

# Package ‘RIttools’

February 14, 2012

**Version** 0.1-11

**Date** 2010-03-04

**Title** Randomization inference tools

**Author** Jake Bowers <jwbowers@illinois.edu>, Mark Fredrickson  
<mark.m.fredrickson@gmail.com>, and Ben Hansen <ben.hansen@umich.edu>

**Maintainer** Jake Bowers <jwbowers@illinois.edu>

**Description** Tools for randomization inference.

**License** GPL (>= 2)

**Depends** R (>= 2.2.0), methods

**Imports** graphics, stats, lattice, grid, SparseM, xtable

**Suggests** optmatch, xtable

**Enhances** optmatch, xtable

**URL** <http://CRAN.R-project.org/package=RIttools>,  
<http://www.jakebowers.org/RIttools.html>

**Repository** CRAN

**Date/Publication** 2011-03-05 16:39:38

## R topics documented:

nuclearplants . . . . .	2
plot.xbal . . . . .	3
print.xbal . . . . .	4
xBalance . . . . .	6
xtable.xbal . . . . .	9

<b>Index</b>	<b>12</b>
--------------	-----------

---

nuclearplants

*Nuclear Power Station Construction Data*


---

### Description

The nuclearplants data frame has 32 rows and 11 columns.

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

This data frame contains the following 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.xbal *Plotting xBalance Objects*

---

## Description

A plot method for xBalance objects. WORK IN PROGRESS. PATCHES AND CODE SUGGESTIONS APPRECIATED.

We aim this plot to be a diagnostic tool rather than a publication quality presentation tool.

## Usage

```
## S3 method for class 'xbal'
plot(x, adjustxaxis=.25, segments=TRUE, legend=TRUE,
      mar=c(3,3,2,0)+0.1, mgp=c(1.5, .5, 0), tck=-.01,
      which.strata=dimnames(x$results)[["strata"]], thestratalabs=which.strata,
      which.stats="std.diff", ##dimnames(x$results)[["stat"]],
      which.vars=dimnames(x$results)[["vars"]], thevarlabs=which.vars,
      thexlab="Standardized Differences",
      thecols=rainbow(length(which.strata)),
      thesymbols=c(19,22,23,24,25)[1:length(which.strata)],...)
```

## Arguments

x	An object of class "xbal" — the result of a call to xBalance()
adjustxaxis	amount by which the x-axis should be expanded.
segments	Should thin horizontal lines be plotted connecting the statistics for the different stratifications.
legend	Should a legend be plotted?
mar	Preliminary margin setting
mgp	Preliminary setting for axis labels
tck	Length of the tick marks
which.strata	The stratification candidates to include in the printout. Default is all.
thestratalabs	The text labels for the strata.
which.stats	a character vector of length 1. The test statistics to include. Default is the standardized difference.
which.vars	The variables for which test information should be displayed. Default is all.
thevarlabs	The text labels for the variables.
thexlab	The label for the x-axis (should tell viewers about the statistic chosen).
thecols	A vector of colors either (a) one per strata or (b) one for all strata.
thesymbols	A vector of plotting symbols either (a) one per strata or (b) one for all strata.
...	other arguments to the <code>plot.default</code> function setting up the plotting region.

**Value**

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.

**References**

[xBalance](#)

**Examples**

```
data(nuclearplants)

xb0<-xBalance(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n,data=nuclearplants)

plot(xb0)

xb1<-xBalance(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
              strata=data.frame(unstrat=factor(character(32)),
                                pt=factor(nuclearplants$pt)),
              data=nuclearplants,
              report=c("adj.means","adj.mean.diffs","std.diffs", "z.scores", "chisquare.test", "p.values"))

plot(xb1)
```

---

print.xbal

*Printing xBalance Objects*

---

**Description**

A print method for xBalance objects.

**Usage**

```
## S3 method for class 'xbal'
print(x,which.strata=dimnames(x$results)[["strata"]],which.stats=dimnames(x$results)[["stat"]],which
```

**Arguments**

<code>x</code>	An object of class "xbal" — the result of a call to <code>xBalance()</code>
<code>which.strata</code>	The stratification candidates to include in the printout. Default is all.
<code>which.stats</code>	The test statistics to include. Default is all those requested from the call to <code>xBalance</code> .

