MIS-64018: Assignment 4

Steve Spence

9/30/2019

First, we will load the "lpSolve" and "lpSolveAPI" to solve the problem outlined in parts A and B. This will load the appropriate tools needed to solve this problem.

```
# Load the "lpSolveAPI" package into the R environment.

library(lpSolve)
library(lpSolveAPI)
```

Next, we need to create a linear programming object. Based on our linear programming setup, this program should have 6 decision variables, 5 constraints, and minimum boundary conditions for each variable.

```
# Begin the lp problem with 5 constraints and 6 decision variables.

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

Then, we must set the objective function into our program. As a default, the program sets the objective function to find the minimum.

```
# Set the objective function for the problem.

set.objfn(lprec, c(622,614,630,641,645,649))

# Confirm the direction is set to minimization

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 of the constraint values will also need to be added into the program. These are the production capacity and warehouse demand in this problem. These will be input using the "set.row" command and defining which index to put the values at.

```
# 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))
```

Now, we will need to set the constraint values from the problem statement. In this case, these values correspond to production capacity by plant and demand requirements at each warehouse.

```
# Set the right hand side values

rhs <- c(100,120,80,60,70)
set.rhs(lprec, rhs)
```

Next, we need to define the inequality values for the problem. Since the capacity exceeds the demand in this case, we do not require dummy variables. Therefore, we can place less than inequalities for the plant production capacities and equal signs for the warehouse demands.

```
# 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))
```

This set of code will name the decision variables and the constraints for the model.

```
# 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("PA to W1", "PA to W2", "PA to W3", "PB to W1", "PB to W2",
```

```
"PB to W3")
dimnames(lprec) <- list(lp.rownames, lp.colnames)
```

This command should return the linear program outline before executing the code, so we can determine if all the values are correct.

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

lprec
```

```
## Model name:
##                      PA to W1  PA to W2  PA to W3  PB to W1  PB to W2  PB
to W3
## 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 return what the minimum value for the objective function will be.

```
# 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
```

Optimum decision variable values based on the lpSolveAPI output is:

Plant A Units Shipped to Warehouse 1: 0 units Plant A Units Shipped to Warehouse 2: 60 units Plant A Units Shipped to Warehouse 3: 40 units Plant B Units Shipped to Warehouse 1: 80 units Plant B Units Shipped to Warehouse 2: 0 units Plant B Units Shipped to Warehouse 3: 30 units