Package ‘elec.strat’

February 19, 2015

Type Package
Title Functions for election audits using stratified random samples
Version 0.1.1
Date 2011-07-19
Author Mike Higgins, Hua Yang
Maintainer Mike Higgins <mikehiggins83@gmail.com>
Depends R (>= 2.10), elec
Description An extension of the elec package intended for use on election audits using stratified random samples. Includes functions to obtain conservative and exact p-values, and functions that give sample sizes that may make election audits more efficient.
License GPL (>= 2)
LazyLoad yes
Repository CRAN
Date/Publication 2012-04-10 04:23:04
NeedsCompilation yes

R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>BaB</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>CA_House_2008</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>eqValBound</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>first.r</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>get.first.r.samp</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>getEbsMargin</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>getQ</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>is.strat.elec.data</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>LKPBound</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>makeStratObj</td>
<td></td>
<td>12</td>
</tr>
<tr>
<td>MN_Senate_2006</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>optStrat</td>
<td></td>
<td>14</td>
</tr>
</tbody>
</table>
Finding the exact p-value.

Description

BaB finds an exact p-value by solving a 0-1 knapsack problem. The 0-1 knapsack problem is solved by a branch and bound algorithm. For more details, see Higgins, Rivest, Stark.

Usage

```r
BaB(Z, t = NULL, asTaint = FALSE, asNumber = FALSE,
M = NULL, takeOutZeroMBM = TRUE, give.strategy = FALSE,
bound.col = "e.max", calc.e_p = calc.pairwise.e_p,
w_p = weight.function("no.weight"))
```

Arguments

- **Z**
  - A `strat.elec.data` object.

- **t**
  - Value of the observed maximum, either as the MRO, as taint, or as the overstatement of the margin in votes.

- **asTaint**
  - Set `asTaint = TRUE` if `t` is the maximum observed taint.

- **asNumber**
  - Set `asNumber` if `t` is the maximum observed overstatement of the margin in votes.

- **M**
  - *A priori* margin. If NULL, M defaults to 1.

- **takeOutZeroMBM**
  - Setting `takeOutZeroMBM = TRUE` will consider batches with a maximumMarginBound of zero as having no chance of being sampled.

- **give.strategy**
  - If `give.strategy = TRUE`, output will include the solution to the 0-1 knapsack problem.

- **bound.col, calc.e_p, w_p**
  - Arguments used to compute `t` from audit data, instead of passing `t` directly. These arguments are ignored if `t` is not NULL. See `compute.stark.t` for details.

Details

BaB pre-processes the data to make the branch and bound algorithm more efficient, and obtains all information from `Z` necessary to perform the branch and bound algorithm. BaB then calls `runBaB`, which calls the branch and bound function.

When `give.strategy = TRUE`, the output of the solution will be a vector `strategy` of size `length(nrow(Z$strat))`. The solution can be obtained by, for each stratum `i`, putting `e_max` amount of difference in the `strategy[[i]]` batches corresponding to the largest values of `u`. For more details, see Higgins, Rivest, Stark.
Author(s)
Mike Higgins, Hua Yang

References
M. Higgins, R. L. Rivest, P. B. Stark. Sharper p-Values for Stratified Election Audits

See Also
See LKPBound for finding a p-value through a continuous relaxation. See eqValBound and withReplaceBound for finding a p-value through other relaxations. See runBaB for running the branch and bound algorithm given a value vector \( u \), a cost vector \( q \), a margin \( M \), and a CIDnum vector. See compute.stark.t for computing \( t \) through audit data.

Examples
```r
data(MN_Senate_2006)
BaB(MN_Senate_2006.strat, takeOutZeroMMB = FALSE, give.strategy = TRUE)
```

---

**CA_House_2008**

**Set of 2008 California U.S. House Races**

**Description**
A list of 20 strat.elec.data objects, each containing voting data for a contest in the 2008 California U.S. House Race. The data for contest \( i \) is contained in the strat.elec.data object `CA_House_2008.strat[[i]]`.

**Usage**
```r
data(CA_House_2008)
```

**Details**
Each of the 20 contests had exactly two candidates that received a large portion of the vote. Each contest was contained within 2 to 5 counties. optStrat can find sample sizes for most of these contests in a reasonable amount of time.

