Package: SparseMSE (via r-universe)

November 3, 2024

Title 'Multiple Systems Estimation for Sparse Capture Data'

Version 3.0.1

Author Lax Chan [aut, cre], Bernard Silverman [aut], Kyle Vincent [aut]

Maintainer Lax Chan <laxchan77@gmail.com>

Description Implements the routines and algorithms developed and analysed in ``Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists" Chan, L, Silverman, B. W., Vincent, K (2019) <https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748> and in ``Bootstrapping multiple systems estimates to account for model selection" Silverman, B. W., Chan, L, Vincent, K (2023)<https://doi.org/10.1007/s11222-023-10346-9>. This package explicitly handles situations where there are pairs of lists which have no observed individuals in common. It deals correctly with parameters whose estimated values can be considered as being negative infinity. It also addresses other possible issues of non-existence and non-identifiability of maximum likelihood estimates.

URL https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748 https://doi.org/10.1007/s11222-023-10346-9

Depends R (>= 4.2.0)

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Suggests knitr, rmarkdown

Imports lpSolve, Rcapture

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

RemoteUrl https://github.com/laxchan/sparsemse

RemoteRef HEAD

RemoteSha 627b8bbb2e9b545e9a14f0d355ba96111d960fc3

Contents

`ancestors` *Find the "ancestors" of a given capture history*

Description

Given any encoded capture history and the number of lists, find all the encoded capture histories that are included in the original capture history

Usage

```
ancestors(k, nlists = 10)
```

Arguments

`k`	An encoded capture history
`nlists`	The total number of lists

Value

a vector giving the encoded versions of the ancestors

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
ancestors(2,10)
ancestors(1,5)
```

Artificial_3 *Artificial data set to demonstrate possible instabilities*

Description

This is a simple data set based on three lists, which gives examples of models that fail on one or the other of the criteria tested by checkident. This is Table 2 in Chan, Silverman and Vincent (2021).

Usage

```
Artificial_3
```

Format

An object of class data.frame with 4 rows and 4 columns.

Details

If all three two-list effects are included in the fitted model then the linear program in checkident yields a strictly positive value but the matrix A is not of full column rank, so the parameters are not identifiable. If the model contains AB either alone or in conjunction with one of AC and BC, then the linear program result is zero, so the MLE does not exist. If only main effects are considered, or if either or both of AC and BC, but not AB are included, then the model passes both tests.

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

assemble_bic *Models BICS, abundance and maxorder.*

Description

This routine sorts the models in increasing order according to their BICs, returns the sorted models with their corresponding BICs and abundance. The original data as well as the maxorder of the models considered are returned as well.

Usage

```
assemble_bic(
  xdata,
  maxorder = dim(xdata)[2] - 2,
  checkexist = T,
  removeFRfail = T,
  ...
)
```

Arguments

xdata	The original data matrix with capture histories and counts.
maxorder	Maximum order of models to be included
checkexist	If T then the Fienberg-Rinaldo condition is checked for each model
removeFRfail	If checkexist is T then models which fail the FR condition are removed from the results
...	Parameters to be fed to gethiermodels.

Value

A list with the following components

res A matrix with the models considered, their abundance, BIC and their order, sorted into increasing order of BIC

xdata The original data matrix with capture histories and counts.

maxorder Order parameter that was feed into gethiermodels

Examples

```
data(hiermodels)
data(Korea)
assemble_bic(Korea, checkexist=T)
```

---

bcaconfvalues *BCa confidence intervals*

Description

The BCa confidence intervals use percentiles of the bootstrap distribution of the population size , but adjust the percentile actually used. The adjusted percentiles depend on an estimated bias parameter, and the quantile function of the estimated bias parameter is the proportion of the bootstrap estimates that fall below the estimate from the original data, and an estimated acceleration factor, which derivation depends on a jackknife approach. This routine is called internally by estimatepopulation.

Usage

```
bcaconfvalues(
  bootreps,
  popest,
  ahat,
  alpha = c(0.025, 0.05, 0.1, 0.16, 0.84, 0.9, 0.95, 0.975)
)
```

Arguments

bootreps	Point estimates of total population sizes from each bootstrap sample.
popest	A point estimate of the total population of the original data set.
ahat	the estimated acceleration factor
alpha	Bootstrap quantiles of interests

Value

BCa confidence intervals

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

DiCiccio, T. J. and Efron, B. (1996). Bootstrap Confidence Intervals. *Statistical Science*, **40(3)**, 189-228.

Efron, B. (1987). Better Bootstrap Confidence Intervals. *Journal of the American Statistical Association*, **82(397)**, 171-185.

BICandbootstrapsim *Comparison of BIC approach and BCa approach*

Description

This routine carries out the simulation study as detailed in Section 3.4 of Chan, Silverman and Vincent (2021). If the original data set has low counts, so that there is a possibility of a simulated data set containing empty lists, then it may be advisable to use the option noninformativelist=TRUE.

Usage

```
BICandbootstrapsim(
  zdat,
  nsims = 100,
  nboot = 100,
  pthresh = 0.02,
  iseed = 1234,
  alpha = c(0.025, 0.05, 0.1, 0.16, 0.5, 0.84, 0.9, 0.95, 0.975),
  noninformativelist = F,
  verbose = F,
  ...
)
```

Arguments

zdat	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has zero count.
nsims	Number of simulations to be carried out.
nboot	Number of bootstrap replications for each simulation
pthresh	p-value threshold used in estimatepopulation.0.
iseed	seed for initialization.
alpha	bootstrap quantiles of interests.
noninformativelist	if noninformativelist=TRUE then each generated data set in the simulation study (including all bootstrap replications) will be passed to removenoninformativelists.
verbose	If verbose=FALSE, then the progress of the simulation will not show. If verbose=TRUE, then the progress of the simulation will be shown.
...	other arguments.

Value

A list with components as below

popest Total population point estimate from the original data using estimatepopulation.0 with default threshold.

BICmodels The best model chosen by the BIC at each simulation.

BICvals Point estimates of the total population and standard error of the best model chosen by the BIC at each simulation.

simreps Counts associated to each capture history at each simulation.

modelmat A full capture history matrix excluding the row corresponding to the dark figure.

popestsim Total population estimate given by the BCa method in each simulation.

BCaquantiles bootstrap confidence intervals given by the BCa method.

BICconf confidence interval given by the BIC method.

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

DiCiccio, T. J. and Efron, B. (1996). Bootstrap Confidence Intervals. *Statistical Science*, **40(3)**, 189-228.

Rivest, L-P. and Baillargeon, S. (2014) Rcapture. CRAN package. Available from Available from https://CRAN.R-project.org/package=Rcapture.

Examples

```
zdat=UKdat_5
BICandbootstrapsim(zdat,nsims=1000, nboot=100, pthresh=0.02, iseed=1234, noninformativelist=T)
```

`bicktopahatcal` *Acceleration factor calculation*

Description

Returns acceleration factors for each value of ntop for given modelorder

Usage

```
bicktopahatcal(z, modelorder)
```

Arguments

`z`	The output of `assemble_bic`, `bootstrapcal` and `jackknifecal`
`modelorder`	the order of the considered models

Value

the estimated acceleration factor

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
z=assemble_bic(Korea, checkexist=T)
z=jackknifecal(z, checkexist=T)
nmods= dim(z$jackabund)[1]
modscores = (1:nmods)
modelorder= order(modscores)
bicktopahatcal(z,modelorder)
```

`BICrank_tiebreak` *Return a BIC rank matrix breaking ties with each roles*

Description

This routine takes the ouput of `find_bic_rank_matrix` and produce a BIC rank matrix with a specified order of rank breaking ties with each previous column.

