Package 'RItools'

May 17, 2025

Version 0.3-5

Title Randomization Inference Tools

Description Tools for randomization-based inference. Current focus is on the d^2 omnibus test of differences of means following Hansen and Bowers (2008) <doi:10.1214/08-STS254> . This test is useful for assessing balance in matched observational studies or for analysis of outcomes in block-randomized experiments.

License GPL (>= 2)

LazyData true

Depends R (>= 3.5.0), ggplot2, survival

Imports grDevices, abind, xtable, svd, stats, graphics, methods, SparseM, tidyr, tibble, dplyr

Suggests knitr, rmarkdown, testthat, roxygen2, MASS

Enhances optmatch

URL https://cran.r-project.org/package=RItools

RoxygenNote 7.3.2

Encoding UTF-8

NeedsCompilation no

Author Jake Bowers [aut, cre],
Mark Fredrickson [aut],
Ben Hansen [aut],
Josh Errickson [ctb]

Maintainer Jake Bowers <jwbowers@illinois.edu>

Repository CRAN

Date/Publication 2025-05-17 13:30:02 UTC

Contents

`balanceplot` *Create a plot of the balance on variables across different stratifications.*

Description

This plotting function summarizes variable by stratification matrices. For each variable (a row in the `x` argument), the values are under each stratification (the columns of `x`) plotted on the same line.

Usage

```
balanceplot(
  x,
  ordered = FALSE,
  segments = TRUE,
  colors = "black",
  shapes = c(15, 16, 17, 18, 0, 1, 10, 12, 13, 14),
  segments.args = list(col = "grey"),
  points.args = list(cex = 1),
  xlab = "Balance",
  xrange = NULL,
  groups = NULL,
  tiptext = NULL,
  include.legend = TRUE,
```

```
  legend.title = NULL,
  plotfun = .balanceplot,
  ...
)
```

Arguments

x	A matrix of variables (rows) by strata (columns).
ordered	Should the variables be ordered from most to least imbalance on the first statistic?
segments	Should lines be drawn between points for each variable?
colors	Either a vector or a matrix of shape indicators suitable to use as a col argument to the points function. If the argument is a vector, the length should be the same as the number of columns in x. If the argument is a matrix, it should have the same dims as x.
shapes	Either a vector or a matrix of shape indicators suitable to use as a pch argument to the points function. If the argument is a vector, the length should be the same as the number of columns in x. If the argument is a matrix, it should have the same dims as x. <!– The suggested vector has been selected to work with RSVGTipsDevice tool tips.–>
segments.args	A list of arguments to pass to the segments function.
points.args	A list of arguments to pass to the points function.
xlab	The label of the x-axis of the plot.
xrange	The range of x-axis. By default, it is 1.25 times the range of x.
groups	A factor that indicates the group of each row in x. Groups are printed under a common header.
tiptext	ignored (legacy argument retained for internal reasons) <!– If you are using the RSVGTipsDevice library for rendering, you can include an array of the dimensions of x with another dimension of length 2. For example, if there are 4 observations and 2 strata, the array should be 4 by 2 by 2. The tiptext[i, j, 1] entry will be the first line of the tool tip for the data in x[i, j]. Likewise for the second row of the tool tip. –>
include.legend	Should a legend be included?
legend.title	An optional title to attach to the legend.
plotfun	Function to do the plotting; defaults to [RItools:::.balanceplot]
...	Additional arguments to pass to plot.default.

Details

It is conventional to standardize the differences to common scale (e.g. z-scores), but this is not required. When ordered is set to true, plotting will automatically order the data from largest imbalance to smallest based on the first column of x.

You can fine tune the colors and shapes with the like named arguments. Any other arguments to the points function can be passed in a list as points.args. Likewise, you can fine tune the segments between points with segments.args.

Value

Returns NULL, displays plot

See Also

plot.xbal, xBalance, segments, points

Examples

```
set.seed(20121204)

# generate some balance data
nvars <- 10
varnames <- paste("V", letters[1:nvars])

balance_data <- matrix(c(rnorm(n = nvars, mean = 1, sd = 0.5),
                         rnorm(n = nvars, mean = 0, sd = 0.5)),
                       ncol = 2)

colnames(balance_data) <- c("Before Adjustment", "After Matching")

rownames(balance_data) <- varnames

balanceplot(balance_data,
                     colors = c("red", "green"),
                     xlab = "Balance Before/After Matching")

# base R graphics are allowed

abline(v = colMeans(balance_data), lty = 3, col = "grey")
```

---

balanceTest *Standardized Differences for Stratified Comparisons*

Description

Covariate balance, with treatment/covariate association tests

Usage

```
balanceTest(
  fmla,
  data,
  strata = NULL,
  unit.weights,
  stratum.weights = harmonic_times_mean_weight,
  subset,
  include.NA.flags = TRUE,
```

```
  covariate.scales = setNames(numeric(0), character(0)),
  post.alignment.transform = NULL,
  inferentials.calculator = HB08,
  p.adjust.method = "holm"
)
```

Arguments

fmla	A formula containing an indicator of treatment assignment on the left hand side and covariates at right.
data	A data frame in which fmla and strata are to be evaluated.
strata	A list of right-hand-side-only formulas containing the factor(s) identifying the strata, with NULL entries interpreted as no stratification; or a factor with length equal to the number of rows in data; or a data frame of such factors. See below for examples.
unit.weights	Per-unit weight, or 0 if unit does not meet condition specified by subset argument. If there are clusters, the cluster weight is the sum of unit weights of elements within the cluster. Within each stratum, unit weights will be normalized to sum to the number of clusters in the stratum.
stratum.weights	Function returning non-negative weight for each stratum; see details.
subset	Optional: condition or vector specifying a subset of observations to be permitted to have positive unit weights.
include.NA.flags	Present item missingness comparisons as well as covariates themselves?
covariate.scales	covariate dispersion estimates to use as denominators ofstd.diffs (optional).
post.alignment.transform	Optional transformation applied to covariates just after their stratum means are subtracted off. Should accept a vector of weights as its second argument.
inferentials.calculator	Function; calculates 'inferential' statistics. (Not currently intended for use by end-users.)
p.adjust.method	Method of p-value adjustment for the univariate tests. See the p.adjust function for available methods. By default the "holm" method is used.

