Package 'jmvReadWrite'

August 9, 2022

Title Read and Write 'jamovi' Files ('.omv')

Version 0.3.3

Description The free and open a statistical spreadsheet 'jamovi' (<https://www.jamovi.org>) aims to make statistical analyses easy and intuitive. 'jamovi' produces syntax that can directly be used in R (in connection with the R-package 'jmv'). Having import / export routines for the data files 'jamovi' produces ('.omv') permits an easy transfer of analyses between 'jamovi' and R.

License AGPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

VignetteBuilder knitr

Language en-GB

URL https://sjentsch.github.io/jmvReadWrite/

BugReports https://github.com/sjentsch/jmvReadWrite/issues

Depends R (>= 3.5.0)

Imports rjson, zip

Suggests jmv, jmvcore, foreign, haven, knitr, rmarkdown, RProtoBuf, testthat (>= 3.0.0), covr

Config/testthat/edition 3

NeedsCompilation no

Author Sebastian Jentschke [aut, cre] (<https://orcid.org/0000-0003-2576-5432>)

Maintainer Sebastian Jentschke <sebastian.jentschke@uib.no>

Repository CRAN

Date/Publication 2022-08-09 15:30:12 UTC

R topics documented:

`AlbumSales` *Imagine that you worked for a record company and that your boss was interested in predicting album sales from advertising.*

Description

The data set is fictional and was constructed by Andy Field who therefore owns the copyright. The data set is also publicly available on the website that accompanies Andy Field's book, https://edge.sagepub.com/field5e. Without Andy Field's explicit consent, this data set may not be distributed for commercial purposes, this data set may not be edited, and this data set may not be presented without acknowledging its source (i.e., the terms of a CC BY-NC-ND license).

Usage

```
AlbumSales
```

Format

A data.frame with 60 rows, each one representing a different album, and 5 variables

Adverts numericAmount (in thousands of pounds) spent promoting the album before release

Airplay integerHow many times songs from the album were played on a prominent national radio station in the week before release

Image integerHow attractive people found the band's image (out of 10)

Sales integerSales (in thousands) of each album in the week after release

Details

Reference: Field, A. P. (2017). Discovering Statistics Using IBM SPSS Statistics (5th ed.). Sage.

bfi_sample *Twenty-five personality self-report items taken from the International Personality Item Pool*

Description

The data set contains responses from 250 participants filling in twenty-five personality self-report items taken from the International Personality Item Pool (https://ipip.ori.org) as part of the Synthetic Aperture Personality Assessment (SAPA) web-based personality assessment (https://sapa-project.org) project. The 25 items are organized by five putative factors: Agreeableness (A1 to A5), Conscientiousness (C1 to C5), Extraversion (E1 to E5), Neuroticism (N1 to N5), and Openness (N1 to N5). The items were short phrases that the respondent should answer by indicating how accurately the statement describes their typical behaviour or attitude. Responses were collected using a 6-point scale: 1 - Very inaccurate, 2 - Moderately inaccurate, 3 - Slightly inaccurate, 4 - Slightly accurate, 5 - Moderately accurate, 6 - Very accurate.

Usage

```
bfi_sample
```

Format

A data.frame with 254 rows (250 original respondents, 4 manually generated for testing) and 33 variables

ID characterRespondent ID

A1 integerAm indifferent to the feelings of others. (reversed)

A2 integerInquire about others' well-being.

A3 integerKnow how to comfort others.

A4 integerLove children.

A5 integerMake people feel at ease.

C1 integerAm exacting in my work.

C2 integerContinue until everything is perfect.

C3 integerDo things according to a plan.

C4 integerDo things in a half-way manner. (reversed)

C5 integerWaste my time. (reversed)

E1 integerDon't talk a lot. (reversed)

E2 integerFind it difficult to approach others. (reversed)

E3 integerKnow how to captivate people.

E4 integerMake friends easily.

E5 integerTake charge.

N1 integerGet angry easily.

N2 integerGet irritated easily.

N3 integerHave frequent mood swings.

N4 integerOften feel blue.

N5 integerPanic easily.

O1 integerAm full of ideas.

O2 integerAvoid difficult reading material. (reversed)

O3 integerCarry the conversation to a higher level.

O4 integerSpend time reflecting on things.

O5 integerWill not probe deeply into a subject. (reversed)

gender factorGender of the respondent (female, male)

age integerAge of the respondent (years)

AD numericExponent of age (computed: EXP(age))

AF factorRandom data (for testing)

AG factorRandom data (for testing)

age_tr factorAge of the respondent (transformed, as decades: 1 - 10-19, 2 - 20-29, 3 - 30-39, 4 - 40-49, 5 - 50-59, 6 - 60 and over)

ID2 characterRespondent ID (for testing; "A" + random-generated 5-digit-code)

`bfi_sample2` *Twenty-five personality self-report items taken from the International Personality Item Pool (includes jamovi-attributes; should result in a file identical to bfi_sample2.omv under "extdata" when written with write_omv)*

Description

The data set contains responses from 250 participants filling in twenty-five personality self-report items taken from the International Personality Item Pool (https://ipip.ori.org) as part of the Synthetic Aperture Personality Assessment (SAPA) web-based personality assessment (https://sapa-project.org) project. The 25 items are organized by five putative factors: Agreeableness (A1 to A5), Conscientiousness (C1 to C5), Extraversion (E1 to E5), Neuroticism (N1 to N5), and Openness (N1 to N5). The items were short phrases that the respondent should answer by indicating how accurately the statement describes their typical behaviour or attitude. Responses were collected using a 6-point scale: 1 - Very inaccurate, 2 - Moderately inaccurate, 3 - Slightly inaccurate, 4 - Slightly accurate, 5 - Moderately accurate, 6 - Very accurate. The data set includes the jamovi-attributes. It is supposed to result in an identical file compared to the bfi_sample2.omv-file contained in the extdata-directory of the package when written with write_omv.

