ÍNDICE

PRÓLOGO

Las aplicaciones de la optimización son numerosas. Cada proceso extraído de la vida real es factible de ser optimizado. No existe organización alguna que no involucre, dentro de sus actividades, la solución de problemas de optimización. En términos generales, muchas aplicaciones interesantes en las ciencias y la industria, pueden ser formuladas como procesos de optimización.

La gran mayoría de los problemas de optimización con implicaciones prácticas en ciencias, ingeniería, economía y negocios, son muy complejos y difíciles de resolver. Tales problemas no pueden ser resueltos de una manera exacta usando métodos de optimización clásicos. Bajo estas circunstancias, los esquemas metaheurísticos o bio-inspirados se han consolidado como una solución alternativa.

Los algoritmos metaheurísticos son considerados herramientas genéricas de optimización, que pueden resolver problemas muy complejos caracterizados por contar un espacio de búsqueda demasiado grande. Los métodos de cómputo metaheurístico reducen el tamaño efectivo del espacio de búsqueda, mediante el uso de estrategias efectivas de búsqueda. En general, dichos métodos permiten resolver los problemas de una manera más rápida y robusta. En comparación con otros algoritmos heurísticos, las técnicas de cómputo metaheurísticas son más simples de diseñar e implementar.

Los métodos metaheurísticos y bio-inspirados representan un área de la optimización en las ciencias de la computación y las matemáticas aplicadas. En los últimos 10 años, estos métodos han sido testigos del desarrollo de numerosos

enfoques, que permiten la intersección de diferentes disciplinas que incluyen la inteligencia artificial, la biología, estudios sociales y las matemáticas. La mayoría de los métodos metaheurísticos usan como inspiración fenómenos biológicos o sociales, los cuales en cierto nivel de abstracción pueden ser considerados como modelos de optimización.

Recientemente, los algoritmos bio-inspirados se han popularizado tanto en la academia como en la industria. Un indicador de esta situación es el gran número de revistas, sesiones y conferencias especializadas en esta área. En la práctica, estos métodos han despertado gran interés, ya que han demostrado ser herramientas eficientes para la solución de un amplio rango de problemas en dominios tales como la logística, bio-informática, diseño estructural, data mining, finanzas, etc.

Este libro presenta enfoques para la solución de problemas de optimización. En particular son discutidos los esquemas de computación metaheurística. Aunque el libro incluye diferentes aspectos matemáticos de la teoría de optimización y de los operadores metaheurísticos, no debe ser considerado un libro meramente teórico. Su contenido es más relacionado a los aspectos prácticos de estos métodos y a su aplicación como herramientas de ciencias de la computación. El principal objetivo de este libro es brindar una visión unificada de los métodos metaheurísticos, de tal forma que se presentan los principios de diseño fundamentales, así como los operadores de los enfoques que son considerados esenciales.

Los esquemas metaheurísticos tratados en este libro son examinados con el objetivo de que sean implementados y utilizados. Actualmente existen varios libros que tratan los temas de los enfoques metaheurísticos y bio-inspirados. Este libro, a diferencia de todos los demás, ha sido escrito desde una perspectiva pedagógica. El material de este libro es tratado de tal forma que el lector es asistido para obtener una idea clara, pero al mismo tiempo rigurosa de cada método metaheurístico. Muchos libros discuten una gran variedad métodos metaheurísticos y bio-inspirados como un recetario sin ningún soporte teórico. Por otro lado, otros libros consideran los enfoques metaheurísticos en forma solamente teórica siendo no totalmente accesibles a estudiantes o profesionales de otras áreas ajenos a las ciencias de la computación como lo son la administración, economía, etc. Por el contrario, este libro intenta hacer un compromiso entre ambos objetivos, presentando a cada enfoque con el contenido teórico necesario, así como de la información para su fácil implementación.

El libro provee los conceptos necesarios que habilitan al lector a implementar y modificar los métodos de cómputo metaheurísticos, para obtener los desempeños proyectados en las necesidades específicas de cada problema. Para ello, el libro

contiene numerosos ejemplos de problemas y soluciones que demuestran la potencia de estos métodos. Su contenido provee en cada enfoque metaheurístico o bio-inspirado ejemplos simples que son fáciles de entender y que preparan de una manera intuitiva al lector, lo que podría producirse como resultado. Esto contrasta con la mayoría de los libros en esta área, que presentan ejemplos de tal complejidad, que resulta difícil el entendimiento del propio problema, sin contar con la comprensión de su solución.

