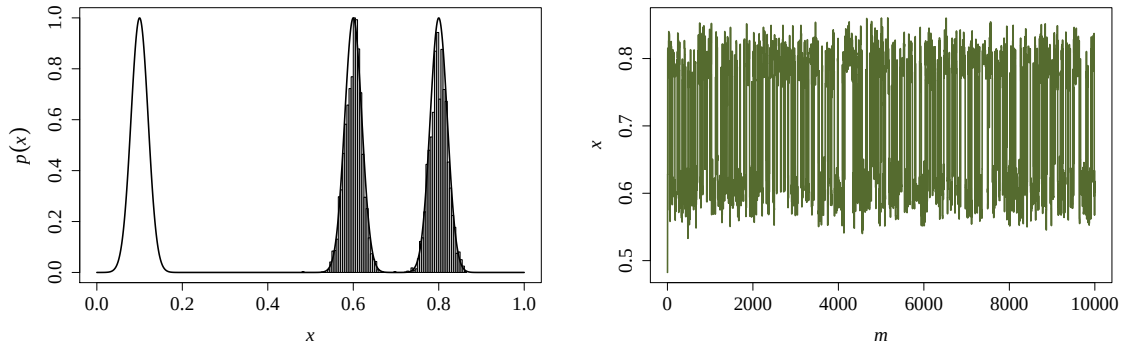


Figura 3.2 – Exploración de modos de una distribución de interés usando propuestas Gaussianas.

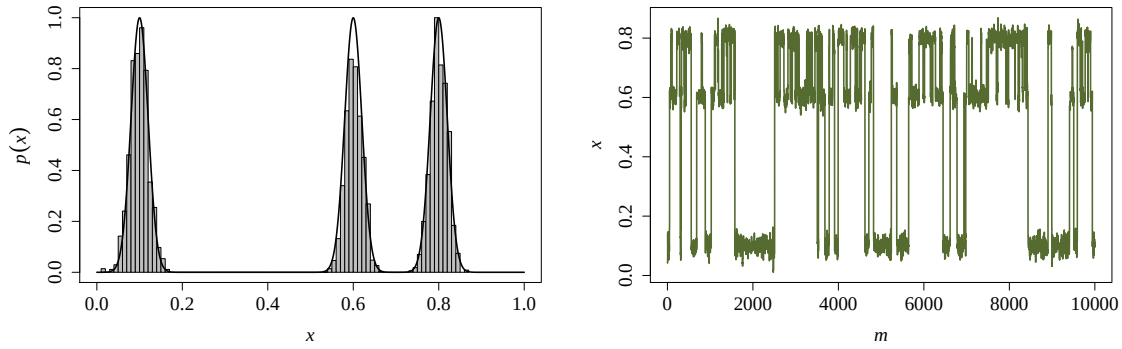


Figura 3.3 – Exploración de modos de una distribución de interés usando propuestas Cauchy.

Más evidencia experimental de las ventajas del muestreo de Gibbs-Cauchy en casos con modos bien separados y espacios de búsqueda relativamente amplios y complejos, se presenta a continuación.

Ejemplos de muestreo en funciones unidimensionales

Con el fin de ilustrar el potencial del muestreador de Gibbs-Cauchy para generar muestras a partir de una distribución de interés con distintas estructuras del espacio, se proponen las siguientes funciones unidimensionales:

$$f_1(x) = x^2 - 1 \qquad (-10 \leq x \leq 10) \tag{3.5}$$

$$f_2(x) = 3x^4 - 28x^3 + 84x^2 - 96x + 42 \qquad (0 \leq x \leq 5) \tag{3.6}$$

$$f_3(x) = \text{sen}(x) + 0.5x \qquad (-10 \leq x \leq 10) \tag{3.7}$$

$$f_4(x) = \exp(\text{sen}(50x)) + \text{sen}(70\,\text{sen}(x)) \qquad (-1 \leq x \leq 1) \tag{3.8}$$

Los resultados de la simulación del muestreador Gibbs-Cauchy se presentan en las Figuras 3.4, 3.5, 3.6 y 3.7. En cada figura se muestran dos imágenes, a lado izquierdo se muestra con una línea de color azul la forma de la función, y el histograma representa la frecuencia de las muestras generadas sobre dicha función. En la parte derecha de la figura, se muestra el comportamiento de la caminata aleatoria Gibbs-Cauchy después de realizar m pasos.

Considerando el caso con un modo, que resulta de la ecuación (3.5), se puede observar en la Figura 3.4 que las muestras generadas por el método propuesto, explora perfectamente el único modo (mínimo) de la función, concentrándose la mayor parte de las muestras alrededor de $x = 0$.

Ahora consideremos el caso con dos modos, que resulta de la ecuación (3.6). La Figura 3.5 muestra que la caminata de Gibbs-Cauchy localiza ambos modos de la función, saltando de una región a otra. Observe que la mayor parte de las muestras se ubican en la región más atractiva de la función, es decir, donde esta el mejor valor mínimo de la función.

Considerando el caso con tres modo, que resulta de la ecuación (3.7), se puede observar en la Figura 3.6 que la caminata aleatoria Gibbs-Cauchy salta de un modo mínimo

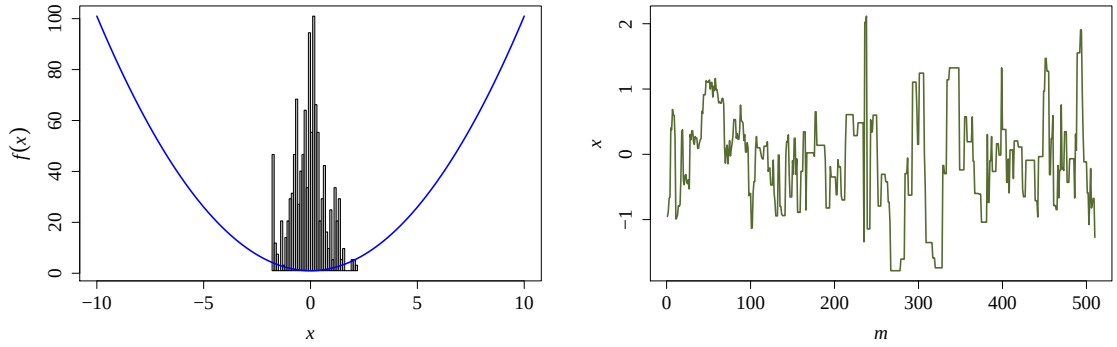


Figura 3.4 – Muestras generadas por Gibbs-Cauchy para la función $f_1(x) = x^2 - 1$ representada por la línea azul.

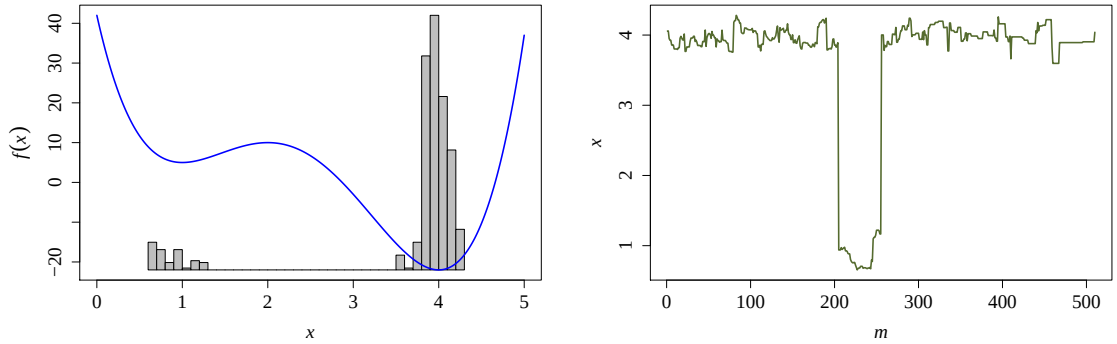


Figura 3.5 – Muestras generadas por Gibbs-Cauchy para la función $f_2(x) = 3x^4 - 28x^3 + 84x^2 - 96x + 42$ representada por la línea azul.

a otro modo mínimo, ubicandose la mayor parte del tiempo sobre el modo más atractivo, es decir, donde se localiza el mejor valor mínimo de la función.

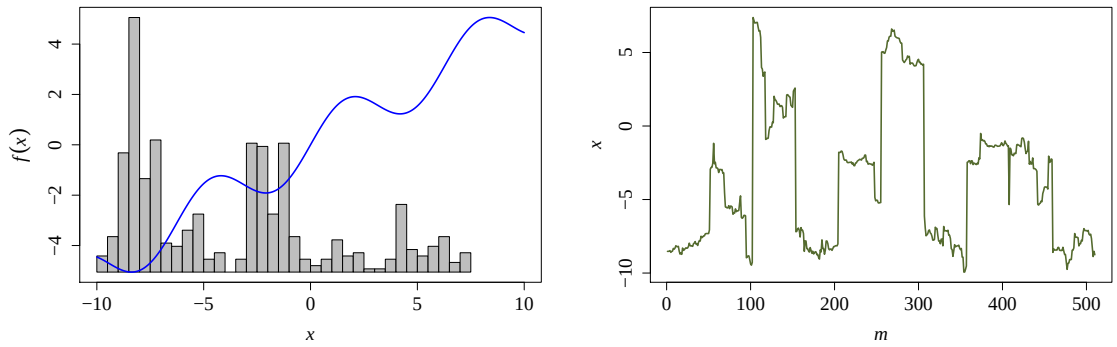


Figura 3.6 – Muestras generadas por Gibbs-Cauchy para la función $f_3(x) = \text{sen}(x) + 0.5x$ representada por la línea azul.

Por último, considerando el caso con múltiples modos, que resulta de la ecuación (3.8), se puede observar en la Figura 3.7 que la función exhibe una estructura muy rugosa y compleja. Note que la caminata de Gibbs-Cauchy salta de una región a otra, explorando todos los modos de la función de interés. Además, se puede observar que la caminata aleatoria explora la mayor parte del tiempo las cuatro regiones más atractivas donde se localizan los mejores mínimos de la función.

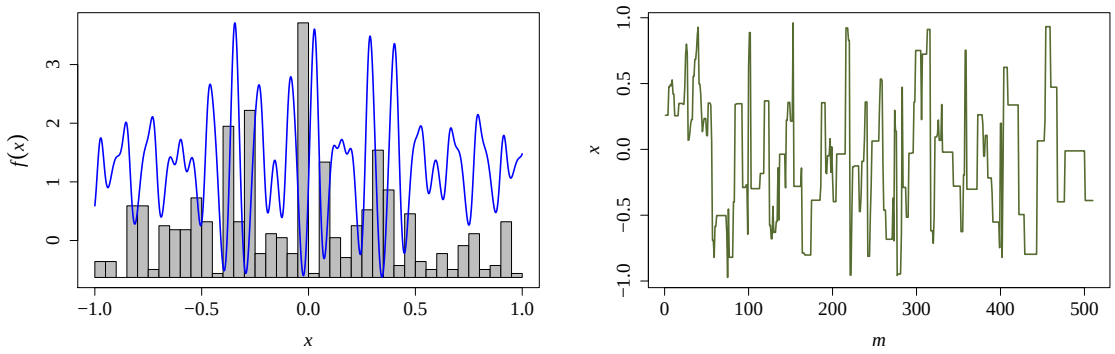


Figura 3.7 – Muestras generadas por Gibbs-Cauchy para la función $f_4(x) = \exp(\text{sen}(50x)) + \text{sen}(70\,\text{sen}(x))$ representada por la línea azul.

Los parámetros utilizados en esta simulación son en particular elegidos como una configuración robusta para todas las funciones utilizadas aquí. Los parámetros para el muestreador de Gibbs-Cauchy se presentan en el Cuadro 3.1.

Parámetro	valor
σ	0.1
ϵ_3	0.7
G	10
m	50

Cuadro 3.1 – Configuración de los parámetros utilizados en el muestreo Gibbs-Cauchy sobre las funciones unidimensionales.

Ejemplos de muestreo sobre distribuciones de probabilidad

Con el fin de muestrear una distribución de probabilidad $\pi(\mathbf{x})$, la función de energía que entra en el muestreador de Gibbs-Cauchy debe darse por

$$f(\mathbf{x}) = -\ln \pi(\mathbf{x})$$

El desempeño del muestreador Gibbs-Cauchy para generar muestras a partir de una distribución de probabilidad, en donde la densidad de las muestras generadas se aproximan a la estructura de la distribución, va a ser ilustrado sobre tres distribuciones elegidas por tener sus modos bien separados sobre el espacio de muestra. Los resultados de la simulación se presentan en las Figuras 3.8, 3.9 y 3.10. Para cada figura se muestran dos imágenes, a lado izquierdo se muestra el perfil de la distribución de interés de color negro, y los puntos rojos sobre dicha distribución representan las muestras generadas por Gibbs-Cauchy. En la imagen de la derecha, se ilustran las muestras generadas por Gibbs-Cauchy sobre el plano $x_1 - x_2$.

Consideremos las siguientes distribuciones,

$$\pi_1(\mathbf{x}) = \frac{1}{Z} \exp\left(\frac{-(x_1^2 x_2^2 + x_1^2 + x_2^2 - 8x_1 - 8x_2)}{2}\right) \tag{3.9}$$

donde la constante de normalización $Z \approx 20216.335877$.

$$\pi_2(\mathbf{x}) = \frac{1}{Z} \exp\left(-\frac{x_1^2 + x_2^2 + (x_1 \cdot x_2)^2 - 24(x_1 \cdot x_2)}{2}\right) \tag{3.10}$$

donde la constante de normalización $Z \approx 3.5390 \times 10^{26}$

$$\pi_3(\mathbf{x}) = 0.2\mathcal{N}(5, 2) + 0.2\mathcal{N}(20, 2) + 0.6\mathcal{N}(40, 2) \tag{3.11}$$

En la Figura 3.8, que resulta de (3.9), se ilustra la estimación de la distribución

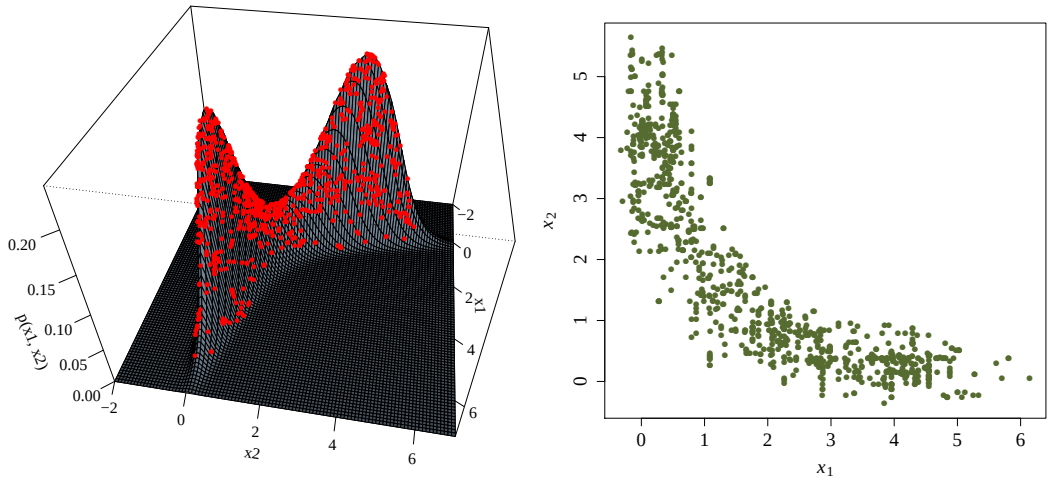


Figura 3.8 – Exploración de los modos de la distribución $\pi_1(\mathbf{x}) = Z^{-1}\exp(-(x_1^2x_2^2 + x_1^2 + x_2^2 - 8x_1 - 8x_2)/2$ usando el muestreador de Gibbs-Cauchy.

mediante el muestreador de Gibbs-Cauchy. Se generaron 1000 muestras para la estimación. En la imagen de la izquierda de la figura se puede observar que las muestras generadas por dicho método (puntos de color rojo) aproximan muy bien la forma de la distribución de interés con dos modos conectados. En la imagen de la derecha se ilustran las mismas muestras sobre el plano x_1-x_2, indicando que la región correcta ha sido muestreada.

En la Figura 3.9, que resulta de (3.10), se ilustra la estimación de la distribución mediante el muestreador de Gibbs-Cauchy. Se generaron 1000 muestras para la estimación. En la imagen de la izquierda de la figura se puede observar que las muestras generadas por dicho método (puntos de color rojo) aproximan muy bien la forma de la distribución de interés con dos modos bien separados. En la imagen de la derecha se ilustran las muestras sobre el plano x_1-x_2, indicando que la región correcta ha sido muestreada. Se observa que el muestreador de Gibbs-Cauchy tiene la capacidad de saltar regiones amplias de baja probabilidad, reduciendo el riesgo de quedar atrapado dentro de una región de alta probabilidad.