Usage

```
bfi_sample2
```

Format

A data.frame with 250 rows and 29 variables

ID characterRespondent ID

A1 integerAm indifferent to the feelings of others. (reversed)

A2 integerInquire about others' well-being.

A3 integerKnow how to comfort others.

A4 integerLove children.

A5 integerMake people feel at ease.

C1 integerAm exacting in my work.

C2 integerContinue until everything is perfect.

C3 integerDo things according to a plan.

C4 integerDo things in a half-way manner. (reversed)

C5 integerWaste my time. (reversed)

E1 integerDon't talk a lot. (reversed)

E2 integerFind it difficult to approach others. (reversed)

E3 integerKnow how to captivate people.

E4 integerMake friends easily.

E5 integerTake charge.

N1 integerGet angry easily.

N2 integerGet irritated easily.

N3 integerHave frequent mood swings.

N4 integerOften feel blue.

N5 integerPanic easily.

O1 integerAm full of ideas.

O2 integerAvoid difficult reading material. (reversed)

O3 integerCarry the conversation to a higher level.

O4 integerSpend time reflecting on things.

O5 integerWill not probe deeply into a subject. (reversed)

gender factorGender of the respondent (female, male)

age integerAge of the respondent (years)

ID2 characterRespondent ID (for testing; "A" + random-generated 4-digit-code)

`bfi_sample3` *Twenty-five personality self-report items taken from the International Personality Item Pool (testing file for ordered factors / "Ordinal"-variables in jamovi)*

Description

The data set contains responses from 250 participants filling in twenty-five personality self-report items taken from the International Personality Item Pool (https://ipip.ori.org) as part of the Synthetic Aperture Personality Assessment (SAPA) web-based personality assessment (https://sapa-project.org) project. The 25 items are organized by five putative factors: Agreeableness (A1 to A5), Conscientiousness (C1 to C5), Extraversion (E1 to E5), Neuroticism (N1 to N5), and Openness (N1 to N5). The items were short phrases that the respondent should answer by indicating how accurately the statement describes their typical behaviour or attitude. Responses were collected using a 6-point scale: 1 - Very inaccurate, 2 - Moderately inaccurate, 3 - Slightly inaccurate, 4 - Slightly accurate, 5 - Moderately accurate, 6 - Very accurate. The data set includes the jamovi-attributes. It is supposed to result in an identical file compared to the bfi_sample2.omv-file contained in the extdata-directory of the package when written with write_omv.

Usage

```
bfi_sample3
```

Format

A data.frame with 250 rows and 28 variables

ID characterRespondent ID

A1 ordered factorAm indifferent to the feelings of others. (reversed)

A2 ordered factorInquire about others' well-being.

A3 ordered factorKnow how to comfort others.

A4 ordered factorLove children.

A5 ordered factorMake people feel at ease.

C1 ordered factorAm exacting in my work.

C2 ordered factorContinue until everything is perfect.

C3 ordered factorDo things according to a plan.

C4 ordered factorDo things in a half-way manner. (reversed)

C5 ordered factorWaste my time. (reversed)

E1 ordered factorDon't talk a lot. (reversed)

E2 ordered factorFind it difficult to approach others. (reversed)

E3 ordered factorKnow how to captivate people.

E4 ordered factorMake friends easily.

E5 ordered factorTake charge.

N1 ordered factorGet angry easily.

N2 ordered factorGet irritated easily.

N3 ordered factorHave frequent mood swings.

N4 ordered factorOften feel blue.

N5 ordered factorPanic easily.

O1 ordered factorAm full of ideas.

O2 ordered factorAvoid difficult reading material. (reversed)

O3 ordered factorCarry the conversation to a higher level.

O4 ordered factorSpend time reflecting on things.

O5 ordered factorWill not probe deeply into a subject. (reversed)

gender factorGender of the respondent (Females, Males)

age integerAge of the respondent (years)

`convert_to_omv` *Convert data files (CSV, R, other statistics packages) into .omv-files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org)*

Description

Convert data files (CSV, R, other statistics packages) into .omv-files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org)

Usage

```
convert_to_omv(
  fleInp = "",
  fleOut = "",
  varSrt = c(),
  usePkg = c("foreign", "haven"),
  selSet = "",
  ...
)
```

Arguments

`fleInp`	Name (including the path, if required) of the data file to be read ("FILENAME.ext"; default: ""); supports CSV and R-files natively, or other file types if "foreign" or "haven" are installed
`fleOut`	Name (including the path, if required) of the data file to be written ("FILENAME.omv"; default: ""); if empty, the extension of fleInp is replaced with ".omv"
`varSrt`	Variable(s) that are used to sort the data frame (see Details; if empty, the row order of the input file is kept; default: c())

usePkg	Name of the package: "foreign" or "haven" that shall be used to read SPSS, Stata and SAS files; "foreign" is the default (it comes with base R), but "haven" is newer and more comprehensive
selSet	Name of the data set that is to be selected from the workspace (only applies when reading .RData-files)
...	Additional arguments passed on to methods; see Details below

