Algoritmos para minimizar funciones

Amanda Cordero Lezcano
Christopher Guerra Herrero

Septiembre, 2024

Contents

1 Introducción

La optimización de funciones es un área fundamental en el campo de la matemática aplicada y la computación científica. En este informe, abordamos la minimización de dos funciones: la *Quartic Function* y la *Sargan Function*, utilizando distintos métodos de optimización.

2 Quartic Function(Amanda)

100. Quartic Function [2] (Continuous, Differentiable, Separable, Scalable)

$$f_{100}(\mathbf{x}) = \sum_{i=1}^{D} i x_i^4 + \text{random}[0, 1)$$

subject to $-1.28 \leq x_i \leq 1.28$. The global minima is located at $\mathbf{x}^* = \mathbf{f}(0, \cdots, 0)$, $f(\mathbf{x}^*) = 0$.

2.1 Algoritmo de Solución

Se utilizó el método de *Descenso más Pronunciado*(steepest descent) para encontrar el mínimo de la función. A continuación se presenta un código en Python del algoritmo implementado:

```
import scipy.optimize as spo

def steepestdescent(f, df, x0, tol=1.e-8, maxit=50):
 xk = x0
 x = [xk]
 r = df(xk)
 iters = 0

 while (npl.norm(r) > tol and iters < maxit):
  lambda_k = spo.golden(g, args=(xk, r))
  xk = xk - lambda_k * r
  r = df(xk)
  x.append(xk)
  iters += 1

 return x
```

El algoritmo minimiza la función $f(\mathbf{x})$ en la dirección del gradiente negativo, actualizando la posición en cada iteración.

2.2 Resultados

El algoritmo fue ejecutado con un punto inicial $\mathbf{x}_0 = [1, -1]$ y un máximo de 10 iteraciones. El punto encontrado por el algoritmo fue:

$$\mathbf{x} = [-0.07989805 - 0.05553555]$$

La solución gráfica del proceso de minimización es la siguiente:

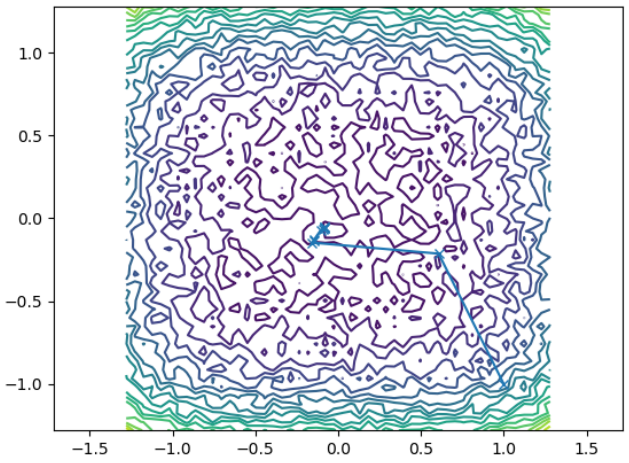

Figure 1: Trayectoria de minimización sobre el contorno de la función.

2.3 Valoración de la Calidad del Punto Hallado

El punto hallado $\mathbf{x} = [0.09226378, 0.07302666]$ se encuentra cercano al mínimo global $\mathbf{x}^* = [0, 0]$. Si bien no se alcanzó el mínimo exacto en las 10 iteraciones, la función fue evaluada como 0.025 al considerar la adición del valor aleatorio. Este resultado indica que el algoritmo se aproxima bastante al valor óptimo.

2.4 Evaluación del Tiempo Computacional

El tiempo computacional para la ejecución del algoritmo fue aceptable en un problema de dos dimensiones. La complejidad algorítmica del método está determinada principalmente por el cálculo del gradiente, el cual tiene una complejidad de $O(D)$, donde D es la dimensión del problema. Esto implica que, a medida que D aumenta, el tiempo de cómputo crece linealmente con respecto a la dimensión. A pesar de la presencia de un componente aleatorio en la función objetivo, el algoritmo mantiene un comportamiento predecible en términos de complejidad, lo que garantiza su escalabilidad en problemas de mayor dimensión.