Data for the contests were obtained through the California Statewide Database (SWDB). The data can be found at [http://swdb.berkeley.edu/pub/data/G08/state/state_g08_sov_data_by_g08_svprec.dbf](http://swdb.berkeley.edu/pub/data/G08/state/state_g08_sov_data_by_g08_svprec.dbf).

**Examples**
```r
data(CA_House_2008)
optStrat(CA_House_2008.strat[[3]], alpha = .1, t = 0)
```
Description

eqValBound and withReplaceBound find a p-value by changing the original constraint (that the total difference is greater than the margin), instead placing a restriction on the number of batches with error no larger than $t$.
eqValBound finds an exact solution with this restriction, whereas withReplaceBound finds a more conservative bound. See Stark for more details about withReplaceBound.

Usage

eqValBound(Z, t = NULL, asTaint = FALSE, asNumber = FALSE, m = NULL, takeOutZeroMMB=TRUE, bound.col = "e.max",
  calc.e.p=calc.pairwise.e_p, w_p = weight.function("no.weight"))
withReplaceBound(Z, t = NULL, asTaint = FALSE, asNumber = FALSE,
  M = NULL, takeOutZeroMMB = TRUE, bound.col = "e.max",
  calc.e.p=calc.pairwise.e_p, w_p = weight.function("no.weight"))

Arguments

$Z$  A strat.elec.data object.
$t$  Value of the observed maximum, either as the MRO, as taint, or as the overstatement of the margin in votes.
asTaint  Set asTaint = TRUE if $t$ is the maximum observed taint.
asNumber  Set asNumber if $t$ is the maximum observed overstatement of the margin in votes.
$M$  A priori margin. If NULL, $M$ defaults to 1.
takeOutZeroMMB  Setting takeOutZeroMMB = TRUE will consider batches with a maximumMarginBound of zero as having no chance of being sampled.
bound.col, calc.e.p, w_p  Arguments used to compute $t$ from audit data, instead of passing $t$ directly. These arguments are ignored if $t$ is not NULL. See compute.stark.t for details.

Author(s)

Mike Higgins, Hua Yang

References

See Also

See LKPBound for finding a p-value through a continuous relaxation. See BaB for finding an exact p-value through solving a 0-1 knapsack problem. See compute.stark.t for computing t through audit data.

Examples

data(MN_Senate_2006)
eqValBound(MN_Senate_2006.strat, takeOutZeroMMB = FALSE)
withReplaceBound(MN_Senate_2006.strat, t = 2, asNumber = TRUE,
takeOutZeroMMB = FALSE)

firstNr  Obtain a Vector of Sample Sizes with Total Number of Samples Fixed

Description

first Nr, next Nr, and propSizes obtain sample sizes so that the total number of samples is fixed.
first Nr uses the first Nr algorithm, next Nr uses the next Nr algorithm, and propSizes finds a vector of sample sizes that is proportional to stratum sizes.

See Higgins, Rivest, Stark for details about the first Nr and the next Nr algorithms.

Usage

firstNr(Z, n, t = 0, astaint = FALSE, asNumber = FALSE, M = NULL, initsamp = NULL)
nextNr(Z, n, t = 0, astaint = FALSE, asNumber = FALSE, M = NULL, initsamp = NULL)
propSizes(Z, n)

Arguments

Z  A strat.elec.data object.
n  The fixed number of samples. When initsamp is provided, firstNr and nextNr will run for n iterations, adding samples iteratively to initsamp; firstNr and nextNr will produce a vector of sample sizes with a total of sum(initsamp) + n) samples.
t  Value of the observed maximum, either as the MRO, as taint, or as the overstatement of the margin in votes.
astaint  Set astaint = TRUE if t is the maximum observed taint.
asNumber  Set asNumber if t is the maximum observed overstatement of the margin in votes.
M  A priori margin. If NULL, M defaults to 1.
initsamp  An initial choice of sample sizes. Used in call of get.first.r.samp and get.next.r.samp to reduce computational time.
Details

The arguments `t`, `asTaint`, `asNumber`, `M` are used in `first.r` and `next.r` in the call of `getEbsMargin`. The `getQ` function is bypassed to increase efficiency.

