Quantitive Management Modelling -Assignment 3 : Shadow Price , Reduced Cost

1. Importing Library

```
# Importing "lpSolveAPI" library
library(lpSolveAPI)
```

2. Assigning Constraints and decision Variables

```
# 1.Assigning number of constraints and decision variables
lprec <- make.lp(0, 9)
```

3. Setting up Objective function ,Constraints , decision variables

```
# 3.1 Objective Function
set.objfn(lprec, c(420, 360, 300, 420, 360, 300, 420, 360, 300))
lp.control(lprec,sense='max') # setting it to maximize(default is minimize)
```

```
## $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"
```

```
# 3.2 Providing values for each constraint

add.constraint(lprec, c(1, 1, 1), "<=", 750,indices =c(1,2,3) )
add.constraint(lprec, c(1, 1, 1), "<=", 900 , indices =c(4,5,6) )
add.constraint(lprec, c(1, 1, 1), "<=", 450,indices=c(7,8,9))
add.constraint(lprec, c(20, 15, 12), "<=", 13000,indices =c(1,2,3) )
add.constraint(lprec, c(20, 15, 12), "<=", 12000,indices = c(4,5,6))
add.constraint(lprec, c(20, 15, 12), "<=", 5000 , indices = c(7,8,9))
```

```r
add.constraint(lprec, c(1, 1, 1), "<=", 900, indices=c(1,2,3))
add.constraint(lprec, c(1, 1, 1), "<=", 1200,indices = c(4,5,6))
add.constraint(lprec, c(1, 1, 1), "<=", 750, indices = c(7,8,9))
add.constraint(lprec, c(900, 900, 900, -750, -750 ,-750), "=", 0,indices = c(1,2,3,4,5,6))
add.constraint(lprec, c(450, 450, 450, -900, -900, -900), "=", 0, indices = c(4,5,6,7,8,9))


# 3.3 Naming the decision variables (column) and constraints (rows)

rownames <- c("Plant1_Production", "Plant2_Production", "Plant3_Production", "Plant1_Storage_Space", "Pl

colnames <- c("Plant1_Large", "Plant1_Medium", "Plant1_Small","Plant2_Large", "Plant2_Medium", "Plant2_S

dimnames(lprec) <- list(rownames,colnames)
```

4. View the linear program external pointer

```r
lprec
```

```
## Model name:
##   a linear program with 9 decision variables and 11 constraints
```

5. Save the LP model into a file

```r
write.lp(lprec, filename = "weiglet_Assignment3_Shadow_Reduced_Price.lp", type = "lp")


weiglet <- read.lp("weiglet_Assignment3_Shadow_Reduced_Price.lp") # create an lp object weiglet
```

6. Reading from file and solving it

```r
solve(weiglet)                    # Solution
```

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

```r
get.objective(weiglet)            # get objective value
```

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

```r
get.variables(weiglet)            # get values of decision variables
```

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

```
get.constraints(weiglet)         # get constraints
```

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

7. Identifying Shadow price :

```
#options(scipen = 0)
get.sensitivity.rhs(weiglet)  # get shadow prices
```

```
## $duals
##  [1]    0.0000000    0.0000000    0.0000000   12.0000000   20.0000000
##  [6]   60.0000000    0.0000000    0.0000000    0.0000000    0.2000000
## [11]    0.4666667    0.0000000    0.0000000  -24.0000000  -40.0000000
## [16]    0.0000000    0.0000000 -360.0000000 -120.0000000    0.0000000
##
## $dualsfrom
##  [1] -1.000000e+30 -1.000000e+30 -1.000000e+30  1.041667e+04  1.000000e+04
##  [6]  4.800000e+03 -1.000000e+30 -1.000000e+30 -1.000000e+30 -4.000000e+04
## [11] -1.500000e+04 -1.000000e+30 -1.000000e+30 -8.611111e+02 -1.000000e+02
## [16] -1.000000e+30 -1.000000e+30 -5.000000e+01 -1.333333e+02 -1.000000e+30
##
## $dualstill
##  [1] 1.000000e+30 1.000000e+30 1.000000e+30 1.388889e+04 1.250000e+04
##  [6] 5.400000e+03 1.000000e+30 1.000000e+30 1.000000e+30 5.000000e+04
## [11] 3.000000e+04 1.000000e+30 1.000000e+30 1.111111e+02 2.500000e+02
## [16] 1.000000e+30 1.000000e+30 2.500000e+01 6.666667e+01 1.000000e+30
```

8. Identifying Reduced Cost:

```
get.sensitivity.obj(weiglet)  # get reduced cost
```

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

9. Results:

- Used **"lpSolveAPI"** library to solve the LP problem.
- Since, it is an maximization problem , we have determined the maximized profit.
- As per the above results:
- The maximum total net profit per day,Z= **696000**.
- **Identified Shadow Price** and **reduced cost**.
- Also determined the **ranges for shadow price** and **reduced costs** , such that within which the optimum solution will not change.