

Package ‘socialmixr’

January 19, 2026

Title Social Mixing Matrices for Infectious Disease Modelling

Version 0.5.0

Description Provides methods for sampling contact matrices from diary data for use in infectious disease modelling, as discussed in Mossong et al. (2008) <[doi:10.1371/journal.pmed.0050074](https://doi.org/10.1371/journal.pmed.0050074)>.

License MIT + file LICENSE

Depends R (>= 3.5.0)

Imports checkmate, countrycode, curl, data.table, grDevices, httr, jsonlite, lifecycle, lubridate, memoise, purrr, oai, wpp2017, xml2, cli, rlang, methods

Suggests ggplot2, here, knitr, quarto, reshape2, rmarkdown, roxyglobals (>= 1.0.0), testthat, withr

VignetteBuilder knitr

Encoding UTF-8

LazyData true

NeedsCompilation no

RoxygenNote 7.3.3

URL <https://github.com/epiforecasts/socialmixr>,
<https://epiforecasts.io/socialmixr/>

BugReports <https://github.com/epiforecasts/socialmixr/issues>

Config/testthat/edition 3

Author Sebastian Funk [aut, cre],
Lander Willem [aut],
Hugo Gruson [aut],
Maria Bekker-Nielsen Dunbar [ctb],
Carl A. B. Pearson [ctb],
Sam Clifford [ctb],
Christopher Jarvis [ctb],
Alexis Robert [ctb],
Niel Hens [ctb],
Pietro Coletti [col, dtm],
Lloyd Chapman [ctb]

Maintainer Sebastian Funk <sebastian.funk@lshtm.ac.uk>
Repository CRAN
Date/Publication 2026-01-19 16:30:02 UTC

Contents

assign_age_groups	2
as_contact_survey	3
check	5
clean	6
contact_matrix	6
download_survey	10
get_citation	11
get_survey	11
is_doi	12
limits_to_agegroups	13
list_surveys	14
load_survey	14
matrix_plot	15
polymod	17
pop_age	18
reduce_agegroups	19
survey	19
survey_countries	20
survey_country_population	21
wpp_age	21
wpp_countries	22

Index	23
--------------	-----------

assign_age_groups	<i>Assign age groups in survey data</i>
-------------------	---

Description

This function processes age data in a survey object. It imputes ages from ranges, handles missing values, and assigns age groups.

Usage

```
assign_age_groups(  
  survey,  
  age_limits = NULL,  
  estimated_participant_age = c("mean", "sample", "missing"),  
  estimated_contact_age = c("mean", "sample", "missing"),  
  missing_participant_age = c("remove", "keep"),  
  missing_contact_age = c("remove", "sample", "keep", "ignore")  
)
```

Arguments

survey	a <code>survey()</code> object
age_limits	lower limits of the age groups over which to construct the matrix. Defaults to NULL. If NULL, age limits are inferred from participant and contact ages.
estimated_participant_age	if set to "mean" (default), people whose ages are given as a range (in columns named "..._est_min" and "..._est_max") but not exactly (in a column named "..._exact") will have their age set to the mid-point of the range; if set to "sample", the age will be sampled from the range; if set to "missing", age ranges will be treated as missing
estimated_contact_age	if set to "mean" (default), contacts whose ages are given as a range (in columns named "..._est_min" and "..._est_max") but not exactly (in a column named "..._exact") will have their age set to the mid-point of the range; if set to "sample", the age will be sampled from the range; if set to "missing", age ranges will be treated as missing
missing_participant_age	if set to "remove" (default), participants without age information are removed; if set to "keep", participants with missing age are kept and treated as a separate age group
missing_contact_age	if set to "remove" (default), participants that have contacts without age information are removed; if set to "sample", contacts without age information are sampled from all the contacts of participants of the same age group; if set to "keep", contacts with missing age are kept and treated as a separate age group; if set to "ignore", contact with missing age are ignored in the contact analysis

Value

The survey object with processed age data.

