Package 'psychReport'

May 9, 2026

Type Package

Title Reproducible Reports in Psychology

Version 4.0.1

Description
Helper functions for producing reports in Psychology (Reproducible Research). Provides required formatted strings (APA style) for use in 'Knitr'/'Latex' integration within *.Rnw files.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports broom, cli, dplyr (>= 1.0), xtable

Suggests testthat

Depends R (>= 4.0.0)

NeedsCompilation no

Author Ian G Mackenzie [cre, aut],
Carolin Dudschig [aut]

Maintainer Ian G Mackenzie <ian.mackenzie@uni-tuebingen.de>

Repository CRAN

Date/Publication 2026-04-10 08:00:02 UTC

Contents

`psychReport-package` *psychReport*

Description

Helper functions for producing reports in Psychology (Reproducible Research). Provides required formatted strings (APA style) for use in 'Knitr'/'Latex' integration within *.Rnw files.

Author(s)

Maintainer: Ian G Mackenzie <ian.mackenzie@uni-tuebingen.de>

Authors:

- Carolin Dudschig <carolin.dudschig@uni-tuebingen.de>

addDataDF *addDataDF*

Description

Add simulated ex-gaussian reaction-time (RT) data and binary error (Error = 1, Correct = 0) data to an R DataFrame. This function can be used to create simulated data sets.

Usage

```
addDataDF(dat, RT = NULL, Error = NULL)
```

Arguments

dat	DataFrame (see createDF)
RT	RT parameters (see rtDist)
Error	Error parameters (see errDist)

Value

DataFrame with RT (ms) and Error (bool) columns

Examples

```
# Example 1: default dataframe
dat <- createDF()
dat <- addDataDF(dat)
hist(dat$RT, 100)
table(dat$Error)

# Example 2: defined overall RT parameters
dat <- createDF(nVP = 50, nTrl = 50, design = list("Comp" = c("comp", "incomp")))
dat <- addDataDF(dat, RT = c(500, 150, 100))
boxplot(dat$RT ~ dat$Comp)
table(dat$Comp, dat$Error)

# Example 3: defined RT + Error parameters across conditions
dat <- createDF(nVP = 50, nTrl = 50, design = list("Comp" = c("comp", "incomp")))
dat <- addDataDF(dat,
  RT = list(
    "Comp comp" = c(500, 80, 100),
    "Comp incomp" = c(550, 80, 140)
  ),
  Error = list(
    "Comp comp" = 5,
    "Comp incomp" = 10
  )
)
boxplot(dat$RT ~ dat$Comp)
```

```
table(dat$Comp, dat$Error)

# Example 4:
# create dataframe with defined RT + Error parameters across different conditions
dat <- createDF(nVP = 50, nTrl = 50, design = list("Comp" = c("comp", "incomp", "neutral")))
dat <- addDataDF(dat,
  RT = list(
    "Comp comp" = c(500, 150, 100),
    "Comp neutral" = c(550, 150, 100),
    "Comp incomp" = c(600, 150, 100)
  ),
  Error = list(
    "Comp comp" = 5,
    "Comp neutral" = 10,
    "Comp incomp" = 15
  )
)
boxplot(dat$RT ~ dat$Comp)
table(dat$Comp, dat$Error)

# Example 5:
# create dataframe with defined RT + Error parameters across different conditions
dat <- createDF(
  nVP = 50, nTrl = 50,
  design = list(
    "Hand" = c("left_a", "right_a"),
    "Side" = c("left_a", "right_a")
  )
)
dat <- addDataDF(dat,
  RT = list(
    "Hand:Side left_a:left_a" = c(400, 150, 100),
    "Hand:Side left_a:right_a" = c(500, 150, 100),
    "Hand:Side right_a:left_a" = c(500, 150, 100),
    "Hand:Side right_a:right_a" = c(400, 150, 100)
  ),
  Error = list(
    "Hand:Side left_a:left_a" = c(5, 4, 2, 2, 1),
    "Hand:Side left_a:right_a" = c(15, 4, 2, 2, 1),
    "Hand:Side right_a:left_a" = c(15, 7, 4, 2, 1),
    "Hand:Side right_a:right_a" = c(5, 8, 5, 3, 1)
  )
)

boxplot(dat$RT ~ dat$Hand + dat$Side)
table(dat$Error, dat$Hand, dat$Side)
```

---

aovDispMeans *aovDispMeans*

Description

Displays marginal means from model.tables in the command window.