En la Figura 3.10, que resulta de la mezcla de densidades normales de (3.11), se ilustra la estimación de la distribución mediante el muestreador de Gibbs-Cauchy. Se

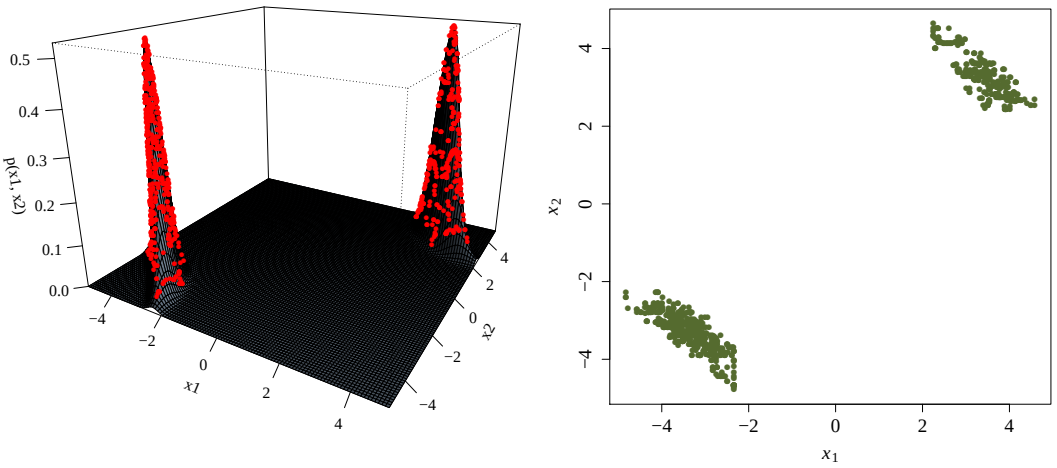


Figura 3.9 – Exploración de los modos de la distribución de interés $\pi_2(\mathbf{x}) = Z^{-1}\exp\{-(x_1^2 + x_2^2 + (x_1 \cdot x_2)^2 - 24(x_1 \cdot x_2))/2\}$, donde $Z \approx 3.5390 \times 10^{26}$, usando el muestreador de Gibbs-Cauchy.

generaron 1000 muestras para la estimación. En la imagen de la izquierda de la figura se puede observar que las muestras generadas por dicho método (puntos de color rojo) aproximan muy bien la forma de la distribución de interés con tres modos bien separados. En la imagen de la derecha se ilustran las muestras sobre el plano $x_1 - x_2$, indicando que la región correcta ha sido muestreada. Se observa que el muestreador de Gibbs-Cauchy tiene la capacidad de saltar regiones amplias de baja probabilidad, reduciendo el riesgo de quedar atrapado dentro de una región de alta probabilidad.

Los parámetros utilizados en esta simulación se eligieron como una configuración robusta para cada distribución. Los parámetros se presentan en el Cuadro 3.2.

	distribución		
Parámetro	π_1	π_2	π_3
σ	0.1	1.0	1.0
ϵ_3	0.7	0.7	0.7
G	10	10	10
m	100	100	100

Cuadro 3.2 – Configuración de parámetros utilizados para generar muestras a partir de distribuciones de probabilidad.

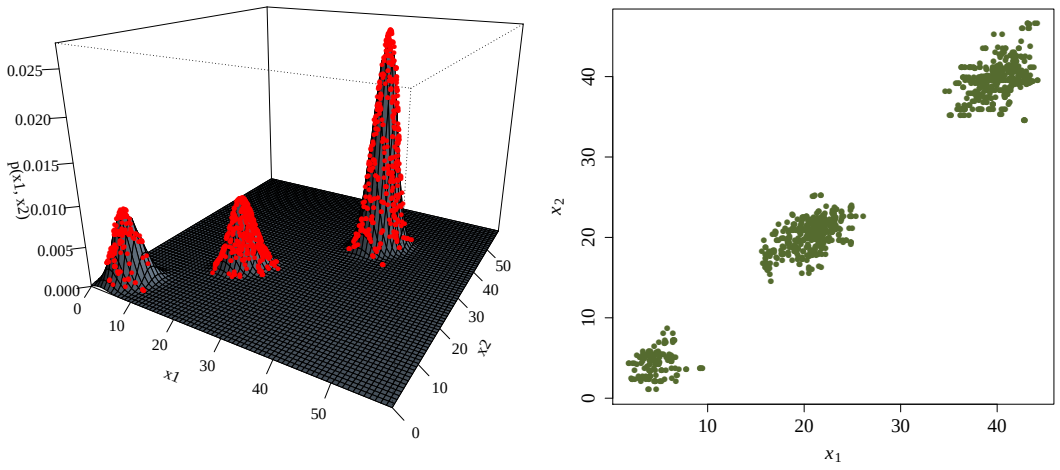


Figura 3.10 – Exploración de los modos de la distribución $\pi_3(\mathbf{x}) = 0.2\mathcal{N}(5,2) + 0.2\mathcal{N}(20,2) + 0.6\mathcal{N}(40,2)$ usando el muestreador de Gibbs-Cauchy.

3.2. RWM-ES: Estrategia evolutiva con caminatas aleatorias de Metrópolis

La estrategia evolutiva con caminatas aleatorias de Metrópolis (RWM-ES por sus siglas en inglés, *Random Walk Metropolis Evolution Strategy*), es una nuevo enfoque heurístico de busqueda y optimización dentro del campo de la computación evolutiva. El RWM-ES se basa en las ideas escenciales de las estrategias evolutivas, agregando un ingrediente particular de los métodos MCMC, el criterio de aceptación-rechazo de Metrópolis [2]. El criterio de Metrópolis garantiza que el proceso sigue la distribución de interés del problema que se quiere optimizar. La información acerca del espacio de búsqueda se refleja en las tasas de aceptación generadas por dicho criterio. Las tasas de aceptación son el porcentaje de las veces que se realizó un movimiento hacia un nuevo valor. Los movimientos propuestos provienen de una distribución de salto de Cauchy. La ventaja de los saltos de Cauchy es que reducen el riesgo de quedar atrapado dentro de una región de alta probabilidad (mínimo local), saltando extensas regiones de baja probabildad. La longitud o escala de los saltos de la caminata aleatoria se adaptan de acuerdo con la

información obtenida de las tasa de aceptación.

La clave principal del desempeño del método RWM-ES es que, en cada iteración, se mantiene un balance adecuado entre la exploración y explotación mediante la adaptación de sus parámetros. Lo anterior indica que el método RWM-ES previene la convergencia hacia un óptimo local, incrementando la probabilidad de encontrar un óptimo global. Además, con el fin de explotar en las regiones prometedoras, el método propuesto integra un procedimiento de búsqueda local mejorando significativamente los resultados.

Los principales pasos de la estrategia evolutiva con caminatas aleatorias de Metrópolis se describen a continuación.

1. **Inicialización:** Inicializar un vector $\mathbf{x}^o = (x_1, ..., x_i, ..., x_n)$ arbitrariamente y definir el vector de parámetros de escala con valores iniciales $\sigma^o = (\sigma_1, ..., \sigma_i, ..., \sigma_n)$.

2. **Muestreo:** Con los valores iniciales de $\mathbf{x}^o$ y σ^o generar una muestra mediante la actualización de cada componente x_i, en donde $i = 1, 2, ..., n$, desarrollando los siguientes pasos:

   a) Generar un valor propuesto $y = x_i + \sigma_i \cdot C(0, 1)$, donde $C(0, 1)$ es un valor aleatorio de la distribución estandar de Cauchy.

   b) Generar un valor aleatorio uniforme $u = \mathcal{U}[0, 1]$

   c) Si $u < \exp\{f(x_i|\mathbf{x}_{-i}) - f(y|\mathbf{x}_{-i})\}$, donde $\mathbf{x}_{-i} = (x'_1, ..., x'_{i-1}, x_{i+1}, ..., x_n)$, entonces $x'_i = y$; en caso contrario $x'_i = x_i$.

   Al finalizar los pasos anteriores, se termina un ciclo de Gibbs-Cauchy con una muestra $\mathbf{x}_k$ generada. Para obtener una nueva muestra, se realiza un nuevo ciclo de Gibbs-Cauchy a partir de $\mathbf{x}_k$. Con lo anterior, después de haber realizado m ciclos de Gibbs-Cauchy obtendremos un conjunto de muestras $\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_k, ..., \mathbf{x}_m$ que formará parte de la población de las soluciones candidatas. Además, se obtiene un vector de tasas de aceptación $\vec{\theta} = (\theta_1, ..., \theta_i, ..., \theta_n)$ que esta dado por la razón del

número de valores propuestos aceptados (criterio de Metrópolis) y el total de cambios propuestos.

3. **Selección:** Estimar la moda para cada variable de la población **X**. Cada componente de la moda representa los valores más probables de la población. En general, la moda extrae toda la información acerca de la población generada mediante el pasos del muestreo, tomando la región más probable de contener un óptimo global. En este trabajo, la moda se calcula haciendo uso del método de estimación de densidad basada en núcleos, más detalles en [24]. El método de estimación basada en núcleos aproxima una función de densidad de probabilidad sobre un conjunto de datos. La Figura 3.11 ilustra la estimación de la densidad a partir de un conjunto de datos (línea continua) y el máximo de la densidad representa la moda de esos datos (línea punteada). Dada las observaciones de cada columna de la población **X** con m datos, el estimador de densidad basado en nucleos se expresa de la siguiente manera:

$$\hat{f}(z) = \frac{1}{mh} \sum_{k=1}^{m} K\left(\frac{z - \mathbf{X}_k}{h}\right)$$

donde $K(x)$ es la función núcleo, también conocida como la función peso, que generalmente es una función simétrica como la de la distribución Gaussiana, y h es el parámetro de suavización o ancho de ventana. La salida de este paso es un vector moda $\mathbf{x}_{mod} = (x_1, ..., x_i, ..., x_n)$.

4. **Actualización:** Los parámetros de escala se ajustan mediante la siguiente regla:

$$\sigma_i = \begin{cases} \sigma_i \cdot \exp\left(\tau_0 \cdot \mathcal{N}(0,1) + \tau \cdot \mathcal{N}_i(0,1)\right) & \text{si} \quad \theta_i < 0.3 \\ \sigma_i & \text{si} \quad 0.3 \leq \theta_i \leq 0.4 \\ \sigma_i / \exp\left(\tau_0 \cdot \mathcal{N}(0,1) + \tau \cdot \mathcal{N}_i(0,1)\right) & \text{si} \quad \theta_i > 0.4 \end{cases}$$

donde $\mathcal{N}_i(0,1)$ es un valor aleatorio de la distribución Gaussiana estandar, $\tau_0 =$

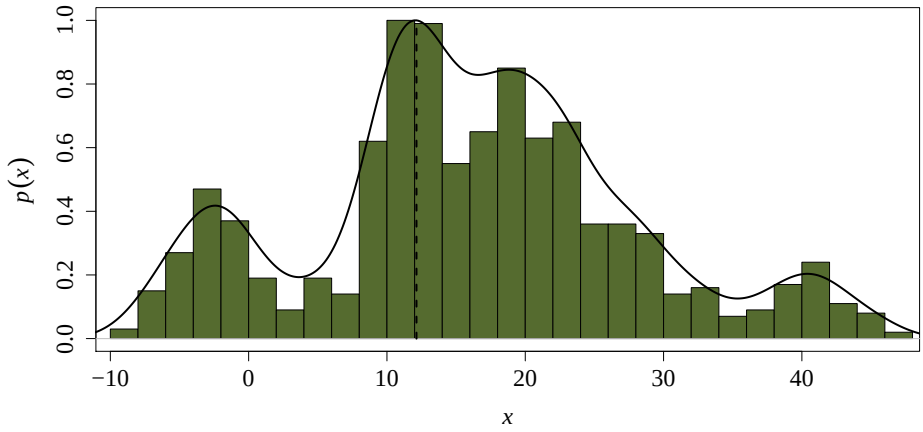


Figura 3.11 – Estimación de la moda (linea punteada) mediante el método de estimación de densidad basada en núcleos.

$\frac{1}{\sqrt{2n}}$ y $\tau = \frac{1}{\sqrt{2\sqrt{n}}}$. El vector $\mathbf{x}^o$ que se elige como punto de partida para la siguiente iteración, se actualiza mediante la siguiente regla:

$$\mathbf{x}^o = \begin{cases} \mathbf{x} \quad \text{arbitrario} & \text{si} \quad \frac{\sum_{i=1}^{n} \theta_i}{n} > \epsilon_3 \\ \mathbf{x}_{mod} \quad \text{en caso contrario} \end{cases}$$

Además, si se cumple que $\frac{\sum_{i=1}^{n} \theta_i}{n} > \epsilon_3$, se reinician el vector de parámetros de escala σ a sus valores iniciales predeterminados. Con el fin de explorar las regiones prometedoras descrita por la moda, el vector moda $\mathbf{x}_{mod}$ pasa como punto inicial de un procedimiento de búsqueda local, si la condición $\frac{\sum_{i=1}^{n} \theta_i}{n} > \epsilon_3$ se cumple; en caso contrario terminar la iteración.

El criterio de paro del algoritmo RWM-ES es cuando se alcanza un número máximo de iteraciones, o cuando un máximo número de evaluaciones de la función de costos se ha alcanzado. Por otro lado, se observa que el método RWM-ES no requiere de la evaluación de las derivadas de la función objetivo. Por lo tanto, es útil cuando se requiere tratar con funciones de costos estocásticas, muy rugosas (multimodales) y no diferenciables.

Ejemplo ilustrativo de optimización mediante RWM-ES

Con el fin de ilustrar el potencial del método propuesto, vamos a considerar el siguiente problema de optimización,

$$\text{mín } f(\mathbf{x}) = \text{sen}(x_1) \cdot \exp((1 - \cos(x_2))^2) + \cos(x_2) \cdot \exp((1 - \text{sen}(x_1))^2) + (x_1 - x_2)^2$$

donde el espacio de búsqueda es definido por $-6 \leq x_1, x_2 \leq 6$. Esta función minimización es conocida como el problema de Bird y tiene dos óptimos globales y unos pocos óptimos locales. Los óptimos globales son $\mathbf{x}^* = (-1.58, -3.13)$ y $\mathbf{x}^* = (4.7, 3.15)$, y tienen un valor de la función de costos $f^* = -106.764537$.

El proceso de búsqueda se inicia con un vector x^o arbitrario y con los parámetros de escala como $\sigma = (0.1, 0.1)$. En la Figura 3.12 se muestra las soluciones candidatas generadas mediante los pasos del muestreo sobre el perfil de la función (imagen de la izquierda), y sobre la gráfica de contorno en el plano $x_1 - x_2$ (imagen de la derecha). La caminata aleatoria de Gibbs-Cauchy se desarrollo con $m = 1000$ pasos. En la Figura 3.12 se observa claramente como la caminata de Gibbs-Cauchy explora perfectamente cuatro modos muy separados de la función de Bird, estos son, los dos modos óptimos globales y dos modos óptimos locales más favorables. La mayor parte de las muestras generadas se concentran sobre las regiones de los dos óptimos globales, ver Figura 3.13.

Una vez generadas las muestras por el paso del muestreo se procede al paso de selección, en el cual se estima la moda haciendo uso del método de estimación de densidades basada en nucleos. La Figura 3.13 muestra la estimación de la moda para la variable x_1 (izquierda) y para la variable x_2 (derecha). Se puede observar que la moda se estima alrededor de los valores para uno de los óptimos globales, esto es $\mathbf{x}_{mod} = (-1.763, -3.126)$, con valor de $f(\mathbf{x}) = -102.441$. Además, se puede observa que el otro óptimo global también se define perfectamente por la estimación de la densidad.

Con el procedimiento anterior, los pasos del muestreo y selección, se observa que

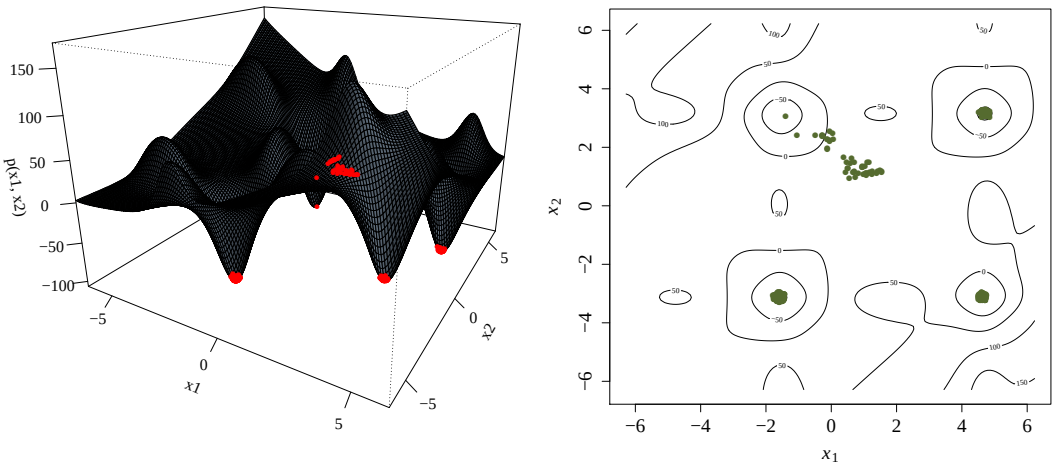


Figura 3.12 – Exploración de los modos de la función de Bird mediante el muestreo de Gibbs-Cauchy.