Details

The ellipsis-parameter (...) can be used to submit arguments / parameters to the functions that are used for reading the data. These are: `read_omv` (for jamovi-files), `read.table` (for CSV / TSV files), `readRDS` (for rds-files), `read_sav` (needs R-package "haven") or `read.spss` (needs R-package "foreign") for SPSS-files, read_dta ("haven") / read.dta ("foreign") for Stata-files, `read_sas` ("haven") for SAS-data-files, and `read_xpt` ("haven") / `read.xport` ("foreign") for SAS-transport-files. For reading CSV / TSV files, similar defaults as `read.csv` (CSV) and `read.delim` (TSV) are used; both are based upon `read.table` but with setting reasonable defaults for the respective file types. If you would like to use "haven", it may be needed to install it manually (i.e., `install.packages("haven", dep = TRUE)`).

Examples

```
## Not run:
library(jmvReadWrite);

# Example 1: Convert from RDS
# (use ToothGrowth as example, save it as RDS)
nmeInp <- paste0(tempfile(), ".rds");
nmeOut <- paste0(tempfile(), ".omv");
saveRDS(jmvReadWrite::ToothGrowth, nmeInp);
convert_to_omv(fleInp = nmeInp, fleOut = nmeOut);
cat(list.files(dirname(nmeOut), basename(nmeOut)));
# -> "file[...].omv" ([...] contains a random combination of numbers / characters
cat(file.info(nmeOut)$size);
# -> 2448 (size may differ on different OSes)
cat(str(read_omv(nmeOut, sveAtt = FALSE)));
# gives a overview of the dataframe (all columns and some attributes,
# sveAtt is intentionally set to FALSE to make the output not too overwhelming)
unlink(nmeInp);
unlink(nmeOut);

# Example 2: Convert from CSV
# (use ToothGrowth again as example, this time save it as CSV)
nmeInp <- paste0(tempfile(), ".csv");
nmeOut <- paste0(tempfile(), ".omv");
write.csv(jmvReadWrite::ToothGrowth, nmeInp);
convert_to_omv(fleInp = nmeInp, fleOut = nmeOut);
cat(list.files(dirname(nmeOut), basename(nmeOut)));
cat(file.info(nmeOut)$size);
# -> 2104 (size may differ acc. to OS; the size is smaller than for the RDS-file
# because CSV can store fewer attributes, e.g., labels)
cat(str(read_omv(nmeOut, sveAtt = FALSE)));
```

```
# gives a overview of the dataframe (all columns and some attributes)
unlink(nmeInp);
unlink(nmeOut);

## End(Not run)
```

`long2wide_omv` *Converts .omv-files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org) from long to wide format*

Description

Converts .omv-files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org) from long to wide format

Usage

```
long2wide_omv(
  fleInp = "",
  fleOut = "",
  varID = "ID",
  varTme = c(),
  varExc = c(),
  varTgt = c(),
  varSep = "_",
  varOrd = c("times", "vars"),
  varSrt = c(),
  usePkg = c("foreign", "haven"),
  selSet = "",
  ...
)
```

Arguments

`fleInp`	Name (including the path, if required) of the data file to be read (e.g., "FILE_IN.omv"; default: ""); can be any supported file type, see Details below
`fleOut`	Name (including the path, if required) of the data file to be written (e.g., "FILE_OUT.omv"; default: ""); if empty, FILE_IN from fleInp is extended with "_wide(file extension -> .omv)"
`varID`	Names of one or more variables that identify the same group / individual (default: c())
`varTme`	Name of the variable(s) that differentiates multiple records from the same group / individual (default: c())
`varExc`	Name of the variable(s) should be excluded from the transformation, typically this will be between-subject-variable(s) (default: c())

varTgt	Names of one or more variables to be transformed / reshaped (other variables are excluded, if empty(c()) all variables except varTme, varID and varExc are included; default: c())
varSep	Separator character when concatenating the fixed and time-varying part of the variable name ("VAR1_1", "VAR1_2"; default: "_")
varOrd	How variables / columns are organized: for "times" (default) the steps of the time varying variable are adjacent, for "vars" the steps of the original columns in the long dataset
varSrt	Variable(s) that are used to sort the data frame (see Details; if empty, the order returned from reshape is kept; default: c())
usePkg	Name of the package: "foreign" or "haven" that shall be used to read SPSS, Stata and SAS files; "foreign" is the default (it comes with base R), but "haven" is newer and more comprehensive
selSet	Name of the data set that is to be selected from the workspace (only applies when reading .RData-files)
...	Additional arguments passed on to methods; see Details below

Details

The ellipsis-parameter (...) can be used to submit arguments / parameters to the functions that are used for transforming or reading the data. The transformation uses `reshape`. When reading the data, the functions are: `read_omv` (for jamovi-files), `read.table` (for CSV / TSV files; using similar defaults as `read.csv` for CSV and `read.delim` for TSV which both are based upon `read.table` but with adjusted defaults for the respective file types), `readRDS` (for rds-files), `read_sav` (needs R-package "haven") or `read.spss` (needs R-package "foreign") for SPSS-files, `read_dta` ("haven") / `read.dta` ("foreign") for Stata-files, `read_sas` ("haven") for SAS-data-files, and `read_xpt` ("haven") / `read.xport` ("foreign") for SAS-transport-files. If you would like to use "haven", it may be needed to install it manually (i.e., `install.packages("haven", dep = TRUE)`).

