Package: EpiStandard (via r-universe)

June 1, 2026

Title Directly Standardise Rates by Age

Version 0.2.0

Maintainer Elin Rowlands `<elin.rowlands@ndorms.ox.ac.uk>`

Description Provides functions for age standardisation of epidemiological measures such as incidence and prevalence rates. It allows users to apply standard population structures to observed age-specific estimates in order to obtain comparable summary measures across populations or time periods. Functions support calculation of standardised rates, outcome counts, and corresponding confidence intervals. The tools are designed to facilitate reproducible and transparent adjustment for differences in age distributions in epidemiological and public health research.

License Apache License (>= 2)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Suggests gt, knitr, rmarkdown, testthat (>= 3.0.0), DBI, duckdb, IncidencePrevalence, omopgenerics

Config/testthat/edition 3

Imports cli, dplyr, rlang, stringr, tidyr, tidyselect, tibble

VignetteBuilder knitr

Depends R (>= 4.1.0)

URL `https://oxford-pharmacoepi.github.io/EpiStandard/`, `https://github.com/oxford-pharmacoepi/EpiStandard`

Language en-GB

LazyData true

BugReports `https://github.com/oxford-pharmacoepi/EpiStandard/issues`

Config/pak/sysreqs libicu-dev

Repository https://oxford-pharmacoepi.r-universe.dev

Date/Publication 2026-03-31 18:47:32 UTC

RemoteUrl https://github.com/oxford-pharmacoepi/epistandard

RemoteRef HEAD

RemoteSha 39c8069577ba3d81989757f20d5de2dc939d19ee

Contents

`directlyStandardiseRates`

Calculate directly standardised rates

Description

Computes crude and directly standardised rates. Rates can be stratified by variables of interest.

Usage

```
directlyStandardiseRates(
  data,
  event,
  denominator,
  age = "age_group",
  pop = "pop",
  strata = NULL,
  addMissingGroups = TRUE,
  refdata = standardPopulation("Europe")
)
```

Arguments

`data`	A data frame with the event counts to be standardised.
`event`	Name of the column in data that corresponds to the event counts.
`denominator`	Name of the column in data that corresponds to the denominator population (in person-time, e.g person-days, person-years etc).
`age`	Name of the column in data and refdata that corresponds to age groups.
`pop`	Name of the column in refdata that corresponds to the standard population in each age group.
`strata`	Name of the columns in data for which rates are calculated by.

`addMissingGroups`	If TRUE, any age groups present in refdata but not in data will be added and set to 0. If false, these age groups will be removed from refdata.
`refdata`	A data frame representing the standard population. It must contain two columns: age, with the different age groups (notice that this column name must be the same as in data, defined by the input age); and pop, with the number of individuals in each corresponding age group.

Value

Data frame with crude and standardised rates.

Examples

```
# An example of calculating directly standardised rates
# Data example is from Table 1 (p.132) of Fundamentals of Epidemiology by Schoenbach, 2000.

# The following table shows the number of deaths, for 5 different age groups,
# in the states of Miami and Alaska:
data <- data.frame(
      state = rep(c('Miami',"Alaska"), c(5,5)),
      age_groups = rep(c('00-14','15-24','25-44','45-64','65+'),2),
      deaths = c(136, 57, 208, 1016, 3605, 59, 18, 37, 90, 81),
    general_population = c(114350,80259,133440,142670,92168,37164,20036,32693,14947,2077))

# We aim to standardise the number of deaths per each state. To do that, we will use the following
# US standard population:
standardised_population <- data.frame(
                            age_groups = c('00-14','15-24','25-44','45-64','65+'),
                            pop = c(23961000,15420000,21353000,19601000,10685000))

# Now we will use the function dsr to calculate the direct standardised rates
# (per 1000 individuals) using a 95% CI calculated by the gamma method:
my_results <- directlyStandardiseRates(data = data,
                  event = "deaths",
                  denominator  = "general_population",
                  age   = "age_groups",
                  pop   = "pop",
                  strata = "state",
                  refdata = standardised_population)
# View results
my_results
```

`mergeAgeGroups` *Create new merged age groups*

Description

Create new merged age groups

Usage

```
mergeAgeGroups(
  refdata,
  newGroups,
  event = NULL,
  age = "age_group",
  pop = "pop",
  ageRange = c(0, 150),
  strata = NULL
)
```

Arguments

`refdata`	Standard population dataset you want to use.
`newGroups`	Create a list of new age groups you want to create.
`event`	Column in refdata with outcome counts.
`age`	Column in refdata with age values.
`pop`	Column in refdata with population counts, preferably in person-time.
`ageRange`	Specify the age range of the population of interest.
`strata`	Column or columns to stratify by.

Value

Data frame with age groups and population counts.

Examples

```
standardPopulation("Europe")|> dplyr::glimpse()

merged_data <- mergeAgeGroups(standardPopulation("Europe"), c("0-19", "20-64", "65-150"))

merged_data |> dplyr::glimpse()
```

`standardPopulation` *Standard population*

Description

Standard population

Usage

```
standardPopulation(region = "Europe")
```

Arguments

`region`	Region of standard population. Can be either 'Europe' or 'World'.

Value

Data frame with age groups and population counts for chosen region.

Examples

```
standard_data <- standardPopulation(region = "Europe")
```

Index