Programación Lineal: El Método Simplex

Ana Toxqui, *MMOP, CIMAT,*

***Abstract*—La programación lineal y el Método Simplex han demostrado ser herramientas indispensables para la resolución eficiente de problemas complejos en una amplia gama de disciplinas, en este trabajo se muestra una implementación en Python de dicho método.**

***Index Terms*—Programación Lineal, Optimización, Minimización, Métodos iterativos, Método Simplex**

I. Introducción

HOY en día la programación lineal y el método simplex son herramientas de optimización ampliamente usadas en diferentes campos del conocimiento y las organizaciones para la toma de decisiones, por un lado la programación lineal es una técnica matemática que permite resolver una amplia variedad de problemas de optimización bajo la idea de maximizar o minimizar los beneficios modelados en una función lineal, sujeta a un conjunto de restricciones lineales.

Debido a que las organizaciones en enfrentan desafíos más complejos existe la necesidad de proporcionar algoritmos que resuelvan de manera rápida y eficiente estos problemas, entre los primeros algoritmos propuestos se encuentra el Método Simplex el cual sigue siendo una herramienta fundamental en la optimización lineal. Este método proporciona una metodología elegante y eficiente de encontrar la solución óptima. En esencia, el Método Simplex se basa en un proceso iterativo de mejora continua, donde en cada iteración se acerca a una mejora del valor objetivo.

En este artículo, se explora el Método Simplex mediante su formulación teórica y posteriormente se muestra su funcionamiento en Python con una función creada, se probará la función para un problema especifico de programación lineal y se verá como después de una cantidad de repeticiones se llega a la solución óptima, finalmente se evaluaran los resultados obtenidos y se darán las conclusiones generales.

II. Marco teórico

El Método Simplex es un algoritmo iterativo que resuelve problemas de programación lineal al explorar sistemáticamente las soluciones factibles en busca de la óptima. La metodología detrás del Método Simplex involucra la iteración entre soluciones factibles mejoradas para maximizar o minimizar la función objetivo.

El algoritmo comienza con una solución inicial y, en cada paso, busca una variable básica para reemplazar por una no básica, con el objetivo de mejorar el valor objetivo. Se extraen las matrices y vectores relevantes, se calculan relaciones y se toman decisiones basadas en cálculos intermedios para ajustar la solución actual.

Durante cada iteración, el Método Simplex evalúa si se cumplen las condiciones de optimización y si la solución es acotada. Si se cumplen, se declara un punto óptimo. Si no, se continúa el proceso de exploración y ajuste hasta que se alcance la solución óptima.

III. Implementación del Método de Simplex en Python

Para observar el funcionamiento del Método Simplex se creo una función en Python con la etiqueta "SimplexOneStep", la cual muestra el proceso que se sigue en un solo paso del Método Simplex. A cotinuación se meustra el código de dicha función.

```
def SimplexOneStep(A,b,c,B,N):
    matriz_B = A[:, B]
    B_inv = np.linalg.inv(matriz_B)
    x_b = np.dot(B_inv, b)
    c_B = [c[indice] for indice in B]
    c_N = [c[indice] for indice in N]
    lamda =  np.dot(B_inv.T, c_B)
    matriz_N = A[:, N]
    s_n = c_N - np.dot(matriz_N.T, lamda)


    if np.all(s_n >= 0):
        print("Punto optimo encontrado")
        return x_b

    q = N[0]
    A_q = A[:, q].T
    d = np.dot(B_inv, A_q)

    if np.all(d <= 0):
        print("El problema no esta acotado")
        return
    residuo_min = 10000
    p = -1
    for i in range(len(d)):
        if d[i] > 0:
            cociente_q = x_b[i] / d[i]
            if cociente_q < residuo_min:
                residuo_min = cociente_q
                p = B[i]
    for i in range(len(B)):
        if B[i] == p:
            B[i] = q
```

```
    N = list(set(range(A.shape[1]))
        - set(B))

    return x_b,B, N
```

La función comienza por calcular las matrices y vectores necesarios para el cálculo del paso. Extrae las columnas correspondientes a las variables básicas y no básicas de la matriz de coeficientes, realiza cálculos importantes como la inversa de la matriz básica y los vectores x_b, c_B y c_N, calcula el vector lambda utilizando la transpuesta de la matriz inversa de las variables básicas. Estos cálculos preparan el terreno para las decisiones que se tomarán en el paso.