Usage

```
aovDispMeans(aovObj, value = "value", caption = sys.call())
```

Arguments

aovObj	Output from aov
value	String for column name
caption	Required for heading

Value

NULL (prints means to console)

Examples

```
# Example 1:
# create dataframe
dat <- createDF(nVP = 50, nTrl = 1,
                design = list("Comp" = c("comp", "incomp")))

dat <- addDataDF(dat, RT = list("Comp comp"   = c(500, 100, 100),
                                "Comp incomp" = c(520, 100, 100)))

aovRT <- aov(RT ~ Comp + Error(VP/(Comp)), dat)
aovDispMeans(aovRT)
```

---

aovDispTable *aovDispTable*

Description

Display formatted ANOVA table in command window.

Usage

```
aovDispTable(aovObj, caption = sys.call())
```

Arguments

aovObj	Output from aov
caption	Required for heading

Value

NULL (prints table to console)

Examples

```
# Example 1:
# create dataframe
dat <- createDF(nVP = 6, nTrl = 1,
                design = list("Comp" = c("comp", "incomp")))

dat <- addDataDF(dat, RT = list("Comp comp"   = c(500, 150, 100),
                                "Comp incomp" = c(520, 150, 100)))

aovObj <- aov(RT ~ Comp + Error(VP/(Comp)), dat)
aovDispTable(aovObj)
```

---

aovEffectSize *aovEffectSize*

Description

Add partial eta squared (pes) effect size to ANOVA table.

Usage

```
aovEffectSize(aovObj)
```

Arguments

aovObj	Output from aov

Value

list

Examples

```
# Example 1:
# create dataframe with 2(Comp: comp vs. incomp) and 2(Side: left vs. right) factors/levels
dat <- createDF(nVP = 20, nTrl = 1,
                design = list("Comp" = c("comp", "incomp", "neutral"),
                              "Side" = c("left", "right")))

dat <- addDataDF(dat,
                 RT = list("Comp:Side comp:left"     = c(500, 150, 150),
                           "Comp:Side comp:right"    = c(500, 150, 150),
                           "Comp:Side incomp:left"   = c(550, 150, 150),
                           "Comp:Side incomp:right"  = c(550, 150, 150),
```

```
                            "Comp:Side neutral:left"  = c(525, 150, 150),
                            "Comp:Side neutral:right" = c(525, 150, 150)))

aovRT <- aov(RT ~ Comp * Side + Error(VP/(Comp*Side)), dat)
aovDispMeans(aovRT)
aovRT <- aovEffectSize(aovRT)
aovRT <- aovDispTable(aovRT)
```

---

aovJackknifeAdjustment

aovJackknifeAdjustment

Description

Adjust ANOVA table with corrected F (Fc = F/(n-1)^2) and p values for jackkniffed data (see Ulrich and Miller, 2001. Using the jackknife-based scoring method for measuring LRP onset effects in factorial designs. Psychophysiology, 38, 816-827.)

Usage

```
aovJackknifeAdjustment(aovObj, numVPs)
```

Arguments

aovObj	Output from aov
numVPs	The number of participants

Value

list

Examples

```
# Example 1:
# create dataframe with 2(Comp: comp vs. incomp) and 2(Side: left vs. right) factors/levels
dat <- createDF(nVP = 20, nTrl = 1,
                design = list("Comp" = c("comp", "incomp"),
                              "Side" = c("left", "right")))

dat <- addDataDF(dat,
                 RT = list("Comp:Side comp:left"    = c(500, 150, 150),
                           "Comp:Side comp:right"   = c(500, 150, 150),
                           "Comp:Side incomp:left"  = c(500, 150, 150),
                           "Comp:Side incomp:right" = c(500, 150, 150)))

aovRT <- aov(RT ~ Comp*Side + Error(VP/(Comp*Side)), dat)
aovRT <- aovJackknifeAdjustment(aovRT, length(unique(dat$VP)))
aovDispTable(aovRT)
```

