

Package ‘rsurvstat’

September 7, 2026

Title Download Infectious Disease Data from 'SurvStat' (Robert Koch Institute)

Version 0.1.5

Description Provides an interface to the 'SurvStat' web service from the Robert Koch Institute (<<https://tools.rki.de/SurvStat/SurvStatWebService.svc>>) allowing downloads of disease time series stratified by pathogen type and subtype, age, and geography from notifiable disease reports in Germany.

License MIT + file LICENSE

Encoding UTF-8

Suggests knitr, rmarkdown, ggplot2, testthat

VignetteBuilder knitr

Imports dplyr, magrittr, xml2, stringr, tibble, httr, curl, whisker, fs, purrr, tidyr, cli, locfit, rlang, sf

Depends R (>= 4.0)

LazyData true

Language en-GB

LazyDataCompression xz

URL <https://bristol-vaccine-centre.github.io/rsurvstat/index.html>,
<https://github.com/bristol-vaccine-centre/rsurvstat>,
<https://bristol-vaccine-centre.github.io/rsurvstat/>

BugReports <https://github.com/bristol-vaccine-centre/rsurvstat/issues>

Config/Needs/build terminological/pkgtools, robchallen/roxygen2

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Robert Challen [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-5504-7768>>),
Bristol Vaccine Centre [fnd, cph]

Maintainer Robert Challen <rob.challen@bristol.ac.uk>

Repository CRAN

Date/Publication 2026-09-07 18:30:08 UTC

Contents

age_groups	2
BerlinMap	3
cache_clear	3
CountyKey71Map	4
diseases	4
FedStateKey71Map	8
fit_population	8
get_snapshot	10
get_timeseries	12
NutsKey71Map	14
set_cache_settings	14
Index	16

age_groups	SurvStat <i>age group list</i>
------------	--------------------------------

Description

- single_year
- children_coarse: from 0, 15, 20, 25, 30, 40, 50, 60, 70, 80 years
- children_medium: from 0, 5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80 years
- children_fine: from 0, 1, 2, 3, 4, 5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80 years
- five_year: from 0, 1, 5, 10, 15, 20, ... , 75, 80 years
- zero_fifteen: from 0, 15+ years
- zero_fifteen_sixty: from 0, 15, 60+ years
- zero_one_4_20_40_60_80: from 0, 4, 20, 40, 60, 80+ years

Usage

age_groups

References

<https://survstat.rki.de/Content/Query/Create.aspx>

BerlinMap	<i>A Berlin outline sf map</i>
-----------	--------------------------------

Description

A Berlin outline sf map

Usage

```
data(BerlinMap)
```

Format

A sf dataframe containing the following columns:

- Name (character) - the Name column

1 rows

cache_clear	<i>Delete all cached SurvStat requests</i>
-------------	--

Description

This function is only intended to be used interactively. The cache can be controlled with `set_cache_settings()`

Usage

```
cache_clear(confirm = utils::askYesNo("Are you sure?"))
```

Arguments

`confirm` can be set to TRUE to make function non interactive.

Value

nothing. called for side effects

Examples

```
#' # Disable the cache for CRAN
set_cache_settings(active = FALSE)

cache_clear(confirm = interactive())
```

CountyKey71Map	<i>The CountyKey71Map dataset</i>
----------------	-----------------------------------

Description

This matches the CountyKey71 dimension in SurvStat. This is the 400 Stadtkreis and Landkreise administrative regions in Germany, plus 12 Berlin boroughs (Bezirke) which replace the Berlin Kriese (Id: 11000). The boroughs have sequential Ids from [11001] to [11012]

Usage

```
data(CountyKey71Map)
```

Format

A sf dataframe containing the following columns:

- Id - the full SurvStat identifier for this region (includes hierarchical information)
- ComponentId - the id of the most granular geographical unit (which can be used to link out to other data sets)
- HierarchyId - the id of the geographical unit type
- Name - the name of the region

Any grouping allowed.