Examples

```
## Not run:
library(jmvReadWrite)
# generate a test dataframe with 100 (imaginary) participants / units of
#  observation (ID), 8 measurement (measure) of one variable (X)
dtaInp <- data.frame(ID = rep(as.character(seq(1, 100)), each = 8),
                     measure = rep(seq(1, 8), times = 100),
                     X = runif(800, -10, 10))
cat(str(dtaInp))
# the output should look like this
# 'data.frame': 800 obs. of  3 variables:
#  $ ID     : chr  "1" "1" "1" "1" ...
#  $ measure: int  1 2 3 4 5 6 7 8 1 2 ...
#  $ X      : num  ...
# this data set is stored as (temporary) RDS-file and later processed by long2wide
nmeInp <- paste0(tempfile(), ".rds")
nmeOut <- paste0(tempfile(), ".omv")
saveRDS(dtaInp, nmeInp)
long2wide_omv(fleInp = nmeInp, fleOut = nmeOut, varID = "ID", varTme = "measure", varTgt = "X")
```

```
# it is required to give at least the arguments fleInp, varID and varTme
# check whether the file was created and its size
cat(list.files(dirname(nmeOut), basename(nmeOut)))
# -> "file[...].omv" ([...] contains a random combination of numbers / characters
cat(file.info(nmeOut)$size)
# -> 6851 (approximate size; size may differ in every run [in dependence of
#          how well the generated random data can be compressed])
cat(str(read_omv(nmeOut, sveAtt = FALSE)))
# the data set is now transformed into wide (and each the measurements is now
# indicated as a suffix to X; X_1, X_2, ...)
# 'data.frame': 100 obs. of  9 variables:
#  $ ID : chr  "1" "2" "3" "4" "5" "6" "7" "8" "9" "10" ...
#   ..- attr(*, "jmv-id")= logi TRUE
#   ..- attr(*, "missingValues")= list()
#  $ X_1: num  ...
#   ..- attr(*, "missingValues")= list()
#  $ X_2: num  ...
#   ..- attr(*, "missingValues")= list()
#  $ X_3: num  ...
#   ..- attr(*, "missingValues")= list()
#  $ X_4: num  ...
#   ..- attr(*, "missingValues")= list()
#  $ X_5: num  ...
#   ..- attr(*, "missingValues")= list()
#  $ X_6: num  ...
#   ..- attr(*, "missingValues")= list()
#  $ X_7: num  ...
#   ..- attr(*, "missingValues")= list()
#  $ X_8: num  ...
#   ..- attr(*, "missingValues")= list()

unlink(nmeInp)
unlink(nmeOut)

## End(Not run)
```

`merge_cols_omv` *Merges two or more data files by adding the content of other input files as columns to the first input file and outputs them as files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org)*

Description

Merges two or more data files by adding the content of other input files as columns to the first input file and outputs them as files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org)

Usage

```
merge_cols_omv(
```

```
  fleInp = c(),
  fleOut = "",
  typMrg = c("outer", "inner", "left", "right"),
  varBy = list(),
  varSrt = c(),
  usePkg = c("foreign", "haven"),
  selSet = "",
  ...
)
```

Arguments

fleInp	vector with the file names of the input files (including the path, if required; c("FILE_IN1.omv", "FILE_IN2.omv"); default: c()); can be any supported file type, see Details below
fleOut	Name of the data file to be written (including the path, if required; "FILE_OUT.omv"; default: ""); if empty, the data frame with the added columns is returned as variable (but not written)
typMrg	Type of merging operation: "outer" (default), "inner", "left" or "right"; see Details below
varBy	Name of the variable by which the data sets are matched, can either be a string, a character or a list (see Details below; default: list())
varSrt	Variable(s) that are used to sort the data frame (see Details; if empty, the order after merging is kept; default: c())
usePkg	Name of the package: "foreign" or "haven" that shall be used to read SPSS, Stata and SAS files; "foreign" is the default (it comes with base R), but "haven" is newer and more comprehensive
selSet	Name of the data set that is to be selected from the workspace (only applies when reading .RData-files)
...	Additional arguments passed on to methods; see Details below

Details

There are four different types of merging operations: "outer" keeps all cases (but columns in the resulting data set may be empty if they did not contain values in same input data sets), "inner" keeps only those cases where all datasets contain the same value in the matching variable, for "left" all cases from the first data set in fleInp are kept (whereas cases that are only contained in input data set two or higher are dropped), for "right" all cases from the second (or any higher) data set in fleInp are kept. The behaviour of "left" and "right" may be somewhat difficult to predict in case of merging several data sets, therefore "outer" might be a safer choice if several data sets are merged. The variable that is used for matching (varBy) can either be a string (if all datasets contain a matching variable with the same name), a character vector (containing several matching variables that are the same for all data sets) or a list with the same length as fleInp. In the latter case, each cell of that list can again contain either a string (one matching variable for each data set in fleInp) or a character vector (several matching variables for each data set in fleInp; NB: all character vectors in the cells of the list must have the same length as it is necessary to always use the same number of matching variables when merging). The ellipsis-parameter (...) can be used to submit arguments / parameters

to the functions that are used for merging or reading the data. Adding columns uses `merge`. When reading the data, the functions are: `read_omv` (for jamovi-files), `read.table` (for CSV / TSV files; using similar defaults as `read.csv` for CSV and `read.delim` for TSV which both are based upon `read.table` but with adjusted defaults for the respective file types), `readRDS` (for rds-files), `read_sav` (needs R-package "haven") or `read.spss` (needs R-package "foreign") for SPSS-files, `read_dta` ("haven") / `read.dta` ("foreign") for Stata-files, `read_sas` ("haven") for SAS-data-files, and `read_xpt` ("haven") / `read.xport` ("foreign") for SAS-transport-files. If you would like to use "haven", it may be needed to install it manually (i.e., `install.packages("haven", dep = TRUE)`).

