Package ‘msSPChelpR’

January 23, 2024

Title Helper Functions for Second Primary Cancer Analyses

Version 0.9.1

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 cli, dplyr (>= 1.0.0), lubridate, magrittr, purrr, rlang (>= 0.1.2), sjlabelled, stringr, tidyselect, tidytable (>= 0.9.0), tidyr (>= 1.0.0)

Suggests haven, tibble, rmarkdown, knitr, testthat (>= 3.0.0)

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.0

Language en-US

LazyData true

Config/testthat/edition 3

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 2024-01-23 22:40:02 UTC

R topics documented:

---

asir *Calculate age-standardized incidence rates*

Description

Calculate age-standardized incidence rates

Usage

```
asir(
  df,
  dattype = NULL,
  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 NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.
`std_pop`	can be either "ESP2013, ESP1976, WHO1960, WHO2000
`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". Default is "refpop".
`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 = NULL,
  check = TRUE,
  time_unit = "years",
  status_var = "p_status",
  lifedat_var = NULL,
  fcdat_var = NULL,
  spcdat_var = NULL,
  quiet = FALSE
)
```

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`	can be "zfkd" or "seer" or NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.
`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.
`quiet`	If TRUE, warnings and messages will be suppressed. 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::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 = NULL,
  check = TRUE,
  time_unit = "years",
  status_var = "p_status",
  lifedat_var = NULL,
  fcdat_var = NULL,
  spcdat_var = NULL,
  quiet = FALSE
)
```

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`	can be "zfkd" or "seer" or NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.
`check`	Check newly calculated variable "p_futimeyrs" 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.
`quiet`	If TRUE, warnings and messages will be suppressed. Default is FALSE.

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")
```

calc_refrates *Calculate age-, sex-, cohort-, region-specific incidence rates from a cohort*

Description

Calculate age-, sex-, cohort-, region-specific incidence rates from a cohort

Usage

```
calc_refrates(
  df,
  dattype = NULL,
  count_var,
  refpop_df,
  calc_totals = FALSE,
  fill_sites = "no",
  region_var = NULL,
  age_var = NULL,
  sex_var = NULL,
  year_var = NULL,
  race_var = NULL,
  site_var = NULL,
  quiet = FALSE
)
```

Arguments

`df`	dataframe in long format
`dattype`	can be "zfkd" or "seer" or NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.
`count_var`	variable to be counted as observed case. Should be 1 for case to be counted.
`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.
`calc_totals`	option to calculate totals for all age-groups, all sexes, all years, all races, all sites. Default is FALSE.
`fill_sites`	option to fill missing sites in observed with incidence rate of 0. Needs to define the coding system used. Can be either "no" for not filling missing sites. "icd2d" for ICD-O-3 2 digit (C00-C80), "icd3d" for ICD-O-3 3digit, "icd10gm2d" for ICD-10-GM 2-digit (C00-C97), "sitewho" for Site SEER WHO coding (no 1-89 categories), "sitewho_b" for Site SEER WHO B recoding (no. 1-111 categories), "sitewho_epi" for SITE SEER WHO coding with additional sums, "sitewhogen" for SITE WHO coding with less categories to make compatible for international rates, "sitewho_num" for numeric coding of Site SEER WHO coding (no 1-89 categories), "sitewho_b_num" for numeric coding of Site SEER WHO B recoding (no. 1-111 categories), "sitewhogen_num" for numeric international rates, c("manual", char_vector) of sites manually defined
`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, if rates should be calculated stratified by race. If you want to use this option, provide variable name of df that contains race information. If race_var is provided refpop_df needs to contain the variable "race".
`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.
`quiet`	If TRUE, warnings and messages will be suppressed. Default is FALSE.

Value

df

Examples

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

us_second_cancer %>%
  #create variable to indicate to be counted as case
  dplyr::mutate(is_case = 1) %>%
 #calculate refrates - warning: these are not realistic numbers, just showing functionality
  calc_refrates(dattype = "seer", , count_var = "is_case", refpop_df = population_us,
               region_var = "registry", age_var = "fc_agegroup", sex_var = "sex",
               site_var = "t_site_icd")
```

histgroup_iarc *Create variable for groups of malignant neoplasms considered to be histologically 'different' for the purpose of defining multiple tumors, ICD-O-3*

Description

Create variable for groups of malignant neoplasms considered to be histologically 'different' for the purpose of defining multiple tumors, ICD-O-3

Usage

```
histgroup_iarc(df, hist_var, new_var_hist = t_histgroupiarc, version = "3.1")
```

Arguments

`df`	dataframe in long or wide format
`hist_var`	variable in df that contains first 4 digits of tumor histology (without behavior)
`new_var_hist`	Name of the newly calculated variable for histology groups. Default is t_histgroupiarc.
`version`	Version of ICD-O-3 classification used. Can be either "3.0" for 2000 publication, "3.1" for 2013 first revision or "3.2" for 2019 second revision. Default is `version = "3.1"` for ICD-O-3 revision 1, released 2013.

Value

df

Examples

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

us_second_cancer %>%
   msSPChelpR::histgroup_iarc(., hist_var = t_hist) %>%
   dplyr::select(fake_id, t_hist, t_histgroupiarc)
```

