Assignment 1 - Linear Programming

Contents

The problem . . . 1
More data . . . 1
The decision variables . . . 2
The objective function . . . 2
The constraints . . . 2
Building the model . . . 3
Sensitivity analysis . . . 6
Questions about LP **9**

The problem

A construction materials company is looking for a way to maximize profit per transportation of their goods. The company has a train available with 4 wagons.

When stocking the wagons they can choose among 3 types of cargo, each with its own specifications. How much of each cargo type should be loaded on which wagon in order to maximize profit?

More data

TRAIN WAGON j	WEIGHT CAPACITY (TONNE) w_j	VOLUME CAPACITY (m^2) s_j
(wag) 1	10	5000
(wag) 2	8	4000
(wag) 3	12	8000
(wag) 4	6	2500

CARGO TYPE i	AVAILABLE (TONNE) a_i	VOLUME (m^2/t) v_i	PROFIT (PER TONNE) p_i
(cg) 1	20	500	3500
(cg) 2	10	300	2500
(cg) 3	18	400	2000

The decision variables

Define the decision variables for the problem described above.

X_{ij}, where i = cargo and j = train wagon.

In particular, we have 12 decision variables (3 cargos * 4 train wagons).

E.g. X_{23} represents the number of tonnes from the cargo type 2 that goes into the train wagon 3.

The objective function

Define the objective function for the problem described above.

We have to maximize the profit, associated with the tonnes:

$$\max(3500(X_{11}+X_{12}+X_{13}+X_{14})+2500(X_{21}+X_{22}+X_{23}+X_{24})+2000(X_{31}+X_{32}+X_{33}+X_{34}))$$

The constraints

Define the constraints for the problem described above.

We have three main constraints, i.e.:

- The number of available tonnes from each cargo.
- The weight capacity of each train wagon.
- The volume capacity of each train wagon.

In conclusion:

1)

$$\sum_j X_{1j} \leq 20, \sum_j X_{2j} \leq 10, \sum_j X_{3j} \leq 18$$

2)

$$\sum_i X_{i1} \leq 10, \sum_i X_{i2} \leq 8, \sum_i X_{i3} \leq 12, \sum_i X_{i4} \leq 6$$

3)

$$500X_{11}+300X_{21}+400X_{31} \leq 5000$$

$$500X_{12}+300X_{22}+400X_{32} \leq 4000$$

$$500X_{13} + 300X_{23} + 400X_{33} \leq 8000$$

$$500X_{14} + 300X_{24} + 400X_{34} \leq 2500$$

Finally, let's put a lower bound:

$$X_{ij} \geq 0$$

for each i, j.

Building the model

Build and solve the model with a suitable solver. You might want to use the `lpSolveAPI` library.

```
library(lpSolveAPI)


# Create model

model <- make.lp(0,12)


# Name the model

name.lp(model, "Cargo and train")


# Define objective function

lp.control(model, sense = "max")
```

```
## $anti.degen
## [1] "fixedvars" "stalling"
##
## $basis.crash
## [1] "none"
##
## $bb.depthlimit
## [1] -50
##
## $bb.floorfirst
## [1] "automatic"
##
## $bb.rule
## [1] "pseudononint" "greedy"       "dynamic"      "rcostfixing"
##
## $break.at.first
## [1] FALSE
##
## $break.at.value
```

```
## [1] 1e+30
##
## $epsilon
##       epsb       epsd      epsel     epsint epsperturb   epspivot
##      1e-10      1e-09      1e-12      1e-07      1e-05      2e-07
##
## $improve
## [1] "dualfeas" "thetagap"
##
## $infinite
## [1] 1e+30
##
## $maxpivot
## [1] 250
##
## $mip.gap
## absolute relative
##    1e-11    1e-11
##
## $negrange
## [1] -1e+06
##
## $obj.in.basis
## [1] TRUE
##
## $pivoting
## [1] "devex"    "adaptive"
##
## $presolve
## [1] "none"
##
## $scalelimit
## [1] 5
##
## $scaling
## [1] "geometric"   "equilibrate" "integers"
##
## $sense
## [1] "maximize"
##
## $simplextype
## [1] "dual"   "primal"
##
## $timeout
## [1] 0
##
## $verbose
## [1] "neutral"
```

```r
set.objfn(model, c(3500,3500,3500,3500,2500,2500,2500,2500,2000,2000,2000,2000))


# Add constraints
```

```
add.constraint(model,
               xt = c(1,1,1,1),
               type = "<=",
               rhs = 20,
               indices = c(1,2,3,4))
add.constraint(model,
               xt = c(1,1,1,1),
               type = "<=",
               rhs = 10,
               indices = c(5,6,7,8))
add.constraint(model,
               xt = c(1,1,1,1),
               type = "<=",
               rhs = 18,
               indices = c(9,10,11,12))

add.constraint(model,
               xt = c(1,1,1),
               type = "<=",
               rhs = 10,
               indices = c(1,5,9))
add.constraint(model,
               xt = c(1,1,1),
               type = "<=",
               rhs = 8,
               indices = c(2,6,10))
add.constraint(model,
               xt = c(1,1,1),
               type = "<=",
               rhs = 12,
               indices = c(3,7,11))
add.constraint(model,
               xt = c(1,1,1),
               type = "<=",
               rhs = 6,
               indices = c(4,8,12))

add.constraint(model,
               xt = c(500,300,400),
               type = "<=",
               rhs = 5000,
               indices = c(1,5,9))
add.constraint(model,
               xt = c(500,300,400),
               type = "<=",
               rhs = 4000,
               indices = c(2,6,10))
add.constraint(model,
               xt = c(500,300,400),
               type = "<=",
               rhs = 8000,
               indices = c(3,7,11))
add.constraint(model,
```

```r
               xt = c(500,300,400),
               type = "<=",
               rhs = 2500,
               indices = c(4,8,12))


# Define boundaries

set.bounds(model, lower = c(0,0,0,0,0,0,0,0,0,0,0,0))


# Solve model

solve(model)
```