Details

Given a grouping variable (treatment assignment, exposure status, etc) and variables on which to compare the groups, compare averages across groups and test hypothesis of no selection into groups on the basis of that variable. The multivariate test is the method of combined differences discussed by Hansen and Bowers (2008, Statist. Sci.), a variant of Hotelling's T-squared test; the univariate tests are presented with multiplicity adjustments, the details of which can be controlled by the user. Clustering, weighting and/or stratification variables can be provided, and are addressed by the tests.

The function assembles various univariate descriptive statistics for the groups to be compared: (weighted) means of treatment and control groups; differences of these (adjusted differences); and

adjusted differences as multiples of a pooled s.d. of the variable in the treatment and control groups (standard differences). Pooled s.d.s are calculated with weights but without attention to clustering, and ordinarily without attention to stratification. (If the user does not request unstratified comparisons, overriding the default setting, then pooled s.d.s are calculated with weights corresponding to the first stratification for which comparison is requested. In this case as in the default setting, the same pooled s.d.s are used for standardization under each stratification considered. This facilitates comparison of standard differences across stratification schemes.) Means are contrasted separately for each provided stratifying factor and, by default, for the unstratified comparison, in each case with weights reflecting a standardization appropriate to the designated (post-) stratification of the sample. In the case without stratification or clustering, the only weighting used to calculate treatment and control group means is that provided by the user as `unit.weights`; in the absence of such an argument, these means are unweighted. When there are strata, within-stratum means of treatment or of control observations are calculated using `unit.weights`, if provided, and then these are combined across strata according to a 'effect of treatment on treated'-type weighting scheme. (The function's `stratum.weights` argument figures in the function's inferential calculations but not these descriptive calculations.) To figure a stratum's effect of treatment on treated weight, the sum of all `unit.weights` associated with treatment or control group observations within the stratum is multiplied by the fraction of clusters in that stratum that are associated with the treatment rather than the control condition. (Unless this fraction is 0 or 1, in which case the stratum is downweighted to 0.)

The function also calculates univariate and multivariate inferential statistics, targeting the hypothesis that assignment was random within strata. These calculations also pool `unit.weights`-weighted, within-stratum group means across strata, but the default weighting of strata differs from that of the descriptive calculations. With `stratum.weights=harmonic_times_mean_weight` (the default), each stratum is weighted in proportion to the product of the stratum mean of `unit.weights` and the harmonic mean $1/[(1/a+1/b)/2] = 2*a*b/(a+b)$ of the number of treated units (a) and control units (b) in the stratum; this weighting is optimal under certain modeling assumptions (discussed in Kalton 1968 and Hansen and Bowers 2008, Sections 3.2 and 5). The multivariate assessment is based on a Mahalanobis-type distance that combines each of the univariate mean differences while accounting for correlations among them. It's similar to the Hotelling's T-squared statistic, except standardized using a permutation covariance. See Hansen and Bowers (2008).

In contrast to the earlier function `xBalance` that it is intended to replace, `balanceTest` accepts only binary assignment variables (for now).

`stratum.weights` must be a function of a single argument, a data frame containing the variables in `data` and additionally `Tx.grp`, `stratum.code`, and `unit.weights`, returning a named numeric vector of non-negative weights identified by stratum. (For an example, enter `getFromNamespace("harmonic", "RItools")`.) the data `stratum.weights` function.

If the stratifying factor has NAs, these cases are dropped. On the other hand, if NAs in a covariate are found then those observations are dropped for descriptive calculations and "imputed" to the stratum mean of the variable for inferential calculations. When covariate values are dropped due to missingness, proportions of observations not missing on that variable are recorded and returned. The printed output presents non-missing proportions alongside of the variables themselves, distinguishing the former by placing them at the bottom of the list and enclosing the variable's name in parentheses. If a variable shares a missingness pattern with other another variable, its missingness information may be labeled with the name of the other variable in the output.

Value

An object of class `c("balancetest", "xbal", "list")`. Several methods are inherited from the "xbal" class returned by `xBalance` function.

Note

Evidence pertaining to the hypothesis that a treatment variable is not associated with differences in covariate values is assessed by comparing the differences of means, without standardization, to their distributions under hypothetical shuffles of the treatment variable, a permutation or randomization distribution. For the unstratified comparison, this reference distribution consists of differences as the treatment assignments of clusters are freely permuted. For stratified comparisons, the reference distributions describes re-randomizations of this type performed separately in each stratum. Significance assessments are based on large-sample approximations to these reference distributions.

Author(s)

Ben Hansen and Jake Bowers and Mark Fredrickson

References

Hansen, B.B. and Bowers, J. (2008), "Covariate Balance in Simple, Stratified and Clustered Comparative Studies," *Statistical Science* **23**.