411 rows

diseases	<i>SurvStat disease list</i>
----------	------------------------------

Description

Supported diseases:

- Acinetobacter (key: Acinetobacter-Infektion oder –Kolonisation)
- Adenovirus (key: Adenovirus (andere Form, Meldepflichtig gemäß Landesmeldeverordnung))
- Amoebiasis (key: Amoebiasis)
- Anthrax (key: Milzbrand)
- Arbovirus (key: Arbovirus-Erkrankung)
- Astrovirus (key: Astrovirus-Infektion)
- Bornavirus (key: Bornavirus)
- Botulism (key: Botulismus)
- Brucellosis (key: Brucellose)

- CJD (key: CJK)
- CJD, variant (key: vCJK)
- COVID-19 (key: COVID-19)
- Campylobacter (key: Campylobacter-Enteritis)
- Candida auris (invasive) (key: Candida auris, invasive Infektion)
- Chickenpox (key: Windpocken)
- Chickenpox (state) (key: Windpocken (Meldepflicht gemäß Landesmeldeverordnung))
- Chikungunya (key: Chikungunya-Fieber)
- Chlamydia Trachomatis (key: Chlamydia-trachomatis-Infektion)
- Cholera (key: Cholera)
- Clostridium difficile / mild (key: Clostridium difficile, nicht schwerer Verlauf)
- Clostridium difficile / moderate (key: Clostridium difficile, schwerer Verlauf)
- Cryptosporidiosis (key: Kryptosporidiose)
- Cytomegalovirus (key: Cytomegalie)
- Dengue (key: Denguefieber)
- Diphtheria (key: Diphtherie)
- E. Coli, enteritis (key: E.-coli-Enteritis)
- E. Coli, enterohemorrhagic (key: EHEC-Erkrankung)
- Ebola (key: Ebolafieber)
- Echinococcosis (key: Echinokokkose)
- Enterobacteria colonisation (key: Enterobacteriaceae-Infektion oder –Kolonisation)
- Enterovirus (key: Enterovirus)
- Gas gangrene (key: Gasbrand)
- Gastroenteritis (other) (key: Weitere bedrohliche Krankheit (gastro))
- Giardia (key: Giardiasis)
- Gonorrhoea (key: Gonorrhoe)
- Group B Streptococcus (key: Gruppe-B-Streptokokken)
- HIV (key: HIV-Infektion)
- Haemolytic-uraemic syndrome (key: HUS (Hämolytisch-urämisches Syndrom), enteropathisch)
- Haemophilus influenza, invasive (key: Haemophilus influenzae, invasive Erkrankung)
- Hand foot mouth disease (key: Hand-Fuß-Mund-Krankheit)
- Hantavirus (key: Hantavirus-Erkrankung)
- Head lice (key: Kopflausbefall)
- Hepatitis (general) (key: Hepatitis (allgemein))
- Hepatitis A (key: Hepatitis A)
- Hepatitis B (key: Hepatitis B)
- Hepatitis C (key: Hepatitis C)

- Hepatitis D (key: Hepatitis D)
- Hepatitis E (key: Hepatitis E)
- Hepatitis non A-E (key: Hepatitis Non A-E)
- Herpes Zoster (key: Herpes Zoster)
- Influenza, seasonal (key: Influenza, saisonal)
- Influenza, zoonotic (key: Influenza, zoonotisch)
- Keratoconjunctivitis (IfSG) (key: Keratokunjunktivitis (Meldepflicht gemäß IfSG))
- Keratoconjunctivitis (state) (key: Keratokunjunktivitis (Meldepflicht gemäß Landesmeldeverordnung))
- Lassa fever (key: Lassafieber)
- Legionella (key: Legionellose)
- Leprosy (key: Lepra)
- Leptospirosis (key: Leptospirose)
- Listeriosis (key: Listeriose)
- Lyme Disease (key: Borreliose)
- MERS (key: Middle East Respiratory Syndrome)
- MRSA, invasive (key: MRSA, invasive Infektion)
- Malaria (IfSG) (key: Malaria (§7(3) IfSG))
- Malaria (state) (key: Malaria, Länderverordnung)
- Marburg virus (key: Marburgfieber)
- Measles (key: Masern)
- Meningitis (other) (key: Meningitis, andere)
- Meningococcal, invasive (key: Meningokokken, invasive Erkrankung)
- Mpox (key: Affenpocken)
- Mpox (key: Affenpocken)
- Mumps (IfSG) (key: Mumps (Meldepflicht gemäß IfSG))
- Mumps (state) (key: Mumps (Meldepflicht gemäß Landesmeldeverordnung))
- Mycoplasma (key: Mycoplasma)
- Norovirus (key: Norovirus-Gastroenteritis)
- Orthinovirus (key: Ornithose)
- Orthopox (key: Orthopocken)
- Parainfluenza (key: Parainfluenza)
- Paratyphus (key: Paratyphus)
- Plague (key: Pest)
- Pneumococcus (IfSG) (key: Pneumokokken (Meldepflicht gemäß IfSG))
- Pneumococcus (state) (key: Pneumokokken (Meldepflicht gemäß Landesverordnung))
- Poliomyelitis (key: Poliomyelitis)

