Package: MultipleSystemsEstimation (via r-universe)

October 22, 2026

Title Methods for Multiple Systems Estimation

Version 1.3.2

Description Methods for multiple systems estimation using Poisson log-linear models, with particular attention to sparse capture data, model selection and its associated uncertainty, and Bayesian thresholding of interaction effects. The package implements and extends methodology from Silverman (2020) <doi:10.1111/rssa.12505>, Chan, Silverman and Vincent (2021) <doi:10.1080/01621459.2019.1708748>, and Silverman, Chan and Vincent (2024) <doi:10.1007/s11222-023-10346-9>.

URL https://bernardsilverman.github.io/MultipleSystemsEstimation/, https://bernardsilverman.co.uk/contact

Depends R (>= 4.2.0)

License GPL (>= 2)

Encoding UTF-8

LazyData true

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Imports lpSolve, MCMCpack

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Config/roxygen2/version 8.1.0

Maintainer Bernard W. Silverman <bernard.silverman@stats.ox.ac.uk>

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

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

Date/Publication 2026-10-22 19:20:25 UTC

RemoteUrl https://github.com/cran/MultipleSystemsEstimation

RemoteRef HEAD

RemoteSha b263729c5decbfd08f6118e3df7b2ff69bd69111

Contents

Artificial_3 . . . 2
check_extended_MLE . . . 3
estimate_population . . . 4
estimate_population_bayesthresh . . . 6
estimate_population_bic . . . 9
estimate_population_fixed . . . 12
estimate_population_stepwise . . . 14
find_neighbour_hierarchies . . . 16
get_hierarchical_models . . . 18
Korea . . . 19
Kosovo . . . 19
Ned . . . 20
Ned_5 . . . 20
NewOrl . . . 21
NewOrl_5 . . . 21
tidy_lists . . . 22
UKdat . . . 23
UKdat_5 . . . 24
Western . . . 24

Index **25**

`Artificial_3` *Artificial three-list data*

Description

An artificial three-list data set used to demonstrate failures of the conditions tested by `check_extended_MLE`. The data appear in Table 2 of Chan, Silverman and Vincent (2021).

Usage

```
Artificial_3
```

Format

A data frame with 4 rows and 4 columns. Columns `A`, `B` and `C` are binary list-membership indicators. Column `n` gives the number of cases having each observed capture history.

Details

If all three two-list interactions are included, `check_extended_MLE()` returns status 2: the extended MLE exists, but the model matrix is not of full rank and the parameters are not identifiable. If the model contains AB, either alone or together with AC or BC, the extended MLE does not exist. The main-effects model and models containing either or both of AC and BC, but not AB, pass both checks.

References

Chan, L., Silverman, B. W. and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges When There Are Nonoverlapping Lists. *Journal of the American Statistical Association*, **116**(535), 1297–1306. doi:10.1080/01621459.2019.1708748.

`check_extended_MLE` *Check existence and identifiability of the extended MLE*

Description

Checks whether a hierarchical log-linear model satisfies both the Fienberg–Rinaldo condition for existence of the extended maximum-likelihood estimate and the rank condition required for identifiable model parameters.

Usage

```
check_extended_MLE(data, model)
```

Arguments

`data`	Capture history data in the standard package format, with one indicator column for each list followed by a count column.
`model`	A character string specifying a hierarchical model, such as `"[12,13,23]"`.

Details

Two distinct conditions are checked.

The Fienberg–Rinaldo condition determines whether the required extended maximum-likelihood fit exists, allowing for parameters corresponding to sparse interactions to be estimated at minus infinity.

The relevant model matrix is also checked for full column rank. Failure of this condition means that the model parameters are not identifiable.

The two checks are carried out separately, so either one or both may fail.

Value

A single integer status code:

`0` Both conditions are satisfied.

`1` The Fienberg–Rinaldo condition fails.

`2` The model parameters are not identifiable.

`3` Both conditions fail.

References

Fienberg, S. E. and Rinaldo, A. (2012). Maximum likelihood estimation in log-linear models. *The Annals of Statistics*, **40**, 996–1023.

Chan, L., Silverman, B. W. and Vincent, K. (2021). Multiple Systems Estimation for Sparse Capture Data: Inferential Challenges When There Are Nonoverlapping Lists. *Journal of the American Statistical Association*, **116**(535), 1297–1306. doi:10.1080/01621459.2019.1708748.

Examples

```
data(Artificial_3)

check_extended_MLE(
  Artificial_3,
  "[12,13,23]"
)

# The result is 2: the existence condition is satisfied,
# but the model parameters are not identifiable.
```

`estimate_population` *Estimate population size*

Description

Provides a common interface to the principal population-estimation methods in **MultipleSystemsEstimation**.

