Package 'msSPChelpR'

November 4, 2020

Title Helper Functions for Second Primary Cancer Analyses

Version 0.8.6

Description A collection of helper functions for analyzing Second Primary Cancer data, including functions to reshape data, to calculate patient states and analyze cancer incidence.

License GPL-3

URL https://marianschmidt.github.io/msSPChelpR/

BugReports https://github.com/marianschmidt/msSPChelpR/issues

Depends R (>= 3.5)

Imports data.table (>= 1.12.9), dplyr (>= 1.0.0), lubridate, magrittr, progress, purrr, rlang (>= 0.1.2), sjlabelled, stringr, tidyselect, tidytable (>= 0.5.5), tidyr (>= 1.0.0), utils

Suggests haven, tibble, rmarkdown, knitr, testthat (>= 2.1.0)

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.1.1

Language en-US

LazyData true

NeedsCompilation no

Author Marian Eberl [aut, cre] (<https://orcid.org/0000-0001-6584-3197>)

Maintainer Marian Eberl <marian.eberl@tum.de>

Repository CRAN

Date/Publication 2020-11-04 10:50:03 UTC

R topics documented:

`asir` *Calculate age-standardized incidence rates*

Description

Calculate age-standardized incidence rates

Usage

```
asir(
  df,
  dattype = "zfkd",
  std_pop = "ESP2013",
  truncate_std_pop = FALSE,
  futime_src = "refpop",
  summarize_groups = "none",
  count_var,
  stdpop_df = standard_population,
  refpop_df = population,
  region_var = NULL,
  age_var = NULL,
  sex_var = NULL,
  year_var = NULL,
  site_var = NULL,
  futime_var = NULL,
  pyar_var = NULL,
  alpha = 0.05
)
```

Arguments

df	dataframe in wide format
dattype	can be "zfkd" or "seer" or empty. Will set default variable names from dataset.
std_pop	can be either "ESP2013, ESP1976, WHO1960
truncate_std_pop	if TRUE standard population will be truncated for all age-groups that do not occur in df
futime_src	can be either "refpop" or "cohort"
summarize_groups	option to define summarizing stratified groups. Default is "none". If you want to define variables that should be summarized into one group, you can chose from region_var, sex_var, year_var. Define multiple summarize variables by summarize_groups = c("region", "sex", "year")
count_var	variable to be counted as observed case. Should be 1 for case to be counted.
stdpop_df	df where standard population is defined. It is assumed that stdpop_df has the columns "sex" for biological sex, "age" for age-groups, "standard_pop" for name of standard population (e.g. "European Standard Population 2013) and "population_n" for size of standard population age-group. stdpop_df must use the same category coding of age and sex as age_var and sex_var.
refpop_df	df where reference population data is defined. Only required if option futime = "refpop" is chosen. It is assumed that refpop_df has the columns "region" for region, "sex" for biological sex, "age" for age-groups (can be single ages or 5-year brackets), "year" for time period (can be single year or 5-year brackets), "population_pyar" for person-years at risk in the respective age/sex/year cohort. refpop_df must use the same category coding of age, sex, region, year and site as age_var, sex_var, region_var, year_var and site_var.
region_var	variable in df that contains information on region where case was incident. Default is set if dattype is given.
age_var	variable in df that contains information on age-group. Default is set if dattype is given.
sex_var	variable in df that contains information on biological sex. Default is set if dattype is given.
year_var	variable in df that contains information on year or year-period when case was incident. Default is set if dattype is given.
site_var	variable in df that contains information on ICD code of case diagnosis. Default is set if dattype is given.
futime_var	variable in df that contains follow-up time per person (in years) in cohort (can only be used with futime_src = "cohort"). Default is set if dattype is given.
pyar_var	variable in refpop_df that contains person-years-at-risk in reference population (can only be used with futime_src = "refpop") Default is set if dattype is given.
alpha	significance level for confidence interval calculations. Default is alpha = 0.05 which will give 95 percent confidence intervals.

Value

df

Examples

```
#load sample data
data("us_second_cancer")
data("standard_population")
data("population_us")

#make wide data as this is the required format
usdata_wide <- us_second_cancer %>%
                    #only use sample
                    dplyr::filter(as.numeric(fake_id) < 200000) %>%
                    msSPChelpR::reshape_wide_tidyr(case_id_var = "fake_id",
                    time_id_var = "SEQ_NUM", timevar_max = 2)

#create count variable
usdata_wide <- usdata_wide %>%
                   dplyr::mutate(count_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ 1,
                    TRUE ~ 0))

#remove cases for which no reference population exists
usdata_wide <- usdata_wide %>%
             dplyr::filter(t_yeardiag.2 %in% c("1990 - 1994", "1995 - 1999", "2000 - 2004",
                                               "2005 - 2009", "2010 - 2014"))


#now we can run the function
msSPChelpR::asir(usdata_wide,
      dattype = "seer",
      std_pop = "ESP2013",
      truncate_std_pop = FALSE,
      futime_src = "refpop",
      summarize_groups = "none",
      count_var = "count_spc",
      refpop_df = population_us,
      region_var = "registry.1",
      age_var = "fc_agegroup.1",
      sex_var = "sex.1",
      year_var = "t_yeardiag.2",
      site_var = "t_site_icd.2",
      pyar_var = "population_pyar")
```

`calc_futime` *Calculate follow-up time per case until end of follow-up depending on pat_status - tidyverse version*

Description

Calculate follow-up time per case until end of follow-up depending on pat_status - tidyverse version

Usage

```
calc_futime(
  wide_df,
  futime_var_new = "p_futimeyrs",
  fu_end,
  dattype = "zfkd",
  check = TRUE,
  time_unit = "years",
  status_var = "p_status",
  lifedat_var = NULL,
  fcdat_var = NULL,
  spcdat_var = NULL
)
```

Arguments

wide_df	dataframe in wide format
futime_var_new	Name of the newly calculated variable for follow-up time. Default is p_futimeyrs.
fu_end	end of follow-up in time format YYYY-MM-DD.
dattype	Type of cancer registry data. Can be "seer" or "zfkd". Default is "zfkd".
check	Check newly calculated variable p_status by printing frequency table. Default is TRUE.
time_unit	Unit of follow-up time (can be "days", "weeks", "months", "years"). Default is "years".
status_var	Name of the patient status variable that was previously created. Default is p_status.
lifedat_var	Name of variable containing Date of Death. Will override dattype preset.
fcdat_var	Name of variable containing Date of Primary Cancer diagnosis. Will override dattype preset.
spcdat_var	Name of variable containing Date of SPC diagnosis Will override dattype preset.