El libro está basado enteramente en MATLAB, el cual es considerado hoy en día como un estándar en la programación científica e industrial. MATLAB contiene ya dentro de sus funciones poderosos métodos numéricos que pueden ser adaptados a aplicaciones particulares. Bajo estas condiciones, el usuario puede estar más concentrado en la estructura de su aplicación que en la programación misma. Presentar el código de los programas como ayuda al lector es uno de los principales activos de un libro en el área de las ciencias de la computación. Hacerlos disponibles en una página web, puede derivar en su desaparición con el transcurso de los años. Así, una de las principales características de este libro es que todos los enfoques presentados han sido programados en MATLAB y su código se presenta impreso en el mismo. Es importante recalcar que los códigos presentes en el libro no pretenden ser eficientes o competitivos desde la perspectiva de programación. En lugar de esto, se ha decidido presentar el código de la implementación de cada método de una forma pedagógica de tal manera que el lector comprenda cómo el método se comporta y ejecuta.

Una característica interesante del libro es que incluye algoritmos recientemente desarrollados que no están disponibles en otros textos. Estos tópicos incluyen el algoritmo de seno-coseno, el algoritmo de la polilla, de los lobos grises y otros. Estos métodos han sido recientemente considerados como métodos metaheurísticos populares, con una amplia aceptación en la comunidad de optimización.

La notación usada en el libro asume que el lector está familiarizado con conceptos básicos de álgebra, geometría, teoría de conjuntos y cálculo. En términos generales, un lector no aprovechará esta obra si no tiene la intención de implementar computacionalmente los métodos presentados. Bajo estas condiciones, es considerado como prerrequisito el contar con conocimientos básicos de programación.

Por su enfoque y estructura, el libro es adecuado para estudiantes y profesionales en el área de ciencias de la computación, inteligencia artificial, investigación de operaciones, matemáticas aplicadas y algunas otras disciplinas. De igual manera, muchos ingenieros que trabajan en la industria pueden encontrar interesante el contenido de esta obra. En este caso, las sencillas exposiciones y el código provisto

pueden servir de ayuda para la rápida solución a problemas de optimización que normalmente surgen en varios nichos y proyectos industriales.

Escribir un libro de métodos de cómputo metaheurístico parece intrascendente. Existen ya varios libros de texto y de consulta dirigidos a lectores con diferentes niveles de conocimiento en la materia. Sin embargo, de las decenas de libros que existen, estos se reducen a prácticamente muy pocos si se restringen las opciones a los escritos en español. Bajo estas circunstancias, cuando se concibió el proyecto de escritura, se decidió que era necesario tener algo que decir sobre el orden, la profundidad y la manera de exponer los conceptos de cómputo metaheurísticos en nuestro idioma.

Nuestra premisa original fue que los métodos de cómputo metaheurístico pueden exponerse de manera comprensible para lectores con poco entrenamiento matemático. En consecuencia, intentamos escribir un libro cuyo contenido fuese no solo entendible, sino aplicable por cualquier estudiante de licenciatura. Aunque algunos conceptos pueden ser complejos para su comprensión cabal, tratamos de exponerlos con claridad y sin pretender disimular su dificultad implícita. Sin sacrificar las necesidades inmediatas del lector, subrayamos la importancia de que se comprendan los métodos descritos, bajo el supuesto de que a veces es preferible no usar estos métodos de optimización que usarlos mal.

En el largo proceso de escribir este libro, nuestra perspectiva ha variado. Planeado para un curso completo, el material que presentamos puede darse en un semestre. Su contenido consta de 14 capítulos; los detalles en el tratamiento de cada uno de ellos se describen a continuación:

En el capítulo 1 se introduce el concepto de optimización. Primero, una vez formulado de manera genérica el problema de optimización, se clasifican los métodos utilizados para su solución. Después, se hace un breve repaso de las técnicas clásicas basadas en gradiente más populares. El capítulo expone las principales características de los algoritmos evolutivos además de introducir el dilema de la exploración y explotación. De manera importante, se analiza la aceptación y selección probabilística, dos de las principales operaciones usadas por la mayoría de los métodos metaheurísticos. El capítulo termina, exponiendo dos de los primeros métodos metaheurísticos, que han servido de base para la creación de nuevos algoritmos, la idea con esta exposición es introducir a los conceptos de los métodos metaheurísticos, mediante la implementación de técnicas relativamente fáciles de implementar.