- Q-fever (key: Q-Fieber)
- RSV (IfSG) (key: RSV (Meldepflicht gemäß IfSG))
- RSV (state) (key: RSV (Meldepflicht gemäß Landesmeldeverordnung))
- Rabies (confirmed) (key: Tollwut)
- Rabies (suspected) (key: Tollwutexpositionsverdacht)
- Relapsing fever (key: Läuserückfallfieber)
- Ringworm (key: Ringelröteln)
- Rotavirus gastroenteritis (key: Rotavirus-Gastroenteritis)
- Rubella (key: Röteln, postnatal)
- Rubella (state) (key: Röteln (Meldepflicht gemäß Landesmeldeverordnung))
- Rubella, congenital (key: Röteln, konnatal)
- SARS (key: SARS)
- Salmonellosis (key: Salmonellose)
- Scabies (key: Krätzmilbenbefall)
- Scarlet fever (key: Scharlach)
- Sepsis (other) (key: Weitere bedrohliche Krankheit)
- Shigellosis (key: Shigellose)
- Smallpox (key: Pocken)
- Subacute Sclerosing Panencephalitis (key: Subakute Sklerosierende Panenzephalitis)
- Syphilis (key: Syphilis)
- Tetanus (key: Tetanus)
- Tick borne encephalitis (key: FSME (Frühsommer-Meningoenzephalitis))
- Toxoplasmosis (key: Toxoplasmose)
- Toxoplasmosis, congenital (key: Toxoplasmose, konnatal)
- Trichinellosis (key: Trichinellose)
- Tuberculosis (key: Tuberkulose)
- Tulareamia (key: Tularämie)
- Typhoid (key: Fleckfieber)
- Typhoid, abdominal (key: Typhus abdominalis)
- Typhus/Paratyphus (key: Typhus/Paratyphus)
- Varicella, congenital (key: Fetales (kongenitales) Varizellensyndrom)
- Vibria (key: Vibrionen)
- Viral haemorrhagic fever (key: Virale hämorrhagische Fieber)
- West Nile Virus (key: West-Nil-Virus)
- Whooping cough (IfSG) (key: Keuchhusten (Meldepflicht gemäß IfSG))
- Whooping cough (state) (key: Keuchhusten (Meldepflicht gemäß Landesmeldeverordnung))
- Yellow fever (key: Gelbfieber)
- Yersinia (key: Yersiniose)
- Zika (key: Zikavirus-Erkrankung)

Usage

```
diseases
```

References

<https://survstat.rki.de/Content/Query/Create.aspx>

FedStateKey71Map	<i>The FedStateKey71Map dataset.</i>
------------------	--------------------------------------

Description

This matches the FedStateKey71 dimension in SurvStat. This is the 16 federal states in Germany.

Usage

```
data(FedStateKey71Map)
```

Format

A sf dataframe containing the following columns:

- Id - the full SurvStat identifier for this region (includes hierarchical information)
- ComponentId - the id of the most granular geographical unit (which can be used to link out to other data sets)
- HierarchyId - the id of the geographical unit type
- Name - the name of the region

16 rows

fit_population	<i>Infer and fit a population model from SurvStat output</i>
----------------	--

Description

SurvStat can be queried for count or incidence. From the combination of these metrics queried across the whole range of disease notifications for any given year we can infer a stratified population size, that SurvStat is using to calculate it's incidence. This is simply modelled with a local polynomial over time to allow us to fill in weekly population denominators.

Usage

```
fit_population(count_df, .progress = TRUE)

infer_population(
  age_group = NULL,
  geography = NULL,
  years = NULL,
  .progress = TRUE
)
```

Arguments

count_df	a dataframe from the output of <code>get_timeseries()</code> or <code>get_snapshot()</code>
.progress	by default a progress bar is shown, which may be important if many downloads are needed to fulfil the request. It can be disabled by setting this to <code>FALSE</code> here.
age_group	(optional) the age group of interest as a <code>SurvStat</code> key, see <code>rsurvstat::age_groups</code> for a list of valid options.
geography	(optional) one of "state", "nuts", or "county" to define the resolution of the query. Does not accept a <code>sf</code> map or subset of (unlike <code>get_timeseries()</code>).
years	(optional) a vector of years to limit the response to. This may be useful to limit the size of returned pages in the event the <code>SurvStat</code> service hits a data transfer limit.