aovRoundDigits *aovRoundDigits*

Description

Round digits to n decimal places in ANOVA table

Usage

```
aovRoundDigits(aovObj)
```

Arguments

aovObj	Output from aov

Value

dataframe

Examples

```
# Example 1:
# create dataframe with 2(Comp: comp vs. incomp) and 2(Side: left vs. right) factors/levels
dat <- createDF(nVP = 20, nTrl = 1,
                design = list("Comp" = c("comp", "incomp"),
                              "Side" = c("left", "right")))

dat <- addDataDF(dat,
                 RT = list("Comp:Side comp:left"    = c(500, 150, 150),
                           "Comp:Side comp:right"   = c(500, 150, 150),
                           "Comp:Side incomp:left"  = c(500, 150, 150),
                           "Comp:Side incomp:right" = c(500, 150, 150)))

aovRT <- aov(RT ~ Comp*Side + Error(VP/(Comp*Side)), dat)
aovRT <- aovRoundDigits(aovRT)
aovDispTable(aovRT)
```

aovSphericityAdjustment
aovSphericityAdjustment

Description

Adjust ANOVA table with corrections for sphericity (Greenhouse-Geisser or Huynh-Feldt). Called by default within aovTable. Note: sphericity corrections require an object with a "Sphericity Corrections" component.

Usage

```
aovSphericityAdjustment(aovObj, type = "GG", adjDF = TRUE)
```

Arguments

aovObj	The returned ANOVA object
type	"GG" (Greenhouse-Geisser) or "HF" (Huynh-Feldt)
adjDF	TRUE/FALSE Should DFs be adjusted?

Value

list

Examples

```
# Example 1:
# create dataframe with 3(Comp: neutral vs. comp vs. incomp) factors/levels
dat <- createDF(nVP = 20, nTrl = 1,
                design = list("Comp" = c("neutral", "comp", "incomp")))

dat <- addDataDF(dat,
                 RT = list("Comp neutral" = c(510, 150, 100),
                           "Comp comp"    = c(500, 150, 100),
                           "Comp incomp"  = c(520, 150, 100)))
aovRT <- aov(RT ~ Comp + Error(VP/(Comp)), dat)
aovRT <- aovTable(aovRT)
```

aovTable *aovTable*

Description

Adjust ANOVA table output. Options include calculation of alternative effect sizes (eta squared, partial eta squared), the calculation of marginal means and formatting options for the ANOVA table (e.g., rounding).

Usage

```
aovTable(
  aovObj,
  effectSize = "pes",
  sphericityCorrections = TRUE,
  sphericityCorrectionType = "GG",
  sphericityCorrectionAdjDF = FALSE,
  removeSumSquares = TRUE
)
```

Arguments

aovObj	Output from aov
effectSize	Effect size (only "pes", partial eta squared, is supported)
sphericityCorrections	TRUE/FALSE
sphericityCorrectionType	"GG" (default) vs. "HF"
sphericityCorrectionAdjDF	TRUE/FALSE Should DFs be corrected?
removeSumSquares	TRUE/FALSE Remove SSn/SSd columns from the ANOVA table

Value

list

Examples

```
# Example 1:
# create dataframe with 2(Comp: comp vs. incomp) and 2(Side: left vs. right) factors/levels
dat <- createDF(nVP = 20, nTrl = 1,
                design = list("Comp" = c("comp", "incomp"),
                              "Side" = c("left", "right")))

dat <- addDataDF(dat,
                 RT = list("Comp:Side comp:left"    = c(500, 150, 150),
```

```
                            "Comp:Side comp:right"   = c(500, 150, 150),
                            "Comp:Side incomp:left"  = c(500, 150, 150),
                            "Comp:Side incomp:right" = c(500, 150, 150)))

aovRT <- aov(RT ~ Comp*Side + Error(VP/(Comp*Side)), dat)
aovRT <- aovTable(aovRT)
```

---

aovTidyTable *aovTidyTable*

Description

Take output from base aov function and produce a "tidy" ANOVA table. The output also contains the marginal means.