En el capítulo 2 se analizan los algoritmos genéticos. En él se mencionan algunas de las diferentes variantes que existen hasta el momento, así como algunos puntos

importantes a considerar al implementarse como: a) los distintos tipos de cruce, b) se explica el algoritmo de selección de ruleta, y c) se establecen sus principales operadores; se termina con un pequeño código de ejemplo que resuelve una función con cierto grado de complejidad en dos dimensiones.

En el capítulo 3 se realiza un tratamiento del Algoritmo Evolución Diferencial (DE por sus siglas en inglés, Differential Evolution). En este capítulo, se discuten los parámetros del algoritmo Evolución Diferencial, así como los pasos necesarios para implementarlo.

En el capítulo 4 se presentan las Estrategias Evolutivas (ES), las cuales pertenecen al primer bloque de algoritmos inspirados en la evolución. En él se explica de manera detallada cada uno de los operadores, poniendo especial énfasis en la mutación. La parte de las explicaciones se acompaña con segmentos de código en MatLAB.

En el capítulo 5 se describe el algoritmo de optimización por enjambre de partículas (PSO por sus siglas en inglés, Particle Swarm Optimization), uno de los algoritmos de cómputo metaheurísticos más usados, debido a la sencillez de sus operadores. En el tratamiento de este capítulo se detallan las bases del PSO; además de analizar la teoría, se representan un ejemplo práctico y su codificación.

En el capítulo 6 se presenta el algoritmo colonia artificial de abejas (ABC por sus siglas en inglés Artificial Bee Colony). En él se discuten los parámetros del algoritmo, así como la codificación necesaria para implementarlo. De manera adicional, se presenta una reseña acerca del algoritmo ABC aplicado al área de Procesamiento de Imágenes.

En el capítulo 7 se describe el algoritmo de colonia de hormigas (Ant Colony Optimization, ACO). Este esquema es una estrategia probabilística de búsqueda para resolver problemas de optimización. Esta técnica es especialmente efectiva para encontrar trayectorias óptimas en formulaciones que modelan estos problemas. Este enfoque se basa en la manera en que las hormigas refuerzan la caracterización del mejor camino que conduce a la comida por medio de la feromona.

En el capítulo 8 se analiza el método de búsqueda del cuco (Cuckoo Search). Este método de optimización fue introducido originalmente por Xin-She Yang y Suash Deb en 2009. El esquema de búsqueda del cuco se inspira en el comportamiento reproductivo de los pájaros cucos que colocan sus huevos en los nidos de otras especies. Los huevos simbolizan potenciales soluciones a los problemas de optimización. De esta manera, los mejores huevos pasan de generación en generación mientras que los peores son descartados para luego ser reemplazados.

En el capítulo 9 se presenta el algoritmo de búsqueda de cuervos (Crow Search Algorithm). Este esquema es un método metaheurístico que emula el comportamiento de los cuervos en sus procesos de ocultar y recuperar comida. Utilizando estos procesos, el método define operadores especiales para explorar eficientemente el espacio de búsqueda para encontrar una solución competitiva para el problema de optimización.

En el capítulo 10 se describe el método de las polillas (Moth-Flame Optimization, MFO). Esta técnica de optimización emula el método de navegación de las polillas llamado orientación transversa. Pese a su popularidad, este esquema es un método reciente que no es tratado prácticamente en ningún texto de métodos metaheurísticos.

En el capítulo 11 se analiza el método de los lobos grises (Grey Wolf Optimizer, GWO). El algoritmo de los lobos grises es un esquema inspirado en la forma en cómo los lobos grises estructuran su jerarquía y en consecuencia elaboran estrategias de caza.

