Package: ILSAstats (via r-universe)

May 8, 2026

Type Package

Title Statistics for International Large-Scale Assessments (ILSA)

Version 0.4.5

Maintainer Andrés Christiansen <andres.christiansen@iea-hamburg.de>

Description Calculates point estimates and standard errors using replicate weights and plausible values for International Large-Scale Assessments (ILSA), including: means, proportions, quantiles, correlations, singlelevel regressions, and multilevel regressions.

Encoding UTF-8

LazyData true

LazyDataCompression xz

BuildResaveData best

RoxygenNote 7.3.3

Suggests wCorr, lme4, MuMIn, WeMix, knitr, rmarkdown

Depends R (>= 3.5.0)

License GPL (>= 3)

VignetteBuilder knitr

URL https://dopatendo.github.io/ILSAstats/

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

Date/Publication 2026-05-08 05:46:13 UTC

RemoteUrl https://github.com/dopatendo/ilsastats

RemoteRef HEAD

RemoteSha 9a87d5c113ef98b7471cf9a954814dd35a54c5a6

Contents

`autoILSA` *Options for automatic functions*

Description

Shows all options for the automatic functions: `leaguetable`, and `proflevels`.

Usage

```
autoILSA(func = c("leaguetable", "proflevels"), study = NULL)
```

Arguments

`func`	a character value indicating the options of which function should be printed.
`study`	an optional character vector indicating the ILSA name.

Value

a list.

Examples

```
autoILSA()
```

center *Centering*

Description

Centers a vector, a matrix or a data frame to the grand mean or the group mean.

Usage

```
center(X, group = NULL, grandmean = NULL, groupmean = NULL, wt = NULL)

grand.mean(x, wt = NULL)

group.mean(x, group, wt = NULL)

getgroup.mean(x, group, wt = NULL)
```

Arguments

X	a matrix or a data frame.
group	a vector indicating the group for centering.
grandmean	a numeric or character vector indicating the number or the the names of columns of X to which grand-mean should be applied.
groupmean	a numeric or character vector indicating the number or the the names of columns of X to which group-mean should be applied.
wt	a numeric vector of weights.
x	a vector, a matrix or a data frame.

Value

a data frame, or a vector.

Examples

```
# Less data for shorter example
repdata2 <- repdata[1:10,c(1:3,6:10,51)]

### One variable ----

# grand-mean
grand.mean(repdata2$item01)
grand.mean(repdata2$item01,wt = repdata2$wt)

# group-mean
group.mean(repdata2$item01,group = repdata2$GROUP)
group.mean(repdata2$item01,group = repdata2$GROUP,wt = repdata2$wt)
```

```
### More than one variable with the same rule ----

# grand-mean
grand.mean(repdata2[,4:8])
grand.mean(repdata2[,4:8],wt = repdata2$wt)

# group-mean
group.mean(repdata2[,4:8],group = repdata2$GROUP)
group.mean(repdata2[,4:8],group = repdata2$GROUP,wt = repdata2$wt)

### More than one variable with different rules ----
center(repdata2, group = repdata2$GROUP, grandmean = 4:5, groupmean = 6:8, wt = repdata2$wt)
center(repdata2, group = repdata2$GROUP, grandmean = 6:8, groupmean = 4:5, wt = repdata2$wt)

center(repdata2, group = repdata2$GROUP, wt = repdata2$wt,
       grandmean = paste0("item0",1:3), groupmean = paste0("item0",4:5))
center(repdata2, group = repdata2$GROUP, wt = repdata2$wt,
       grandmean = paste0("item0",4:5), groupmean = paste0("item0",1:3))
```

`icc` *Intraclass Correlation Coefficient*

Description

Calculates the intraclass correlation coefficient (ICC) fitting a linear mixed-effects model using lmer.

Usage

```
icc(x, PV = FALSE, group, data, weights = NULL, ...)
```

Arguments

`x`	a string vector specifying variable names (within `data`).
`PV`	a logical value indicating if the variables in `x` are plausible values.
`group`	a string specifying the variable name (within `data`) to be used for grouping.
`data`	an optional data frame containing the variables named in `formula`. By default the variables are taken from the environment from which `lmer` is called. While `data` is optional, the package authors *strongly* recommend its use, especially when later applying methods such as `update` and `drop1` to the fitted model (*such methods are not guaranteed to work properly if* `data` *is omitted*). If `data` is omitted, variables will be taken from the environment of `formula` (if specified as a formula) or from the parent frame (if specified as a character vector).
`weights`	an optional vector of 'prior weights' to be used in the fitting process. Should be `NULL` or a numeric vector. Prior `weights` are *not* normalized or standardized in any way. In particular, the diagonal of the residual covariance matrix is the squared residual standard deviation parameter `sigma` times the vector of inverse

weights. Therefore, if the weights have relatively large magnitudes, then in order to compensate, the sigma parameter will also need to have a relatively large magnitude.

... Arguments passed on to lme4::lmer

formula a two-sided linear formula object describing both the fixed-effects and random-effects part of the model, with the response on the left of a ~ operator and the terms, separated by + operators, on the right. Random-effects terms are distinguished by vertical bars (|) separating expressions for design matrices from grouping factors. Two vertical bars (||) can be used to specify multiple uncorrelated random effects for the same grouping variable. (Because of the way it is implemented, the ||-syntax *works only for design matrices containing numeric (continuous) predictors*; to fit models with independent categorical effects, see dummy or the lmer_alt function from the **afex** package.)

REML logical scalar - Should the estimates be chosen to optimize the REML criterion (as opposed to the log-likelihood)?

control a list (of correct class, resulting from lmerControl() or glmerControl() respectively) containing control parameters, including the nonlinear optimizer to be used and parameters to be passed through to the nonlinear optimizer, see the *lmerControl documentation for details.

start a named list of starting values for the parameters in the model. For lmer this can be a numeric vector or a list with one component named "theta".

verbose integer scalar. If > 0 verbose output is generated during the optimization of the parameter estimates. If > 1 verbose output is generated during the individual penalized iteratively reweighted least squares (PIRLS) steps.

subset an optional expression indicating the subset of the rows of data that should be used in the fit. This can be a logical vector, or a numeric vector indicating which observation numbers are to be included, or a character vector of the row names to be included. All observations are included by default.

na.action a function that indicates what should happen when the data contain NAs. The default action (na.omit, inherited from the 'factory fresh' value of getOption("na.action")) strips any observations with any missing values in any variables.

offset this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. One or more offset terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See model.offset.

contrasts an optional list. See the contrasts.arg of model.matrix.default.

devFunOnly logical - return only the deviance evaluation function. Note that because the deviance function operates on variables stored in its environment, it may not return *exactly* the same values on subsequent calls (but the results should always be within machine tolerance).

Value

a numeric value or a list.

Examples

```
# ICC of one variable
icc(x = "Math1",group = "GROUP", weights = repdata$wt, data = repdata)


# ICC of more than one variable
icc(x = c("Math1","Math2","Math3","Math4","Math5","SES"),
    group = "GROUP", weights = repdata$wt, data = repdata)


# ICC of PVs
icc(x = c("Math1","Math2","Math3","Math4","Math5"), PV = TRUE,
    group = "GROUP", weights = repdata$wt, data = repdata)
```

ILSAinfo *ILSA information*

Description

ILSA information

---

leaguetable *ILSA's league tables*

Description

Estimates the mean score for all countries within a cycle of an ILSA. Arguments `method`, `reps`, and `var`, are extracted from `autoILSA` and can be overridden by the user.

Usage

```
leaguetable(
  df,
  study = NULL,
  year,
  subject = NULL,
  specification = NULL,
  addCI = TRUE,
  alpha = 0.05,
  method = NULL,
  reps = NULL,
  fixN = TRUE
)
```

Arguments

df	a data frame.
study	an optional character vector indicating the ILSA name, for a list of available ILSA, check autoILSA. If NULL, the ILSA name will be determined by the column names in the data frame.
year	a numeric vector indicating the ILSA name, for a list of available cycles, check autoILSA.
subject	an optional character vector indicating the subject for a list of available ILSA, check autoILSA.
specification	a character value indicating extra specification like grade (e.g., "G8" for TIMSS) or subject (e.g., "Math" for TIMSSADVANCED).
addCI	a logical value indicating if confidence intervals should be added. Defaults is TRUE.
alpha	a numeric value indicating confidence level.
method	a string indicating the name of the replication method. Available options are: "JK2-full", "JK2-half", "FAY-0.5", and "JK2-half-1PV". Additionally, ILSA names can be used, defaulting into: • "TIMSS", "PIRLS", or "LANA" for "JK2-full"; • "ICILS", "ICCS", or "CIVED" for "JK2-half"; • "PISA" or "TALIS" for "FAY-0.5"; • and "oldTIMSS", "oldPIRLS", or "RLII" for "JK2-half-1PV". Note that "oldTIMSS" and "oldPIRLS" refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.
reps	an integer indicating the number of replications to be created. If NULL the maximum number of zones will be used.
fixN	a logical value indicating if data should be "fixed" to meet official criteria. For example, reducing the sample for certain countries in TIMSS 1995. Default is TRUE.

