Package 'stratallo'

April 13, 2022

Title Optimum Sample Allocation in Stratified Sampling Schemes

Version 2.0.1

Description Functions in this package provide solution to classical problem in survey methodology - an optimum sample allocation in stratified sampling schemes. In this context, the optimal allocation is in the classical Tschuprov-Neyman's sense and it satisfies additional either lower or upper bounds restrictions imposed on sample sizes in strata. There are few different algorithms available to use, and one them is based on popular sample allocation method that applies Neyman allocation to recursively reduced set of strata.
This package also provides the function that computes a solution to the minimum sample size allocation problem, which is a minor modification of the classical optimium sample allocation. This problems lies in the determination of a vector of strata sample sizes that minimizes total sample size, under assumed fixed level of the pi-estimator's variance. As in the case of the classical optimal allocation, the problem of minimum sample size allocation can be complemented by imposing upper bounds constraints on sample sizes in strata.

License GPL-2

URL https://github.com/wwojciech/stratallo

BugReports https://github.com/wwojciech/stratallo/issues

Copyright Wojciech Wójciak

Language en-US

Encoding UTF-8

LazyData true

Imports checkmate, lifecycle

RoxygenNote 7.1.2

Suggests rmarkdown, knitr, spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

Collate 'helpers.R' 'algorithms_rna.R' 'algorithms_upper.R' 'opt.R' 'pop.R' 'stratallo-package.R' 'variance.R'

VignetteBuilder knitr

NeedsCompilation no

Author Wojciech Wójciak [aut, cre],
Jacek Wesołowski [sad],
Robert Wieczorkowski [ctb]

Maintainer Wojciech Wójciak <wojciech.wojciak@gmail.com>

Repository CRAN

Date/Publication 2022-04-13 14:40:02 UTC

R topics documented:

`stratallo-package` *Functions for Optimal Sample Allocation in Stratified Sampling Schemes*

Description

Optimal Sample Allocation in Stratified Sampling Schemes

Author(s)

Wojciech Wójciak <wojciech.wojciak@gmail.com>

References

Wesołowski, J., Wieczorkowski, R., Wójciak, W. (2021), Optimality of the recursive Neyman allocation, *Journal of Survey Statistics and Methodology*, doi: 10.1093/jssam/smab018, doi: 10.48550/arXiv.2105.14486

Wójciak, W. (2022), Minimum sample size allocation in stratified sampling under constraints on variance and strata sample sizes, doi: 10.48550/arXiv.2204.04035

Wójciak, W. (2019), Optimal allocation in stratified sampling schemes, *MSc Thesis*, Warsaw University of Technology, Warsaw, Poland. http://home.elka.pw.edu.pl/~wwojciak/msc_optimal_allocation.pdf

Sarndal, C.-E., Swensson, B., and Wretman, J. (1992), *Model Assisted Survey Sampling*, New York, NY: Springer.

`dopt` *Optimum Sample Allocation in Stratified Sampling Schemes*

Description

[Stable]

A classical problem in survey methodology in stratified sampling is an optimum sample allocation problem. This problem is formulated as determination of a vector of strata sample sizes that minimizes the variance of the pi-estimator of the population total of a given study variable, under constraint on total sample size.

The `dopt()` function solves the problem of optimum sample allocation under either lower or upper bounds constraints, optionally imposed on strata sample sizes. The allocation computed is valid for all stratified sampling schemes for which the variance of the stratified pi-estimator is of the form

$$D(x_1, ..., x_H) = a_1^2/x_1 + ... + a_H^2/x_H - b,$$

where H denotes total number of strata, $x_1, ..., x_H$ are the strata sample sizes, and b, $a_w > 0$ do not depend on $x_w, w = 1, ..., H$.

The `dopt()` function makes use of the following allocation algorithms: `rNa`, `sga`, `sgaplus`, `coma` for optimal sample allocation under upper bounds constraints only, and `LrNa` for optimal sample allocation under lower bounds constraints only. The `rNa`, `sga`, and `coma` are described in Wesołowski et al. (2021), the `sgaplus` in Wójciak (2019), and the `LrNa` is introduced in Wójciak (2022). If no inequality constraints are added, then no any special algorithm is used as the allocation is given as a closed form expression, known as Neyman allocation

$$x_w = a_w * n/(a_1 + ... + a_H), w = 1, ..., H.$$

Usage

```
dopt(n, a, m = NULL, M = NULL, M_method = "rna")
```

Arguments

n	(number) total sample size. A strictly positive scalar.
a	(numeric) parameters $a_1, ..., a_H$ of variance function D. Strictly positive numbers.
m	(numeric or NULL) lower bounds constraints optionally imposed on strata sample sizes. If different than NULL, it is then required that n >= sum(m). Must be assigned with NULL (default) if M is not NULL. Strictly positive numbers.
M	(numeric or NULL) upper bounds constraints optionally imposed on strata sample sizes. If different than NULL, it is then required that n <= sum(M). Must be assigned with NULL (default) if m is not NULL. Strictly positive numbers.
M_method	(string) the name of the underlying algorithm to be used for computing a sample allocation under one-sided upper bounds constraints (Case I). One of the following: *"rna"* (default), *"sga"*, *"sgaplus"*, *"coma"*.