Examples

```
polymod_grouped <- assign_age_groups(polymod)
polymod_grouped
polymod_custom <- assign_age_groups(polymod, age_limits = c(0, 5, 10, 15))
polymod_custom
```

as_contact_survey	<i>Check contact survey data</i>
-------------------	----------------------------------

Description

Checks that a survey fulfills all the requirements to work with the 'contact_matrix' function

Usage

```
as_contact_survey(
  x,
  id_column = "part_id",
  country_column = NULL,
  year_column = NULL,
  ...,
  id.column = deprecated(),
  country.column = deprecated(),
  year.column = deprecated()
)
```

Arguments

<code>x</code>	list containing • an element named 'participants', a data frame containing participant information • an element named 'contacts', a data frame containing contact information • (optionally) an element named 'reference', a list containing information needed to reference the survey, in particular it can contain\$a "title", "bibtype", "author", "doi", "publisher", "note", "year"
<code>id_column</code>	the column in both the participants and contacts data frames that links contacts to participants
<code>country_column</code>	the column in the participants data frame containing the country in which the participant was queried; if NULL (default), will use "country" column if present
<code>year_column</code>	the column in the participants data frame containing the year in which the participant was queried; if NULL (default), will use "year" column if present
<code>...</code>	additional arguments (currently ignored)
<code>id.column</code> , <code>country.column</code> , <code>year.column</code>	[Deprecated] Use the underscore versions (e.g., <code>id_column</code>) instead.

Value

invisibly returns a character vector of the relevant columns

Examples

```
data(polymod)
check(polymod)
```

check	<i>Check contact survey data</i>
-------	----------------------------------

Description

Checks that a survey fulfills all the requirements to work with the 'contact_matrix' function

Usage

```
## S3 method for class 'contact_survey'
check(
  x,
  id.column = "part_id",
  participant.age.column = "part_age",
  country.column = "country",
  year.column = "year",
  contact.age.column = "cnt_age",
  ...
)
```

Arguments

x	A survey() object
id.column	the column in both the participants and contacts data frames that links contacts to participants
participant.age.column	the column in the participants data frame containing participants' age; if this does not exist, at least columns "..._exact", "..._est_min" and "..._est_max" must (see the <code>estimated.participant.age</code> option in contact_matrix())
country.column	the column in the participants data frame containing the country in which the participant was queried
year.column	the column in the participants data frame containing the year in which the participant was queried
contact.age.column	the column in the contacts data frame containing the age of contacts; if this does not exist, at least columns "..._exact", "..._est_min" and "..._est_max" must (see the <code>estimated.contact.age</code> option in contact_matrix())
...	ignored

Value

invisibly returns a character vector of the relevant columns

Examples

```
data(polymod)
check(polymod)
```

clean	<i>Clean contact survey data</i>
-------	----------------------------------

Description

Cleans survey data to work with the 'contact_matrix' function

Usage

```
## S3 method for class 'contact_survey'
clean(
  x,
  participant_age_column = "part_age",
  ...,
  participant.age.column = deprecated()
)
```

Arguments

x A [survey\(\)](#) object

participant_age_column the column in x\$participants containing participants' age

... ignored

participant.age.column **[Deprecated]** Use participant_age_column instead.

Value

a cleaned survey in the correct format

Examples

```
data(polymod)
cleaned <- clean(polymod) # not really necessary as the 'polymod' data set has already been cleaned
```

contact_matrix	<i>Generate a contact matrix from diary survey data</i>
----------------	---

Description

Samples a contact survey

Usage

```

contact_matrix(
  survey,
  countries = NULL,
  survey_pop = NULL,
  age_limits = NULL,
  filter = NULL,
  counts = FALSE,
  symmetric = FALSE,
  split = FALSE,
  sample_participants = FALSE,
  estimated_participant_age = c("mean", "sample", "missing"),
  estimated_contact_age = c("mean", "sample", "missing"),
  missing_participant_age = c("remove", "keep"),
  missing_contact_age = c("remove", "sample", "keep", "ignore"),
  weights = NULL,
  weigh_dayofweek = FALSE,
  weigh_age = FALSE,
  weight_threshold = NA,
  symmetric_norm_threshold = 2,
  sample_all_age_groups = FALSE,
  sample_participants_max_tries = 1000,
  return_part_weights = FALSE,
  return_demography = NA,
  per_capita = FALSE,
  ...,
  survey.pop = deprecated(),
  age.limits = deprecated(),
  sample.participants = deprecated(),
  estimated.participant.age = deprecated(),
  estimated.contact.age = deprecated(),
  missing.participant.age = deprecated(),
  missing.contact.age = deprecated(),
  weigh.dayofweek = deprecated(),
  weigh.age = deprecated(),
  weight.threshold = deprecated(),
  symmetric.norm.threshold = deprecated(),
  sample.all.age.groups = deprecated(),
  sample.participants.max.tries = deprecated(),
  return.part.weights = deprecated(),
  return.demography = deprecated(),
  per.capita = deprecated()
)