Usage

```
aovTidyTable(aovObj, data = NULL)
```

Arguments

aovObj	Output from aov function
data	Optional. The original data frame used in the aov call. When provided, sphericity corrections (Mauchly's test, GG/HF epsilon) are computed.

Value

list

Examples

```
# create dataframe
dat <- createDF(nVP = 6, nTrl = 1,
                design = list("Comp" = c("comp", "incomp")))

dat <- addDataDF(dat, RT = list("Comp comp"   = c(500, 150, 100),
                                "Comp incomp" = c(520, 150, 100)))

aovObj <- aov(RT ~ Comp + Error(VP/(Comp)), dat)
aovObj <- aovTable(aovObj)
aovObj$ANOVA
printTable(aovObj$ANOVA)
```

ciStrT *ciStrT*

Description

Returns a string with the 95% CI from a t.test in Latex format.

Usage

```
ciStrT(tObj, numDigits = 0, unit = "")
```

Arguments

tObj	The returned object from a call to t.test
numDigits	The number of digits to round to
unit	"" vs. "ms" vs. "mv" vs. "%"

Value

character

Examples

```
# Example 1:
# create dataframe and add data with 2(Comp: comp vs. incomp) levels
dat <- createDF(nVP = 50,
                nTrl = 1,
                design = list("Comp" = c("comp", "incomp")))

dat <- addDataDF(dat, RT = list("Comp comp"   = c(500, 100, 100),
                                "Comp incomp" = c(600, 100, 100)))

tObj <- t.test(dat$RT[dat$Comp == "incomp"],
               dat$RT[dat$Comp == "comp"],
               paired = TRUE)

ciString <- ciStrT(tObj, unit = "ms")
```

createDF *createDF*

Description

Create dataframe (see also addDataDF)

Usage

```
createDF(
  nVP = 20,
  nTrl = 50,
  design = list(A = c("A1", "A2"), B = c("B1", "B2"))
)
```

Arguments

nVP	Number of participants
nTrl	Number of trials per factor/level for each participant
design	Factors and levels

Value

dataframe

Examples

```
# Example 1
dat <- createDF()

# Example 2
dat <- createDF(nVP = 50, nTrl = 50, design = list("Comp" = c("comp", "incomp")))

# Example 3
dat <- createDF(nVP = 50, nTrl = 50, design = list(
  "Comp" = c("comp", "incomp"),
  "Side" = c("left", "right", "middle")
))
```

effectsizeValueString *effectsizeValueString*

Description

Returns required Latex formatted string for effect size (partial eta squared) = XXX for R/knitr integration. Returns values to 2 sig decimal places.

Usage

```
effectsizeValueString(aovObj, effect, effectSize = "pes")
```

Arguments

aovObj	Output from aov
effect	The effect within the ANOVA table to return
effectSize	Only "pes" (partial eta squared) is supported

Value

character

Examples

```
# Example 1:
# create dataframe and add data with 2(Comp: comp vs. incomp) and 2(Side: left vs. right)
dat <- createDF(nVP = 20, nTrl = 1,
                design = list("Comp" = c("comp", "incomp"),
                              "Side" = c("left", "right")))

dat <- addDataDF(dat, RT = list("Comp:Side comp:left"    = c(500, 150, 100),
                                "Comp:Side comp:right"   = c(500, 150, 100),
                                "Comp:Side incomp:left"  = c(520, 150, 100),
                                "Comp:Side incomp:right" = c(520, 150, 100)))

aovRT <- aov(RT ~ Comp*Side + Error(VP/(Comp*Side)), dat)
aovRT <- aovTable(aovRT)

pesString <- effectsizeValueString(aovRT, "Comp")  # partial eta squared
pesString <- effectsizeValueString(aovRT, "Comp:Side")
```

errDist *errDist*

Description

Returns a random vector of 0's (correct) and 1's (incorrect) with defined proportions (default = 10% errors).