Usage

```
BICrank_tiebreak(BICmatrix_prop, k)
```

Arguments

`BICmatrix_prop`	The output from `find_bic_rank_matrix`
`k`	The order of BIC rank

Value

A BIC rank matrix with the last column breaking ties with the previous ones.

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
z=assemble_bic(Korea)
z=jackknifecal(z)
BICmatrix_prop= find_bic_rank_matrix(z)
BICmatrix_break = BICrank_tiebreak(BICmatrix_prop, 2)
```

`bootstrapcal` *Bootstrap abundance and bic*

Description

This routine takes the output from `assemble_bic` or `subsetmat` and returns bootstrap abundance matrix and BIC matrix. This version makes use of the `checkident.2`.

Usage

```
bootstrapcal(
  z,
  nboot = 1000,
  iseed = 1234,
  checkexist = T,
  saveinterval = Inf,
  savefile = "bootout.Rdata"
)
```

Arguments

`z`	Results from `assemble_bic` or `subsetmat`
`nboot`	The number of bootstrap replications.
`iseed`	Integer seed to allow for replicability.
`checkexist`	If `checkexist=TRUE`, check for existence, else it does not check for existence.
`saveinterval`	If this is set to a finite value, the output list `z` will be saved every time the number of replications is a multiple of it. A message will be printed every time it is saved.
`savefile`	The file to which the output will be saved if `saveinterval` is set to a finite value.

Value

The original input list `z` with the additional components

bootabund Bootstrap abundance matrix

bootbic Bootstrap BIC matrix

If there are already components with these names they will be overwritten.

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
z=assemble_bic(Korea)
bootstrapcal(z, checkexist=T, saveinterval=50, savefile="Koreabootresults.Rdata")
```

`bootstrap_mse` *A routine for naive user*

Description

A routine for naive user

Usage

```
bootstrap_mse(
  zdat,
  maxorder = dim(zdat)[2] - 2,
  nboot = 10000,
  iseed = 1234,
  alpha = c(0.025, 0.1, 0.9, 0.975)
)
```

Arguments

`zdat`	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history.
`maxorder`	Maximum order of models to be included.
`nboot`	Number of bootstrap replications.
`iseed`	seed for initialisation
`alpha`	Levels of confidence intervals to be constructed and assessed

Value

A list with the following components

confvals BCa confidence intervals for each considered model

probests Corresponding probabilities of the estimates

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
bootstrap_mse(Korea)
```

`boundary_captures` *Given a vector of captures, find those which are not in the vector but all of whose parents are*

Description

Call the resulting set the "boundary". Supposing that the current set of captures is a hierarchical model, that property will be preserved if a capture in the boundary is added to it. The routine is called internally by `find_neighbour_hierarchies`.

Usage

```
boundary_captures(kcap, nlists)
```

Arguments

`kcap`	An encoded capture history that corresponds to the row number of the capture history data set
`nlists`	The total number of lists

Value

a vector giving the encoded versions of the descendants

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
modelstr = "[12,23]"
nlists=4
zhierroots = convert_from_hierarchy(modelstr, F)
zhier = unique(ancestors(zhierroots, nlists))
boundary_captures(zhier,nlists)
```

`buildmodel` *Build model for multiple systems estimation*

Description

For multiple systems estimation model corresponding to a specified set of two-list effects, set up the GLM model formula and data matrix.

Usage

```
buildmodel(zdat, mX)
```

Arguments

`zdat`	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has zero counts.
`mX`	A $2 \times k$ matrix giving the k two-list effects to be included in the model. Each column of `mX` contains the numbers of the corresponding pair of lists. If `mX = 0`, then all two-list effects are included. If `mX = NULL`, no such effects are included and the main effects model is fitted.

Value

A list with components as below.

`datamatrix` A matrix with all possible capture histories, other than those equal to or containing non-overlapping pairs indexed by parameters that are within the model specified by `mX`. A non-overlapping pair is a pair of lists (i, j) such that no case is observed in both lists, regardless of whether it is present on any other lists. If (i, j) is within the model specified by `mX`, all capture histories containing both i and j are then excluded.

`modelform` The model formula suitable to be called by the Generalized Linear Model function `glm`. Model terms corresponding to non-overlapping pairs are not included, because they are handled by removing appropriate rows from the data matrix supplied to glm. The list of non-overlapping pairs are provided in `emptyoverlaps`. See Chan, Silverman and Vincent (2021) for details.

`emptyoverlaps` A matrix with two rows, whose columns give the indices of non-overlapping pairs of lists where the parameter indexed by the pair is within the specified model. The column names give the names of the lists corresponding to each pair.

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

Examples

```
data(NewOrl)
buildmodel(NewOrl, mX=NULL)
#Build a matrix that contains all two-list effects
m=dim(NewOrl)[2]-1
mX = t(expand.grid(1:m, 1:m)); mX = mX[ , mX[1,]<mX[2,]]
# With one two-list effect
buildmodel(NewOrl, mX=mX[,1])
#With three two-list effects
buildmodel(NewOrl, mX=mX[,1:3])
```

`buildmodelmatrix` *Build the model matrix based on particular data, as required to check for identifiability and existence of the maximum likelihood estimate*

Description

This routine builds a model matrix as required by the linear program check `checkident` and checks if the matrix is of full rank. In addition, for each individual list, and for each pair of lists included in the model, it returns the total count of individuals appearing on the specific list or lists whether or not in combination with other lists.

Usage

```
buildmodelmatrix(zdat, mX = NULL)
```

Arguments

`zdat`	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has zero count.
`mX`	A $2 \times k$ matrix giving the k two-list parameters to be included in the model. Each column of `mX` contains the numbers of the corresponding pair of lists. If `mX = 0`, then all two-list parameters are included. If `mX = NULL`, no such parameters are included and the main effects model is fitted.

Value

A list with components as below

`modmat` The matrix that maps the parameters in the model (excluding any corresponding to non-overlapping lists) to the log expected value of the counts of capture histories that do not contain non-overlapping pairs in the data.

`tvec` A vector indexed by the parameters in the model, excluding those corresponding to non-overlapping pairs of lists. For each parameter the vector contains the total count of individuals in all the capture histories that include both the lists indexed by that parameter.

`rankdef` The column rank deficiency of the matrix `modmat`. If `rankdef = 0`, the matrix has full column rank.

Examples

```
data(NewOrl)
buildmodelmatrix(NewOrl, mX=NULL)
```

`checkallmodels` *Check all possible models for existence and identifiability*

Description

This routine efficiently checks whether every possible model satisfies the conditions for the existence and identifiability of an extended maximum likelihood estimate. For identifiability it is only necessary to check the model containing all two-list effects. For existence, the approach set out in Chan, Silverman and Vincent (2021) is used. This uses the linear programming approach described in a more general and abstract context by Fienberg and Rinaldo (2012).