Luego, la función verifica si la solución actual satisface las condiciones de optimización. Si es así, declara haber encontrado un punto óptimo. Sin embargo, si las condiciones no se cumplen, la función realiza una serie de cálculos y evaluaciones adicionales. La función SimplexOneStep identifica una variable no básica q, a través de cálculos verifica si la solución es acotada. Si la solución no es acotada, se presenta un mensaje de que el problema no es acotado, es decir, que no se puede resolverse con el Método Simplex.

Si la solución es acotada, se determina qué variable básica p debe ser reemplazada por la variable no básica q. Luego, se actualizan las listas de variables básicas y no básicas para reflejar este cambio. Finalmente se regresa un nuevo punto x_b junto con los nuevos conjuntos de variables básicas y no básicas.

El proceso se puede apreciar en el siguiente pseudocodigo:

Procedimiento (Un Paso de Simplex).
Dado $\mathcal{B}, \mathcal{N}, x_{\mathrm{B}} = B^{-1}b \geq 0, x_{\mathrm{N}} = 0$;
Resuelva $B^T\lambda = c_{\mathrm{B}}$ para λ,
Calcular $s_{\mathrm{N}} = c_{\mathrm{N}} - N^T\lambda$; (* precio *)
si $s_{\mathrm{N}} \geq 0$
detener; (* punto óptimo encontrado *)
Seleccione $q \in \mathcal{N}$ con $s_q < 0$ como índice de entrada;
Resuelva $Bd = A_q$ para d;
si $d \leq 0$
detener; (* el problema es ilimitado *)
Calcule $x_q^+ = \min_{i|d_i>0} (x_{\mathrm{B}})_i / d_i$, y use p para denotar la minimización de i ;
Actualizar $x_{\mathrm{B}}^+ = x_{\mathrm{B}} - dx_q^+, x_{\mathrm{N}}^+ = \left(0, \ldots, 0, x_q^+, 0, \ldots, 0\right)^T$;
Cambie $\mathcal{B}$ agregando q y eliminando la variable básica correspondiente a la columna p de B.

Esta función se realiza varias veces hasta encontrar la solución del problema, para cada nueva iteración de la función es necesario actualizar los valores iniciales de la función con los resultantes de la ejecución realizada con anterioridad.

A. Problema de Prueba

Para probar la eficacia del Método Simplex se tomará el siguiente problema de programación lineal.

$$\begin{aligned} \min -4x_1 - 2x_2 \text{ sujeto a} \\ x_1 + x_2 + x_3 = 5, \\ 2x_1 + (1/2)x_2 + x_4 = 8, \\ x \geq 0. \end{aligned}$$

Este problema cuenta con 4 variables de respuesta, sujeto a dos restricciones de desigualdad y una de no negatividad. Tiene una solución óptima en el punto $(\frac{11}{3}, \frac{4}{3}, 0, 0)$

IV. Resultados

Se modelaron las entradas del problema de la siguiente manera:

```
B = np.array([2, 3])
N = np.array([0, 1])
A = np.array([[1, 1, 1, 0],[2, 0.5, 0, 1]])
b = np.array([5,8])
c = np.array([-4,-2,0,0])
```

De esta manera se obtuvo:

Iteración	x_b	B	N
1	[5. 8,0,0]	[2 0]	[1,3]
2	[1. 4, 0, 0]	[1 0]	[2,3]
3	[3.6, 1.3. 0, 0]	[1 0]	[2,3]

TABLE I
Resultados de la función "SimplexOneStep"

V. Análisis y Conclusión

Con la implementación de un paso del Método Simplex aplicado de manera iterativa se llego al resultad óptimo del problema de programación lineal planteado, se mostró la eficacia del método y se reafirmo su potencia en estos problemas, cabe mencionar que esta implementación es solo una versión sencilla del método Simplex, ya que actualmente existen versiones más sofisticadas e incluso existen desarrollos de software basado en el método Simplex y su variantes.