Value

wide_df

Examples

```
#load sample data
data("us_second_cancer")

#prep step - make wide data as this is the required format
usdata_wide <- us_second_cancer %>%
```

```
                    msSPChelpR::reshape_wide_tidyr(case_id_var = "fake_id",
                    time_id_var = "SEQ_NUM", timevar_max = 10)

#prep step - calculate p_spc variable
usdata_wide <- usdata_wide %>%
                dplyr::mutate(p_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ "No SPC",
                                                !is.na(t_site_icd.2)   ~ "SPC developed",
                                                       TRUE ~ NA_character_)) %>%
                 dplyr::mutate(count_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ 1,
                                                            TRUE ~ 0))

#prep step - create patient status variable
usdata_wide <- usdata_wide %>%
                  msSPChelpR::pat_status(., fu_end = "2017-12-31", dattype = "seer",
                                  status_var = "p_status", life_var = "p_alive.1",
                               birthdat_var = "datebirth.1", lifedat_var = "datedeath.1")

#now we can run the function
msSPChelpR::calc_futime(usdata_wide,
                        futime_var_new = "p_futimeyrs",
                        fu_end = "2017-12-31",
                        dattype = "seer",
                        time_unit = "years",
                        status_var = "p_status",
                        lifedat_var = "datedeath.1",
                        fcdat_var = "t_datediag.1",
                        spcdat_var = "t_datediag.2")
```

`calc_futime_tt` *Calculate follow-up time per case until end of follow-up depending on pat_status - tidytable version*

Description

Calculate follow-up time per case until end of follow-up depending on pat_status - tidytable version

Usage

```
calc_futime_tt(
  wide_df,
  futime_var_new = "p_futimeyrs",
  fu_end,
  dattype = "zfkd",
  check = TRUE,
  time_unit = "years",
  status_var = "p_status",
  lifedat_var = NULL,
  fcdat_var = NULL,
```

```
  spcdat_var = NULL
)
```

Arguments

wide_df	dataframe or data.table in wide format
futime_var_new	Name of the newly calculated variable for follow-up time. Default is p_futimeyrs.
fu_end	end of follow-up in time format YYYY-MM-DD.
dattype	Type of cancer registry data. Can be "seer" or "zfkd". Default is "zfkd".
check	Check newly calculated variable p_status by printing frequency table. Default is TRUE.
time_unit	Unit of follow-up time (can be "days", "weeks", "months", "years"). Default is "years".
status_var	Name of the patient status variable that was previously created. Default is p_status.
lifedat_var	Name of variable containing Date of Death. Will override dattype preset.
fcdat_var	Name of variable containing Date of Primary Cancer diagnosis. Will override dattype preset.
spcdat_var	Name of variable containing Date of SPC diagnosis Will override dattype preset.

Value

wide_df

Examples

```
#load sample data
data("us_second_cancer")

#make wide data as this is the required format
usdata_wide <- us_second_cancer %>%
                   msSPChelpR::reshape_wide_tidyr(case_id_var = "fake_id",
                   time_id_var = "SEQ_NUM", timevar_max = 10)

#prep step - calculate p_spc variable
usdata_wide <- usdata_wide %>%
               dplyr::mutate(p_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ "No SPC",
                                                      !is.na(t_site_icd.2)   ~ "SPC developed",
                                                             TRUE ~ NA_character_)) %>%
                dplyr::mutate(count_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ 1,
                                                           TRUE ~ 0))

#prep step - create patient status variable
usdata_wide <- usdata_wide %>%
                 msSPChelpR::pat_status(., fu_end = "2017-12-31", dattype = "seer",
                                        status_var = "p_status", life_var = "p_alive.1",
                                birthdat_var = "datebirth.1", lifedat_var = "datedeath.1")
```

```
#now we can run the function
msSPChelpR::calc_futime_tt(usdata_wide,
                        futime_var_new = "p_futimeyrs",
                        fu_end = "2017-12-31",
                        dattype = "seer",
                        time_unit = "years",
                        status_var = "p_status",
                        lifedat_var = "datedeath.1",
                        fcdat_var = "t_datediag.1",
                        spcdat_var = "t_datediag.2")
```

`ir_crosstab` *Calculate crude incidence rates and crosstabulate results by break variables*

Description

Calculate crude incidence rates and crosstabulate results by break variables

Usage

```
ir_crosstab(
  df,
  dattype = "zfkd",
  count_var,
  xbreak_var = "none",
  ybreak_vars,
  collapse_ci = FALSE,
  add_total = "no",
  add_n_percentages = FALSE,
  futime_var = NULL,
  alpha = 0.05
)
```

Arguments

`df`	dataframe in wide format
`dattype`	can be "zfkd" or "seer" or empty. Will set default variable names from dataset.
`count_var`	variable to be counted as observed case. Should be 1 for case to be counted.
`xbreak_var`	variable from df by which rates should be stratified in columns of result df. Default is "none".
`ybreak_vars`	variables from df by which rates should be stratified in rows of result df. Multiple variables will result in appended rows in result df. y_break_vars is required.
`collapse_ci`	If TRUE upper and lower confidence interval will be collapsed into one column separated by "-". Default is FALSE.

add_total option to add a row of totals. Can be either "no" for not adding such a row or "top" or "bottom" for adding it at the first or last row. Default is "no".

add_n_percentages
option to add a column of percentages for n_base in its respective yvar_group. Can only be used when xbreak_var = "none". Default is FALSE.

futime_var variable in df that contains follow-up time per person (in years). Default is set if dattype is given.

alpha significance level for confidence interval calculations. Default is alpha = 0.05 which will give 95 percent confidence intervals.