Usage

```
estimate_population(
  zdat,
  method = c("auto", "bic", "stepwise", "fixed", "bayesthresh"),
  ...
)
```

Arguments

`zdat`	Capture history data. A matrix with one more column than the number of lists. Each row corresponds to an observed capture history with 1 in the columns corresponding to the capture lists in the history, and 0 otherwise. The last column contains the observed count for the capture history.
`method`	Estimation method. One of `"auto"`, `"bic"`, `"stepwise"`, `"bayesthresh"`, or `"fixed"`. The individual methods and their available arguments are documented in `estimate_population_bic`, `estimate_population_stepwise`, `estimate_population_bayesthre` and `estimate_population_fixed`.

... Additional arguments passed to the selected estimation function. See estimate_population_bic, estimate_population_stepwise, estimate_population_bayesthresh, and estimate_population_fixed for the arguments available for each method.

Details

The default method is "auto". This selects the BIC-based method when there are no more than five lists, and the stepwise method when there are six or more lists.

For six-list data, the BIC method remains available by specifying method = "bic", but exhaustive BIC enumeration is computationally burdensome. Accordingly, method = "auto" selects the stepwise method for six-list data.

The estimation method can be selected explicitly using method = "bic", "stepwise", "bayesthresh", or "fixed".

Through their respective maxorder arguments, the stepwise, BIC and Bayesian-threshold methods can all be restricted to two-list interactions, or consider higher-order interactions.

Method-specific arguments are passed through ... to the selected estimation function. See the documentation for the individual methods for details of the available arguments and their defaults.

Value

A list with components:

input A list containing the original call and data.

method The method actually used, including after method = "auto" has been resolved.

estimate A named numeric vector containing the estimated dark_figure and total population.

fitted_model The supplied, selected or retained model in hierarchy notation.

uncertainty Confidence or credible endpoints for the dark figure and total population, or an explanatory character string when bootstrap inference was not requested.

details Method-specific supporting output when requested, and "not requested" otherwise.

See Also

estimate_population_bic, estimate_population_stepwise, estimate_population_bayesthresh, estimate_population_fixed

Examples

```
data(Korea)

# Three lists: automatically uses the BIC method.
estimate_population(Korea)

# Pass BIC-specific arguments through ...
estimate_population(
  Korea,
  method = "bic",
  nboot = 100
)
```

```
# Pass stepwise-specific arguments through ...
estimate_population(
  Korea,
  method = "stepwise",
  pthresh = 0.02
)

# The bayesthresh method
   estimate_population(
    Western,
    method = "bayesthresh",
    burnin = 100,
    mcmc = 1000
  )

# Pass a fixed-model specification through ...
estimate_population(
  Korea,
  method = "fixed",
  model = "[12,23]"
)
```

`estimate_population_bayesthresh`

Bayesian-thresholding multiple systems estimation

Description

Fits the Bayesian-threshold estimator of Silverman (2020), based on a Poisson log-linear model for the capture history counts.

The intercept and main effects have independent improper flat priors. For the interaction parameters, a proper normal prior is used by default; an improper flat prior can instead be specified. For the Bayesian parts of the procedure, inference is carried out by Markov chain Monte Carlo, calling `MCMCpack::MCMCpoisson()`.

The method begins by including all two-list interactions. The interactions are then thresholded by discarding those for which the absolute ratio of posterior mean to posterior standard deviation is below `threshold`. The model containing the retained interactions is then re-estimated, and the posterior distribution of the total population is obtained by adding the observed population to the posterior estimate of the unobserved cell.

Usage

```
estimate_population_bayesthresh(
  zdat,
  prior = "proper",
  prior_variance = 1,
```

```
  threshold = 2,
  maxorder = 2,
  alpha = c(0.025, 0.1, 0.9, 0.975),
  return_details = FALSE,
  ...
)
```

Arguments

`zdat`	Capture history data in the standard package format.
`prior`	Either `"proper"` (the default) or `"improper"`, specifying the prior for the interaction parameters.
`prior_variance`	Prior variance for interaction parameters when a proper prior is used.
`threshold`	Threshold applied to the absolute ratio of posterior mean to posterior standard deviation for interaction parameters.
`maxorder`	Maximum interaction order, 2, 3 or 4.
`alpha`	Numeric vector of cumulative probability levels at which posterior credible-interval endpoints are to be reported. The default is `c(0.025, 0.1, 0.9, 0.975)`. The posterior median used for the point estimate is calculated separately and need not be included in `alpha`.
`return_details`	Logical. If `TRUE`, include thresholding information, effects at minus infinity, and the full posterior sample of total population size. The default is `FALSE`.
`...`	Additional arguments passed to `MCMCpack::MCMCpoisson()`. Useful arguments include: • `burnin`: number of burn-in iterations; default `1000`. • `mcmc`: number of MCMC iterations retained after burn-in; default `10000`. • `thin`: thinning interval; default `1`. • `tune`: Metropolis tuning parameter; default `1.1`. • `seed`: random-number seed. The default `NA` causes `MCMCpack::MCMCpoisson()` to use the Mersenne Twister generator with fixed seed 12345, so repeated calls with otherwise identical arguments are reproducible. Supply another seed to obtain a different MCMC run.