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

```r
# Explore outcomes

print.lpExtPtr(model)
```

```
## Model name: Cargo and train
##   a linear program with 12 decision variables and 11 constraints
```

```r
get.variables(model)
```

```
##  [1] 10  8  2  0  0  0 10  0  0  0  0  6
```

```r
get.objective(model)
```

```
## [1] 107000
```

```r
get.constraints(model)
```

```
##  [1]   20   10    6   10    8   12    6 5000 4000 4000 2400
```

```r
get.primal.solution(model) # [obj_value, constraints_values, decision_variables]
```

```
##  [1] 107000     20     10      6     10      8     12      6   5000   4000
## [11]   4000   2400     10      8      2      0      0      0     10      0
## [21]      0      0      0      6
```

```r
get.basis(model, nonbasic = F) # optimal basis
```

```
##  [1] -14 -19  -3 -16 -17 -18 -23 -12 -13 -10 -11
```

Sensitivity analysis

Perform the sensitivity analysis for the model solved.

```r
# RHS
printSensitivityRHS <- function(model){
  options(scipen = 999)
  arg.rhs <- get.sensitivity.rhs(model)
  numRows <- length(arg.rhs$duals)
  symb <- c()
  for (i in c(1:numRows)) symb[i] <- paste("B", i, sep = "")

  rhs <- data.frame(rhs = symb, arg.rhs)

  rhs <- rhs %>%
    mutate(dualsfrom=replace(dualsfrom, dualsfrom < -1.0e4, "-inf")) %>%
    mutate(dualstill=replace(dualstill, dualstill > 1.0e4, "inf")) %>%
    unite(col = "Sensitivity",
          dualsfrom,
          rhs,
          dualstill ,
          sep = " <= ", remove = FALSE) %>%
    select(c("rhs","Sensitivity"))
  colnames(rhs)[1] <- c('Rhs')
  print(rhs)
}

# Obj fun
printSensitivityObj <- function(model){
  options(scipen=999)
  arg.obj = get.sensitivity.obj(model)

  numRows <- length(arg.obj$objfrom)
  symb <- c()
  for (i in c(1:numRows)) symb[i] <- paste("C", i, sep = "" )

  obj <- data.frame(Objs = symb, arg.obj)

  obj<-
    obj %>%
    mutate(objfrom=replace(objfrom, objfrom < -1.0e4, "-inf")) %>%
    mutate(objtill=replace(objtill, objtill > 1.0e4, "inf")) %>%
    unite(col = "Sensitivity",
          objfrom, Objs, objtill ,
          sep = " <= ", remove = FALSE) %>%
    select(c("Objs","Sensitivity"))
  print(obj)
}

printSensitivityObj(model)
```

```
##   Objs                  Sensitivity
## 1   C1 3500 <= C1 <= 4833.33333333
## 2   C2 3500 <= C2 <= 4833.33333333
## 3   C3            3500 <= C3 <= 3500
## 4   C4            -inf <= C4 <= 3500
## 5   C5            -inf <= C5 <= 2500
```

```
## 6    C6             -inf <= C6 <= 2500
## 7    C7             2500 <= C7 <= 2500
## 8    C8             -inf <= C8 <= 2500
## 9    C9             -inf <= C9 <= 2000
## 10  C10            -inf <= C10 <= 2000
## 11  C11            -inf <= C11 <= 2000
## 12  C12            2000 <= C12 <= 2500
```

```r
printSensitivityRHS(model)
```

```