Value

df

Examples

```
#load sample data
data("us_second_cancer")

#prep step - make wide data as this is the required format
usdata_wide <- us_second_cancer %>%
                   msSPChelpR::reshape_wide_tidyr(case_id_var = "fake_id",
                   time_id_var = "SEQ_NUM", timevar_max = 10)

#prep step - calculate p_spc variable
usdata_wide <- usdata_wide %>%
                dplyr::mutate(p_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ "No SPC",
                                                !is.na(t_site_icd.2)   ~ "SPC developed",
                                                       TRUE ~ NA_character_)) %>%
                 dplyr::mutate(count_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ 1,
                                                            TRUE ~ 0))

#prep step - create patient status variable
usdata_wide <- usdata_wide %>%
                  msSPChelpR::pat_status(., fu_end = "2017-12-31", dattype = "seer",
                                  status_var = "p_status", life_var = "p_alive.1",
                             birthdat_var = "datebirth.1", lifedat_var = "datedeath.1")

#now we can run the function
usdata_wide <- usdata_wide %>%
                 msSPChelpR::calc_futime(.,
                        futime_var_new = "p_futimeyrs",
                        fu_end = "2017-12-31",
                        dattype = "seer",
                        time_unit = "years",
                        status_var = "p_status",
                        lifedat_var = "datedeath.1",
                        fcdat_var = "t_datediag.1",
                        spcdat_var = "t_datediag.2")

#for example, you can calculate incidence and summarize by sex and registry
```

```
msSPChelpR::ir_crosstab(usdata_wide,
      dattype = "seer",
      count_var = "count_spc",
      xbreak_var = "none",
      ybreak_vars = c("sex.1", "registry.1"),
      collapse_ci = FALSE,
      add_total = "no",
      add_n_percentages = FALSE,
      futime_var = "p_futimeyrs",
      alpha = 0.05)
```

`ir_crosstab_byfutime` *Calculate crude incidence rates and cross-tabulate results by break variables; cumulative FU-times as are used as xbreak_var*

Description

Calculate crude incidence rates and cross-tabulate results by break variables; cumulative FU-times as are used as xbreak_var

Usage

```
ir_crosstab_byfutime(
  df,
  dattype = "zfkd",
  count_var,
  futime_breaks = c(0, 0.5, 1, 5, 10, Inf),
  ybreak_vars,
  collapse_ci = FALSE,
  add_total = "no",
  futime_var = NULL,
  alpha = 0.05
)
```

Arguments

`df`	dataframe in wide format
`dattype`	can be "zfkd" or "seer" or empty. Will set default variable names from dataset.
`count_var`	variable to be counted as observed case. Should be 1 for case to be counted.
`futime_breaks`	vector that indicates split points for follow-up time groups (in years) that will be used as xbreak_var. Default is c(0, .5, 1, 5, 10, Inf) that will result in 5 groups (up to 6 months, 6-12 months, 1-5 years, 5-10 years, 10+ years).
`ybreak_vars`	variables from df by which rates should be stratified in rows of result df. Multiple variables will result in appended rows in result df. y_break_vars is required.

collapse_ci	If TRUE upper and lower confidence interval will be collapsed into one column separated by "-". Default is FALSE.
add_total	option to add a row of totals. Can be either "no" for not adding such a row or "top" or "bottom" for adding it at the first or last row. Default is "no".
futime_var	variable in df that contains follow-up time per person (in years). Default is set if dattype is given.
alpha	significance level for confidence interval calculations. Default is alpha = 0.05 which will give 95 percent confidence intervals.

Value

df

Examples

```
#load sample data
data("us_second_cancer")

#prep step - make wide data as this is the required format
usdata_wide <- us_second_cancer %>%
                    #only use sample
                    dplyr::filter(as.numeric(fake_id) < 200000) %>%
                    msSPChelpR::reshape_wide_tidyr(case_id_var = "fake_id",
                    time_id_var = "SEQ_NUM", timevar_max = 2)

#prep step - calculate p_spc variable
usdata_wide <- usdata_wide %>%
                dplyr::mutate(p_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ "No SPC",
                                                !is.na(t_site_icd.2)   ~ "SPC developed",
                                                        TRUE ~ NA_character_)) %>%
                 dplyr::mutate(count_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ 1,
                                                            TRUE ~ 0))

#prep step - create patient status variable
usdata_wide <- usdata_wide %>%
                  msSPChelpR::pat_status(., fu_end = "2017-12-31", dattype = "seer",
                                  status_var = "p_status", life_var = "p_alive.1",
                               birthdat_var = "datebirth.1", lifedat_var = "datedeath.1")

#now we can run the function
usdata_wide <- usdata_wide %>%
                msSPChelpR::calc_futime(.,
                       futime_var_new = "p_futimeyrs",
                       fu_end = "2017-12-31",
                       dattype = "seer",
                       time_unit = "years",
                       status_var = "p_status",
                       lifedat_var = "datedeath.1",
                       fcdat_var = "t_datediag.1",
                       spcdat_var = "t_datediag.2")
```

```
#for example, you can calculate incidence and summarize by sex and registry
msSPChelpR::ir_crosstab_byfutime(usdata_wide,
      dattype = "seer",
      count_var = "count_spc",
      futime_breaks = c(0, .5, 1, 5, 10, Inf),
      ybreak_vars = c("sex.1", "registry.1"),
      collapse_ci = FALSE,
      add_total = "no",
      futime_var = "p_futimeyrs",
      alpha = 0.05)
```