`propStrat` obtains a vector of sample sizes that has exactly `n` samples. It obtains such a sample by sorting values of \( k \times \text{sum}(Z_{strat}n) / Z_{strat}n \), where \( k = 0, 1, \ldots \), in increasing order and allocating a sample to the strata corresponding to the first `n` values. Ties are broken by choosing the strata with the largest number of batches. See Higgins, Rivest, Stark for details.

Author(s)

Mike Higgins, Hua Yang

References

M. Higgins, R. L. Rivest, P. B. Stark. *Sharper p-Values for Stratified Election Audits*

See Also

See `get.first.r.samp`, `get.next.r.samp`, and `get.prop.samp` for finding sample sizes given constraints on the p-value and the largest observed overstatement.

Examples

data(MN_Senate_2006)
MN_Senate_2006.strat$strat$audit <- first.r(MN_Senate_2006.strat, n = 150)
BaB(MN_Senate_2006.strat)
MN_Senate_2006.strat$strat$audit <- next.r(MN_Senate_2006.strat, n = 150)
BaB(MN_Senate_2006.strat)
MN_Senate_2006.strat$strat$audit <- propSizes(MN_Senate_2006.strat, n = 150)
BaB(MN_Senate_2006.strat)

---

`get.first.r.samp` *Obtain a Vector of Sample Sizes Given Constraint on p-Value*

Description

`get.first.r.samp`, `get.next.r.samp`, and `get.prop.samp` obtain sample sizes so that, if a maximum observed overstatement of `t` or less is observed, the sample will produce a p-value less than `alpha`.

`get.first.r.samp` uses the `first.r` algorithm to obtain the sample, `get.next.r.samp` uses the `next.r` algorithm to obtain the sample, and `get.prop.samp` finds a vector of sample sizes that is proportional to stratum sizes.

For details about the `first.r` and the `next.r` algorithms, and for a description on how to produce a sample that will ensure that the p-value is less than `alpha` when no overstatement greater than `t` is uncovered, see Higgins, Rivest, Stark.
Usage

get.first.r.samp(Z, alpha, t, bal=TRUE, numSamp = TRUE, initn = 1,
  asTaint = FALSE, asNumber = FALSE, M = NULL,
  takeOutZeroMBB=TRUE)
get.next.r.samp(Z, alpha, t, bal=TRUE, numSamp = TRUE, initn = 1,
  asTaint = FALSE, asNumber = FALSE, M = NULL,
  takeOutZeroMBB=TRUE)
get.prop.samp(Z, alpha, t, bal=TRUE, numSamp = TRUE, initn = 1,
  asTaint = FALSE, asNumber = FALSE, M = NULL,
  takeOutZeroMBB=TRUE)

Arguments

Z  A strat.elec.data object.
t  Value of the observed maximum, either as the MRO, as taint, or as the overstatement of the margin in votes.
alpha  Threshold for the p-value. If an audit does not uncover an overstatement less than t, the sample obtained will ensure that the p-value is less than alpha.
bals  If bal = TRUE, the output will include the expected number of audited ballots for the sample.
numSamp  If numSamp = TRUE, the output will include the total number of audited batches.
initn  The first sample size checked by algorithm will have a total of initn samples. If this first sample will not produce a p-value less than alpha, the algorithm will increment the number of samples until such a vector of sample sizes is found. initn may be adjusted to dramatically decrease the runtime of algorithms.
asTaint  Set asTaint = TRUE if t is the maximum observed taint.
asNumber  Set asNumber if t is the maximum observed overstatement of the margin in votes.
M  A priori margin. If NULL, M defaults to 1.
takeOutZeroMBB  Setting takeOutZeroMBB = TRUE will consider batches with a maximumMarginBound of zero as having no chance of being sampled.

Details

Sample sizes from get.first.r.samp and get.next.r.samp are obtained by repeatedly calling first.r and next.r, respectively, while incrementing the total number of samples n. The algorithm stops when the sample produced will ensure a p-value less than alpha.

Author(s)

Mike Higgins

References

M. Higgins, R. L. Rivest, P. B. Stark. Sharper p-Values for Stratified Election Audits
getEbsMargin

See Also
See first.r, next.r and propSizes for finding sample sizes given constraints on the p-value and the largest observed overstatement. Also, see first.r and next.r for a brief description of the first.r and next.r algorithms. See optStrat for finding optimal sample sizes so that, if a maximum observed overstatement of t or less is observed, the sample will produce a p-value less than alpha. Optimal sample sizes will minimize the number of batches required for audit.