Details

By default, the interaction parameters have independent zero-centred normal priors, with variance specified by `prior_variance`. Setting `prior = "improper"` instead assigns independent improper flat priors to these parameters.

Under improper priors, sparse effects may be handled separately at minus infinity. For example, if two lists are non-overlapping and their two-list interaction is included in the model, its parameter has a posterior distribution concentrated at minus infinity. Such effects are retained in the model and reported in `minus_infinity_effects`, but their parameters are excluded from the MCMC.

Thresholding proceeds in increasing order of interaction, starting with the two-list interactions. If `maxorder >= 3`, a three-list interaction is eligible for consideration only if all three of its constituent two-list interactions have been retained with *finite* parameters. Thus, under improper priors, a three-list interaction containing a non-overlapping pair of lists cannot be eligible.

The model containing the retained two-list interactions, including any effects handled at minus infinity, and all eligible three-list interactions is then fitted. The eligible three-list interactions are thresholded, except that, under improper priors, any newly minimal sparse effects are handled separately at minus infinity.

If `maxorder = 4`, the procedure continues to four-list interactions after thresholding the three-list interactions. A four-list interaction is eligible only if all four of its constituent three-list interactions have been retained with *finite* parameters. Eligible four-list interactions are fitted and thresholded in the same way.

Under improper priors, the initial model containing all two-list interactions is checked for identifiability and against the Fienberg–Rinaldo existence criterion before thresholding begins. If either check fails, the procedure stops.

Before extending the analysis to three-list interactions, the model containing the retained two-list interactions and all eligible three-list interactions is also checked against the Fienberg–Rinaldo criterion. If this check fails, the extension is not carried out. Instead, the completed two-list analysis is returned, together with the eligible triples. The analogous check is applied before extending to four-list interactions; failure returns the completed three-list analysis.

These existence checks are needed because, under improper priors on the interaction parameters, a model without an extended maximum-likelihood estimate will not have a proper posterior. However, results of Forster (2010) show that proper priors on the interaction parameters can yield a proper posterior even when a finite maximum-likelihood estimate does not exist. Consequently, when proper priors are used, Bayesian fitting and thresholding proceed without applying the Fienberg–Rinaldo criterion.

When proper priors are used for the interaction parameters, the improper priors on the intercept and main effects are approximated internally by independent zero-centred normal priors with very large variance. This is an implementation device arising from the way `MCMCpack::MCMCpoisson()` specifies its prior distribution, rather than a change to the underlying prior specification.

Value

A list with components:

`input` A list containing the original `call` and `data`.

`method` The character string `"bayesthresh"`.

`estimate` A named numeric vector containing posterior median estimates of the `dark_figure` and `total` population.

`fitted_model` The retained model in hierarchy notation.

`uncertainty` A two-row matrix of posterior quantiles for the dark figure and total population at the probabilities in `alpha`.

`details` If `return_details = TRUE`, a list containing `minus_infinity_effects`, pairwise `threshold_statistics`, applicable `eligible_triples`, `retained_triples`, and `triple_threshold_statistics`, `triple_status`, and when order four is requested, `eligible_quads`, `retained_quads`, `quad_threshold_statistics` and `quad_status`, and `posterior`, the complete MCMC sample of total population size. Otherwise `"not requested"`.

References

Silverman, B. W. (2020). Multiple-systems analysis for the quantification of modern slavery: classical and Bayesian approaches. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 183, 691–736.

Fienberg, S. E. and Rinaldo, A. (2012). Maximum likelihood estimation in log-linear models. *The Annals of Statistics*, 40, 996–1023.

Forster, J. J. (2010). Bayesian inference for Poisson and multinomial log-linear models. *Statistical Methodology*, 7, 210–224.

Martin, A. D., Quinn, K. M. and Park, J. H. (2011). MCMCpack: Markov Chain Monte Carlo in R. *Journal of Statistical Software*, 42(9), 1–21.

Examples

```
data(Western, package = "MultipleSystemsEstimation")

fit <- estimate_population_bayesthresh(
    Western,
    burnin = 100,
    mcmc = 1000,
    seed = 1234
)
fit$estimate
fit$fitted_model
```

`estimate_population_bic`

Population estimation using BIC model selection

Description

Selects a model using the BIC criterion, accounting for sparse data and checking parameter identifiability and existence of the extended MLE. Hierarchical log-linear models are fitted to the observed data and ordered by increasing BIC. The point estimate is based on the model with the lowest BIC value.