`pat_status` *Calculate patient status at specific end of follow-up - tidyverse version*

Description

Calculate patient status at specific end of follow-up - tidyverse version

Usage

```
pat_status(
  wide_df,
  fu_end = NULL,
  dattype = "zfkd",
  status_var = "p_status",
  life_var = NULL,
  spc_var = NULL,
  birthdat_var = NULL,
  lifedat_var = NULL,
  lifedatmin_var = NULL,
  fcdat_var = NULL,
  spcdat_var = NULL,
  life_stat_alive = NULL,
  life_stat_dead = NULL,
  spc_stat_yes = NULL,
  spc_stat_no = NULL,
  lifedat_fu_end = NULL,
  use_lifedatmin = FALSE,
  check = TRUE,
  as_labelled_factor = FALSE
)
```

Arguments

`wide_df`	dataframe in wide format
`fu_end`	end of follow-up in time format YYYY-MM-DD.

dattype	Type of cancer registry data. Can be "seer" or "zfkd". Default is "zfkd".
status_var	Name of the newly calculated variable for patient status. Default is p_status.
life_var	Name of variable containing life status. Will override dattype preset.
spc_var	Name of variable containing SPC status. Will override dattype preset.
birthdat_var	Name of variable containing Date of Birth. Will override dattype preset.
lifedat_var	Name of variable containing Date of Death. Will override dattype preset.
lifedatmin_var	Name of variable containing the minimum Date of Death when true DoD is missing. Will override dattype preset. Will only be used if use_lifedatmin = TRUE.
fcdat_var	Name of variable containing Date of Primary Cancer diagnosis. Will override dattype preset.
spcdat_var	Name of variable containing Date of SPC diagnosis Will override dattype preset.
life_stat_alive	Value for alive status in life_var. Will override dattype preset.
life_stat_dead	Value for dead status in life_var. Will override dattype preset.
spc_stat_yes	Value for SPC occurred in spc_var. Will override dattype preset.
spc_stat_no	Value for no SPC in spc_var. Will override dattype preset.
lifedat_fu_end	Date of last FU of alive status in registry data. Will override dattype preset (2017-03-31 for zfkd; 2018-12-31 for seer).
use_lifedatmin	If TRUE, option to use Date of Death from lifedatmin_var when DOD is missing. Default is FALSE.
check	Check newly calculated variable p_status. Default is TRUE.
as_labelled_factor	If TRUE, output status_var as labelled factor variable. Default is FALSE.

Value

wide_df

Examples

```
#load sample data
data("us_second_cancer")

#prep step - make wide data as this is the required format
usdata_wide <- us_second_cancer %>%
                   msSPChelpR::reshape_wide_tidyr(case_id_var = "fake_id",
                   time_id_var = "SEQ_NUM", timevar_max = 10)

#prep step - calculate p_spc variable
usdata_wide <- usdata_wide %>%
                dplyr::mutate(p_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ "No SPC",
                                                !is.na(t_site_icd.2)   ~ "SPC developed",
                                                       TRUE ~ NA_character_)) %>%
                 dplyr::mutate(count_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ 1,
```

```
                                                                TRUE ~ 0))

#now we can run the function
msSPChelpR::pat_status(usdata_wide,
                       fu_end = "2017-12-31",
                       dattype = "seer",
                       status_var = "p_status",
                       life_var = "p_alive.1",
                       spc_var = NULL,
                       birthdat_var = "datebirth.1",
                       lifedat_var = "datedeath.1",
                       use_lifedatmin = FALSE,
                       check = TRUE,
                       as_labelled_factor = FALSE)
```

---

pat_status_tt *Calculate patient status at specific end of follow-up - tidytable version*

Description

Calculate patient status at specific end of follow-up - tidytable version

Usage

```
pat_status_tt(
  wide_df,
  fu_end = NULL,
  dattype = "zfkd",
  status_var = "p_status",
  life_var = NULL,
  spc_var = NULL,
  birthdat_var = NULL,
  lifedat_var = NULL,
  lifedatmin_var = NULL,
  fcdat_var = NULL,
  spcdat_var = NULL,
  life_stat_alive = NULL,
  life_stat_dead = NULL,
  spc_stat_yes = NULL,
  spc_stat_no = NULL,
  lifedat_fu_end = NULL,
  use_lifedatmin = FALSE,
  check = TRUE,
  as_labelled_factor = FALSE
)
```

Arguments

wide_df	dataframe or data.table in wide format
fu_end	end of follow-up in time format YYYY-MM-DD.
dattype	Type of cancer registry data. Can be "seer" or "zfkd". Default is "zfkd".
status_var	Name of the newly calculated variable for patient status. Default is p_status.
life_var	Name of variable containing life status. Will override dattype preset.
spc_var	Name of variable containing SPC status. Will override dattype preset.
birthdat_var	Name of variable containing Date of Birth. Will override dattype preset.
lifedat_var	Name of variable containing Date of Death. Will override dattype preset.
lifedatmin_var	Name of variable containing the minimum Date of Death when true DoD is missing. Will override dattype preset. Will only be used if use_lifedatmin = TRUE.
fcdat_var	Name of variable containing Date of Primary Cancer diagnosis. Will override dattype preset.
spcdat_var	Name of variable containing Date of SPC diagnosis Will override dattype preset.
life_stat_alive	Value for alive status in life_var. Will override dattype preset.
life_stat_dead	Value for dead status in life_var. Will override dattype preset.
spc_stat_yes	Value for SPC occurred in spc_var. Will override dattype preset.
spc_stat_no	Value for no SPC in spc_var. Will override dattype preset.
lifedat_fu_end	Date of last FU of alive status in registry data. Will override dattype preset (2017-03-31 for zfkd; 2018-12-31 for seer).
use_lifedatmin	If TRUE, option to use Date of Death from lifedatmin_var when DOD is missing. Default is FALSE.
check	Check newly calculated variable p_status. Default is TRUE.
as_labelled_factor	If TRUE, output status_var as labelled factor variable. Default is FALSE.