Examples

data(CA_House_2008)
get.first.r.samp(CA_House_2008.strat[[3]], alpha = .1, t = .01, asTaint = TRUE)
get.next.r.samp(CA_House_2008.strat[[3]], alpha = .1, t = .01, asTaint = TRUE)
get.prop.samp(CA_House_2008.strat[[3]], alpha = .1, t = .01, asTaint = TRUE)

getEbsMargin

Updating Error Bounds and Margin Given the Observed Maximum

Description
Gives updated values u and M given a value of the observed maximum t.

Usage
getEbsMargin(Z, t, asTaint = FALSE, asNumber = FALSE, M = NULL)

Arguments

Z A strat.elec.data object.
t Value of the observed maximum, either as the MRO, as taint, or as the overstatement of the margin in votes.
asTaint Set asTaint = TRUE if t is the maximum observed taint.
asNumber Set asNumber if t is the maximum observed overstatement of the margin in votes.
M A priori margin. If NULL, M defaults to 1.

Details
Creates values u and margin M that can be passed into the branch and bound function. The following definitions for u and M are described in Higgins, Rivest, Stark. The quantity e.max is obtained through maximumMarginBound.

- Default u = e.max - min(e.max, t). M = M - sum(min(e.max, t)).
- asTaint = TRUE u = e.max*(1 - t). M = M - sum(e.max*t)
- asNumber = TRUE Same as Default with t = t/Z$Margin.

The output of getEbsMargin is a list consisting of

- M The updated margin.
- u The updated value vector.
getQ

Author(s)
Mike Higgins, Hua Yang

References
M. Higgins, R. L. Rivest, P. B. Stark. *Sharper p-Values for Stratified Election Audits*

Examples
```r
data(MN_Senate_2006)
getEbsMargin(MN_Senate_2006.strat, t = 0.009, asTaint = TRUE)
```

---

**getQ**

*Obtaining the Cost Vector.*

Description
Obtains the cost value \( q \), which can then be passed into the branch and bound function. See Higgins, Rivest, Stark for details.

Usage
```r
getQ(Z)
```

Arguments
- \( Z \)
  - A `strat.elec.data` object.

Author(s)
Mike Higgins, Hua Yang

References
M. Higgins, R. L. Rivest, P. B. Stark. *Sharper p-Values for Stratified Election Audits*

Examples
```r
data(MN_Senate_2006)
getQ(MN_Senate_2006.strat)
```
is.strat.elec.data  Verifying a strat.elec.data Object.

Description

Verifies that an object is a strat.elec.data object.

Usage

is.strat.elec.data(Z)

Arguments

Z  An object. is.strat.elec.data is TRUE when Z is a strat.elec.data object.

Author(s)

Mike Higgins

See Also

See strat.elec.data or makeStratObj for building a strat.elec.data object.

Examples

data(MN_Senate_2006)
is.strat.elec.data(MN_Senate_2006.strat)

LKPBound  p-value Through a Continuous Relaxation

Description

Finds a p-value through the LKP Bound: a continuous relaxation bound of the original 0-1 knapsack problem. Offers an option to include a lower-bound in output, thus computing an upper and lower bound on the exact p-value. See Higgins, Rivest, Stark for more details.

Usage

LKPBound(Z, t = NULL, asTaint = FALSE, asNumber = FALSE, M = NULL, takeOutZeroMMB = TRUE, LKP.lower.bound = FALSE, bound.col = "e.max", calc.e_p=calc.pairwise.e_p, w_p = weight.function("no.weight"))
Arguments

- **Z**
  A *strat.elec.data* object.

- **t**
  Value of the observed maximum, either as the MRO, as taint, or as the overstatement of the margin in votes.

- **asTaint**
  Set `asTaint = TRUE` if `t` is the maximum observed taint.

- **asNumber**
  Set `asNumber` if `t` is the maximum observed overstatement of the margin in votes.

- **M**
  *A priori* margin. If NULL, `M` defaults to 1.

- **takeOutZeroMB**
  Setting `takeOutZeroMB = TRUE` will consider batches with a `maximumMarginBound` of zero as having no chance of being sampled.