Usage

```
errDist(n = 10000, proportion = 10)
```

Arguments

n	Number
proportion	Approximate proportion of errors in percentage

Value

double

Examples

```
# Example 1: approx 10% errors
x <- errDist(1000)
table(x)

# Example 2: approx 20% errors
x <- errDist(1000, 20)
table(x)
```

fValueString *fValueString*

Description

Returns required Latex formatted string for F(df1, df2) = XXX for R/knitr integration. For example, $F(1, 23) = 3.45$. Returns values to 2 sig decimal places.

Usage

```
fValueString(aovObj, effect)
```

Arguments

aovObj	Output from aov
effect	The effect within the ANOVA table to return

Value

character

Examples

```
# Example 1:
# create dataframe and add data with 2(Comp: comp vs. incomp) and 2(Side: left vs. right)
dat <- createDF(nVP = 20, nTrl = 1,
                design = list("Comp" = c("comp", "incomp"),
                              "Side" = c("left", "right")))

dat <- addDataDF(dat, RT = list("Comp:Side comp:left"    = c(500, 150, 100),
                                "Comp:Side comp:right"   = c(500, 150, 100),
                                "Comp:Side incomp:left"  = c(520, 150, 100),
                                "Comp:Side incomp:right" = c(520, 150, 100)))

aovRT <- aov(RT ~ Comp*Side + Error(VP/(Comp*Side)), dat)
aovRT <- aovTable(aovRT)

fString <- fValueString(aovRT, "Comp")
fString <- fValueString(aovRT, "Comp:Side")
```

---

mathString *mathString*

Description

Returns formatted string following addition/subtraction.

Usage

```
mathString(str1, str2, operation = "-", numDigits = 0, unit = "ms")
```

Arguments

str1	string
str2	string
operation	"+", "-", "*", "/"
numDigits	number 0 (default)
unit	"ms" , "mV" , "mv", or "%"

Value

character

Examples

```
# Example 1:
string <- mathString("550 ms", "480 ms", "-")

# Example 2:
string <- mathString("2.34", "1.65", "+", numDigits = 2, unit = "mV")
```

---

meanStrAov *meanStrAov*

Description

Returns marginal means from ANOVA object for requested effect in Latex format.

Usage

```
meanStrAov(aovObj, effect, level, unit = "ms", numDigits = 0)
```

Arguments

aovObj	Output from aov
effect	Effect to return
level	Level of effect
unit	"ms" vs. "mv" vs. "%"
numDigits	Number of decimal places (default 0)

Value

character

Examples

```
# Example 1:
# create dataframe and add data with 2(Comp: comp vs. incomp) and 2(Side: left vs. right)
dat <- createDF(nVP = 20, nTrl = 1,
                design = list("Comp" = c("comp", "incomp"),
                              "Side" = c("left", "right")))

dat <- addDataDF(dat, RT = list("Comp:Side comp:left"    = c(500, 150, 100),
                                "Comp:Side comp:right"   = c(500, 150, 100),
                                "Comp:Side incomp:left"  = c(520, 150, 100),
                                "Comp:Side incomp:right" = c(520, 150, 100)))

aovRT <- aov(RT ~ Comp*Side + Error(VP/(Comp*Side)), dat)
aovRT <- aovTable(aovRT)

meanString <- meanStrAov(aovRT, "Comp", "comp")
```

```
meanString <- meanStrAov(aovRT, "Comp:Side", "incomp:left")
```

---

meanStrT *meanStrT*

Description

Returns a string with the mean value from a t.test in Latex format.

Usage

```
meanStrT(tObj, numDigits = 0, unit = "")
```

Arguments

tObj	The returned object from a call to t.test
numDigits	The number of digits to round to
unit	"" vs. "ms" vs. "mv" vs. "%"

Value

character

Examples

```
# Example 1:
# create dataframe and add data
dat <- createDF(nVP = 50,
                nTrl = 1,
                design = list("Comp" = c("comp", "incomp")))

dat <- addDataDF(dat, RT = list("Comp comp"   = c(500, 100, 100),
                                "Comp incomp" = c(600, 100, 100)))

tObj <- t.test(dat$RT[dat$Comp == "incomp"],
               dat$RT[dat$Comp == "comp"],
               paired = TRUE)

tString <- meanStrT(tObj, numDigits = 0, unit = "ms")
```

normData *normData*

Description

Normalise within-subjects data by removing between-subjects variability. Each participant's scores are centered on the grand mean.

