

Package: csdata (via r-universe)

September 26, 2026

Title Structural Data for Norway

Version 2026.8.21

Description Datasets relating to population in municipalities,
municipality/county matching, and how different municipalities
have merged/redistricted over time from 2006 to 2024.

URL <https://niphr.github.io/csdata/>, <https://github.com/niphr/csdata>

BugReports <https://github.com/niphr/csdata/issues>

Depends R (>= 3.5.0)

License MIT + file LICENSE

Encoding UTF-8

Imports data.table, stats, utils

Suggests testthat, broom, PxWebApiData, crayon, dplyr, forcats, fs,
geojsonio, ggplot2, glue, gt, knitr, lubridate, mapproj,
methods, ncdf4, purrr, readxl, reshape2, rmarkdown, rmapshaper,
rstudioapi, stringr, sp, sf, tidyr, zoo

Roxygen list(markdown = TRUE)

VignetteBuilder knitr

Config/Needs/website niphr/cstemplate

Config/roxygen2/version 8.0.0

Config/testthat/edition 3

Repository <https://niphr.r-universe.dev>

Date/Publication 2026-08-26 21:51:06 UTC

RemoteUrl <https://github.com/niphr/csdata>

RemoteRef HEAD

RemoteSha 516bb47576efee0f687c9f3e44cc885138914c7b

Contents

add_granularity_geo_to_data_set	2
add_iso3_to_data_set	3
config	4
location_code_to_granularity_geo	5
location_code_to_iso3	6
nb	7
nor_locations_hierarchy_from_to	7
nor_locations_names	9
nor_locations_redistricting	10
nor_population_by_age_cats	11
nor_population_by_sex_age_cats	12
se	14
set_config	15
Index	16

add_granularity_geo_to_data_set

Add a granularity_geo column to a data set

Description

Derives the geographic granularity label from the location_code column. Adds the label as a new granularity_geo column, and changes x in place. When location_reference is NULL, the granularity comes from the location code prefix. When you supply a reference table, the granularity comes from that table.

Usage

```
add_granularity_geo_to_data_set(x, location_reference = NULL)
```

Arguments

- x A data.table with a column named location_code. Only a data.table method exists, so any other class raises "no applicable method".
- location_reference A data.table with columns location_code and granularity_geo to use for lookup. When NULL (default), granularity comes from the location code prefix (e.g. "county_nor03" -> "county"). Its location_code values SHOULD be unique. A code that appears twice in the reference produces more labels than x has rows. The assignment then fails with "Supplied N items to be assigned to M items of column 'granularity_geo'". `nor_locations_names()` holds one such code today, "lab_nor084467", which two laboratories share.

Value

x, invisibly, with the granularity_geo column added or updated.

See Also

`vignette("csdata", package = "csdata")`, which calls this function on a worked example. [location_code_to_granularity_geo_to_data_set\(\)](#) returns the same labels as a vector instead of writing them onto `x`.

Other data set column adders: [add_iso3_to_data_set\(\)](#)

Examples

```
library(data.table)
data <- data.table(location_code = c("nation_nor", "county_nor03", "blah"))
csdata::add_granularity_geo_to_data_set(data)
print(data)

library(data.table)
data <- data.table(location_code = c("nation_nor", "county_nor03", "blah"))
csdata::add_granularity_geo_to_data_set(data, location_reference = csdata::nor_locations_names())
print(data)
```

`add_iso3_to_data_set` *Add an iso3 column to a data set*

Description

Adds an `iso3` column that holds the ISO 3166-1 alpha-3 country code, and changes `x` in place. `csdata` carries Norwegian data only, so the value is "nor" on every row. The function does not read the contents of `x`, so `x` gains the column even when it has no `location_code` column.

Usage

```
add_iso3_to_data_set(x)
```

Arguments

`x` A `data.table`. Only a `data.table` method exists, so any other class raises "no applicable method".

Value

`x`, invisibly, with a new `iso3` column holding "nor" on every row.

See Also

No vignette covers this function. [location_code_to_iso3\(\)](#) returns the same values as a vector instead of writing them onto `x`.

Other data set column adders: [add_granularity_geo_to_data_set\(\)](#)

Examples

```
library(data.table)
data <- data.table(location_code = c("nation_nor", "county_nor03", "blah"))
csdata::add_iso3_to_data_set(data)
print(data)
```

config

Package configuration environment

Description

An environment that stores package-level configuration variables. Change it with [set_config\(\)](#). Read it directly, for example `config$border_nor`.

Usage

```
config
```

Details

Available variables:

- `border_nor` (default 2024): the border year that selects which Norwegian geographic datasets a function reads. Valid values: 2024.

`border_nor` supplies the default border argument of [nor_locations_names\(\)](#), [nor_locations_redistricting\(\)](#), [nor_locations_hierarchy_from_to\(\)](#), [nor_population_by_age_cats\(\)](#) and [nor_population_by_sex_age_cats\(\)](#). No other exported function reads it.

See Also

No vignette covers this object.

Examples

```
print(ls(csdata::config))
for(i in names(csdata::config)){
  cat(i, ":", csdata::config[[i]], "\n")
}
```

location_code_to_granularity_geo

Convert location codes to granularity_geo

Description

Extracts the geographic granularity label from one or more location codes. When `location_reference` is `NULL`, the granularity comes from the lowercase alphabetic prefix of the location code (e.g. "county_nor03" -> "county"). The prefix "norge" is the one special case, and it gives "nation". When you supply a reference table, the granularity comes from that table.