- **lkp.lowerbound**
  Set `lkp.lowerbound = TRUE` to compute a lower-bound of the exact p-value in addition to the upper-bound computed from the continuous relaxation. Lower-bound computed according to Higgins, Rivest, Stark.

- **bound.col, calc.e_p, w_p**
  Arguments used to compute `t` from audit data, instead of passing `t` directly. These arguments are ignored if `t` is not NULL. See *compute.stark.t* for details.

Author(s)

Mike Higgins, Hua Yang

References

M. Higgins, R. L. Rivest, P. B. Stark. *Sharper p-Values for Stratified Election Audits*

See Also

See `eqValBound` and `withReplaceBound` for finding a p-value through other relaxations. See *BaB* for finding an exact p-value through solving a 0-1 knapsack problem. See *compute.stark.t* for computing `t` through audit data.

Examples

```r
data(MN_Senate_2006)
LKPBound(MN_Senate_2006.strat, takeOutZeroMB = FALSE)
LKPBound(MN_Senate_2006.strat, t = 2, asNumber = TRUE, takeOutZeroMB = FALSE, LKP.lowerbound = TRUE)

data(CA_House_2008)
CA_House_2008.strat[[1]]$strat$audit <- 1
LKPBound(CA_House_2008.strat[[1]], t = 0, LKP.lowerbound = TRUE)
```
makeStratObj  
Making a strat.elec.data Object from an elec.data Object

Description
Makes a strat.elec.data object from an elec.data object.

Usage
makeStratObj(Z, strat.col = NULL, CID = NULL, auditTable = NULL)

Arguments

Z  An elec.data object.
strat.col  Name of column in Z$V that identifies strata. If no value of strat.col is passed, makeStratObj will assume that stratum ID is contained in Z$V$CID.
CID  A vector of length nrow(Z$V) that identifies strata.
auditTable  A data.frame of dimension length(unique(CID)) x 2 used to identify the number of samples taken from each stratum. Including auditTable is not necessary if Z contains audit information.

Details

makeStratObj requires as input a valid elec.data object Z such that one of the following is true:

- Z has a vector Z$V$CID that identifies strata.
- A strat.col name identifying the stratification column in Z$V$ is passed to makeStratObj.
- A CID vector of length nrow(Z$V$) specifying the stratification is passed to makeStratObj.

If Z$audit is NULL, information giving the number of sampled batches in each stratum can be included through auditTable. The argument auditTable should be a data.frame of dimensions unique(CID) x 2. The first column is a list of unique stratum IDs. The second column is the number of batches sampled within the corresponding stratum.

makeStratObj creates a data.frame Z$strat. The columns of Z$strat are

- CID The ID of the stratum.
- CIDnum A number between 1 and length(Z$strat$CID) assigned to that stratum.
- n The number of batches contained in that stratum.
- audit The number of batches sampled from that stratum.

If no audit data is provided, Z$strat$audit defaults to a zero vector.

If Z$V$CID is NULL, makeStratObj will copy the stratum labels into Z$V$CID. makeStratObj will also create

- Z$CID.col"CID"
• Z$CIDnum The CIDnum of the stratum.
• Z$V$e.max maximumMarginBound(Z)

If Z$audit[Z$PID.col] is not NULL, makeStratObj will create Z$audit$e.max, the maximumMarginBound(Z) for batches in Z$audit.

After sending an elec.data object through makeStratObj, the object will be both an elec.data object and a strat.elec.data object.

Author(s)
Mike Higgins, Hua Yang

See Also
See strat.elec.data to create a strat.elec.data object from a votes data.frame and an audit data.frame.

Examples

data("CA_House_2008")
dstrat <- CA_House_2008.strat[[1]]
auditTable <- cbind(unique(dstrat$V$CID),1)
dstrat <- makeStratObj(dstrat,auditTable = auditTable)

MN_Senate_2006  2006 Minnesota U.S. Senate Race

Description
Contains a strat.elec.data object for the 2006 Minnesota U.S. Senate Race named MN_Senate_2006.strat.