Usage

```
normData(data, idvar, dvs)
```

Arguments

data	A dataframe
idvar	Column indicating the individual participants
dvs	List of numeric data columns to normalise

Value

dataframe

Examples

```
# Example 1:
library(dplyr)
dat <- createDF(nVP = 50, nTrl = 50, design = list("Comp" = c("comp", "incomp")))
dat <- addDataDF(dat,
  RT = list(
    "Comp comp" = c(500, 80, 100),
    "Comp incomp" = c(550, 80, 140)
  ),
  Error = list(
    "Comp comp" = 5,
    "Comp incomp" = 10
  )
)
datAggVP <- dat %>%
  group_by(VP, Comp) %>%
  summarize(
    N = n(),
    RT = mean(RT[Error == 0]),
    ER = (sum(Error) / N) * 100
  )
datAggVP <- normData(datAggVP, "VP", c("RT", "ER"))
```

numValueString *numValueString*

Description

Returns numerical value with requested unit in Latex format with numDigits number of decimal places and unit symbol.

Usage

```
numValueString(value, numDigits = 2, unit = "")
```

Arguments

value	number
numDigits	number 2 (default)
unit	"ms", "mv", "mV", or "%" or "" (default)

Value

character

Examples

```
# Example 1:
string <- numValueString(100.341, 0, "ms")

# Example 2:
string <- numValueString(2.3412, 2, "mv")

# Example 3:
string <- numValueString(63.9812, 2, "")
```

printAovMeans *printAovMeans*

Description

Returns Latex formatted table of marginal means from model.tables. Uses printTable (xtable) latex package with some basic defaults. For more examples, see R package xtable

Usage

```
printAovMeans(..., caption = "Mean", digits = 3, dv = "ms")
```

Arguments

...	Output from aov
caption	Title for the table
digits	Number of digits to round to
dv	Name of the dependent variable (e.g., "ms", "%")

Value

character

Examples

```
# Example 1:
# create dataframe
dat <- createDF(nVP = 6, nTrl = 1,
                design = list("Comp" = c("comp", "incomp")))

dat <- addDataDF(dat, RT = list("Comp comp"   = c(500, 150, 100),
                                "Comp incomp" = c(520, 150, 100)))

aovRT <- aov(RT ~ Comp + Error(VP/(Comp)), dat)
aovRT <- aovTable(aovRT)
printAovMeans(aovRT, digits = 3, dv = "ms")  # latex formatted
```

---

printTable *printTable*

Description

Returns Latex formatted table from dataframe or ANOVA table. Uses xtable latex package with some basic defaults. For more examples, see R package xtable

Usage

```
printTable(obj, caption = "DF", digits = 3, onlyContents = FALSE)
```

Arguments

obj	Dataframe or ANOVA table to print
caption	Title of the dataframe
digits	Number of digits to round to NB. length can be 1, or vector with length equal to the number of numeric columns
onlyContents	TRUE/FALSE

Value

character

Examples

```
# Example 1:
# create dataframe
dat <- createDF(nVP = 6, nTrl = 1,
                design = list("Comp" = c("comp", "incomp", "neutral")))

dat <- addDataDF(dat, RT = list("Comp comp"    = c(500, 150, 100),
                                "Comp incomp"  = c(520, 150, 100),
                                "Comp neutral" = c(510, 150, 100)))
printTable(dat, digits = c(0, 2)) # latex formatted
printTable(dat, digits = 0)       # latex formatted

aovRT <- aov(RT ~ Comp + Error(VP/(Comp)), dat)
aovRT <- aovTable(aovRT)
printTable(aovRT$ANOVA) # latex formatted
printTable(aovRT$ANOVA, digits = c(0,2,2,2)) # latex formatted
```

---

pValueString *pValueString*

Description

Returns Latex formatted string from a p-value required for R/knitr integration. For example, $p = 0.11$ or $p < 0.01$ Returns values to 3 sig decimal places or < .001

Usage

```
pValueString(pVal)
```

Arguments

pVal	p-value between 0 and 1

Value

character

Examples

```
# Example 1:
pString <- pValueString(0.670)

# Example 2:
pString <- pValueString(0.1234)

# Example 3:
pString <- pValueString("0.03")
```

---

pValueSummary *pValueSummary*

Description

Returns p-values summarized using ***, **, *, or exact value when $p > .05$ (default 2 significant decimal places).