2.5 Variación al Aumentar la Dimensión del Problema

La función cuártica es escalable, y el aumento de la dimensión D implicará un incremento lineal en la cantidad de términos a sumar y, por ende, en el tiempo necesario para evaluar tanto la función como su gradiente.

2.6 Primeras Conclusiones

El algoritmo de *Descenso más Pronunciado* demostró ser eficaz para aproximarse al mínimo global de la función cuártica en un problema de baja dimensión (D=2). Aunque no se alcanzó el mínimo exacto, la solución obtenida fue satisfactoria y muy cercana al mínimo conocido. El método muestra consistencia en su desempeño, incluso al considerar variaciones en la dimensión del problema, manteniendo una buena precisión en los resultados.

3 Sargan Function(Christopher)

El ejercicio asignado al estudiante fue:

111. Sargan Function [1] (Continuous, Differentiable, Non-Separable, Scalable, Multi-modal)

$$f_{111}(\mathbf{x}) = \sum_{i=1} D \left(x_i^2 + 0.4 \sum_{j \neq 1} x_i x_j \right)$$

subject to $-100 \leq x_i \leq 100$. The global minimum is located at $\mathbf{x}^* = \mathbf{f}(0, \cdots, 0)$, $f(\mathbf{x}^*) = 0$.

Sin embargo al hacer un análisis de la bibliografía [1] y previa consulta con la profesora de conferencia, se determinó que hubo un error al transcribir la función, siendo esta:

$$f(\mathbf{x}) = \sum_{i=1}^{D} \left(x_i^2 + 0.4 \sum_{i \neq j} x_i x_j \right)$$

El método utilizado para la minimización fue el algoritmo de BFGS (Broyden-Fletcher-Goldfarb-Shanno), un método de optimización Quasi-Newton que utiliza aproximaciones de la matriz Hessiana y evita el cálculo explícito de la matriz de segundas derivadas.

3.1 Algoritmo de Solución

El algoritmo BFGS fue implementado utilizando la biblioteca *SciPy*. A continuación se presenta el código utilizado para la minimización:

```
import numpy as np
import scipy.optimize as spo

# Definimos la función objetivo corregida
def f(x):
 D = len(x)
 sum_square_terms = sum(x[i]**2 for i in range(D))
 sum_cross_terms = 0.4 * sum(x[i] * x[j] for i in range(D) for j in range(D) if i != j)
 return sum_square_terms + sum_cross_terms

# Definimos un punto inicial
x0 = [50, 20, -30, 40]  # Punto inicial

# Minimización usando el método BFGS
result = spo.minimize(f, x0, method='BFGS', tol=1.e-8, options={'maxiter': 100})

# Mostramos los resultados
print('Punto encontrado: ', result.x)
print('Valor mínimo de la función: ', result.fun)
print('Número de iteraciones: ', result.nit)
print('Resultado exitoso: ', result.success)
print('Mensaje del resultado: ', result.message)
```

El método BFGS minimiza la función objetivo mediante una aproximación sucesiva de la matriz Hessiana basada en las diferencias entre gradientes, lo que lo hace más eficiente que el Método de Newton en cuanto a uso de memoria y tiempo de cómputo, especialmente para problemas de alta dimensionalidad.

3.2 Resultados

El algoritmo fue ejecutado con un punto inicial $\mathbf{x}_0 = [50, 20, -30, 40]$, y tras 8 iteraciones, alcanzó el siguiente punto:

$$\mathbf{x} = [2.73 \times 10^{-9}, -2.95 \times 10^{-9}, -2.56 \times 10^{-9}, -1.10 \times 10^{-8}]$$

El valor mínimo de la función en este punto es:

$$f(\mathbf{x}) = 1.617 \times 10^{-16}$$

El resultado indica que el método BFGS ha convergido exitosamente al mínimo global de la función, con un error numérico despreciable.