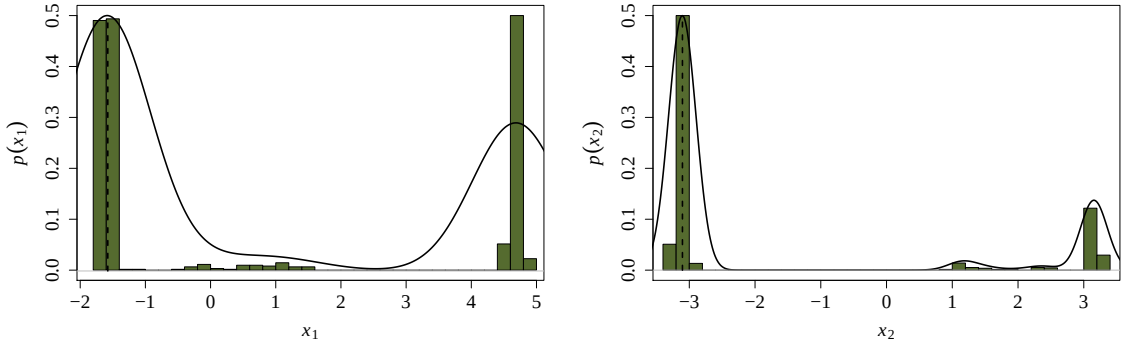


Figura 3.13 – Estimación de la moda (linea punteada) mediante el método de estimación de densidad basada en núcleos para la función de Bird.

se ha conseguido detectar muy bien los modos más probables, según la muestra generada. Entonces, la moda se estima sobre el modo más probable de contener uno de los óptimos globales, logrando tener una solución muy aproximada a los valores óptimos. Con el fin de conseguir una solución más fina, se procede a explotar la región prometedora definida por la moda mediante un procedimiento de búsqueda local. Después de someter el vector moda al procedimiento de búsqueda local, se obtiene el valor óptimo para las variables de $x_1 = -1.582136$ y $x_2 = -3.130218$, con un valor óptimo de la función de costos $f(\mathbf{x}) = -106.7645$.

3.2.1. Procedimiento de búsqueda local

En este trabajo se utiliza como procedimiento de búsqueda local una técnica de búsqueda directa conocida como método de Nelder-Mead o método Downhill Simplex. El método fue propuesto por Nelder y Mead en 1965 [23], y es un método simplex para encontrar una solución óptima local de una función de costos con n variables. El método solo utiliza evaluaciones de la función de costos y no de sus derivadas. En el espacio n-dimensional, un simplex es un poliedro con $n + 1$ puntos (vértices). Por ejemplo, en dos dimensiones, el simplex es un triángulo, en tres dimensiones es un tetraedro, y así sucesivamente.

La idea básica del método es comparar el valor de la función objetivo de los $n + 1$ puntos de un simplex e iterativamente mover el simplex hacia el punto óptimo mediante cuatro operaciones: reflexión, expansión, contracción y reducción. El método procede de la siguiente manera. Dado un punto inicial $\mathbf{x}$, el método genera otros n puntos que forman el simplex inicial. Supongamos que $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_{n+1}$ son los puntos actuales del simplex, el método tendrá que primero ordenar y renombrar estos puntos de tal manera que

$$f(\mathbf{x}_1) \leq f(\mathbf{x}_2) \leq f(\mathbf{x}_3) \leq ... \leq f(\mathbf{x}_{n+1})$$

Con lo anterior nos referimos a que $\mathbf{x}_1$ es el mejor de los puntos y $\mathbf{x}_{n+1}$ es el peor. Entonces, un nuevo punto se produce por actualizar al peor punto actual $\mathbf{x}_{n+1}$, o se puede producir n nuevos puntos que, junto con el mejor punto actual $\mathbf{x}_1$, formarán el simplex de la siguiente iteración. Una iteración del método de Nelder-Mead esta dada con los siguientes pasos:

1. **Ordenar:** Ordenar y renombrar los $n + 1$ puntos del simplex mediante

$$f(\mathbf{x}_1) \leq f(\mathbf{x}_2) \leq f(\mathbf{x}_3) \leq ... \leq f(\mathbf{x}_{n+1})$$

2. **Reflexión:** Calcular la reflexión del punto $\mathbf{x}_{ref}$ mediante

$$\mathbf{x}_{ref} = \bar{\mathbf{x}} + (\bar{\mathbf{x}} - \mathbf{x}_{n+1})$$

en donde $\bar{\mathbf{x}} = \frac{\sum_{i=1}^{n} \mathbf{x}_i}{n}$ es el centroide de los n puntos. Si $f(\mathbf{x}_1) < f(\mathbf{x}_{ref}) < f(\mathbf{x}_n)$, actualizar $\mathbf{x}_{n+1}$ por $\mathbf{x}_{ref}$ y terminar esa iteración.

3. **Expansión:** Si $\mathbf{x}_{ref}$ es mejor que $\mathbf{x}_1$, esto es $f(\mathbf{x}_{ref}) < f(\mathbf{x}_1)$, calcular la expansión del punto $\mathbf{x}_{exp}$ mediante

$$\mathbf{x}_{exp} = \bar{\mathbf{x}} + 2(\bar{\mathbf{x}} - \mathbf{x}_{n+1})$$

y actualizar $\mathbf{x}_{n+1}$ por el mejor punto de $\mathbf{x}_{ref}$ y $\mathbf{x}_{exp}$, y terminar esa iteración.

4. **Contracción interna:** Si $\mathbf{x}_{ref}$ no es mejor que $\mathbf{x}_{n+1}$, esto es $f(\mathbf{x}_{ref}) \geq f(\mathbf{x}_{n+1})$, calcular

$$\mathbf{x}_{ci} = \bar{\mathbf{x}} - \frac{1}{2}(\bar{\mathbf{x}} - \mathbf{x}_{n+1})$$

si $f(\mathbf{x}_{ci}) < f(\mathbf{x}_{n+1})$, actualizar $\mathbf{x}_{n+1}$ por $\mathbf{x}_{ci}$ y terminar esa iteración; en otro caso ir al paso 6.

5. **Contracción externa:** Si $\mathbf{x}_{ref}$ es mejor que $\mathbf{x}_{n+1}$, esto es $f(\mathbf{x}_n) \leq f(\mathbf{x}_{ref}) < f(\mathbf{x}_{n+1})$, calcular

$$\mathbf{x}_{ce} = \bar{\mathbf{x}} + \frac{1}{2}(\mathbf{x}_{ref} - \bar{\mathbf{x}})$$

si $f(\mathbf{x}_{ce}) \leq f(\mathbf{x}_{ref})$, actualizar $\mathbf{x}_{n+1}$ por $\mathbf{x}_{ce}$ y terminar esa iteración; en otro caso ir al paso 6.

6. **Reducción:** Calcular

$$\mathbf{y}_i = \mathbf{x}_1 + \frac{1}{2}(\mathbf{x}_i - \mathbf{x}_1), \qquad i = 2, 3, .., n + 1$$

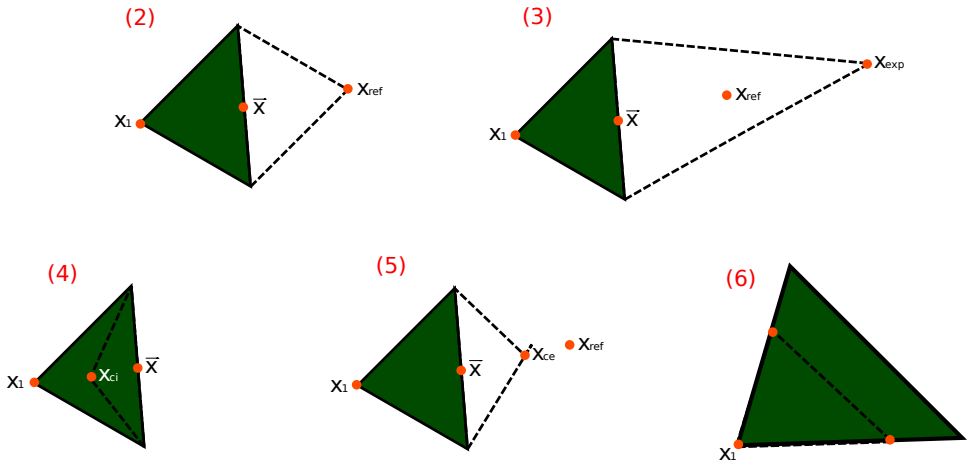


Figura 3.14 – Ejemplo hipotético de los operadores del método de Nelder-Mead en dos dimensiones. El operador (2) es la reflexión, (3) expansión, (4) contracción interna, (5) contracción externa y (6) reducción.

y actualizar $\mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_{n+1}$ por $\mathbf{y}_2, \mathbf{y}_3, ..., \mathbf{y}_{n+1}$ y terminar esa iteración.

La Figura 3.14 muestra los efectos de los pasos del método de Nelder-Mead para un simplex de dos dimensiones (triángulo). El triángulo de color verde representa el simplex original y la línea punteada representa los pasos del método. El método de Nelder-Mead requiere de pocas evaluaciones de la función de costos en cada iteración, y es uno de los métodos de búsqueda directa utilizado sobre problemas de optimización no lineal que tienen una única solución óptima. La implementación computacional del método de Nelder-Mead que se utiliza en este trabajo es la del paquete de `R` llamando `neldermead`, disponible en http://cran.r-project.org/web/packages/neldermead/.

Capítulo 4

El paquete RWM-ES en R

La motivación principal de implementar la metodología propuesta como un paquete escrito en el lenguaje R puro, es extender el conjunto de algoritmos para optimización global en R, y que esté disponible para la comunidad interesada en resolver dichos problemas. Otros paquetes de optimización se resumen en http://CRAN.R-project.org/view=Optimization. La ventaja de R con respecto a otros lenguajes de programación, en muchos de los casos, es que te permite la rápida creación de funciones para realizar cálculo numérico, así como acceder a una amplia gama de herramientas para modelado estadístico, y la capacidad de crear gráficos personalizados de los resultados obtenidos con mucha facilidad. Además, R es un software de código abierto y liberado bajo los términos de la licencia de GNU (*General Public Licence*), lo que implica que los paquetes implementados pueden adquirirse sin costo.

El paquete `RWMES` está publicado en la red integral de archivos de R (CRAN por sus siglas en inglés, the *Comprehensive* R *Archive Network*), y disponible en el sitio de internet http://cran.r-project.org/.

4.1. Implementación computacional

La implementación de la estrategia evolutiva con caminatas aleatorias de Metrópolis se utiliza mediante la llamada a la funcion `RWMES` de la siguiente manera:

```
RWMES(fn, Nvar = NULL, Lmin = NULL, Lmax = NULL, ..., control = list(),
 options = list())
```

Los argumentos de entrada de la implementación son:

- `fn:` Es la función de costos que debe de minimizarse. Esta función debe tener como primer argumento de entrada el vector **x** a optimizar y regresa un escalar como resultado.
- `Nvar:` Es un escalar que especifica el número de variables a optimizar.
- `Lmin:` Es un vector que especifica los límites inferiores para cada variable a optimizar.
- `Lmax:` Es un vector que especifica los límites superiores para cada variable a optimizar.
- `...:` Permite que el usuario pase argumentos adicionales a la función `fn`.
- `control:` Es una lista de parámetros de control, discutidos más adelante.
- `options:` Es una lista de parámetros de opciones, discutidos más adelante.

Los argumentos de `control` es una lista que especifica cualquier cambio a los valores predeterminados de los parámetros de la implementación. Los nombres de los argumentos de `control` deben especificarse completamente. Los elementos de la lista son los siguientes:

- `max_evaluations:` Es un escalar que espacifica el número máximo de evaluaciones de la función de costos. Por defecto es 5000*Nvar.

- `G_max_iterations:` Es un escalar que espacifica el número máximo de iteraciones. Por defecto es 100*Nvar.

Los argumentos de `options` es una lista que especifica cualquier cambio a los valores predeterminados de los parámetros de la implementación. Los nombres de los argumentos de `options` deben especificarse completamente. Los elementos de la lista son los siguientes:

- `m:` Es un escalar que especifica el número de muestras generadas en cada iteración muestreador de Gibbs-Cauchy. Por defecto es 100.

- `sigma0:` Es un vector de parámetros de escala iniciales del muestreador de Gibbs-Cauchy. Por defecto es 0.1 para las n variables.

- `epsilon1:` Es un escalar que especifica el nivel mínimo permitido para las tasas de aceptación en el intervalo [0,1]. Por defecto es 0.3.

- `epsilon2:` Es un escalar que especifica el nivel máximo permitido para las tasas de aceptación en el intervalo [0,1]. Por defecto es 0.4.

- `epsilon3:` Es un escalar que especifica el nivel permitido de la tasa de aceptación para que reinicie el punto inicial del muestreo en el intervalo [0,1]. Por defecto es 0.7.

- `max_local:` Es un escalar que especifica el número de iteraciones permitido en la estrategia de búsqueda local. Por defecto es 2000.

- `plotfn:` Permite al usuario habilitar la opción de graficar los resultados de la implementación. Se grafica el número de evaluaciones contra la función de costos. Por defecto es FALSE.

Los argumentos de salida de la implementación son:

- `par:` Es un vector de parámetros encontrado, minimizando la función de costos.

- `value:` Es el valor mínimo de la funcion de costos en el mejor conjunto de parámetros encontrado.
- `ctime:` Es el tiempo de cómputo requerido por el método.
- `eval:` Es el número de evaluaciones de la función de costos.
- `iter:` Es el número de iteraciones realizadas.

4.2. Ejemplo de uso del paquete

Antes de iniciar a hacer uso de la implementación, es necesario invocar al paquete `RWMES` mediante el comando de `R` como

```
library(RWMES)
```

Para ilustrar el uso del paquete `RWMES` se procede a optimizar la función de Rosenbrock de n variables. La función generalizada de Rosenbrock puede ser representada en una función de `R` como

```
myfn <- function(x){
Nvar <- length(x)
val <- sum(100*(x[1:(Nvar-1)]^2 - x[2:Nvar])^2 + (x[1:(Nvar-1)]-1)^2)
return(val)
}
```

El mínimo de la función de Rosenbrock es 0, el cual puede ser localizado por la función `RWMES` para $n = 10$ mediante la llamada del paquete como:

```
Nvar <- 10
Lmin <- rep(-5, Nvar)
Lmax <- rep(5, Nvar)
```

```
optim.myfn <- RWMES(myfn, Nvar, Lmin, Lmax, control= list(
              max_evaluations = 10000))
```

Observe que el primer argumento del paquete es la función de costos `myfn` a optimizar. Después, se especifica el número de variables que toma la función a optimizar. Posteriormente se indican los parámetros de límites inferior y superior de las variables. Por último, se modifica el valor predeterminado del máximo número de evaluaciones de la función de costos a través de los parámetros de control, igual a 10000.

La impresión de los resultados obtenidos por el paquete `RWMES` pueden ser mostrados mediante las siguientes instrucciones de `R`

```
cat("Mejor solución encontrada: ", optim.myfn$par, "\n")
cat("Valor de la función de costos: ", optim.myfn$value, "\n")
cat("Tiempo de cómputo (sec): ", optim.myfn$ctime, "\n")
cat("Número de evaluaciones de fn: ", optim.myfn$eval, "\n")
cat("Número de iteraciones: ", optim.myfn$iter, "\n")
```

Por otro lado, la implementación `RWMES` puede manejar fácilmente las restricciones de un problema de optimización mediante la representación de las mismas en una función de `R` con la siguiente estructura:

```
myfn2 <- function(x)
{
g <- vector(mode = "numeric")
h <- vector(mode = "numeric")
l <- c(0, 0, -0.55, -0.55)
u <- c(1200, 1200, 0.55, 0.55)
if(any(x < l) | any(x > u)){
value <- 100000
```

```
}else{
value = 3.0*x[1] + 0.000001*x[1]^3 + 2.0*x[2] +
(0.000002/3.0)*x[2]^3
g[1] = -x[4] + x[3] - 0.55
g[2] = -x[3] + x[4] - 0.55
h[1] = 1000.0*sin(-x[3] - 0.25) + 1000.0*sin(-x[4] - 0.25) +
894.8 - x[1]
h[2] = 1000.0*sin(x[3] - 0.25) + 1000.0*sin(x[3] - x[4] - 0.25) +
894.8 - x[2]
h[3] = 1000.0*sin(x[4] - 0.25) + 1000.0*sin(x[4] - x[3] - 0.25) +
1294.8
if(any(g > 0)){value <- 100000}
if(any(abs(h)-0.0001 > 0)){value <- 100000}
}
return(value)
}
```

Note que solo es necesario imponer una condición de penalización cuando la restricción es violada, es decir, que el valor de la función de costos es igual a un número muy grande cuando se viola la restricción. Con lo anterior, la solución candidata no será aceptada por el criterio de Metrópolis, alejandose de esa zona de baja probabilidad.