Value

a data frame.

Examples

```
data(timss99)
leaguetable(df = timss99, year = 1999)
```

lmerPV *Linear Mixed-Models with Plausible Values*

Description

Fits a linear mixed-effects model using lmer and plausible values.

Usage

```
lmerPV(
  formula,
  data = NULL,
  weights = NULL,
  pvs,
  relatedpvs = TRUE,
  grandmean = NULL,
  groupmean = NULL,
  group = NULL,
  nullmodel = FALSE,
  barnardrubin = TRUE,
  ...
)
```

Arguments

formula	a two-sided linear formula object describing both the fixed-effects and random-effects part of the model, with the response on the left of a ~ operator and the terms, separated by + operators, on the right. Random-effects terms are distinguished by vertical bars (|) separating expressions for design matrices from grouping factors. Two vertical bars (||) can be used to specify multiple uncorrelated random effects for the same grouping variable. (Because of the way it is implemented, the ||-syntax *works only for design matrices containing numeric (continuous) predictors*; to fit models with independent categorical effects, see dummy or the lmer_alt function from the **afex** package.)
data	an optional data frame containing the variables named in formula. By default the variables are taken from the environment from which lmer is called. While data is optional, the package authors *strongly* recommend its use, especially when later applying methods such as update and drop1 to the fitted model (*such methods are not guaranteed to work properly if* data *is omitted*). If data is omitted, variables will be taken from the environment of formula (if specified as a formula) or from the parent frame (if specified as a character vector).
weights	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector. Prior weights are *not* normalized or standardized in any way. In particular, the diagonal of the residual covariance matrix is the squared residual standard deviation parameter sigma times the vector of inverse weights. Therefore, if the weights have relatively large magnitudes, then in

order to compensate, the sigma parameter will also need to have a relatively large magnitude.

pvs — a list indicating which variables from formula should be replaced by which plausible values variables. For more details check the examples.

relatedpvs — a logical value indicating if pvs are drawn from the same model, and have the same number of plausible values. If TRUE (default), a total of n estimations will be done, where n is the number of plausible values for each plausible value variable. If FALSE, a total of $n_1 \times n_2 \times n_{.}..$ estimations will be done, where n_i is the number of plausible values in each plausible value variable.

grandmean — a character vector indicating the names of columns of data to which grand-mean should be applied.

groupmean — a character vector indicating the names of columns of data to which group-mean should be applied.

group — a string specifying the variable name (within df) to be used for grouping. Categories in group are treated as independent, e.g., countries.

nullmodel — a logical value indicating if the null model should also be estimated.

barnardrubin — a logical value indicating if Barnard & Rubin adjustment should be used for estimating the degrees of freedom. Default is TRUE.

... — Arguments passed on to lme4::lmer

REML logical scalar - Should the estimates be chosen to optimize the REML criterion (as opposed to the log-likelihood)?

control a list (of correct class, resulting from lmerControl() or glmerControl() respectively) containing control parameters, including the nonlinear optimizer to be used and parameters to be passed through to the nonlinear optimizer, see the *lmerControl documentation for details.

start a named list of starting values for the parameters in the model. For lmer this can be a numeric vector or a list with one component named "theta".

verbose integer scalar. If > 0 verbose output is generated during the optimization of the parameter estimates. If > 1 verbose output is generated during the individual penalized iteratively reweighted least squares (PIRLS) steps.

subset an optional expression indicating the subset of the rows of data that should be used in the fit. This can be a logical vector, or a numeric vector indicating which observation numbers are to be included, or a character vector of the row names to be included. All observations are included by default.

na.action a function that indicates what should happen when the data contain NAs. The default action (na.omit, inherited from the 'factory fresh' value of getOption("na.action")) strips any observations with any missing values in any variables.

offset this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. One or more offset terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See model.offset.

`contrasts` an optional list. See the `contrasts.arg` of `model.matrix.default`.

`devFunOnly` logical - return only the deviance evaluation function. Note that because the deviance function operates on variables stored in its environment, it may not return *exactly* the same values on subsequent calls (but the results should always be within machine tolerance).

Value

a list.

Examples

```
# Null model - with PVs
## Named list, with element names matching formula variables
pvs = list(MATH = paste0("Math",1:5))


m1 <- lmerPV(formula = MATH ~ 1 + (1|GROUP), # Intercept varies across GROUP
      pvs = pvs, # Named list
      data = repdata, # Data frame
      weights = repdata$wt) # Weights vector
m1

## Fixed effects
m1$fixef

## Random effects
m1$ranef

## Models for each PV
summary(m1$models)

# Multiple regression
## Named list, with element names matching formula variables
pvs = list(MATH = paste0("Math",1:5))


m2 <- lmerPV(formula = MATH ~ 1 + GENDER + SES + schoolSES + (1|GROUP),
             pvs = pvs, # Named list
             data = repdata, # Data frame
             weights = repdata$wt) # Weights vector
m2

# Multiple regression with grandmean centering
## Named list, with element names matching formula variables
pvs = list(MATH = paste0("Math",1:5))


m3 <- lmerPV(formula = MATH ~ 1 + GENDER + SES + schoolSES + (1|GROUP),
             pvs = pvs, # Named list
             data = repdata, # Data frame
             weights = repdata$wt,
```

```
            grandmean = c("SES","schoolSES"))
m3

# Multiple regression with groupmean centering
## Named list, with element names matching formula variables
pvs = list(MATH = paste0("Math",1:5))


m4 <- lmerPV(formula = MATH ~ 1 + GENDER + SES + schoolSES + (1|GROUP),
             pvs = pvs, # Named list
             data = repdata, # Data frame
             weights = repdata$wt,
             grandmean = "schoolSES",
             groupmean = "SES",
             group = repdata$GROUP)
m4
```

`prepILSA` *Prepare ILSA Data*

Description

Modifies ILSA data to meet official participation cases, selects columns and transforms data into simple data frames converting missing values to NAs.

Usage

```
prepILSA(
  df,
  study = NULL,
  year = NULL,
  specification = NULL,
  fixN = TRUE,
  columns = NULL
)
```

Arguments

`df`	a data frame.
`study`	an optional character vector indicating the ILSA name, for a list of available ILSA, check `autoILSA`. If `NULL`, the ILSA name will be determined by the column names in the data frame.
`year`	a numeric vector indicating the ILSA name, for a list of available cycles, check `autoILSA`.

specification	a character value indicating extra specification like grade (e.g., "G8" for TIMSS) or subject (e.g., "Math" for TIMSSADVANCED).
fixN	a logical value indicating if data should be "fixed" to meet official criteria. For example, reducing the sample for certain countries in TIMSS 1995. Default is TRUE.
columns	a character vector indicating which columns should be selected. If NULL, all columns will be selected.

Examples

```
data(timss99)

head(timss99)

newdata <- prepILSA(df = timss99, columns = paste0("BSMMAT0",1:5),fixN = FALSE)

head(newdata)
```

---

proflevels *ILSA's proficiency levels*

Description

Estimates the proficiency levels for all countries within a cycle of an ILSA. Arguments method, and reps, are extracted from autoILSA and can be overridden by the user.

Usage

```
proflevels(
  df,
  study = NULL,
  year,
  subject = NULL,
  method = NULL,
  reps = NULL,
  type = c("long", "wide1", "wide2"),
  separateSE = TRUE,
  fixN = TRUE,
  accumulated = FALSE
)
```

Arguments

df	a data frame.
study	an optional character vector indicating the ILSA name, for a list of available ILSA, check autoILSA. If NULL, the ILSA name will be determined by the column names in the data frame.

`year` a numeric vector indicating the ILSA name, for a list of available cycles, check `autoILSA`.

`subject` an optional character vector indicating the subject for a list of available ILSA, check `autoILSA`.

`method` a string indicating the name of the replication method. Available options are: `"JK2-full"`, `"JK2-half"`, `"FAY-0.5"`, and `"JK2-half-1PV"`.

Additionally, ILSA names can be used, defaulting into:

- `"TIMSS"`, `"PIRLS"`, or `"LANA"` for `"JK2-full"`;
- `"ICILS"`, `"ICCS"`, or `"CIVED"` for `"JK2-half"`;
- `"PISA"` or `"TALIS"` for `"FAY-0.5"`;
- and `"oldTIMSS"`, `"oldPIRLS"`, or `"RLII"` for `"JK2-half-1PV"`.

Note that `"oldTIMSS"` and `"oldPIRLS"` refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.