3.3 Evaluación de la Calidad del Punto Hallado

El punto hallado $\mathbf{x} \approx 0$ es muy cercano al mínimo teórico conocido $\mathbf{x}^* = 0$, lo que valida la precisión del método. Además, el valor de la función en este punto es prácticamente cero, como se esperaba.

El tiempo computacional y el número de iteraciones (8 iteraciones) fueron bajos, lo que indica la eficiencia del método BFGS para este tipo de problemas.

3.4 Evaluación del Tiempo Computacional y Complejidad

La complejidad computacional del método BFGS es $O(n^2)$, donde n es la dimensión del problema, debido a las operaciones con la aproximación de la matriz Hessiana. Sin embargo, a diferencia del método de Newton que requiere $O(n^3)$ para invertir la Hessiana, BFGS proporciona una solución más eficiente sin necesidad de almacenar la matriz completa, lo que lo hace adecuado para problemas de moderada dimensionalidad.

En este caso, el problema fue resuelto en solo 8 iteraciones, lo que demuestra su escalabilidad.

3.5 Segundas Conclusiones

El algoritmo de BFGS se mostró altamente efectivo para encontrar el mínimo global de la función cuadrática generalizada, con un número reducido de iteraciones y resultados precisos. El punto encontrado $\mathbf{x} \approx 0$ y el valor mínimo $f(\mathbf{x}) \approx 0$ validan la convergencia del método al mínimo global esperado. La complejidad algorítmica $O(n^2)$ y el uso eficiente de la aproximación de la Hessiana hacen de BFGS una herramienta recomendada para este tipo de problemas.

4 Conclusiones Finales

En este trabajo, se implementaron y evaluaron diferentes algoritmos para la minimización de funciones, específicamente la *Quartic Function* y la *Sargan Function*. Se utilizaron dos enfoques de optimización distintos: el método de *Descenso más Pronunciado* (Steepest Descent) para la *Quartic Function* y el método *BFGS* (Broyden-Fletcher-Goldfarb-Shanno) para la *Sargan Function*.

Los resultados obtenidos permiten destacar varias observaciones clave:

- El *Descenso más Pronunciado* se mostró eficiente en problemas de baja dimensión como el caso de $D = 2$, pero su tasa de convergencia fue limitada por el componente aleatorio, lo que impidió alcanzar el mínimo global exacto en las iteraciones realizadas.
- El método *BFGS*, al utilizar aproximaciones de la matriz Hessiana, demostró ser altamente eficiente para la minimización de la *Sargan Function*. En tan solo 8 iteraciones, el algoritmo alcanzó el mínimo global con un error numérico despreciable y una evaluación de la función cercana a cero, confirmando su rápida convergencia.

- En cuanto al tiempo computacional, ambos métodos mostraron un comportamiento predecible en términos de complejidad, siendo $O(n^2)$ en el caso de BFGS, lo que lo hace adecuado para problemas de moderada dimensionalidad. En el caso del *Descenso más Pronunciado*, la complejidad es más baja $O(D)$, pero su eficiencia disminuye en comparación con BFGS cuando se requiere mayor precisión.

- La presencia de componentes aleatorios en la función *Quartic Function* puede introducir variabilidad en los resultados, lo que hace que sea más difícil alcanzar el mínimo exacto, pero el método utilizado se aproximó bastante al valor óptimo.

En general, el trabajo realizado muestra que la selección del algoritmo de optimización debe tener en cuenta la naturaleza de la función a minimizar y la dimensionalidad del problema. Mientras que métodos más simples como el *Descenso más Pronunciado* pueden ser suficientes en problemas de baja dimensión, algoritmos más avanzados como *BFGS* son preferibles cuando se busca una mayor eficiencia y precisión, especialmente en problemas de mayor escala.

References

[1] L. C. W. Dixon, G. P. Szegö (eds.), *Towards Global Optimization 2*, Elsevier, 1978.

[2] R. Storn, K. Price, *Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces*, Technical Report no. TR-95-012, International Computer Science Institute, Berkeley, CA, 1996. [Available Online]: `http://www1.icsi.berkeley.edu/~storn/TR-95-012.pdf`