Mediante estos ejemplos se puede ver que es muy fácil probar el potencial del paquete `RWMES`, bajo un ambiente del lenguaje R, sobre otros problemas de optimización global. Además, es posible comparar su desempeño contra otros métodos implementados y disponibles en el CRAN.

Capítulo 5

Experimentación computacional

En esta sección, se presentan la experimentacion computacional del método de optimización propuesto sobre dos conjuntos de prueba para optimización global. Los conjuntos de prueba que se utilizaron para evaluar el desempeño de la método, son conjuntos estándar propuestos en las competencias del congreso de computación evolutiva (CEC por sus siglas en ingles, *Congress on Evolutionary Computation*), en los años 2005 y 2006. Los resultados obtenidos son discutidos en base a la calidad de las soluciones y a la robustez del método propuesto para tratar problemas restringidos e irrestrictos de optimización. El término de calidad se refiere a que tan cercanas son las soluciones obtenidas por el método con respecto a la mejor solución conocida o la óptima global. El término robusto se refiere a que los valores promedios son aproximaciones cercanas a la mejor solución conocida o la óptima global. Este comportamiento también debería de ser acompañado por una desviación estándar baja. Cuanto menor es la variabilidad de los resultados, el método se considera más robusto. Por lo tanto, un algoritmo robusto es aquel que consistentemente alcanza muy buena aproximacion a la solución óptima global (o alcanza la solución óptima global en cada ejecución).

Los experimentos realizados en este trabajo de tesis fueron ejecutados bajo el sistema operativo Ubuntu 13.04 en computadora de escritorio con procesador Intel I7-3820 con 4

núcleos a 3.60 GHz y 32 GB de memoria RAM.

5.1. Funciones de prueba CEC 2005

Para evaluar y comparar el desempeño de la estrategia evolutiva con caminatas aleatorias de Metrópolis (RWM-ES) en resolver problemas de optimización global sin restricciones, se ha considerado las 25 funciones de prueba proporcionadas por Suganthan et al. [25] en la Sesión Especial "Real-parameter optimization" del IEEE CEC 2005. En la Cuadro 5.1 se muestra la relación de las funciones consideradas, incluyendo el valor óptimo de cada una de ellas y los rangos del espacio de búsqueda. Hay que destacar el hecho de que todas las funciones se encuentran desplazadas para evitar la simetría y la localización centrada del óptimo. Las figuras de las funciones se muestran en el Apéndice A.

Función	Nombre	Óptimo global	Rangos de búsqueda
	Unimodal		
F1	Función esfera desplazada	-450	[-100, 100]
F2	Problema de Schwefel desplazado 1.2	-450	[-100, 100]
F3	Función elíptica desplazada condicionada a alta rotación	-450	[-100, 100]
F4	Problema desplazado de Schwefel con ruido en fitness 1.2	-450	[-100, 100]
F5	Problema de Schwefel con óptimo global en límites 2.6	-310	[-100, 100]
	Multimodal		
F6	Función de Rosenbrock desplazada	390	[-100, 100]
F7	Función de Griewank rotada desplazada sin límites	-180	no acotado
F8	Función de Ackley desplazada rotada con óptimo global en límites	-140	[-32, 32]
F9	Función de Rastringin desplazada	-330	[-5, 5]
F10	Función de Rastringin desplazada y rotada	-330	[-5, 5]
F11	Función de Weiestrass desplazada y rotada	90	[-0.5, 0.5]
F12	Problema de Schwefel 2.13	-460	$[-\pi, \pi]$
F13	Función de Rosenbrock con Griewank extendida expandida (F8F6)	-130	[-3, 1]
F14	Función de Scaffer expandida rotada desplazada F6	-300	[-100, 100]
	Hibridación de multimodales		
F15	Función de composición híbrida	120	[-5, 5]
F16	Función de composición híbrida rotada	120	[-5, 5]
F17	Función de composición híbrida rotada con ruido en fitness	120	[-5, 5]
F18	Función de composición híbrida	10	[-5, 5]
F19	Función de composición híbrida con curvatura estrecha para el óptimo global	10	[-5, 5]
F20	Función de composición híbrida rotada con el óptimo global en el límite	10	[-5, 5]
F21	Función de composición híbrida rotada	360	[-5, 5]
F22	Función de composición híbrida rotada con matriz de alto número de condición	360	[-5, 5]
F23	Función de composición híbrida rotada no continua	360	[-5, 5]
F24	Función de composición híbrida rotada	260	[-5, 5]
F25	Función de composición híbrida sin límites	260	no acotado

Cuadro 5.1 – Funciones de prueba del IEEE CEC 2005.

5.1.1. Especificaciones de la experimentación

El estudio experimental se realizó de acuerdo a los criterios de evaluación del conjunto de prueba CEC 2005. Se evaluaron las 25 funciones de prueba para las dimensiones n = 10 y 30. Cada función de prueba es repetida 25 veces con inicialización uniforme sobre el espacio de búsqueda. El máximo número de evaluaciones de la función de costos permitido es de $n \cdot 10^4$, es decir, 100000 para el caso de 10 variables y 300000 para el caso de 30 variables. Una ejecución del método se termina antes de llegar al máximo número de evaluaciones si el valor de la función de error esta por debajo de 10^{-8}. La función de error es $|f(\mathbf{x}) - f(\mathbf{x}^*)|$, donde $\mathbf{x}$ es la solución propuesta y $\mathbf{x}^*$ es la solución óptima.

Por otro lado, los parámetros utilizados por el método propuestos fueron ajustados y elegidos como una configuración robusta para todas las funciones de prueba. Los parámetros utilizados se resumen en el Cuadro 5.2.

Parámetro	valor
σ	(1.0,...,1.0)
ϵ_3	0.7
m	100

Cuadro 5.2 – Configuración de los parámetros utilizados por el método RWM-ES sobre el conjunto de prueba CEC 2005.

5.1.2. Algoritmos de comparación

Con el fin de comparar los resultados del algoritmo RWM-ES, elegimos a los tres métodos que alcanzaron el mejor desempeño en el conjunto de funciones de prueba CEC 2005 [26]. Especificamente, los nombres de los métodos que se eligieron para comparar los resultados del RWM-ES son:

- G-CMA-ES: Estrategia de evolución mediante adaptación de la matriz de covarianza donde la población se incrementa en cada reinicialización (CMA-ES por sus siglas en inglés, *Covariance Matrix Adaptation Evolution Strategy*) [27].

- DE: Algoritmo de evolución diferencial (DE por sus siglas en inglés, *Differential Evolution algorithm*) [28].

- L-SaDE: Variante auto-adaptiva de la evolución diferencial (SaDE por sus siglas en inglés, *Self adaptive Differential Evolution algorithm*) [29].

- K-PCX: Algoritmo estacionario con búsqueda basada en población (PCX por sus siglas en inglés, *Population based steady-state algorithm*) [30].

- L-CMA-ES: Estrategia de reinicialización de un algoritmo de búsqueda local avanzado (*Advanced local search evolutionary algorithm*) [31].

5.1.3. Resultados y discusiones

Los resultados obtenidos en la experimentación del RWM-ES sobre las 25 funciones de prueba se presentan en los Cuadros 5.3-5.9.

Los Cuadros 5.3 y 5.4 presentan el número de evaluaciones (mínimo, mediana, máximo, promedio y su desviación) requeridos para alcanzar un nivel de precisión (segunda columna) que se establece para cada problema con 10 y 30 variables, respectivamente. Las funciones que aparecen en dichos cuadros solo son en las que se alcanzan tasa de éxito distintas de cero, por tal razón se descartan las funciones F16-F24. El nivel de precisión o tolerancia mide la distancia o el error entre la mejor solución encontrada y el óptimo global. La tasa de éxito se calcula básandose en el porcentaje de ejecuciones que alcanzan ese nivel, es decir, la razón entre el total de ejecuciones exitosas y el total de ejecuciones realizadas. Además, en la columna Desempeño se registra el número de evaluaciones de la función de costos necesarias para alcanzar ese nivel de precisión, esto es, Desempeño = promedio(evaluaciones para ejecuciones exitosas)*(total de ejecuciones)/(# de ejecuciones exitosas).

Se puede observar en el Cuadro 5.3 que el método propuesto alcanza el nivel de

precisión impuesto para las funciones F1, F2, F3, F4, F6, F9, F12 y F15, realizando en promedio un número de evaluaciones de la función de costos por debajo al impuesto, es decir, 100000 para $n = 10$. La función F9 y F12 presentan una tasa de éxito por debajo del 50 %, mientras que las funciones F1, F2, F3, F4, F6 y F15 presentan tasas de éxito altas. Por otro lado, en el Cuadro 5.4 se observa que el método alcanza el nivel de precisión para las funciones F1, F2, F4, F7 y F15, realizando en promedio un esfuerzo relativamente bajo de evaluaciones menor al impuesto, es decir, 300000 para $n = 30$. La función F1, F2 y F4 obtiene una tasa de éxito del 100 %, mientras que la función F7 y F15 tiene una tasa de éxito menor al 50 %. Note que los funciones unimodales F1, F2, F3, F4 y F6 son muy fáciles de resolver (dentro del error especificado) con el método propuesto. Las funciones multimodales F6, F9 y F12 solo se resuelve para la dimensión $n = 10$ y la función F7 solo se resuelve para la dimensión $n = 30$. La función multimodal híbrida F15 pudo ser resuelta en ambas dimensiones.

Función	Tol.	Min.	Med.	Max.	Prom.	Desv.	Tasa de éxito	Desempeño
F1	1e-6	1.89e+3	2.45e+3	5.19e+3	2.55e+3	6.23e+2	1.00	2.55e+3
F2	1e-6	2.76e+3	3.00e+3	5.78e+3	3.15e+3	7.50e+2	1.00	3.15e+3
F3	1e-6	1.50e+4	4.40e+4	1.00e+5	4.58e+4	2.27e+4	0.96	4.53e+4
F4	1e-6	2.43e+3	2.90e+3	5.92e+3	3.00e+3	6.25e+2	1.00	3.00e+3
F5	1e-6	·	·	·	·	·	0.00	·
F6	1e-2	6.60e+3	1.00e+4	3.84e+4	1.10e+4	6.36e+3	0.96	1.15e+4
F7	1e-2	·	·	·	·	·	0.00	·
F8	1e-2	·	·	·	·	·	0.00	·
F9	1e-2	5.70e+4	1.01e+5	1.02e+5	9.87e+4	9.03e+3	0.08	9.14e+5
F10	1e-2	·	·	·	·	·	0.00	·
F11	1e-2	·	·	·	·	·	0.00	·
F12	1e-2	6.46e+3	1.00e+5	1.02e+5	6.34e+4	4.66e+4	0.40	1.89e+4
F13	1e-2	·	·	·	·	·	0.00	·
F14	1e-2	·	·	·	·	·	0.00	·
F15	1e-2	1.10e+4	5.31e+4	1.01e+5	5.85e+4	2.81e+4	0.88	6.00e+4

Cuadro 5.3 – Número de evaluaciones requeridas para alcanzar un nivel de precisión dado para las funciones F1-F15 con dimensión $n = 10$.

Los Cuadros 5.5 y 5.6 presentan los valores de error (mínimo, mediana, máximo, promedio y su desviación) alcanzados para cada función realizando un determinado número de evaluaciones de la función de costos. Los puntos de revisión en términos del número de evaluación de la función de costos son de 1e3, 1e4 y 1e5. Los Cuadros 5.7 y 5.8 presentan los valores de error (mínimo, mediana, máximo, promedio y su desviación) alcanzados

Función	Tol.	Min.	Med.	Max.	Prom.	Desv.	Tasa de éxito	Desempeño
F1	1e-6	1.50e+4	2.50e+4	3.50e+4	2.56e+4	6.51e+3	1.00	2.56e+4
F2	1e-6	6.50e+4	8.00e+4	1.45e+5	8.56e+4	1.65e+4	1.00	8.56e+4
F3	1e-6	·	·	·	·	·	0.00	·
F4	1e-6	6.00e+4	8.50e+4	1.05e+5	8.56e+4	1.20e+4	1.00	8.56e+4
F5	1e-6	·	·	·	·	·	0.00	·
F6	1e-2	·	·	·	·	·	0.00	·
F7	1e-2	8.10e+4	3.01e+5	3.01e+5	2.20e+5	1.01e+5	0.40	2.49e+5
F8	1e-2	·	·	·	·	·	0.00	·
F9	1e-2	·	·	·	·	·	0.00	·
F10	1e-2	·	·	·	·	·	0.00	·
F11	1e-2	·	·	·	·	·	0.00	·
F12	1e-2	·	·	·	·	·	0.00	·
F13	1e-2	·	·	·	·	·	0.00	·
F14	1e-2	·	·	·	·	·	0.00	·
F15	1e-2	1.74e+5	3.01e+5	3.05e+5	2.92e+5	2.80e+4	0.16	1.51e+6

Cuadro 5.4 – Número de evaluaciones requeridas para alcanzar un nivel de precisión dado para las funciones F1-F15 con dimensión $n = 30$.

para cada función con $n = 30$. Los puntos de revisión para esta dimensión son 1e3, 1e4, 1e5 y 3e5 evaluaciones de la función de costos.

En los Cuadros 5.5-5.8 se observa que las soluciones encontradas por el método propuesto en cada función, cada punto de revisión y para cada dimensión, no presentan una variabilidad alta (fila Desv.). Además, se observa que en algunos casos como la función F3, F5, F6, F7, F9 y F13, en ambas dimensiones, es necesario de un poco más de evaluaciones de las impuestas para alcanzar mejores valores de la función de costos. Con lo anterior, se observa que el comportamiento del método es robusto ante las distintas clases de funciones y la dimensionalidad.

Eval.	Func.	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
1e3	Min.	0.00e+0	0.00e+0	4.61e+3	0.00e+0	6.21e+2	3.00e+2	1.40e+9	2.00e+1	1.59e+1	3.88e+1	6.29e+0	1.41e+1
	Med.	2.27e-13	2.84e-13	1.13e+5	2.27e-13	2.83e+3	3.38e+5	1.40e+9	2.00e+1	2.49e+1	6.17e+1	8.95e+0	2.60e+2
	Max.	9.09e-13	1.15e-09	8.65e+5	1.02e-9	6.71e+3	5.90e+6	1.40e+9	2.00e+1	3.28e+1	8.06e+1	1.12e+1	1.44e+4
	Prom.	2.59e-13	5.79e-11	2.09e+5	8.73e-11	3.10e+3	8.20e+5	1.40e+9	2.00e+1	2.44e+1	6.05e+1	8.91e+0	1.22e+3
	Desv.	2.37e-13	2.30e-10	2.32e+5	2.50e-10	1.76e+3	1.31e+6	1.40e+9	7.78e-12	5.03e+0	1.04e+1	1.18e+0	2.88e+3
1e4	Min.	0.00e+0	0.00e+0	2.03e-3	0.00e+0	7.44e+1	0.00e+0	1.48e-2	2.00e+1	8.95e+0	3.88e+1	3.31e+0	0.00e+0
	Med.	2.27e-13	2.84e-13	1.28e+1	2.27e-13	7.07e+2	1.48e-12	2.12e-1	2.00e+1	1.49e+1	6.17e+1	5.54e+0	1.00e+1
	Max.	9.09e-13	1.15e-09	2.56e+3	1.02e-9	1.56e+3	4.12e+1	3.18e+0	2.00e+1	2.29e+1	8.06e+1	8.02e+0	1.69e+3
	Prom.	2.59e-13	5.79e-11	1.91e+2	8.73e-11	7.62e+2	2.16e+0	4.35e-1	2.00e+1	1.57e+1	6.05e+1	5.67e+0	2.91e+2
	Desv.	2.37e-13	2.30e-10	5.39e+2	2.50e-10	4.35e+2	8.25e+0	7.54e-1	7.78e-12	4.52e+0	1.04e+1	1.30e+0	6.06e+2
1e5	Min.	0.00e+0	0.00e+0	2.84E-13	0.00e+0	1.84e+0	0.00e+0	1.48e-2	2.00e+1	0.00e+0	3.88e+1	1.22e+0	0.00e+0
	Med.	2.27e-13	2.84e-13	1.06E-09	2.27e-13	6.13e+1	3.98e-13	2.12e-1	2.00e+1	2.98e+0	6.17e+1	3.37e+0	1.00e+1
	Max.	9.09e-13	1.15e-09	2.96E-04	1.02e-9	2.32e+2	3.99e+0	6.96e-1	2.00e+1	5.97e+0	8.06e+1	6.44e+0	1.69e+3
	Prom.	2.59e-13	5.79e-11	1.18E-05	8.73e-11	7.30e+1	1.59e-1	2.17e-1	2.00e+1	2.87e+0	6.05e+1	3.60e+0	2.91e+2
	Desv.	2.37e-13	2.30e-10	5.91E-05	2.50e-10	5.75e+1	7.97e-1	1.70e-1	7.78e-12	1.61e+0	1.04e+1	1.45e+0	6.06e+2

Cuadro 5.5 – Los valores de error alcanzados en evaluaciones de la función = 1e3, 1e4, 1e5 para las funciones F1-F12 con dimensión $n = 10$.