`reps` an integer indicating the number of replications to be created. If `NULL` the maximum number of zones will be used.

`type` a character value indicating the type of table to produce. Options include: `"long"`, for a long table with a column with the proportions and another one for the standard error; `"wide1"` for a wide table where groups are distributed in lines; `"wide2"` for a wide table where groups are distributed in columns.

`separateSE` a logical value indicating if standard errors should be separated from proportions, each as an element from a list. Only works for wide tables. Default is `TRUE`.

`fixN` a logical value indicating if data should be "fixed" to meet official criteria. For example, reducing the sample for certain countries in TIMSS 1995. Default is `TRUE`.

`accumulated` a logical value indicating if proficiency levels should be accumulated.

Value

a data frame or a list.

Examples

```
data(timss99)

proflevels(timss99,year = 1999,type = "long",subject = "math")
```

proflevels.get *Get ILSA's proficiency levels*

Description

Converts ILSA scores into proficiency levels.

Usage

```
proflevels.get(df, study = NULL, subject = NULL, combine = TRUE)
```

Arguments

df	a data frame.
study	an optional character vector indicating the ILSA name, for a list of available ILSA, check autoILSA. If NULL, the ILSA name will be determined by the column names in the data frame.
subject	an optional character vector indicating the subject for a list of available ILSA, check autoILSA.
combine	a logical value indicating if subjects should be combined in a single data frame.

Value

a data frame or a list.

Examples

```
data(timss99)

proflevels.get(timss99, subject = "math")
```

repcreate *Creation of Replicate Weights*

Description

Creates replicate weights given jackknife replicates and jackknife zones.

Usage

```
repcreate(
  df,
  wt,
  jkzone,
  jkrep,
  repwtname = "RWT",
  reps = NULL,
  method,
  index = FALSE
)

repcreateILSA(study, year, df, repwtname = "RWT", index = FALSE)
```

Arguments

df	a data frame.
wt	a string specifying the name of the column (within df) with the total weights.
jkzone	a string specifying the name of the column in df that contains the jackknife zone information.
jkrep	a string specifying the name of the column in df that contains the jackknife replicate information.
repwtname	a string specifying the variable names for the replicate weights.
reps	an integer indicating the number of replications to be created. If NULL the maximum number of zones will be used.
method	a string indicating the name of the replication method. Available options are: "JK2-full", "JK2-half", "FAY-0.5", and "JK2-half-1PV". Additionally, ILSA names can be used, defaulting into: • "TIMSS", "PIRLS", or "LANA" for "JK2-full"; • "ICILS", "ICCS", or "CIVED" for "JK2-half"; • "PISA" or "TALIS" for "FAY-0.5"; • and "oldTIMSS", "oldPIRLS", or "RLII" for "JK2-half-1PV". Note that "oldTIMSS" and "oldPIRLS" refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.
index	a logical value indicating if the result should be just an index of zero and double weights instead of a matrix. Default is FALSE.
study	a string indicating the study name. For checking available studies use ILSAinfo$weights.
year	a numeric value indicating the study year. For checking available years use ILSAinfo$weights.

Value

a data frame or a list.

Examples

```
head(repdata)

# Creation of replicate weights
RW <- repcreate(df = repdata, # the data frame with all the information
                wt = "wt", # the total weights column name
                jkzone = "jkzones", # the jkzones column name
                jkrep = "jkrep", # the jkreps column name
                repwtname = "REPWT", # the desired name for the rep weights
                reps = 50, # the number of replications
                method = "ICILS") # the name of the method aka the study name

head(RW)
```

---

repdata *Simulated data with 5000 cases*

Description

Simulated data with 5000 cases

---

repglm *Generalized Linear Models with Replicate Weights*

Description

Fits a generalized linear model using glm for replicate weights. For a detailed explanation on how the standard errors are estimated see repse.

Usage

```
repglm(
  formula,
  family = stats::gaussian,
  pvs = NULL,
  relatedpvs = TRUE,
  quiet = FALSE,
  summarize = TRUE,
  setup = NULL,
  df,
  wt,
  repwt,
  group = NULL,
  exclude = NULL,
  na.action = getOption("na.action"),
  method
)
```

Arguments

formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
family	a description of the error distribution and link function to be used in the model. For glm this can be a character string naming a family function, a family function or the result of a call to a family function. For glm.fit only the third option is supported. (See family for details of family functions.)
pvs	if plausible values are not used, this should be NULL. Otherwise it is a list indicating which variables from formula should be replaced by which plausible values variables. For more details check the examples.
relatedpvs	a logical value indicating if pvs are drawn from the same model. If TRUE (default), a total of n estimations will be done, where n is the number of plausible values for each plausible value variable. If FALSE, a total of $n_1 \times n_2 \times n_{..}$ estimations will be done, where n_i is the number of plausible values in each plausible value variable.
quiet	a logical value indicating if progress status should be shown while estimating models by group. Default is FALSE.
summarize	a logical value indicating if lm objects should be converted to summary.lm or summary.glm objects and stripped from certain elements to reduce the size of the output object. Default is TRUE.
setup	an optional list produced by repsetup.
df	a data frame.
wt	a string specifying the name of the column (within df) with the total weights.
repwt	a string indicating the common names for the replicate weights columns (within df), or a data frame with the replicate weights.
group	a string specifying the variable name (within df) to be used for grouping. Categories in group are treated as independent, e.g., countries.
exclude	a vector indicating which groups (in the same format as group) should be excluded from the pooled and composite estimates.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
method	a string indicating the name of the replication method. Available options are: "JK2-full", "JK2-half", "FAY-0.5", and "JK2-half-1PV". Additionally, ILSA names can be used, defaulting into: • "TIMSS", "PIRLS", or "LANA" for "JK2-full"; • "ICILS", "ICCS", or "CIVED" for "JK2-half"; • "PISA" or "TALIS" for "FAY-0.5"; • and "oldTIMSS", "oldPIRLS", or "RLII" for "JK2-half-1PV". Note that "oldTIMSS" and "oldPIRLS" refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.

Value

a list with the standard errors and the total weights models.

Examples

```
# Less data for shorter example
repdata2 <- repdata[1:200,]

# Creation of replicate weights
RW <- repcreate(df = repdata2, # the data frame with all the information
                wt = "wt", # the total weights column name
                jkzone = "jkzones", # the jkzones column name
                jkrep = "jkrep", # the jkreps column name
                repwtname = "REPWT", # the desired name for the rep weights
                reps = 50, # the number of replications
                method = "ICILS") # the name of the method aka the study name

### No groups ----

# Simple regression - default family = gaussian
repglm(formula = GENDER ~ 1 + Math1,
        family = gaussian, # Link function
        wt = "wt", # Name of total weight column within df
        repwt = RW, # Data frame of weights
        df = repdata2, # Data frame
        method = "ICILS") # the name of the method aka the study name


# Simple regression - change link function
repglm(formula = GENDER ~ 1 + Math1,
        family = quasibinomial, # Link function
        wt = "wt", # Name of total weight column within df
        repwt = RW, # Data frame of weights
        df = repdata2, # Data frame
        method = "ICILS") # the name of the method aka the study name

# Multiple regression
repglm(formula = GENDER ~ 1 + Math1 + Reading1,
        family = quasibinomial, # Link function
        wt = "wt", # Name of total weight column within df
        repwt = RW, # Data frame of weights
        df = repdata2, # Data frame
        method = "ICILS") # the name of the method aka the study name

# Multiple regression - with PVs
## Named list, with element names matching formula variables
pvs = list(Math = paste0("Math",1:3))
pvs

repglm(formula = GENDER ~ 1 + Math + Reading1, # Math1 now is "Math"
        family = quasibinomial, # Link function
        wt = "wt", # Name of total weight column within df
```

```
          repwt = RW, # Data frame of weights
          df = repdata2, # Data frame
          pvs = pvs, # Named list
          method = "ICILS") # the name of the method aka the study name
```

`replm` *Linear Models with Replicate Weights*

Description

Fits a linear model using lm for replicate weights. For a detailed explanation on how the standard errors are estimated see `repse`.