```

Arguments

survey	a survey() object.
countries	limit to one or more countries; if NULL (default), will use all countries in the

	survey; these can be given as country names or 2-letter (ISO Alpha-2) country codes.
survey_pop	survey population – either a data frame with columns 'lower.age.limit' and 'population', or a character vector giving the name(s) of a country or countries from the list that can be obtained via <code>wpp_countries</code> ; if NULL (default), will use the country populations from the chosen countries, or all countries in the survey if countries is NULL.
age_limits	lower limits of the age groups over which to construct the matrix. If NULL (default), age limits are inferred from participant and contact ages.
filter	any filters to apply to the data, given as list of the form (column=filter_value) - only contacts that have 'filter_value' in 'column' will be considered. If multiple filters are given, they are all applied independently and in the sequence given. Default value is NULL; no filtering performed.
counts	whether to return counts (instead of means).
symmetric	whether to make matrix symmetric, such that $c_{ij}N_i = c_{ji}N_j$.
split	whether to split the contact matrix into the mean number of contacts, in each age group (split further into the product of the mean number of contacts across the whole population (<code>mean.contacts</code>), a normalisation constant (<code>normalisation</code>) and age-specific variation in contacts (<code>contacts</code>)), multiplied with an assortativity matrix (<code>assortativity</code>) and a population multiplier (<code>demography</code>). For more detail on this, see the "Getting Started" vignette.
sample_participants	whether to sample participants randomly (with replacement); done multiple times this can be used to assess uncertainty in the generated contact matrices. See the "Bootstrapping" section in the vignette for how to do this.
estimated_participant_age	if set to "mean" (default), people whose ages are given as a range (in columns named "..._est_min" and "..._est_max") but not exactly (in a column named "..._exact") will have their age set to the mid-point of the range; if set to "sample", the age will be sampled from the range; if set to "missing", age ranges will be treated as missing
estimated_contact_age	if set to "mean" (default), contacts whose ages are given as a range (in columns named "..._est_min" and "..._est_max") but not exactly (in a column named "..._exact") will have their age set to the mid-point of the range; if set to "sample", the age will be sampled from the range; if set to "missing", age ranges will be treated as missing.
missing_participant_age	if set to "remove" (default), participants without age information are removed; if set to "keep", participants with missing age are kept and will appear in the contact matrix in a row labelled "NA".
missing_contact_age	if set to "remove" (default), participants that have contacts without age information are removed; if set to "sample", contacts without age information are sampled from all the contacts of participants of the same age group; if set to "keep", contacts with missing age are kept and will appear in the contact matrix

in a column labelled "NA"; if set to "ignore", contacts without age information are removed from the analysis (but the participants that made them are kept).

weights column names(s) of the participant data of the `survey()` object with user-specified weights (default = empty vector).

weigh_dayofweek whether to weigh social contacts data by the day of the week (weight $(5/7 / N_{\text{week}} / N)$ for weekdays and $(2/7 / N_{\text{weekend}} / N)$ for weekends).

weigh_age whether to weigh social contacts data by the age of the participants (vs. the populations' age distribution).

weight_threshold threshold value for the standardized weights before running an additional standardisation (default 'NA' = no cutoff).

symmetric_norm_threshold threshold value for the normalization weights when `symmetric = TRUE` before showing a warning that that large differences in the size of the sub-populations are likely to result in artefacts when making the matrix symmetric (default 2).

sample_all_age_groups what to do if sampling participants (with `sample_participants = TRUE`) fails to sample participants from one or more age groups; if `FALSE` (default), corresponding rows will be set to NA, if `TRUE` the sample will be discarded and a new one taken instead.

sample_participants_max_tries maximum number of attempts when `sample_all_age_groups = TRUE`; defaults to 1000.

return_part_weights boolean to return the participant weights.

return_demography boolean to explicitly return demography data that corresponds to the survey data (default 'NA' = if demography data is requested by other function parameters).

per_capita whether to return a matrix with contact rates per capita (default is `FALSE` and not possible if 'counts=TRUE' or 'split=TRUE').