En el capítulo 12 se describe el algoritmo de búsqueda gravitacional (Gravitational Search algorithm, GSA). Este esquema se basa en las leyes de Newton de movimiento y gravitación. En el método, la calidad de las soluciones es representado por su masa. De esta manera, las soluciones son atraídas y repelidas de acuerdo a esta información y a la distancia que mantienen entre ellas.

En el capítulo 13 se analiza el método de los estados de la materia (States of Matter Search, SMS). En este método, el proceso de optimización es considerado como un esquema en el que un material experimenta los tres estados de la materia. De esta manera, empezando por el estado gaseoso, el material también experimenta el estado líquido y sólido. En todos estos estados, las moléculas (que simbolizan las soluciones) presentan un patrón diferente de movimiento. Empezando por el estado gaseoso, las soluciones se desplazan en todo el espacio de búsqueda (total exploración). Después en el estado líquido, las soluciones se desplazan localmente en áreas promisorias del espacio de búsqueda (equilibrio de exploración-explotación). Finalmente, en el estado sólido las soluciones solo vibran en sus posiciones originales (total explotación).

Finalmente, en el capítulo 14 se describe el método de búsqueda del seno-coseno (Sine-Cosine Algorithm, SCA). Esta técnica utiliza como patrón de búsqueda la forma en cómo las funciones del seno y coseno evolucionan en el tiempo. Este método es muy simple de implementar y no presenta parámetros de configuración.

Durante más de diez años hemos ensayado múltiples maneras de exponer este material a auditorios disímiles. En el camino se ha contado con la invaluable tolerancia de nuestros alumnos, principalmente del CUCEI en la Universidad de Guadalajara en México. Se agradece de manera especial a nuestros compañeros profesores del Centro Universitario de Ciencias Exactas e Ingenierías. Tantas colaboraciones, ayudas y discusiones con colegas, merecerían un capítulo adicional. A todos, los mencionados y en especial a los omitidos, nuestro testimonio de gratitud.

Erik Cuevas
Fernando A. Fausto
Jorge Gálvez
Alma Rodríguez

Guadalajara, México

1 OPTIMIZACIÓN

1.1 INTRODUCCIÓN

Optimización se ha convertido en una parte importante en todas las disciplinas. Una razón de esta inclusión es la motivación de producir productos o servicios de calidad a precios competitivos. En general, optimización representa el proceso de encontrar la "mejor solución" a un problema entre un conjunto muy grande de soluciones posibles [1].

Un problema de optimización puede ser formulado como un proceso donde se desea encontrar el valor óptimo $\mathbf{x}^*$ que minimiza o maximiza la función objetivo $f(\mathbf{x})$. Tal que:

$$\begin{aligned} &\text{Minimizar/ Maximizar} && f(\mathbf{x}),\ x = (x_1, \dots, x_d) \in \mathbb{R}^d \\ &\text{considerando} && \mathbf{x} \in \mathbf{X} \end{aligned} \tag{1.1}$$

Donde $\mathbf{x}$ representa el vector de variables de decisión mientras d especifica su número. $\mathbf{X}$ representa el conjunto de soluciones candidato, conocido también como espacio de búsqueda o espacio de soluciones. En muchas ocasiones el espacio de búsqueda se encuentra limitado por los límites inferior (l_i) o superior (u_i) de cada una de las d variables de decisión tal que $\mathbf{X} = \{\mathbf{x} \in \mathbb{R}^d | l_i \leq x_i \leq u_i, i = 1, \dots, d\}$.

Algunas veces es necesario minimizar $f(\mathbf{x})$, otras veces maximizar. Estos dos tipos de problemas son fácilmente convertidos uno a otro mediante la siguiente relación:

$$\begin{aligned} \min_{\mathbf{x}^*} f(\mathbf{x}) &\Leftrightarrow \max_{\mathbf{x}^*} [-1 \cdot f(\mathbf{x})] \\ \max_{\mathbf{x}^*} f(\mathbf{x}) &\Leftrightarrow \min_{\mathbf{x}^*} [-1 \cdot f(\mathbf{x})] \end{aligned} \tag{1.2}$$