Usage

```
checkallmodels(zdat, nreport = 1024)
```

Arguments

`zdat`	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has zero count.
`nreport`	A message is printed for every `nreport` models checked. It gives the number of top level models to be checked altogether, and also the number of models found so far for which the estimate does not exist.

Details

If the extended maximum likelihood estimator exists for a particular model, then it will still exist if one or more overlapping pairs are removed from the model. This allows the search to be carried out efficiently, as follows:

1. Search over 'top-level' models which contain all overlapping pairs. The number of such models is 2^k, where k is the number of non-overlapping pairs, because there are 2^k possible choices of the set of non-overlapping pairs to include in the model.

2. For each top-level model, if the estimate exists there is no need to consider that model further. If the estimate does not exist, then a branch search is carried out over the tree structure obtained by successively leaving out overlapping pairs.

Value

The routine prints a message if the model with all two-list parameters is not identifiable. As set out above, it gives regular progress reports in the case where there are a large number of models to be checked.

If all models give estimates which exist, then a message is printed to that effect.

If there are models for which the estimate does not exist, the routine reports the number of such models and returns a matrix whose rows give the parameters included in them.

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

Fienberg, S. E. and Rinaldo, A. (2012). Maximum likelihood estimation in log-linear models. *Ann. Statist.* **40**, 996-1023. Supplementary material: Technical report, Carnegie Mellon University. Available from http://www.stat.cmu.edu/~arinaldo/Fienberg_Rinaldo_Supplementary_Material.pdf.

Examples

```
data(Artificial_3)
data(Western)
checkallmodels(Artificial_3)
checkallmodels(Western)
```

`checkident` *Check a model for the existence and identifiability of the maximum likelihood estimate*

Description

Apply the linear programming test as derived by Fienberg and Rinaldo (2012), and a calculation of the rank of the design matrix, to check whether a particular model yields an identifiable maximum likelihood estimate based on the given data. The linear programming problem is as described on page 3 of Fienberg and Rinaldo (2012), with a typographical error corrected. Further details are given by Chan, Silverman and Vincent (2021).

Usage

```
checkident(zdat, mX = 0, verbose = FALSE)
```

Arguments

zdat	Data matrix with $t+1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has observed count zero.
mX	A $2\times k$ matrix giving the k two-list parameters to be included in the model. Each column of mX contains the numbers of the corresponding pair of lists. If mX = 0, then all two-list interactions are included. If mX = NULL, no two-list parameters are included and the main effects model is fitted.
verbose	Specifies the output. If FALSE then the error code is returned. If TRUE then in addition the routine prints an error message if the model/data fail either of the two tests, and also returns both the error code and the lp object.

Value

If verbose=FALSE, then return the error code ierr which is 0 if there are no errors, 1 if the linear program test shows that the maximum likelihood estimate does not exist, 2 if it is not identifiable, and 3 if both tests are failed.

If verbose=TRUE, then return a list with components as below

ierr As described above.

zlp Linear programming object, in particular giving the value of the objective function at optimum.

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

Fienberg, S. E. and Rinaldo, A. (2012). Maximum likelihood estimation in log-linear models. *Ann. Statist.* **40**, 996-1023. Supplementary material: Technical report, Carnegie Mellon University. Available from http://www.stat.cmu.edu/~arinaldo/Fienberg_Rinaldo_Supplementary_Material.pdf.

Examples

```
data(Artificial_3)
#Build a matrix that contains all two-list effects
m=dim(Artificial_3)[2]-1
mX = t(expand.grid(1:m, 1:m)); mX = mX[ , mX[1,]<mX[2,]]
# When the model is not identifiable
checkident(Artificial_3,mX=mX, verbose=TRUE)
# When the maximum likelihood estimate does not exist
checkident(Artificial_3, mX=mX[,1],verbose=TRUE)
#Model passes both tests
checkident(Artificial_3, mX=mX[,2:3],verbose=TRUE)
```

checkident.1 *The Fienberg-Rinaldo linear program check for the existence of the estimates*

Description

This routine performs the Fienberg-Rinaldo linear program check in the framework of extended maximum likelihood estimates, the parameters estimates exist if and only if the return value of the check is nonzero

Usage

```
checkident.1(parset, datlist)
```

Arguments

parset	Either the hierarchical representation of the model, or the vector of the corresponding capture histories to the model.
datlist	The output of ingest_data on the data set

Value

The value of the linear program. The parameter estimates within the extended ML framework exist if and only if this value is nonzero.

References

Silverman, B. W., Chan, L. and Vincent, K., (2022). Bootstrapping Multiple Systems Estimates to Account for Model Selection

Fienberg, S. E. and Rinaldo, A. (2012). Maximum likelihood estimation in log-linear models. *Ann. Statist.* **40**, 996-1023. Supplementary material: Technical report, Carnegie Mellon University. Available from http://www.stat.cmu.edu/~arinaldo/Fienberg_Rinaldo_Supplementary_Material.pdf.

Examples

```
data(Korea)
datlist = ingest_data(Korea)
parset ="[0,0,1]"
checkident.1(parset,datlist)
```

checkident.2 *Carry out the Fienberg-Rinaldo procedure on an array of data vectors and a vector of models*

Description

Suppose we have a collection of different data outcomes on the same set of capture histories and a vector of models. Typically the data outcomes will be bootstrap replications. This routine finds the unique support patterns among the data and hence economises the task of finding which model/data combinations satisfy the Fienberg-Rinaldo condition

Usage

```
checkident.2(x, xcap, zmods)
```

Arguments

x	a matrix of data observations for a common capture matrix
xcap	the incidence matrix of the capture histories corresponding to the rows of x
zmods	a vector of models

Value

a matrix with rows corresponding to the models and columns to the columns of x, with elements taking the value T if the FR linear program for a vector of 0s and 1s with the same zero pattern as the x data yields a strictly positive value

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
z=assemble_bic(Korea)
z=bootstrapcal(z)
n2=dim(z$xdata)[2]
checkident.2(z$bootreplications, z$xdata[,-n2], dimnames(z$res)[[1]])
```

checkthetasubset *Check a subset of the parameter set theta*

Description

This routine leaves out a particular set of parameters corresponding to the two-list effects from the parameter set theta. For the resulting model, it constructs the linear programming problem to check whether the extended maximum likelihood estimates of the parameters exists. It is called internally by `checkallmodels`.

Usage

```
checkthetasubset(zset, amat, tvec, nlists)
```

Arguments

`zset`	set of indices that is not included, numbered among the two-list effects only
`amat`	a design matrix
`tvec`	vector of sufficient statistics
`nlists`	number of lists in the original capture-recapture matrix

Value

If the return result is `TRUE`, the linear program shows that the extended maximum likelihood estimate does not exist. If the return result is `FALSE`, the estimate exists.

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

child_captures *Find the "children" of a given capture history*

Description

Given any encoded capture history that corresponds to the row number of the capture history data set and the number of lists, find the encoded capture histories which are obtained by adding one more list in turn

Usage