... further arguments to pass to `get_survey()`, `check()` and `pop_age()` (especially column names).

`survey.pop,` `age.limits,` `sample.participants,`
`estimated.participant.age,` `estimated.contact.age,`
`missing.participant.age,` `missing.contact.age,` `weigh.dayofweek,`
`weigh.age,` `weight.threshold,` `symmetric.norm.threshold,`
`sample.all.age.groups,` `sample.participants.max.tries,`
`return.part.weights,` `return.demography,` `per.capita`

[Deprecated] Use the underscore-separated versions of these arguments instead.

Value

a contact matrix, and the underlying demography of the surveyed population

Author(s)

Sebastian Funk

Examples

```
data(polymod)
contact_matrix(
  survey = polymod,
  countries = "United Kingdom",
  age_limits = c(0, 1, 5, 15)
)
```

download_survey

Download a survey from its Zenodo repository

Description**[Deprecated]**

download_survey() has been deprecated in favour of `contactsurveys::download_survey()`.
 download_survey() downloads survey data from Zenodo.

Usage

```
download_survey(survey, dir = NULL, sleep = 1)
```

Arguments

survey	a URL (see <code>contactsurveys::list_surveys()</code>)
dir	a directory to save the files to; if not given, will save to a temporary directory
sleep	time to sleep between requests to avoid overloading the server (passed on to Sys.sleep)

Value

a vector of filenames that can be used with [load_survey](#)

Examples

```
# we recommend using the contactsurveys package for download_survey()
## Not run:
# if needed, discover surveys with:
contactsurveys::list_surveys()
peru_survey <- download_survey("https://doi.org/10.5281/zenodo.1095664")
# -->
peru_survey <- contactsurveys::download_survey(
  "https://doi.org/10.5281/zenodo.1095664"
)

## End(Not run)
```

get_citation	<i>Citation for a survey</i>
--------------	------------------------------

Description

[Deprecated]

get_citation() has been deprecated in favour of `contactsurveys::get_citation()`.

Gets a full citation for a [survey\(\)](#).

Usage

```
get_citation(x)
```

Arguments

x a character vector of surveys to cite

Value

citation as bibentry

Examples

```
# we recommend using the contactsurveys package for get_citation()
## Not run:
data(polymod)
citation <- contactsurveys::get_citation(polymod)
print(citation)
print(citation, style = "bibtex")

## End(Not run)
```

get_survey	<i>Get a survey, either from its Zenodo repository, a set of files, or a survey variable</i>
------------	--

Description

[Deprecated]

get_survey() has been deprecated in favour of using `contactsurveys::download_survey()` and then [load_survey\(\)](#).

Downloads survey data, or extracts them from files, and returns a clean data set. If a survey URL is accessed multiple times, the data will be cached (unless `clear_cache` is set to TRUE) to avoid repeated downloads.

If survey objects are used repeatedly the downloaded files can be saved and reloaded between sessions then survey objects can be saved/loaded using `base::saveRDS()` and `base::readRDS()`, or via the individual survey files that can be downloaded using `download_survey()` and subsequently loaded using `load_survey()`.

Usage

```
get_survey(survey, clear_cache = FALSE, ...)
```

Arguments

<code>survey</code>	a DOI or url to get the survey from, or a <code>survey()</code> object (in which case only cleaning is done).
<code>clear_cache</code>	logical, whether to clear the cache before downloading the survey; by default, the cache is not cleared and so multiple calls of this function to access the same survey will not result in repeated downloads.
<code>...</code>	options for <code>clean()</code> , which is called at the end of this

Value

a survey in the correct format

Examples

```
## Not run:
list_surveys()
peru_doi <- "https://doi.org/10.5281/zenodo.1095664"
peru_survey <- get_survey(peru_doi)
## --> We now recommend:
peru_survey <- contactsurveys::download_survey(peru_doi)
peru_data <- load_survey(peru_survey)

## End(Not run)
```

is_doi

Checks if a character string is a DOI

Description

Checks if a character string is a DOI

Usage

```
is_doi(x)
```

Arguments

<code>x</code>	Character vector; the string or strings to check
----------------	--