Value

a data frame (if fleOut is empty) with where the columns of all input data sets (in the files given to fleInp) are concatenated

Examples

```
## Not run:
library(jmvReadWrite);
dtaInp <- bfi_sample2;
nmeInp <- paste0(tempfile(), "_", 1:3, ".rds");
nmeOut <- paste0(tempfile(), ".omv");
for (i in seq_along(nmeInp)) {
   saveRDS(stats::setNames(dtaInp, c("ID", paste0(names(dtaInp)[-1], "_", i))), nmeInp[i]);
}
# save dtaInp three times (i.e., the length of nmeInp), adding "_" + 1 ... 3 as index
# to the data variables (A1 ... O5, gender, age → A1_1, ...)
merge_cols_omv(fleInp = nmeInp, fleOut = nmeOut, varBy = "ID");
cat(file.info(nmeOut)$size);
# -> 17731 (size may differ on different OSes)
dtaOut <- read_omv(nmeOut, sveAtt = FALSE);
# read the data set where the three original datasets were added as columns and show
# the variable names
cat(names(dtaOut));
cat(names(dtaInp));
# compared to the input data set, we have the same names (expect for "ID" which was
# used for matching and that each variable had added an indicator from which data
# set they came)
cat(dim(dtaInp), dim(dtaOut));
# the first dimension of the data sets (rows) stayed the same (250), whereas the
# second dimension is now approx. three times as large (28 -> 82):
# 28 - 1 (for "ID") = 27 * 3 + 1 (for "ID") = 82
cat(colMeans(dtaInp[2:11]));
cat(colMeans(dtaOut[2:11]));
# it's therefore not much surprise that the values of the column means for the first
# 10 variables of dtaInp and dtaOut are the same too

unlink(nmeInp);
unlink(nmeOut);

## End(Not run)
```

merge_rows_omv *Merges two .omv-files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org) by adding the content of the second, etc. file(s) as rows to the first file*

Description

Merges two .omv-files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org) by adding the content of the second, etc. file(s) as rows to the first file

Usage

```
merge_rows_omv(
  fleInp = c(),
  fleOut = "",
  typMrg = c("all", "common"),
  colInd = FALSE,
  rstRwN = TRUE,
  rmvDpl = FALSE,
  varSrt = c(),
  usePkg = c("foreign", "haven"),
  selSet = "",
  ...
)
```

Arguments

fleInp	Vector with file names (including the path, if required) of the data files to be read (c("FILE1.omv", "FILE2.omv"); default: c()); can be any supported file type, see Details below
fleOut	Name of the data file to be written (including the path, if required; "FILE_OUT.omv"; default: ""); if empty, the data frame with the added columns is returned as variable (but not written)
typMrg	Type of merging operation: "all" (default) or "common"; see also Details
colInd	Add a column with an indicator (the basename of the file minus the extension) marking from which input data set the respective rows are coming (default: FALSE)
rstRwN	Reset row names (i.e., do not keep the row names of the original input data sets but number them consecutively - one to the row number of all input data sets added up; default: TRUE)
rmvDpl	Remove duplicated rows (i.e., rows with the same content as a previous row in all columns; default: FALSE)
varSrt	Variable(s) that are used to sort the data frame (see Details; if empty, the order after merging is kept; default: c())

usePkg	Name of the package: "foreign" or "haven" that shall be used to read SPSS, Stata and SAS files; "foreign" is the default (it comes with base R), but "haven" is newer and more comprehensive
selSet	Name of the data set that is to be selected from the workspace (only applies when reading .RData-files)
...	Additional arguments passed on to methods; see Details below

Details

The different types of merging operations: "all" keeps all existing variables / columns that are contained in any of the input data sets and fills them up with NA where the variable / column doesn't exist in a input data set. "common" only keeps the variables / columns that are common to all input data sets (i.e., that are contained in all data sets). The ellipsis-parameter can be used to submit arguments / parameters to the functions that are used for merging or reading the data. The merging operation uses `rbind`. When reading the data, the functions are: `read_omv` (for jamovi-files), `read.table` (for CSV / TSV files; using similar defaults as `read.csv` for CSV and `read.delim` for TSV which both are based upon `read.table` but with adjusted defaults for the respective file types), `readRDS` (for rds-files), `read_sav` (needs R-package "haven") or `read.spss` (needs R-package "foreign") for SPSS-files, read_dta ("haven") / read.dta ("foreign") for Stata-files, read_sas ("haven") for SAS-data-files, and read_xpt ("haven") / read.xport ("foreign") for SAS-transport-files. If you would like to use "haven", it may be needed to install it manually (i.e., `install.packages("haven", dep = TRUE)`).

