ProgM Lab04

Javier Ortín

2026-02-09

Continuación de solución gráfica

Maximizar $20x_1 + 60x_2$ sujeto a:

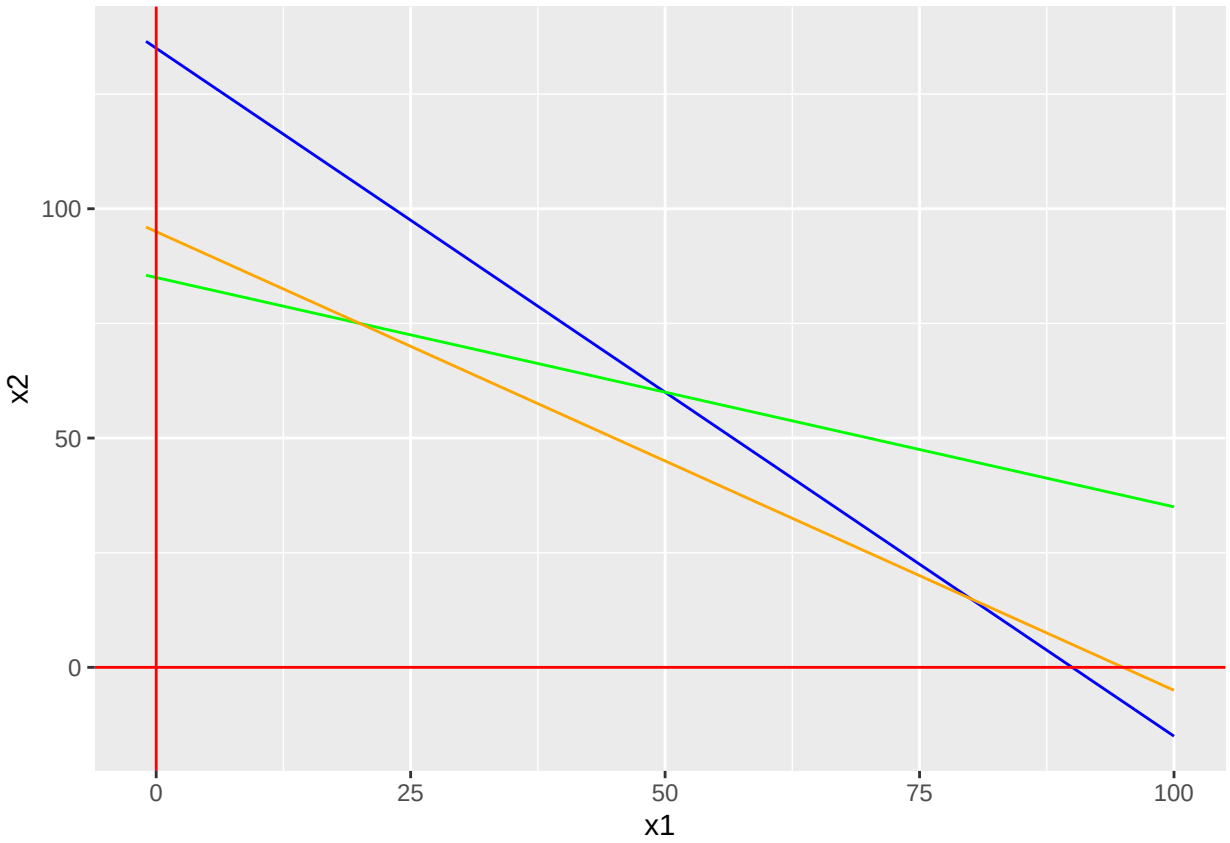
$$\begin{cases} 30x_1 + 20x_2 \leq 2700 \\ 5x_1 + 10x_2 \leq 850 \\ x_1 + x_2 \geq 95 \\ x_1, x_2 \geq 0 \end{cases}$$

```
# Restricciones
R1 = function(x1) 135-1.5*x1
R2 = function(x1) 85-0.5*x1
R3 = function(x1) 95-x1

x1 = seq(-1,100, length.out=100)
datos = data.frame(x1,
                   x2=R1(x1),
                   x2.2=R2(x1),
                   x2.3=R3(x1))
```

```
library(ggplot2)
```