Eval.	Func.	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24	F25
	Min.	6.77e-1	3.97e+0	6.16e+1	1.87e+2	2.29e+2	9.00e+2	9.00e+2	9.00e+2	5.00e+2	7.94e+2	8.55e+2	2.00e+2	4.35e+2
	Med.	2.13e+0	4.70e+0	1.77e+2	2.95e+2	3.93e+2	9.01e+2	9.02e+2	9.01e+2	1.21e+3	1.01e+3	1.29e+3	1.25e+3	1.14e+3
1e3	Max.	2.96e+0	4.94e+0	3.20e+2	5.93e+2	7.84e+2	1.28e+3	1.28e+3	1.57e+3	1.36e+3	1.58e+3	1.48e+3	1.61e+3	1.48e+3
	Prom.	1.93e+0	4.66e+0	1.86E+2	3.38e+2	4.04e+2	9.54e+2	1.02e+3	1.01e+3	1.17e+3	1.01e+3	1.25e+3	8.99e+2	9.52e+2
	Desv.	7.14e-1	2.51e-1	6.55e+1	1.02e+2	1.22e+2	1.14e+2	1.36e+2	1.83e+2	1.89e+2	1.46e+2	1.45e+2	5.45e+2	4.07e+2
	Min.	4.40e-1	3.97e+0	0.00e+0	1.84e+2	2.26e+2	9.00e+2	9.00e+2	9.00e+2	2.00e+2	7.80e+2	4.25e+2	2.00e+2	2.00e+2
	Med.	1.06e+0	4.47e+0	7.90e+1	2.93e+2	3.61e+2	9.00e+2	9.00e+2	9.00e+2	9.00e+2	8.92e+2	9.55e+2	2.00e+2	4.28e+2
1e4	Max.	2.47e+0	4.74e+0	3.02e+2	5.40e+2	5.65e+2	1.19e+3	1.18e+3	1.32e+3	1.25e+3	9.90e+2	1.24e+3	1.03e+3	1.46e+3
	Prom.	1.12e+0	4.40e+0	8.81e+1	3.23e+2	3.71e+2	9.43e+2	9.93e+2	9.71e+2	8.27e+2	8.83e+2	8.56e+2	2.57e+2	7.44e+2
	Desv.	5.30e-1	2.33e-1	5.78e+1	8.50e+1	8.87e+1	9.10e+1	1.10e+2	1.18e+2	3.41e+2	6.79e+1	3.01e+2	1.82e+2	4.58e+2
	Min.	4.40e-1	2.77e+0	0.00e+0	1.39e+2	1.60e+2	9.00e+2	9.00e+2	9.00e+2	2.00e+2	7.80e+2	4.25e+2	2.00e+2	2.00e+2
	Med.	7.39e-1	3.72e+0	2.98e-13	2.00e+2	2.26e+2	9.00e+2	9.00e+2	9.00e+2	4.10e+2	8.14e+2	5.59e+2	2.00e+2	4.21e+2
1e5	Max.	1.51e+0	4.38e+0	3.01E+2	2.71e+2	3.64e+2	1.13e+3	1.11e+3	1.08e+3	5.00e+2	9.07e+2	8.51e+2	2.00e+2	1.36e+3
	Prom.	8.64e-1	3.74e+0	1.57e+1	2.07e+2	2.38e+2	9.34e+2	9.71e+2	9.40e+2	4.41e+2	8.19e+2	5.69e+2	2.00e+2	5.92e+2
	Desv.	3.09e-1	3.18e-1	6.08e+1	4.22e+1	5.60e+1	7.25e+1	8.45e+1	6.25e+1	6.72e+1	3.20e+1	6.86e+1	9.35e-13	3.73e+2

Cuadro 5.6 – Los valores de error alcanzados en evaluaciones de la función = 1e3, 1e4, 1e5 para las funciones F13-F25 con dimensión $n = 10$.

Eval.	Func.	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
	Min.	6.85e+0	1.34e+3	2.11e+7	1.68e+3	1.15e+04	4.19e+7	1.11e+10	2.01e+1	9.63e+1	4.24e+2	2.43e+1	8.06e+3
	Med.	7.63e+2	6.28e+3	4.66e+7	6.87e+3	1.90e+04	3.28e+8	1.11e+10	2.02e+1	1.49e+2	4.46e+2	3.30e+1	2.03e+4
1e3	Max.	3.72e+3	1.10e+4	1.30e+8	1.65e+4	3.12e+04	2.04e+9	1.11e+10	2.04e+1	1.66e+2	5.19e+2	3.71e+1	5.10e+4
	Prom.	1.09e+3	6.22e+3	5.77e+7	6.88e+3	1.97e+04	5.26e+8	1.11e+10	2.02e+1	1.43e+2	4.54e+2	3.29e+1	2.62e+4
	Desv.	1.04e+3	2.39e+3	2.98e+7	3.50e+3	4.71e+03	4.88e+8	1.96e+5	9.37e-2	1.61e+1	2.72e+1	2.74e+0	1.37e+4
	Min.	9.47e-5	1.49e+2	1.92e+7	1.36e+2	6.61e+3	2.75e+7	3.44e+9	2.00e+1	6.67e+1	4.18e+2	2.10e+1	2.04e+3
	Med.	6.58e+1	2.26e+3	4.21e+7	2.47e+3	1.45e+4	2.25e+8	3.91e+9	2.01e+1	1.14e+2	4.40e+2	2.69e+1	1.33e+4
1e4	Max.	7.23e+2	4.89e+3	1.10e+8	6.61e+3	1.88e+4	1.66e+9	4.19e+9	2.03e+1	1.34e+2	5.13e+2	3.23e+1	4.30e+4
	Prom.	1.31e+2	2.25e+3	5.02e+7	2.74e+3	1.43e+4	4.20e+8	3.91e+9	2.01e+1	1.07e+2	4.51e+2	2.65e+1	1.57e+4
	Desv.	2.01e+2	1.08e+3	2.50e+7	2.00e+3	3.07e+3	4.45e+8	1.54e+8	6.24e-2	1.84e+1	2.70e+1	2.54e+0	1.07e+4
	Min.	1.50e-9	6.95e-9	8.96e+4	9.83e-9	2.28e+3	1.62e+1	3.09e-8	2.00e+1	2.39e+1	3.85e+2	1.97e+1	8.21e+0
	Med.	3.68e-9	6.34e-8	2.97e+5	1.60e-7	4.83e+3	2.98e+2	1.03e+0	2.00e+1	4.68e+1	4.33e+2	2.43e+1	9.34e+2
1e5	Max.	9.47e-9	9.88e-4	7.09e+7	3.44e-6	7.34e+3	1.19e+7	1.77e+1	2.00e+1	6.07e+1	5.13e+2	2.78e+1	1.08e+4
	Prom.	4.11e-9	4.00e-5	3.20e+6	3.69e-7	4.79e+3	5.45e+5	2.32e+0	2.00e+1	4.59e+1	4.36e+2	2.41e+1	2.10e+3
	Desv.	2.12e-9	1.98e-4	1.41e+7	7.08e-7	1.26e+3	2.38e+6	4.15e+0	3.72e-3	8.29e+0	3.22e+1	2.69e+0	2.81e+3
	Min.	1.50e-9	6.95e-9	1.35e+4	7.52e-9	1.98e+3	3.74e-1	3.81e-9	2.00e+1	9.95e+0	2.84e+2	1.36e+1	1.40e-1
	Med.	3.68e-9	8.70e-9	5.58e+4	9.45e-9	3.08e+3	9.19e+0	3.19e-2	2.00e+1	2.59e+1	3.97e+2	2.20e+1	4.77e+2
3e5	Max.	9.47e-9	9.95e-9	3.63e+5	9.96e-9	6.54e+3	1.28e+3	3.19e-2	2.00e+1	4.38e+1	5.13e+2	2.78e+1	1.08e+4
	Prom.	4.11e-9	8.68e-9	7.81e+4	9.28e-9	3.30e+3	9.81e+1	1.88e-2	2.00e+1	2.58e+1	3.93e+2	2.23e+1	1.94e+3
	Desv.	2.12e-9	7.83e-10	8.36e+4	6.64e-10	1.09e+3	2.89e+2	1.53e-2	1.54e-3	7.72e+0	5.51e+1	3.40e+0	2.88e+3

Cuadro 5.7 – Los valores de error alcanzados en evaluaciones de la función = 1e3, 1e4, 1e5 para las funciones F1-F12 con dimensión $n = 30$.

Eval.	Func.	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24	F25
	Min.	7.60e+0	1.29e+1	1.27e+2	4.15e+2	4.21e+2	9.02e+2	9.03e+2	9.02e+2	5.00e+2	9.93e+2	5.34e+2	2.00e+2	2.13e+2
	Med.	1.35e+1	1.42e+1	3.85e+2	5.34e+2	5.54e+2	9.04e+2	9.04e+2	9.04e+2	5.00e+2	1.32e+3	5.34e+2	1.04e+3	1.29e+3
1e3	Max.	1.57e+1	1.48e+1	4.79e+2	7.42e+2	9.85e+2	9.07e+2	9.06e+2	9.06e+2	1.25e+3	1.70e+3	1.32e+3	1.76e+3	1.43e+3
	Prom.	1.31e+1	1.42e+1	3.49e+2	5.52e+2	5.98e+2	9.05e+2	9.04e+2	9.04e+2	7.40e+2	1.32e+3	7.56e+2	9.98e+2	8.43e+2
	Desv.	1.80e+0	4.04e-1	9.70e+1	1.04e+2	1.43e+2	1.16e+0	9.17e-1	1.04e+0	3.15e+2	1.64e+2	2.72e+2	5.53e+2	5.69e+2
	Min.	3.64e+0	1.29e+1	5.90e+1	4.04e+2	3.94e+2	9.00e+2	9.00e+2	9.00e+2	4.10e+2	9.93e+2	5.34e+2	2.00e+2	2.12e+2
	Med.	5.70e+0	1.41e+1	3.35e+2	5.06e+2	5.48e+2	9.01e+2	9.00e+2	9.00e+2	5.00e+2	1.17e+3	5.34e+2	1.01e+3	1.29e+3
1e4	Max.	8.52e+0	1.44e+1	4.07e+2	7.13e+2	9.25e+2	9.02e+2	9.01e+2	9.00e+2	1.14e+3	1.47e+3	1.10e+3	1.49e+3	1.43e+3
	Prom.	5.70e+0	1.40e+1	2.83e+2	5.17e+2	5.65e+2	9.01e+2	9.01e+2	9.00e+2	5.50e+2	1.22e+3	5.57e+2	7.49e+2	8.42e+2
	Desv.	1.23e+0	3.38e-1	1.10e+2	8.95e+1	1.35e+2	3.48e-1	3.25e-1	2.12e-1	1.59e+2	1.21e+2	1.13e+2	5.10e+2	5.69e+2
	Min.	1.99e+0	1.29e+1	2.48e+1	3.39e+2	3.68e+2	9.00e+2	9.00e+2	9.00e+2	2.00e+2	9.44e+2	5.17e+2	2.00e+2	2.12e+2
	Med.	4.78e+0	1.35e+1	3.10e+2	4.77e+2	5.04e+2	9.00e+2	9.00e+2	9.00e+2	5.00e+2	1.09e+3	5.34e+2	2.00e+2	1.29e+3
1e5	Max.	6.93e+0	1.40e+1	3.38e+2	5.89e+2	7.97e+2	9.00e+2	9.00e+2	9.00e+2	5.00e+2	1.25e+3	5.34e+2	1.01e+3	1.43e+3
	Prom.	4.77e+0	1.35e+1	2.51e+2	4.62e+2	5.00e+2	9.00e+2	9.00e+2	9.00e+2	4.81e+2	1.10e+3	5.32e+2	2.65e+2	8.41e+2
	Desv.	1.12e+0	3.10e-1	1.20e+2	6.18e+1	9.41e+1	1.14e-9	1.03e-9	3.11e-10	6.36e+1	6.52e+1	5.23e+0	2.24e+2	5.69e+2
	Min.	1.96e+0	1.22e+1	2.02e-9	3.37e+2	2.97e+2	9.00e+2	9.00e+2	9.00e+2	2.00e+2	9.44e+2	5.17e+2	2.00e+2	2.12e+2
	Med.	4.08e+0	1.33e+1	3.07e+2	4.40e+2	4.44e+2	9.00e+2	9.00e+2	9.00e+2	5.00e+2	1.07e+3	5.34e+2	2.00e+2	1.29e+3
3e5	Max.	6.34e+0	1.40e+1	3.35e+2	5.47e+2	6.13e+2	9.00e+2	9.00e+2	9.00e+2	5.00e+2	1.25e+3	5.34e+2	2.00e+2	1.43e+3
	Prom.	4.28e+0	1.33e+1	2.39e+2	4.48e+2	4.58e+2	9.00e+2	9.00e+2	9.00e+2	4.81e+2	1.06e+3	5.32e+2	2.00e+2	8.41e+2
	Desv.	1.07e+0	3.93e-1	1.33e+2	5.80e+1	7.08e+1	2.32e-11	2.28e-11	2.33e-11	6.36e+1	6.48e+1	5.23e+0	1.37e-11	5.69e+2

Cuadro 5.8 – Los valores de error alcanzados en evaluaciones de la función = 1e3, 1e4, 1e5 para las funciones F13-F25 con dimensión $n = 30$.

El Cuadro 5.9 presenta los resultados obtenidos (valores promedio de error) por el método propuesto y los obtenidos por los algoritmos de comparación para $n = 10$ y 30. Los resultados de los algoritmos con los que comparamos se han extraído de los correspondientes trabajos publicados en la Sesión Especial. Los resultados resaltados por letras negritas son los que tienen mejor valor de error con respecto al resto de los métodos. Los resultados del método propuesto para la dimensión $n = 10$ se discuten a continuación.

- En las funciones F1, F2 y F3 se obtiene un excelente valor de error por debajo al impuesto 10^{-8}.
- Para la función F3 se obtiene un nivel de precisión mejor que el obtenido por los algoritmos L-SaDE y K-PCX.
- En la función F5 se obtienen una solución comparable con la obtenida por el algoritmo K-PCX.
- Para las funciones F6 y F7 se obtienen resultados superiores a los del K-PCX y muy comparables con los del algoritmo DE.
- Los resultados obtenidos en las funciones F8 y F24 fueron similares con los obtenidos por el resto de los métodos.
- En la función F9, el valor de error del método propuesto supera al obtenido por el método L-CMA-ES.
- En la función F10 se obtienen una solución comparable con la obtenida por los algoritmos DE y L-CMA-ES.
- En la función F11 se obtienen una solución mejor que las de los métodos L-SaDE, K-PCX y L-CMA-ES.
- Para la función F12 se obtienen una solución comparable con la obtenida por los algoritmos K-PCX y L-CMA-ES.

- Las funciones F13 y F14 exhibe resultados similares a los obtenidos por los métodos de comparación.

- La solución obtenida para la función F15 es superior al del resto de los algoritmos.

- El resultado para la función F16 es comparable con los resultados de los métodos DE, L-SaDE y L-CMA-ES.

- Los resultados para las funciones F17-F23 y F25 son muy comparables con respecto a las obtenidas por los métodos de comparación. Los resultados del K-PCX son inferiores en las funciones F21 y F23.

Los resultados obtenidos por el método propuesto para la dimensión $n = 30$ se discuten a continuación.

- En las funciones F1, F2 y F4 se obtienen un excelente valor de error por debajo al impuesto 10^{-8}.

- Para la función F3 y F6 se obtiene un nivel de precisión comparable con respecto al obtenido por los algoritmos DE y L-SaDE.

- En la función F5 y F7 se obtienen una solución comparable con la obtenida por el algoritmo K-PCX.

- La función F8 y F11 exhibe resultados similares a los obtenidos por los métodos de comparación.

- Para la función F9 se obtienen una solución comparable con la obtenida por los algoritmos DE y L-CMA-ES.

- Para la función F10 se obtienen un valor de error comparable con los que obtuvieron los algoritmos L-SaDE, DE y L-CMA-ES.

- Para la función F12 se obtienen un valor de error muy similar con los que obtuvieron los algoritmos K-PCX y DE.

- Los resultados para las funciones F13-F15 y F25 son muy comparables con respecto a los obtenidos por los métodos de comparación. Los resultados del K-PCX son inferiores en la función F13.

- El método de comparación L-SaDE ya no presenta resultados para las funciones F16-F25.

- El valor de error obtenido para la función F16 es comparable con el valor de error del método DE.

- Los valores de error para las funciones F17-F20 son muy comparables con respecto a los obtenidos por los métodos de comparación. Los resultados del algoritmo L-CMA-ES son inferiores en la función F17.