##    Rhs        Sensitivity
## 1   B1      20 <= B1 <= 26
## 2   B2      10 <= B2 <= 16
## 3   B3  -inf <= B3 <= inf
## 4   B4      10 <= B4 <= 10
## 5   B5        8 <= B5 <= 8
## 6   B6       6 <= B6 <= 12
## 7   B7     0 <= B7 <= 6.25
## 8   B8 3000 <= B8 <= 5000
## 9   B9 2400 <= B9 <= 4000
## 10 B10 -inf <= B10 <= inf
## 11 B11 -inf <= B11 <= inf
## 12 B12 -inf <= B12 <= inf
## 13 B13 -inf <= B13 <= inf
## 14 B14 -inf <= B14 <= inf
## 15 B15    -10 <= B15 <= 0
## 16 B16 -inf <= B16 <= inf
## 17 B17 -inf <= B17 <= inf
## 18 B18 -inf <= B18 <= inf
## 19 B19 -inf <= B19 <= inf
## 20 B20      0 <= B20 <= 0
## 21 B21      0 <= B21 <= 0
## 22 B22      0 <= B22 <= 6
## 23 B23 -inf <= B23 <= inf
```

```r
get.dual.solution(model) # dual solution
```

```
##  [1]    1 1500  500    0 2000 2000 2000 2000    0    0    0    0    0    0    0
## [16]    0    0    0    0    0    0    0    0    0
```

- (Assumiamo cambiamenti singoli dei coefficienti, i.e. uno varia e gli altri rimangono costanti).
- Le variabili C3 e C7 hanno un range di variazione pari a 0. Questo significa che una minima variazione del loro valore può significare un cambiamento della soluzione ottima.
- Un minimo cambiamento dei vincoli B4, B20 e B21 ha la possibilità di cambiare sia la soluzione ottima, sia la regione ammissibile.
- Tuttavia, dalla soluzione duale vediamo che gli unici vincoli non-binding sono B1, B2, B3, B5, B6, B7 e B8.
- Un cambiamento dei vincoli B5, B6, B7 e B8 (shadow price = 2000) comporterebbe la maggiore variazione della funzione obiettivo.

Questions about LP

1. Can an LP model have more than one optimal solution. Is it possible for an LP model to have exactly two optimal solutions? Why or why not?

   Un modello di PL può avere soluzioni ottime multiple. Prendendo in esame il caso a due variabili, questo è verificabile graficamente quando l'ottimo della curva di livello coincide, almeno in parte, con il confine di una regione identificata da un vincolo. Non è invece possibile avere 2 soluzioni ottime poichè, anche supponendo di trovare due punti, sarebbero ottime anche le soluzioni comprese tra questi due punti.

2. Are the following objective functions for an LP model equivalent? That is, if they are both used, one at a time, to solve a problem with exactly the same constraints, will the optimal values for x_1, x_2 and x_3 be the same in both cases? Why or why not?

$$\max 2x_1 + 3x_2 - x_3$$

$$\min -2x_1 - 3x_2 + x_3$$

Dato che

$$min(f(x)) = max(-f(x))$$

e viceversa, considerando $f(x) = 2x_1 + 3x_2 - x_3$, allora $-f(x) = -2x_1 - 3x_2 + x_3$ e le due espressioni risultano equivalenti. Di conseguenza, non cambiano né la funzione obiettivo, né la regione ammissibile identificata dai vincoli.

3. Which of the following constraints are not linear or cannot be included as a constraint in a linear programming problem?

a. $2x_1 + x_2 - 3x_3 \geq 50$

b. $2x_1 + \sqrt{x_2} \geq 60$

c. $4x_1 - \frac{1}{2}x_2 = 75$

d. $\frac{3x_1+2x_2x_1-3x_3}{x_1+x_2+x_3} \leq 0.9$

e. $3x_1^2 + 7x_2 \leq 45$

a. Sì

b. No

c. Sì

d. No

e. No