```
child_captures(k, nlists)
```

Arguments

k	An encoded capture history that corresponds to the row number of the capture history data set
nlists	The total number of lists

Value

a vector giving the encoded versions of the children

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
child_captures(2,5)
child_captures(1,4)
```

---

convert_from_hierarchy

Find the vector of captures corresponding to a given hierarchical model

---

Description

Given a hierarchical model, find the vector of all the corresponding encoded captures

Usage

```
convert_from_hierarchy(modelstr, findancestors = T)
```

Arguments

modelstr	A given hierarchical model
findancestors	If T then find all the captures. If F then just return the encoded defining histories of the hierarchy

Value

The encoded capture histories that corresponds to the row number of the capture history data set

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
modelstr = "[12,23]"
convert_from_hierarchy(modelstr)
modelstr = "[12,3]"
convert_from_hierarchy(modelstr, findancestors=F)
```

`convert_to_hierarchy` *Find hierarchical representation of a vector of captures*

Description

Given a vector of encoded captures defining a hierarchical model, re-express it in hierarchical model form

Usage

```
convert_to_hierarchy(kcap, nlists)
```

Arguments

`kcap`	A vector of captures
`nlists`	The number of lists

Value

A hierarchical representation of the vector of encoded captures.

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
kcap=c(1,2,3,5,4)
nlists=3
convert_to_hierarchy(kcap, nlists)
```

count_triples *Count number of triples of overlapping lists*

Description

The routine counts the number of subsets of size three of lists such that every pair of lists in the triple overlaps. If the number is zero, then the model with all two-list effects is unidentifiable.

Usage

```
count_triples(zdat)
```

Arguments

zdat	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has zero count.

Value

a count of subsets of size three of lists such that every pair of lists in the triple overlaps.

Examples

```
data(Western)
data(Artificial_3)
count_triples(Western)
count_triples(Artificial_3)
```

decode_capture *Decode capture history*

Description

Given a capture history as a number and the number of lists, decode it into a logical vector giving presence or absence in the capture history.

Usage

```
decode_capture(k, nlists)
```

Arguments

k	The capture history to be decoded
nlists	The number of lists

Value

A logical vector of length nlists giving presence or absence in the capture history

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
decode_capture(2,5)
decode_capture(1,4)
```

---

descendants *Find the "descendants" of a given capture history*

Description

Given any encoded capture history, find all the encoded capture histories that include the original capture history and any other lists

Usage

```
descendants(k, nlists, omitk = FALSE)
```

Arguments

k	An encoded capture history
nlists	The total number of lists
omitk	Determine whether the original capture history is included as a descendant of itself. If omitk=T it is not.

Value

a vector giving the encoded versions of the descendants

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
descendants(2,5)
descendants(5,10)
```

`downhill_bootstrapcal` *Bootstrap downhill*

Description

Construct bootstrap replications and use the downhill fit method to obtain point estimates of total population sizes from each bootstrap sample.

Usage

```
downhill_bootstrapcal(
  xdata,
  nboot = 1000,
  iseed = 1234,
  checkid = T,
  verbose = F,
  maxorder = dim(xdata)[2] - 2
)
```

Arguments

`xdata`	original data matrix
`nboot`	number of bootstrap replicates
`iseed`	random seed
`checkid`	If it is T, then `checkident.1` is called and it performs the Fienberg-Rinaldo linear program check for the existence of the estimates
`verbose`	If TRUE, return the list of extra output from `downhill_fit`
`maxorder`	Maximum order of models to be included

Value

Point estimates of total population sizes from each bootstrap sample.

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
downhill_bootstrapcal(Korea)
```

`downhill_fit` *Conduct downhill search among hierarchical models starting from the main effects only.*

Description

Find a local optimum by downhill search among hierarchical models

Usage

```
downhill_fit(
  counts,
  desmat,
  maxorder = dim(desmat)[2] - 1,
  checkid = T,
  niter = 20,
  verbose = F
)
```

Arguments

`counts`	Observed counts for the capture histories defined by desmat
`desmat`	Incidence matrix defining the capture histories observed with counts given by counts
`maxorder`	Maximum order of models to be included
`checkid`	If it is T, then `checkident.1` is called and it performs the Fienberg-Rinaldo linear program check for the existence of the estimates
`niter`	Number of iterations
`verbose`	Specifies the output, if F then only returns the best value, if T, returns a more detailed list of objects

Value

A list with the following components

optimum_hierarchy Optimal hierarchical model

minimum_value hierarchical model with the minimum value

hierarchies_considered hierarhical models considered

function_values Values of function

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
xdata=Korea
counts = xdata[,dim(xdata)[2]]
desmat = xdata[,1:(dim(xdata)[2]-1)]
downhill_fit(counts, desmat)
```

`downhill_funs` *Downhill wrapper*

Description

The routine carries out pointwise, bootstrap and jackknife BCa confidence interval calculation and it calls the following routines `downhill_fit`, `downhill_bootstrapcal`, `downhill_jackknifecal` and `bcaconfvalues`.

Usage

```
downhill_funs(
  xdata,
  maxorder = dim(xdata)[2] - 2,
  checkid = T,
  verbose = F,
  nboot = 1000,
  alpha = c(0.025, 0.05, 0.95, 0.975)
)
```

Arguments

`xdata`	The data matrix
`maxorder`	Maximum order of models to be included
`checkid`	if TRUE, do the Fienburg-Renaldo test
`verbose`	if TRUE, return the list of extra output from `downhill_fit`
`nboot`	The number of bootstrap replications
`alpha`	Bootstrap quantiles of interests.

Value

A list with the following components

point_est A point estimate using the downhill fit approach

boot_res Point estimates of total population sizes from each bootstrap sample

jack_res The estimated acceleration factor

BCaconf_int BCa confidence intervals

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
xdata=Korea
downhill_funs(xdata)
```

`downhill_jackknifecal` *Jackknife downhill*

Description

It uses the downhill approach to calculate the jackknife abundance and returns the estimated acceleration factor

Usage

```
downhill_jackknifecal(xdata, checkid = T, maxorder = dim(xdata)[2] - 2)
```

Arguments

`xdata`	original data matrix
`checkid`	If it is T, then `checkident.1` is called and it performs the Fienberg-Rinaldo linear program check for the existence of the estimates
`maxorder`	Maximum order of models to be included

Value

the estimated acceleration factor

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
downhill_jackknifecal(Korea)
```

encode_capture *Encode capture history*

Description

Given a 0/1 capture history, encode it as number that corresponds to the row number of the capture history data set

Usage

```
encode_capture(z)
```

Arguments

z	The capture history to be encoded, as a logical vector or a vector of 0s and 1s

Value

The capture history encoded as a number that corresponds to the row number of the capture history data set

Examples

```
encode_capture(c(1,0,0,0,0))
encode_capture(c(1,1,1,1,0))
encode_capture(c(T,F,T,F))
```

estimatepopulation *Bootstrapping to evaluate confidence intervals using BCa methods*

Description

This routine implements the bootstrapping and jacknife approach as detailed in Section 3.3 of Chan, Silverman and Vincent (2021). It calls the routine `estimatepopulation.0` and so is the preferred routine to be called if a user wishes to estimate the population and obtain BCa confidence intervals.