Usage

```
replm(
  formula,
  pvs = NULL,
  relatedpvs = TRUE,
  quiet = FALSE,
  summarize = TRUE,
  setup = NULL,
  df,
  wt,
  repwt,
  group = NULL,
  exclude = NULL,
  na.action = getOption("na.action"),
  method,
  aggregates = c("pooled", "composite")
)
```

Arguments

`formula`	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted.
`pvs`	if plausible values are not used, this should be `NULL`. Otherwise it is a list indicating which variables from `formula` should be replaced by which plausible values variables. For more details check the examples.
`relatedpvs`	a logical value indicating if `pvs` are drawn from the same model. If `TRUE` (default), a total of n estimations will be done, where n is the number of plausible values for each plausible value variable. If `FALSE`, a total of $n_1 \times n_2 \times n_{..}$ estimations will be done, where n_i is the number of plausible values in each plausible value variable.
`quiet`	a logical value indicating if progress status should be shown while estimating models by group. Default is `FALSE`.

summarize — a logical value indicating if lm objects should be converted to summary.lm or summary.glm objects and stripped from certain elements to reduce the size of the output object. Default is TRUE.

setup — an optional list produced by repsetup.

df — a data frame.

wt — a string specifying the name of the column (within df) with the total weights.

repwt — a string indicating the common names for the replicate weights columns (within df), or a data frame with the replicate weights.

group — a string specifying the variable name (within df) to be used for grouping. Categories in group are treated as independent, e.g., countries.

exclude — a vector indicating which groups (in the same format as group) should be excluded from the pooled and composite estimates.

na.action — a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.

method — a string indicating the name of the replication method. Available options are: "JK2-full", "JK2-half", "FAY-0.5", and "JK2-half-1PV".

Additionally, ILSA names can be used, defaulting into:

- "TIMSS", "PIRLS", or "LANA" for "JK2-full";
- "ICILS", "ICCS", or "CIVED" for "JK2-half";
- "PISA" or "TALIS" for "FAY-0.5";
- and "oldTIMSS", "oldPIRLS", or "RLII" for "JK2-half-1PV".

Note that "oldTIMSS" and "oldPIRLS" refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.

aggregates — a string vector indicating which aggregates should be included, options are "pooled" and "composite", both options can be used at the same time. If NULL no aggregate will be estimated.

Value

a list.

Examples

```
# Less data for shorter example
repdata2 <- repdata[1:1000,]

RW <- repcreate(df = repdata2, # the data frame with all the information
                wt = "wt", # the total weights column name
                jkzone = "jkzones", # the jkzones column name
                jkrep = "jkrep", # the jkreps column name
                repwtname = "REPWT", # the desired name for the rep weights
                reps = 50, # the number of replications
```

```
                  method = "ICILS") # the name of the method aka the study name

### No groups ----

# Simple regression - weights within df
replm(formula = Math1 ~ 1 + GENDER,
      wt = "wt", # Name of total weight column within df
      repwt = "REPWT", # Common names of replicate weights within df
      df = cbind(repdata2,RW), # Data frame
      method = "ICILS") # the name of the method aka the study name

# Simple regression - weights as a separate data frame
replm(formula = Math1 ~ 1 + GENDER,
       wt = "wt", # Name of total weight column within df
       repwt = RW, # Data frame of weights
       df = repdata2, # Data frame
       method = "ICILS") # the name of the method aka the study name

# Multiple regression
replm(formula = Math1 ~ 1 + GENDER + Reading1,
       wt = "wt", # Name of total weight column within df
       repwt = RW, # Data frame of weights
       df = repdata2, # Data frame
       method = "ICILS") # the name of the method aka the study name

# Multiple regression - with PVs
## Named list, with element names matching formula variables
pvs = list(Math = paste0("Math",1:3))
pvs

replm(formula = Math ~ 1 + GENDER + Reading1, # Math1 now is "Math"
       wt = "wt", # Name of total weight column within df
       repwt = RW, # Data frame of weights
       df = repdata2, # Data frame
       pvs = pvs, # Named list
       method = "ICILS") # the name of the method aka the study name

# Multiple regression - with more than one related PV variable
## Named list, with element names matching formula variables
pvs = list(Math = paste0("Math",1:3),
           Reading = paste0("Reading",1:3))
pvs

replm(formula = Math ~ 1 + GENDER + Reading, # Reading1 now is "Reading"
       wt = "wt", # Name of total weight column within df
       repwt = RW, # Data frame of weights
       df = repdata2, # Data frame
       pvs = pvs, # Named list
       method = "ICILS") # the name of the method aka the study name

# Multiple regression - with more than UNrelated PV variables
## Named list, with element names matching formula variables
pvs = list(Math = paste0("Math",1:3),
```

```
            Reading = paste0("Reading",1:3))
pvs

replm(formula = Math ~ 1 + GENDER + Reading, # Reading1 now is "Reading"
       wt = "wt", # Name of total weight column within df
       repwt = RW, # Data frame of weights
       df = repdata2, # Data frame
       pvs = pvs, # Named list
       relatedpvs = FALSE, # Unrelated PVs
       method = "ICILS") # the name of the method aka the study name


### Groups ----

# Simple regression - weights within df
replm(formula = Math1 ~ 1 + GENDER,
      wt = "wt", # Name of total weight column within df
      repwt = "REPWT", # Common names of replicate weights within df
      df = cbind(repdata2,RW), # Data frame
      group = "GROUP",
      method = "ICILS") # the name of the method aka the study name

# Simple regression - weights as a separate data frame
replm(formula = Math1 ~ 1 + GENDER,
      wt = "wt", # Name of total weight column within df
      repwt = RW, # Data frame of weights
      df = repdata2, # Data frame
      group = "GROUP",
      method = "ICILS") # the name of the method aka the study name

# Multiple regression
replm(formula = Math1 ~ 1 + GENDER + Reading1,
      wt = "wt", # Name of total weight column within df
      repwt = RW, # Data frame of weights
      df = repdata2, # Data frame
      group = "GROUP",
      method = "ICILS") # the name of the method aka the study name

# Multiple regression - with PVs
## Named list, with element names matching formula variables
pvs = list(Math = paste0("Math",1:3))
pvs

replm(formula = Math ~ 1 + GENDER + Reading1, # Math1 now is "Math"
      wt = "wt", # Name of total weight column within df
      repwt = RW, # Data frame of weights
      df = repdata2, # Data frame
      pvs = pvs, # Named list
      group = "GROUP",
      method = "ICILS") # the name of the method aka the study name

# Multiple regression - with more than one related PV variable
## Named list, with element names matching formula variables
```

```
pvs = list(Math = paste0("Math",1:3),
           Reading = paste0("Reading",1:3))
pvs

replm(formula = Math ~ 1 + GENDER + Reading, # Reading1 now is "Reading"
      wt = "wt", # Name of total weight column within df
      repwt = RW, # Data frame of weights
      df = repdata2, # Data frame
      pvs = pvs, # Named list
      group = "GROUP",
      method = "ICILS") # the name of the method aka the study name

# Multiple regression - with UNrelated PV variables
## Named list, with element names matching formula variables
pvs = list(Math = paste0("Math",1:3),
           Reading = paste0("Reading",1:3))
pvs

replm(formula = Math ~ 1 + GENDER + Reading, # Reading1 now is "Reading"
      wt = "wt", # Name of total weight column within df
      repwt = RW, # Data frame of weights
      df = repdata2, # Data frame
      pvs = pvs, # Named list
      relatedpvs = FALSE, # Unrelated PVs
      group = "GROUP",
      method = "ICILS") # the name of the method aka the study name
```

---

repmean *Mean, Variance and Standard Deviation with Replicate Weights*

Description

Estimates the mean, variance and standard deviation with replicate weights for a variable or a group of variables and for one or more populations. For a detailed explanation on how the standard errors are estimated see repse.

Usage

```
repmean(
  x,
  PV = FALSE,
  setup = NULL,
  repwt = NULL,
  repindex = NULL,
  wt,
  df,
  method,
  var = c("unbiased", "ML", "none"),
  group = NULL,
```

```
  by = NULL,
  exclude = NULL,
  aggregates = c("pooled", "composite"),
  simplify = TRUE
)
```

Arguments

x	a string vector specifying variable names (within df) for analysis.
PV	a logical value indicating if the variables in x are plausible values.
setup	an optional list produced by repsetup.
repwt	a string indicating the common names for the replicate weights columns (within df), or a data frame with the replicate weights.
repindex	a repweights.index object generated with repcreate(..., index = TRUE). Using this argument instead of repwt will speed up the estimations considerably.
wt	a string specifying the name of the column (within df) with the total weights.
df	a data frame.
method	a string indicating the name of the replication method. Available options are: "JK2-full", "JK2-half", "FAY-0.5", and "JK2-half-1PV". Additionally, ILSA names can be used, defaulting into: • "TIMSS", "PIRLS", or "LANA" for "JK2-full"; • "ICILS", "ICCS", or "CIVED" for "JK2-half"; • "PISA" or "TALIS" for "FAY-0.5"; • and "oldTIMSS", "oldPIRLS", or "RLII" for "JK2-half-1PV". Note that "oldTIMSS" and "oldPIRLS" refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.
var	a string indicating the method to use for the variance: "unbiased" calculates the unbiased estimate (n-1); "ML" calculates the maximum likelihood estimate.
group	a string specifying the variable name (within df) to be used for grouping. Categories in group are treated as independent, e.g., countries.
by	a string specifying a second variable (within df) for grouping. Categories used in by are not considered independent, e.g., gender within a country. If used, the output will be a list with the same length as the unique values of by. This can only be used for analyses with one variable or a group of PVs.
exclude	a vector indicating which groups (in the same format as group) should be excluded from the pooled and composite estimates.
aggregates	a string vector indicating which aggregates should be included, options are "pooled" and "composite", both options can be used at the same time. If NULL no aggregate will be estimated.
simplify	a logical value indicating if only the summary statistics should be printed. If FALSE estimations for all replicated will be provided and no aggregates will be estimated. Default is TRUE. This argument will be ignored if by is used.