Usage

```
pValueSummary(pVal)
```

Arguments

pVal	vector with p-value between 0 and 1

Value

character

Examples

```
# Examples:
psum <- pValueSummary(0.0067)
psum <- pValueSummary(c(0.0001, 0.002, 0.02, 0.1))
```

requiredPackages *requiredPackages*

Description

Installs (default if required) and loads specified packages.

Usage

```
requiredPackages(
  packages,
  installPackages = FALSE,
  lib = .libPaths()[1],
  repos = "https://cloud.r-project.org"
)
```

Arguments

packages	A list of packages
installPackages	TRUE/FALSE Install package if not installed
lib	character vector giving the library directories where to install the packages. Recycled as needed. If missing, defaults to the first element of .libPaths()
repos	character vector, the base URL(s) of the repositories to use, e.g., the URL of a CRAN mirror such as "https://cloud.r-project.org". For more details on supported URL schemes see url. Can be NULL to install from local files, directories or URLs: this will be inferred by extension from pkgs if of length one.

rtDist *rtDist*

Description

Returns value(s) from a distribution appropriate to simulate reaction times. The distribution is a combined exponential and gaussian distribution called an exponentially modified Gaussian (EMG) distribution or ex-gaussian distribution.

Usage

```
rtDist(n = 10000, gaussMean = 600, gaussSD = 50, expRate = 200)
```

Arguments

n	Number of observations
gaussMean	Mean of the gaussian distribution
gaussSD	SD of the gaussian distribution
expRate	Rate of the exponential function

Value

double

Examples

```
# Example 1:
x <- rtDist()
hist(x, 100)

# Example 2:
x <- rtDist(n = 20000, gaussMean = 800, gaussSD = 50, expRate = 100)
hist(x, 100)
```

---

sphericityValueString *sphericityValueString*

Description

Returns required Latex formatted string for sphericity epsilon values (HF, GG) = XXX for R/knitr integration. Returns values to 2 sig decimal places.

Usage

```
sphericityValueString(aovObj, effect)
```

Arguments

aovObj	The returned ANOVA object
effect	The effect within the ANOVA table to return

Value

character

Examples

```
# Example 1
# create dataframe and add data with 3(Comp: neutral vs. comp vs. incomp) levels
dat <- createDF(nVP = 20, nTrl = 1,
                design = list("Comp" = c("neutral", "comp", "incomp")))

dat <- addDataDF(dat, RT = list("Comp neutral" = c(510, 150, 100),
                                "Comp comp"    = c(500, 150, 100),
                                "Comp incomp"  = c(520, 150, 100)))
aovRT <- aov(RT ~ Comp + Error(VP/(Comp)), dat)
aovRT <- aovTable(aovRT)
sphericityValueString(aovRT, "Comp")
```

`statStrAov` *statStrAov*

Description

Returns Latex formatted string from ANOVA required for R/knitr integration. For example,

$$F(1, 20) = 8.45, p < 0.01, pes = 0.45$$

Returns values to 2 sig decimal places and < 0.01, < 0.001 for p values.

Usage

```
statStrAov(aovObj, effect)
```

Arguments

`aovObj`	Output from aov
`effect`	The effect required from the anova table

Examples

```
# Example 1:
# create dataframe and add data with 2(Comp: comp vs. incomp) and 2(Side: left vs. right)
dat <- createDF(nVP = 20, nTrl = 1,
                design = list("Comp" = c("comp", "incomp"),
                              "Side" = c("left", "right")))

dat <- addDataDF(dat, RT = list("Comp:Side comp:left"    = c(500, 150, 100),
                                "Comp:Side comp:right"   = c(500, 150, 100),
                                "Comp:Side incomp:left"  = c(520, 150, 100),
                                "Comp:Side incomp:right" = c(520, 150, 100)))

aovRT <- aov(RT ~ Comp*Side + Error(VP/(Comp*Side)), dat)
aovRT <- aovTable(aovRT)
```

```
aovString <- statStrAov(aovRT, "Comp")
aovString <- statStrAov(aovRT, "Comp:Side")
```

---

statStrT *statStrT*

Description

Returns required Latex formatted string T-test required for R/Knitr integration. For example, $t(11) = 3.45$, $p < 0.05$. Returns values to 2 sig decimal places and < 0.01, < 0.001 for p values.