Value

Logical; TRUE if x is a DOI, FALSE otherwise

Author(s)

Sebastian Funk

limits_to_agegroups	<i>Convert lower age limits to age groups.</i>
---------------------	--

Description

Mostly used for plot labelling

Usage

```
limits_to_agegroups(  
  x,  
  limits = sort(unique(x)),  
  notation = c("dashes", "brackets")  
)
```

Arguments

x	age limits to transform
limits	lower age limits; if not given, will use all limits in x
notation	whether to use bracket notation, e.g. [0,4) or dash notation, e.g. 0-4)

Value

Age groups as specified in notation

Examples

```
limits_to_agegroups(c(0, 5, 10))
```

list_surveys	<i>List all surveys available for download</i>
--------------	--

Description

[Deprecated]

list_surveys() has been deprecated in favour of `contactsurveys::list_surveys()`.

Usage

```
list_surveys(clear_cache = FALSE)
```

Arguments

clear_cache	logical, whether to clear the cache before downloading the survey; by default, the cache is not cleared and so multiple calls of this function to access the same survey will not result in repeated downloads.
-------------	---

Value

character vector of surveys

Examples

```
# we recommend using the contactsurveys package now for listing surveys.
## Not run:
contactsurveys::list_surveys()

## End(Not run)
```

load_survey	<i>Load a survey from local files</i>
-------------	---------------------------------------

Description

Loads a survey from a local file system. Tables are expected as csv files, and a reference (if present) as JSON.

Usage

```
load_survey(files, participant_key = NULL, ...)
```

Arguments

`files` a vector of file names as returned by `download_survey()`

`participant_key` character vector specifying columns that uniquely identify participant observations. For cross-sectional surveys this is typically just "part_id" (the default). For longitudinal surveys with multiple observations per participant, specify additional columns like `c("part_id", "wave")`. When NULL (the default), the function will auto-detect if additional columns are needed and inform you.

`...` options for `clean()`, which is called at the end of this

Value

a survey in the correct format. For longitudinal surveys with multiple observations per participant, the returned object includes an `observation_key` field containing the column names (excluding `part_id`) that distinguish observations for the same participant.

Examples

```
## Not run:
list_surveys()
peru_files <- download_survey("https://doi.org/10.5281/zenodo.1095664")
peru_survey <- load_survey(peru_files)

# For longitudinal surveys, specify the unique key explicitly:
france_files <- download_survey("https://doi.org/10.5281/zenodo.1157918")
france_survey <- load_survey(france_files,
  participant_key = c("part_id", "wave", "studyDay")
)

## End(Not run)
```

matrix_plot	<i>Draws an image plot of a contact matrix with a legend strip and the numeric values in the cells.</i>
-------------	---

Description

This function combines the R `image.plot` function with numeric contact rates in the matrix cells.

Usage

```
matrix_plot(
  mij,
  min.legend = 0,
  max.legend = NA,
  num.digits = 2,
  num.colors = 50,
```

```

    main,
    xlab,
    ylab,
    legend.width,
    legend.mar,
    legend.shrink,
    cex.lab,
    cex.axis,
    cex.text,
    color.palette = heat.colors
  )

```

Arguments

<code>mij</code>	a contact matrix containing contact rates between participants of age <i>i</i> (rows) with contacts of age <i>j</i> (columns). This is the default matrix format of <code>contact_matrix()</code> .
<code>min.legend</code>	the color scale minimum (default = 0). Set to NA to use the minimum value of <code>mij</code> .
<code>max.legend</code>	the color scale maximum (default = NA). Set to NA to use the maximum value of <code>mij</code> .
<code>num.digits</code>	the number of digits when rounding the contact rates (default = 2). Use NA to disable this.
<code>num.colors</code>	the number of color breaks (default = 50)
<code>main</code>	the figure title
<code>xlab</code>	a title for the x axis (default: "Age group (years)")
<code>ylab</code>	a title for the y axis (default: "Contact age group (years)")
<code>legend.width</code>	width of the legend strip in characters. Default is 1.
<code>legend.mar</code>	width in characters of legend margin. Default is 5.1.
<code>legend.shrink</code>	amount to shrink the size of legend relative to the full height or width of the plot. Default is 0.9.
<code>cex.lab</code>	size of the x and y labels (default: 1.2)
<code>cex.axis</code>	size of the axis labels (default: 0.8)
<code>cex.text</code>	size of the numeric values in the matrix (default: 1)
<code>color.palette</code>	the color palette to use (default: <code>heat.colors()</code>). Other examples are <code>topo.colors()</code> , <code>terrain.colors()</code> and <code>hcl.colors()</code> . User-defined functions are also possible if they take the number of colors to be in the palette as function argument.