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 $\max(2, \text{getOptions}(\text{"digits"}) - 4)$ .
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).
...	Other arguments. Not currently used.

**Value**

variable	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.

**References**

[xBalance](#)

**Examples**

```

data(nuclearplants)

xb0<-xBalance(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n,data=nuclearplants)

print(xb0)

xb1<-xBalance(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
              strata=data.frame(unstrat=factor(character(32)),
                                pt=factor(nuclearplants$pt)),
              data=nuclearplants,
              report=c("adj.means", "adj.mean.diffs", "std.diffs", "z.scores", "chisquare.test", "p.values"))

str(xb1)

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"))
##The following example prints the adjusted means --- labeled as "treatmentvar=0" and "treatmentvar=1" using the fo

```

```

print(xb1,which.vars=c("date","t1"),which.stats=c("pr=0","pr=1","z","p"))
##Now, not asking for the omnibus test
xb2<-xBalance(pr~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
             strata=data.frame(unstrat=factor(character(32)),
                               pt=factor(nuclearplants$pt)),
             data=nuclearplants,
             report=c("adj.means","adj.mean.diffs","std.diffs", "z.scores", "p.values"))

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

```

---

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)

```

### 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"). P-values reported are two-sided for the null-hypothesis of no effect. The option "all" requests all measures.
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 RIttools versions 0.1-9 and before the default imputation

was the mean, in RIttools versions 0.1-11 and henceforth the default is the median). See the example below.

`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.

`covariate.scaling`

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=TRUE`. Currently `median`. To impute means use `mean.default`.

## Details

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 `sd` (standard deviation) in the same variable estimated by pooling treatment and control group `sds` 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 `formula`, 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 `sd`" 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 nonnegative 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 nonnegative weights with stratum codes as names; for an example, do `getFromNamespace("harmonic", "RIttools")`.

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.

### 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

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

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

xb1<-xBalance(pr ~ date, data = testdata, na.rm = TRUE,report="all", impfn=median.default)
xb2<-xBalance(pr ~ date, data = testdata, na.rm = TRUE,report="all", impfn=mean.default)
all.equal(xb1,xb2)

mean.imputed.dat<-RIttools:::naImpute(pr~date, testdata, impfn=mean.default,na.rm=TRUE)
RIttools:::naImpute(pr~date, testdata, impfn=mean.default,na.rm=TRUE) ##naImpute returns a data.frame
mean.default(testdata$date,na.rm=TRUE)

RIttools:::naImpute(pr~date, testdata,na.rm=TRUE)
median.default(testdata$date,na.rm=TRUE)

```

---

xtable.xbal

*An xtable method for xbalance objects*


---

## Description

This function uses the `xtable` package framework to display the results of a call to `xBalance` 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

```

## Arguments

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

## Details

The resulting LaTeX will present one row for each variable in the formula originally passed to `xBalance`, 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 `\newcolumnntype{.}{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 <- xBalance(pr ~ date + t1 + t2 + cap + ne + ct + bw + cum.n,
               strata = data.frame(unstrat = factor(character(32)),
                                   pt = factor(nuclearplants$pt)),
               data = nuclearplants,
               report = c('adj.means', 'adj.mean.diffs', 'std.diffs',
                          'z.scores', 'chisquare.test', 'p.values'))

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 Construction Time",
                      "License to Construction Time", "Net Capacity", "Northeast Region", "Cooling Tower",
```

```
"Babcock-Wilcox Steam", "Cumulative 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")
```

# Index

- \*Topic **datasets**
  - nuclearplants, [2](#)
- \*Topic **design, nonparametric**
  - xBalance, [6](#)
- \*Topic **hplot, print**
  - plot.xbal, [3](#)
- \*Topic **print**
  - print.xbal, [4](#)

mean.default, [7](#)  
median, [7](#)

nuclearplants, [2](#)

plot (plot.xbal), [3](#)  
plot.default, [3](#)  
plot.xbal, [3](#)  
print (print.xbal), [4](#)  
print.xbal, [4](#), [4](#), [10](#)  
print.xtable, [10](#)

xBalance, [4](#), [5](#), [6](#), [9](#), [10](#)  
xtable, [9](#), [10](#)  
xtable.xbal, [9](#)