Details

The dopt() function computes

$$argminD(x_1, ..., x_H),$$

under the equality constraint imposed on total sample size

$$x_1 + ... + x_H = n.$$

Optionally, one of the following set of one-sided inequality constraints can be added

$$x_w <= M_w, w = 1, ..., H, (CaseI)$$

or

$$x_w >= m_w, w = 1, ..., H, (CaseII)$$

where $n > 0$ denotes overall sample size, $m_w > 0$, and $M_w > 0, w = 1, ..., H$, are the lower and upper bounds respectively, imposed on sample strata sizes.
User of dopt() can choose whether the inequality constraints will be added to the optimization problem or not. This is achieved with the proper use of m and M arguments of the function. In case of no inequality constraints to be added, m and M must be both specified as NULL (default). If only upper bounds constraints should be added (Case I), user specifies these bounds with M argument, while leaving m as NULL. Finally, if only lower bounds constraints should be added (Case II), user specifies these bounds with m argument, while leaving M as NULL. At the moment, dopt() function does not allow to add box-constraints simultaneously.

For the Case I, there are four different underlying algorithms available to use. These are abbreviated as: "rNa" (rna_one_sided()), "sga" (sga()), "sgaplus" (sgaplus()), "coma" (coma()). Functions names that perform given algorithms are given in the brackets. See its help page for more details. For the Case II, the "rNa" (rna_one_sided()) is used.

Value

Numeric vector with optimal sample allocation in strata.

Note

For simple random sampling without replacement design in each stratum, parameters of the variance function D are $b = N_1 * S_1^2 + ... + N_H * S_H^2$, and $a_w = N_w * S_w$, where $N_w, S_w, w = 1, ..., H$, are strata sizes and standard deviations of a study variable in strata respectively.

References

Wesołowski, J., Wieczorkowski, R., Wójciak, W. (2021), Optimality of the recursive Neyman allocation, *Journal of Survey Statistics and Methodology*, doi: 10.1093/jssam/smab018, doi: 10.48550/arXiv.2105.14486

Wójciak, W. (2022), Minimum sample size allocation in stratified sampling under constraints on variance and strata sample sizes, doi: 10.48550/arXiv.2204.04035

Wójciak, W. (2019), Optimal allocation in stratified sampling schemes, *MSc Thesis*, Warsaw University of Technology, Warsaw, Poland. http://home.elka.pw.edu.pl/~wwojciak/msc_optimal_allocation.pdf

Sarndal, C.-E., Swensson, B., and Wretman, J. (1992), *Model Assisted Survey Sampling*, New York, NY: Springer.

See Also

nopt(), rna_one_sided(), sga(), sgaplus(), coma().

Examples

```
a <- c(3000, 4000, 5000, 2000)
bounds <- c(100, 90, 70, 80)

# Only lower bounds.
dopt(n = 340, a = a, m = bounds)
dopt(n = 400, a = a, m = bounds)
dopt(n = 600, a = a, m = bounds)

# Only upper bounds.
dopt(n = 190, a = a, M = bounds)
dopt(n = 300, a = a, M = bounds)
dopt(n = 340, a = a, M = bounds)

# Example of execution-time comparison of different algorithms
# using bench R package.
## Not run:
N <- pop969[, "N"]
S <- pop969[, "S"]
```

```
a <- N * S
nfrac <- seq(0.01, 0.9, 0.05)
n <- setNames(as.integer(nfrac * sum(N)), nfrac)
lapply(
  n,
  function(ni) {
    bench::mark(
      dopt(ni, a, M = N, M_method = "rna"),
      dopt(ni, a, M = N, M_method = "sga"),
      dopt(ni, a, M = N, M_method = "sgaplus"),
      dopt(ni, a, M = N, M_method = "coma"),
      iterations = 200
    )[c(1, 3)]
  }
)

## End(Not run)
```