- Los resultados obtenidos por el método propuesto para las funciones multimodales híbridas F21-F24 superan a los resultados del resto de los métodos.

Conclusiones

Con base en los resultados que se obtuvieron mediante la experimentación del algoritmo RWM-ES sobre el conjunto de 25 funciones de prueba CEC 2005 y a las comparaciones con otros métodos, se observa que el comportamiento del método propuesto, en términos de la calidad de las soluciones, es muy similar al de los métodos DE, L-SaDE, K-PCX y L-CMA-ES. Además, es fácil ver que los resultados del método propuesto son robustos, ya que no empeora drásticamente cuando hay un crecimiento en la dimensión, siendo esto sin duda una gran ventaja con respecto al resto de los métodos de comparación. En general, se concluye que el método propuesto es una herramienta muy prometedora y útil

	$n = 10$						$n = 30$					
Función	RWM-ES	G-CMA-ES	DE	L-SaDE	K-PCX	L-CMA-ES	RWM-ES	G-CMA-ES	K-PCX	L-CMA-ES	DE	L-SaDE
F1	2.59e-13	5.20e-9	**0.00e+0**	**0.00e+0**	8.71e-9	5.14e-9	4.11e-9	5.42e-9	9.47e-9	5.28e-9	**0.00e+0**	**0.00e+0**
F2	5.79e-11	4.70e-9	**0.00e+0**	1.04e-13	9.68e-9	5.31e-9	8.68e-9	**6.22e-9**	9.85e-9	6.93e-9	3.33e-2	9.71e-8
F3	1.18e-5	5.60e-9	1.94e-6	1.67e-5	4.15e-1	**4.94e-9**	7.81e+4	5.55e-9	5.79e+1	**5.18e-9**	6.92e+5	5.05e+4
F4	8.73e-11	5.02e-9	**9.09e-15**	1.41e-5	7.94e-7	1.79e+6	**9.28e-9**	1.11e+4	1.11e+3	9.26e+7	1.52e+1	5.81e-6
F5	7.30e+1	6.58e-9	**0.00e+0**	1.23e-2	4.85e+1	6.57e-9	3.30e+3	8.62e-9	2.04e+3	**8.30e-9**	1.70e+2	7.88e+2
F6	1.59e-1	**4.87e-9**	1.59e-1	1.19e-8	4.78e-1	5.41e-9	9.81e+1	**5.90e-9**	1.75e+0	6.31e-9	2.51e+1	2.12e+1
F7	2.17e-1	**3.31e-9**	1.46e-1	1.99e-2	2.31e-1	4.91e-9	1.88e-2	**5.31e-9**	1.50e-2	6.48e-9	2.96e-3	8.27e-3
F8	**2.00e+1**	**2.00e+1**	2.04e+1	**2.00e+1**	**2.00e+1**	**2.00e+1**	**2.00e+1**	2.01e+1	2.01e+1	**2.00e+1**	2.10e+1	2.01e+1
F9	2.87e+0	2.39e-1	9.55e-1	**0.00e+0**	1.19e-1	4.49e+1	2.58e+1	9.38e-1	2.79e-1	2.91e+2	1.85e+1	**2.27e-15**
F10	6.05e+1	**7.96e-2**	1.25e+1	4.97e+0	2.39e-1	4.08e+1	3.93e+2	1.65e+0	**5.17e-1**	5.63e+2	9.69e+1	3.57e+1
F11	3.60e+0	9.34e-1	**8.47e-1**	4.89e+0	6.65e+0	3.65e+0	2.23e+1	**5.48e+0**	2.95e+1	1.52e+1	3.42e+1	2.65e+1
F12	2.91e+2	2.93e+1	3.17e+1	**4.50e-7**	1.49e+2	2.09e+2	1.94e+3	4.43e+4	1.68e+3	1.32e+4	2.75e+3	**8.73e+2**
F13	8.64e-1	6.96e-1	9.77e-1	**2.20e-1**	6.53e-1	4.94e-1	4.28e+0	2.49e+0	1.19e+1	2.32e+0	3.23e+0	**1.20e+0**
F14	3.74e+0	3.01e+0	3.45e+0	2.92e+0	**2.35e+0**	4.01e+0	1.33e+1	1.29e+1	1.38e+1	1.40e+1	1.34e+1	**1.23e+1**
F15	**1.57e+1**	2.28e+2	2.59e+2	3.20e+1	5.10e+2	2.11e+2	2.39e+2	**2.08e+2**	8.76e+2	2.16e+2	3.60e+2	3.27e+2
F16	2.07e+2	**9.13e+1**	1.13e+2	1.01e+2	9.59e+1	1.05e+2	4.48e+2	**3.50e+1**	7.15e+1	5.84e+1	2.12e+2	·
F17	2.38e+2	1.23e+2	1.15e+2	1.14e+2	**9.73e+1**	5.49e+2	4.58e+2	2.91e+2	**1.56e+2**	1.07e+3	2.37e+2	·
F18	9.34e+2	**3.32e+2**	4.00e+2	7.19e+2	7.52e+2	4.97e+2	9.00e+2	9.04e+2	**8.30e+2**	8.90e+2	9.04e+2	·
F19	9.71e+2	**3.26e+2**	4.20e+2	7.05e+2	7.51e+2	5.16e+2	9.00e+2	9.04e+2	**8.31e+2**	9.03e+2	9.04e+2	·
F20	9.40e+2	**3.00e+2**	4.60e+2	7.13e+2	8.13e+2	4.42e+2	9.00e+2	9.04e+2	**8.31e+2**	8.89e+2	9.04e+2	·
F21	4.41e+2	5.00e+2	4.92e+2	4.64e+2	1.05e+3	**4.04e+2**	**4.81e+2**	5.00e+2	8.59e+2	4.85e+2	5.00e+2	·
F22	8.19e+2	7.29e+2	7.18e+2	7.35e+2	**6.59e+2**	7.40e+2	**1.06e+2**	8.03e+2	1.56e+3	8.71e+2	8.97e+2	·
F23	5.69e+2	**5.59e+2**	5.72e+2	6.64e+2	1.06e+3	7.91e+2	**5.32e+2**	5.34e+2	8.66e+2	5.35e+2	5.34e+2	·
F24	**2.00e+2**	**2.00e+2**	**2.00e+2**	**2.00e+2**	4.06e+2	8.65e+2	**2.00e+2**	9.10e+2	2.13e+2	1.41e+3	**2.00e+2**	·
F25	5.92e+2	**3.74e+2**	9.23e+2	3.76e+2	4.06e+2	4.42e+2	8.41e+2	**2.11e+2**	2.13e+2	6.91e+2	7.30e+2	·

Cuadro 5.9 – Los valores de error promedio obtenidos sobre el conjunto de funciones de prueba CEC 2005.

para resolver problemas de óptimización global sin restricciones con diferentes estructuras del espacio de búsqueda.

5.2. Funciones de prueba CEC 2006

Para evaluar el desempeño de la estrategia evolutiva con caminatas aleatorias de Metrópolis (RWM-ES) en resolver problemas de optimización global con restricciones, se ha considerado 13 de las 24 funciones de prueba proporcionadas por Liang et al. [32] en la Sesión Especial "Constrained Real-parameter optimization" del IEEE CEC 2006. Este conjunto consiste en problemas con características tales como diferentes tipos de función de costo (lineal, cuadrática, no lineal), diferentes tipos de restricciones (lineal, no lineal, de igualdad y/o desigualdad) y diferente dimensionalidad. En la Cuadro 5.10 se muestra la relación de las funciones consideradas, incluyendo el valor óptimo de cada una de ellas y los rangos del espacio de búsqueda. Las expresiones de cada función de prueba se proporcionan en el Apéndice B.

Función	n	Óptimo global	Rangos de búsqueda
g01	13	-15.0000000000	$0 \leq x_i \leq 1$ (i = 1,...,9), $0 \leq x_i \leq 100$ (i = 10, 11, 12) y $0 \leq x_{13} \leq 1$
g02	20	-0.8036191042	$0 < x_i \leq 10$ (i = 1,...,n)
g03	10	-1.0005001000	$0 \leq x_i \leq 1$ (i = 1,...,n)
g04	5	-30665.5386717834	$78 \leq x_1 \leq 102$, $33 \leq x_2 \leq 45$ y $27 \leq x_i \leq 45$ (i = 3, 4, 5)
g05	4	5126.4967140071	$0 \leq x_1 \leq 1200$, $0 \leq x_2 \leq 1200$, $-0.55 \leq x_3 \leq 0.55$ y $-0.55 \leq x_4 \leq 0.55$
g06	2	-6961.8138755802	$13 \leq x_1 \leq 100$ y $0 \leq x_2 \leq 100$
g07	10	24.3062090681	$-10 \leq x_i \leq 10$ (i = 1,...,n)
g08	2	-0.0958250415	$0 \leq x_1 \leq 10$ y $0 \leq x_2 \leq 10$
g09	7	680.6300573745	$-10 \leq x_i \leq 10$ (i = 1,...,n)
g10	8	7049.2480205286	$100 \leq x_1 \leq 10000$, $1000 \leq x_i \leq 10000$ (i = 2, 3) y $10 \leq x_i \leq 1000$ (i = 4,...,n)
g11	2	0.7499000000	$-1 \leq x_1 \leq 1$ y $-1 \leq x_2 \leq 1$
g12	3	-1.0000000000	$0 \leq x_i \leq 10$ (i = 1, 2, 3) y p, q, r = 1, 2,...,9
g13	5	0.0539415140	$-2.3 \leq x_i \leq 2.3$ (i = 1, 2) y $-3.2 \leq x_i \leq 3.2$ (i = 3, 4, 5)

Cuadro 5.10 – Funciones de prueba del IEEE CEC 2006.

5.2.1. Especificaciones de la experimentación

El estudio experimental se realizó de acuerdo a los criterios de evaluación del conjunto de prueba CEC 2006 y son los siguentes. Cada función de prueba es repetida 25 veces con

inicialización uniforme sobre el espacio de búsqueda. El máximo número de evaluaciones de la función de costos permitido es de 500000. Una ejecución del método se termina antes de llegar al máximo número de evaluaciones si el valor de la función de error esta por debajo de 10^{-4}. La función de error es $|f(\mathbf{x}) - f(\mathbf{x}^*)|$, donde $\mathbf{x}$ es la solución propuesta factible y $\mathbf{x}^*$ es la solución óptima. Una solución es considerada factible si satisface a todas las restricciones (ver sección 2.6.1). Las restricciones de igualdad se transforman en desigualdades de la forma

$$\left|h_j(\mathbf{x})\right| - \varepsilon \leq 0, \qquad j = 1, ..., J$$

donde ε es un valor de tolerancia y se establece en 0.0001.

Por otro lado, los parámetros utilizados por el método propuestos fueron ajustados y elegidos como una configuración robusta para cada función de prueba. Los parámetros utilizados se resumen en el Cuadro 5.11. En la primer columna se indica el nombre de la función, en la segunda el número de muestras generadas en cada iteración, en la tercer columna se indica el valor de tolerancia para reiniciar el punto de partida del muestreador. Finalmente, la cuarta columna indica los valores de los parámetros de escala de la caminata aleatoria.

Función	m	ϵ_3	σ
g01	100	0.7	(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,1.0,1.0,1.0,1.0,0.1)
g02	100	0.7	(0.1,...,0.1)
g03	100	0.7	(0.1,...,0.1)
g04	100	0.7	(1.0,0.1,1.0,1.0,1.0)
g05	100	0.7	(1.0,1.0,0.1,0.1)
g06	100	0.7	(1.0,1.0)
g07	100	0.7	(1.0,...,1.0)
g08	100	0.7	(0.1, 0.1)
g09	100	0.7	(1.0,...,1.0)
g10	100	0.7	(5.0,5.0,5.0,1.0,1.0,1.0,1.0,1.0)
g11	100	0.7	(0.1, 0.1)
g12	100	0.7	(0.1,...,0.1)
g13	100	0.7	(0.1,...,0.1)

Cuadro 5.11 – Configuración de los parámetros utilizados por el método RWM-ES sobre el conjunto de prueba CEC 2006.

5.2.2. Resultados y discusiones

Los resultados obtenidos en la experimentación del RWM-ES sobre las 13 funciones de prueba se presenta en el Cuadro 5.12. En la primer columna se indica el nombre de la función, en la segunda columna se muestra el mínimo valor de error, en la tercera la mediana y en la cuarta columna se muestra el máximo valor de error que se obtiene sobre las ejecuciones. En las columnas 5 y 6, se muestran los valores de error promedio y su desviación estándar, respectivamente. En la columna 7 se indica la tasa de éxito para la cual se alcanzó el nivel de precisión (ejecución exitosa). En la columna 8 se indica la tasa de ejecuciones para las cuales se encuentran soluciones factibles. Por último, en la columna 9, se registra el número de evaluaciones de la función de costos necesarias para alcanzar el nivel de precisión, es decir, el número de evaluaciones en las cuales se logró obtener ejecuciones exitosas. La columna Desempeño es igual a promedio(evaluaciones para ejecuciones exitosas)*(total de ejecuciones)/(# de ejecuciones exitosas). Los puntos en la tabla indican que no se reportan soluciones factibles en dicha función.

Como se observa en el Cuadro 5.12, el método propuesto fue capaz de encontrar el óptimo global en seis funciones de prueba (g01, g04, g06, g08, g11 y g12) y se encontró soluciones muy cerca del óptimo global en tres funciones (g02, g03 y g09). El método fue capaz de llegar a la región factible en diez funciones (g01, g02, g03, g04, g06, g08, g09, g10, g11, g12) y para el resto de las funciones (g05, g07 y g13) no se encontraron soluciones factibles. Para los problemas g01 y g04, el método presenta dificultades para generar soluciones factibles, es decir, que presentan tasas de éxito menores al 50 %. Los problemas g06, g08, g11 y g12 presentan tasas de éxitos superiores al 50 % y se consideran como fáciles de encontras soluciones factibles. Por otro lado, en promedio, el método presenta soluciones de buena calidad con desviación estándar relativamente baja. Para los problemas donde se localizan los óptimos locales (g01, g04, g06, g08, g11 y g12), el método presenta un buen desempeño exhibiendo un bajo número de evaluaciones de la

función de costos.

Función	Min.	Med.	Max	Prom.	Desv.	Tasa de éxito	Tasa factible	Desempeño
g01	7.65e-09	2.00e+00	6.91e+00	2.32e+00	2.25e+00	0.28	1.00	9.56e+05
g02	3.89e-01	5.65e-01	6.27e-01	5.51e-01	5.32e-02	0.00	1.00	·
g03	9.65e-01	1.00e+00	1.00e+00	9.97e-01	9.77e-03	0.00	0.60	·
g04	1.07e-06	3.50e-02	3.79e+02	2.19e+01	7.77e+01	0.16	1.00	2.47e+06
g05	·	·	·	·	·	·	·	·
g06	1.60e-09	1.51e-07	9.90e-03	7.89e-04	2.40e-03	0.68	1.00	8.75e+04
g07	·	·	·	·	·	·	·	·
g08	8.20e-11	8.20e-11	8.20e-11	8.20e-11	2.09e-17	1.00	1.00	4.90e+03
g09	1.51e-01	1.21e+01	1.84e+02	2.92e+01	4.17e+01	0.00	0.96	·
g10	4.91e+02	1.41e+03	1.93e+03	1.28e+03	7.27e+02	0.00	0.12	·
g11	0.00e+00	6.66e-16	2.50e-01	6.00e-02	1.09e-01	0.76	1.00	2.82e+04
g12	0.00e+00	0.00e+00	5.65e-03	1.13e-03	2.30e-03	0.80	1.00	4.23e+04
g13	·	·	·	·	·	·	·	·

Cuadro 5.12 – Los valores de error obtenidos sobre el conjunto de funciones de prueba CEC 2006.

Basado en los resultados obtenidos, se clasifican las 13 funciones en cuatro categorías: muy difícil, difícil, promedio y fácil. La clasificación correspondiente se presenta en el Cuadro 5.13.

Categoría	Función
Muy difícil	g05, g07, g13
Difícil	g02, g03, g09, g10
Promedio	g01, g04
Fácil	g06, g08, g11, g12

Cuadro 5.13 – Clasificación del conjunto de funciones CEC 2006 en la fase experimental.

Conclusiones

Con base en los resultados que se obtuvieron mediante la experimentación del algoritmo RWM-ES sobre el conjunto de 13 funciones de prueba CEC 2006, se observa que el método puede tratar bien los problemas de optimización global restringidos sin hacer uso de técnicas especializadas para el manejo de restricciones como las funciones de penalización [33], las técnicas basadas en conceptos multiobjetivo [34], jerarquización estocástica [35], etcétera.

Los resultados obtenidos indican que el método RWM-ES es una herramienta muy prometedora para resolver problemas de optimización global sobre espacios restringidos.

Sin embargo se requiere de un mecanismo adicional para mejorar el desempeño de la propuesta, ya presenta obvios problemas en alcanzar el óptimo global en algunas de las funciones de prueba utilizada en este trabajo.