Con el objetivo de aclarar los conceptos anteriores se presenta como ejemplo el siguiente problema de minimización:

$$\text{Minimizar} \qquad f(\mathbf{x}) = x^4 + 5x^3 + 4x^2 - 4x + 1 \qquad (1.3)$$

$$\text{considerando} \qquad \mathbf{x} \in [-4,1]$$

En esta formulación, se presenta como problema, la minimización de una función de una sola variable de decisión (d =1). El espacio de búsqueda $\mathbf{X}$ para este problema está integrado por el intervalo de -4 a 1. Bajo estas circunstancias, la idea es encontrar el valor de *x*, para el cual $f(\mathbf{x})$ presenta su valor mínimo, considerando como el conjunto de soluciones posibles, el intervalo definido [-4,1]. El valor de x que resuelve esta formulación es llamado el valor óptimo y es representado como x^*. La Figura 1.1 presenta una representación gráfica de la función a minimizar.

De la figura 1.1 pueden distinguirse las soluciones A y B que corresponden a dos diferentes mínimos obtenidos de $f(\mathbf{x})$. Este tipo de funciones son conocidas como multimodales por sus características de contener varios mínimos prominentes. El mínimo representado por el punto A representa la solución óptima x^*del problema, mientras que B es solo un mínimo local.

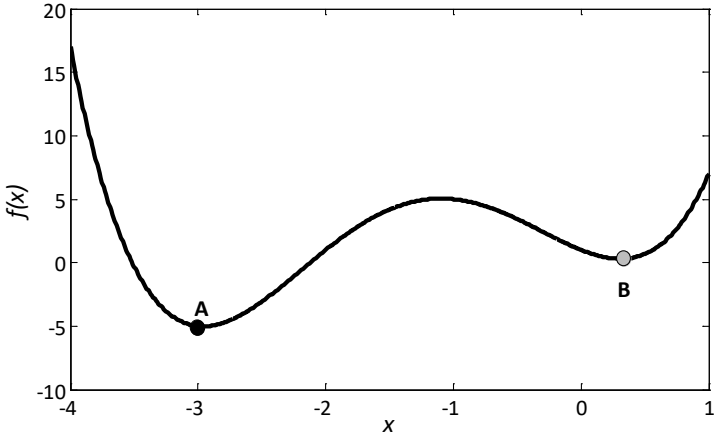

Fig. 1.1. Representación gráfica del problema de optimización formulado en la ecuación 1.3.

1.2 MÉTODOS CLÁSICOS DE OPTIMIZACIÓN

En general, la función $f(\mathbf{x})$ puede tener una forma no lineal con respecto a las variables de decisión $\mathbf{x}$. Debido a esta complejidad, los métodos de optimización plantean procesos iterativos para la exploración del eficiente del espacio de búsqueda. Existen 2 tipos de familias de algoritmos usados para resolver esta clase de problemas [2]: las técnicas clásicas y los esquemas metaheurísticos. Los métodos clásicos se basan en el uso del gradiente de la función $f(\mathbf{x})$ para la generación de nuevas soluciones candidatas. En el caso de los métodos metaheurísticos no necesitan información funcional de la derivada para la búsqueda del vector de decisión que minimiza (o maximiza) una determinada función objetivo. En lugar de ello, en estos métodos se implementa un conjunto de reglas heurísticas que dirigen el proceso de búsqueda. Algunas de estas reglas se basan en la reproducción de fenómenos presentes en la naturaleza o la sociedad.

Por las características de funcionamiento de los algoritmos clásicos de optimización, estos requieren que la función objetivo $f(\mathbf{x})$ presente dos requisitos fundamentales para su uso: que sea dos veces diferenciable y que sea unimodal [3].

En esta sección se hace una revisión básica de las técnicas de optimización basadas en gradiente. Como el libro analiza en profundidad los métodos metaheurísticos, el tema de la optimización clásica es tratado solo marginalmente, se recomienda al lector la consulta de otros libros si desea profundizar en estos temas.

1.2.1 Método del gradiente descendiente

El método del gradiente descendiente, también conocido solamente como el método de gradiente, es una de las primeras técnicas utilizadas para la minimización de funciones objetivo multidimensionales [4]. Este método forma la base en la que se fundamentan varios otros algoritmos de optimización más sofisticados. A pesar de su lenta convergencia, el método de gradiente descendiente es el que más frecuentemente se usa para la optimización de funciones no lineales. Tal hecho se debe a su simplicidad y facilidad de programación.