Value

a data frame (if fleOut is empty) with where the rows of all input data sets (i.e., the files given in the fleInp-argument) are concatenated

Examples

```
## Not run:
library(jmvReadWrite);
dtaInp <- bfi_sample2;
nmeInp <- paste0(tempfile(), "_", 1:3, ".rds");
nmeOut <- paste0(tempfile(), ".omv");
for (i in seq_along(nmeInp)) saveRDS(dtaInp[-i - 1], nmeInp[i]);
# save dtaInp three times (i.e., the length of nmeInp), removing one data columns in
# each data set (for demonstration purposes, A1 in the first, A2 in the second, ...)
merge_rows_omv(fleInp = nmeInp, fleOut = nmeOut, colInd = TRUE);
cat(file.info(nmeOut)$size);
# -> 10767 (size may differ on different OSes)
dtaOut <- read_omv(nmeOut, sveAtt = FALSE);
# read the data set where the three original datasets were added as rows and show
# the variable names
cat(names(dtaInp));
cat(names(dtaOut));
# compared to the input data set, we have the same variable names; fleInd (switched
# on by colInd = TRUE and showing from which data set the rows are coming from) is
# new and A1 is moved to the end of the list (the "original" order of variables may
# not always be preserved and columns missing from at least one of the input data
```

```
# sets may be added at the end)
cat(dim(dtaInp), dim(dtaOut));
# the first dimension of the data sets (rows) is now three times of that of the input
# data set (250 -> 750), the second dimension (columns / variables) is increased by 1
# (for "fleInd")

merge_rows_omv(fleInp = nmeInp, fleOut = nmeOut, typMrg = "common");
# the argument typMrg = "common" removes the columns that are not present in all of
# the input data sets (i.e., A1, A2, A3)
dtaOut <- read_omv(nmeOut, sveAtt = FALSE);
# read the data set where the three original datasets were added as rows and show
# the variable names
cat(names(dtaInp));
cat(names(dtaOut));
# compared to the input data set, the variables that were missing in at least one
# data set (i.e., "A1", "A2" and "A3") are removed
cat(dim(dtaInp), dim(dtaOut));
# the first dimension of the data sets (rows) is now three times of that of the
# input data set (250 -> 750), the second dimension (columns / variables) is
# reduced by 3 (i.e., "A1", "A2", "A3")

unlink(nmeInp);
unlink(nmeOut);

## End(Not run)
```

---

read_omv *Read files created of the statistical spreadsheet 'jamovi' (https://www.jamovi.org)*

Description

Read files created of the statistical spreadsheet 'jamovi' (https://www.jamovi.org)

Usage

```
read_omv(
  fleInp = "",
  useFlt = FALSE,
  rmMsVl = FALSE,
  sveAtt = TRUE,
  getSyn = FALSE,
  getHTM = FALSE
)
```

Arguments

fleInp	Name (including the path, if required) of the 'jamovi'-file to be read ("FILENAME.omv"; default: "")

useFlt	Apply filters (remove the lines where the filter is set to 0; default: FALSE)?
rmMsVl	Remove values defined as missing values (replace them with NA; default: FALSE)?
sveAtt	Store attributes that are not required in the data set (if you want to write the same data set using write_omv; default: FALSE)?
getSyn	Extract syntax from the analyses in the 'jamovi'-file and store it in the attribute "syntax" (default: FALSE)?
getHTM	Store index.html in the attribute "HTML" (default: FALSE)?

Value

data frame (can be directly used with functions included in the R-package 'jmv' and syntax from 'jamovi'; also compatible with the format of the R-package "foreign")

Examples

```
## Not run:
library(jmvReadWrite);
fleOMV <- system.file("extdata", "ToothGrowth.omv", package = "jmvReadWrite");
data <- read_omv(fleInp = fleOMV, getSyn = TRUE);
# if the syntax couldn't be extracted, an empty list - length = 0 - is returned,
# otherwise, the commands are shown and the first analysis is run, with the output
# from the second analysis being assigned to the variable result
if (length(attr(data, "syntax")) >= 1) {
    print(attr(data, "syntax"));
    # the print-function is only used to force devtools::run_examples() to show output
    eval(parse(text=paste0("result = ", attr(data, "syntax")[[1]])));
    # without assigning the output to a variable, the command would be:
    # eval(parse(text=attr(data, "syntax")[[1]]))
    print(names(result));
    print(result$main);
    # -> "main"      "assump"    "contrasts" "postHoc"   "emm"       "residsOV"
    # (the names of the six output tables)
}

## End(Not run)
```

---

sort_omv — *Sort data (using one or more variables) in .omv-files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org)*

Description

Sort data (using one or more variables) in .omv-files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org)

Usage

```
sort_omv(
  fleInp = c(),
  fleOut = "",
  varSrt = c(),
  usePkg = c("foreign", "haven"),
  selSet = "",
  ...
)
```

Arguments

fleInp	Name (including the path, if required) of the data file to be read ("FILENAME.ext"; default: ""); can be any supported file type, see Details below
fleOut	Name (including the path, if required) of the data file to be written ("FILENAME.omv"; default: ""); if empty, the extension of fleInp is replaced with "_sorted(file extension -> .omv)"
varSrt	Variable(s) that are used to sort the data frame (see Details; default: c())
usePkg	Name of the package: "foreign" or "haven" that shall be used to read SPSS, Stata and SAS files; "foreign" is the default (it comes with base R), but "haven" is newer and more comprehensive
selSet	Name of the data set that is to be selected from the workspace (only applies when reading .RData-files)
...	Additional arguments passed on to methods; see Details below