Usage
data(MN_Senate_2006)

Details
The winner of the election was Amy Klobuchar. Mark Kennedy was the runner-up. There were a total of 2,217,818 voters, and the margin of victor was 443,196 votes. The largest precinct wise difference between the hand count and machine count was a 2-vote swing from Amy Klobuchar to Mark Kennedy.

References
Examples

data(MN_Senate_2006)
BaB(MN_Senate_2006.strat)

optStrat

Obtain an Optimal Vector of Sample Sizes Given Constraint on p-Value

Description

optStrat will obtain sample sizes so that, if a maximum observed overstatement of \( t \) or less is observed, the sample will produce a p-value less than \( \alpha \). The sample that \( \text{optStrat} \) obtains minimizes the total number of batches required for audit. \( \text{optStrat} \) includes options so that, given the number of samples required for audit for optimal sample sizes, the sample that minimizes the expected number of audited ballots is found.

\( \text{optStrat} \) can be a very computationally expensive function, and should only be used for small contests.

Usage

\[
\text{optStrat}(Z, \alpha, t, \text{bal=} \text{TRUE}, \text{optBal=} \text{FALSE}, \text{numSamp=} \text{TRUE}, \text{asTaint=} \text{FALSE}, \text{asNumber=} \text{FALSE}, M=\text{NULL}, \text{takeOutZeroMMB=} \text{TRUE})
\]

Arguments

\( Z \quad \) A \texttt{strat.elec.data} object.
\( t \quad \) Value of the observed maximum, either as the MRO, as taint, or as the overstatement of the margin in votes.
\( \alpha \quad \) Threshold for the p-value. If an audit does not uncover an overstatement less than \( t \), the sample obtained will ensure that the p-value is less than \( \alpha \).
\( \text{bal} \quad \) If \( \text{bal=} \text{TRUE} \), the output will include the expected number of audited ballots for the sample.
\( \text{optBal} \quad \) If \( \text{optBal=} \text{TRUE} \), given the number of batches required for audit in an optimal sample, \( \text{optSamp} \) will find the sample that minimizes the expected number of audited ballots. This may dramatically increase the runtime of \( \text{optStrat} \).
\( \text{numSamp} \quad \) If \( \text{numSamp=} \text{TRUE} \), the output will include the total number of audited batches.
\( \text{asTaint} \quad \) Set \( \text{asTaint=} \text{TRUE} \) if \( t \) is the maximum observed taint.
\( \text{asNumber} \quad \) Set \( \text{asNumber} \) if \( t \) is the maximum observed overstatement of the margin in votes.
\( M \quad \) \textit{A priori} margin. If NULL, \( M \) defaults to 1.
\( \text{takeOutZeroMMB} \quad \) Setting \( \text{takeOutZeroMMB=} \text{TRUE} \) will consider batches with a \texttt{maximumMarginBound} of zero as having no chance of being sampled.
runBaB

Author(s)

Mike Higgins

See Also

See `get.first.r.samp`, `get.next.r.samp`, and `get.prop.samp` for other methods to obtain sample sizes so that, if a maximum observed overstatement of $t$ or less is observed, the sample will produce a p-value less than $\alpha$. `get.first.r.samp` uses the `first.r` algorithm to obtain the sample, `get.next.r.samp` uses the `next.r` algorithm to obtain the sample, and `get.prop.samp` finds a vector of sample sizes that is proportional to stratum sizes.

Examples

data(CA_House_2008)
optStrat(CA_House_2008.strat[[3]], alpha = .1, t = .01, asTaint = TRUE)
optStrat(CA_House_2008.strat[[3]], alpha = .1, t = .01,
asTaint = TRUE, optBal = TRUE)

runBaB Calling the Branch and Bound Algorithm

Description

`runBaB` calls the branch and bound algorithm. The branch and bound algorithm is coded in C.

Usage

`runBaB(u, q, M, CIDnum)`

Arguments

- `u`: A vector of values. Can be obtained through `getEbsMargin`.
- `q`: A vector of costs. Can be obtained through `getQ`.
- `M`: The constraint on the values. Can be obtained through `getEbsMargin`.
- `CIDnum`: A vector that gives the CIDnum identification for each batch. Can be found at `Z$V$CIDnum`.