Value

wide_df

Examples

```
#load sample data
data("us_second_cancer")

#prep step - make wide data as this is the required format
usdata_wide <- us_second_cancer %>%
                   msSPChelpR::reshape_wide_tidyr(case_id_var = "fake_id",
                   time_id_var = "SEQ_NUM", timevar_max = 10)

#prep step - calculate p_spc variable
usdata_wide <- usdata_wide %>%
```

```
                dplyr::mutate(p_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ "No SPC",
                                                       !is.na(t_site_icd.2)   ~ "SPC developed",
                                                              TRUE ~ NA_character_)) %>%
                 dplyr::mutate(count_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ 1,
                                                            TRUE ~ 0))

#now we can run the function
msSPChelpR::pat_status_tt(usdata_wide,
                          fu_end = "2017-12-31",
                          dattype = "seer",
                          status_var = "p_status",
                          life_var = "p_alive.1",
                          spc_var = NULL,
                          birthdat_var = "datebirth.1",
                          lifedat_var = "datedeath.1",
                          use_lifedatmin = FALSE,
                          check = TRUE,
                          as_labelled_factor = FALSE)
```

`population_us` *US Populations*

Description

Dataset that contains different standard populations needed to run some package functions

Usage

```
population_us
```

Format

A data frame with the following variables:

`region` Region / Registry

`year` Year group

`sex` Sex

`age` Age group

`race` Race

`population_pyar` Population Years used for rate calculation (PYAR)

`population_n_per_year` Absolute Population in single years or periods (PYAR / 5 years)]

renumber_time_id *Renumber the time ID per case (i.e. Tumor sequence)*

Description

Renumber the time ID per case (i.e. Tumor sequence)

Usage

```
renumber_time_id(
  df,
  new_time_id_var,
  dattype = "zfkd",
  case_id_var = NULL,
  time_id_var = NULL,
  diagdat_var = NULL,
  timevar_max = Inf
)
```

Arguments

df	dataframe
new_time_id_var	Name of the newly calculated variable for time_id. Required.
dattype	Type of cancer registry data. Can be "seer" or "zfkd". Default is "zfkd".
case_id_var	String with name of ID variable indicating same patient. E.g. `case_id_var="PUBCSNUM"` for SEER data.
time_id_var	String with name of variable that indicates diagnosis per patient. E.g. `time_id_var="SEQ_NUM"` for SEER data.
diagdat_var	String with name of variable that indicates date of diagnosis per event. E.g. `diagdat_var="t_datediag"` for SEER data.
timevar_max	Numeric; default Inf. Maximum number of cases per id. All tumors > timevar_max will be deleted.

Value

df

Examples

```
data(us_second_cancer)
us_second_cancer %>%
 #only select first 10000 rows so example runs faster
 dplyr::slice(1:10000) %>%
 msSPChelpR::renumber_time_id(new_time_id_var = "t_tumid",
```

```
                    dattype = "seer",
                    case_id_var = "fake_id")
```

`renumber_time_id_tt` *Renumber the time ID per case (i.e. Tumor sequence) - tidytable version*

Description

Renumber the time ID per case (i.e. Tumor sequence) - tidytable version

Usage

```
renumber_time_id_tt(
  df,
  new_time_id_var,
  dattype = "zfkd",
  case_id_var = NULL,
  time_id_var = NULL,
  diagdat_var = NULL,
  timevar_max = Inf
)
```

Arguments

`df`	dataframe
`new_time_id_var`	Name of the newly calculated variable for time_id. Required.
`dattype`	Type of cancer registry data. Can be "seer" or "zfkd". Default is "zfkd".
`case_id_var`	String with name of ID variable indicating same patient. E.g. `case_id_var="PUBCSNUM"` for SEER data.
`time_id_var`	String with name of variable that indicates diagnosis per patient. E.g. `time_id_var="SEQ_NUM"` for SEER data.
`diagdat_var`	String with name of variable that indicates date of diagnosis per event. E.g. `diagdat_var="t_datediag"` for SEER data.
`timevar_max`	Numeric; default Inf. Maximum number of cases per id. All tumors > timevar_max will be deleted.

Value

df

Examples

```
data(us_second_cancer)
us_second_cancer %>%
 #only select first 10000 rows so example runs faster
 dplyr::slice(1:10000) %>%
 msSPChelpR::renumber_time_id_tt(new_time_id_var = "t_tumid",
                            dattype = "seer",
                            case_id_var = "fake_id")
```

---

reshape_long *Reshape dataset to long format - stats::reshape version*

Description

Reshape dataset to long format - stats::reshape version

Usage

```
reshape_long(wide_df, case_id_var, time_id_var, datsize = Inf, chunks = 1)
```

Arguments

wide_df	dataframe in wide format
case_id_var	String with name of ID variable indicating same patient. E.g. idvar="PUBCSNUM" for SEER data.
time_id_var	String with name of variable that indicates diagnosis per patient. E.g. timevar="SEQ_NUM" for SEER data.
datsize	Number of rows to be taken from df. This parameter is mainly for testing. Default is Inf so that df is fully processed.
chunks	Numeric; default 1. Technical parameter how the data is split during reshaping.

Value

long df

Examples

```
data(us_second_cancer)

#prep step - reshape wide a sample of 10000 rows from us_second_cancer
usdata_wide_sample <- msSPChelpR::reshape_wide(us_second_cancer,
                         case_id_var = "fake_id",
                         time_id_var = "SEQ_NUM",
                         timevar_max = 2,
```

```
                          datsize = 10000)

#now we can reshape long again
msSPChelpR::reshape_long(usdata_wide_sample,
                          case_id_var = "fake_id",
                          time_id_var = "SEQ_NUM")
```

---

reshape_long_tidyr *Reshape dataset to wide format - tidyr version*

Description

Reshape dataset to wide format - tidyr version

Usage

```
reshape_long_tidyr(wide_df, case_id_var, time_id_var, datsize = Inf)
```

Arguments

wide_df	dataframe
case_id_var	String with name of ID variable indicating same patient. E.g. idvar="PUBCSNUM" for SEER data.
time_id_var	String with name of variable that indicates diagnosis per patient. E.g. timevar="SEQ_NUM" for SEER data.
datsize	Number of rows to be taken from df. This parameter is mainly for testing. Default is Inf so that df is fully processed.