Capítulo 6

Conclusiones y trabajo futuro

En este trabajo doctoral se presentó un mecanismo de muestreo y un simple pero novedoso enfoque de búsqueda aleatoria evolutiva para resolver problemas de optimización global. Básicamente la metodología que se propone integra las técnicas de muestreo tipo Monte Carlo con las ideas esenciales de las estrategias evolutivas. El método de búsqueda propuesto genera un población mediante el mecanismo de muestreo propuesto. El mecanismo de muestreo fue construído de tal forma que la mayor parte del tiempo se explora las regiones más importantes del espacio, y por otro lado, da saltos largos con la finalidad de recorrer extensas regiones de baja probabilidad para intentar encontrar una región más prometedora. La población generada por los pasos del muestreo brinda información acerca de la estructura del espacio de búsqueda, de tal forma que las regiones prometedoras pueden ser localizadas de manera sencilla. Una vez que se localiza dichas regiones, un procedimiento de búsqueda local es acoplado con el fin de explotar dicha región, realizando movimientos finos para conseguir soluciones de mejor calidad. La clave principal del desempeño del método propuesto se obtiene a través de la auto-adaptación de sus parámetros, los cuales mantienen un balance adecuado entre la exploración y la explotación durante el proceso de búsqueda. Con lo anterior, el método propuesto asegura prevenir una convergencia hacia un óptimo local, incrementando la probabilidad

de encontrar un óptimo global.

El potencial del método fue probado sobre 25 funciones de prueba para optimización global sin restricciones y 13 funciones de prueba para optimización global con restricciones. Ambos conjuntos de prueba son propuestos en la comunidad científica como conjuntos estandar para evaluar técnicas evolutivas. Los resultados para el primer conjunto de prueba CEC 2005 fueron comparados contra los cinco algoritmos evolutivos de referencia en el estado del arte. Los resultados para el segundo conjunto de prueba CEC 2006 solo muestra la capacidad del método propuesto para tratar problemas de optimización con restricciones. Además, el estudio experimental provee un cierto conocimiento acerca del comportamiento del método propuesto sobre distintas clases de funciones y espacios de búsqueda. Para el conjunto de funciones CEC 2005 se presentaron funciones unimodales, multimodales, multimodales híbridas, y estas a su vez con características como ruido, desplazamiento, rotación, óptimo global en los límites, etcétera. En las funciones con restricciones CEC 2006 se presentaron problemas con características tales como diferentes tipos de función de costo (lineal, cuadrática, no lineal), diferentes tipos de restricciones (lineal, no lineal, de igualdad y/o desigualdad) y diferente dimensionalidad.

Se mostró empíricamente que el algoritmo de búsqueda propuesto en algunos casos obtiene soluciones óptimas, y en otros casos obtiene soluciones cercanas al óptimo global. Se muestra también, que en general el método brinda soluciones de buena calidad, exhibiendo un comportamiento robusto ante un amplio rango de problemas de optimización y cuando hay crecimiento en la dimensionalidad. El esfuerzo computacional requerido, en términos de evaluaciones de la función de costos, se muestra relativamente bajo. Por lo tanto, el estudio indica que el método propuesto es una herramienta muy prometedora y útil para resolver problemas irrestrictos y restringidos de óptimización global.

6.1. Trabajo futuro

Dado que el método propuesto es novedoso, existen bastantes mejoras que quedan como trabajo futuro. Algunas ideas que se desprenden como trabajo futuro se describen brevemente a continuación.

Dentro de los mecanismo internos del método se desprenden las siguientes ideas:

- Proponer otras reglas para el ajuste de los parámetros de escala del método.
- Incorporar variaciones de distribuciones de salto de cola pesada, por ejemplo, una distribución de salto de Weierstrass o la distribución de Levy, que son poco estudiadas en el campo de la optimización.
- Incorporar las distribuciones de salto multivariadas con el fin de capturar mejorar las correlaciones entre variables.
- Incorporar mecanismos para el manejo de restricciones.

Otras ideas como parte del trabajo futuro es introducir al método sobre otro enfoque de optimización y estos son los siguientes:

- Optimización multimodal: Dada la capacidad del método propuesto para encontrar los distintos modos de una función de interés, como se presentó en la sección 3.1.2, se puede extender la metodología a un enfoque como los métodos de búsqueda por nichos. Esto es con el fin de que se requiera encontrar los múltiples óptimos locales y/o globales.
- Optimización multiobjetivo: Es posible integrar las técnicas de optimización multiobjetivo para resolver problemas de este tipo.

Por otro lado, otra posible línea de investigación es la paralelización de ciertos mecanismo internos del método, los cuales se verían beneficiados por el tiempo de cómputo.

Apéndice A

Funciones CEC 2005

Las 24 imágenes que se presentan en este apéndice fueron tomadas del documento de la sesión especial CEC 2005 en [25]. Las funciones se enumeran de acuerdo a su posición, esto es de arriba hacia abajo y de izquierda a derecha. Por ejemplo, en la figura A.1, en la parte superior izquierda se encuentra la función F1 y a su derecha la función F2. En la parte inferior izquierda se encuentra la función F3 y a su derecha la función F4, y así sucesivamente.