Value

a data frame or a list.

Examples

```
# Creation of replicate weights
RW <- repcreate(df = repdata, # the data frame with all the information
                wt = "wt", # the total weights column name
                jkzone = "jkzones", # the jkzones column name
                jkrep = "jkrep", # the jkreps column name
                repwtname = "REPWT", # the desired name for the rep weights
                reps = 50, # the number of replications
                method = "ICILS") # the name of the method aka the study name

### No groups ----

# One variable - weights within df
repmean(x = c("item01"),
        PV = FALSE,
        repwt = "REPWT", wt = "wt", df = cbind(repdata,RW),
        method = "ICILS",var = "ML")

# One variable - weights as a separate data frame
repmean(x = c("item01"),
        PV = FALSE,
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS",var = "ML")

# Multiple variables
repmean(x = c("item01","item02","item03"),
        PV = FALSE,
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS",var = "ML")

# One PV variable
repmean(x = paste0("Math",1:5),
        PV = TRUE, # if set to TRUE, PVs will be treated as separate variables
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS",var = "ML")

### Groups ----

# One variable
repmean(x = c("item01"),
        PV = FALSE,
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS",var = "ML",
        group = "GROUP",
        exclude = "GR2") # GR2 will not be used for Pooled nor Composite

# Multiple variables
repmean(x = c("item01","item02","item03"),
```

```
        PV = FALSE,
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS",var = "ML",
        group = "GROUP",
        exclude = "GR2") # GR2 will not be used for Pooled nor Composite

# One PV variable
repmean(x = paste0("Math",1:5),
        PV = TRUE, # if set to TRUE, PVs will be treated as separate variables
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS",var = "ML",
        group = "GROUP",
        exclude = "GR2") # GR2 will not be used for Pooled nor Composite

### Groups and By ----

# One variable
repmean(x = c("item01"),
        PV = FALSE,
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS",var = "ML",
        group = "GROUP",
        by = "GENDER", # results will be separated by GENDER
        exclude = "GR2") # GR2 will not be used for Pooled nor Composite

# One PV variable
repmean(x = paste0("Math",1:5),
        PV = TRUE, # if set to TRUE, PVs will be treated as separate variables
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS",var = "ML",
        group = "GROUP",
        by = "GENDER", # results will be separated by GENDER
        exclude = "GR2") # GR2 will not be used for Pooled nor Composite
```

`repmeanCI` *Confidence Intervals for Replicated Means*

Description

Calculates the confidence intervals for a `repmean` object.

Usage

```
repmeanCI(x, alpha = 0.05, add = TRUE)
```

Arguments

`x`	an object produced by `repmean`.
`alpha`	a numeric value indicating confidence level.
`add`	a logical value indicating if the confidence intervals should be added to the object or not. Defaults is `TRUE`.

Value

a data frame or a list.

Examples

```
# Creation of replicate weights
RW <- repcreate(df = repdata, # the data frame with all the information
                 wt = "wt", # the total weights column name
                 jkzone = "jkzones", # the jkzones column name
                 jkrep = "jkrep", # the jkreps column name
                 repwtname = "REPWT", # the desired name for the rep weights
                 reps = 50, # the number of replications
                 method = "ICILS") # the name of the method aka the study name


### Groups ----

# One variable
reme <- repmean(x = c("item01"),
                PV = FALSE,
                repwt = RW, wt = "wt", df = repdata,
                method = "ICILS",var = "ML",
                group = "GROUP",
                exclude = "GR2") # GR2 will not be used for Pooled nor Composite

repmeanCI(reme)


# One PV variable
reme <- repmean(x = paste0("Math",1:5),
                PV = TRUE, # if set to TRUE, PVs will be treated as separate variables
                repwt = RW, wt = "wt", df = repdata,
                method = "ICILS",var = "ML",
                group = "GROUP",
                exclude = "GR2") # GR2 will not be used for Pooled nor Composite

repmeanCI(reme)

### Groups and By ----

# One variable
reme <- repmean(x = c("item01"),
                PV = FALSE,
                repwt = RW, wt = "wt", df = repdata,
                method = "ICILS",var = "ML",
                group = "GROUP",
                by = "GENDER", # results will be separated by GENDER
                exclude = "GR2") # GR2 will not be used for Pooled nor Composite

repmeanCI(reme)

# One PV variable
```

```
reme <- repmean(x = paste0("Math",1:5),
                PV = TRUE, # if set to TRUE, PVs will be treated as separate variables
                repwt = RW, wt = "wt", df = repdata,
                method = "ICILS",var = "ML",
                group = "GROUP",
                by = "GENDER", # results will be separated by GENDER
                exclude = "GR2") # GR2 will not be used for Pooled nor Composite

repmeanCI(reme)
```

---

repmeandif *Mean Difference of Independent Samples with Replicate Weights*

Description

Estimates the mean difference for a single variable with replicate weights. For a detailed explanation on how the standard errors are estimated see repse.

Usage

```
repmeandif(x)
```

Arguments

x	a data frame produced by repmean for a single variable or an object produced by leaguetable.

Value

a data frame or a list.

Examples

```
# Creation of replicate weights
RW <- repcreate(df = repdata, # the data frame with all the information
                wt = "wt", # the total weights column name
                jkzone = "jkzones", # the jkzones column name
                jkrep = "jkrep", # the jkreps column name
                repwtname = "REPWT", # the desired name for the rep weights
                reps = 50, # the number of replications
                method = "ICILS") # the name of the method aka the study name


### Groups ----

# One variable
reme <- repmean(x = c("item01"),
                PV = FALSE,
                repwt = RW, wt = "wt", df = repdata,
                method = "ICILS",var = "ML",
```

```
                group = "GROUP",
                exclude = "GR2") # GR2 will not be used for Pooled nor Composite

repmeandif(reme)


# One PV variable
reme <- repmean(x = paste0("Math",1:5),
                PV = TRUE, # if set to TRUE, PVs will be treated as separate variables
                repwt = RW, wt = "wt", df = repdata,
                method = "ICILS",var = "ML",
                group = "GROUP",
                exclude = "GR2") # GR2 will not be used for Pooled nor Composite

repmeandif(reme)

### Groups and By ----

# One variable
reme <- repmean(x = c("item01"),
                PV = FALSE,
                repwt = RW, wt = "wt", df = repdata,
                method = "ICILS",var = "ML",
                group = "GROUP",
                by = "GENDER", # results will be separated by GENDER
                exclude = "GR2") # GR2 will not be used for Pooled nor Composite

repmeandif(reme)

# One PV variable
reme <- repmean(x = paste0("Math",1:5),
                PV = TRUE, # if set to TRUE, PVs will be treated as separate variables
                repwt = RW, wt = "wt", df = repdata,
                method = "ICILS",var = "ML",
                group = "GROUP",
                by = "GENDER", # results will be separated by GENDER
                exclude = "GR2") # GR2 will not be used for Pooled nor Composite

repmeandif(reme)
```

---

repprop *Proportions with Replicate Weights*

Description

Estimates proportions using replicate weights for a variable or a group of plausible values variables and for one or more populations. For a detailed explanation on how the standard errors are estimated see repse.

Usage

```
repprop(
  x,
  categories = NULL,
  setup = NULL,
  repwt = NULL,
  repindex = NULL,
  wt,
  df,
  method,
  group = NULL,
  exclude = NULL,
  aggregates = c("pooled", "composite")
)
```

Arguments

x	a string vector specifying variable names (within df) for analysis.
categories	a vector indicating all possible response categories. If NULL, categories will be derived from the data.
setup	an optional list produced by repsetup.
repwt	a string indicating the common names for the replicate weights columns (within df), or a data frame with the replicate weights.
repindex	a repweights.index object generated with repcreate(..., index = TRUE). Using this argument instead of repwt will speed up the estimations considerably.
wt	a string specifying the name of the column (within df) with the total weights.
df	a data frame.
method	a string indicating the name of the replication method. Available options are: "JK2-full", "JK2-half", "FAY-0.5", and "JK2-half-1PV". Additionally, ILSA names can be used, defaulting into: • "TIMSS", "PIRLS", or "LANA" for "JK2-full"; • "ICILS", "ICCS", or "CIVED" for "JK2-half"; • "PISA" or "TALIS" for "FAY-0.5"; • and "oldTIMSS", "oldPIRLS", or "RLII" for "JK2-half-1PV". Note that "oldTIMSS" and "oldPIRLS" refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.
group	a string specifying the variable name (within df) to be used for grouping. Categories in group are treated as independent, e.g., countries.
exclude	a vector indicating which groups (in the same format as group) should be excluded from the pooled and composite estimates.
aggregates	a string vector indicating which aggregates should be included, options are "pooled" and "composite", both options can be used at the same time. If NULL no aggregate will be estimated.