If requested, constructs bootstrap BCa confidence intervals for population size that allow for uncertainty arising from BIC-based model selection. To reduce the computational load to feasible levels, an approximation is used in which the bootstrap and jackknife calculations are restricted to a specified number of the best-ranked models.

Usage

```
estimate_population_bic(
  zdat,
  nboot = 0,
  iseed = 1234,
```

```
  alpha = c(0.025, 0.1, 0.9, 0.975),
  maxorder = NULL,
  ntopmodels = NULL,
  return_details = FALSE,
  maxmodel = 100000L
)
```

Arguments

`zdat`	A capture history data matrix with $t+1$ columns, where t is the number of lists. The first t columns correspond to the capture lists and contain zeros and ones defining the observed capture histories. The final column contains the number of cases having each capture history. Capture histories not explicitly included in the data are assumed to have zero count.
`nboot`	Non-negative integer giving the number of bootstrap replications. If `nboot = 0`, the best-BIC population estimate and model are returned together with the complete original-data BIC enumeration, but no bootstrap or jackknife calculations are performed.
`iseed`	Integer seed used to initialise the random-number generator. The default is 1234.
`alpha`	Numeric vector of cumulative probability levels at which the endpoints of the BCa confidence intervals are to be evaluated. The default is `c(0.025, 0.1, 0.9, 0.975)`.
`maxorder`	Maximum order of interaction to include in the hierarchical models considered. If `NULL`, an automatic choice is made according to the number of lists: 1 for two lists, 2 for three lists, 3 for four or five lists, and 2 for six or more lists.
`ntopmodels`	Number of models, ranked by BIC on the original data, to retain for bootstrap inference. If `NULL`, an automatic choice is made when `nboot > 0`: all available models for two or three lists, 20 models for four lists, and 100 models for five or more lists. This argument is irrelevant when `nboot = 0`, because no model subsetting is then performed. Masked enlargements are excluded from the initial enumeration before any model is fitted. If `ntopmodels` is greater than the number of remaining models, then all remaining models are considered.
`return_details`	Logical. If `TRUE`, include the bootstrap estimates, BCa acceleration, selected-model BIC, complete original-data BIC enumeration, and effects represented at minus infinity. The default is `FALSE`.
`maxmodel`	Maximum number of candidate models to generate. If this limit would be exceeded, the maximum interaction order is reduced until the complete set of eligible models is within the limit.

Details

If `nboot > 0`, the routine implements the bootstrap procedure described by Silverman, Chan and Vincent (2024). Multinomial bootstrap samples are generated and the BIC model-selection procedure is repeated for each bootstrap sample, restricting consideration to a set of models having the smallest BIC values for the original data.

With the default settings, all available models are retained for bootstrap inference for two- and three-list data. For four-list data, the 20 models with the smallest BIC values are retained, and for

five- and six-list data the 100 models with the smallest BIC values are retained. Focusing attention on a subset of all possible models enables considerable computational economies.

Models are generated directly. When their number exceeds `maxmodel`, the maximum interaction order is reduced to a lower value as necessary. This may even reduce the search to the main-effects-only model. Requested and actual orders are recorded in `details`.

A masked interaction is a higher-order effect whose contribution cannot change any extended fitted mean because a lower-order effect is already estimated at minus infinity. For example, if lists A and B have no overlap then AB, if included in the model, is estimated at minus infinity. The interaction ABC is masked. Including ABC in the hierarchical model will not change the fitted mean of any history but will increase the number of nominal parameters, and so the BIC criterion cannot select this model.

A small value of `nboot`, such as that used in the example, is useful only for checking that the routine runs. A substantially larger number of bootstrap replications should be used for substantive inference.

Value

A list with components:

`input` A list containing the original `call` and `data`.

`method` The character string `"bic"`.

`estimate` A named numeric vector containing the estimated `dark_figure` and `total` population.

`fitted_model` The hierarchy with the smallest BIC on the original data.

`uncertainty` A two-row matrix of BCa endpoints for the dark figure and total population when `nboot > 0`; otherwise an explanatory character string.

`details` If `return_details = TRUE`, a list containing `minus_infinity_effects`, `bootstrap_estimates`, `bca_acceleration`, `BIC`, `maxorder`, `requested_maxorder`, and `bic_results`. The last is the original-data enumeration after masked enlargements are excluded, with one row per eligible model giving its total-population estimate, BIC and maximum interaction order, ordered by increasing BIC. Otherwise `"not requested"`.

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 doi:10.1007/s11222023103469.

Examples

```
data(Korea)

# A very small number of bootstrap replications is used here only
# to keep the example quick.
estimate_population_bic(Korea, nboot = 10)
```

estimate_population_fixed

Population estimation using a fixed hierarchical model

Description

Estimates the total population, including the unobserved population, using a specified hierarchical log-linear model.