Details

varSrt can be either a character or a character vector (with one or more variables respectively). The sorting order for a particular variable can be inverted with preceding the variable name with "-". Please note that this doesn't make sense and hence throws a warning for certain variable types (e.g., factors). The ellipsis-parameter can be used to submit arguments / parameters to the functions that are used for reading the data. These are: `read_omv` (for jamovi-files), `read.table` (for CSV / TSV files; using similar defaults as `read.csv` for CSV and `read.delim` for TSV which both are based upon `read.table` but with adjusted defaults for the respective file types), `readRDS` (for rds-files), `read_sav` (needs R-package "haven") or `read.spss` (needs R-package "foreign") for SPSS-files, `read_dta` ("haven") / `read.dta` ("foreign") for Stata-files, `read_sas` ("haven") for SAS-data-files, and `read_xpt` ("haven") / `read.xport` ("foreign") for SAS-transport-files. If you would like to use "haven", it may be needed to install it manually (i.e., `install.packages("haven", dep = TRUE)`).

Examples

```
## Not run:
library(jmvReadWrite);
fleOMV <- system.file("extdata", "AlbumSales.omv", package = "jmvReadWrite");
fleTmp <- paste0(tempfile(), ".omv");
sort_omv(fleInp = fleOMV, fleOut = fleTmp, varSrt = "Image");
dtaFrm <- read_omv(fleInp = fleTmp);
cat(dtaFrm$Image);
```

```
# shows that the variable "Image" is sorted in ascending order
cat(is.unsorted(dtaFrm$Image));
# is.unsorted (which checks for whether the variable is NOT sorted) returns FALSE
sort_omv(fleInp = fleOMV, fleOut = fleTmp, varSrt = "-Image");
# variables can also be sorted in descending order by preceding them with "-"
dtaFrm <- read_omv(fleInp = fleTmp);
cat(dtaFrm$Image);
# shows that the variable "Image" is now sorted in descending order
cat(is.unsorted(dtaFrm$Image));
# this first returns TRUE (the variable is not in ascending order, i.e., unsorted)
cat(is.unsorted(-dtaFrm$Image));
# if the sign of the variable is changed, it returns FALSE (i.e., the variable is
# NOT unsorted)
unlink(fleTmp);

## End(Not run)
```

`ToothGrowth` *The Effect of Vitamin C on Tooth Growth in Guinea Pigs*

Description

The Effect of Vitamin C on Tooth Growth in Guinea Pigs

Usage

```
ToothGrowth
```

Format

A data.frame with 60 rows and 6 variables

ID characterID of the guinea pig

supp factorSupplement type (VC: Vitamin C or OJ: Orange juice)

supp2 factorTransformation of the supplement type (factor to numerical: VC = 1; OJ = 2)

dose numericDose in grams / day

dose2 ordered factorDose in grams / day

len numericTooth length

logLen numericNatural logarithm of the tooth length (len)

wide2long_omv *Converts .omv-files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org) from wide to long format*

Description

Converts .omv-files for the statistical spreadsheet 'jamovi' (https://www.jamovi.org) from wide to long format

Usage

```
wide2long_omv(
  fleInp = "",
  fleOut = "",
  varLst = c(),
  varExc = c(),
  varID = "ID",
  varTme = "cond",
  varSep = "_",
  varSrt = c(),
  usePkg = c("foreign", "haven"),
  selSet = "",
  ...
)
```

Arguments

fleInp	Name (including the path, if required) of the data file to be read ("FILENAME.omv"; default: ""); can be any supported file type, see Details below
fleOut	Name (including the path, if required) of the data file to be written ("FILENAME.omv"; default: ""); if empty, FILENAME from fleInp is extended with "_long(file extension -> .omv)"
varLst	List / set of variables that are to be transformed into single (time-varying) variables in long format (default: c())
varExc	List / set of variables to be excluded from the variable list (default: c())
varID	Name(s) of one or more variables that (is created to) identify the same group / individual (if empty, "ID" is added with row numbers identifying cases; default: "ID")
varTme	Name of the variable that (is created to) differentiate multiple records from the same group / individual (default: "cond"; a counter is added for each time-varying part)
varSep	Character that separates the variables in varLst into a time-varying part and a part that forms the variable name in long format (*"" in "VAR_1", "VAR_2", default: ""*)

varSrt	Variable(s) that are used to sort the data frame (see Details; if empty, the order returned from reshape is kept; default: c())
usePkg	Name of the package: "foreign" or "haven" that shall be used to read SPSS, Stata and SAS files; "foreign" is the default (it comes with base R), but "haven" is newer and more comprehensive
selSet	Name of the data set that is to be selected from the workspace (only applies when reading .RData-files)
...	Additional arguments passed on to methods; see Details below

Details

If varLst is empty, it is tried to generate it using all variables in the data frame except those defined by varExc and varID. The variable(s) in varID have to be unique identifiers (in the original dataset), those in varExc don't have this requirement. It is generally recommended that the variable names in varExc and varID should not include the variable separator (defined in varSep; default: "_") For further arguments, see the help for reshape (where varLst ~ varying, varSep ~ sep, varID ~ idvar, varTme ~ timevar). varSrt is a character vector containing column names that are used to sort the data frame before it is written. The ellipsis-parameter (...) can be used to submit arguments / parameters to the functions that are used for transforming or reading the data. The transformation uses `reshape`. When reading the data, the functions are: `read_omv` (for jamovi-files), `read.table` (for CSV / TSV files; using similar defaults as `read.csv` for CSV and `read.delim` for TSV which both are based upon `read.table` but with adjusted defaults for the respective file types), `readRDS` (for rds-files), `read_sav` (needs R-package "haven") or `read.spss` (needs R-package "foreign") for SPSS-files, `read_dta` ("haven") / `read.dta` ("foreign") for Stata-files, `read_sas` ("haven") for SAS-data-files, and `read_xpt` ("haven") / `read.xport` ("foreign") for SAS-transport-files. If you would like to use "haven", it may be needed to install it manually (i.e., `install.packages("haven", dep = TRUE)`).