`dopt_upper` *Algorithms for Optimum Sample Allocation in Stratified Sampling Under Upper Bounds Constraints*

Description

[Stable]

Internal functions that implement the optimal sample allocation algorithms: `sga`, `sgaplus` and `coma`. Functions from this family compute

$$argminD(x_1, ..., x_H) = a_1^2/x_1 + ... + a_H^2/x_H - b,$$

under the equality constraint imposed on total sample size

$$x_1 + ... + x_H = n,$$

and upper bounds constraints imposed on strata sample sizes

$$x_w <= M_w, w = 1, ..., H.$$

Here, H denotes total number of strata, $x_1, ..., x_H$ are the strata sample sizes, and $n > 0$, b, $a_w > 0$, $M_w > 0, w = 1, ..., H$ are given numbers.
The `sga()`, `sgaplus()` and `coma()` are internal implementations of the algorithms in subject, and hence, users should not use any of these functions directly. Instead, the `dopt()` should be used.

Usage

```
sga(n, a, M)

sgaplus(n, a, M)

coma(n, a, M)
```

Arguments

`n`	(`number`) total sample size. A strictly positive scalar.
`a`	(`numeric`) parameters $a_1, ..., a_H$ of variance function D. Strictly positive numbers.
`M`	(`numeric`) upper bounds constraints imposed on strata sample sizes. It is required that `n <= sum(M)`. Strictly positive numbers.

Value

Numeric vector with optimal sample allocations in strata.

Functions

- `sga`: implementation of the Stenger-Gabler type algorithm `SGa`, described in Wesołowski et al. (2021) and in Stenger and Gabler (2005).
- `sgaplus`: implementation of the modified Stenger-Gabler type algorithm, described in Wójciak (2019) as Sequential Allocation (version 1) algorithm.
- `coma`: implementation of the Change of Monotonicity Algorithm, or `coma`, described in Wesołowski et al. (2021).

Note

For simple random sampling without replacement design in each stratum, parameters of the variance function D are $b = N_1 * S_1^2 + ... + N_H * S_H^2$, and $a_w = N_w * S_w$, where $N_w, S_w, w = 1, ..., H$, are strata sizes and standard deviations of a study variable in strata respectively.

References

Wesołowski, J., Wieczorkowski, R., Wójciak, W. (2021), Optimality of the recursive Neyman allocation, *Journal of Survey Statistics and Methodology*, doi: 10.1093/jssam/smab018, doi: 10.48550/arXiv.2105.14486

Stenger, H., Gabler, S. (2005), Combining random sampling and census strategies - Justification of inclusion probabilities equal 1, *Metrika*, 61, 137-156

Wójciak, W. (2019), Optimal allocation in stratified sampling schemes, *MSc Thesis*, Warsaw University of Technology, Warsaw, Poland. http://home.elka.pw.edu.pl/~wwojciak/msc_optimal_allocation.pdf

See Also

`dopt()`, `rna_one_sided()`.

Examples

```
a <- c(3000, 4000, 5000, 2000)
M <- c(100, 90, 70, 80)
sga(n = 190, a = a, M = M)
sgaplus(n = 190, a = a, M = M)
coma(n = 190, a = a, M = M)
```

`h_get_which_violated` *Get Proper Version of* `which_violated` *Function.*

Description

[Stable]

Internal function that prepares a simple 2-arguments wrapper of `base::which()` that checks whether its first argument exceeds the second one. Both arguments are numeric. This excess is hard coded in the returned wrapper function and it is defined either as $>=$ ("greater or equal") or $<=$ ("lower or equal"), depending on the value of the `geq` flag.

Usage

```
h_get_which_violated(geq = TRUE)
```

Arguments

`geq`	(`flag`) if `TRUE`, then "greater or equal" condition is set. Otherwise, "less then or equal" is set.

Value

2-arguments function that checks whether its first argument exceeds the second one. Both arguments must be numeric.

See Also

`rna_one_sided()`.

Examples

```
which_violated <- stratallo:::h_get_which_violated()
which_violated(1:3, 3:1)

which_violated <- stratallo:::h_get_which_violated(geq = FALSE)
which_violated(1:3, 3:1)
```

nopt *Minimum Sample Size Allocation in Stratified Sampling Schemes*

Description

[Stable]

User function that determines fixed strata sample sizes that minimize total sample size, under assumed level of the variance of the stratified pi-estimator of the total and optional one-sided upper bounds imposed on strata sample sizes. The algorithm used by `nopt()` is described in Wójciak (2022). The allocation computed is valid for all stratified sampling schemes for which the variance of the stratified pi-estimator is of the form

$$D(x_1, ..., x_H) = a_1^2/x_1 + ... + a_H^2/x_H - b,$$

where H denotes total number of strata, $x_1, ..., x_H$ are the strata sample sizes, and b, $a_w > 0$ do not depend on $x_w, w = 1, ..., H$.