Usage

```
estimate_population_fixed(
  zdat,
  model = NULL,
  nboot = 0,
  iseed = 1234,
  alpha = c(0.025, 0.1, 0.9, 0.975),
  return_details = FALSE
)
```

Arguments

`zdat`	A capture history data matrix with $t+1$ columns, where t is the number of lists. The first t columns correspond to the capture lists and contain zeros and ones defining the observed capture histories. The final column contains the number of cases having each capture history. Capture histories not explicitly included in the data are assumed to have zero count.
`model`	A character string specifying a hierarchical log-linear model by its maximal interactions. For example, `"[12,23]"` specifies the hierarchical model generated by interactions 12 and 23, including all implied lower-order terms. If `NULL`, the main-effects model is used. The default is `NULL`.
`nboot`	Non-negative integer giving the number of bootstrap replications. If `nboot = 0`, only the point estimate and fitted model are returned and no bootstrap or jackknife calculations are performed. The default is 0.
`iseed`	Integer seed used to initialise the random-number generator when `nboot > 0`. The default is 1234.
`alpha`	Numeric vector of cumulative probability levels at which the endpoints of the BCa confidence intervals are to be calculated. This argument is used only when `nboot > 0`. The default is `c(0.025, 0.1, 0.9, 0.975)`.
`return_details`	Logical. If `TRUE`, include the original-data GLM fit, bootstrap estimates, BCa acceleration, and effects estimated at minus infinity. The default is `FALSE`.

Details

Optional bootstrap and jackknife calculations provide BCa confidence intervals while fitting the same fixed model to every resampled data set.

The specified hierarchical model is fitted to the observed data. Parameter identifiability and existence of the extended MLE are checked for the original data and for every bootstrap and jackknife data set. Minimal zero-sufficient-statistic effects are represented at minus infinity by the fitting routine.

A masked interaction is a higher-order effect whose contribution cannot change any extended fitted mean because a lower-order effect is already estimated at minus infinity. For example, if lists A and B have no overlap then AB, if included in the model, is estimated at minus infinity. The interaction ABC is masked. Including ABC in the hierarchical model will not change the fitted mean of any history but will increase the number of nominal parameters, and so the BIC criterion cannot select this model.

If `nboot > 0`, multinomial bootstrap samples are generated from the observed capture history counts. The same fixed hierarchical model is fitted to every bootstrap sample. Unlike `estimate_population_stepwise`, there is no model-selection step within the bootstrap replications.

Bootstrap replications for which the fixed model cannot be fitted are omitted; their number is available as `details$bootstrap_failed` when `return_details = TRUE`. If the model cannot be fitted to any bootstrap sample, the routine stops with an error.

A delete-one jackknife calculation is also carried out to estimate the acceleration parameter required for the BCa confidence intervals. If the fixed model cannot be fitted after a positive-count deletion, the routine stops with an informative error.

A small positive value of `nboot`, such as that used in the example, is useful only for checking that the routine runs. A substantially larger number of bootstrap replications should be used for substantive inference.

Value

A list with components:

`input` A list containing the original `call` and `data`.

`method` The character string `"fixed"`.

`estimate` A named numeric vector containing the estimated `dark_figure` and `total` population.

`fitted_model` The hierarchy fitted to the original data.

`masked_effects` Effects masked in the original data, whose reported coefficients are zero by convention. Always returned, including when `return_details = FALSE`.

`uncertainty` A two-row matrix of BCa endpoints for the dark figure and total population when `nboot > 0`; otherwise an explanatory character string.

`details` If `return_details = TRUE`, a list containing `minus_infinity_effects`, `masked_effects`, `bootstrap_estimates`, `bootstrap_failed`, `bca_acceleration`, and `glm_fit`. The last is the complete conditional GLM fit for the fixed model. Masked effects have conventional coefficient zero and are listed separately in `masked_effects`. Otherwise `"not requested"`.

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 the American Statistical 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*, **11**(3), 189–228.

Examples

```
data(Korea)

# Main-effects model without bootstrapping
estimate_population_fixed(Korea)

# A fixed pairwise-interaction model using hierarchy notation
estimate_population_fixed(Korea, model = "[23,1]")

# A very small number of bootstrap replications is used here only
# to keep the example quick.
estimate_population_fixed(
  Korea,
  model = "[23,1]",
  nboot = 2
)
```

---

estimate_population_stepwise

Population estimation using stepwise model selection

Description

Estimates the total population, including the unobserved population, using an extension of the stepwise model-selection procedure of Chan, Silverman and Vincent (2021). Optional bootstrap and jackknife calculations provide BCa confidence intervals while repeating the stepwise selection procedure for each resampled data set.

