Assignment-4

TarakRam Nunna

22/10/2021

Setting working directory

```
getwd()
```

```
## [1] "C:/Users/TARAKRAM/OneDrive/Desktop/QMM_code/Assignment-4"
```

```
setwd("C:/Users/TARAKRAM/OneDrive/Desktop/QMM_code/Assignment-4")
```

Load the "lpSolveAPI" package.

```
library(lpSolve)
library(lpSolveAPI)
```

This lp problem has 5 constraints, 6 decision variables and it has minimisation function

```
lprec <- make.lp(5,6)

# Set the objective function for the problem.
set.objfn(lprec, c(622,614,630,641,645,649))
# set the direction towards minimum
lp.control(lprec, sense = "min")
```

```
## $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] "minimize"
##
## $simplextype
## [1] "dual"   "primal"
##
## $timeout
## [1] 0
##
## $verbose
## [1] "neutral"
```

All constraints to be added into the program.

```
# Set the constraint values row by row

# Production Capacity Constraints:
set.row(lprec, 1, c(1,1,1), indices = c(1,2,3))
set.row(lprec, 2, c(1,1,1), indices = c(4,5,6))

# Warehouse Demand Constraints:
set.row(lprec, 3, c(1,1), indices = c(1,4))
set.row(lprec, 4, c(1,1), indices = c(2,5))
set.row(lprec, 5, c(1,1), indices = c(3,6))

# Set the right hand side values
rhs <- c(100,120,80,60,70)
set.rhs(lprec, rhs)

# Set the constraint type
set.constr.type(lprec, c("<=","<=","=","=","="))
```

For this problem, all values must be greater than 0.

```
# Set the boundary condiiton for the decision variables
set.bounds(lprec, lower = rep(0, 6))

# Set the names of the rows (constraints) and columns (decision variables)
lp.rownames <- c("Plant A Capacity", "Plant B Capacity", "Warehouse 1
Demand", "Warehouse 2 Demand", "Warehouse 3 Demand")
lp.colnames <- c("PlantA to Warehouse1", "PlantA to Warehouse2", "PlantA to
Warehouse3", "PlantB to Warehouse1", "PlantB to Warehouse2", "PlantB to
Warehouse3")
dimnames(lprec) <- list(lp.rownames, lp.colnames)

# Return the linear programming object to ensure the values are correct
lprec

## Model name:
##                      PlantA to Warehouse1  PlantA to Warehouse2  PlantA to
Warehouse3  PlantB to Warehouse1  PlantB to Warehouse2  PlantB to Warehouse3
## Minimize                              622                   614
630                   641                   645                   649
## Plant A Capacity                        1                     1
1                     0                     0                     0  <=  100
## Plant B Capacity                        0                     0
0                     1                     1                     1  <=  120
## Warehouse 1 Demand                      1                     0
0                     1                     0                     0   =   80
## Warehouse 2 Demand                      0                     1
0                     0                     1                     0   =   60
## Warehouse 3 Demand                      0                     0
```

```
1                    0                    0                    1   =   70
## Kind                              Std                  Std
Std                  Std                  Std                  Std
## Type                             Real                 Real
Real                 Real                 Real                 Real
## Upper                             Inf                  Inf
Inf                  Inf                  Inf                  Inf
## Lower                               0                    0
0                    0                    0                    0

# The model can also be saved to a file
write.lp(lprec, filename = "Assignment_4-1.lp", type = "lp")
```

The following code will now look for an optimal solution. If it returns a "0" value then that means the model has found an optimal solution.

```
# Solve the linear program
solve(lprec)

## [1] 0
```

The model returned a "0", so it has found an optimal solution to the problem.

#The function below will give the minimum value for the objective function.

```
# Review the objective function value
get.objective(lprec)

## [1] 132790
```

The minimum combined shipping and production costs will be $132,790 based on the given information and constraints.

Next, we will return the values of the decision variables to decide how many units should be produced and shipped from each plant.

```
# Get the optimum decision variable values
get.variables(lprec)

## [1]  0 60 40 80  0 30
```

PlantA ships 0 units to Warehouse1.

PlantA ships 60 units to Warehouse2.

PlantA ships 40 units to Warehouse3.

PlantB ships 80 units to Warehouse1.

PlantB ships 0 units to Warehouse2.

PlantB ships 30 units to Warehouse3.