---

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 = NULL,
  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 NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.
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 = NULL,
  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 NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.
`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 *Determine patient status at specific end of follow-up - tidyverse version*

Description

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

Usage

```
pat_status(
  wide_df,
  fu_end = NULL,
  dattype = NULL,
  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`	can be "zfkd" or "seer" or NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.

`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` *Determine patient status at specific end of follow-up - tidytable version*

Description

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

Usage

```
pat_status_tt(
  wide_df,
  fu_end,
  dattype = NULL,
  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`	can be "zfkd" or "seer" or NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.
`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 Data*

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 = NULL,
  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`	can be "zfkd" or "seer" or NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.
`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 = NULL,
  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`	can be "zfkd" or "seer" or NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.
`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_long_tt *Reshape dataset to wide format - tidytable version*

Description

Reshape dataset to wide format - tidytable version

Usage

```
reshape_long_tt(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_tt(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 = NULL,
  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,
  expect_missing_refstrata_df = NULL,
  alpha = 0.05,
  quiet = FALSE
)
```

Arguments

`df`	dataframe in wide format
`dattype`	can be "zfkd" or "seer" or NULL. Will set default variable names if dattype is "seer" or "zfkd". Default is NULL.
`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.The variable "race" is additionally required if the option "race_var" is used. 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, if SIR should be calculated stratified by race. If you want to use this option, provide variable name of df that contains race information. If race_var is provided refrates_df needs to contain the variable "race".
`site_var`	variable in df that contains information on site or subsite (e.g. ICD code, SEER site code or others that matches t_site in refrates_df) 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.
`expect_missing_refstrata_df`	optional argument, if strata with missing refrates are expected, because incidence rates of value 0 are not explicit, but missing from refrates_df. It is assumed that expect_missing_refstrata_df is a data.frame 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), and "t_site" for The variable "race" is additionally required if the option "race_var" is used. 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.
`alpha`	significance level for confidence interval calculations. Default is alpha = 0.05 which will give 95 percent confidence intervals.
`quiet`	If TRUE, warnings and messages will be suppressed. Default is FALSE.

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)
```

`sir_ratio` *Calculate Ratio of two SIRs or SMRs*

Description

Calculate ratio of two SIRs by providing observed and expected counts to `sir_ratio` The related functions `sir_ratio_lci` and `sir_ratio_uci` can also calculate lower and upper estimates of the confidence interval Calculations are based on formulas suggested by Breslow & Day 1987

Usage

```
sir_ratio(o1, o2, e1, e2)

sir_ratio_lci(o1, o2, e1, e2, alpha = 0.05)

sir_ratio_uci(o1, o2, e1, e2, alpha = 0.05)
```

Arguments

`o1`	observed count for SIR 1
`o2`	observed count for SIR 2
`e1`	expected count for SIR 1
`e2`	observed count for SIR 2
`alpha`	alpha significance level for confidence interval calculations. Default is alpha = 0.05 which will give 95 percent confidence intervals.

Value

num numeric value of SIR / SMR estimate

References

Breslow NE, Day NE. Statistical Methods in Cancer Research Volume II: The Design and Analysis of Cohort Studies. Lyon, France: IARC; 1987. (IARC Scientific Publications IARC Scientific Publications No. 82). Available from: http://publications.iarc.fr/Book-And-Report-Series/Iarc-Scientific-Publications/Statistical-Methods-In-Cancer-Research-Volume-II-The-Design-And-Analysis-Of-Cohort-Studies-1986

Examples

```
#provide the two expected and observed count to get the ratio of SIRs/SMRs
msSPChelpR::sir_ratio(o1 = 2140, o2 = 3158, e1 = 1993, e2 = 2123)

#calculate lower confidence limit
msSPChelpR::sir_ratio_lci(o1 = 2140, o2 = 3158, e1 = 1993, e2 = 2123, alpha = 0.05)

#calculate upper confidence limit
msSPChelpR::sir_ratio_uci(o1 = 2140, o2 = 3158, e1 = 1993, e2 = 2123, alpha = 0.05)

#functions can be easily used inside dplyr::mutate function
library(dplyr)
test_df <- data.frame(sir_oth = c(1.07, 1.36, 0.96),
                  sir_smo = c(1.49, 1.81, 1.41),
                  observed_oth = c(2140, 748, 1392),
                  expected_oth = c(1993, 550, 1443),
                  observed_smo = c(3158, 744, 2414),
                  expected_smo = c(2123, 412, 1711))

test_df %>%
  mutate(smo_ratio = sir_ratio(observed_oth, observed_smo, expected_oth, expected_smo),
      smo_ratio_lci = sir_ratio_lci(observed_oth, observed_smo, expected_oth, expected_smo),
      smo_ratio_uci = sir_ratio_uci(observed_oth, observed_smo, expected_oth, expected_smo))
```

`standard_population` *Standard Populations Data*

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 Data (ICD-O 2digit code)*

Description

Synthetic dataset of reference incidence rates for the US population to demonstrate package functions Cancer site is coded using ICD-O 2digit code

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 Data*

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_hist` Histology, i.e. ICD-O-3-Code on tumor morphology (4 digits)

`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 — *Determine vital status at end of follow-up depending on pat_status - tidyverse version*

Description

Determine 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 — *Determine vital status at end of follow-up depending on pat_status - tidytable version*

Description

Determine 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