Kalton, G. (1968), "Standardization: A technique to control for extraneous variables," *Applied Statistics* **17**, 118–136.

See Also

`HB08`

Examples

```
data(nuclearplants)
## No strata
balanceTest(pr ~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
         data=nuclearplants)

## Stratified
## Note use of the `. - cost` to use all columns except `cost`
balanceTest(pr ~ . - cost + strata(pt),
         data=nuclearplants)

##Missing data handling.
testdata <- nuclearplants
testdata$date[testdata$date < 68] <- NA
balanceTest(pr ~ . - cost + strata(pt),
            data = testdata)

## Variable-by-variable Wilcoxon rank sum tests, with an omnibus test
## of multivariate differences on rank scale.
balanceTest(pr ~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
```

```
          data = nuclearplants,
        post.alignment.transform = function(x, weights) rank(x))

## (Note that the post alignment transform is expected to be a function
## accepting a second argument, even if the argument is not used.
## The unit weights vector will be provided as this second argument,
## enabling use of e.g. `post.alignment.transform=Hmisc::wtd.rank`
## to furnish a version of the Wilcoxon test even when there are clusters and/or weights.)

## An experiment where clusters of individuals are assigned to treatment within strata
## assessing balance of cluster level treatment on both cluster
## and individual level baseline attributes
data(ym_long)
## Look at balance on teriles of cluster size as well as other variables
teriles <- quantile(ym_long$n_practice, seq(1/3,1,by=1/3))
teriles <- c(0, teriles)

balanceTest(trt ~ cut(n_practice, teriles)+assessed+hypo+lipid+
            aspirin+strata(assess_strata)+cluster(practice),
            data=ym_long)
```

`balanceTest.make.stratwts` *balanceTest helper function*

Description

Makes strata weights

Usage

```
balanceTest.make.stratwts(stratum.weights, ss.df, zz, data, normalize.weights)
```

Arguments

`stratum.weights`	Weights
`ss.df`	df.
`zz`	treatment
`data`	data
`normalize.weights`	weights

Value

list

`balanceTestEngine` *xBalance helper function*

Description

Make engine

Usage

```
balanceTestEngine(
  ss,
  zz,
  mm,
  report,
  swt,
  s.p,
  normalize.weights,
  zzname,
  post.align.trans,
  p.adjust.method
)
```

Arguments

`ss`	ss
`zz`	zz
`mm`	mm
`report`	report
`swt`	swt
`s.p`	s.p
`normalize.weights`	normalize.weights
`zzname`	zzname
`post.align.trans`	post.align.trans
`p.adjust.method`	Method to adjust P.

Value

List

flatten.xbalresult *Flattens xBalance output.*

Description

Details...

Usage

```
flatten.xbalresult(
  x,
  show.signif.stars = getOption("show.signif.stars"),
  show.pvals = !show.signif.stars,
  ...
)
```

Arguments

x	x
show.signif.stars	Should signif stars be shown?
show.pvals	Should p-vals be shown?
...	Ignored

Value

Structure

formula.xbal *Returns* formula *attribute of an* xbal *object.*

Description

Returns formula attribute of an xbal object.

Usage

```
## S3 method for class 'xbal'
formula(x, ...)
```

Arguments

x	An xbal object.
...	Ignored.

Value

The formula corresponding to `xbal`.

`gramian_reduction` *Helper function to slm_fit_csr*

Description

This function generates a matrix that can be used to reduce the dimensions of x'x and xy such that positive definiteness is ensured and more practically, that SparseM::chol will work

Usage

```
gramian_reduction(zeroes)
```

Arguments

`zeroes`	logical vector indicating which entries of the diagonal of x'x are zeroes.

Value

SparseM matrix that will reduce the dimension of x'x and xy

`makePval` *Get p-value for Z-stats*

Description

Get p-value for Z-stats

Usage

```
makePval(zs)
```

Arguments

`zs`	A Z-statistic.

Value

A P-value

naImpute *Impute NA's*

Description

Function used to fill NAs with imputation values, while adding NA flags to the data.

Usage

```
naImpute(FMLA, DATA, impfn = median, na.rm = TRUE, include.NA.flags = TRUE)
```

Arguments

FMLA	Formula
DATA	Data
impfn	Function for imputing.
na.rm	What to do with NA's
include.NA.flags	Should NA flags be included

Value

Structure

nuclearplants *Nuclear Power Station Construction Data*

Description

The data relate to the construction of 32 light water reactor (LWR) plants constructed in the U.S.A in the late 1960's and early 1970's. The data was collected with the aim of predicting the cost of construction of further LWR plants. 6 of the power plants had partial turnkey guarantees and it is possible that, for these plants, some manufacturers' subsidies may be hidden in the quoted capital costs.

Usage

```
nuclearplants
```

Format

A data frame with 32 rows and 11 columns

- cost: The capital cost of construction in millions of dollars adjusted to 1976 base.
- date: The date on which the construction permit was issued. The data are measured in years since January 1 1990 to the nearest month.
- t1: The time between application for and issue of the construction permit.
- t2: The time between issue of operating license and construction permit.
- cap: The net capacity of the power plant (MWe).
- pr: A binary variable where 1 indicates the prior existence of a LWR plant at the same site.
- ne: A binary variable where 1 indicates that the plant was constructed in the north-east region of the U.S.A.
- ct: A binary variable where 1 indicates the use of a cooling tower in the plant.
- bw: A binary variable where 1 indicates that the nuclear steam supply system was manufactured by Babcock-Wilcox.
- cum.n: The cumulative number of power plants constructed by each architect-engineer.
- pt: A binary variable where 1 indicates those plants with partial turnkey guarantees.