Usage

```
estimate_population_stepwise(
  zdat,
  nboot = 0,
  pthresh = 0.02,
  maxorder = Inf,
  iseed = 1234,
  alpha = c(0.025, 0.1, 0.9, 0.975),
  return_details = FALSE
)
```

Arguments

`zdat`	A capture history data matrix with $t+1$ columns. The first t columns correspond to the capture lists and contain zeros and ones defining the observed capture histories. The final column contains the number of cases having each capture history. List names `A`, `B`, and so on are constructed if they are not supplied. Capture histories not explicitly included in the data are assumed to have zero count.
`nboot`	Non-negative integer giving the number of bootstrap replications. If `nboot = 0`, only the point estimate and fitted model are returned and no bootstrap or jackknife calculations are performed. The default is 0.
`pthresh`	P-value threshold used by the stepwise model-selection procedure. The default is 0.02.
`maxorder`	Maximum order of interactions considered for selection. An integer of at least 2, or `Inf` to allow interactions of any order. The default is `Inf`.
`iseed`	Integer seed used to initialise the random-number generator when `nboot > 0`. The default is 1234.
`alpha`	Numeric vector of cumulative probability levels at which the endpoints of the BCa confidence intervals are to be calculated. This argument is used only when `nboot > 0`. The default is `c(0.025, 0.1, 0.9, 0.975)`.
`return_details`	Logical. If `TRUE`, include the bootstrap estimates, BCa acceleration, and effects represented at minus infinity. The default is `FALSE`.

Details

By default, interactions of any order may be considered. Setting `maxorder = 2` restricts the procedure to the pairwise method of Chan, Silverman and Vincent (2021). An interaction is considered only when all its lower-order terms are already present, so every candidate model is hierarchical. A masked sparse interaction is not eligible for selection, including when `pthresh = 1`.

The procedure starts with a main-effects model. At each stage it considers eligible interactions not already included and adds the interaction having the smallest p-value if that p-value does not exceed `pthresh`. It stops when no eligible interaction meets the threshold. Candidate models are checked for parameter identifiability and existence of the extended MLE.

The procedure is first applied to the observed data to obtain the point estimate and fitted model.

If `nboot > 0`, multinomial bootstrap samples are generated from the observed capture history counts. The complete stepwise model-selection procedure is repeated for each bootstrap sample, so the resulting inference allows for variation in the selected model rather than treating the model selected from the original data as fixed.

A delete-one jackknife calculation is also carried out to estimate the acceleration parameter required for the BCa confidence intervals. The jackknife calculation takes account of the number of individuals having each observed capture history.

A small positive value of `nboot`, such as that used in the example, is useful only for checking that the routine runs. A substantially larger number of bootstrap replications should be used for substantive inference.

Value

A list with components:

`input` A list containing the original `call` and `data`.

`method` The character string `"stepwise"`.

`estimate` A named numeric vector containing the estimated `dark_figure` and `total` population.

`fitted_model` The hierarchy selected from the original data.

`uncertainty` A two-row matrix of BCa endpoints for the dark figure and total population when `nboot > 0`; otherwise an explanatory character string.

`details` If `return_details = TRUE`, a list containing `minus_infinity_effects`, `bootstrap_estimates`, and `bca_acceleration`. Otherwise `"not requested"`.

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 the American Statistical 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*, **11**(3), 189–228.

Examples

```
data(Korea)

# Point estimate and fitted model without bootstrapping
estimate_population_stepwise(Korea)

# Allow interactions of any order
estimate_population_stepwise(Kosovo, maxorder = Inf)

# A very small number of bootstrap replications is used here only
# to keep the example quick.
estimate_population_stepwise(Korea, nboot = 10)
```

`find_neighbour_hierarchies`

Find the neighbours of a hierarchical model

Description

Given a hierarchical model, finds its outer neighbours, its inner neighbours, or both.

Usage

```
find_neighbour_hierarchies(
  modelstr,
  nlists = NA,
  keepmaineffects = TRUE,
  maxorder = nlists - 1,
  type = c("all", "outer", "inner")
)
```

Arguments

`modelstr`	A model string written in hierarchical form.
`nlists`	The total number of lists. If `NA`, it is inferred from the largest list number appearing in `modelstr`.
`keepmaineffects`	If `TRUE`, main effects are not removed when constructing inner neighbours.
`maxorder`	The maximum interaction order permitted in outer neighbours.
`type`	Which neighbours to return. `"all"` returns both inner and outer neighbours, `"outer"` returns only models obtained by adding a term, and `"inner"` returns only models obtained by removing a generator.

Details

An outer neighbour is obtained by adding an interaction term all of whose subsets are already in the model.