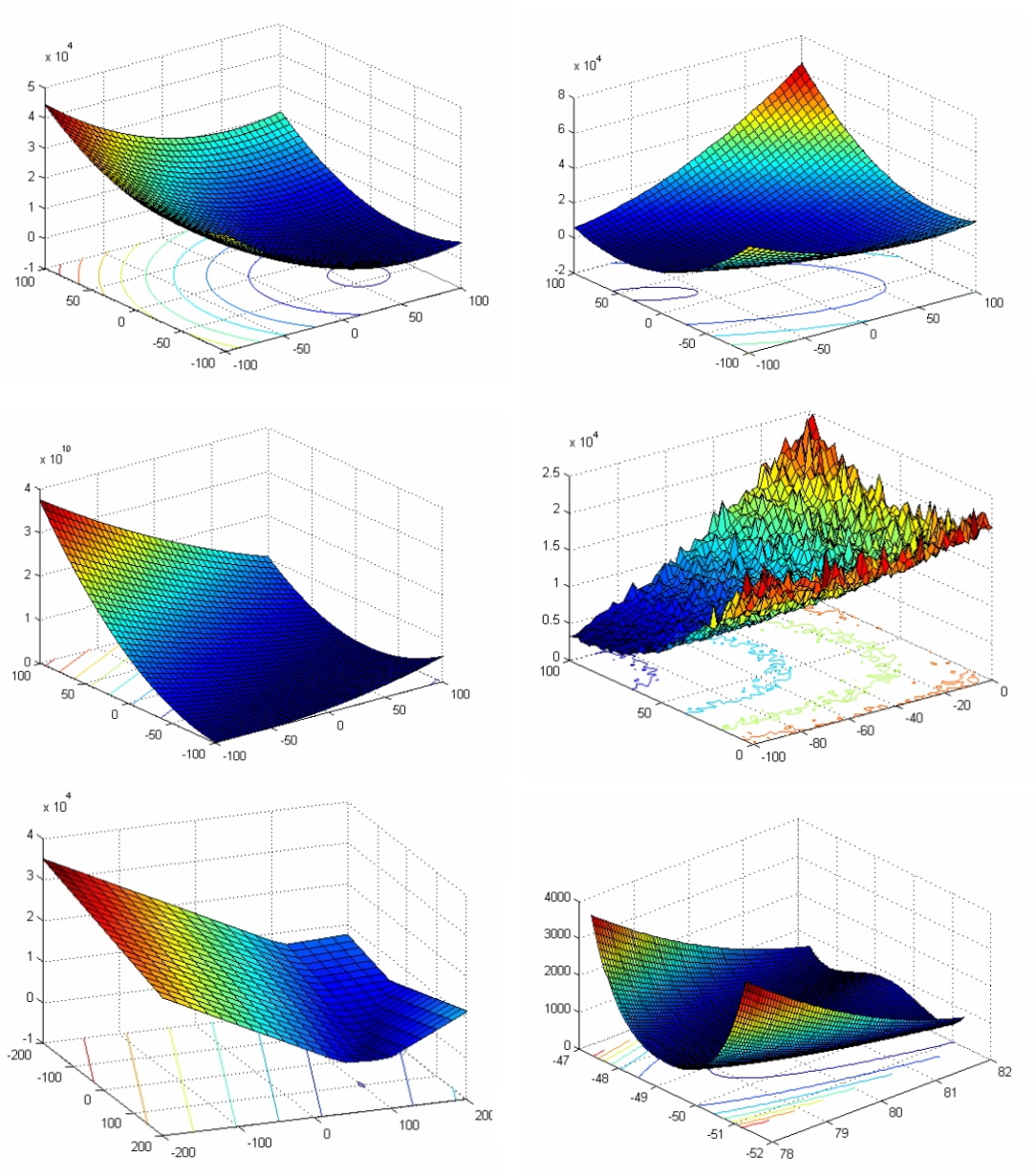


Figura A.1 – Funciones F1-F6

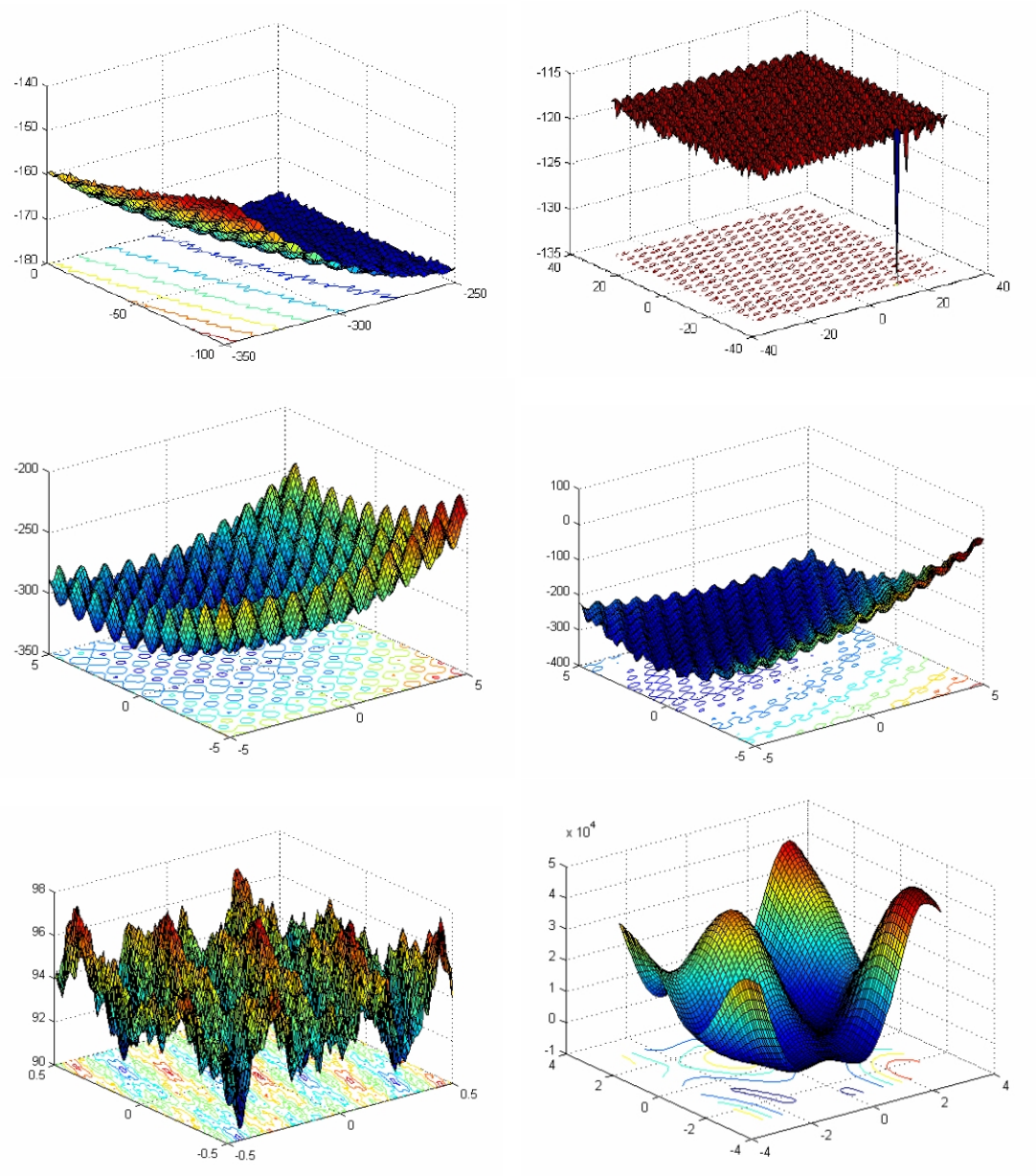


Figura A.2 – Funciones F7-F12

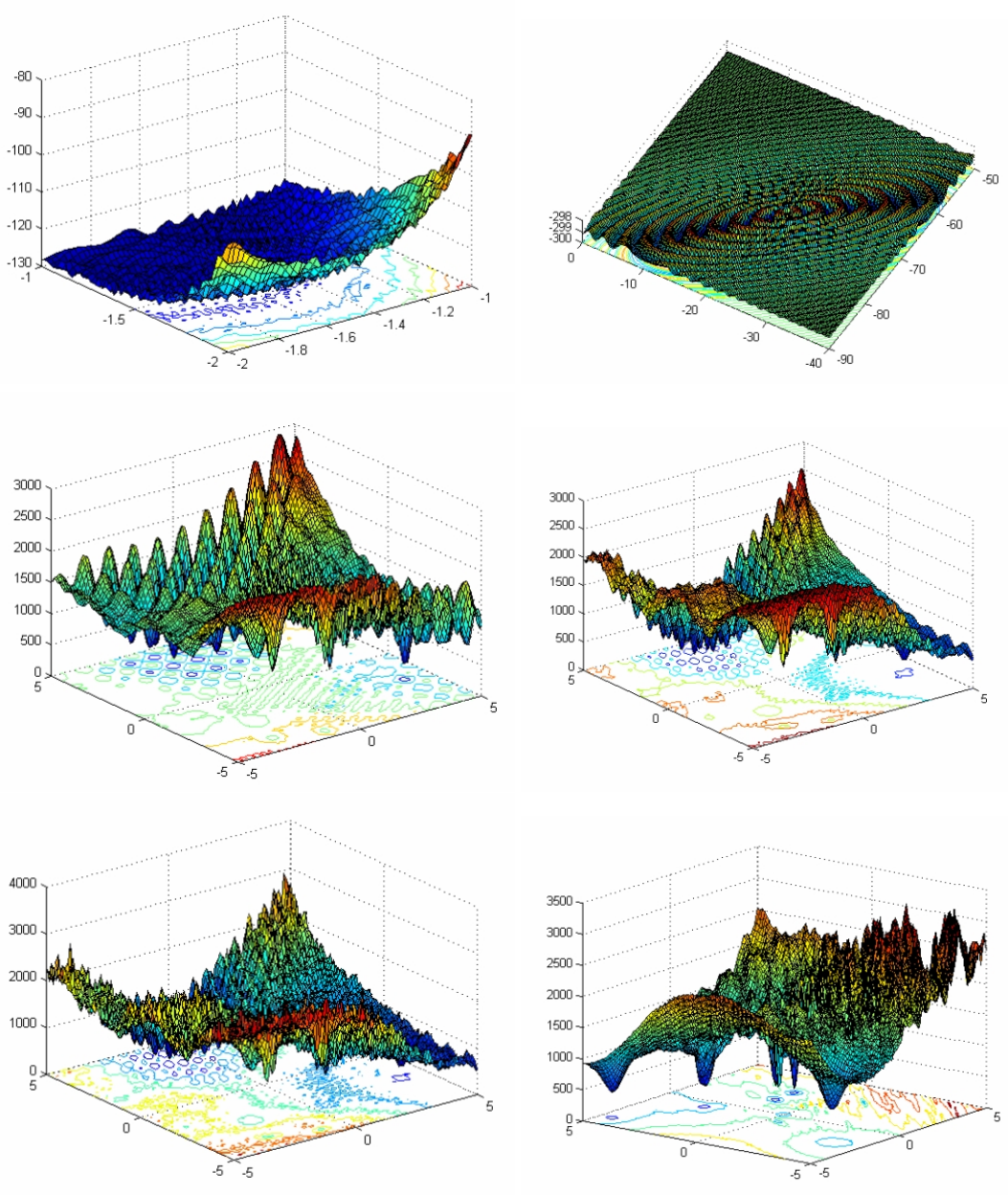


Figura A.3 – Funciones F13-F18

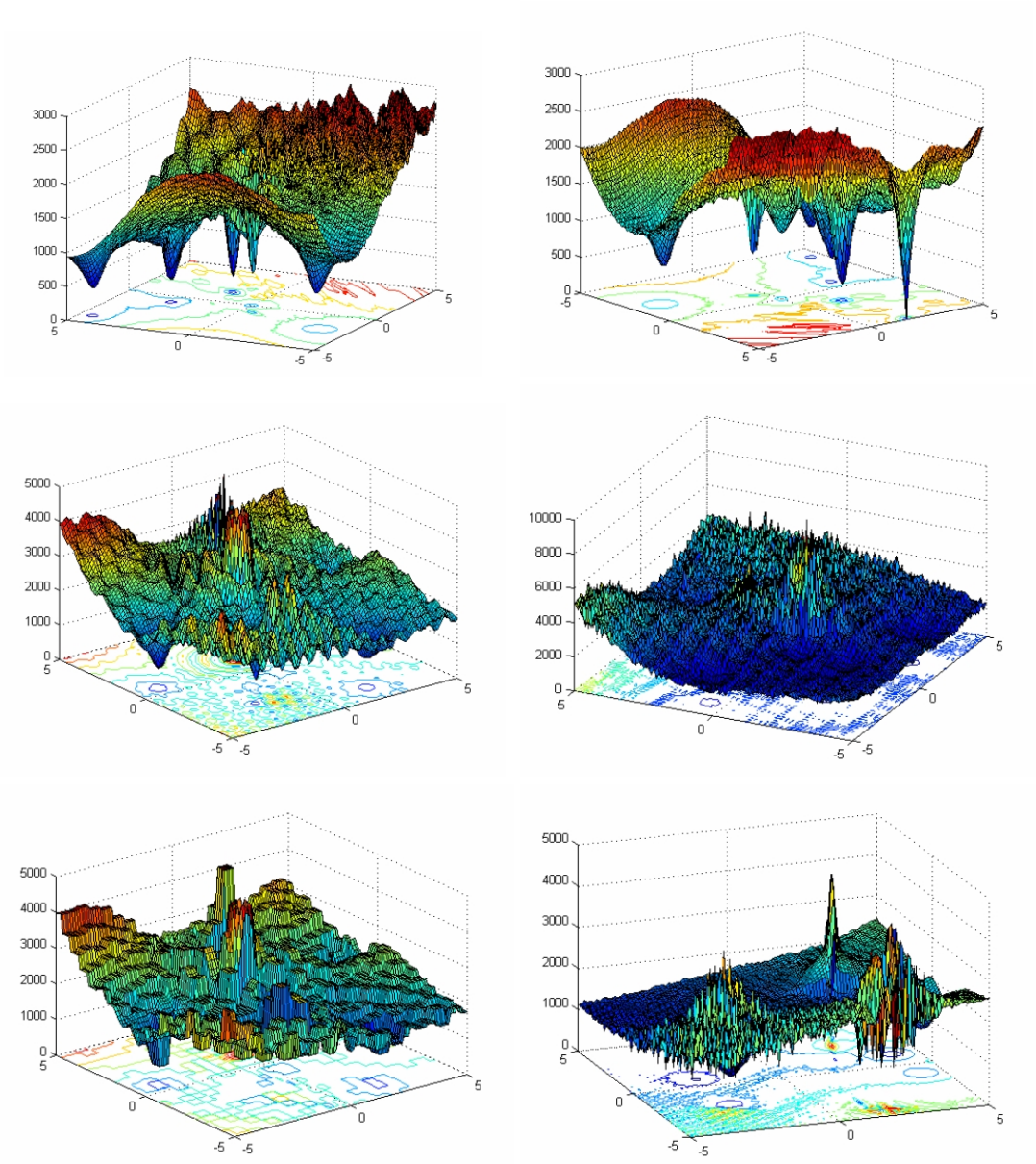


Figura A.4 – Funciones F19-F24

Apéndice B

Funciones CEC 2006

En este apéndice, se describen los 13 problemas de optimización con restricciones utilizados en el presente trabajo.

- **g01**

  $\min f(\mathbf{x}) = 5 \sum_{i=1}^{4} x_i - 5 \sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$

  sujeto a:

  $g_1(\mathbf{x}) = 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0$

  $g_2(\mathbf{x}) = 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0$

  $g_3(\mathbf{x}) = 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0$

  $g_4(\mathbf{x}) = -8x_1 + x_{10} \leq 0$

  $g_5(\mathbf{x}) = -8x_2 + x_{11} \leq 0$

  $g_6(\mathbf{x}) = -8x_3 + x_{12} \leq 0$

  $g_7(\mathbf{x}) = -2x_4 - x_5 + x_{10} \leq 0$

  $g_8(\mathbf{x}) = -2x_6 - x_7 + x_{11} \leq 0$

  $g_9(\mathbf{x}) = -2x_8 - x_9 + x_{12} \leq 0$

- **g02**

  $\min f(\mathbf{x}) = -\left| \frac{\sum_{i=1}^{n} \cos^4(x_i) - 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^2}} \right|$

  sujeto a:

$$g_1(\mathbf{x}) = 0.74 - \prod_{i=1}^{n} x_i \leq 0$$

$$g_2(\mathbf{x}) = \sum_{i=1}^{n} x_i - 7.5n \leq 0$$

- **g03**

  $$\text{mín } f(\mathbf{x}) = -(\sqrt{n})^n \prod_{i=1}^{n} x_i$$

  sujeto a:

  $$h_1(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 - 1 = 0$$

- **g04**

  $$\text{mín } f(\mathbf{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141$$

  sujeto a:

  $$g_1(\mathbf{x}) = 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 - 92 \leq 0$$

  $$g_2(\mathbf{x}) = -85.334407 - 0.0056858x_2x_5 - 0.0006262x_1x_4 + 0.0022053x_3x_5 \leq 0$$

  $$g_3(\mathbf{x}) = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 - 110 \leq 0$$

  $$g_4(\mathbf{x}) = -80.51249 - 0.0071317x_2x_5 - 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \leq 0$$

  $$g_5(\mathbf{x}) = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25 \leq 0$$

  $$g_6(\mathbf{x}) = -9.300961 - 0.0047026x_3x_5 - 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \leq 0$$

- **g05**

  $$\text{mín } f(\mathbf{x}) = 3x_1 + 0.000001x_1^3 + 2x_2 + (0.000002/3)x_2^3$$

  sujeto a:

  $$g_1(\mathbf{x}) = -x_4 + x_3 - 0.55 \leq 0$$

  $$g_2(\mathbf{x}) = -x_3 + x_4 - 0.55 \leq 0$$

  $$h_3(\mathbf{x}) = 1000\sin(-x_3 - 0.25) + 1000\sin(-x_4 - 0.25) + 894.8 - x_1 = 0$$

  $$h_4(\mathbf{x}) = 1000\sin(x_3 - 0.25) + 1000\sin(x_3 - x_4 - 0.25) + 894.8 - x_2 = 0$$

  $$h_5(\mathbf{x}) = 1000\sin(x_4 - 0.25) + 1000\sin(x_4 - x_3 - 0.25) + 1294.8 = 0$$

- **g06**

  $$\text{mín } f(\mathbf{x}) = (x_1 - 10)^3 + (x_2 - 20)^3$$

sujeto a:

$g_1(\mathbf{x}) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0$

$g_2(\mathbf{x}) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0$

- **g07**

$\text{mín } f(\mathbf{x}) = x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2$
$+2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45$

sujeto a:

$g_1(\mathbf{x}) = -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 0$

$g_2(\mathbf{x}) = 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0$

$g_3(\mathbf{x}) = -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0$

$g_4(\mathbf{x}) = 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0$

$g_5(\mathbf{x}) = 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0$

$g_6(\mathbf{x}) = x_1^2 + 2(x_2 - 2)^2 - 2x_1 x_2 + 14x_5 - 6x_6 \leq 0$

$g_7(\mathbf{x}) = 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \leq 0$

$g_8(\mathbf{x}) = -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0$

- **g08**

$\text{mín } f(\mathbf{x}) = \frac{\sin(2\pi x_1)^3 \sin(2\pi x_2)}{x_1^3(x_1 + x_2)}$

sujeto a:

$g_1(\mathbf{x}) = x_1^2 - x_2 + 1 \leq 0$

$g_2(\mathbf{x}) = 1 - x_1 + (x_2 - 4)^2 \leq 0$

- **g09**

$\text{mín } f(\mathbf{x}) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6 x_7 - 10x_6 - 8x_7$

sujeto a:

$g_1(\mathbf{x}) = -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 \leq 0$

$g_2(\mathbf{x}) = -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \leq 0$

$g_3(\mathbf{x}) = -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \leq 0$

$g_4(\mathbf{x}) = 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0$

- **g10**

  mín $f(\mathbf{x}) = x_1 + x_2 + x_3$

  sujeto a:

  $g_1(\mathbf{x}) = -1 + 0.0025(x_4 + x_6) \leq 0$

  $g_2(\mathbf{x}) = -1 + 0.0025(x_5 + x_7 - x_4) \leq 0$

  $g_3(\mathbf{x}) = -1 + 0.01(x_8 - x_5) \leq 0$

  $g_4(\mathbf{x}) = -x_1x_6 + 833.33252x_4 + 100x_1 - 83333.333 \leq 0$

  $g_5(\mathbf{x}) = -x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4 \leq 0$

  $g_6(\mathbf{x}) = -x_3x_8 + 1250000 + x_3x_5 - 2500x_5 \leq 0$

- **g11**

  mín $f(\mathbf{x}) = x_1^2 + (x_2 - 1)^2$

  sujeto a:

  $h_1(\mathbf{x}) = x_2 - x_1^2 = 0$

- **g12**

  mín $f(\mathbf{x}) = -(100 - (x_1 - 5)^2 - (x_2 - 5)^2 - (x_3 - 5)^2)/100$

  sujeto a:

  $g_1(\mathbf{x}) = (x1 - p)^2 + (x_2 - q)^2 + (x_3 - r)^2 - 0.0625 \leq 0$

- **g13**

  mín $f(\mathbf{x}) = e^{x_1x_2x_3x_4x_5}$

  sujeto a:

  $h_1(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0$

  $h_2(\mathbf{x}) = x_2x_3 - 5x_4x_5 = 0$

  $h_3(\mathbf{x}) = x_1^3 + x_2^3 + 1 = 0$

Bibliografía

[1] Nemhauser G. L., Rinnooy Kan A. H. G., Todd M. J., 1989. *Handbooks in Operations Research and Management Science*, Vol. 1, Optimization (North-Holland, Amsterdam).

[2] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. Teller, E., 1953. Equations of state calculations by fast computing machines, *Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087-1092.

[3] Berrones, A., 2008. Stationary probability density of stochastic search processes in global optimization, *J. Stat. Mech.: Theory Exp.*, p. P01013.

[4] Berrones, A., 2010. Bayesian Inference Based on Stationary Fokker–Planck Sampling, *Neural Computation*, vol. 22, no. 6, pp. 1573-1596.

[5] Canty, A., 1999. Hypothesis tests of convergence in markov chain monte carlo, *J. of Computational and Graphical Statistics*, vol. 8, no. 1, pp. 93-108.

[6] Roberts, G. O., Polson, N. G., 1994. On the Geometric Convergence of the Gibbs Sampler, *J. R. Statist. Soc. B*, vol. 56, no. 2, pp. 377-384.

[7] Wolpert, D. H., Macready, W. G., 1995. *No Free Lunch Theorems for Search*, Technical Report SFI-TR-95-02-010, Santa Fe Institute.

[8] Wolpert, D. H., Macready, W. G., 1997. No free luch theorems for optimization, *IEEE Transaction on Evolutionary Computation*, vol. 1, no. 1, pp. 67-82.

[9] Ho, Y. C., Pepyne, D. L., 2002. Simple Explanation of the No-Free-Lunch Theorem and Its Implications, *Journal of Optimization Theory and Applications* vol. 115, pp. 549-570.

[10] Metropolis, N., Ulam, S., 1949. The Monte Carlo method. *Journal of the American Statistical Association*, vol. 44, no. 247, pp. 335–341.

[11] Neal, R. M., 1993. *Probabilistic inference using Markov chain Monte Carlo methods*. Technical Report CRG-TR-93-1, Dept. of Computer Science, Univ. of Toronto.

[12] Gilks, W. R., Richardson, S., Spiegelhalter, D. J., 1996. *Markov Chain Monte Carlo in Practice*. Chapman and Hall.

[13] Tanner, M. A., 1996. *Tools for Statistical Inference: Methods for the Exploration of Posterior Distributions and Likelihood Functions*. Springer Series in Statistics. Springer, 3rd edition.

[14] Hastings W. K., 1970. *Monte Carlo sampling methods using Markov chains and their applications*. Biometrika, vol. 57, pp. 97-109.

[15] Roberts, G.O., Smith, A. F. M., 1994. Simple conditions for the convergence of the Gibbs sampler and Hastings-Metropolis algorithms. *Stochastic Process. Appl.*, vol. 49, pp. 207-216.

[16] Geman S., Geman D., 1984. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721–741.

[17] Goldberg D. E., 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley.

[18] Rechenberg I., 1965. Cybernetic solution path of an experimental problem. *Royal Aircraft Establishment.* Library Translation No. 1122, Farnborough, Hants, UK.

[19] Schwefel H-P., 1981. *Numerical Optimization of Computer Models.* John Wiley & Sons, Great Britain.

[20] Rechenberg I., 1973. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, Stuttgart.

[21] Beyer H. G., Schwefel H. P., 2002. Evolution strategies - A comprehensive introduction. *Natural Computing*, vol. 1, pp. 3-52.

[22] Neal, P., Roberts, G., 2011. Optimal Scaling of Random Walk Metropolis Algorithms with Non-Gaussian Proposals. *Methodology & Computing in Applied Probability*, Vol. 13, pp. 583-601.

[23] Nelder J. A., Mead R., 1965. A simplex method for function minimization. *The Computer Journal*, vol. 7, pp. 308-313.

[24] Parzen E., 1962. On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, vol. 33, pp. 1065-1076.

[25] Suganthan P. N., Hansen N., Liang J. J., Deb K., Chen Y.-P., Auger A., Tiwari S., 2005. Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization. Nanyang Technological University, Singapore, Tech. Rep.

[26] Hansen N., 2006. Compilation of Results on the 2005 CEC Benchmark Function Set. Computational Laboratory (CoLab), Institute of Computation Science, ETH Zurich, Tech. Rep.

[27] Auger A., Hansen N., 2005. A Restart CMA Evolution Strategy With Increasing Population Size. *In Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005*, vol. 3, pp. 1769-1776.

[28] Rönkkönen J., Kukkonen S., Price K. V., 2005. Real-Parameter Optimization with Differential Evolution. *In Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005*, vol.3, pp. 506-513.

[29] Qin A. K., Suganthan P. N., 2005. Self-adaptive Differential Evolution Algorithm for Numerical Optimization. *In Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005*, vol.3, pp. 1785-1791.

[30] Sinha A., Tiwari S., Deb K., 2005. A Population-Based, Steady-State Procedure for Real-Parameter Optimization. *In Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005*, vol.3, pp. 514-521.

[31] Auger A., Hansen N., 2005. Performance Evaluation of an Advanced Local Search Evolutionary Algorithm. *In Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005*, vol.3, pp. 1777-1784.

[32] Liang J. J., Runarsson T. P., Mezura-montes E., Clerc M., Suganthan P.N., Coello C. A. C., Deb K., 2006. Problem Definitions and Evaluation Criteria for the CEC 2006 Special Session on Constrained Real-Paremeter Optimization. Technical Report.

[33] Alice E. S., David W. C., 1997. Constraint Handling Techniques–Penalty Functions, in Thomas Bäck, David B. Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, chapter C 5.2. Oxford University Press and Institute of Physics Publishing.

[34] Coello C. C. A., 2000. Constraint-handling using an evolutionary multiobjective

optimization technique. *Civil Engineering and Environmental Systems*, vol. 17, pp. 319–346.

[35] Runarsson T. P., Yao X., 2000. Stochastic Ranking for Constrained Evolutionary Optimization. *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 3, pp. 284–294.

Biografía

Nací el 24 de Abril de 1984, en la ciudad de Los Mochis, Sinaloa, México, siendo el hijo más pequeño del Sr. René Velasco Flores y la Sra. Rosa Martha Álvarez Lopez. Estudié Ingeniería Industrial con especialidad en Sistemas de Calidad en la Universidad de Occidente, campus Los Mochis, en los años 2002 al 2006. Obtuve el grado de Maestro en Ciencias en Ingeniería de Sistemas en el 2009 con la tesis titulada «Una heurística basada en búsqueda estocástica para problemas de optimización global», por la Universidad Autónoma de Nuevo León. Después, en el 2009, inicié mis estudios de doctorado en el mismo Posgrado en Ingeniería de Sistemas de la UANL, con el tema de tesis «Estrategia evolutiva con caminatas aleatorias de Metrópolis para problemas de optimización global», bajo la supervisión del Dr. Arturo Berrones. Este trabajo de tesis, representa el fin de mis estudios, fungiendo como opción para conseguir el título de Doctor en Ingeniería con especialidad en Ingeniería de Sistemas en el 2013. Actualmente, soy colaborador de diversos proyectos de investigación, y me despeño como profesor en las áreas de las Tecnologías de la Información.