Usage

```
estimatepopulation(
  zdat,
  nboot = 1000,
  pthresh = 0.02,
  iseed = 1234,
  alpha = c(0.025, 0.05, 0.1, 0.16, 0.84, 0.9, 0.95, 0.975),
  ...
)
```

Arguments

zdat	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has zero count.
nboot	Number of bootstrap replications.
pthresh	p-value threshold used in estimatepopulation.0.
iseed	seed for initialisation.
alpha	Bootstrap quantiles of interests.
...	other arguments which will be passed to estimatepopulation.0

Value

A list with components as below:

popest point estimate of the total population of the original data set

MSEfit model fitted to the data, in the format described in modelfit

bootreps point estimates of total population sizes from each bootstrap sample

ahat the estimated acceleration factor

BCaquantiles BCa confidence intervals

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

DiCiccio, T. J. and Efron, B. (1996). Bootstrap Confidence Intervals. *Statistical Science*, **40(3)**, 189-228.

Examples

```
data(Korea)
zdat=Korea
estimatepopulation(zdat,nboot=10)
```

`estimatepopulation.0` *Estimate the total population including the dark figure. If the user wishes to find bootstrap confidence intervals then the routine* `estimatepopulation` *should be used instead.*

Description

This routine estimates the total population size, which includes the dark figure, together with confidence intervals as specified. It also returns the details of the fitted model. The user can choose whether to fit main effects only, to fit a particular model containing specified two-list parameters, or to choose the model using the stepwise approach described by Chan, Silverman and Vincent (2021).

Usage

```
estimatepopulation.0(
  zdat,
  method = "stepwise",
  quantiles = c(0.025, 0.975),
  mX = NULL,
  pthresh = 0.02
)
```

Arguments

`zdat`	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has zero count.
`method`	If `method = "stepwise"` the stepwise method implemented in `stepwisefit` is used. If `method = "fixed"` then a specified fixed model is used; the model is then given by `mX`. If `method = "main"` then main effects only are fitted.
`quantiles`	Quantiles of interest for confidence intervals.
`mX`	A $2 \times k$ matrix giving the k two-list parameters to be included in the model if `method = "fixed"`. Each column of `mX` contains the numbers of the corresponding pair of lists. If `mX = 0`, then all two-list parameters are included. If `mX = NULL`, no two-list parameters are included and the main effects model is fitted. If only one two-list parameter is to be fitted, it is sufficient to specify it as a vector of length 2, e.g `mX=c(1,3)` for the parameter indexed by lists 1 and 3. If `method` is equal to `"stepwise"` or `"main"`, then `mX` is ignored.
`pthresh`	Threshold p-value used if `method = "stepwise"`.

Value

A list with components as below

`estimate` Point estimate and confidence interval estimates corresponding to specified quantiles.

`MSEfit` The model fitted to the data in the format described in `modelfit`.

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

Examples

```
data(NewOrl)
data(NewOrl_5)
estimatepopulation.0(NewOrl, method="stepwise", quantiles=c(0.025,0.975))
estimatepopulation.0(NewOrl_5, method="main", quantiles=c(0.01, 0.05,0.95, 0.99))
```

`find_bic_rank_matrix` *Find BIC rank matrix up to a specified number of bic ranks for a given data set*

Description

Find BIC rank matrix up to a specified number of bic ranks for a given data set

Usage

```
find_bic_rank_matrix(zsortbic, nbicranks = 5)
```

Arguments

`zsortbic`	Output from applying `assemble_bic` to the data
`nbicranks`	The number of bic ranks to be propagated

Value

A bic rank matrix encoding how many steps the models considered need to get to the optimal one.

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
zsortbic=assemble_bic(Korea, checkexist=T)
find_bic_rank_matrix(zsortbic, nbicranks=5)
```

`find_neighbour_hierarchies` *Find all neighbouring hierarchical model to a given one*

Description

Find all neighbouring hierarchical model to a given one

Usage

```
find_neighbour_hierarchies(
  modelstr,
  nlists = NA,
  keepmaineffects = T,
  maxorder = nlists - 1
)
```

Arguments

`modelstr`	Model string written in hierarchical form
`nlists`	Number of lists.
`keepmaineffects`	If T, keep the main effects. If F remove.
`maxorder`	Maximum order of models to be included

Value

neighbour hierarchical models

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
modelstr = "[12,23]"
find_neighbour_hierarchies(modelstr)
```

`find_unique_patterns` *Find unique patterns in matrix columns*

Description

Given a matrix (for example of bootstrap replications) construct the matrix of unique patterns of non-zeroes, together with a vector of pointers back to that matrix.

Usage

```
find_unique_patterns(x)
```

Arguments

x	a matrix

Value

The original data x with the additional components

yuniq matrix of unique patterns of non-zeroes/zeroes in the columns of x

pointers vector of length dim(x)[2] giving the column of yuniq corresponding to each column of x

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
z=assemble_bic(Korea)
z=bootstrapcal(z)
find_unique_patterns(z$bootreplications)
```

fit_hier_model — *Fit a hierarchical model taking account of possible sparsity*

Description

Fit a hierarchical model taking account of possible sparsity

Usage

```
fit_hier_model(xdatin, hiermod, bicRcap = T, checkid = F)
```

Arguments

xdatin	data obtained using ingest_data
hiermod	hierarchical model to fit
bicRcap	if T then use the Rcapture convention that the BIC sample size is the number of cases observed. Otherwise use the number of cells in the Poisson log linear model.
checkid	if T then checkident.1 is called inside the routine

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
xdatin = ingest_data(Korea)
fit_hier_model(xdatin,"[12,23]")
fit_hier_model(xdatin,"[12,3]")
```

gethiermodels — *Get a list of all hierarchical models for given number of lists and maximum order*

Description

Extracts from a larger vector of hierarchical models the ones satisfying the given criterion

Usage

```
gethiermodels(nlists, maxorder = nlists - 1, modelvec = hiermodels)
```

Arguments

nlists	Number of lists
maxorder	Maximum order of models to be returned (defaults to nlists-1)
modelvec	vector of hierarchical models (defaults to hiermodels)

Value

A list of models satisfying the given criteria

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(hiermodels)
# Five lists with maximum order of 4
gethiermodels(nlists=5,maxorder=4)
# Five lists with maximum order of 2
gethiermodels(nlists=5, maxorder=2)
```

---

hiermodels *Hierarchical models*

Description

Hierarchical models from two lists to six lists

Usage

```
hiermodels
```

Format

An object of class `character` of length 39783.

Details

These data provide hierarchical models constructed from two lists to six lists of maximal order 2.

`ingest_data` *Preliminary processing of a data matrix*

Description

Perform various preprocessing tasks on the data

Usage

```
ingest_data(xdat)
```

Arguments

`xdat`	Data matrix of the usual kind

Value

A list with the following elements

nobs Numbers of observations indexed by encoded histories

nstar For each capture history, total number of observations for that capture history and all its descendants

nlists Total number of lists

listnames Names of the lists, constructed to be A, B, ... if necessary

data The input data matrix

notestimable A vector indicating which parameters are not estimable, because they are strict descendants of parameters which would be already estimated to be $-\infty$ if they are included in the model

masterdesign The inclusion matrix as constructed by `make_master_design`

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
#Korea data
data(Korea)
ingest_data(Korea)
#Kosovo data
data(Kosovo)
ingest_data(Kosovo)
```

`investigateAIC` *Plot of simulation study*

Description

This routine reproduces Figure 1 of Chan, Silverman and Vincent (2021).