En este método, partiendo de un punto inicial $\mathbf{x}^0$, el vector de decisión se modifica iterativamente hasta que transcurrido un número *Niter* de iteraciones pueda tentativamente encontrarse la solución óptima $\mathbf{x}^*$. Tal modificación iterativa es determinada por la siguiente expresión [5]:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha \mathbf{g}(f(\mathbf{x})) \tag{1.4}$$

donde k representa la iteración actual y α representa el tamaño del paso de la búsqueda.

En 1.4 el término $\mathbf{g}(f(\mathbf{x}))$ representa el gradiente de la función $f(\mathbf{x})$. El gradiente **g** de una función $f(\mathbf{x})$ en el punto **x** expresa la dirección en la que la función $f(\mathbf{x})$ presenta su máximo crecimiento. De esta manera, en el caso de un problema de minimización, la dirección de descenso puede ser obtenido en sentido contrario a **g** (multiplicando por -1). Bajo esta regla, se garantiza que $f(\mathbf{x}^{k+1}) < f(\mathbf{x}^k)$, lo cual implica que la nueva solución generada es mejor que la anterior.

Normalmente, aunque la formulación de un problema de optimización involucra la definición de una función objetivo $f(\mathbf{x})$, esto es solo con fines didácticos y de demostración. En la práctica no se conoce en forma determinista la definición de $f(\mathbf{x})$. Sus valores son conocidos solamente en los puntos requeridos por el algoritmo de optimización. Bajo estas circunstancias, el gradiente $\mathbf{g}$ es calculado mediante el uso de métodos numéricos.

1.2.1.1 CÁLCULO DEL GRADIENTE

El gradiente de una función multidimensional $f(\mathbf{x})$ $(\mathbf{x} = (x_1, \dots, x_d) \in \mathbb{R}^d)$ representa la manera en que la función varia con respecto a uno de sus *d* dimensiones. De esta manera el gradiente g_{x_1} expresa la forma en que varía $f(\mathbf{x})$ con respecto a x_1. Dicho gradiente es definido apropiadamente como:

$$g_{x_1} = \frac{\partial f(\mathbf{x})}{\partial x_1} \tag{1.5}$$

Para calcular numéricamente el gradiente g_{x_i} se sigue este procedimiento [6].

1. Se genera un nuevo vector de decisión $\tilde{\mathbf{x}}_i$. Dicho vector es igual en todas las variables de decisión que $\mathbf{x}$ excepto en x_i. Este valor será sustituido por $x_i + h$, donde h es un valor muy pequeño. Bajo estas condiciones el nuevo vector es definido como:

$$\tilde{\mathbf{x}}_i = (x_1, x_2, \dots, x_i + h, \dots, x_d) \tag{1.6}$$

2. Se calcula el gradiente g_{x_i} mediante el siguiente modelo:

$$g_{x_i} = \frac{f(\tilde{\mathbf{x}}_i) - f(\mathbf{x})}{h} \tag{1.7}$$

La figura 1.2 muestra una representación gráfica del proceso de cálculo numérico del gradiente considerando un ejemplo simple, donde $f(\mathbf{x})$ tiene 2 dimensiones (x_1, x_2).

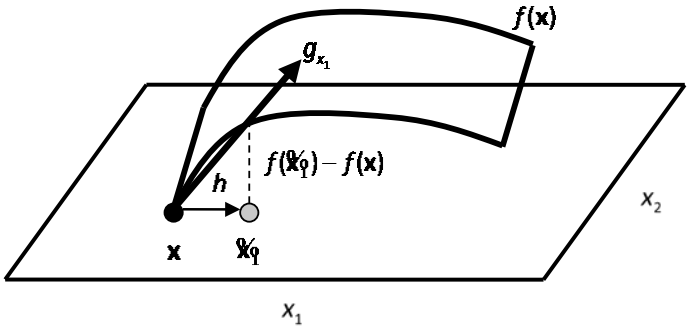

Fig. 1.2. Representación gráfica del cálculo numérico del gradiente.

