Lab03-22

Diego Fernández

7 de marzo de 2022

LECTURA DE DATOS

Escribir una función que lea los datos de un archivo y los organice en una lista.

```
setwd("C://Users//diego//OneDrive//Escritorio//Universidad//2º Curso//2//Investigacion Operativa//Labora

datos_a_lista <- function (archivo="datos"){

  datos <- read.table(archivo)

  print(datos)
  n <- ncol(datos)-2
  m <- nrow(datos)-1
  lista <- list(n=n, m=m)

  lista$A <- as.matrix(datos[1:m,1:n])
  lista$AI <- cbind(lista$A,diag(datos[1:m,n+1]))
  lista$b <- as.numeric(datos[1:m,n+2])
  lista$c <- (datos[m+1,1:n])
  #opt= óptimo: min o max
  lista$opt <- as.matrix(datos[m+1,n+1])
  lista$datos <- as.matrix(datos[m+1,n+2])


 return(lista)

}

datos_a_lista() -> problema
```

```
##   V1 V2 V3  V4    V5
## 1 -1  1 -1   0     6
## 2 -1  2 -1  -1     4
## 3  1  1  2   1     8
## 4 -5 -1 -2 min datos
```

Cálculo de soluciones básicas

Copiar aqui el código generado en el laboratorio anterior

```r
library("gtools")
```

```
## Warning: package 'gtools' was built under R version 4.0.5
```

```r
m <- 2
n <- 4
v <- c(2,1,1,8,1,1,2,4)
b <- c(6,4)
A <- matrix(v, nrow = 2, byrow = TRUE)
AI <- cbind(A, diag(m))




calcula_sb <- function(M,J,b){

aux <- M[ , J]
solucion <- rep(0, m+n)
if(det(aux) == 0){
  return(F)
  }
solucion[J] <- solve(aux, b)
return(solucion)
}
calcula_sb(AI, c(1,2), b)
```

```
## [1] 2 2 0 0 0 0
```

```r
ncom <- choose(m+n, m)
combinations(n+m, m) -> C


for (i in 1 : nrow(C)) {
sb <- calcula_sb(AI, C[i, ], b)
print(sb)
}
```

```
## [1] 2 2 0 0 0 0
## [1] 2.6666667 0.0000000 0.6666667 0.0000000 0.0000000 0.0000000
## [1] FALSE
## [1]  4  0  0  0 -2  0
## [1] 3 0 0 0 0 1
## [1]  0  8 -2  0  0  0
## [1] 0.0 2.0 0.0 0.5 0.0 0.0
## [1] 0 4 0 0 2 0
## [1]  0  6  0  0  0 -2
## [1] 0.0000000 0.0000000 0.6666667 0.6666667 0.0000000 0.0000000
## [1] 0 0 2 0 4 0
## [1]  0  0  6  0  0 -8
## [1]  0  0  0  1 -2  0
## [1] 0.00 0.00 0.00 0.75 0.00 1.00
## [1] 0 0 0 0 6 4
```

Usar la funcion “apply” para generar TODAS las soluciones básicas.

```
library("gtools")




calcula_sb <- function(J){

M<-problema$AI
b<-problema$b
n<-problema$n
m<-problema$m

  aux <- M[ , J]
  solucion <- rep(0, m+n)
  if(det(aux) == 0){ return(F) }

  solucion[J] <- solve(aux, b)

  return(solucion)
}


ncom <- choose(m+n, m)
combinations(n+m, m) -> C



COM<-combinations(problema$n+problema$m,problema$m)
apply(COM, 1, calcula_sb)
```

```
## [[1]]
## [1] -26  -2  18   0   0   0
##
## [[2]]
## [1] FALSE
##
## [[3]]
## [1] 1 7 0 0 9 0
##
## [[4]]
## [1] -8 -2  0  0  0 18
##
## [[5]]
## [1] FALSE
##
## [[6]]
## [1] -20   0  14   0   2   0
##
## [[7]]
## [1] FALSE
##
## [[8]]
```

```
## [1] FALSE
##
## [[9]]
## [1] FALSE
##
## [[10]]
## [1] -6  0  0  0  2 14
##
## [[11]]
## [1] FALSE
##
## [[12]]
## [1] 0.0000000 6.6666667 0.6666667 0.0000000 8.6666667 0.0000000
##
## [[13]]
## [1]  0 -2 -8  0  0 26
##
## [[14]]
## [1] FALSE
##
## [[15]]
## [1] FALSE
##
## [[16]]
## [1] 0 6 0 0 8 2
##
## [[17]]
## [1] FALSE
##
## [[18]]
## [1] FALSE
##
## [[19]]
## [1]  0  0 -6  0  2 20
##
## [[20]]
## [1] FALSE
```

Cálculo del optimo

A partir de la lista de puntos anterior, seleccionar las soluciones factibles básicas y, usando los costes, calcular el punto donde la función objetivo toma el valor optimo.

```
calcula_opt<-function(lista)
  if(min(lista$i)>0){

  }
```