Usage

```
investigateAIC(nsim = 10000, Nsamp = 1000, seed = 1001)
```

Arguments

`nsim`	The number of simulation replications
`Nsamp`	The expected value of the total population size within each simulation
`seed`	The random number seed

Details

Simulations are carried out for two different three-list models. In one model, the probabilities of capture are 0.01, 0.04 and 0.2 for the three lists respectively, while in the other the probability is 0.3 on all three lists. In both cases, captures on the lists occur independently of each other, so there are no two-list effects. The first model is chosen to be somewhat more typical of the sparse capture case, of the kind which often occurs in the human trafficking context, while the second, more reminiscent of a classical mark-recapture study.

The probability of an individual having each possible capture history is first evaluated. Then these probabilities are multiplied by `Nsamp = 1000` and, for each simulation replicate, Poisson random values with expectations equal to these values are generated to give a full set of observed capture histories; together with the null capture history the expected number of counts (population size) is equal to `Nsamp`. Inference was carried out both for the model with main effects only, and for the model with the addition of a correlation effect between the first two lists. The reduction in deviance between the two models was determined. Ten thousand simulations were carried out.

Checking for compliance with the conditions for existence and identifiability of the estimates shows that a very small number of the simulations for the sparse model (two out of ten thousand) fail the checks for existence even within the extended maximum likelihood context. Detailed investigation shows that in neither of these cases is the dark figure itself not estimable; although the parameters themselves cannot all be estimated, there is a maximum likelihood estimate of the expected capture frequencies, and hence the deviance can still be calculated.

The routine produces QQ-plots of the resulting deviance reductions against quantiles of the χ^2_1 distribution, for `nsim` simulation replications.

Value

An `nsim` $\times$ 2 matrix giving the changes in deviance for each replication for each of the two models.

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

`jackknifecal` *Jackknife abundance and Jackknife bic*

Description

This routine takes the output from `subsetmat` or from `assemble_bic` and returns the jackknife abundance matrix and jackknife BIC matrix.

Usage

```
jackknifecal(z, checkexist = T)
```

Arguments

`z`	Results from `assemble_bic` or `subsetmat`.
`checkexist`	If `checkexist=TRUE`, check for existence in cases where the jackknife introduces an additional zero, else it does not check for existence. Note that in the current version it is assume that models for which the fit doesn't exist for the original data have already been excluded.

Value

A list with the following components

jackabund Jackknife abundance matrix

jackbic Jackknife BIC matrix

countsobserved Capture counts in the same order as the columns of `jackabund` and `jackbic`

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
z=assemble_bic(Korea)
jackknifecal(z,checkexist=T)
```

Korea *Korea data*

Description

Korean woman held in sexual slavery by the Japanese military

Usage

```
Korea
```

Format

An object of class `matrix` (inherits from `array`) with 7 rows and 4 columns.

Details

These data are collected into three lists. Full details are given in Figure 1 and the Data section of Ball et al. (2018)

References

Ball, P., Shin, E. H-S. and Yang, H. (2018). There may have been 14 undocumented Korean "comfort women" in Palembang, Indonesia. Technical Report, Humans Rights Data Analysis Group. Available from https://hrdag.org/wp-content/uploads/2018/12/KP-Palemban-ests.pdf

Kosovo *Kosovo data*

Description

Data on 4400 observed killings in the Kosovo war between 20 March and 22 June 1999

Usage

```
Kosovo
```

Format

An object of class `data.frame` with 15 rows and 5 columns.

Details

These data give the numbers of cases on each possible combination of four lists. The lists are labelled as follows: EXH = exhumations; ABA = American Bar Association Central and East European Law Initiative; OSCE = Organization for Security and Cooperation in Europe; HRW = Human Rights Watch. All 15 combinations have a nonzero count.

References

Ball, P., W. Betts, F. Scheuren, J. Dudukovich, and J. Asher (2002). Killings and Refugee Flow in Kosovo March-June 1999. American Association for the Advancement of Science. A Report to the International Criminal Tribunal for the Former Yugoslavia.

`ktopBCa` *Find BCa confidence intervals using ktop idea.*

Description

Find BCa confidence intervals using ktop idea.

Usage

```
ktopBCa(
  z,
  BICmatrix_break,
  alpha = c(0.025, 0.05, 0.1, 0.16, 0.2, 0.5, 0.8, 0.84, 0.9, 0.95, 0.975),
  maxorder = Inf,
  BCaFD = FALSE
)
```

Arguments

`z`	The output of `assemble_bic`, `bootstrapcal` and `jackknifecal`
`BICmatrix_break`	output from `BICrank_tiebreak`
`alpha`	Levels of confidence intervals to be constructed and assessed
`maxorder`	Maximum order of models to be included
`BCaFD`	If T, return also the probabilities of the estimates

Value

A list with the following components

confvals BCa confidence intervals for each considered model

probests Corresponding probabilities of the estimates

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
z=assemble_bic(Korea)
z=bootstrapcal(z)
z=jackknifecal(z)
BICmatrix_prop= find_bic_rank_matrix(z)
BICmatrix_break = BICrank_tiebreak(BICmatrix_prop, 2)
ktopBCa(z,BICmatrix_break)
```

---

make_master_design *Set up the inclusion matrix for all possible capture histories*

Description

This is the master design matrix which maps parameters to observations. Rows correspond to observations and columns to parameters.

Usage

```
make_master_design(nlists)
```

Arguments

nlists	The number of lists

Value

A matrix whose (i, j) element is 1 if the expected log of observation i depends on parameter j, in other words if j is an ancestor of i.