Value

a list.

Examples

```
# Creation of replicate weights
RW <- repcreate(df = repdata, # the data frame with all the information
                wt = "wt", # the total weights column name
                jkzone = "jkzones", # the jkzones column name
                jkrep = "jkrep", # the jkreps column name
                repwtname = "REPWT", # the desired name for the rep weights
                reps = 50, # the number of replications
                method = "ICILS") # the name of the method aka the study name

### No groups ----

# One variable - weights within df
repprop(x = c("item01"),
        repwt = "REPWT", wt = "wt", df = cbind(repdata,RW),
        method = "ICILS")

# One variable - weights weights as a separate data frame
repprop(x = c("item01"),
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS")

# Multiple variables - PVs are assumed
repprop(x = c("CatMath1","CatMath2","CatMath3"),
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS")

### Groups ----

# One variable - weights within df
repprop(x = c("item01"),
        group = "GROUP",
        repwt = "REPWT", wt = "wt", df = cbind(repdata,RW),
        method = "ICILS")

# One variable - weights weights as a separate data frame
repprop(x = c("item01"),
        group = "GROUP",
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS")

# Multiple variables - PVs are assumed
repprop(x = c("CatMath1","CatMath2","CatMath3"),
        group = "GROUP",
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS")

# Multiple variables - excluding one group
```

```
repprop(x = c("CatMath1","CatMath2","CatMath3"),
        group = "GROUP",
        exclude = "GR2",
        repwt = RW, wt = "wt", df = repdata,
        method = "ICILS")
```

---

repprop.table *Tables for Proportions with Replicate Weights*

Description

Creates tables for proportions using replicate weights for a variable or a group of plausible values variables and for one or more populations.

Usage

```
repprop.table(x, type = c("long", "wide1", "wide2"), separateSE = TRUE)
```

Arguments

x	a list produced by repprop.
type	a character value indicating the type of table to produce. Options include: "long", for a long table with a column with the proportions and another one for the standard error; "wide1" for a wide table where groups are distributed in lines; "wide2" for a wide table where groups are distributed in columns.
separateSE	a logical value indicating if standard errors should be separated from proportions, each as an element from a list. Only works for wide tables. Default is TRUE.

Value

a adata frame or a list.

Examples

```
# Creation of replicate weights
RW <- repcreate(df = repdata, # the data frame with all the information
                wt = "wt", # the total weights column name
                jkzone = "jkzones", # the jkzones column name
                jkrep = "jkrep", # the jkreps column name
                repwtname = "REPWT", # the desired name for the rep weights
                reps = 50, # the number of replications
                method = "ICILS") # the name of the method aka the study name
```

```
x = repprop(x = c("item01"),
            group = "GROUP",
            repwt = "REPWT", wt = "wt", df = cbind(repdata,RW),
            method = "ICILS")

repprop.table(x, type = "long")


repprop.table(x, type = "wide1", separateSE = TRUE)


repprop.table(x, type = "wide1", separateSE = FALSE)


repprop.table(x, type = "wide2", separateSE = TRUE)


repprop.table(x, type = "wide2", separateSE = FALSE)
```

`repquant` *Quantiles with Replicate Weights*

Description

Estimates quantiles with replicate weights for a variable or a group of variables and for one or more populations. For a detailed explanation on how the standard errors are estimated see `repse`.

Usage

```
repquant(
  x,
  qtl = c(0.05, 0.25, 0.75, 0.95),
  setup = NULL,
  repwt,
  wt,
  df,
  method,
  group = NULL,
  by = NULL,
  exclude = NULL
)
```

Arguments

`x`	a string vector specifying variable names (within `df`) for analysis.
`qtl`	a numeric vector indicating the desired quantiles (between 0 and 1).
`setup`	an optional list produced by `repsetup`.

repwt — a string indicating the common names for the replicate weights columns (within df), or a data frame with the replicate weights.

wt — a string specifying the name of the column (within df) with the total weights.

df — a data frame.

method — a string indicating the name of the replication method. Available options are: "JK2-full", "JK2-half", "FAY-0.5", and "JK2-half-1PV".

Additionally, ILSA names can be used, defaulting into:

- "TIMSS", "PIRLS", or "LANA" for "JK2-full";
- "ICILS", "ICCS", or "CIVED" for "JK2-half";
- "PISA" or "TALIS" for "FAY-0.5";
- and "oldTIMSS", "oldPIRLS", or "RLII" for "JK2-half-1PV".

Note that "oldTIMSS" and "oldPIRLS" refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.

group — a string specifying the variable name (within df) to be used for grouping. Categories in group are treated as independent, e.g., countries.

by — a string specifying a second variable (within df) for grouping. Categories used in by are not considered independent, e.g., gender within a country. If used, the output will be a list with the same length as the unique values of by. This can only be used for analyses with one variable or a group of PVs.

exclude — a vector indicating which groups (in the same format as group) should be excluded from the pooled and composite estimates.

Value

a data frame or a list.

Examples

```
RWT <- repcreate(df = repdata, # the data frame with all the information
                 wt = "wt", # the total weights column name
                 jkzone = "jkzones", # the jkzones column name
                 jkrep = "jkrep", # the jkreps column name
                 repwtname = "REPWT", # the desired name for the rep weights
                 reps = 50, # the number of replications
                 method = "ICILS") # the name of the method aka the study name

### No groups ----

# One variable - weights within df
repquant(x = c("item01"),
        qtl = c(0.05, 0.25, 0.75, 0.95),
        repwt = "REPWT", wt = "wt", df = cbind(repdata,RWT),
        method = "ICILS")

# One variable - weights as a separate data frame
repquant(x = c("item01"),
```

```
        qtl = c(0.05, 0.25, 0.75, 0.95),
        repwt = RWT, wt = "wt", df = repdata,
        method = "ICILS")

# One PV variable
repquant(x = paste0("Math",1:5),
        qtl = c(0.05, 0.25, 0.75, 0.95),
        repwt = RWT, wt = "wt", df = repdata,
        method = "ICILS")

### Groups ----

# One variable
repquant(x = c("item01"),
        qtl = c(0.05, 0.25, 0.75, 0.95),
        repwt = RWT, wt = "wt", df = repdata,
        method = "ICILS",
        group = "GROUP",
        exclude = "GR2") # GR2 will not be used for Pooled nor Composite


# One PV variable
repquant(x = paste0("Math",1:5),
        qtl = c(0.05, 0.25, 0.75, 0.95),
        repwt = RWT, wt = "wt", df = repdata,
        method = "ICILS",
        group = "GROUP",
        exclude = "GR2") # GR2 will not be used for Pooled nor Composite

### Groups and By ----

# One variable
repquant(x = c("item01"),
        qtl = c(0.05, 0.25, 0.75, 0.95),
        repwt = RWT, wt = "wt", df = repdata,
        method = "ICILS",
        group = "GROUP",
        by = "GENDER", # results will be separated by GENDER
        exclude = "GR2") # GR2 will not be used for Pooled nor Composite

# One PV variable
repquant(x = paste0("Math",1:5),
        qtl = c(0.05, 0.25, 0.75, 0.95),
        repwt = RWT, wt = "wt", df = repdata,
        method = "ICILS",
        group = "GROUP",
        by = "GENDER", # results will be separated by GENDER
        exclude = "GR2") # GR2 will not be used for Pooled nor Composite
```

---

reprho *Correlations with Replicate Weights*

Description

Estimates correlation coefficients using replicate weights. For a detailed explanation on how the standard errors are estimated see repse.

Usage

```
reprho(
  x = NULL,
  pv = NULL,
  pv2 = NULL,
  relatedpvs = TRUE,
  setup = NULL,
  repwt,
  wt,
  df,
  rho = c("pearson", "spearman", "polychoric"),
  method,
  group = NULL,
  exclude = NULL,
  aggregates = c("pooled", "composite")
)
```

Arguments

x	a string vector specifying variable names (within df) for analysis. If pv is NULL, this function estimates correlations between all variables in the vector. If pv2 is NOT NULL, then x should be set to NULL.
pv	a string vector indicating the variable names for all plausible values of a construct. If not NULL, this function estimates correlations only between x and the plausible values construct.
pv2	a string vector indicating the variable names for all plausible values of a second construct (distinct from pv).
relatedpvs	a logical value indicating if pv and pv2 are drawn from the same model, and have the same number of plausible values. If TRUE (default), a total of n estimations will be done, where n is the number of plausible values of each. If FALSE, a total of $n_1 \times n_2$ estimations will be done, where n_1 is the number of plausible values in pv and n_2 is the number of plausible values in pv2.
setup	an optional list produced by repsetup.
repwt	a string indicating the common names for the replicate weights columns (within df), or a data frame with the replicate weights.
wt	a string specifying the name of the column (within df) with the total weights.
df	a data frame.
rho	a string indicating the correlation coefficient to be computed: "pearson", "polychoric", or "spearman" (lower or uppercase).