Details

This is a function using basic R graphics to visualise a social contact matrix.

Author(s)

Lander Willem

Examples

```
## Not run:
data(polymod)
mij <- contact_matrix(polymod, countries = "United Kingdom", age_limits = c(0, 18, 65))$matrix
matrix_plot(mij)

## End(Not run)
```

polymod

Social contact data from 8 European countries

Description

A dataset containing social mixing diary data from 8 European countries: Belgium, Germany, Finland, Great Britain, Italy, Luxembourg, The Netherlands and Poland. The Data are fully described in Mossong J, Hens N, Jit M, Beutels P, Auranen K, Mikolajczyk R, et al. (2008) Social Contacts and Mixing Patterns Relevant to the Spread of Infectious Diseases. PLoS Med 5(3): e74.

Usage

polymod

Format

A list of two data frames:

participants the study participant, with age, country, year and day of the week (starting with 1 = Monday)

contacts reported contacts of the study participants. The variable `phys_contact` has two levels (1 denotes physical contact while 2 denotes non-physical contact), `duration_multi` has five levels (1 is less than 5 minutes while 5 is more than 4 hours, increasing in the order found in Figure 1 in Mossong et al.), and `frequency_multi` has five levels (1 is daily, 2 is weekly, 3 is monthly, 4 is less often, and 5 is first time) All other variables are described on the Zenodo repository of the data, available at [doi:10.5281/zenodo.1043437](https://doi.org/10.5281/zenodo.1043437)

Source

[doi:10.1371/journal.pmed.0050074](https://doi.org/10.1371/journal.pmed.0050074)

pop_age

*Change age groups in population data***Description**

This changes population data to have age groups with the given age_limits, extrapolating linearly between age groups (if more are requested than available) and summing populations (if fewer are requested than available)

Usage

```
pop_age(
  pop,
  age_limits = NULL,
  pop_age_column = "lower.age.limit",
  pop_column = "population",
  ...,
  age_limits = deprecated(),
  pop_age_column = deprecated(),
  pop_column = deprecated()
)
```

Arguments

pop	a data frame with columns indicating lower age limits and population sizes (see 'pop_age_column' and 'pop_column')
age_limits	lower age limits of age groups to extract; if NULL (default), the population data is returned unchanged
pop_age_column	column in the 'pop' data frame indicating the lower age group limit
pop_column	column in the 'pop' data frame indicating the population size
...	ignored
age_limits, pop_age_column, pop_column	[Deprecated] Use the underscore versions (e.g., age_limits) instead.

Value

data frame of age-specific population data

Examples

```
ages_it_2015 <- wpp_age("Italy", 2015)

# Modify the age data.frame to get age groups of 10 years instead of 5
pop_age(ages_it_2015, age_limits = seq(0, 100, by = 10))

# The function will also automatically interpolate if necessary
pop_age(ages_it_2015, age_limits = c(0, 18, 40, 65))
```

reduce_agegroups	<i>Reduce the number of age groups given a broader set of limits</i>
------------------	--

Description

Operates on lower limits

Usage

```
reduce_agegroups(x, limits)
```

Arguments

<code>x</code>	vector of limits
<code>limits</code>	new limits

Value

vector with the new age groups

Examples

```
reduce_agegroups(seq_len(20), c(0, 5, 10))
```

survey	<i>Contact survey</i>
--------	-----------------------

Description

Deprecated. Use `as_survey` instead.