Value

the count_df dataframe with an additional population column
 a dataframe with geography, age grouping, year and population columns

Functions

- `infer_population()`: Query `SurvStat` for data to impute a population denominator

Examples

```
# Disable the cache for CRAN
rsurvstat::set_cache_settings(active = FALSE)

# snapshot:
get_snapshot(
  disease = diseases$`COVID-19`,
  geography = "state",
  season = 2024
) %>%
  fit_population() %>%
  dplyr::glimpse()

# timeseries
# A weekly population estimate is inferred from the yearly data:
```

```

get_timeseries(
  diseases$`COVID-19`,
  measure = "Count",
  age_group = age_groups$children_coarse
) %>%
  fit_population() %>%
  dplyr::glimpse()

# Disable the cache for CRAN
rsurvstat::set_cache_settings(active = FALSE)

infer_population(years = 2020:2025) %>% dplyr::glimpse()

```

get_snapshot	<i>Retrieve data from the SurvStat web service relating to a single time period.</i>
--------------	--

Description

This function gets a snapshot of disease count or incidence data from the Robert Koch Institute SurvStat web service, based on either whole epidemiological season or an individual week within a season. Seasons are whole years starting either at the beginning of the calendar year, at week 27 or at week 40.

Usage

```

get_snapshot(
  disease = NULL,
  measure = c("Count", "Incidence"),
  ...,
  season,
  season_week = NULL,
  season_start = 1,
  age_group = NULL,
  age_range = c(0, Inf),
  disease_subtype = FALSE,
  geography = NULL,
  .progress = TRUE
)

```

Arguments

disease	the disease of interest as a SurvStat key, see <code>rsurvstat::diseases</code> for a current list of these. This is technically optional, and if omitted the counts of all diseases will be returned. Keys are the same as the options in the SurvStat user
---------	--

	interface found here . IfSG and state variants of diseases are counts that are reported directly to the Robert Koch Institute or indirectly via state departments.
measure	one of "Count" (default) or "Incidence" per 100,000 per week or year depending on the context.
...	not used, must be empty.
season	the start year of the season in which the snapshot is taken
season_week	the start week within the season of the snapshot. If missing then the whole season is used
season_start	the week of the calendar year in which the season starts this can be one of 1, 27 or 40.
age_group	(optional) the age group of interest as a SurvStat key, see <code>rsurvstat::age_groups</code> for a list of valid options.
age_range	(optional) a length 2 vector with the minimum and maximum ages to consider
disease_subtype	if TRUE the returned count will be broken down by disease or pathogen subtype (assuming disease was provided).
geography	(optional) a geographical breakdown. This can be given as a character where it must be one of state, nuts, or county specifying the 16 region FedStateKey71Map, 38 region NutsKey71Map, or 411 region CountyKey71Map data respectively. Alternatively it can be given as a sf dataframe, subsetting one of these maps, in which case only that subset of regions will be returned.
.progress	by default a progress bar is shown, which may be important if many downloads are needed to fulfil the request. It can be disabled by setting this to FALSE here.

Details

The snapshot can be stratified by any combination of age, geography, disease, disease subtype. Queries to SurvStat are cached and paged, but obviously multidimensional extracts have the potential to need a lot of downloading.

Value

a data frame with at least year (the start of the epidemiological season) and start_week (the calendar week in which the epidemiological season starts), and one of count or incidence columns. Most likely it will also have disease_name and disease_code columns, and some of age_name, age_code, age_low, age_high, geo_code, geo_name, disease_subtype_code, disease_subtype_name depending on options.

Examples

```
# Disable the cache for CRAN
rsurvstat::set_cache_settings(active=FALSE)

get_snapshot(
  diseases$`COVID-19`,
  measure = "Count",
```

```

    season = 2024,
    age_group = age_groups$children_coarse
  )

  get_snapshot(
    diseases$`COVID-19`,
    measure = "Count",
    age_group = age_groups$children_coarse,
    season = 2024,
    geography = rsurvstat::FedStateKey71Map[1:10,]
  )

```

get_timeseries

Retrieve time series data from the SurvStat web service.

Description

This function gets a weekly timeseries of disease count or incidence data from the Robert Koch Institute SurvStat web service. The timeseries can be stratified by any combination of age, geography, disease, disease subtype. Queries to SurvStat are cached and paged, but obviously multidimensional extracts have the potential to need a lot of downloading.