Examples

```
## Not run:
library(jmvReadWrite)
# generate a test dataframe with 100 (imaginary) participants / units of
# observation (ID), and 8 repeated measurements of variable (X_1, X_2, ...)
dtaInp <- cbind(data.frame(ID = as.character(seq(1:100))),
                stats::setNames(
                    as.data.frame(matrix(runif(800, -10, 10), nrow = 100)),
                    paste0("X_", 1:8)))
cat(str(dtaInp))
# 'data.frame': 100 obs. of  9 variables:
#  $ ID : chr  "1" "2" "3" "4" ...
#  $ X_1: num  ...
#  $ X_2: num  ...
#  $ X_3: num  ...
#  $ X_4: num  ...
#  $ X_5: num  ...
#  $ X_6: num  ...
#  $ X_7: num  ...
#  $ X_8: num  ...
# this data set is stored as (temporary) RDS-file and later processed by wide2long
```

```
nmeInp <- paste0(tempfile(), ".rds")
nmeOut <- paste0(tempfile(), ".omv")
saveRDS(dtaInp, nmeInp)
wide2long_omv(fleInp = nmeInp, fleOut = nmeOut, varID = "ID", varTme = "measure",
    varLst = setdiff(names(dtaInp), "ID"), varSrt = c("ID", "measure"))
# it is required to give at least the arguments fleInp and varID
# "reshape" then assigns all variables expect the variable defined by varID to
# varLst (but throws a warning)
# varSrt enforces sorting the data set after the transformation (sorted, the
# measurements within one person come after another; unsorted all measurements
# for one repetition would come after another)

# check whether the file was created and its size
cat(list.files(dirname(nmeOut), basename(nmeOut)))
# -> "file[...].omv" ([...] contains a random combination of numbers / characters
cat(file.info(nmeOut)$size)
# -> 6939 (approximate size; size may differ in every run [in dependence of how
#          well the generated random data can be compressed])
cat(str(read_omv(nmeOut, sveAtt = FALSE)))
# the data set is now transformed into long (and each the measurements is now
# indicated by the "measure")
# 'data.frame': 800 obs. of  3 variables:
#  $ ID     : Factor w/ 100 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 2 2 ...
#   ..- attr(*, "missingValues")= list()
#  $ measure: Factor w/ 8 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 1 2 ...
#   ..- attr(*, "missingValues")= list()
#  $ X      : num  ...
#   ..- attr(*, "missingValues")= list()

unlink(nmeInp)
unlink(nmeOut)

## End(Not run)
```

`write_omv` *Write files to be used with the statistical spreadsheet 'jamovi' (https://www.jamovi.org)*

Description

Write files to be used with the statistical spreadsheet 'jamovi' (https://www.jamovi.org)

Usage

```
write_omv(dtaFrm = NULL, fleOut = "", retDbg = FALSE)
```

Arguments

dtaFrm	Data frame to be exported (default: NULL)
fleOut	Name / position of the output file to be generated ("FILENAME.omv"; default: "")
retDbg	Whether to return a list with debugging information (see Value; default: FALSE)

Details

jamovi has a specific measurement level / type "ID" (in addition to the "standard" ones "Nominal", "Ordinal", and "Continuous"). "ID" is used for columns that contain some form of ID (e.g., a participant code). In order to set a variable of your data frame to "ID", you have to manually set an attribute `jmv-id` (e.g., `attr(dtaFrm$column, "jmv-id") = TRUE`).

Value

a list (if retDbg == TRUE), containing the meta data (mtaDta, metadata.json in the OMV-file), the extended data (xtdDta, xdata.json in the OMV-file) and the original data frame (dtaFrm)

Examples

```
## Not run:
library(jmvReadWrite);

# use the data set "ToothGrowth" and, if it exists, write it as
# jamovi-file using write_omv()
data("ToothGrowth");
fleOMV <- paste0(tempfile(), ".omv");
# typically, one would use a "real" file name instead of tempfile(),
# e.g., "Data1.omv"
dtaDbg = write_omv(ToothGrowth, fleOMV, retDbg = TRUE);
print(names(dtaDbg));
# the print-function is only used to force devtools::run_examples()
# to show output
# -> "mtaDta" "xtdDta" "dtaFrm"
# returns a list with the metadata (mtaDta, e.g., column and data type),
# the extended data (xtdDta, e.g., variable lables), and the data frame
# (dtaFrm) the purpose of these variables is merely for checking (under-
# standing the file format) and debugging

# check whether the file was written to the disk, get the file informa-
# tion (size, etc.) and delete the file afterwards
print(list.files(dirname(fleOMV), basename(fleOMV)));
# -> "file[...].omv" ([...] is a combination of random numbers / characters
print(file.info(fleOMV)$size);
# -> 2448 (size may differ on different OSes)
unlink(fleOMV);

## End(Not run)
```

Index