An inner neighbour is obtained by removing one of the generators defining the hierarchical model. Removing a generator removes only the defining interaction term itself, not the lower-order terms that it implies. For example, removing the generator `123` from `[123,34]` yields `[12,13,23,34]`, not `[34]`. Main effects are not removed when `keepmaineffects = TRUE`.

Value

A character vector containing the requested neighbouring 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. doi:10.1007/s11222023103469.

Examples

```
modelstr <- "[12,23]"

find_neighbour_hierarchies(modelstr)
find_neighbour_hierarchies(modelstr, type = "outer")
find_neighbour_hierarchies(modelstr, type = "inner")
```

get_hierarchical_models

Generate hierarchical models

Description

Generates all hierarchical models satisfying specified criteria. The user can specify the number of lists, the maximum order of models to be considered, and (optionally) a data set for which any masked sparse interactions will not be included in the models. Intercept and main effects are always included.

Usage

```
get_hierarchical_models(
  nlists,
  maxorder = nlists - 1L,
  xdata = NULL,
  maxmodel = 100000L
)
```

Arguments

`nlists`	Number of lists (an integer from 2 to 9).
`maxorder`	Maximum interaction order; Inf permits all orders.
`xdata`	Optional capture history data. If supplied, models including masked sparse interactions for this data set will not be included. Minimal sparse interactions remain eligible.
`maxmodel`	Maximum number of models to be generated. If exceeded, the maximum order will be reduced, if necessary as far as 1, to bring the number of models below the specified limit.

Value

A named list of integer vectors specifying the hierarchical models. The name of each vector in the list is the hierarchy string representation of the model, and the vector itself is the vector of encoded capture histories included in the model. A capture history is encoded as the integer $1 + \sum_{i \in S} 2^{i-1}$, where S is the set of capture lists in the history. The returned list has two attributes:

`requested_maxorder` The value of `maxorder` supplied by the user, or its default value if omitted.

`maxorder` The actual maximum interaction order of the generated models. This is capped at `nlists` and may be reduced further to satisfy `maxmodel`.

Examples

```
models <- get_hierarchical_models(3)
names(models)
models[[1]]
```

Korea — *Korean military-sexual-slavery data*

Description

Three-list capture-history data concerning Korean women held in sexual slavery by the Japanese military in Palembang, Indonesia. Further details are given in the data section and Figure 1 of Ball, Shin and Yang (2018).

Usage

```
Korea
```

Format

A numeric matrix with 7 rows and 4 columns. Columns `b`, `c` and `d` are binary list-membership indicators. Column `Count` gives the number of cases having each capture history.

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, Human Rights Data Analysis Group.

Kosovo — *Kosovo conflict data*

Description

Four-list capture-history data for 4,400 documented killings during the Kosovo conflict between 20 March and 22 June 1999. The lists are exhumations (`EXH`), the American Bar Association Central and East European Law Initiative (`ABA`), the Organization for Security and Co-operation in Europe (`OSCE`), and Human Rights Watch (`HRW`). All 15 observable capture histories have positive count.

Usage

```
Kosovo
```

Format

A data frame with 15 rows and 5 columns. The first four columns are binary list-membership indicators. Column `Frequency` gives the number of cases having each capture history.

References

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

Ned — *Netherlands human-trafficking data*

Description

Six-list capture-history data for identified victims of human trafficking in the Netherlands. The data are described in Table 2 of Silverman (2020).

Usage

```
Ned
```

Format

A data frame with 24 rows and 7 columns. Columns `I`, `K`, `O`, `P`, `R` and `Z` are binary list-membership indicators. Column `frequency` gives the number of cases having each observed capture history.

References

Silverman, B. W. (2020). Multiple-systems analysis for the quantification of modern slavery: classical and Bayesian approaches. *Journal of the Royal Statistical Society: Series A*, **183**(3), 691–736. doi:10.1111/rssa.12505.

Ned_5 — *Five-list version of the Netherlands data*

Description

A five-list version of `Ned`, constructed by combining lists I and O into a single list, `IO`.

Usage

```
Ned_5
```

Format

A data frame with 17 rows and 6 columns. Columns `IO`, `K`, `P`, `R` and `Z` are binary list-membership indicators. Column `frequency` gives the number of cases having each observed capture history.

NewOrl — *Greater New Orleans human-trafficking data*

Description

Capture-history data for identified victims of human trafficking in Greater New Orleans. The data comprise eight confidentially labelled lists, denoted A to H. Further details are given by Bales, Murphy and Silverman (2020).

Usage

```
NewOrl
```

Format

A data frame with 19 rows and 9 columns. Columns `A` to `H` are binary indicators of inclusion in the eight lists. Column `n` gives the number of cases having each observed capture history.

References

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

NewOrl_5 — *Five-list version of the Greater New Orleans data*

Description

A five-list version of `NewOrl`, constructed by combining the four smallest lists B, E, F and G into a single list, `BEFG`.