Source

The data were obtained from the `boot` package, for which they were in turn taken from Cox and Snell (1981). Although the data themselves are the same as those in the `nuclear` data frame in the `boot` package, the row names of the data frame have been changed. (The new row names were selected to ease certain demonstrations in `optmatch`.)

This documentation page is also adapted from the `boot` package, written by Angelo Canty and ported to R by Brian Ripley.

References

Cox, D.R. and Snell, E.J. (1981) *Applied Statistics: Principles and Examples*. Chapman and Hall.

`plot.balancetest` *Plot of balance across multiple strata*

Description

The plot allows a quick visual comparison of the effect of different stratification designs on the comparability of different variables. This is not a replacement for the omnibus statistical test reported as part of `print.xbal`. This plot does allow the analyst an easy way to identify variables that might be the primary culprits of overall imbalances and/or a way to assess whether certain important covariates might be imbalanced even if the omnibus test reports that the stratification overall produces balance.

Usage

```
## S3 method for class 'balancetest'
plot(
  x,
  xlab = "Standardized Differences",
  statistic = "std.diff",
  absolute = FALSE,
  strata.labels = NULL,
  variable.labels = NULL,
  groups = NULL,
  ...
)
```

Arguments

x	An object returned by xBalance
xlab	The label for the x-axis of the plot
statistic	The statistic to plot. The default choice of standardized difference is a good choice as it will have roughly the same scale for all plotted variables.
absolute	Convert the results to the absolute value of the statistic.
strata.labels	A named vector of the from c(strata1 = "Strata Label 1", ...) that maps the stratification schemes to textual labels.
variable.labels	A named vector of the from c(var1 = "Var Label1", ...) that maps the variables to textual labels.
groups	A vector of group names for each variable in x$results. By default, factor level variables will be grouped.
...	additional arguments to pass to balanceplot

Details

By default all variables and all strata are plotted. The scope of the plot can be reduced by using the subset.xbal function to make a smaller xbal object with only the desired variables or strata.

balanceTest can produce several different summary statistics for each variable, any of which can serve as the data for this plot. By default, the standardized differences between treated and control units makes a good choice as all variables are on the same scale. Other statistics can be selected using the statistic argument.

The result of this function is a ggplot object. Most display of the plot can be manipulated using additional commands appended to the plot option. For example, the entire theme of the plot can be changed to black and white using plot(b) + theme_bw(), where b is the result of a call to balanceTest. The points on the plot are known as "values", so colors or symbols used for each strata can be updated using the scale_color_manual function. For example, plot(b) + scale_color_manaual(values = c('red', 'green', 'blue')) for a balance test of three stratification variables.

Value

A ggplot2 object that can be further manipulated (e.g., to set the colors or text).

See Also

balanceTest, ggplot

---

plot.xbal *Plot of balance across multiple strata*

Description

The plot allows a quick visual comparison of the effect of different stratification designs on the comparability of different variables. This is not a replacement for the omnibus statistical test reported as part of print.xbal. This plot does allow the analyst an easy way to identify variables that might be the primary culprits of overall imbalances and/or a way to assess whether certain important covariates might be imbalanced even if the omnibus test reports that the stratification overall produces balance.

Usage

```
## S3 method for class 'xbal'
plot(
  x,
  xlab = "Standardized Differences",
  statistic = "std.diff",
  absolute = FALSE,
  strata.labels = NULL,
  variable.labels = NULL,
  groups = NULL,
  ggplot = FALSE,
  ...
)
```

Arguments

x	An object returned by xBalance
xlab	The label for the x-axis of the plot
statistic	The statistic to plot. The default choice of standardized difference is a good choice as it will have roughly the same scale for all plotted variables.
absolute	Convert the results to the absolute value of the statistic.
strata.labels	A named vector of the from c(strata1 = "Strata Label 1", ...) that maps the stratification schemes to textual labels.
variable.labels	A named vector of the from c(var1 = "Var Label1", ...) that maps the variables to textual labels.

groups	A vector of group names for each variable in x$results. By default, factor level variables will be grouped.
ggplot	Use ggplot2 to create figure. By default, uses base R graphics.
...	additional arguments to pass to balanceplot

Details

By default all variables and all strata are plotted. The scope of the plot can be reduced by using the subset.xbal function to make a smaller xbal object with only the desired variables or strata.

xBalance can produce several different summary statistics for each variable, any of which can serve as the data for this plot. By default, the standardized differences between treated and control units makes a good choice as all variables are on the same scale. Other statistics can be selected using the statistic argument.

Value

Returns NULL, displays plot

See Also

xBalance, subset.xbal, balanceplot

Examples

```
data(nuclearplants)

xb <- xBalance(pr ~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
               strata = list(none = NULL, pt = ~pt),
               data = nuclearplants)

# Using the default grouping:
plot(xb, variable.labels = c(date = "Date",
             t1 = "Time 1",
             t2 = "Time 2",
             cap = "Capacity",
             ne = "In North East",
             ct = "Cooling Tower",
             bw = "Babcock-Wilcox",
             cum.n = "Total Plants Built"),
     strata.labels = c("--" = "Raw Data", "pt" = "Partial Turn-key"),
     absolute = TRUE)

# Using user supplied grouping
plot(xb, variable.labels = c(date = "Date",
             t1 = "Time 1",
             t2 = "Time 2",
             cap = "Capacity",
             ne = "In North East",
             ct = "Cooling Tower",
             bw = "Babcock-Wilcox",
             cum.n = "Total Plants Built"),
```

```
strata.labels = c("--" = "Raw Data", "pt" = "Partial Turn-key"),
absolute = TRUE,
groups = c("Group A", "Group A", "Group A", "Group B",
           "Group B", "Group B", "Group A", "Group B"))
```

---

print.xbal *Printing xBalance and balanceTest Objects*

Description

A print method for balance test objects produced by xBalance and balanceTest.

