Solving Linear Programming Problems in R with lpSolve

Basic formulation

lpSolve (and lpSolveAPI) are packages in R that allow for linear programming formulation and resolution. First we must install and load the packages:

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

As as simple example, we first create an empty model x:

```
x <- make.lp(2,2)
```

This creates a LP model with 2 constraints and 2 decision variables.

To solve problem 10 from the textbook *QMFB*, which is:

Max $2A + 3B$

$s.t.$

$1A + 2B \leq 6$

$5A + 3B \leq 15$

$A, B \geq 0$

This has 2 constraints and 2 decision variables. We can setup the problem in the following way:

```
  lp_ex <- make.lp(2, 2)
  lp.control(lp_ex, 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"
```

```r
set.objfn(lp_ex, c(2, 3))
add.constraint(lp_ex, c(1,2), "<=", 6)
add.constraint(lp_ex, c(5,3), "<=", 15)
set.bounds(lp_ex, lower = c(0, 0), columns = c(1, 2))
lp_ex
```

```
## Model name:
##             C1    C2
## Maximize     2     3
## R1           0     0  free   0
## R2           0     0  free   0
## R3           1     2    <=   6
```

```
## R4           5     3    <=  15
## Kind        Std   Std
## Type       Real  Real
## Upper       Inf   Inf
## Lower         0     0
```

The functions that operate on the lp_ex object are self-explanatory and set up the constraints, objective function and bounds. When we look at the object it has two free contraints that we can delete, and add in names for the constraints:

```
delete.constraint(lp_ex, 1)
delete.constraint(lp_ex, 1)
RowNames <- c("Constraint 1", "Constraint 2")
ColNames <- c("A", "B")
dimnames(lp_ex) <- list(RowNames, ColNames)
lp_ex
```

```
## Model name:
##                 A     B
## Maximize        2     3
## Constraint 1    1     2  <=   6
## Constraint 2    5     3  <=  15
## Kind          Std   Std
## Type         Real  Real
## Upper         Inf   Inf
## Lower           0     0
```

And to solve the problem:

```
solve(lp_ex)
```

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

```
get.objective(lp_ex)
```

```
## [1] 9.857143
```

```
get.variables(lp_ex)
```

```
## [1] 1.714286 2.142857
```

```
get.constraints(lp_ex)
```

```
## [1]  6 15
```

Graphically, we can plot constraints and objective function, as well as dashed lines for the approximate solution point.

```
b <- seq(0,3,by=.1)
a <- 6 - 2*b
b2 <- seq(0,5,by=.1)
a2 <- (15 - 3*b2)/5
o1 <- seq(0,3,by=.1)
o2 <- (9-3*o1)/2
plot(b,a, col='red',type='l')
lines(b2,a2, col='blue')
lines(o1,o2,col='green')
abline(h=2.142,lty = 2)
abline(v=1.714, lty =2)
```

```
legend(2,4.5,legend = c("constraint1", "constraint2", "objective"),lty = c(1,1),col = c('red','blue','g
```