Optimización

Algoritmo BFGS

Tarea 12

20 de mayo de 2021

Erika Rivadeneira Pérez
erika.rivadeneira@cimat.mx
Matemáticas Aplicadas - CIMAT

1. Resumen

En el presente reporte se implementan los métodos del descenso por gradiente y del método Cuasi-Newton BFGS. Se comparan ambos métodos en cuanto al tiempo de cómputo, número de iteraciones y error. Se observa que el método BFGS converge mucho más rápido que el descenso por gradiente.

2. Introducción

Los métodos Cuasi-Newton son de gran importancia para la resolución de problemas de programación no lineal, dado que el costo computacional comparado con el método de Newton, el cual tiene una tasa de convergencia cuadrática, es mucho menor en problemas donde la matriz es muy grande. Dos de los métodos cuasi Newton son el método DFP y el BFGS. La fórmula de actualización DFP es bastante efectiva, pero pronto fue superada por la fórmula BFGS, la cual es actualmente considerada la más eficiente de todas las fórmula de actualización Quasi-Newton.
En el presente trabajo se enfoca en el método BFGS ya que este método es más robusto que el DFP como se mencionó anteriormente y se lo compara con el método de máximo descenso (mencionado en reportes pasados). A continuación se menciona cómo se construye el método y posteriormente se muestra la implementación del método en una función y se comparan los resultados con el método del máximo descenso.

3. Metodología

3.1. Algoritmo BFGS

En el BFGS la idea es calcular la aproximacion de matriz inversa del Hessiano $\mathbf{H}_{k+1}$ basado en $\mathbf{B}_{k+1}$ Para ello se considera la ecuacion DFP, con $\rho_k = \frac{1}{s_k^T y_k}$

$$\mathbf{B}_{k+1} = \left(\mathbf{I} - \rho_k y_k \boldsymbol{s}_k^T\right) \mathbf{B}_k \left(\mathbf{I} - \rho_k \boldsymbol{s}_k y_k^T\right) + \rho_k y_k \boldsymbol{y}_k^T$$

y las relaciones

$$\mathbf{B}_{k+1} \boldsymbol{s}_k = y_k$$

y

$$\mathbf{H}_{k+1} \boldsymbol{y}_k = s_k$$

Ahora $\rho_k = \frac{1}{\boldsymbol{y}_k^T s_k}$ y por tanto:

$$\mathbf{H}_{k+1} = \left(\mathbf{I} - \rho_k s_k \boldsymbol{y}_k^T\right) \mathbf{H}_k \left(\mathbf{I} - \rho_k \boldsymbol{y}_k s_k^T\right) + \rho_k s_k s_k^T$$

Como Hessiano inicial H_0 se puede usar información especifica acerca del problema, por ejemplo, configurándolo como el inverso de una aproximación del Hessiano calculado por diferencias finitas en x_0. De otra manera, se puede simplemente configurarla para que sea la matriz identidad, o un múltiplo de la matriz identidad, donde el múltiplo es escogido para reflejar el escalamiento de las variables. En nuestro problema se escoge un múltiplo de la matriz identidad.

4. Implementación

A continuación se muestra el pseudo-código del método BFGS. Se requiere de input el vector x_0 de valores iniciales y un Hessiano inicial H_0, el cual es un múltiplo de la matriz identidad como se mencionó en la anterior sección.

BFGS

Input: x_0, H_0
1: Hacer $k = 0$
2: **while** $||\nabla f_k|| \neq 0$ no converja **do**
3: $d_k = -\boldsymbol{H}_k \nabla f_k$
4: Calcular α_k usando búsqueda en línea.
5: $x_{k+1} = x_k + \alpha_k d_k$
6: Calcular $\nabla f_{k+1}, \boldsymbol{y}_k, \boldsymbol{s}_k$ y actualizar $\mathbf{H}_{k+1} = \left(\mathbf{I} - \rho_k \boldsymbol{s}_k \boldsymbol{y}_k^T\right) \mathbf{H}_k \left(\mathbf{I} - \rho_k \boldsymbol{y}_k \boldsymbol{s}_k^T\right) + \rho_k \boldsymbol{s}_k \boldsymbol{s}_k^T$
7: $k = k + 1$
8: **end while**
Output: x^*