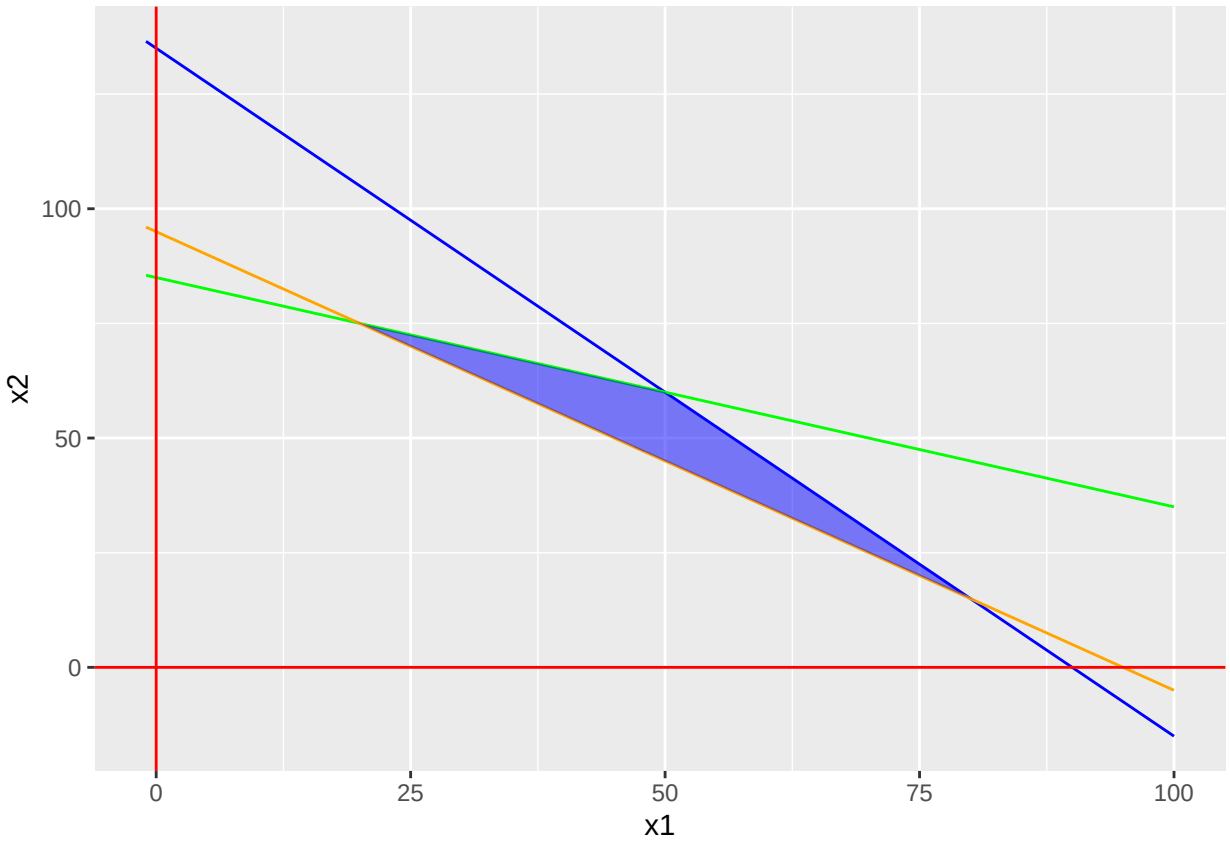
```
## Warning: package 'ggplot2' was built under R version 4.4.3
```

```
p = ggplot(datos, aes(x=x1)) +
  geom_line(aes(y=x2), col="blue") +
  geom_line(aes(y=x2.2), col="green") +
  geom_line(aes(y=x2.3), col="orange") +
  geom_vline(xintercept = 0, col="red") +
  geom_hline(yintercept = 0, col="red")

p
```



Calculamos ahora la región crítica a partir de las restricciones.

```
datos = transform(datos,
                  z=pmax(x2.3,pmin(x2,x2.2)))

p = p + geom_ribbon(data=datos,
                    aes(ymin=x2.3,ymax=z),
                    fill="blue", alpha=0.5)
p
```