`method` a string indicating the name of the replication method. Available options are: `"JK2-full"`, `"JK2-half"`, `"FAY-0.5"`, and `"JK2-half-1PV"`.

Additionally, ILSA names can be used, defaulting into:

- `"TIMSS"`, `"PIRLS"`, or `"LANA"` for `"JK2-full"`;
- `"ICILS"`, `"ICCS"`, or `"CIVED"` for `"JK2-half"`;
- `"PISA"` or `"TALIS"` for `"FAY-0.5"`;
- and `"oldTIMSS"`, `"oldPIRLS"`, or `"RLII"` for `"JK2-half-1PV"`.

Note that `"oldTIMSS"` and `"oldPIRLS"` refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.

`group` a string specifying the variable name (within `df`) to be used for grouping. Categories in `group` are treated as independent, e.g., countries.

`exclude` a vector indicating which groups (in the same format as `group`) should be excluded from the estimation of pooled and composite estimates.

`aggregates` a string vector indicating which aggregates should be included, options are `"pooled"` and `"composite"`, both options can be used at the same time. If `NULL` no aggregate will be estimated.

Value

a data frame.

Examples

```
# Creation of replicate weights
RW <- repcreate(df = repdata, # the data frame with all the information
                wt = "wt", # the total weights column name
                jkzone = "jkzones", # the jkzones column name
                jkrep = "jkrep", # the jkreps column name
                repwtname = "REPWT", # the desired name for the rep weights
                reps = 50, # the number of replications
                method = "ICILS") # the name of the method aka the study name


### No groups ----

# Non PVs
reprho(x = c("GENDER",paste0("Math",1:3)),
       pv = NULL,
       pv2 = NULL,
       rho = "pearson",
       repwt = RW,
       wt = "wt",
       df = repdata,
       method = "ICILS")

# X var and PVs
reprho(x = c("GENDER",paste0("Math",1:3)),
```

```
       pv = paste0("Reading",1:5),
       pv2 = NULL,
       rho = "pearson",
       repwt = RW,
       wt = "wt",
       df = repdata,
       method = "ICILS")

# PVs and PVs (related)
reprho(x = NULL,
       pv = paste0("Math",1:5),
       pv2 = paste0("Reading",1:5),
       rho = "pearson",
       repwt = RW,
       wt = "wt",
       df = repdata,
       method = "ICILS")

# PVs and PVs (UNrelated)
reprho(x = NULL,
       pv = paste0("Math",1:5),
       pv2 = paste0("Reading",1:5),
       relatedpvs = FALSE,
       rho = "pearson",
       repwt = RW,
       wt = "wt",
       df = repdata,
       method = "ICILS")


### Groups ----

# Non PVs
reprho(x = c("GENDER",paste0("Math",1:3)),
       pv = NULL,
       pv2 = NULL,
       rho = "pearson",
       repwt = RW,
       wt = "wt",
       df = repdata,
       group = "GROUP",
       method = "ICILS")

# X var and PVs
reprho(x = c("GENDER",paste0("Math",1:3)),
       pv = paste0("Reading",1:5),
       pv2 = NULL,
       rho = "pearson",
       repwt = RW,
       wt = "wt",
       df = repdata,
       group = "GROUP",
       method = "ICILS")
```

```
# PVs and PVs (related)
reprho(x = NULL,
       pv = paste0("Math",1:5),
       pv2 = paste0("Reading",1:5),
       rho = "pearson",
       repwt = RW,
       wt = "wt",
       df = repdata,
       group = "GROUP",
       method = "ICILS")

# PVs and PVs (UNrelated)
reprho(x = NULL,
       pv = paste0("Math",1:5),
       pv2 = paste0("Reading",1:5),
       relatedpvs = FALSE,
       rho = "pearson",
       repwt = RW,
       wt = "wt",
       df = repdata,
       group = "GROUP",
       method = "ICILS")
```

---

repse *Standard Error for Estimates with Replicate Weights and Plausible Values*

Description

Calculates the standard error given a vector or list of previous estimations.

Usage

```
repse(er, e0, setup = NULL, method)

repsecomp(se)

pvse(PVse, PVe0, df = FALSE, n = NULL, k = NULL, barnardrubin = TRUE)
```

Arguments

er	a vector or a list containing any statistic of interest (e.g., percent, mean, variance, regression coefficient). If it is a vector or list of `length==1`, the function estimates standard errors without plausible values. If it is a list with `length>1`, it estimates standard errors with plausible values.
e0	a numeric vector or a vector containing any statistic of interest (e.g., percent, mean, variance, regression coefficient), computed using total weights. For scenarios without plausible values, `e0` should be a single value. For scenarios with plausible values, `e0` should be a vector of the same length as `er`.

`setup` an optional list produced by `repsetup`.

`method` a string indicating the name of the replication method. Available options are: `"JK2-full"`, `"JK2-half"`, `"FAY-0.5"`, and `"JK2-half-1PV"`.

Additionally, ILSA names can be used, defaulting into:

- `"TIMSS"`, `"PIRLS"`, or `"LANA"` for `"JK2-full"`;
- `"ICILS"`, `"ICCS"`, or `"CIVED"` for `"JK2-half"`;
- `"PISA"` or `"TALIS"` for `"FAY-0.5"`;
- and `"oldTIMSS"`, `"oldPIRLS"`, or `"RLII"` for `"JK2-half-1PV"`.

Note that `"oldTIMSS"` and `"oldPIRLS"` refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.

`se` a numeric vector with standard errors, used by `repsecomp()` to estimate a composite standard error.

`PVse` a numeric vector containing the standard errors of the estimates of each plausible value.

`PVe0` a numeric vector containing the point estimates of each plausible value.

`df` a logical value indicating if degrees should be calculated.

`n` a numeric value indicating the sample size.

`k` a numeric value indicating the number of estimated parameters.

`barnardrubin` a logical value indicating if Barnard & Rubin adjustment should be used for estimating the degrees of freedom. Default is `TRUE`.

Details

The standard errors are calculated using a modifier m, for JK2-full: $m = 0.5$; for JK2-half: $m = 1$; and for FAY-0.5: $\frac{1}{R(1-0.5)^2}$. Depending on the statistic, one of the following formulas is used.

The standard error not involving plausible values is calculated by:

$$\sqrt{m \times \sum_{r=1}^{R} (\varepsilon_r - \varepsilon_0)^2}.$$

The standard error involving plausibles values and replicate weights is calculated by:

$$\sqrt{\left[\sum_{p=1}^{P} \left(m \times \sum_{r=1}^{R} (\varepsilon_{rp} - \varepsilon_{0p})^2\right) \frac{1}{P}\right] + \left[\left(1 + \frac{1}{P}\right) \frac{\sum_{p=1}^{P} (\varepsilon_{0p} - \bar{\varepsilon}_{0p})^2}{P-1}\right]}.$$

The standard error involving plausibles values without replicate weights is calculated by:

$$\sqrt{\frac{\sum_{p=1}^{P} SE_{\varepsilon_{0P}}^2}{P} + \left[\left(1 + \frac{1}{P}\right) \frac{\sum_{p=1}^{P} (\varepsilon_{0p} - \bar{\varepsilon}_{0p})^2}{P-1}\right]}.$$

The standard error of the difference of two statistics (a and b) from independent samples is calculated by:

$$\sqrt{SE_a^2 + SE_b^2}.$$

The standard error of the difference of two statistics (a and b) from dependent samples not involving plausible values is calculated by:

$$\sqrt{m \times \sum_{r=1}^{R}((a_r - b_r) - (a_0 - b_0))^2}.$$

The standard error of the difference of two statistics (a and b) from dependent samples involving plausible values is calculated by:

$$\sqrt{\left[\sum_{p=1}^{P}\left(m \times \sum_{r=1}^{R}((a_{rp} - b_{rp}) - (a_{0p} - b_{0p}))^2\right)\frac{1}{P}\right] + \left[\left(1 + \frac{1}{P}\right)\frac{\sum_{p=1}^{P}\left((a_{0p} - b_{0p}) - (\overline{a}_{0p} - \overline{b}_{0p})\right)^2}{P-1}\right]}.$$

The standard error of a composite estimate is calculated by:

$$\sqrt{\frac{\sum_{c=1}^{C} SE_{\varepsilon_c}^2}{C^2}}.$$

The standard error of the difference between an element (a) of the composite and the composite is calculated by:

$$\sqrt{\frac{\sum_{c=1}^{C} SE_{\varepsilon_c}^2}{C^2} + \left(\frac{(C-1)^2 - 1}{C^2}\right) SE_a^2}.$$

Where ε represents a statistic of interest, the subindex 0 indicates an estimate using the total weights, r indicates a replicate from a total of R, p indicates a plausible value from a total of P, and c indicates an element in a composite estimate from value a total of C.