5. Resultados

Se utilizaron los métodos de descenso por gradiente y BFGS para resolver el siguiente problema de optimización

$$F(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left(h_\theta\left(\boldsymbol{x}_i\right) - y_i\right)^2 \qquad (1)$$

donde $\left(\boldsymbol{x}_i, y_i\right), \boldsymbol{x}_i \in \mathbb{R}^n, y_i \in \{0,1\}$, $i = 1, 2, \cdots, N$ vienen dados y

$$h_\theta(\boldsymbol{x}) = f_{\boldsymbol{a},b}\left(g_{\boldsymbol{c},\boldsymbol{d}}(\boldsymbol{x})\right)$$
$$g_{c,d} : \mathbb{R}^n \to \mathbb{R}^m$$
$$f_{\boldsymbol{a},b} : \mathbb{R}^m \to \mathbb{R},$$

$m \in \mathbb{N}$ es conocida,

$$g_{c,d}(\boldsymbol{x}) = \left[\sigma\left(\boldsymbol{c}_j^T \boldsymbol{x} + d_j\right)\right]_{j=1}^m$$
$$f_{\boldsymbol{a},b}(\boldsymbol{z}) = \sigma\left(\boldsymbol{a}^T \boldsymbol{z} + b\right)$$
$$\sigma(t) = \frac{1}{1 + e^{-t}}, t \in \mathbb{R}$$

y θ correponde al conjunto de parámetros $\boldsymbol{a}, b, \boldsymbol{c}, \boldsymbol{d}$.

Dados los datos, se utilizó una tolerancia de 10^{-4} para ambos métodos y se consideró un número máximo de iteraciones de 10000 para el método BFGS. Además, se consideró $m = 2$ con parámetros inciales

$$a = \left[1, 1\right], b = 1, c = \begin{bmatrix} 1 & 1 & ... & 1 \\ 1 & 1 & ... & 1 \end{bmatrix} \text{ y } d = \left[1, 1\right],$$

para los cuales se obtuvieron los resultados de la tabla (1)

	Descenso por Gradiente	**BFGS**
Iteraciones	25	1
Tiempo (s)	899.23	37.81
Error	0.46484	0.46484

Tabla 1: Iteraciones, tiempo de compilación y error de los métodos de descenso por gradiente y BFGS

El error fue calculado de la siguiente manera

$$\text{error } = \frac{1}{|\mathcal{T}|} \sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{T}} \left| \mathbf{1}_{h_{\hat{\theta}}(x_i) > 0{,}5}\left(\boldsymbol{x}_i\right) - y_i \right|,$$

donde $|\ \mathcal{T}\ \}$ representa al número de elementos del conjunto $\mathcal{T}$.
Adicionalmente, se calculó la norma del gradiente (aproximado) de la función (1) en cada método obteniendo

- Por el método de descenso por gradiente:
$$|\nabla f(\theta^*)| = 0{,}000561359$$
- Por el método BFGS:
$$|\nabla f(\theta^*)| = 0{,}0$$

6. Conclusiones

A medida que se aumentó el valor de m el tiempo de cómputo aumenta, dado este hecho se decidió tomar $m = 2$. Se puede observar en la table (1) de resultados que el mejor método fue el de BFGS dado que su tiempo de cómputo fue mucho más bajo, al igual que sus iteraciones, que el método de descenso por gradiente. El error fue el mismo para ambos métodos. Adicionalmente, es necesario mencionar que se aproximó al gradiente de la función (1) por diferencias de Taylor hacia adelante.

Referencias

[1] J. Nocedal and S. J. Wright. Numerical Optimization. Springer Series in Operation Research, 2000.