MSDS650 Week 8 Linear Programming Assignment - Nathan Worsham

I first tried doing the python excercise with the happiness based on wedding guest seatings, but I was having trouble following the logic of the excercise. This might partially be because the python code made frequent use of list comprehension which can make python code tough to read, also it wasn't very clear what was trying to be accomplished. So instead I worked on the R excercise with a farmer trying to maximize profit from 2 different crops. The problem was very clear and the result we were after was also very clear. I first needed to install the package `lpSolveAPI` and then load the package:

```
> install.packages("lpSolveAPI")
trying URL 'http://cran.rstudio.com/bin/macosx/mavericks/contrib/3.2/
lpSolveAPI_5.5.2.0-14.tgz'
Content type 'application/x-gzip' length 1311753 bytes (1.3 MB)
==================================================
downloaded 1.3 MB
The downloaded binary packages are in
/var/folders/8s/f9vh3ckj0m98v0ykc52r9lpr0000gp/T/
/RtmpaRwcQP/downloaded_packages
> library("lpSolveAPI")
```

Next we are instructed to create a linear program model object with 0 rows and 2 columns, and then set the paramaters to the model. The options for sense are "max" or "min" specifying whether the model is a maximization or a minimization problem, in this case it is a maximization problem, so we use "max".

```
> lprec <- make.lp(0, 2)
Warning messages:
1: In .HTMLsearch(query) : Unrecognized search field: title
2: In .HTMLsearch(query) : Unrecognized search field: keyword
3: In .HTMLsearch(query) : Unrecognized search field: alias
> 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
NULL
$timeout
[1] 0
$verbose
[1] "neutral"
```

With $1.30 profit per bushel at 110 bushels equals $143 and $2 profit per bushel at 30 bushels equals $60, we next are to use these values to set the objective function. From the help file: ”a numeric vector of length n (where n is the number of decision variables in lprec) containing the coefficients of the objective function”, so in this case there are 2 decision variables, and our vector is the profits per acre from each crop which must mean they are the coefficients.

```
> set.objfn(lprec, c(143, 60))
```

Next we are to add several constraints using the `add.constraint` command. Our first constraint is that it costs the farmer $120 per acre for the wheat and $210 per acre for the barley. The `type` of constraint is less than or equal to the `rhs` or ”right-hand side” or the constraint which is the farmer has $15000 available for expenses. Another constratint is that each acre produces an average of 110 bushels of wheat or 30 bushels of barley but the farmer only has storage space for less than or equal to 4000 bushels. The last constraint the program adds is a little confusing, as the farmer only has less than or equal to 75 acres to plant 2 crops, in this case the 2 crops ”constraints” are written as a vector of `(1,1)`.

```
> add.constraint(lprec, c(120, 210), "<=", 15000)
> add.constraint(lprec, c(110, 30), "<=", 4000)
> add.constraint(lprec, c(1, 1), "<=", 75)
```

Now we view the matrix and solve it. The lprec object shows us the ”Maximize” on the profit values we gave, and then the constraints, and finally ”Kind”, ”Type”–which I don’t quite understand–and an upper bounds of infinity and lower bounds of 0.

```
> lprec
Model name:
            C1    C2
Maximize   143    60
R1         120   210  <=  15000
R2         110    30  <=   4000
R3           1     1  <=     75
Kind       Std   Std
Type      Real  Real
Upper      Inf   Inf
Lower        0     0
```

```
> solve(lprec)
[1] 0
```

Finally we can get both the maximum profit that can be made and the amount of each crop to acheive this profit:

```
> get.objective(lprec)
[1] 6315.625
> get.variables(lprec)
[1] 21.875 53.125
```

So to get the maximum profit of $6315.63, the farmer needs to plant 21.875 acres of wheat and 53.125 acres of barley. Though in real life if you told the farmer this, he is likely going to round and plant 22 acres of wheat and 53 acres of barley. The surprising thing here is that is costs more to produce barley and it yields less bushels per acre than wheat, but to maximize profit, much more barley is required than the wheat, likely because the barley fetches more profit than wheat.

References

icyrock.com, 2013. Retrieved from http://icyrock.com/blog/2013/12/linear-programming-in-r-using-lpsolve/
Konis, Kjell. 2014. lpsolveAPI docs, retrieved from https://cran.r-project.org/web/packages/lpSolveAPI/index.html