Value

long_df

Examples

```
data(us_second_cancer)

#prep step - reshape wide a sample of 10000 rows from us_second_cancer
usdata_wide_sample <- msSPChelpR::reshape_wide(us_second_cancer,
                          case_id_var = "fake_id",
                          time_id_var = "SEQ_NUM",
                          timevar_max = 2,
                          datsize = 10000)

#now we can reshape long again
msSPChelpR::reshape_long_tidyr(usdata_wide_sample,
```

```
                    case_id_var = "fake_id",
                    time_id_var = "SEQ_NUM")
```

`reshape_wide` *Reshape dataset to wide format*

Description

Reshape dataset to wide format

Usage

```
reshape_wide(
  df,
  case_id_var,
  time_id_var,
  timevar_max = 6,
  datsize = Inf,
  chunks = 10
)
```

Arguments

`df`	dataframe
`case_id_var`	String with name of ID variable indicating same patient. E.g. `idvar="PUBCSNUM"` for SEER data.
`time_id_var`	String with name of variable that indicates diagnosis per patient. E.g. `timevar="SEQ_NUM"` for SEER data.
`timevar_max`	Numeric; default 6. Maximum number of cases per id. All tumors > timevar_max will be deleted before reshaping.
`datsize`	Number of rows to be taken from df. This parameter is mainly for testing. Default is Inf so that df is fully processed.
`chunks`	Numeric; default 10. Technical parameter how the data is split during reshaping.

Value

df

Examples

```
data(us_second_cancer)

msSPChelpR::reshape_wide(us_second_cancer,
                         case_id_var = "fake_id",
                         time_id_var = "SEQ_NUM",
                         timevar_max = 2,
                         datsize = 10000)
```

`reshape_wide_tidyr` *Reshape dataset to wide format - tidyr version*

Description

Reshape dataset to wide format - tidyr version

Usage

```
reshape_wide_tidyr(
  df,
  case_id_var,
  time_id_var,
  timevar_max = 6,
  datsize = Inf
)
```

Arguments

`df`	dataframe
`case_id_var`	String with name of ID variable indicating same patient. E.g. `idvar="PUBCSNUM"` for SEER data.
`time_id_var`	String with name of variable that indicates diagnosis per patient. E.g. `timevar="SEQ_NUM"` for SEER data.
`timevar_max`	Numeric; default 6. Maximum number of cases per id. All tumors > timevar_max will be deleted before reshaping.
`datsize`	Number of rows to be taken from df. This parameter is mainly for testing. Default is Inf so that df is fully processed.

Value

df

Examples

```
data(us_second_cancer)

msSPChelpR::reshape_wide_tidyr(us_second_cancer,
                          case_id_var = "fake_id",
                          time_id_var = "SEQ_NUM",
                          timevar_max = 2,
                          datsize = 10000)
```

---

reshape_wide_tt *Reshape dataset to wide format - tidytable version*

Description

Reshape dataset to wide format - tidytable version

Usage

```
reshape_wide_tt(df, case_id_var, time_id_var, timevar_max = 6, datsize = Inf)
```

Arguments

df	dataframe
case_id_var	String with name of ID variable indicating same patient. E.g. idvar="PUBCSNUM" for SEER data.
time_id_var	String with name of variable that indicates diagnosis per patient. E.g. timevar="SEQ_NUM" for SEER data.
timevar_max	Numeric; default 6. Maximum number of cases per id. All tumors > timevar_max will be deleted before reshaping.
datsize	Number of rows to be taken from df. This parameter is mainly for testing. Default is Inf so that df is fully processed.

Value

wide_df

Examples

```
data(us_second_cancer)

msSPChelpR::reshape_wide_tt(us_second_cancer,
                          case_id_var = "fake_id",
                          time_id_var = "SEQ_NUM",
                          timevar_max = 2,
                          datsize = 10000)
```

sir_byfutime *Calculate standardized incidence ratios with custom grouping variables stratified by follow-up time*

Description

Calculate standardized incidence ratios with custom grouping variables stratified by follow-up time

Usage

```
sir_byfutime(
  df,
  dattype = "zfkd",
  ybreak_vars = "none",
  xbreak_var = "none",
  futime_breaks = c(0, 0.5, 1, 5, 10, Inf),
  count_var,
  refrates_df = rates,
  calc_total_row = TRUE,
  calc_total_fu = TRUE,
  region_var = NULL,
  age_var = NULL,
  sex_var = NULL,
  year_var = NULL,
  race_var = NULL,
  site_var = NULL,
  futime_var = NULL,
  alpha = 0.05
)
```

Arguments

df	dataframe in wide format
dattype	can be "zfkd" or "seer" or empty. Will set default variable names from dataset.
ybreak_vars	variables from df by which SIRs should be stratified in result df. Multiple variables will result in appended rows in result df. Careful: do not chose any variables that are dependent on occurrence of count_var (e.g. Histology of second cancer). If y_break_vars = "none", no stratification is performed. Default is "none".
xbreak_var	One variable from df by which SIRs should be stratified as a second dimension in result df. This variable will be added as a second stratification dimension to ybreak_vars and all variables will be calculated for subpopulations of x and y combinations. Careful: do not chose any variables that are dependent on occurrence of count_var (e.g. Year of second cancer). If y_break_vars = "none", no stratification is performed. Default is "none".