Author(s)

Mike Higgins
Examples

data(MN_Senate_2006)
M.u <- getEbsMargin(MN_Senate_2006.strat, t=2, asNumber = TRUE)
u <- M.u$u
M <- M.u$M
q <- getQ(MN_Senate_2006.strat)
CIDnum <- MN_Senate_2006.strat$V$CIDnum
runBaB(u,q,M,CIDnum)

strat.elec.data  Making a strat.elec.data Object from a Votes data.frame and an Audit data.frame

Description

Makes a strat.elec.data and an elec.data object from a votes data.frame and an audit data.frame.

Usage

strat.elec.data(V, C.names=names(V)[2:length(V)], f = 1,
 audit=NULL, pool=TRUE, tot.votes.col="tot.votes", PID.col="PID",
 strat.col = NULL, CID = NULL, auditTable = NULL)

Arguments

V  A data.frame of votes.
C.names  Names of candidates.
f  The number of winners.
audit  An audit data.frame.
pool  Combine small candidates into single pseudo-candidates to increase power.
tot.votes.col  Name of column that has the total votes for the batches.
PID.col  Name of column that identifies unique batches.
strat.col  Name of column in votes that identifies strata.
CID  A vector of length nrow(votes) that identifies strata.
auditTable  A data.frame of dimension length(unique(CID)) x 2 used to identify the number of samples taken from each stratum. The auditTable is not necessary if an audit data.frame is included.

Details

strat.elec.data creates a strat.elec.data object: an elec.data object with additional entries for easy use with the elec.strat package.

strat.elec.data allows for two ways to specify the stratification:

- Specify strat.col: the name of the column in V that contains strata information.
• Provide a CID vector of length nrow(V) specifying the stratification.

If neither method is used to specify stratification, only an elec.data object is created.

If audit is not NULL and strat.col is provided, strat.elec.data will find the strat.col column in audit to create an auditTable. If no column in audit is labeled as strat.col, strat.elec.data will throw an error.

If audit is not NULL, strat.col is NULL, and CID is provided, strat.elec.data will try to find the column in audit labeled PID.col to generate the auditTable. If the PID.col column is not in audit, then auditTable will need to be given; otherwise Z$strat$audit defaults to a zero vector.

The argument auditTable should be a data.frame of dimensions unique(CID) x 2. The first column is a list of unique stratum IDs. The second column is the number of batches sampled within the corresponding stratum.

strat.elec.data first calls elec.data to create an elec object Z. An auditTable is either created through audit or given by the auditTable argument, and makeStratObj is called to create a strat.elec.data object.

For a detailed description of the structure of a strat.elec.data object, see makeStratObj. For a more detailed description of the arguments V, C.names, f, pool, audit, tot.votes.col, see elec.data.

Author(s)

Mike Higgins

See Also

See elec.data to create an elec.data object. See makeStratObj to create a strat.elec.data object from an elec.data object. Both elec.data and makeStratObj are called by strat.elec.data.

Examples

data(MN_Senate_2006)
votes <- MN_Senate_2006$strat$V
audit <- MN_Senate_2006$strat$audit
CID <- MN_Senate_2006$strat$V%CID
names <- c("Klo","Ken")
strat.elec.data(V = votes, C.names = names, audit = audit, CID = CID)
Index

*Topic **datasets**
  CA_House_2008, 3
  MN_Senate_2006, 13

BaB, 2, 5, 11

CA_House_2008, 3
compute.stark.t, 2–5, 11

elec.data, 12, 13, 16, 17
eqValBound, 3, 4, 11

first.r, 5, 6–8, 15

general.first.r.samp, 5, 6, 15
general.next.r.samp, 5, 6, 15
general.next.r.samp (general.first.r.samp), 6
general.prop.samp, 6, 15
general.prop.samp (general.first.r.samp), 6
general.getEbsMargin, 6, 8, 15
general.getQ, 6, 9, 15

is.strat.elec.data, 10

list, 3
LKPBound, 3, 5, 10

makeStratObj, 10, 12, 17
maximumMarginBound, 2, 4, 7, 8, 11, 14
MN_Senate_2006, 13

next.r, 6–8, 15
next.r (first.r), 5

optStrat, 3, 8, 14

propSizes, 8
propSizes (first.r), 5

runBaB, 2, 3, 15

strat.elec.data, 2–5, 7–14, 16

withReplaceBound, 3, 11
withReplaceBound (eqValBound), 4