Proyecto de Optimización

Integrantes:
Ana Paula González Muñoz C-312
Dennis Daniel González Durán C-312

Facultad Matemática y Computación, Universidad de La Habana

1 Algoritmos estudiados

1.1 Broyden-Fletcher-Goldfarb-Shanno (BFGS)

El método Broyden-Fletcher-Goldfarb-Shanno (BFGS) es un algoritmo de optimización numérica utilizado para encontrar el mínimo de una función no lineal. Pertenece a la familia de métodos quasi-Newton, que son una generalización de los métodos de Newton para optimización. A diferencia del método de Newton que requiere calcular la matriz Hessiana (segunda derivada) de la función, el BFGS construye una aproximación de la inversa de la Hessiana usando sólo gradientes (primera derivada). Esto lo hace más eficiente en términos de cálculo, especialmente para funciones de muchas variables, y es ampliamente utilizado en problemas de optimización en los que la función objetivo es suave y diferenciable.

El algoritmo BFGS actualiza iterativamente la matriz que aproxima la inversa de la Hessiana y la dirección de descenso, moviéndose hacia el mínimo de la función objetivo. Cada iteración busca una dirección descendente y luego actualiza la matriz con información de los gradientes. El método BFGS es popular debido a su equilibrio entre la rapidez del método de Newton y la simplicidad de los métodos de gradiente. Es particularmente útil en problemas donde evaluar la Hessiana es costoso o impráctico, manteniendo una alta eficiencia en la búsqueda del mínimo de la función.

El algoritmo BFGS es uno de los métodos quasi-Newton más populares y su fórmula de actualización es:

$$x_{k+1} = x_k - H_k * \nabla f(x_k)$$

1.2 Evolución Diferencial

El algoritmo de evolución diferencial es un método de optimización estocástico utilizado para resolver problemas de optimización global en espacios de búsqueda continua. Este algoritmo pertenece a la familia de los algoritmos evolutivos y se inspira en procesos biológicos como la selección natural y la recombinación genética. Funciona manteniendo una población de posibles soluciones que se actualizan iterativamente mediante operadores como la mutación, recombinación y selección. En cada iteración, se crean nuevas soluciones combinando las existentes de manera aleatoria y se seleccionan las mejores para la siguiente generación, buscando minimizar o maximizar una función objetivo.

El diferencial de evolución es especialmente efectivo en problemas donde el espacio de búsqueda es vasto, no lineal o ruidoso. Su simplicidad y capacidad para evitar caer en óptimos locales lo hacen adecuado para una amplia variedad de aplicaciones, desde ingeniería hasta finanzas y biología. A diferencia de otros algoritmos evolutivos, el diferencial de evolución utiliza diferencias entre individuos de la población para guiar la búsqueda, lo que mejora su capacidad para explorar el espacio de soluciones y converger hacia el óptimo global.

2 Problemas a resolver

Para cada problema, se realizó un análisis tomando en cuenta los siguientes aspectos:

1. Tiempo de ejecución.
2. Número de iteraciones.
3. Valor de la función objetivo.

2.1 Problema 1: Rosenbrock Function

$$f_{105}(\mathbf{x}) = \sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$$
$$-30 \leq x_i \leq 30$$

2.1.1 Algoritmo de solución

Para esta función usamos el algoritmo BFGS para encontrar el mínimo. Para D=2 obtuvimos los siguientes resultados:

1. Tiempo de ejecución: 0.04022049903869629 segundos
2. Número de iteraciones: 152
3. Valor de la función objetivo: 2.07474544990368e-11
4. Punto hallado: [0.99999545 0.99999088]

2.1.2 Valoración de la Calidad del Punto Hallado

El punto hallado [0.99999545 0.99999088] se encuentra bastante cercano al punto mínimo global x*=[1,1]. Esto indica que el algoritmo se acerca bastante al punto del mínimo global.

2.1.3 Valoración del tiempo computacional

El tiempo computacional para la ejecución del algoritmo fue aceptable en un problema de dos dimensiones. Aunque a medida que D aumenta, el tiempo de cómputo crecerá en consecuencia.

2.1.4 Variación al Aumentar la Dimensión del Problema

Al aumentar la dimensión del problema observamos que tanto el punto hallado como el míinimo global se mantienen relativamente estables aunque crece el timepo de ejecución.

2.2 Problema 2: Salomon Function

$$f_{110}(\mathbf{x}) = 1 - \cos\left(2\pi\sqrt{\sum_{i=1}^{D} x_i^2}\right) + 0.1\sqrt{\sum_{i=1}^{D} x_i^2}$$
$$-100 \leq x_i \leq 100$$

2.2.1 Algoritmo de solución

Para esta función usamos el algoritmo Evolución diferencial para encontrar el mínimo. Para D=2 obtuvimos los siguientes resultados:

1. Tiempo de ejecución: 0.20726966857910156 segundos
2. Número de iteraciones: 132
3. Valor de la función objetivo: 0
4. Punto hallado: [0 0]

2.2.2 Valoración de la Calidad del Punto Hallado

El punto hallado [0 0] es exactamente el punto para el mínimo global.

2.2.3 Valoración del tiempo computacional

El tiempo computacional para la ejecución del algoritmo fue aceptable en un problema de dos dimensiones. Aunque a medida que D aumenta, el tiempo de cómputo crecerá considerablemente

2.2.4 Variación al Aumentar la Dimensión del Problema

Al aumentar la dimensión del problema observamos que el punto hallado y el mínimo global no son estables con respecto al punto y el mínimo conocido. El tiempo de ejecución aumenta considerablemente.

3 Conclusiones

Los algoritmos estudiados, BFGS y Evolución Diferencial, demostraron ser efectivos en la resolución de problemas de optimización al encontrar puntos cercanos o iguales al mínimo global con tiempos computacionales aceptables para problemas de baja dimensión. Sin embargo, al aumentar la dimensión del problema, se observa un incremento significativo en el tiempo de cómputo y, en algunos casos, una menor estabilidad en los puntos hallados, especialmente con la Evolución Diferencial. Esto resalta la importancia de seleccionar el algoritmo adecuado según la naturaleza y la escala del problema a resolver.