futime_breaks	vector that indicates split points for follow-up time groups (in years) that will be used as xbreak_var. Default is c(0, .5, 1, 5, 10, Inf) that will result in 5 groups (up to 6 months, 6-12 months, 1-5 years, 5-10 years, 10+ years). If you don't want to split by follow-up time, use futime_breaks = "none".
count_var	variable to be counted as observed case. Cases are usually the second cancers. Should be 1 for case to be counted.
refrates_df	df where reference rate from general population are defined. It is assumed that refrates_df has the columns "region" for region, "sex" for biological sex, "age" for age-groups (can be single ages or 5-year brackets), "year" for time period (can be single year or 5-year brackets), "incidence_crude_rate" for incidence rate in the respective age/sex/year cohort. refrates_df must use the same category coding of age, sex, region, year and t_site as age_var, sex_var, region_var, year_var and site_var.
calc_total_row	option to calculate a row of totals. Can be either FALSE for not adding such a row or TRUE for adding it at the first row. Default is TRUE.
calc_total_fu	option to calculate totals for follow-up time. Can be either FALSE for not adding such a column or TRUE for adding. Default is TRUE.
region_var	variable in df that contains information on region where case was incident. Default is set if dattype is given.
age_var	variable in df that contains information on age-group. Default is set if dattype is given.
sex_var	variable in df that contains information on sex. Default is set if dattype is given.
year_var	variable in df that contains information on year or year-period when case was incident. Default is set if dattype is given.
race_var	optional argument for dattype="seer", if SIR should be calculated stratified by race. If you want to use this option, provide variable name of df that contains race information.
site_var	variable in df that contains information on ICD code of case diagnosis. Cases are usually the second cancers. Default is set if dattype is given.
futime_var	variable in df that contains follow-up time per person between date of first cancer and any of death, date of event (case), end of FU date (in years; whatever event comes first). Default is set if dattype is given.
alpha	significance level for confidence interval calculations. Default is alpha = 0.05 which will give 95 percent confidence intervals.

Examples

```
#There are various preparation steps required, before you can run this function.
#Please refer to the Introduction vignette to see how to prepare your data
## Not run:
usdata_wide %>%
  sir_byfutime(
        dattype = "seer",
        ybreak_vars = c("race.1", "t_dco.1"),
        xbreak_var = "none",
        futime_breaks = c(0, 1/12, 2/12, 1, 5, 10, Inf),
```

```
          count_var = "count_spc",
          refrates_df = us_refrates_icd2,
          calc_total_row = TRUE,
          calc_total_fu = TRUE,
          region_var = "registry.1",
          age_var = "fc_agegroup.1",
          sex_var = "sex.1",
          year_var = "t_yeardiag.1",
          site_var = "t_site_icd.1", #using grouping by second cancer incidence
          futime_var = "p_futimeyrs",
          alpha = 0.05)

## End(Not run)
```

`standard_population` *Standard Populations*

Description

Dataset that contains different standard populations needed to run some package functions

Usage

```
standard_population
```

Format

A data frame with the following variables:

`standard_pop` Standard Population

`sex` Sex

`age` Age group

`population_n` Absolute Population number in standard population age group

`group_proportion` Proportion of age-group in gender-specific total population

---

`summarize_sir_results` *Summarize detailed SIR results*

Description

Summarize detailed SIR results

Usage

```
summarize_sir_results(
  sir_df,
  summarize_groups,
  summarize_site = FALSE,
  output = "long",
  output_information = "full",
  add_total_row = "no",
  add_total_fu = "no",
  collapse_ci = FALSE,
  shorten_total_cols = FALSE,
  fubreak_var_name = "fu_time",
  ybreak_var_name = "yvar_name",
  xbreak_var_name = "none",
  site_var_name = "t_site",
  alpha = 0.05
)
```

Arguments

sir_df	dataframe with stratified sir results created using the sir or sir_byfutime functions
summarize_groups	option to define summarizing stratified groups. Default is "none". If you want to define variables that should be summarized into one group, you can chose from age, sex, region, year. Define multiple summarize variables e.g. by summarize_groups = c("region", "sex", "year")
summarize_site	If TRUE results will be summarized over all t_site categories. Default is FALSE.
output	Define the format of the output. Can be either "nested" for nested dataframe with fubreak_var and xbreak_var in separate sub_tables (purrr). Or "wide" for wide format where fubreak_var and xbreak_var are appended as columns. Or "long" for long format where sir_df is not reshaped, but just summarized (ybreak_var, xbreak_var and fubreak_var remain in rows). Default is "long".
output_information	option to define information to be presented in final output table. Default is "full" information, i.e. all variables from from sir_df. "reduced" is observed, expected, sir, sir_ci / sir_lci+sir_uci, pyar, n_base. "minimal" is observed, expected, sir, sir_ci. Default is "full".
add_total_row	option to add a row of totals. Can be either "no" for not adding such a row or "start" or "end" for adding it at the first or last row or "only" for only showing totals and no yvar. Default is "no".
add_total_fu	option to add totals for follow-up time. Can be either "no" for not adding such a column or "start" or "end" for adding it at the first or last column or "only" for only showing follow-up time totals. Default is "no".
collapse_ci	If TRUE upper and lower confidence interval will be collapsed into one column separated by "-". Default is FALSE.

shorten_total_cols
: Shorten text in all results columns that start with "Total". Default == FALSE.

fubreak_var_name
: Name of variable with futime stratification. Default is "fu_time".

ybreak_var_name
: Name of variable with futime stratification. Default is "yvar_name".

xbreak_var_name
: Name of variable with futime stratification. Default is "xvar_name".

site_var_name
: Name of variable with site stratification. Default is "t_site".

alpha
: significance level for confidence interval calculations. Default is alpha = 0.05 which will give 95 percent confidence intervals.