Usage

```

get_timeseries(
  disease = NULL,
  measure = c("Count", "Incidence"),
  ...,
  age_group = NULL,
  age_range = c(0, Inf),
  disease_subtype = FALSE,
  years = NULL,
  geography = NULL,
  trim_zeros = c("leading", "both", "none"),
  .progress = TRUE
)

```

Arguments

disease	the disease of interest as a SurvStat key, see <code>rsurvstat::diseases</code> for a current list of these. This is technically optional, and if omitted the counts of all diseases will be returned. Keys are the same as the options in the SurvStat user interface found here . IfSG and state variants of diseases are counts that are reported directly to the Robert Koch Institute or indirectly via state departments.
measure	one of "Count" (default) or "Incidence" per 100,000 per week or year depending on the context.
...	not used, must be empty.

age_group	(optional) the age group of interest as a SurvStat key, see <code>rsurvstat::age_groups</code> for a list of valid options.
age_range	(optional) a length 2 vector with the minimum and maximum ages to consider
disease_subtype	if TRUE the returned count will be broken down by disease or pathogen subtype (assuming disease was provided).
years	(optional) a vector of years to limit the response to. This may be useful to limit the size of returned pages in the event the SurvStat service hits a data transfer limit.
geography	(optional) a geographical breakdown. This can be given as a character where it must be one of state, nuts, or county specifying the 16 region FedStateKey71Map, 38 region NutsKey71Map, or 411 region CountyKey71Map data respectively. Alternatively it can be given as a sf dataframe, subsetting one of these maps, in which case only that subset of regions will be returned.
trim_zeros	get rid of zero counts. Either "both" (from start and end), "leading" (from start only - the default) or "none".
.progress	by default a progress bar is shown, which may be important if many downloads are needed to fulfil the request. It can be disabled by setting this to FALSE here.

Value

a data frame with at least date (weekly), and one of count or incidence columns. Most likely it will also have disease_name and disease_code columns, and some of age_name, age_code, age_low, age_high, geo_code, geo_name, disease_subtype_code, disease_subtype_name depending on options. The dataframe will be grouped to make sure each group contains a single timeseries.

Examples

```
# Disable the cache for CRAN
rsurvstat::set_cache_settings(active=FALSE)

# age stratified
get_timeseries(
  diseases$`COVID-19`,
  measure = "Count",
  age_group = age_groups$children_coarse
) %>% dplyr::glimpse()

# geographic
get_timeseries(
  diseases$`COVID-19`,
  measure = "Count",
  geography = "state"
) %>% dplyr::glimpse()

# disease stratified, subset of years:
get_timeseries(
  measure = "Count",
```

```
years = 2024
) %>% dplyr::glimpse()
```

NutsKey71Map	<i>The NutsKey71Map dataset</i>
--------------	---------------------------------

Description

This matches the NutsKey71 dimension in SurvStat. This is the 38 NUTS2 level administrative regions in Germany.

Usage

```
data(NutsKey71Map)
```

Format

A sf dataframe containing the following columns:

- Id - the full SurvStat identifier for this region (includes hierarchical information)
- ComponentId - the id of the most granular geographical unit (which can be used to link out to other data sets)
- HierarchyId - the id of the geographical unit type
- Name - the name of the region

38 rows

set_cache_settings	<i>Set options for the rsurvstat cache</i>
--------------------	--

Description

By default successful requests to SurvStat are cached for 7 days to prevent repeated querying of the service. This is stored in the usual R package cache location by default (e.g. "~/ .cache/ rsurvstat " on mac / linux). Caching can be switched off altogether.

Usage

```
set_cache_settings(..., active = NULL, dir = NULL, stale = NULL)
```

Arguments

...	you can also submit the settings as a named list.
active	boolean (optional), set to FALSE to disable caching
dir	file path (optional), the location of the cache
stale	numeric (optional), the number of days before a cached item is considered out of date

Value

the old cache settings as a list

Examples

```
old_settings = set_cache_settings(active = FALSE)
set_cache_settings(old_settings)
```

Index

- * **cache**
 - cache_clear, [3](#)
 - set_cache_settings, [14](#)
- * **data**
 - age_groups, [2](#)
 - BerlinMap, [3](#)
 - CountyKey71Map, [4](#)
 - diseases, [4](#)
 - FedStateKey71Map, [8](#)
 - NutsKey71Map, [14](#)
- * **survstat**
 - fit_population, [8](#)
 - get_snapshot, [10](#)
 - get_timeseries, [12](#)
- age_groups, [2](#)
- BerlinMap, [3](#)
- cache_clear, [3](#)
- CountyKey71Map, [4](#)
- diseases, [4](#)
- FedStateKey71Map, [8](#)
- fit_population, [8](#)
- get_snapshot, [10](#)
- get_timeseries, [12](#)
- infer_population (fit_population), [8](#)
- NutsKey71Map, [14](#)
- set_cache_settings, [14](#)