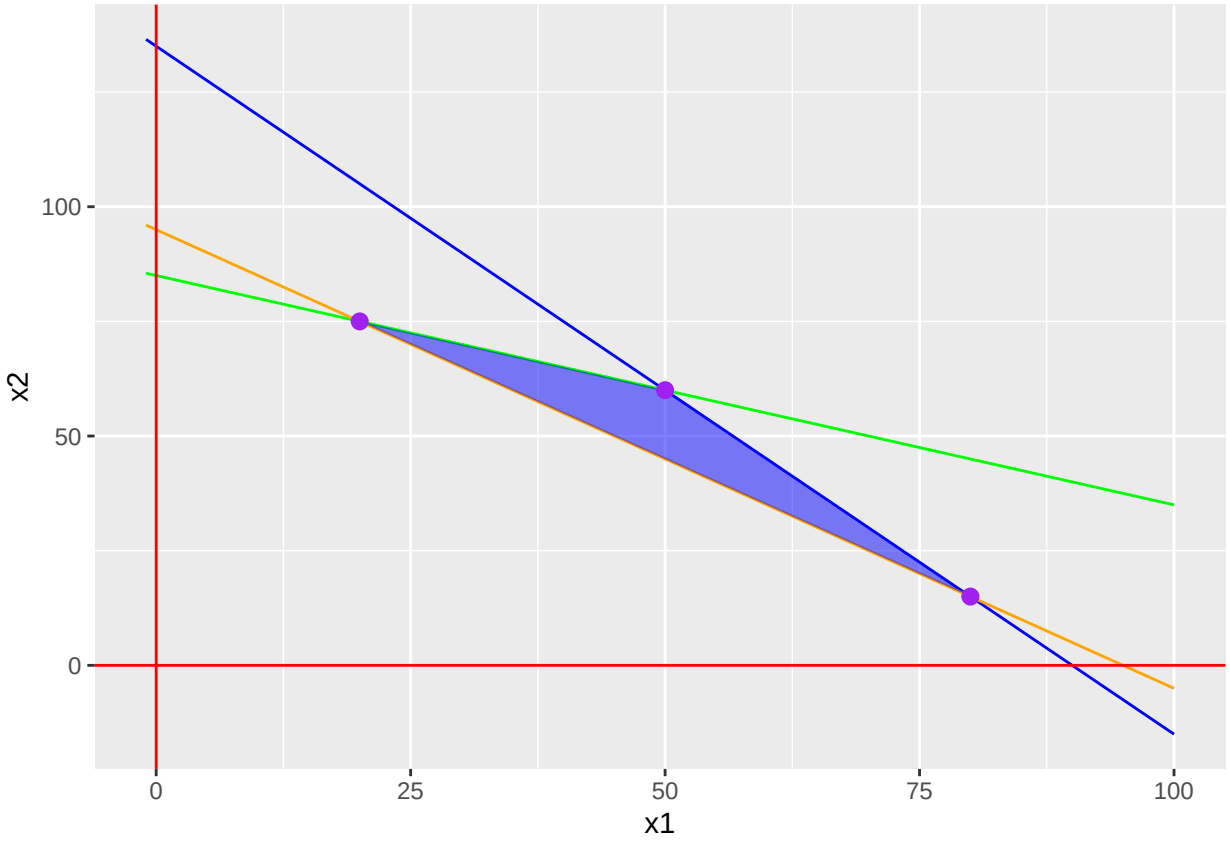


Hallemos los extremos del politopo:

```
A = rbind(c(30,20),
          c(5,10),
          c(1,1))

b = c(2700,850,95)

pto.R1R2 = solve(A[1:2,], b[1:2])
pto.R1R3 = solve(A[c(1,3),], b[c(1,3)])
pto.R2R3 = solve(A[2:3,], b[2:3])

puntos = rbind(pto.R1R2, pto.R1R3, pto.R2R3)
puntos = as.data.frame(puntos)
puntos #podemos cambiar los nombres pero no es necesario
```

```
##          V1 V2
## pto.R1R2 50 60
## pto.R1R3 80 15
## pto.R2R3 20 75
```

```
p = p + geom_point(data=puntos,
                   aes(x=V1, y=V2),
                   col="purple", size=2.5)
p
```



Veamos ahora cuál de ellos es el máximo.

