Modelos de Optimización
Laboratorio 1

Osmany Pérez Rodríguez
Enrique Martínez González
Carmen Irene Cabrera Rodríguez
Grupo C412

La programación de los ejercicios orientados se realizó en todos los casos en el lenguaje de programación Python. La ejecución de los mismos fue llevada a cabo en una computadora HP con procesador AMD10 y 8GB RAM.

1

Los siguientes ejercicios (ej 1, 2, 3 de la CP1) fueron resueltos haciendo uso de PuLP, una biblioteca de Python para optimización lineal.

1 Producto:

- Tomate → 0
- Lechuga → 1
- Acelga → 2

Parámetros:

$p_i \rightarrow$ precio de la semilla del producto i
$h_i \rightarrow$ hombres-día necesarios para producto i
$ct_i \rightarrow$ cantidad de toneladas por hectárea del producto i
$ch_i \rightarrow$ cantidad de hectáreas por tonelada de semilla de producto i

Variables:
$x_i \rightarrow$ cantidad de hectáreas sembradas del producto i

Modelo:

$$max \Sigma_{i=0}^{2} p_i * ct_i * x_i - \frac{x_i}{ch_i} * p_i - x_i * h_i * 5$$

$$s.a:$$

$$\Sigma_{i=0}^{2} h_i * x_i \leqslant 450$$

,

$$\Sigma_{i=0}^{2} x_i \leqslant 1200$$

,

$$x_i \geqslant 0 \ \forall i \in \{0, 1, 2\}$$

Puede encontrar el código del problema a través del siguiente enlace

Los resultados obtenidos fueron los siguientes:

Producto	Cantidad (ha)
Tomate	90
Lechuga	0
Acelga	0

Ganancia: 11430

Cantidad de iteraciones = 1

Tiempo = 0.002

2 Bodega:

- Proa → 0
- Centro → 1
- Popa → 2

Artículo:

- A → a_0
- B → a_1
- C → a_2

Parámetros:

T_i → cantidad de toneladas del artículo i
C_i → capacidad de la bodega i en toneladas
G_i → ganancia por tonelada del artículo i

Variables:
X_{ij} → cantidad de toneladas del artículo A_i en la bodega j

Modelo:

$$min \sum_{j=1}^{2} \sum_{i=0}^{4} X_{i,j} * PR_i$$

$$s.a:$$

$$X_{i,j} \geqslant 0, \forall i, j \in 0, 1, 2$$

$$\sum_{j=0}^{2} X_{i,j} \leqslant T_i, \forall i \in 0, 1, 2$$

$$\sum_{i=0}^{2} X_{i,j} \leqslant C_j, \forall j \in 0, 1, 2$$

$$\frac{\sum_{i=0}^{2} X_{i,0}}{C_0} = \frac{\sum_{i=0}^{2} X_{i,1}}{C_1} = \frac{\sum_{i=0}^{2} X_{i,2}}{C_2}$$

Puede encontrar el código del problema a través del siguiente enlace

Los resultados obtenidos fueron los siguientes:

Producto	Centro	Popa	Proa
A	2500	0	0
B	500	1500	2000
C	0	0	0

Ganancia: 47000

Cantidad de iteraciones = 4

Tiempo = 0.002

3 Productos:

- P1 → 0
- P2 → 1

Ingredientes:
- M1 → 0
- M2 → 1
- Aceite → 2
- Secador → 3
- Solvente → 4

Parámetros:
- $P_{i,j}$ → proporción del ingrediente i en el producto j. Resaltar que la receta de P_1 y P_2 no tienen como ingredientes a los componentes M_1 ni M_2. Por lo que $P_{0,0} = P_{1,0} = P_{0,1} = P_{1,1} = 0$.
- CT_i → cantidad total que se dispone del componente i.
- PR_i → precios del componente i.
- $Y_{i,j}$ → proporción del ingrediente j en el componente i.
- CP_i → cantidad que se necesita producir del producto i.

Variables:
$X_{i,j}$ → cantidad de hl del ingrediente i en el producto j

Modelo:

$$min \sum_{j=0}^{1} \sum_{i=0}^{4} X_{i,j} * PR_i$$

$$s.a:$$

$$X_{i,j} \geqslant 0, \forall i \in 0,1,2,3,4, \forall j \in 0,1$$

$$\sum_{j=0}^{1} X_{i,j} \leqslant CT_i, i \in 0,1$$

$$\sum_{i=0}^{4} X_{i,j} \geqslant CP_j, j \in 0,1$$

$$\sum_{i=0}^{4}(X_{i,j} * Y_{i,k}) = P_{k,j} * \sum_{i=0}^{4} X_{i,j}, k \in 2,3,4, j \in 0,1$$

Puede hallar el código de este problema en el siguiente enlace. Los resultados obtenidos fueron los siguientes:

Cantidad (hl)	P1	P2
M1	0	0
M2	0	150
Aceite	440	171
Solvente	55	36
Secador	55	63

Costo mínimo: 3067.8

Cantidad de iteraciones = 8

Tiempo = 0.002

2

Para la resolución de este ejercicio se empleó la biblioteca de python **Scipy**, con su módulo *optimize*. En este módulo se puede encontrar la función *minimize*, que fue la utilizada en este caso, pues permite minimizar una función de varias variables. Esta función permite pasar como parámetro el solucionador elegido para solucionar el modelo; en este caso, como el modelo no presenta resticciones, se utilizó el método Nelder-Mead que usa en su funcionamiento el algoritmo Simplex.

Es sencillo determinar que el valor mínimo que puede alcanzar esta función es 0; lo cual ocurre cuando $x_i = 1, \forall i = 1 \ldots n$. Puede encontrar el código de este ejercicio a través de este enlace. Para cada n se obtuvieron los siguientes resultados:

$$
\begin{aligned}
n = 2 \Rightarrow & Valor\ de\ la\ funcion : 0.000000 \\
& Iteraciones : 79 \\
& Cantidad\ de\ evaluaciones\ de\ la\ funcion : 150 \\
& Valor\ de\ cada\ variable : [1.1.] \\
n = 3 \Rightarrow & Valor\ de\ la\ funcion : 0.000000 \\
& Iteraciones : 150 \\
& Cantidad\ de\ evaluaciones\ de\ la\ funcion : 270 \\
& Valor\ de\ cada\ variable : [1.1.1.] \\
n = 4 \Rightarrow & Valor\ de\ la\ funcion : 0.000000 \\
& Iteraciones : 235 \\
& Cantidad\ de\ evaluaciones\ de\ la\ funcion : 411 \\
& Valor\ de\ cada\ variable : [1.1.1.1.] \\
n = 5 \Rightarrow & Valor\ de\ la\ funcion : 0.000000 \\
& Iteraciones : 339 \\
& Cantidad\ de\ evaluaciones\ de\ la\ funcion : 571 \\
& Valor\ de\ cada\ variable : [1.1.1.1.1.]
\end{aligned}
$$

3

Para la solución de este ejercicio se utilizó la biblioteca **Gekko**, un paquete de Python para *machine learning* y optimización. Además viene aparejada de solucionadores a gran escala para programación lineal, no lineal, cuadrática, etc.

El código empleado para su solución se puede encontrar a través de este enlace. Los resultados obtenidos fueron los siguientes:

$$
\begin{aligned}
x_0 &= 0.50220271892 \\
x_1 &= 0.24889864054 \\
Objetivo &: 0.24889703506
\end{aligned}
$$

Cantidad de iteraciones realizadas: 11

Tiempo de solución: 2.730000000155997E-002.