1.2.1.2 EJEMPLO COMPUTACIONAL EN MATLAB

Con el objetivo de ejemplificar el funcionamiento del algoritmo de gradiente descendiente y su implementación práctica en MatLAB, se pretende resolver el siguiente problema de minimización:

$$\begin{aligned} &\text{Minimizar} && f(x_1, x_2) = 10 - e^{-\left(x_1^2 + 3x_2^2\right)} \\ &\text{considerando} && -1 \leq x_1 \leq 1 \\ & && -1 \leq x_2 \leq 1 \end{aligned} \tag{1.8}$$

En esta formulación, se considera una función bidimensional $f(x_1, x_2)$ con un espacio de búsqueda definido en el intervalo [-1,1] para cada una de las variables de decisión x_1 y x_2. La figura 1.3 muestra una representación de la función objetivo $f(x_1, x_2)$.

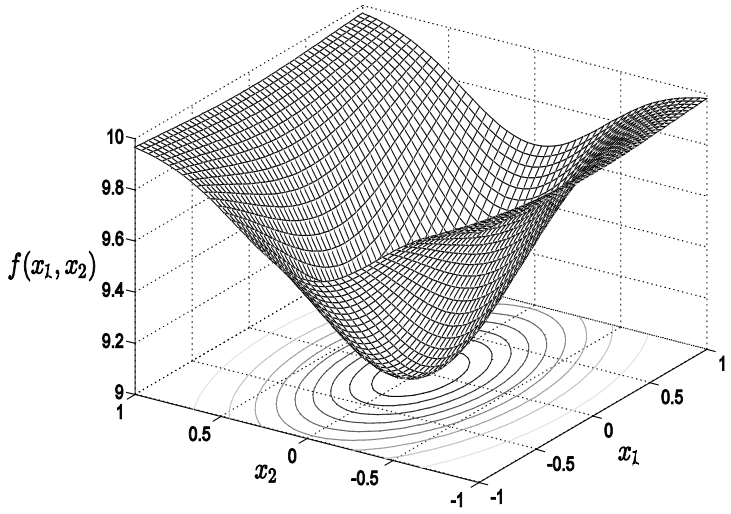

Fig. 1.3. Representación gráfica de la función $10 - e^{-(x_1^2+3x_2^2)}$.

Como puede apreciarse, la función $f(x_1, x_2)$ puede ser derivada dos veces y es unimodal (solo tiene un mínimo). Por tales razones cumple los requisitos para poder ser minimizada bajo el método del gradiente descendiente.

El proceso para la minimización de $f(x_1, x_2)$ mediante el método del gradiente descendiente es presentado en forma de pseudocódigo en el algoritmo 1.1. El procedimiento inicia seleccionando aleatoriamente un vector de decisión $\mathbf{x}^k$ (k=0), dentro del espacio de búsqueda definido en el intervalo [-1,1] para cada una de las variables x_1 y x_2. Después, se calculan los gradientes g_{x_1} y g_{x_2} en el punto $\mathbf{x}^k$. Con esta información, se obtiene una nueva solución $\mathbf{x}^{k+1}$ como consecuencia de aplicar la ecuación de actualización 1.4. Como una posible solución a $f(x_1, x_2)$ está integrada por dos variables de decisión, la actualización de la solución definida por 1.4 considera a cada variable de forma desacoplada. Dicha actualización se define como:

$$x_1^{k+1} = x_1^k - \alpha g_{x_1} \tag{1.9}$$

$$x_2^{k+1} = x_2^k - \alpha g_{x_2}$$

Este proceso es repetido iterativamente hasta que un número máximo de iteraciones *Niter* se hayan alcanzado.