Usage

```
## S3 method for class 'xbal'
print(
  x,
  which.strata = dimnames(x$results)[["strata"]],
  which.stats = dimnames(x$results)[["stat"]],
  which.vars = dimnames(x$results)[["vars"]],
  print.overall = TRUE,
  digits = NULL,
  printme = TRUE,
  show.signif.stars = getOption("show.signif.stars"),
  show.pvals = !show.signif.stars,
  horizontal = TRUE,
  report = NULL,
  ...
)
```

Arguments

x	An object of class "xbal" which is the result of a call to xBalance.
which.strata	The stratification candidates to include in the printout. Default is all.
which.stats	The test statistics to include. Default is all those requested from the call to xBalance.
which.vars	The variables for which test information should be displayed. Default is all.
print.overall	Should the omnibus test be reported? Default is TRUE.
digits	To how many digits should the results be displayed? Default is max(2,getOptions("digits")-4).
printme	Print the table to the console? Default is TRUE.
show.signif.stars	Use stars to indicate z-statistics larger than conventional thresholds. Default is TRUE.
show.pvals	Instead of stars, use p-values to summarize the information in the z-statistics. Default is FALSE.

horizontal	Display the results for different candidate stratifications side-by-side (Default, TRUE), or as a list for each stratification (FALSE).
report	What to report.
...	Other arguements. Not currently used.

Value

vartable The formatted table of variable-by-variable statistics for each stratification.

overalltable If the overall Chi-squared statistic is requested, a formatted version of that table is returned.

See Also

xBalance, balanceTest

Examples

```
data(nuclearplants)


xb1 <- balanceTest(pr ~ date + t1 + t2 + cap + ne + ct + bw + cum.n + strata(pt),
         data = nuclearplants)

print(xb1)

print(xb1, show.pvals = TRUE)

print(xb1, horizontal = FALSE)

## The following doesn't work yet.
## Not run: print(xb1, which.vars=c("date","t1"),
         which.stats=c("adj.means","z.scores","p.values"))
## End(Not run)

## The following example prints the adjusted means
## labeled as "treatmentvar=0" and "treatmentvar=1" using the
## formula provided to xBalance().

# This is erroring with the change to devtools, FIXME
## Not run: print(xb1,
      which.vars = c("date", "t1"),
      which.stats = c("pr=0", "pr=1", "z", "p"))
## End(Not run)

## Only printing out a specific stratification factor
xb2 <- balanceTest(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n + strata(pt),
         data = nuclearplants)

print(xb2, which.strata = "pt")
```

scale.DesignOptions *Scale DesignOptions*

Description

Scale DesignOptions

Usage

```
## S3 method for class 'DesignOptions'
scale(x, center = TRUE, scale = TRUE)
```

Arguments

x	DesignOptions object
center	logical, or a function acceptable as post.alignment.transform arg of alignDesignsByStrata()
scale	logical, whether to scale

slm_fit_csr *SparseM::slm.fit.csr, made tolerant to faults that recur in RItools*

Description

[SparseM's slm.fit.csr()] expects a full-rank x that's not just a column of 1s. This variant somewhat relaxes these expectations.

Usage

```
slm_fit_csr(x, y, ...)
```

Arguments

x	As slm.fit.csr
y	As slm.fit.csr
...	As slm.fit.csr

Details

'slm.fit.csr' has a bug for intercept only models (admittedly, these are generally a little silly to be done as a sparse matrix), but in order to avoid duplicate code, if everything is in a single strata, we use the intercept only model.

This function's expectation of x is that either it has full column rank, or the reduced submatrix of x that excludes all-zero columns has full column rank. (When this expectation is not met, it's likely that [SparseM::chol()] will fail, causing this function to error; the error messages won't necessarily suggest this.) The positions of nonzero x-columns (ie columns with nonzero entries) are returns as the value of 'gramian_reduction_index', while 'chol' is the Cholesky decomposition of that submatrix's Gramian.

Value

A list consisting of:

`coefficients`	coefficients
`chol`	Cholesky factor of Gramian matrix $x'x$
`residuals`	residuals
`fitted`	fitted values
`df.residual`	degrees of freedom
`gramian_reduction_index`	Column indices identifying reduction of x matrix of which Gramian is taken; see Details

---

`SparseM_solve` *Helper function to slm_fit_csr*

Description

This function performs some checks and takes action to ensure positive definiteness of matrices passed to SparseM functions.

Usage

```
SparseM_solve(x, y, ...)
```

Arguments

`x`	A slm.fit.csr
`y`	A slm.fit.csr
`...`	A slm.fit.csr

Value

list containing coefficients (vector or matrix), the Cholesky decomposition (of class matrix.csr.chol), and a vector specifying the indices of which values on the diagonal of x'x are nonzero. These are named "coef", "chol" and "gramian_reduction_index", respectively.

`StratumWeightedDesignOptions-class`
Stratum Weighted DesignOptions

Description

Stratum Weighted DesignOptions

Slots

`Sweights` stratum weights

subset.xbal — *Select variables, strata, and statistics from a* xbal *or* balancetest *object*

Description

If any of the arguments are not specified, all the of relevant items are included.