Usage

```
NewOrl_5
```

Format

A data frame with 14 rows and 6 columns. Columns `A`, `BEFG`, `C`, `D` and `H` are binary list-membership indicators. Column `n` gives the number of cases having each observed capture history.

tidy_lists *Consolidate capture history data*

Description

Combines duplicate capture histories by summing their counts. It can also add all observable histories having zero count and remove non-informative capture lists.

Usage

```
tidy_lists(zdat, includezerocounts = FALSE, remove_noninformative = FALSE)
```

Arguments

`zdat` A matrix or data frame with $t + 1$ columns. The first t columns are binary list-membership indicators and the final column contains the capture history counts. Histories not included explicitly are assumed to have zero count.

`includezerocounts` If `TRUE`, include every observable capture history, including those with zero count. The all-zero history representing the unobserved population is never included. If `FALSE`, return only positive-count histories.

`remove_noninformative` If `TRUE`, remove lists containing all or none of the observed cases and remove duplicate list columns, retaining the first copy.

Value

A data frame with one row for each retained capture history and one column for each retained list, followed by the count column.

Examples

```
# Two rows have the same capture history: A only.
zdat <- data.frame(
  A = c(1, 1, 0, 1),
  B = c(0, 0, 1, 1),
  C = c(0, 0, 0, 1),
  count = c(3, 5, 4, 2)
)
zdat

# Consolidate rows with the same capture history by summing their counts.
# The two A-only rows become one row with count 3 + 5 = 8.
tidy_lists(zdat)

# Include all seven observable histories, filling missing counts with zero.
# The unobserved history (0, 0, 0) is excluded.
tidy_lists(zdat, includezerocounts = TRUE)
```

```
# Add a duplicate list, an empty list and a list containing every case.
# Keep the count column last.
redundant <- data.frame(
  zdat[c("A", "B", "C")],
  A_copy = zdat$A,
  empty = 0,
  everyone = 1,
  count = zdat$count
)
redundant

# Remove the three redundant lists and consolidate repeated histories.
tidy_lists(redundant, remove_noninformative = TRUE)
```

UKdat *United Kingdom modern-slavery data*

Description

Six-list capture-history data from the United Kingdom 2013 strategic assessment of modern slavery. The lists are local authorities (LA), non-governmental organisations (NG), police forces (PF), government organisations (GO), the general public (GP), and the National Crime Agency (NCA).

Usage

```
UKdat
```

Format

A data frame with 25 rows and 7 columns. The first six columns are binary list-membership indicators. Column count gives the number of cases having each observed capture history. Capture histories having zero count are omitted.

References

Home Office (2014). Modern Slavery: an application of multiple systems estimation. https://www.gov.uk/government/publications/modern-slavery-an-application-of-multiple-systems-estimation.

UKdat_5 *Five-list version of the United Kingdom data*

Description

A five-list version of UKdat, constructed by combining the police-force and National Crime Agency lists into a single list, PFNCA.

Usage

```
UKdat_5
```

Format

A data frame with 18 rows and 6 columns. Columns LA, NG, PFNCA, GO and GP are binary list-membership indicators. Column count gives the number of cases having each observed capture history.

Western *Western-site sex-trafficking data*

Description

Capture-history data for identified victims of sex trafficking at a western United States site. The five source lists are confidentially labelled A to E. Further details are given by Farrell et al. (2018).

Usage

```
Western
```

Format

A data frame with 13 rows and 6 columns. Columns A to E are binary list-membership indicators. Column n gives the number of cases having each observed capture history.

References

Farrell, A., Dank, M., Kafafian, M., Lockwood, S., Pfeffer, R., Hughes, A. and Vincent, K. (2018). Capturing human trafficking victimization through crime reporting. Technical Report 2015-VF-GX-0105, National Institute of Justice. https://www.ojp.gov/pdffiles1/nij/grants/252520.pdf.

Index

∗ **datasets**
    Artificial_3, 2
    Korea, 19
    Kosovo, 19
    Ned, 20
    Ned_5, 20
    NewOrl, 21
    NewOrl_5, 21
    UKdat, 23
    UKdat_5, 24
    Western, 24

Artificial_3, 2

check_extended_MLE, *2*, 3

estimate_population, 4
estimate_population_bayesthresh, *4, 5*, 6
estimate_population_bic, *4, 5*, 9
estimate_population_fixed, *4, 5*, 12
estimate_population_stepwise, *4, 5*, *13*, 14

find_neighbour_hierarchies, 16

get_hierarchical_models, 18

Korea, 19
Kosovo, 19

Ned, 20, *20*
Ned_5, 20
NewOrl, 21, *21*
NewOrl_5, 21

tidy_lists, 22

UKdat, 23, *24*
UKdat_5, 24

Western, 24