Usage

```
survey(participants, contacts, reference = NULL)
```

Arguments

<code>participants</code>	a <code>data.frame</code> containing information on participants
<code>contacts</code>	a <code>data.frame</code> containing information on contacts
<code>reference</code>	a list containing information needed to reference the survey, in particular it can contain a "title", "bibtype", "author", "doi", "publisher", "note", "year"

Value

a new survey object

Author(s)

Sebastian Funk

survey_countries	<i>List all countries contained in a survey</i>
------------------	---

Description**[Deprecated]****Usage**

```
survey_countries(survey, country.column = "country", ...)
```

Arguments

survey	a DOI or url to get the survey from, or a survey() object (in which case only cleaning is done).
country.column	column in the survey indicating the country
...	further arguments for get_survey()

Details

`survey_countries()` has been deprecated in favour of using `contactsurveys::download_survey()`, and [load_survey\(\)](#), and then exploring the country column yourself.

Value

list of countries

Examples

```
data(polymod)
survey_countries(polymod)
## --> we now recommend
## Not run:
doi_peru <- "10.5281/zenodo.1095664" # nolint
# download the data with the contactsurveys package
peru_survey <- contactsurveys::download_survey(doi_peru)
# load the survey with socialmixr
peru_data <- socialmixr::load_survey(peru_survey)
# find the unique country - assuming your data has a "country" column:
unique(peru_data$participants$country)

## End(Not run)
```

survey_country_population

Get survey country population data

Description

Looks up the country and year inside a survey, or a provided "countries" value, and determines the corresponding demographics in the world population prospects data using `wpp_age()`.

Usage

```
survey_country_population(survey, countries = NULL)
```

Arguments

survey	A <code>survey()</code> object, with column "country" in "participants".
countries	Optional. A character vector of country names. If specified, this will be used instead of the potential "country" column in "participants".

Value

A data table with population data by age group for the survey countries, aggregated by lower age limit. The function will error if no country information is available from either the survey or countries argument.

Examples

```
survey_country_population(polymod)
survey_country_population(polymod, countries = "Belgium")
survey_country_population(polymod, countries = c("Belgium", "Italy"))
```

wpp_age

Get age-specific population data according to the World Population Prospects 2017 edition

Description

This uses data from the wpp2017 package but combines male and female, and converts age groups to lower age limits. If the requested year is not present in the historical data, wpp projections are used.

Usage

```
wpp_age(countries, years)
```

Arguments

countries	countries, will return all if not given
years	years, will return all if not given

Value

data frame of age-specific population data

Examples

```
wpp_age("Italy", c(1990, 2000))
```

wpp_countries	<i>List all countries and regions for which socialmixr has population data</i>
---------------	--

Description

Uses the World Population Prospects data from the wpp2017 package

Usage

```
wpp_countries()
```

Value

list of countries

Examples

```
wpp_countries()
```

Index

- * **datasets**
 - polymod, [17](#)
- as_contact_survey, [3](#)
- assign_age_groups, [2](#)
- base::readRDS(), [12](#)
- base::saveRDS(), [12](#)
- check, [5](#)
- check(), [9](#)
- clean, [6](#)
- clean(), [12](#), [15](#)
- contact_matrix, [6](#)
- contact_matrix(), [5](#), [16](#)
- download_survey, [10](#)
- download_survey(), [12](#), [15](#)
- get_citation, [11](#)
- get_survey, [11](#)
- get_survey(), [9](#), [20](#)
- hcl.colors(), [16](#)
- heat.colors(), [16](#)
- is_doi, [12](#)
- limits_to_agegroups, [13](#)
- list_surveys, [14](#)
- load_survey, [10](#), [14](#)
- load_survey(), [11](#), [12](#), [20](#)
- matrix_plot, [15](#)
- polymod, [17](#)
- pop_age, [18](#)
- pop_age(), [9](#)
- reduce_agegroups, [19](#)
- survey, [19](#)
- survey(), [3](#), [5–7](#), [9](#), [11](#), [12](#), [20](#), [21](#)
- survey_countries, [20](#)
- survey_country_population, [21](#)
- Sys.sleep, [10](#)
- terrain.colors(), [16](#)
- topo.colors(), [16](#)
- wpp_age, [21](#)
- wpp_age(), [21](#)
- wpp_countries, [22](#)