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
#Create master design matrix with 5 lists
make_master_design(5)
#Create master design matrix with 3 lists
make_master_design(3)
```

modelfit *Fit a specified model to multiple systems estimation data*

Description

This routine fits a specified model to multiple systems estimation data, taking account of the possibility of empty overlaps between pairs of observed lists.

Usage

```
modelfit(zdat, mX = NULL, check = TRUE)
```

Arguments

zdat	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has zero count.
mX	A $2 \times k$ matrix giving the k two-list parameters to be included in the model. Each column of mX contains the numbers of the corresponding pair of lists. If mX = 0, then all two-list parameters are included. If mX = NULL, no two-list parameters are included and the main effects model is fitted. If only one two-list parameter is to be fitted, it may be specified as a vector of length 2, e.g mX=c(1,3) for the parameter corresponding to lists 1 and 3.
check	If check = TRUE check first of all if the maximum likelihood estimate exists and is identifiable, using the routine checkident. If either condition fails, print an appropriate error message and return the error code.

Value

A list with components as below

fit Details of the fit of the specified model as output by glm. The Akaike information criterion is adjusted to take account of the number of parameters corresponding to non-overlapping pairs.

emptyoverlaps Matrix with two rows, giving the list pairs within the model for which no cases are observed in common. Each column gives the indices of a pair of lists, with the names of the lists in the column name.

poisspempty the Poisson p-values of the parameters corresponding to non-overlapping pairs.

Examples

```
data(NewOrl)
modelfit(NewOrl,mX= c(1,3), check=TRUE)
```

modelorder *Order of models*

Description

This routine returns the order of the model given the model character string

Usage

```
modelorder(x)
```

Arguments

x	a model character string

Value

the order of the model character string

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
x="[1,2,3,4,5]"
modelorder(x)
y="[12,24,25,35,45]"
modelorder(y)
```

Ned *The Netherlands data*

Description

Victims related to human trafficking in the Netherlands

Usage

```
Ned
```

Format

An object of class `data.frame` with 24 rows and 7 columns.

Details

These data are collected into six lists. Full details are given in Table 2 of Silverman (2020).

References

Silverman, B. W. (2020). Model fitting in Multiple Systems Analysis for the quantification of Modern Slavery: Classical and Bayesian approaches *Journal of Royal Statistical Society: Series A*, **183(3)**, 691-736, Available from https://rss.onlinelibrary.wiley.com/doi/full/10.1111/rssa.12505

Ned_5 *Netherlands data five list version*

Description

Netherlands data consolidated into five lists

Usage

```
Ned_5
```

Format

An object of class `data.frame` with 17 rows and 6 columns.

Details

This reduces the Netherlands data `Ned` into five lists, constructed by combining the two smallest lists I and O into a single list.

NewOrl *New Orleans data*

Description

Victims related to human trafficking in Greater New Orleans

Usage

```
NewOrl
```

Format

An object of class `data.frame` with 19 rows and 9 columns.

Details

These data are collected into 8 lists. For reasons of confidentiality the lists are only labelled as A, B, ..., H. Full details are given in Bales, Murphy and Silverman (2020).

References

K. Bales, L. Murphy and B. W. Silverman (2020). How many trafficked people are there in Greater New Orleans? *Journal of Human Trafficking*, **6(4)**, 375-384, available from https://www.tandfonline.com/doi/full/10.1080/23322705.2019.1634936

`NewOrl_5` *New Orleans data five list version*

Description

New Orleans data consolidated into five lists

Usage

```
NewOrl_5
```

Format

An object of class `data.frame` with 14 rows and 6 columns.

Details

This reduces the New Orleans data `NewOrl` into five lists, constructed by combining the four smallest lists B, E, F and G into a single list.

`ntopBCa` *Find BCa confidence intervals for all possible ntop*

Description

Find BCa confidence intervals for all possible ntop

Usage

```
ntopBCa(
  z,
  alpha = c(0.025, 0.05, 0.1, 0.16, 0.2, 0.5, 0.8, 0.84, 0.9, 0.95, 0.975),
  maxorder = Inf,
  BCaFD = FALSE
)
```

Arguments

z	The output of assemble_bic, bootstrapcal and jackknifecal
alpha	Levels of confidence intervals to be constructed and assessed
maxorder	Maximum order of models to be included
BCaFD	If T, return also the probabilities of the estimates

Value

A list with the following components

confvals BCa confidence intervals for each considered model

probests Corresponding probabilities of the estimates

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
data(Korea)
z=assemble_bic(Korea, checkexist=T)
z=bootstrapcal(z, checkexist=T)
z=jackknifecal(z,checkexist=T)
ntopBCa(z)
```

---

ordercaptures *Order capture histories*

Description

Given a matrix with capture histories only, the routine orders the capture histories first by the number of 1s in the capture history and then lexicographically by columns.

Usage

```
ordercaptures(zmat)
```

Arguments

zmat	Data matrix with t columns. The t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. Where a capture history is not explicitly listed, it is assumed that it has observed count zero.

Value

A data matrix that is ordered first by the number of 1s in the capture history and then lexicographically by columns.

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

Examples

```
#UK 5 list data
data(UKdat_5)
# Obtain matrix with capture histories only
UKdat_5_cap <- UKdat_5[, 1:dim(UKdat_5)[2]-1]
ordercaptures(UKdat_5_cap)
```

---

parent_captures *Find the "parents" of a given capture history*

Description

Given any encoded capture history and the number of lists, find the encoded capture histories which are obtained by leaving out just one list in turn

Usage

```
parent_captures(k, nlists = 10)
```

Arguments

k	An encoded capture history that corresponds to the row number of the capture history data set
nlists	The total number of lists

Value

a vector giving the encoded versions of the parents

References

Silverman, B. W., Chan, L. and Vincent, K., (2024). Bootstrapping Multiple Systems Estimates to Account for Model Selection *Statistics and Computing*, **34(44)**, Available from https://doi.org/10.1007/s11222-023-10346-9.

Examples

```
parent_captures(2,10)
parent_captures(1,4)
```

`removenoninformativelists` *Remove non-informative list*

Description

The routine cleans up the data set by removing any lists that contain no data, any lists which contain all the observed data, and any list whose results duplicate those of another list. If as a result the original data set has no list left, it returns a matrix with the value of the total count.

Usage

```
removenoninformativelists(zdat)
```

Arguments

`zdat`	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history.

Value

data matrix that contains no duplicate lists, no lists with no data, and no lists that contain all the observed data. If all lists are removed, the total count is returned.

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

`sortmodelsbic` *Models BICS, abundance and maxorder.*

Description

This routine sorts the models in increasing order according to their BICs, returns the sorted models with their corresponding BICs and abundance. The original data as well as the maxorder of the models considered are returned as well.

Usage

```
sortmodelsbic(
  xdata,
  maxorder = dim(xdata)[2] - 2,
  checkexist = T,
  removeFRfail = T,
  ...
)
```

Arguments

`xdata`	The original data matrix with capture histories and counts.
`maxorder`	Maximum order of models to be included
`checkexist`	If T then the Fienberg-Rinaldo condition is checked for each model
`removeFRfail`	If checkexist is T then models which fail the FR condition are removed from the results
`...`	Parameters to be fed to `gethiermodels`.

Value

A list with the following components

res A matrix with the models considered, their abundance, BIC and their order, sorted into increasing order of BIC

xdata The original data matrix with capture histories and counts.

maxorder Order parameter that was feed into `gethiermodels`

Examples

```
data(hiermodels)
data(Korea)
sortmodelsbic(Korea, checkexist=T)
```

`stepwisefit` *Stepwise fit using Poisson p-values.*

Description

Starting with a model with main effects only, two-list parameters are added one by one. At each stage the parameter with the lowest p-value is added, provided that p-value is lower than `pthresh`, and provided that the resulting model does not fail either of the tests in `checkident`.

Usage