Examples

```
#There are various preparation steps required, before you can run this function.
#Please refer to the Introduction vignette to see how to prepare your data
## Not run:
summarize_sir_results(.,
    summarize_groups = c("region", "age", "year", "race"),
    summarize_site = TRUE,
    output = "long",  output_information = "minimal",
    add_total_row = "only",  add_total_fu = "no",
    collapse_ci = FALSE,  shorten_total_cols = TRUE,
    fubreak_var_name = "fu_time", ybreak_var_name = "yvar_name",
    xbreak_var_name = "none", site_var_name = "t_site",
    alpha = 0.05
    )

## End(Not run)
```

---

us_refrates_icd2 *US Reference Rates for Cancer using ICD-O 2digit code for cancer site*

Description

Synthetic dataset of reference incidence rates for the US population to demonstrate package functions

Usage

```
us_refrates_icd2
```

Format

A data frame with the following variables:

`t_site` Tumor Site

`region` Region / Region groups

`year` Year / Periods

`sex` Sex

`age` Age / Age groups

`race` Race

`comment` Comment

`incidence_cases` Incident Cases (raw count)

`incidence_crude_rate` Incidence Rate (crude rate)

`population_pyar` Population Years used for rate calculation (PYAR)

`population_n_per_year` Absolute Population number used for rate calculation (PYAR / 5 years)

`us_second_cancer` *US Second Cancer*

Description

Synthetic dataset of patients with cancer to demonstrate package functions

Usage

```
us_second_cancer
```

Format

A data frame with the following variables:

`fake_id` ID of patient

`SEQ_NUM` Original tumor sequence

`registry` SEER registry

`sex` Biological sex of patient

`race` Race

`datebirth` Date of birth

`t_datediag` Date of diagnosis of tumor

`t_site_icd` Primary site of tumor in ICD-O coding

`t_dco` Tumor diagnosis is based on Death Certificate only

`fc_age` Age at first primary cancer in years

`datedeath` Date of death

`p_alive` Patient alive at end of follow-up 2019

`p_dodmin` Minimum Date of Death if datedeath is missing

`fc_agegroup` Age group of first cancer diagnosis

`t_yeardiag` Time period of diagnosis of tumor

`vital_status` *Calculate vital status at end of follow-up depending on pat_status - tidyverse version*

Description

Calculate vital status at end of follow-up depending on pat_status - tidyverse version

Usage

```
vital_status(
  wide_df,
  status_var = "p_status",
  life_var_new = "p_alive",
  check = TRUE,
  as_labelled_factor = FALSE
)
```

Arguments

`wide_df`	dataframe in wide format
`status_var`	Name of the patient status variable that was previously created. Default is p_status.
`life_var_new`	Name of the newly calculated variable for patient vital status. Default is p_alive.
`check`	Check newly calculated variable life_var_new by printing frequency table. Default is TRUE.
`as_labelled_factor`	If true, output life_var_new as labelled factor variable. Default is FALSE.

Value

wide_df

Examples

```
#load sample data
data("us_second_cancer")

#prep step - make wide data as this is the required format
usdata_wide <- us_second_cancer %>%
                    msSPChelpR::reshape_wide_tidyr(case_id_var = "fake_id",
```

```
                    time_id_var = "SEQ_NUM", timevar_max = 10)

#prep step - calculate p_spc variable
usdata_wide <- usdata_wide %>%
                dplyr::mutate(p_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ "No SPC",
                                                       !is.na(t_site_icd.2)   ~ "SPC developed",
                                                              TRUE ~ NA_character_)) %>%
                 dplyr::mutate(count_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ 1,
                                                            TRUE ~ 0))

#prep step - create patient status variable
usdata_wide <- usdata_wide %>%
                  msSPChelpR::pat_status(., fu_end = "2017-12-31", dattype = "seer",
                                         status_var = "p_status", life_var = "p_alive.1",
                                 birthdat_var = "datebirth.1", lifedat_var = "datedeath.1")

#now we can run the function
msSPChelpR::vital_status(usdata_wide,
                         status_var = "p_status",
                         life_var_new = "p_alive_new",
                         check = TRUE,
                         as_labelled_factor = FALSE)
```

`vital_status_tt` *Calculate vital status at end of follow-up depending on pat_status - tidytable version*

Description

Calculate vital status at end of follow-up depending on pat_status - tidytable version

Usage

```
vital_status_tt(
  wide_df,
  status_var = "p_status",
  life_var_new = "p_alive",
  check = TRUE,
  as_labelled_factor = FALSE
)
```

Arguments

`wide_df`	dataframe or data.table in wide format
`status_var`	Name of the patient status variable that was previously created. Default is p_status.
`life_var_new`	Name of the newly calculated variable for patient vital status. Default is p_alive.

check Check newly calculated variable life_var_new by printing frequency table. Default is TRUE.

as_labelled_factor
If true, output life_var_new as labelled factor variable. Default is FALSE.

Value

wide_df

Examples

```
#load sample data
data("us_second_cancer")

#prep step - make wide data as this is the required format
usdata_wide <- us_second_cancer %>%
                   msSPChelpR::reshape_wide_tidyr(case_id_var = "fake_id",
                   time_id_var = "SEQ_NUM", timevar_max = 10)

#prep step - calculate p_spc variable
usdata_wide <- usdata_wide %>%
                dplyr::mutate(p_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ "No SPC",
                                                !is.na(t_site_icd.2)   ~ "SPC developed",
                                                        TRUE ~ NA_character_)) %>%
                 dplyr::mutate(count_spc = dplyr::case_when(is.na(t_site_icd.2)   ~ 1,
                                                            TRUE ~ 0))

#prep step - create patient status variable
usdata_wide <- usdata_wide %>%
                  msSPChelpR::pat_status(., fu_end = "2017-12-31", dattype = "seer",
                                  status_var = "p_status", life_var = "p_alive.1",
                            birthdat_var = "datebirth.1", lifedat_var = "datedeath.1")

#now we can run the function
msSPChelpR::vital_status_tt(usdata_wide,
                        status_var = "p_status",
                        life_var_new = "p_alive_new",
                        check = TRUE,
                        as_labelled_factor = FALSE)
```

Index