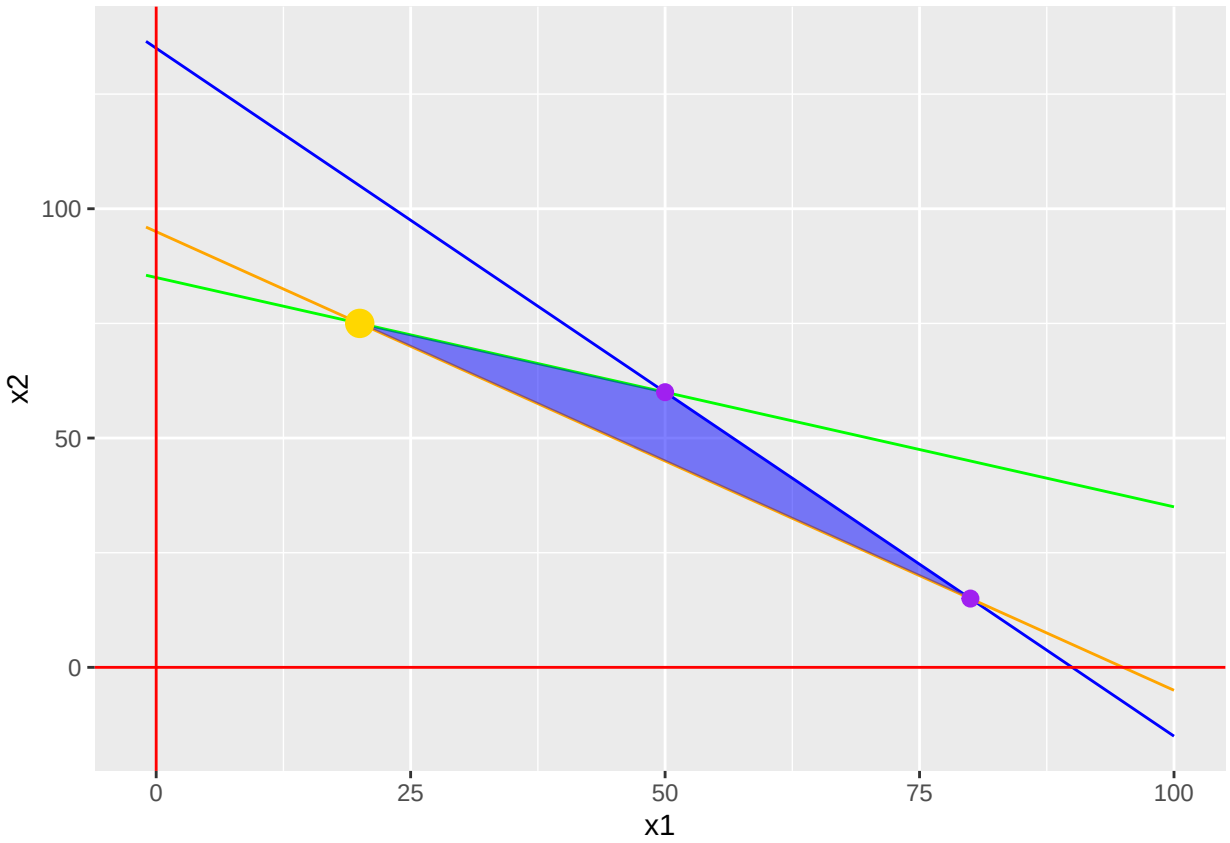
```
f.obj = function(x) 20*x[1] + 60*x[2]
aux = f.obj(puntos)
aux # arrastra el nombre de la variable de puntos
```

```
##            V1
## pto.R1R2 4600
## pto.R1R3 2500
## pto.R2R3 4900
```

```
puntos[which.max(aux$V1),]
```

```
##          V1 V2
## pto.R2R3 20 75
```

```
p + geom_point(data= puntos[which.max(aux$V1),],
               aes(x=V1, y=V2),
               col="gold", size=4.5)
```



Más allá de dos variables de decisión

Algoritmo simplex

R tiene dos librerías que implementan este algoritmo:

- `boot` tiene `simplex`
- `lpsolve` tiene `lp`

La primera opción tiene un rendimiento malo, luego no la veremos en clase. En su lugar, usaremos `lpsolve`.

Maximizar $x_1 + 9x_2 + 3x_3$ sujeto a:

$$\begin{cases} x_1 + 2x_2 + 3x_3 \geq 1 \\ 3x_1 + 2x_2 + 2x_3 \leq 15 \\ x_1, x_2, x_3 \geq 0 \end{cases}$$

Se asume que las variables son no negativas, luego esta última restricción viene de manera implícita.

```
coef = c(1,9,3) # función objetivo

A = rbind(c(1,2,3),
          c(3,2,2)) # restricciones explícitas

dir = c(">=", "<=") # sentido de las restricciones

b = c(1,15) # vector del lado derecho
```

```r
sol = lp("max",coef,A,dir,b) #guardamos la solución
sol
```

```
## Success: the objective function is 67.5
```

```r
sol$status
```

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

```r
sol$solution
```

```
## [1] 0.0 7.5 0.0
```

```r
sol$objval
```

```
## [1] 67.5
```

```r
?lp.object # más información en el menú de ayuda
```

```
## starting httpd help server ... done
```

```r
# También podemos pulsar "view" en el panel "Data" de RStudio
```

Cuando se marca como "éxito", se sabe que existe una solución óptima. No obstante, puede no ser única. Ninguno de los algoritmos nos garantiza la unicidad de la solución (en caso de haberla). La librería por defecto minimiza.