Lab05

Javier Ortín

2026-02-16

La función lp$

Minimizar $-2x_2 + x_3$ sujeto a:

$$\begin{cases} -x_1 - 2x_2 & \geq -3 \\ 4x_1 + x_2 + 7x_3 & \geq -1 \\ 2x_1 - 2x_2 + x_3 & \geq -5 \\ x_i & \geq 0 \quad \text{con } i \in \{1,2,3\} \end{cases}$$

```r
library(lpSolve)
coef = c(0,-2,1) # función objetico

A = rbind(c(-1,-2,0),
          c(4,1,7),
          c(2,-3,1))

dir = rep(">=", 3) # las restricciones de xi positivas se dan por hecho
b = c(-3,-1,-5)

sol = lp("min", coef, A, dir, b)
sol
```

```
## Success: the objective function is -3
```

```r
sol$solution
```

```
## [1] 0.0 1.5 0.0
```

¿Y si queremos resolverlo para $x_3 \in \mathbb{R}$ cualquiera? Podemos reformular el problema para que dependa únicamente de variables no negativas. Así, podemos descomponer como $x_3 = x_{3,1} - x_{3_2}$ con $x_{3,1}, x_{3,2} \geq 0$. Este razonamiento puede ser extendido para todas las variables.

```r
coef2= c(0,0,-2,2,1,-1)
A2 = rbind(c(-1,1,-2,2,0,0),
           c(4,-4,1,-1,7,-7),
           c(2,-2,-3,3,1,-1))
sol2 = lp("min", coef2, A2, dir, b)
sol2
```

```
## Success: the objective function is -3.380952
```

```r
point = sol2$solution
point # En mayor dimensión
```

```
## [1] 0.00000000 0.04761905 1.52380952 0.00000000 0.00000000 0.33333333
```

```r
# Deshacemos el aumento de dimensión
c(point[1]-point[2],
  point[3]-point[4],
  point[5]-point[6])
```

```
## [1] -0.04761905  1.52380952 -0.33333333
```

Formulación y modelización

Enunciado

Unos grandes almacenes desean liquidar 200 camisas y 100 pantalones de la temporada anterior. Para ello, lanzan dos ofertas: A y B. La oferta A consiste en un lote de una camisa y un pantalón, que se venden a 30€. La oferta B consiste en un lote de tres camisas y un pantalón, que se vende a 50€. No se desea ofrecer menos de 20 lotes de A ni menos de 10 de B. ¿Cuántos lotes ha de vender de cada tipo para maximizar la ganancia?

Solución

Es imprescindible comenzar declarando qué significan nuestras variables de decisión: -x_A: Lotes vendidos del tipo A -x_B: Lotes vendidos del tipo B

Nuestra función objetivo es la de ganancia, que viene dada por:

$$30x_A + 50x_B$$

Ahora veamos qué restricciones hemos de considerar, dadas por las prendas disponibles:

$$\begin{cases} x_A + 3x_B & \leq 200 \\ x_A + x_B & \leq 100 \\ x_A & \geq 20 \\ x_B & \geq 10 \\ x_A, x_B \in \mathbb{Z}^+ \end{cases}$$

```r
coef = c(30,50)
A = rbind(c(1,0),
          c(0,1),
          c(1,3),
          c(1,1))

dir = c(">=", ">=", "<=", "<=")

b = c(20,10,200,100)
sol = lp("max", coef, A, dir, b)
sol
```

```
## Success: the objective function is 4000
```

```r
sol$solution
```

```
## [1] 50 50
```

```r
# ¿Sobran camisas?
A%*%sol$solution
```

```
##      [,1]
## [1,]   50
## [2,]   50
```

```
## [3,]  200
## [4,]  100
```

Otro problema

Enunciado

Un fabricante de muebles…

Solución

En este caso, el problema es de probramación entera.

-x_1: Unidades fabricadas del modelo I -x_2: Unidades fabricadas del modelo II

Función objetivo (a maximizar) dada por la ganancia:

$$120x_1 + 80x_2$$

Restricciones:

$$\begin{cases} 2x_1 + x_2 \leq 6 \\ 7x_1 + 8x_2 \leq 28 \\ x_1, x_2 \geq 0 \end{cases}$$

Las restricciones de no negatividad quedan implícitas en el código, pero hay que incluirlas en el planteamiento.

```r
coef = c(120,80)

A = rbind(c(2,1),
          c(7,8))

dir = rep("<=", 2)
b = c(6,28)
sol=lp("max",coef,A,dir,b)
sol
```

```
## Success: the objective function is 391.1111
```

```r
sol$solution
```

```
## [1] 2.222222 1.555556
```

En principio, la solución no tiene sentido, pues no podemos fabricar fracciones de biombos. Así, es necesario trabajar con programación entera.

Para modelar este problema con R, podemos usar el parámetro `int.vec`$ de la función *lp*, que guarda qué variables son enteras. En lugar de concatenar `true`$ y `false`, *podemospasarunvectordeíndices.Porejemplo, silasvariables1y3sonl* Si todas las variables son enteras, podemos usar `all.int=TRUE`$ en lugar de usar un vector de índices que los contenga todos.

Podemos seguir este razonamiento con `bin.vec`$ y `all.bin`$ para la programación binaria.

```r
sol=lp("max",coef,A,dir,b, all.int=TRUE)
sol
```

```
## Success: the objective function is 360
```

```r
sol$solution
```

```
## [1] 3 0
```

Ejercicio 4

Enunciado

Una oficina de correos. . .

Resolución

Podemos modelar el problema con siete variables de decisión: -x_1: Trabajadores que empiezan a trabajar el lunes -x_2: Trabajadores que empiezan a trabajar el martes . . . -x_7: Trabajadores que empiezan a trabajar el domingo

Evidentemente, las variables son enteras y no negativas.

La función objetivo (a minimizar) es el número de trabajadores totales:

$$x_1 + x_2 + \ldots + x_7$$

Sujeto a:

$$\begin{cases} x_1 + x_4 + x_5 + x_6 + x_7 \geq 17 \\ x_2 + x_1 + x_7 + x_6 + x_5 \geq 13 \\ x_3 + x_2 + x_1 + x_7 + x_6 \geq 11 \\ x_4 + x_3 + x_2 + x_1 + x_7 \geq 17 \\ x_5 + x_4 + x_3 + x_2 + x_1 \geq 17 \\ x_6 + x_5 + x_4 + x_3 + x_2 \geq 17 \\ x_7 + x_6 + x_5 + x_4 + x_3 \geq 17 \end{cases}$$

```r
coef = rep(1,7)
A = rbind(c(1,0,0,1,1,1,1),
          c(1,1,0,0,1,1,1),
          c(1,1,1,0,0,1,1),
          c(1,1,1,1,0,0,1),
          c(1,1,1,1,1,0,0),
          c(0,1,1,1,1,1,0),
          c(0,0,1,1,1,1,1))

b = c(17,13,15,19,14,16,11)
dir = rep(">=", 7)

sol = lp("min", coef, A, dir, b, all.int=TRUE)
sol
```

```
## Success: the objective function is 23
```

```r
sol$solution
```

```
## [1] 7 5 1 6 2 2 0
```

```r
sum(sol$solution)
```

```
## [1] 23
```

¿Sobran trabajadores algún día? Podemos comprobar el valor de `sol$solution` con el de b. Alternativamente, podemos comprobarlo con "==" en lugar de "<=" al imponer las restricciones en dir. Como podemos comprobar, no es posible hacerlo sin trabajadores sobrantes.