Using The Lobico Function

Bo Li
Christopher Eeles
Benjamin Haibe-Kains

09/07/2019

Introduction

Please write an introduction about the package and its use cases.

Installing the Package

Please note that installation of this file depends on the package Rcpp to compile the C code. Additionally, to use the package and its function you must have a working installation of IBM ILOG CPLEX.

```
devtools::install_github("bhklab/RLOBICO", ref="RLOBICO_CRAN")
```

```
## Downloading GitHub repo bhklab/RLOBICO@RLOBICO_CRAN
```

```
##
##
   checking for file ‘/tmp/RtmpyF5ZnQ/remotes37fc3d2fc8a4/bhklab-RLOBICO-4f760c8/DESCRIPTION’ ...

v  checking for file ‘/tmp/RtmpyF5ZnQ/remotes37fc3d2fc8a4/bhklab-RLOBICO-4f760c8/DESCRIPTION’
##

-  preparing ‘rlobico’:
##

   checking DESCRIPTION meta-information ...

v  checking DESCRIPTION meta-information
##

-  cleaning src
##



-  checking for LF line-endings in source and make files and shell scripts
##

-  checking for empty or unneeded directories
## -  looking to see if a ‘data/datalist’ file should be added
## -  building ‘rlobico_0.1.0.tar.gz’
##


##
```

```
## Installing package into '/home/chris/R/x86_64-pc-linux-gnu-library/3.6'
## (as 'lib' is unspecified)
```

```
library(rlobico)
```

Data Loading

Please describe the data being used, as well as the meaing of each column accessed for this example.

```
load("../data/bibw2992.RData")
MutationMatrix <- bibw2992
Samples <- MutationMatrix$Cell.lines
IC50s <- MutationMatrix$BIBW2992
MutationMatrix <- MutationMatrix[, -2:-1]
Features <- colnames(MutationMatrix)
rownames(MutationMatrix) <- Samples
```

Configuring Parameters

Write some stuff about this. Feel free to merge and rename code blocks as you see fit.

```
## Create binary input, output, and weight vector

#binary input
X <- MutationMatrix

#write.csv(X, file="mutaionMatrix.csv", row.names=F, col.names=F)
N <- nrow(X)
P <- ncol(X)

#binarization threshold th
th <- 0.063218
Y <- as.double(IC50s < th)
W <- abs(IC50s - th)

#class weights
FPW <- 1
FPN <- 1

#normalize weights
W[Y == 1] <- FPW * W[Y == 1] / sum(W[Y == 1])
W[Y != 1] <- -(FPN * W[Y != 1] / sum(W[Y != 1]))

## Logic model complexity
K <- 2
M <- 1
```

CPLEX Options

Notes about configuring the options for CPLEX and the use cases for each configuration.

```
## Cplex options
# param <- rbind(list('timelimit.Cur', 60000, 'MaxTime'), #Maximum time for IP )in seconds)
#                list('mipltolerances.integrality.Cur', 1e-5, 'Integrality'), #Integrality constraint;
#                list('mip.tolerancs.mipgap.Cur', 1e-4, 'RelGap'), #Optimality gap tolerance; default =
#                list('threads.Cur', 8, 'Threads'), #Number of threads to use (default = 0, automatic)
#                list('parallel.Cur', -1, 'ParallelMode'), #Parallel optimization mode, -1 = opportunis
#                list('mip.pool.relgap.Cur', 1e-1, 'Pool_relgap'), #Relative gap for suboptimal solutio
```

```
#                list('mip.pool.intensity.Cur', 1, 'Pool_intensity'), #Pool intensity; default 1 = mild
#                list('mip.pool.replace.Cur', 2, 'Pool_replace'), #Pool replacement strategy; default 2
#                list('mip.pool.capacity.Cur', 11, 'Pool_capacity'), #Pool capacity; default 11 = best
#                list('mip.limits.populate.Cur', 11, 'Pool_size'))#Number of solutions generated; defau
param <- rbind(list('tilim', 60000, 'MaxTime'), #Maximum time for IP )in seconds)
               list('mipltolerances.integrality.Cur', 1e-5, 'Integrality'), #Integrality constraint; de
               list('epgap', 1e-4, 'RelGap'), #Optimality gap tolerance; default = 1e-4 (0.01% of optim
               list('threads.Cur', 8, 'Threads'), #Number of threads to use (default = 0, automatic) (s
               list('parallel.Cur', -1, 'ParallelMode'), #Parallel optimization mode, -1 = opportunisti
               list('solnpoolgap', 1e-1, 'Pool_relgap'), #Relative gap for suboptimal solutions in the
               list('mip.pool.intensity.Cur', 1, 'Pool_intensity'), #Pool intensity; default 1 = mild:
               list('mip.pool.replace.Cur', 2, 'Pool_replace'), #Pool replacement strategy; default 2 =
               list('mip.pool.capacity.Cur', 11, 'Pool_capacity'), #Pool capacity; default 11 = best +
               list('mip.limits.populate.Cur', 11, 'Pool_size'), #Number of solutions generated; defaul
               list('preind', 0, 'Presolver'),
               list('aggind', 1, 'Aggregator'))

param <- lapply(1:ncol(param), function(x){

  if (x==2) {
    return(as.numeric(unlist(param[, x])))
  }
  return(unlist(param[, x]))

})
param <- data.frame("V1" = param[[1]], "V2" = param[[2]], "V3" = param[[3]])
```

CPLEX Solver

Explaning the parameters and function of lobico.

```
## Cplex solver
sol <- lobico(X = X,
              Y = W,
              K = K,
              M = M,
              solve = 1,
              param = param)
```

```
## Construction optimizing function...
## Constructing constaints...
```

Validating Results

Some comments about the package.

```
## Check solution
print('**********')
```

```
## [1] "**********"
```

```
solMat <- .getsolution(sol, K, M, P)
print(solMat)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
## [1,]    0    0    0    0    0    0    0    0    0     0     1     1     0
```

```
##      [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24]
## [1,]     0     0     0     0     0     0     0     0     0     0     0
##      [,25] [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35]
## [1,]     0     0     0     0     0     0     0     0     0     0     0
##      [,36] [,37] [,38] [,39] [,40] [,41] [,42] [,43] [,44] [,45] [,46]
## [1,]     0     0     0     0     0     0     0     0     0     0     0
##      [,47] [,48] [,49] [,50] [,51] [,52] [,53] [,54] [,55] [,56] [,57]
## [1,]     0     0     0     0     0     0     0     0     0     0     0
##      [,58] [,59] [,60]
## [1,]     0     0     0
```

```
str <- .showformula(solMat, K, M, Features)
print('Inferred logical model')
```

```
## [1] "Inferred logical model"
```

```
print(str)
```

```
## [1] "EGFR  | ERBB2 "
```