Código 1.1 Algoritmo del Gradiente descendiente

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Algoritmo de Gradiente descendiente                                       %
% Erik Cuevas, Fernando A. Fausto, Jorge Gálvez y Alma Rodríguez            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Limpiar memoria
clear all
%Se define la función a optimizar formulada en 1.8
funstr='10-(exp(-1*(x^2+3*y^2)))';
f=vectorize(inline(funstr));
range=[-1 1 -1 1];
%Se dibuja la función como referencia
Ndiv=50;
dx=(range(2)-range(1))/Ndiv; dy=(range(4)-range(3))/Ndiv;
[x,y] =meshgrid(range(1):dx:range(2),range(3):dy:range(4));
z=(f(x,y));
figure(1);   surfc(x,y,z);
%Se define el número de iteraciones
k=0;
niter=200;
%Se establece el tamaño de la perturbación para el cálculo de los
gradientes
hstep = 0.001;
%Se fija el paso de optimización
alfa=0.05;
%Generacion del punto inicial
xrange=range(2)-range(1);
yrange=range(4)-range(3);
x1=rand*xrange+range(1);
x2=rand*yrange+range(3);
figure
%Se establece el proceso iterativo de optimización
while (k<niter)
%Se evalúa la función
zn=f(x1,x2);
%Se calculan los Gradientes gx1 y gx2
vx1=x1+hstep;
vx2=x2+hstep;
gx1=(f(vx1,x2)-zn)/hstep;
gx2=(f(x1,vx2)-zn)/hstep;
%Se grafica el punto
contour(x,y,z,15); hold on;
plot(x1,x2,'.','markersize',10,'markerfacecolor','g');
drawnow;
hold on;
%Se calcula el nuevo punto
x1=x1-alfa*gx1;
x2=x2-alfa*gx2;
k=k+1;
end
```

Con el objetivo de brindar al lector una ayuda en la implementación el algoritmo 1.1, el código 1.1 muestra el código en MatLAB. En el funcionamiento del código 1.1 primero es graficada la función $f(x_1, x_2)$ con el objetivo de apreciar sus

características. Después, de manera iterativa se muestra la trayectoria que siguen las soluciones candidato desde su valor inicial $\mathbf{x}^0$ hasta encontrar el valor óptimo $\mathbf{x}^*$. La figura 1.4 muestra un ejemplo de esta trayectoria, sobre un contorno de MatLAB, obtenida por el código 1.1.

Algoritmo 1.1 Método de gradiente descendiente para resolver 1.8	
1.	$k \leftarrow 0$
2.	$x_1^k \leftarrow$ Aleatorio [-1,1], $x_2^k \leftarrow$ Aleatorio [-1,1]
3.	**while** (k< *Niter*) **{**
4.	$g_{x_1} \leftarrow \frac{f(x_1^k+h,x_2^k)-f(x_1^k,x_2^k)}{h}$, $g_{x_2} \leftarrow \frac{f(x_1^k,x_2^k+h)-f(x_1^k,x_2^k)}{h}$
5.	$x_1^{k+1} \leftarrow x_1^k - \alpha g_{x_1}, x_2^{k+1} \leftarrow x_2^k - \alpha g_{x_2}$
6.	$k \leftarrow k$+1 **}**

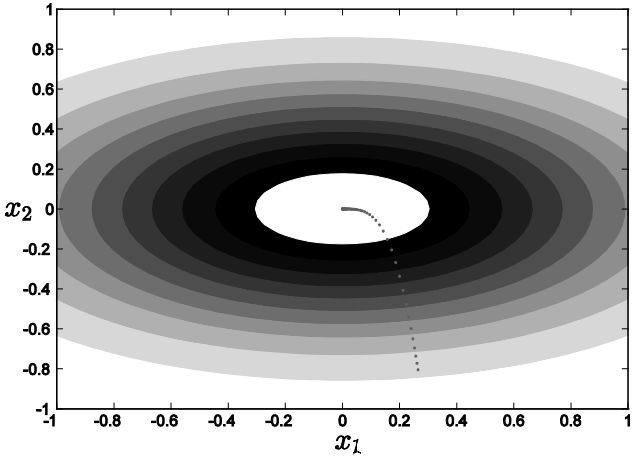

Fig. 1.4. Trayectoria de las soluciones obtenida por el algoritmo de gradiente descendiente al minimizar la función definida en 1.8.

1.2.2 Métodos newtonianos

La dirección descendiente $\alpha \cdot \mathbf{g}(f(\mathbf{x}))$ puede ser determinada también mediante el uso de la segunda derivada de la función de costo $f(\mathbf{x})$ si es disponible o a través de sus muestras. En general, una función continua presenta contornos casi elípticos en sus valores en la proximidad a un mínimo local. De esta manera, si se analiza la función considerando como punto de partida una posición $\mathbf{x}^0$sucificientemente cercana a un mínimo local, la función objetivo puede ser aproximada como una forma cuadrática con la expresión: