Assignment 3

SABIHA MHATARNAIK

10/11/2021

Q1. Solve the problem using lpSolve or any other equivalent library in R. #Install lpSolveAPI package if not installed

```
#install.packages("lpSolveAPI")
```

```
#Now, load the library
library(lpSolveAPI)

#create an lp object named 'lprec' with 0 constraints and 9 decision variables
lprec <- make.lp(0,9)

#Now create the objective function. The default is a minimization problem.
set.objfn(lprec, c(420,420,420,
                   360,360,360,
                   300,300,300))

# As the default is a minimization problem, we change the direction to set maximization
lp.control(lprec,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"
```

```
#Add the 12 constraints based on the plant's number and products made on those plants.
add.constraint(lprec ,c(1,0,0,1,0,0,1,0,0), "<=", 750)
add.constraint(lprec ,c(0,1,0,0,1,0,0,1,0), "<=", 900)
add.constraint(lprec ,c(0,0,1,0,0,1,0,0,1), "<=", 450)

add.constraint(lprec ,c(20,0,0,15,0,0,12,0,0), "<=", 13000)
add.constraint(lprec ,c(0,20,0,0,15,0,0,12,0), "<=", 12000)
add.constraint(lprec ,c(0,0,20,0,0,15,0,0,12), "<=", 5000)
```

```
add.constraint(lprec ,c(1,1,1,0,0,0,0,0,0), "<=", 900)
add.constraint(lprec ,c(0,0,0,1,1,1,0,0,0), "<=", 1200)
add.constraint(lprec ,c(0,0,0,0,0,0,1,1,1), "<=", 750)

add.constraint(lprec ,c(900,-750,0,900,-750,0,900,-750,0), "=", 0)
add.constraint(lprec ,c(0,450,-900,0,450,-900,0,450,-900), "=", 0)
add.constraint(lprec,c(450,0,-750,450,0,-750,450,0,-750),"=",0)
```

Set bounds for variables.

Remember that all variables had to be non-negative. We don't need to do it here,as this is the default , we can set bounds explicitly.

```
#Set bounds for variables.
set.bounds(lprec ,lower =c(0,0,0,0,0,0,0,0,0),
           columns= c (1:9))  #Not really needed

# To identify  the variables and constraints, we can set variable names and constraint names.
RowNames <-c("P1ProductionCapacity","P2ProductionCapacity","P3ProductionCapacity",
             "P1StorageSpace","P2StorageSpace","P3StorageSpace",
             "SalesForecastLarge","SalesForecastMedium","SalesForecastSmall",
             "PercentCapacityP1andP2","PercentCapacityP2andP3","PercentCapacityP1andP3")

ColNames <- c("Plant1Large","Plant2Large","Plant3Large",
              "Plant1Medium","Plant2Medium","Plant3Medium",
              "Plant1Small","Plant2Small","Plant3Small")

dimnames(lprec)<- list (RowNames,ColNames)

#Now view the Model
lprec
```

```
## Model name:
##   a linear program with 9 decision variables and 12 constraints
```

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

Now we can solve the above LP Problem

```
solve(lprec)
```

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

The output above indicated that the answer is 0, means there was a successful solution. We now output the value of the objective function, and the variables.

```
get.objective(lprec)
```

```
## [1] 696000
```

```r
get.variables(lprec)
```

```
## [1] 516.6667   0.0000   0.0000 177.7778 666.6667   0.0000   0.0000 166.6667
## [9] 416.6667
```

From the above solution, we can infer the following : Plant 1 : 516.67 of Large Products and 177.78 of Medium Products. Plant 2 : 666.67 of Medium Products and 166.67 of Small products. Plant 3 : 416.67 of Small Products

```r
get.constraints(lprec)
```

```
##  [1]   694.4444   833.3333   416.6667 13000.0000 12000.0000  5000.0000
##  [7]   516.6667   844.4444   583.3333     0.0000     0.0000     0.0000
```

We now read the lp formulation using an lp file. I am using the same R file which I have saved.

```r
a <- read.lp ("weigelt.lp")  # create an lp object a
a                            #display a
```

```
## Model name:
##   a linear program with 9 decision variables and 12 constraints
```

Solve the lp model

```r
solve(a)                     #get objective value
```

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

```r
get.objective(a)             #get values of decision variables
```

```
## [1] 696000
```

```r
get.constraints(a)           #get constraint values
```

```
##  [1]   694.4444   833.3333   416.6667 13000.0000 12000.0000  5000.0000
##  [7]   516.6667   844.4444   583.3333     0.0000     0.0000     0.0000
```

```r
get.variables(lprec)
```

```
## [1] 516.6667   0.0000   0.0000 177.7778 666.6667   0.0000   0.0000 166.6667
## [9] 416.6667
```

Q2. Identify the shadow prices,dual solution and reduced costs.

```r
get.sensitivity.rhs(lprec) # get Shadow Prices
```

```
## $duals
##  [1]    0.00    0.00    0.00   12.00   20.00   60.00    0.00    0.00    0.00
## [10]   -0.08    0.00    0.56    0.00  -40.00 -360.00    0.00    0.00 -120.00
## [19]  -24.00    0.00    0.00
##
## $dualsfrom
##  [1] -1.000000e+30 -1.000000e+30 -1.000000e+30  1.122222e+04  1.150000e+04
##  [6]  4.800000e+03 -1.000000e+30 -1.000000e+30 -1.000000e+30  0.000000e+00
## [11] -1.000000e+30  0.000000e+00 -1.000000e+30 -1.000000e+02 -2.000000e+01
## [16] -1.000000e+30 -1.000000e+30 -4.444444e+01 -2.222222e+02 -1.000000e+30
## [21] -1.000000e+30
##
## $dualstill
##  [1] 1.000000e+30 1.000000e+30 1.000000e+30 1.388889e+04 1.250000e+04
##  [6] 5.181818e+03 1.000000e+30 1.000000e+30 1.000000e+30 0.000000e+00
## [11] 1.000000e+30 0.000000e+00 1.000000e+30 1.000000e+02 2.500000e+01
## [16] 1.000000e+30 1.000000e+30 6.666667e+01 1.111111e+02 1.000000e+30
## [21] 1.000000e+30
```

Note that the shadow prices are expressed here under $duals. We also have valid ranges for shadow price calculations. Those are given under $dualsfrom and $dualstill.

```
get.sensitivity.obj(lprec) # get Reduced Costs
```

```
## $objfrom
## [1]  3.60e+02 -1.00e+30 -1.00e+30  3.45e+02  3.45e+02 -1.00e+30 -1.00e+30
## [8]  2.52e+02  2.04e+02
##
## $objtill
## [1] 4.60e+02 4.60e+02 7.80e+02 4.20e+02 4.20e+02 4.80e+02 3.24e+02 3.24e+02
## [9] 1.00e+30
```

The reduced costs are expressed here until $objfrom and $objtill.

Dual solution

```
get.dual.solution(lprec )   #get dual Solution
```

```
##  [1]    1.00    0.00    0.00    0.00   12.00   20.00   60.00    0.00    0.00
## [10]    0.00   -0.08    0.00    0.56    0.00  -40.00 -360.00    0.00    0.00
## [19] -120.00  -24.00    0.00    0.00
```

Q3. Further,identify the sensitivity of the above prices and costs.That is,specify the range of shadow prices and reduced cost within which the optimal solution will not change

```
cbind( price=get.sensitivity.rhs(lprec)$duals[1:11],lowerRange=get.sensitivity.rhs(lprec)$dualsfrom[1:1
```

```
##       price    lowerRange   upperRange
##  [1,]  0.00 -1.000000e+30 1.000000e+30
##  [2,]  0.00 -1.000000e+30 1.000000e+30
##  [3,]  0.00 -1.000000e+30 1.000000e+30
##  [4,] 12.00  1.122222e+04 1.388889e+04
```

```
##   [5,] 20.00  1.150000e+04 1.250000e+04
##   [6,] 60.00  4.800000e+03 5.181818e+03
##   [7,]  0.00 -1.000000e+30 1.000000e+30
##   [8,]  0.00 -1.000000e+30 1.000000e+30
##   [9,]  0.00 -1.000000e+30 1.000000e+30
##  [10,] -0.08  0.000000e+00 0.000000e+00
##  [11,]  0.00 -1.000000e+30 1.000000e+30
```

```
cbind( cost=get.sensitivity.rhs(lprec)$duals[12:21],lowerRange=get.sensitivity.rhs(lprec)$dualsfrom[12:2
```

```
##           cost    lowerRange   upperRange
##   [1,]    0.56  0.000000e+00 0.000000e+00
##   [2,]    0.00 -1.000000e+30 1.000000e+30
##   [3,]  -40.00 -1.000000e+02 1.000000e+02
##   [4,] -360.00 -2.000000e+01 2.500000e+01
##   [5,]    0.00 -1.000000e+30 1.000000e+30
##   [6,]    0.00 -1.000000e+30 1.000000e+30
##   [7,] -120.00 -4.444444e+01 6.666667e+01
##   [8,]  -24.00 -2.222222e+02 1.111111e+02
##   [9,]    0.00 -1.000000e+30 1.000000e+30
##  [10,]    0.00 -1.000000e+30 1.000000e+30
```

The shadow prices are expressed here comprising of $dualsfrom and $dualstill. The above is the range specified for the shadow prices within which the optimal solution will not change.

```
cbind(get.sensitivity.obj(lprec)$duals[1:9],lowerRange=get.sensitivity.obj(lprec)$objfrom[1:9],upperRan
```

```
##       lowerRange upperRange
##   [1,]   3.60e+02   4.60e+02
##   [2,]  -1.00e+30   4.60e+02
##   [3,]  -1.00e+30   7.80e+02
##   [4,]   3.45e+02   4.20e+02
##   [5,]   3.45e+02   4.20e+02
##   [6,]  -1.00e+30   4.80e+02
##   [7,]  -1.00e+30   3.24e+02
##   [8,]   2.52e+02   3.24e+02
##   [9,]   2.04e+02   1.00e+30
```

The reduced costs are expressed here until $objfrom and $objtill. The above is the range specified for the reduced cost within which the optimal solution will not change.

```
get.sensitivity.rhs(lprec)$duals
```

```
##  [1]    0.00    0.00    0.00   12.00   20.00   60.00    0.00    0.00    0.00
## [10]   -0.08    0.00    0.56    0.00  -40.00 -360.00    0.00    0.00 -120.00
## [19]  -24.00    0.00    0.00
```

Q4. Formulation of the dual of the above problem

```r
lpDual <- make.lp(0,12)
```

```r
set.objfn(lpDual, c(750,900,450,
                    13000,12000,5000,
                    900,1200,750,
                    0,0,0))
```

```r
lp.control(lpDual,sense='min',simplextype="dual")
```

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

```
add.constraint(lpDual ,c(1,0,0,20,0,0,1,0,0,900,0,450), ">=", 420)
add.constraint(lpDual ,c(0,1,0,0,20,0,1,0,0,-750,450,0), ">=", 420)
add.constraint(lpDual ,c(0,0,1,0,0,20,1,0,0,0,-900,-750), ">=", 420)

add.constraint(lpDual ,c(1,0,0,15,0,0,0,1,0,900,0,450), ">=", 360)
add.constraint(lpDual ,c(0,1,0,0,15,0,0,1,0,-750,450,0), ">=", 360)
add.constraint(lpDual ,c(0,0,1,0,0,15,0,1,0,0,-900,-750), ">=", 360)

add.constraint(lpDual ,c(1,0,0,12,0,0,0,0,1,900,0,450), ">=", 300)
add.constraint(lpDual ,c(0,1,0,0,12,0,0,0,1,-750,450,0), ">=", 300)
add.constraint(lpDual ,c(0,0,1,0,0,12,0,0,1,0,-900,-750), ">=", 300)
```

We now read the lp formulation using an lp file

```
lpDual <- read.lp("dual.lp") #Create an lp object lpDual
lpDual                       #Display lpDual
```

```
## Model name:
##   a linear program with 12 decision variables and 9 constraints
```

Solve the lp model

```
solve(lpDual)                          #get objective value
```

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

```r
get.objective(lpDual)             #get values of decision variables
```

```
## [1] 696000
```

```r
get.variables(lpDual)
```

```
##  [1]  0.0000000  0.0000000  0.0000000 12.0000000 20.0000000 60.0000000
##  [7]  0.0000000  0.0000000  0.0000000  0.0000000  0.4000000  0.1333333
```

```r
get.constraints(lpDual)           #get constraint values
```

```
## [1] 420 460 780 360 360 480 324 300 300
```

```r
get.sensitivity.rhs(lpDual)       #get shadow price
```

```
## $duals
##  [1]  5.166667e+02  0.000000e+00  0.000000e+00  1.777778e+02  6.666667e+02
##  [6]  0.000000e+00  0.000000e+00  1.666667e+02  4.166667e+02  5.555556e+01
## [11]  6.666667e+01  3.333333e+01  0.000000e+00  0.000000e+00  0.000000e+00
## [16]  3.833333e+02  3.555556e+02  1.666667e+02 -2.071882e-10  0.000000e+00
## [21]  0.000000e+00
##
## $dualsfrom
##  [1]  3.60e+02 -1.00e+30 -1.00e+30  3.45e+02  3.45e+02 -1.00e+30 -1.00e+30
##  [8]  2.88e+02  2.04e+02 -1.00e+30 -1.00e+30 -1.00e+30 -1.00e+30 -1.00e+30
## [15] -1.00e+30 -4.00e+01 -1.50e+01 -2.40e+01 -8.00e-02 -1.00e+30 -1.00e+30
##
## $dualstill
##  [1] 4.60e+02 1.00e+30 1.00e+30 4.20e+02 3.75e+02 1.00e+30 1.00e+30 3.24e+02
##  [9] 1.00e+30 1.80e+02 6.00e+01 4.80e+02 1.00e+30 1.00e+30 1.00e+30 6.00e+01
## [17] 1.50e+01 1.20e+01 2.00e-01 1.00e+30 1.00e+30
```

```r
get.sensitivity.obj(lpDual)       #get reduced cost
```

```
## $objfrom
##  [1]  6.944444e+02  8.333333e+02  4.166667e+02  1.122222e+04  1.150000e+04
##  [6]  4.800000e+03  5.166667e+02  8.444444e+02  5.833333e+02 -1.000000e+30
## [11] -2.000000e+04 -1.500000e+04
##
## $objtill
##  [1] 1.000000e+30 1.000000e+30 1.000000e+30 1.388889e+04 1.250000e+04
##  [6] 5.181818e+03 1.000000e+30 1.000000e+30 1.000000e+30 2.071882e-10
## [11] 2.500000e+04 1.500000e+04
```

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

```
##  [1]  1.000000e+00  5.166667e+02  0.000000e+00  0.000000e+00  1.777778e+02
##  [6]  6.666667e+02  0.000000e+00  0.000000e+00  1.666667e+02  4.166667e+02
## [11]  5.555556e+01  6.666667e+01  3.333333e+01  0.000000e+00  0.000000e+00
## [16]  0.000000e+00  3.833333e+02  3.555556e+02  1.666667e+02 -2.071882e-10
## [21]  0.000000e+00  0.000000e+00
```

Specifying the range

```
cbind( get.sensitivity.rhs(lpDual)$duals[1:21],lowerRange=get.sensitivity.rhs(lpDual)$dualsfrom[1:21],up
```

```
##                     lowerRange upperRange
##  [1,]  5.166667e+02   3.60e+02   4.60e+02
##  [2,]  0.000000e+00  -1.00e+30   1.00e+30
##  [3,]  0.000000e+00  -1.00e+30   1.00e+30
##  [4,]  1.777778e+02   3.45e+02   4.20e+02
##  [5,]  6.666667e+02   3.45e+02   3.75e+02
##  [6,]  0.000000e+00  -1.00e+30   1.00e+30
##  [7,]  0.000000e+00  -1.00e+30   1.00e+30
##  [8,]  1.666667e+02   2.88e+02   3.24e+02
##  [9,]  4.166667e+02   2.04e+02   1.00e+30
## [10,]  5.555556e+01  -1.00e+30   1.80e+02
## [11,]  6.666667e+01  -1.00e+30   6.00e+01
## [12,]  3.333333e+01  -1.00e+30   4.80e+02
## [13,]  0.000000e+00  -1.00e+30   1.00e+30
## [14,]  0.000000e+00  -1.00e+30   1.00e+30
## [15,]  0.000000e+00  -1.00e+30   1.00e+30
## [16,]  3.833333e+02  -4.00e+01   6.00e+01
## [17,]  3.555556e+02  -1.50e+01   1.50e+01
## [18,]  1.666667e+02  -2.40e+01   1.20e+01
## [19,] -2.071882e-10  -8.00e-02   2.00e-01
## [20,]  0.000000e+00  -1.00e+30   1.00e+30
## [21,]  0.000000e+00  -1.00e+30   1.00e+30
```

The shadow prices are expressed here comprising of $dualsfrom and $dualstill. The above is the range specified for the shadow prices within which the optimal solution will not change.

```
cbind(get.sensitivity.obj(lpDual)$duals[1:12],lowerRange=get.sensitivity.obj(lpDual)$objfrom[1:12],uppe
```

```
##          lowerRange   upperRange
##  [1,]  6.944444e+02 1.000000e+30
##  [2,]  8.333333e+02 1.000000e+30
##  [3,]  4.166667e+02 1.000000e+30
##  [4,]  1.122222e+04 1.388889e+04
##  [5,]  1.150000e+04 1.250000e+04
##  [6,]  4.800000e+03 5.181818e+03
##  [7,]  5.166667e+02 1.000000e+30
##  [8,]  8.444444e+02 1.000000e+30
##  [9,]  5.833333e+02 1.000000e+30
## [10,] -1.000000e+30 2.071882e-10
## [11,] -2.000000e+04 2.500000e+04
## [12,] -1.500000e+04 1.500000e+04
```

The reduced costs are expressed here until $objfrom and $objtill. The above is the range specified for the reduced cost within which the optimal solution will not change.

The Formulation of dual yields results that agree with primal solution.