Value

the standard error.

Examples

```
# Creation of replicate weights
RW <- repcreate(df = repdata, # the data frame with all the information
                wt = "wt", # the total weights column name
                jkzone = "jkzones", # the jkzones column name
                jkrep = "jkrep", # the jkreps column name
                repwtname = "REPWT", # the desired name for the rep weights
                reps = 50, # the number of replications
```

```
                method = "ICILS") # the name of the method aka the study name

# Non-PVs ----

## Mean with total weights
E0 <- stats::weighted.mean(x = repdata$item01, w = repdata$wt, na.rm = TRUE)
E0

## Means by replication
ER <- as.vector(apply(RW,2,function(i){
  stats::weighted.mean(x = repdata$item01, w = i, na.rm = TRUE)
}))
ER

## Standard error by hand
repse(er = ER, e0 = E0, method = "ICILS")

## Standard error with repmean()
repmean(x = "item01",wt = "wt",repwt = RW,df = repdata, method = "ICILS")


# PVs ----

## Mean with total weights
E0 <- sapply(1:5,function(i){
  stats::weighted.mean(x = repdata[,paste0("Math",i)], w = repdata$wt,
                       na.rm = TRUE)
})
E0

## Means by replication
ER <- lapply(1:5, function(j){
  as.vector(apply(RW,2,function(i){
    stats::weighted.mean(x = repdata[,paste0("Math",j)], w = i, na.rm = TRUE)
  }))
})
ER

## Standard error by hand
repse(er = ER, e0 = E0, method = "ICILS")

## Standard error with repmean()
repmean(x = paste0("Math",1:5),wt = "wt",repwt = RW,df = repdata, method = "ICILS",PV = TRUE)
```

`repsetup` *Setup for Analysis with Replicate Weights*

Description

Creates a list with common arguments used for analysis with replicate weights.

Usage

```
repsetup(
  repwt = NULL,
  repindex = NULL,
  wt,
  df,
  method,
  group = NULL,
  exclude = NULL
)

repsetupILSA(
  study,
  year,
  repwt = NULL,
  repindex = NULL,
  df,
  group = NULL,
  exclude = NULL
)
```

Arguments

repwt	a string indicating the common names for the replicate weights columns (within df), or a data frame with the replicate weights.
repindex	a repweights.index object generated with repcreate(..., index = TRUE). Using this argument instead of repwt will speed up the estimations considerably.
wt	a string specifying the name of the column (within df) with the total weights.
df	a data frame.
method	a string indicating the name of the replication method. Available options are: "JK2-full", "JK2-half", "FAY-0.5", and "JK2-half-1PV". Additionally, ILSA names can be used, defaulting into: • "TIMSS", "PIRLS", or "LANA" for "JK2-full"; • "ICILS", "ICCS", or "CIVED" for "JK2-half"; • "PISA" or "TALIS" for "FAY-0.5"; • and "oldTIMSS", "oldPIRLS", or "RLII" for "JK2-half-1PV". Note that "oldTIMSS" and "oldPIRLS" refer to the method used for TIMSS and PIRLS before 2015, where within imputation variance is estimated using only 1 plausible value.
group	a string specifying the variable name (within df) to be used for grouping. Categories in group are treated as independent, e.g., countries.
exclude	a vector indicating which groups (in the same format as group) should be excluded from the pooled and composite estimates.
study	a string indicating the study name. For checking available studies use ILSAinfo$weights.

year — a numeric value indicating the study year. For checking available years use `ILSAinfo$weights`.

Value

a list to be used in other functions.

Examples

```
# Creation of replicate weights
RW <- repcreate(df = repdata, # the data frame with all the information
                wt = "wt", # the total weights column name
                jkzone = "jkzones", # the jkzones column name
                jkrep = "jkrep", # the jkreps column name
                repwtname = "REPWT", # the desired name for the rep weights
                reps = 50, # the number of replications
                method = "ICILS") # the name of the method aka the study name

### No groups ----
stp1 <- repsetup(repwt = RW, wt = "wt", df = repdata, method = "ICILS")
stp1

### Groups ----
stp2 <- repsetup(repwt = RW, wt = "wt", df = repdata, method = "ICILS",
                group = "GROUP", exclude = "GR2")
stp2


### repmean ----

repmean(x = "Math1",setup = stp1)

repmean(x = "Math1",setup = stp2)
```

---

timss99 *Random TIMSS 99 data with 3000 cases and 3 countries*

Description

Random TIMSS 99 data with 3000 cases and 3 countries

`WeMixPV` *Survey Weighted Mixed-Effects Models with Plausible Values*

Description

Fits a linear mixed-effects model using mix and plausible values.

Usage

```
WeMixPV(
  formula,
  data = NULL,
  weights = NULL,
  pvs,
  relatedpvs = TRUE,
  barnardrubin = TRUE,
  ...
)
```

Arguments

`formula`	a formula object in the style of `lme4` that creates the model.
`data`	a data frame containing the raw data for the model.
`weights`	a character vector of names of weight variables found in the data frame starts with units (level 1) and increasing (larger groups).
`pvs`	a list indicating which variables from `formula` should be replaced by which plausible values variables. For more details check the examples.
`relatedpvs`	a logical value indicating if `pvs` are drawn from the same model, and have the same number of plausible values. If `TRUE` (default), a total of n estimations will be done, where n is the number of plausible values for each plausible value variable. If `FALSE`, a total of $n_1 \times n_2 \times n_{\cdot}..$ estimations will be done, where n_i is the number of plausible values in each plausible value variable.
`barnardrubin`	a logical value indicating if Barnard & Rubin adjustment should be used for estimating the degrees of freedom. Default is `TRUE`.
`...`	Arguments passed on to `WeMix::mix` `cWeights` logical, set to `TRUE` to use conditional weights. Otherwise, `mix` expects unconditional weights. `center_group` a list where the name of each element is the name of the aggregation level, and the element is a formula of variable names to be group mean centered; for example to group mean center gender and age within the group student: `list("student"= ~gender+age)`, default value of NULL does not perform any group mean centering. `center_grand` a formula of variable names to be grand mean centered, for example to center the variable education by overall mean of education: `~education`. Default is NULL which does no centering.

max_iteration a optional integer, for non-linear models fit by adaptive quadrature which limits number of iterations allowed before quitting. Defaults to 10. This is used because if the likelihood surface is flat, models may run for a very long time without converging.

nQuad an optional integer number of quadrature points to evaluate models solved by adaptive quadrature. Only non-linear models are evaluated with adaptive quadrature. See notes for additional guidelines.

run logical; TRUE runs the model while FALSE provides partial output for debugging or testing. Only applies to non-linear models evaluated by adaptive quadrature.

verbose logical, default FALSE; set to TRUE to print results of intermediate steps of adaptive quadrature. Only applies to non-linear models.

acc0 deprecated; ignored.

keepAdapting logical, set to TRUE when the adaptive quadrature should adapt after every Newton step. Defaults to FALSE. FALSE should be used for faster (but less accurate) results. Only applies to non-linear models.

start optional numeric vector representing the point at which the model should start optimization; takes the shape of c(coef, vars) from results (see help).

fast logical; deprecated

family the family; optionally used to specify generalized linear mixed models. Currently only binomial() and poisson() are supported.

Value

a list.

Examples

```
# Prepare data weights
repdata2 <- repdata
repdata2$wt1 <- repdata2$wt # weight level 1
repdata2$wt2 <- 1 # weight level 2


# Null model - with PVs
## Named list, with element names matching formula variables
pvs = list(MATH = paste0("Math",1:5))


m1 <- WeMixPV(formula = MATH ~ 1 + (1|GROUP), # Intercept varies across GROUP
              pvs = pvs, # Named list
              data = repdata2, # Data frame
              weights = c("wt1","wt2")) # Weights vector
m1

## Fixed effects
m1$fixef

## Random effects
m1$ranef
```

```
## Models for each PV
summary(m1$models)

# Multiple regression
## Named list, with element names matching formula variables
pvs = list(MATH = paste0("Math",1:5))


m2 <- WeMixPV(formula = MATH ~ 1 + GENDER + SES + schoolSES + (1|GROUP),
              pvs = pvs, # Named list
              data = repdata2, # Data frame
              weights = c("wt1","wt2")) # Weights vector
m2
```

Index