Usage

```
## S3 method for class 'xbal'
subset(x, vars = NULL, strata = NULL, stats = NULL, tests = NULL, ...)
```

Arguments

x	The xbal object, the result of a call to xBalance or balanceTest
vars	The variable names to select.
strata	The strata names to select.
stats	The names of the variable level statistics to select.
tests	The names of the group level tests to select.
...	Other arguments (ignored)

Value

A xbal object with just the appropriate items selected.

tidy.xbal — broom::tidy()/glance() *methods for* balanceTest() *results*

Description

Portion out the value of a balanceTest() call in a manner consistent with assumptions of the broom package.

Usage

```
tidy.xbal(
  x,
  strata = dimnames(x[["results"]])[["strata"]][1],
  varnames_crosswalk = c(z = "statistic", p = "p.value"),
  format = FALSE,
  digits = max(2, getOption("digits") - 4),
  ...
)

glance.xbal(x, strata = dimnames(x[["results"]])[["strata"]][1], ...)
```

Arguments

x	object of class "xbal", result of balanceTest() or xBalance()
strata	which stratification to return info about? Defaults to last one specified in originating function call (which appears first in the xbal array).
varnames_crosswalk	character vector of new names for xbal columns, named by the xbal column
format	if true, apply [RItools:::original_units_var_formatter()] to suitable subarray en route
digits	passed to [RItools:::original_units_var_formatter()]
...	Additional arguments passed to [RItools:::original_units_var_formatter()]

Details

tidy.xbal() gives per-variable statistics whereas glance.xbal() extracts combined-difference related calculations. In both cases one has to specify which stratification one wants statistics about, as xbal objects can store info about several stratifications. tidy.xbal() has a parameter varnames_crosswalk not shared with glance.xbal(). It should be a named character vector, the elements of which give names of columns to be returned and the names of which correspond to columns of xbal objects' 'results' entry. Its ordering dictates the order of the result. The default value translates between conventional xbal column names and broom package conventional names.

vars variable name

Control mean of LHS variable = 0 group

Treatment mean of LHS variable = 1 group

adj.diff T - C diff w/ direct standardization for strata if applicable

std.diff adj.diff/pooled.sd

pooled.sd pooled SD

statistic z column from the xbal object

p.value p column from the xbal object

Additional parameters beyond those listed here are ignored (at this time).

Value

data frame composed of: for [RItools::tidy()], a column of variable labels (vars) and additional columns of balance-related stats; for [RItools::glance()], scalars describing a combined differences test, if found, and otherwise NULL.

`withOptions` *Safe way to temporarily override options()*

Description

Safe way to temporarily override options()

Usage

```
withOptions(optionsToChange, fun)
```

Arguments

`optionsToChange`	Which options.
`fun`	Function to run with new options.

Value

Result of `fun`.

`xBalance` *Standardized Differences for Stratified Comparisons*

Description

Given covariates, a treatment variable, and a stratifying factor, calculates standardized mean differences along each covariate, with and without the stratification and tests for conditional independence of the treatment variable and the covariates within strata.

Usage

```
xBalance(
  fmla,
  strata = list(unstrat = NULL),
  data,
  report = c("std.diffs", "z.scores", "adj.means", "adj.mean.diffs",
    "adj.mean.diffs.null.sd", "chisquare.test", "p.values", "all")[1:2],
  stratum.weights = harmonic,
  na.rm = FALSE,
  covariate.scaling = NULL,
  normalize.weights = TRUE,
  impfn = median,
  post.alignment.transform = NULL,
  pseudoinversion_tol = .Machine$double.eps
)
```

Arguments

fmla	A formula containing an indicator of treatment assignment on the left hand side and covariates at right.
strata	A list of right-hand-side-only formulas containing the factor(s) identifying the strata, with NULL entries interpreted as no stratification; or a factor with length equal to the number of rows in data; or a data frame of such factors. See below for examples.
data	A data frame in which fmla and strata are to be evaluated.
report	Character vector listing measures to report for each stratification; a subset of c("adj.means","adj.mean.diffs", "adj.mean.diffs.null.sd", "chisquare.test","std.diffs "z.scores", "p.values", "all"). P-values reported are two-sided for the null-hypothesis of no effect. The option "all" requests all measures.
stratum.weights	Weights to be applied when aggregating across strata specified by strata, defaulting to weights proportional to the harmonic mean of treatment and control group sizes within strata. This can be either a function used to calculate the weights or the weights themselves; if strata is a data frame, then it can be such a function, a list of such functions, or a data frame of stratum weighting schemes corresponding to the different stratifying factors of strata. See details.
na.rm	Whether to remove rows with NAs on any variables mentioned on the RHS of fmla (i.e. listwise deletion). Defaults to FALSE, wherein rows aren't deleted but for each variable with NAs a missing-data indicator variable is added to the variables on which balance is calculated and medians are imputed for the variable with missing data (in RItools versions 0.1-9 and before the default imputation was the mean, in RItools versions 0.1-11 and henceforth the default is the median). See the example below.
covariate.scaling	A scale factor to apply to covariates in calculating std.diffs. If NULL, xBalance pools standard deviations of each variable in the treatment and control group (defining these groups according to whether the LHS of formula is greater than or equal to 0). Also, see details.
normalize.weights	If TRUE, then stratum weights are normalized so as to sum to 1. Defaults to TRUE.
impfn	A function to impute missing values when na.rm=FALSE. Currently median. To impute means use mean.default.
post.alignment.transform	Optional transformation applied to covariates just after their stratum means are subtracted off.
pseudoinversion_tol	The function uses a singular value decomposition to invert a covariance matrix. Singular values less than this tolerance will be treated as zero.

Details

Note: the newer balanceTest function provides the same functionality as xBalance with additional support for clustered designs. While there are no plans to deprecate xBalance, users are encouraged to use balanceTest going forward.

In the unstratified case, the standardized difference of covariate means is the mean in the treatment group minus the mean in the control group, divided by the S.D. (standard deviation) in the same variable estimated by pooling treatment and control group S.D.s on the same variable. In the stratified case, the denominator of the standardized difference remains the same but the numerator is a weighted average of within-stratum differences in means on the covariate. By default, each stratum is weighted in proportion to the harmonic mean $1/[(1/a + 1/b)/2] = 2 * a * b/(a + b)$ of the number of treated units (a) and control units (b) in the stratum; this weighting is optimal under certain modeling assumptions (discussed in Kalton 1968, Hansen and Bowers 2008). This weighting can be modified using the `stratum.weights` argument; see below.

When the treatment variable, the variable specified by the left-hand side of `fmla`, is not binary, `xBalance` calculates the covariates' regressions on the treatment variable, in the stratified case pooling these regressions across strata using weights that default to the stratum-wise sum of squared deviations of the treatment variable from its stratum mean. (Applied to binary treatment variables, this recipe gives the same result as the one given above.) In the numerator of the standardized difference, we get a "pooled S.D." from separating units into two groups, one in which the treatment variable is 0 or less and another in which it is positive. If `report` includes "adj.means", covariate means for the former of these groups are reported, along with the sums of these means and the covariates' regressions on either the treatment variable, in the unstratified ("pre") case, or the treatment variable and the strata, in the stratified ("post") case.

`stratum.weights` can be either a function or a numeric vector of weights. If it is a numeric vector, it should be non-negative and it should have stratum names as its names. (i.e., its names should be equal to the levels of the factor specified by `strata`.) If it is a function, it should accept one argument, a data frame containing the variables in `data` and additionally `Tx.grp` and `stratum.code`, and return a vector of non-negative weights with stratum codes as names; for an example, do `getFromNamespace("harmonic", "RItools")`.

If `covariate.scaling` is not `NULL`, no scaling is applied. This behavior is likely to change in future versions. (If you want no scaling, set `covariate.scaling=1`, as this is likely to retain this meaning in the future.)

`adj.mean.diffs.null.sd` returns the standard deviation of the Normal approximated randomization distribution of the strata-adjusted difference of means under the strict null of no effect.

Value

An object of class `c("xbal", "list")`. There are `plot`, `print`, and `xtable` methods for class `"xbal"`; the `print` method is demonstrated in the examples.

Note

Evidence pertaining to the hypothesis that a treatment variable is not associated with differences in covariate values is assessed by comparing the differences of means (or regression coefficients), without standardization, to their distributions under hypothetical shuffles of the treatment variable, a permutation or randomization distribution. For the unstratified comparison, this reference distribution consists of differences (more generally, regression coefficients) when the treatment variable is permuted without regard to strata. For the stratified comparison, the reference distribution is determined by randomly permuting the treatment variable within strata, then re-calculating the treatment-control differences (regressions of each covariate on the permuted treatment variable). Significance assessments are based on the large-sample Normal approximation to these reference distributions.

Author(s)

Ben Hansen and Jake Bowers and Mark Fredrickson

References

Hansen, B.B. and Bowers, J. (2008), "Covariate Balance in Simple, Stratified and Clustered Comparative Studies," *Statistical Science* **23**.

Kalton, G. (1968), "Standardization: A technique to control for extraneous variables," *Applied Statistics* **17**, 118–136.

See Also

balanceTest

Examples

```
data(nuclearplants)
##No strata, default output
xBalance(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
         data=nuclearplants)

##No strata, all output
xBalance(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
         data=nuclearplants,
         report=c("all"))

##Stratified, all output
xBalance(pr~.-cost-pt, strata=factor(nuclearplants$pt),
         data=nuclearplants,
         report=c("adj.means", "adj.mean.diffs",
                  "adj.mean.diffs.null.sd",
                  "chisquare.test", "std.diffs",
                  "z.scores", "p.values"))

##Comparing unstratified to stratified, just adjusted means and
#omnibus test
xBalance(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
         strata=list(unstrat=NULL, pt=~pt),
         data=nuclearplants,
         report=c("adj.means", "chisquare.test"))

##Comparing unstratified to stratified, just adjusted means and
#omnibus test
xBalance(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
         strata=data.frame(unstrat=factor('none'),
           pt=factor(nuclearplants$pt)),
         data=nuclearplants,
         report=c("adj.means", "chisquare.test"))

##Missing data handling.
testdata<-nuclearplants
```

```
testdata$date[testdata$date<68]<-NA

##na.rm=FALSE by default
xBalance(pr ~ date, data = testdata, report="all")
xBalance(pr ~ date, data = testdata, na.rm = TRUE,report="all")

##To match versions of RItools 0.1-9 and older, impute means
#rather than medians.
##Not run, impfn option is not implemented in the most recent version
## Not run: xBalance(pr ~ date, data = testdata, na.rm = FALSE,
           report="all", impfn=mean.default)
## End(Not run)

##Comparing unstratified to stratified, just one-by-one wilcoxon
#rank sum tests and omnibus test of multivariate differences on
#rank scale.
xBalance(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
         strata=data.frame(unstrat=factor('none'),
           pt=factor(nuclearplants$pt)),
         data=nuclearplants,
         report=c("adj.means", "chisquare.test"),
 post.alignment.transform=rank)
```

`xtable.xbal` *An* `xtable` *method for* `xbal` *and* `balancetest` *objects*

Description

This function uses the `xtable` package framework to display the results of a call to `balanceTest` in LaTeX format. At the moment, it ignores the omnibus chi-squared test information.

Usage

```
## S3 method for class 'xbal'
xtable(
  x,
  caption = NULL,
  label = NULL,
  align = c("l", rep("r", ncol(xvardf))),
  digits = 2,
  display = NULL,
  auto = FALSE,
  col.labels = NULL,
  ...
)
```

Arguments

x	An object resulting from a call to balanceTest or xBalance.
caption	See xtable.
label	See xtable.
align	See xtable. Our default (as of version 0.1-7) is right-aligned columns; for decimal aligned columns, see details, below.
digits	See xtable. Default is 2.
display	See xtable.
auto	See xtable.
col.labels	Labels for the columns (the test statistics). Default are come from the call to print.xbal.
...	Other arguments to print.xbal.

Details

The resulting LaTeX will present one row for each variable in the formula originally passed to balanceTest, using the variable name used in the original formula. If you wish to have reader friendly labels instead of the original variables names, see the code examples below.

To get decimal aligned columns, specify align=c("l",rep(".", <ncols>)), where <ncols> is the number of columns to be printed, in your call to xtable. Then use the dcolumn package and define ''.'' within LaTeX: add the lines \usepackage{dcolumn} and \newcolumntype{.}{D{.}{.}{2.2}} to your LaTeX document's preamble.

Value

This function produces an xtable object which can then be printed with the appropriate print method (see print.xtable).

Examples

```
data(nuclearplants)
require(xtable)

# Test balance on a variety of variables, with the 'pr' factor
# indicating which sites are control and treatment units, with
# stratification by the 'pt' factor to group similar sites
xb1 <- balanceTest(pr ~ date + t1 + t2 + cap + ne + ct + bw + cum.n + strata(pt),
                data = nuclearplants)

xb1.xtab <- xtable(xb1) # This table has right aligned columns

# Add user friendly names in the final table
rownames(xb1.xtab) <- c("Date", "Application to Contruction Time",
"License to Construction Time", "Net Capacity", "Northeast Region", "Cooling Tower",
"Babcock-Wilcox Steam", "Cumlative Plants")

print(xb1.xtab,
```

```
add.to.row = attr(xb1.xtab, "latex.add.to.row"),
hline.after = c(0, nrow(xb1.xtab)),
sanitize.text.function = function(x){x},
floating = TRUE,
floating.environment = "sidewaystable")
```

---

ym_long *ASSIST Trial Data from Yudkin and Moher 2001*

Description

The ASSIST Trial baseline data from Yudkin and Moher 2001 consist of 21 general practices containing 2142 patients used for the design of a randomized trial which assigned to three treatments aiming to compare methods of preventing coronary heart disease. We have expanded the aggregated data from the practice level to the individual level and added a simulated randomized treatment variable.

Usage

```
ym_long
```

Format

A data frame with 2142 rows and 9 columns

- practice: Identifier for the practice
- id: Identifier for patients
- n_practice: Number of patients with CHD in the practice
- assessed: Whether a patient was coded as "adequately assessed" (the outcome of the study, measured here at baseline).
- aspirin: Whether a patient was treated with aspirin at baseline.
- hypo: Whether a patient was treated with hypotensives at baseline.
- lipid: Whether patient was treated with lipid-lowering drugs at baseline.
- assess_strata: Strata of the practice defined by the proportion of people adequately assessed at baseline (three strata, following Yudkin and Moher 2001).
- trt: A simulated binary treatment, assigned at random at the practice level within levels of assess_strata.

Source

The data come from Table II on page 345 of Yudkin and Moher 2001, Statistics in Medicine.

References

Yudkin, P. L. and Moher, M. 2001. "Putting theory into practice: a cluster randomized trial with a small number of clusters" *Statistics in Medicine*, 20:341-349.

ym_short *ASSIST Trial Data from Yudkin and Moher 2001*

Description

The ASSIST Trial baseline data from Yudkin and Moher 2001 consist of 21 general practices containing 2142 patients used for the design of a randomized trial which assigned to three treatments aiming to compare methods of preventing coronary heart disease. This data frame is aggregated to the practice level. We added a simulated randomized treatment variable.

Usage

```
ym_short
```

Format

A data frame with 21 rows and 8 columns

- practice: Identifier for the practice
- n_practice: Number of patients with CHD in the practice
- assessed: Whether a patient was coded as "adequately assessed" (the outcome of the study, measured here at baseline).
- aspirin: Whether a patient was treated with aspirin at baseline.
- hypo: Whether a patient was treated with hypotensives at baseline.
- lipid: Whether patient was treated with lipid-lowering drugs at baseline.
- assess_strata: Strata of the practice defined by the proportion of people adequately assessed at baseline (three strata, following Yudkin and Moher 2001).
- trt: A simulated binary treatment, assigned at random at the practice level within levels of assess_strata.

Source

The data come from Table II on page 345 of Yudkin and Moher 2001, Statistics in Medicine.

References

Yudkin, P. L. and Moher, M. 2001. "Putting theory into practice: a cluster randomized trial with a small number of clusters" *Statistics in Medicine*, 20:341-349.

Index