```
stepwisefit(zdat, pthresh = 0.02)
```

Arguments

`zdat`	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has zero count.
`pthresh`	this is the threshold below which the p-value of the newly added parameter needs to be in order to be included in the model. If `pthresh = 0` then the model with main effects only is returned.

Details

For each candidate two-list parameter for possible addition to the model, the p-value is calculated as follows. The total of cases occurring on both lists indexed by the parameter (regardless of whether or not they are on any other lists) is calculated. On the null hypothesis that the effect is not included in the model, this statistic has a Poisson distribution whose mean depends on the parameters within the model. The one-sided Poisson p-value of the observed statistic is calculated.

Value

A list with components as below

`fit` Details of the fit of the specified model as output by `glm`. The Akaike information criterion is adjusted to take account of the number of parameters corresponding to non-overlapping pairs.

`emptyoverlaps` Matrix with two rows, giving the list pairs within the model for which no cases are observed in common. Each column gives the indices of a pair of lists, with the names of the lists in the column name.

`poisspempty` the Poisson p-values of the parameters corresponding to non-overlapping pairs.

Examples

```
data(NewOrl)
stepwisefit(NewOrl, pthresh=0.02)
```

subsetmat *Subset matrix*

Description

The idea of this routine is to reduce either or both of ntopmodels and maxorder without the need to recalculate any actual model fits.

Usage

```
subsetmat(z, ntopmodels = Inf, maxorder = Inf)
```

Arguments

z	output from assemble_bic or a list output from applying assemble_bic, jackknifecal and bootstrapcal.
ntopmodels	number of top models. If (taking into account any change in the maximum order of models) there are fewer than ntopmodels in the data supplied, then it will be reduced to that value. If it is not specified then there will be no reduction in the number.
maxorder	the maximum order of the models to be considered. If not specified, it will be set to the corresponding value in the input data.

Details

The routine subsets the results matrix as part of the output given by assemble_bic based on specified parameters ntopmodels and maxorder. It returns the subsetted matrix, original data matrix with capture histories and counts and the new actual value of maxorder (reducing it from the input value if necessary or if the default input value of ∞ is used).

Value

A list with the following components

res a matrix containing models being considered, abundance, BIC and their ordered after being subsetted by maxorder and ntopmodels

xdata Original data matrix with counts and capture histories

maxorder The maximum order of models considered after subsetting

jackabund Jackknife abundance matrix, subsetted by maxorder and ntopmodels

jackbic Jackknife BIC matrix, subsetted by maxorder and ntopmodels

countsobserved Capture counts in the same order as the columns of jackabund and jackbic

bootabund Bootstrap abundance matrix, subsetted by maxorder and ntopmodels

bootbic Bootstrap BIC matrix, subsetted by maxorder and ntopmodels

If the input only has the output from assemble_bic, the last five items of the list do not appear.

Examples

```
data(UKdat_5)
#Example 1
z=assemble_bic(UKdat_5)
subsetmat(z,ntopmodels=100,maxorder=2)
subsetmat(z)
#Example 2
z=assemble_bic(UKdat_5)
z=jackknifecal(z)
z=bootstrapcal(z)
subsetmat(z,ntopmodels=100,maxorder=2)
```

`subsetsearch` *Search subsets for a property which is inherited in a particular way*

Description

The following routine returns a list of all subsets of a given set for which a specified property is `TRUE`. It is assumed that if the property is `FALSE` for any particular subset, it is also `FALSE` for all supersets of that subset. This enables a branch search strategy to be used to obviate the need to search over supersets of subsets already eliminated from consideration. It is used within the hierarchical search step of `checkallmodels`.

Usage

```
subsetsearch(n, checkfun, testnull = TRUE, ...)
```

Arguments

`n`	an integer such that the search is over all subsets of $\{1, ..., n\}$
`checkfun`	a function which takes arguments `zset`, a subset of $\{1, ..., n\}$ and ... The function returns the value `TRUE` or `FALSE`. It needs to have the property that if it is `FALSE` for any particular `zset`, it is also `FALSE` for all supersets of `zset`.
`testnull`	If `TRUE`, then include the null subset in the search. If `FALSE`, do not test the null subset.
`...`	other arguments

Value

A list of all subsets for which the value is `TRUE`

References

Chan, L., Silverman, B. W., and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges when there are Non-Overlapping Lists. *Journal of American Statistcal Association*, **116(535)**, 1297-1306, Available from https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1708748.

tidylists *Produce a data matrix with a unique row for each capture history*

Description

This routine finds rows with the same capture history and consolidates them into a single row whose count is the sum of counts of the relevant rows. If `includezerocounts = TRUE` then it also includes rows for all the capture histories with zero count; otherwise these are all removed.

Usage

```
tidylists(zdat, includezerocounts = FALSE)
```

Arguments

`zdat`	Data matrix with $t + 1$ columns. The first t columns, each corresponding to a particular list, are 0s and 1s defining the capture histories observed. The last column is the count of cases with that particular capture history. List names A, B, ... are constructed if not supplied. Where a capture history is not explicitly listed, it is assumed that it has zero count.
`includezerocounts`	If `FALSE` then remove rows corresponding to capture histories with zero count. If `TRUE` then include all possible capture histories including those with zero count, excluding the all-zero row corresponding to the dark figure.

Value

A data matrix in the form specified above, including all capture histories with zero counts if `includezerocounts=TRUE`.

Examples

```
data(NewOrl)
tidylists(NewOrl,includezerocounts=TRUE)
```

UKdat *UK data*

Description

Data from the UK 2013 strategic assessment

Usage

```
UKdat
```

Format

An object of class `data.frame` with 25 rows and 7 columns.

Details

This is a table of six lists used in the research published by the Home Office as part of the strategy leading to the Modern Slavery Act 2015. The data are considered in six lists, labelled as follows: LA–Local authorities; NG–Non-government organisations; PF–Police forces; GO–Government organisations; GP–General public; NCA–National Crime Agency. Each of the first six columns in the data frame corresponds to one of these lists. Each of the rows of the data frame corresponds to a possible combination of lists, with value 1 in the relevant column if the list is in that particular combination. The last column of the data frame states the number of cases observed in that particular combination of lists. Combinations of lists for which zero cases are observed are omitted.

References

https://www.gov.uk/government/publications/modern-slavery-an-application-of-multiple-systems-estima

---

`UKdat_5` *UK data five list version*

Description

UK data consolidated into five lists

Usage

```
UKdat_5
```

Format

An object of class `data.frame` with 18 rows and 6 columns.

Details

This reduces the UK data `UKdat` into five lists, constructed by combining the PF and NCA lists into a single PFNCA list

Western *Victims related to sex trafficking in a U.S. Western site*

Description

These data are collected into 5 lists. For reasons of confidentiality the lists are only labelled as A, B, C, D and E. Full details are given in Farrell, Dank, Kfafian, Lockwood, Pfeffer, Hughes and Vincent (2019).

Usage

```
Western
```

Format

An object of class `data.frame` with 13 rows and 6 columns.

References

Farrell, A., Dank, M., Kfafian, M., Lockwood, S., Pfeffer, R., Hughes, A., and Vincent, K. (2019). Capturing human trafficking victimization through crime reporting. Technical Report 2015-VF-GX-0105, National Institute of Justice. Available from https://www.ncjrs.gov/pdffiles1/nij/grants/252520.pdf.

Index