Package: lpcover (via r-universe)

November 1, 2024

Title LPCover: Functionality for integer programming methods for covering

Version 0.0.01

Author Wikum Dinalankara <wdd4001@med.cornell.edu>, Luigi Marchionni <lum4003@med.cornell.edu>, Qian Ke <qke1@jhu.edu>

Maintainer Wikum Dinalankara <wdd4001@med.cornell.edu>

Description Integer programming functionality for different 'covering' optimizations as presented in Ke et al, ``Efficient Representations of Tumor Diversity with Paired DNA-RNA Anomalies".

Depends R (>= 3.6), lpSolve

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

biocViews Software, StatisticalMethod

Suggests knitr, rmarkdown

VignetteBuilder knitr

Repository https://marchionnilab.r-universe.dev

RemoteUrl https://github.com/marchionniLab/lpcover

RemoteRef HEAD

RemoteSha ecc16269d15787924976ea57674bfae70c431390

Contents

computeMinimalCovering *Cover a proportion of a given binary set with the smallest number of features*

Usage

```
computeMinimalCovering(mat, alpha = 0.05, maxsol = 100, J = 1, solver = "")
```

Arguments

`mat` A binary data matrix with each column corresponding to a sample and each row corresponding to a feature.

`alpha` A value in the 0 <= alpha < 1 range indicating what proportion of samples to be considered as outlier. By default alpha = 0.05, indicating 95

\itemmaxsolThe number of optimal solutions to be returned. Default is 100.

\itemJThe number of times each sample is to be covered. By default J=1, indicating that each sample is to be covered with at least one feature.

\itemsolverA character string indicating whether to use gurobi or lpSolve.

A list with items "obj": the objective returned by the optimization (as a vector), "sol": a character matrix of solutions(each column a solution), "r": a list where each element contains vectors of results obtained for x and lamba vectors, and "result": the direct output returned by the optimization (by either gurobi or lpSolve).

Function for computing the minimal covering for a given binary data matrix given a minimum proportion of samples to cover

optim.out = computeMinimalCovering(mat=mat, alpha=0.05, maxsol=1, J=1, solver="lpSolve")

cover, optimize

Index