Usage

```
nopt(D, a, b, M = NULL)
```

Arguments

`D`	(`number`) variance equality constraint value. A strictly positive scalar.
`a`	(`numeric`) parameters $a_1, ..., a_H$ of variance function D. Strictly positive numbers.
`b`	(`number`) parameter b of variance function D.
`M`	(`numeric` or `NULL`) upper bounds constraints optionally imposed on strata sample sizes. If different than `NULL`, it is then required that D >= sum(a/M) - b > 0. Strictly positive numbers.

Details

The `nopt()` function computes

$$argminn(x_1, ..., x_H) = x_1 + ... + x_H,$$

under the equality constraint imposed on the variance

$$a_1^2/x_1 + ... + a_H^2/x_H - b = D.$$

Optionally, the following set of one-sided inequality constraints can be added

$$x_w <= M_w, w = 1, ..., H,$$

where $D > 0$ is a given number and $M_w > 0, w = 1, ..., H$, are the upper bounds, imposed on sample strata sizes.

Value

Numeric vector with optimal sample allocation in strata.

Note

For simple random sampling without replacement design in each stratum, parameters of the variance function D are $b = N_1 * S_1^2 + ... + N_H * S_H^2$, and $a_w = N_w * S_w$, where $N_w, S_w, w = 1, ..., H$, are strata sizes and standard deviations of a study variable in strata respectively.

References

Wójciak, W. (2022), Minimum sample size allocation in stratified sampling under constraints on variance and strata sample sizes, doi: 10.48550/arXiv.2204.04035

See Also

`rna_one_sided()`, `dopt()`.

Examples

```
a <- c(3000, 4000, 5000, 2000)
M <- c(100, 90, 70, 80)
nopt(1017579, a = a, b = 579, M = M)
```

---

pop10_mM *Example Population with 10 Strata and Lower and Upper Bounds*

Description

Artificial example population with 10 strata defined by strata sizes and standard deviations of some study variable. Additionally, the lower and upper bounds have been specified for the sample in strata.

Usage

```
pop10_mM
```

Format

A matrix with 10 rows and 5 columns: `N` (strata sizes), `S` (standard deviations of a variable under study in strata), `m` (lower bounds), `M` (upper bounds).

pop507 *Example Population with 507 Strata*

Description

Artificial example population with 507 strata defined by strata sizes and standard deviations of some study variable.

Usage

```
pop507
```

Format

A matrix with 507 rows and 2 columns: `N` with strata sizes and `S` standard deviations of a variable under study in strata.

pop969 *Example Population with 969 Strata*

Description

Artificial example population with 969 strata defined by strata sizes and standard deviations of some study variable.

Usage

```
pop969
```

Format

A matrix with 969 rows and 2 columns: `N` with strata sizes and `S` standard deviations of a variable under study in strata.

rna_one_sided *Recursive Neyman Algorithm for Optimal Sample Allocation in Stratified Sampling Schemes*

Description

[Stable]

Internal function that implements the recursive Neyman optimal allocation algorithms, `rNa` and `LrNa`, described in Wesołowski et al. (2021) and Wójciak (2022) respectively. The `rna_one_sided()` should not be used directly. Instead, user function `dopt()` or `nopt()` should be used.

The `rna_one_sided()` function computes

$$argminD(x_1, ..., x_H) = a_1^2/x_1 + ... + a_H^2/x_H - b,$$

under the equality constraint imposed on total sample size

$$x_1 + ... + x_H = n.$$

Here, H denotes total number of strata, $x_1, ..., x_H$ are the strata sample sizes, and $n > 0$, b, $a_w > 0, w = 1, ..., H$, are given numbers.
Optionally, one of the following set of one-sided inequality constraints can be added

$$x_w <= M_w, w = 1, ..., H, (CaseI)$$

or

$$x_w >= m_w, w = 1, ..., H, (CaseII)$$

where $m_w > 0$ and $M_w > 0, w = 1, ..., H$ are lower and upper bounds respectively, imposed on sample strata sizes.

User of `rna_one_sided()` can choose whether the inequality constraints will be added to the optimization problem or not. This is achieved with the proper use of `bounds` and `upper` arguments of the function. In case of no inequality constraints to be added, `bounds` must be specified as `NULL` (default). If any bounds should be imposed on sample strata sizes, user must specify these with `bounds` argument. For the Case I of the upper bounds, `upper` flag must be set to `TRUE` (default) and then the `rna_one_sided()` performs the `rNa`. For the Case II of lower bounds, `upper` flag must be set to `FALSE` and then the `rna_one_sided()` performs the `LrNa`. The `upper` flag is ignored when `bounds` is `NULL`. If no inequality constraints are added, the allocation is given as a closed form expression, known as Neyman allocation

$$x_w = a_w * n/(a_1 + ... + a_H), w = 1, ..., H.$$

Usage

```
rna_one_sided(n, a, bounds = NULL, upper = TRUE)
```

Arguments

n	(number) total sample size. A strictly positive scalar.
a	(numeric) parameters $a_1, ..., a_H$ of variance function D. Strictly positive numbers.
bounds	(numeric or NULL) optional lower or upper bounds constraints imposed on strata sample sizes. If bounds is not NULL, it is required that n <= sum(bounds) in case of upper = TRUE, and n >= sum(bounds), in case of upper = FALSE. Strictly positive numbers.
upper	(flag) should values of bounds be treated as one-sided upper bounds constraints (default)? Otherwise, they are treated as lower bounds.

Value

Numeric vector with optimal sample allocations in strata.

Note

For simple random sampling without replacement design in each stratum, parameters of the variance function D are $b = N_1 * S_1^2 + ... + N_H * S_H^2$, and $a_w = N_w * S_w$, where $N_w, S_w, w = 1, ..., H$, are strata sizes and standard deviations of a study variable in strata respectively.

The rNa and LrNa are kind of more general versions of popular recursive Neyman allocation procedure that is described in Remark 12.7.1 in Sarndal et al. (1992). It is a procedure of optimal sample allocation in stratified sampling scheme with simple random sampling without replacement design in each stratum while not exceeding strata sizes.

References

Wesołowski, J., Wieczorkowski, R., Wójciak, W. (2021), Optimality of the recursive Neyman allocation, *Journal of Survey Statistics and Methodology*, doi: 10.1093/jssam/smab018, doi: 10.48550/arXiv.2105.14486

Wójciak, W. (2022), Minimum sample size allocation in stratified sampling under constraints on variance and strata sample sizes, doi: 10.48550/arXiv.2204.04035

Sarndal, C.-E., Swensson, B., and Wretman, J. (1992), *Model Assisted Survey Sampling*, New York, NY: Springer.

See Also

dopt(), nopt(), sga(), sgaplus(), coma().

Examples

```
a <- c(3000, 4000, 5000, 2000)
bounds <- c(100, 90, 70, 80)
rna_one_sided(n = 190, a = a, bounds = bounds)
```

`var_tst` *Variance of Stratified Pi-estimator of the Total*

Description

[Stable]

Compute the variance of the stratified pi-estimator of the population total, that is of the following generic form

$$D(x_1, ..., x_H) = a_1^2/x_1 + ... + a_H^2/x_H - b,$$

where H denotes total number of strata, $x_1, ..., x_H$ are the strata sample sizes, and b, $a_w > 0$ do not depend on $x_w, w = 1, ..., H$.

Usage

```
var_tst(x, a, b)

var_tst_si(x, N, S)
```

Arguments

`x`	(`numeric`) sample allocations in strata. Strictly positive numbers.
`a`	(`numeric`) parameters $a_1, ..., a_H$ of variance function D. Strictly positive numbers.
`b`	(`numeric`) parameter b of variance function D.
`N`	(`numeric`) strata sizes. Strictly positive numbers.
`S`	(`numeric`) strata standard deviations of a given study variable. Strictly positive numbers.

Value

Value of the variance D for a given allocation vector `x`.

Functions

- `var_tst_si`: computes variance of stratified pi-estimator of the total for simple random sampling without replacement design in each stratum. Under this design, parameters of the variance function D take the following form

$$a_w = N_w * S_w, w = 1, ..., H,$$

and

$$b = N_1 * S_1^2 + ... + N_H * S_H^2,$$

where $N_w, S_w, w = 1, ..., H$, are strata sizes and standard deviations of a study variable in strata respectively.

References

Sarndal, C.-E., Swensson, B., and Wretman, J. (1992), *Model Assisted Survey Sampling*, Chapter 3.7 *Stratified Sampling*, New York, NY: Springer.

Examples

```
N <- c(3000, 4000, 5000, 2000)
S <- rep(1, 4)
M <- c(100, 90, 70, 80)
opt <- dopt(n = 190, a = N * S, M = M)
var_tst_si(x = opt, N, S)
```

Index