Usage

```
statStrT(tObj)
```

Arguments

tObj	The returned object from a call to t.test

Value

character

Examples

```
# Example 1:
# create dataframe and add data with 2(Comp: comp vs. incomp) levels
dat <- createDF(nVP = 50,
                nTrl = 1,
                design = list("Comp" = c("comp", "incomp")))

dat <- addDataDF(dat, RT = list("Comp comp"   = c(500, 100, 100),
                                "Comp incomp" = c(600, 100, 100)))

tObj <- t.test(dat$RT[dat$Comp == "incomp"],
               dat$RT[dat$Comp == "comp"],
               paired = TRUE)

statStrT <- statStrT(tObj)
```

summaryMSDSE *summaryMSDSE*

Description

Aggregate data returning the mean, standard deviation, and standard error

Usage

```
summaryMSDSE(data, factors, dvs, withinCorrection = NULL)
```

Arguments

data	A dataframe
factors	List of factors over which to aggregate
dvs	List of numeric data columns to aggregate
withinCorrection	List of dvs which to apply within-subjects correction to the calculation of the standard deviation and standard error. Within-subject correction calculated according to Morey (2008). NB Data should be normed first (see normData).

Value

dataframe

Examples

```
# Example 1:
library(dplyr)
dat <- createDF(nVP = 50, nTrl = 50, design = list("Comp" = c("comp", "incomp")))
dat <- addDataDF(dat,
  RT = list(
    "Comp comp" = c(500, 80, 100),
    "Comp incomp" = c(550, 80, 140)
  ),
  Error = list(
    "Comp comp" = 5,
    "Comp incomp" = 10
  )
)
datAggVP <- dat %>%
  group_by(VP, Comp) %>%
  summarize(
    N  = n(),
    RT = mean(RT[Error == 0]),
    ER = (sum(Error) / N) * 100
  )
datAgg <- summaryMSDSE(datAggVP, "Comp", c("RT", "ER"))
```

```
# Example 2:
dat <- createDF(nVP = 50, nTrl = 50, design = list("Comp" = c("comp", "incomp")))
dat <- addDataDF(dat,
  RT = list(
    "Comp comp" = c(500, 80, 100),
    "Comp incomp" = c(550, 80, 140)
  ),
  Error = list(
    "Comp comp" = 5,
    "Comp incomp" = 10
  )
)
datAggVP <- dat %>%
  group_by(VP, Comp) %>%
  summarize(
    N = n(),
    RT = mean(RT[Error == 0]),
    ER = (sum(Error) / N) * 100
  )
datAggVP <- normData(datAggVP, "VP", c("RT", "ER"))
datAgg <- summaryMSDSE(
  datAggVP, "Comp", c("RT", "ER", "RT_norm", "ER_norm"),
  c("RT_norm", "ER_norm")
)
```

`tValueString` *tValueString*

Description

Returns required Latex formatted string for *t*(df) = XXX for R/knitr integration. Returns values to 2 sig decimal places.

Usage

```
tValueString(tObj)
```

Arguments

`tObj`	The returned object from a call to t.test

Value

character

Examples

```
# Example 1:
# create dataframe and add data with 2(Comp: comp vs. incomp) levels
dat <- createDF(nVP = 50,
                nTrl = 1,
                design = list("Comp" = c("comp", "incomp")))

dat <- addDataDF(dat, RT = list("Comp comp"   = c(500, 100, 100),
                                "Comp incomp" = c(600, 100, 100)))

tObj <- t.test(dat$RT[dat$Comp == "incomp"],
               dat$RT[dat$Comp == "comp"],
               paired = TRUE)

tString <- tValueString(tObj)
```

Index