Usage

```
location_code_to_granularity_geo(x, location_reference = NULL)
```

Arguments

`x` A character vector of location codes, or a `data.table` with a column named `location_code`. A plain `data.frame` is not supported: it falls through to the default method, which treats the frame itself as the vector of codes.

`location_reference` A `data.table` with columns `location_code` and `granularity_geo` to use for lookup. When `NULL` (default), granularity comes from the location code prefix.

Value

A character vector of the corresponding `granularity_geo` values. It has the same length as `x`, or as many elements as `x` has rows when `x` is a `data.table`. A code with no leading lowercase letters yields `NA`. An unmatched code also yields `NA` when you supply `location_reference`.

The length guarantee has one exception. `location_reference` is joined on `location_code`, so a code that appears more than once in the reference contributes one element per matching row. [nor_locations_names\(\)](#) holds one such code today, "lab_nor084467", which two laboratories share.

See Also

`vignette("csdata", package = "csdata")`, which calls this function on a worked example. `vignette("locations_norw", package = "csdata")` tabulates the `location_code` values returned by [nor_locations_names\(\)](#).

Other location code converters: [location_code_to_iso3\(\)](#)

Examples

```
csdata::location_code_to_granularity_geo(c("nation_nor", "county_nor03", "municip_nor0301"))

# a code with no lowercase prefix gives NA
csdata::location_code_to_granularity_geo(c("nation_nor", "", NA))
```

```
library(data.table)
dt <- data.table(location_code = c("nation_nor", "county_nor03"))
csdata::location_code_to_granularity_geo(dt)

# looked up against a reference table instead of parsed from the prefix
csdata::location_code_to_granularity_geo(
  c("nation_nor", "county_nor03", "blah"),
  location_reference = csdata::nor_locations_names()
)
```

location_code_to_iso3 *Convert location codes to ISO 3166-1 alpha-3 country codes*

Description

Returns the ISO 3166-1 alpha-3 country code for each location code. `csdata` carries Norwegian data only, so the implementation returns "nor" for every element without inspecting its value. A code from another country, or a string that is not a location code at all, also returns "nor".

Usage

```
location_code_to_iso3(x)
```

Arguments

x A character vector of location codes, or a `data.table` with a column named `location_code`. A plain `data.frame` is not supported: it falls through to the default method, which returns one value per column rather than one per row.

Value

A character vector of "nor". It has the same length as `x`, or as many elements as `x` has rows when `x` is a `data.table`.

See Also

No vignette covers this function. `vignette("locations_norway", package = "csdata")` tabulates the `location_code` values returned by `nor_locations_names()`.

Other location code converters: `location_code_to_granularity_geo()`

Examples

```
csdata::location_code_to_iso3(c("nation_nor", "county_nor03", "municip_nor0301"))

# the input value is not inspected
csdata::location_code_to_iso3(c("county_nor03", "xyz"))

library(data.table)
dt <- data.table(location_code = c("nation_nor", "county_nor03"))
csdata::location_code_to_iso3(dt)
```

nb	<i>Norwegian characters in unicode</i>
----	--

Description

A named list of Norwegian special characters as unicode strings, for use where literal non-ASCII characters are inconvenient. Elements: AA/aa (Aa/aa), OE/oe (Oe/oe), AE/ae (Ae/ae). Each element is a single character.

Usage

```
nb
```

See Also

No vignette covers this object.

Other unicode character lists: [se](#)

Examples

```
print(csdata::nb)
csdata::nb$AA # uppercase Aa
```

nor_locations_hierarchy_from_to	<i>Location hierarchies in Norway</i>
---------------------------------	---------------------------------------

Description

Returns a mapping between two geographic levels in Norway. Both from and to accept character vectors. The function then returns every requested combination, joined into a single data.table.

Usage

```
nor_locations_hierarchy_from_to(
  from,
  to,
  include_to_name = FALSE,
  border = csdata::config$border_nor
)
```

Arguments

<code>from</code>	Character vector. The source geographic granularity. One or more of: "wardoslo", "extrawardoslo", "wardbergen", "wardtrondheim", "wardstavanger", "missingwardoslo", "missingwardbergen", "missingwardtrondheim", "missingwardstavanger", "municip", "baregion", "county", "georegion", "mtregion", "notmainlandmunicip", "notmainlandcounty", "missingmunicip", "missingcounty".
<code>to</code>	Character vector. The target geographic granularity. Same valid values as <code>from</code> .
<code>include_to_name</code>	Logical. If TRUE, include the name of each to location as a third column <code>to_name</code> . Default FALSE.
<code>border</code>	Integer. The geographic border year. Valid values: 2024. Defaults to <code>csdata::config\$border_nor</code> .

Details

A combination that the bundled hierarchy table cannot express returns a zero-row `data.table` rather than an error. A `from` or `to` value of "baregion" always does this, because the bundled table carries no BA-region codes. Use `nor_locations_names()` to list the BA-regions themselves.

Value

A `data.table` with columns:

from_code Location code at the from granularity level.

to_code Location code at the to granularity level.

to_name Name of the to location (only present when `include_to_name = TRUE`).

The table has no key set, and a combination the bundled hierarchy cannot express yields zero rows.

See Also

`vignette("csdata", package = "csdata")`, which calls `nor_locations_hierarchy_from_to()` on a worked example. `vignette("locations_norway", package = "csdata")` tabulates the `location_code` values returned by `nor_locations_names()`, which are the values `from_code` and `to_code` are drawn from.

Examples

```
csdata::nor_locations_hierarchy_from_to(from = "wardoslo", to = "county")

nrow(csdata::nor_locations_hierarchy_from_to(from = "municip", to = "county"))

csdata::nor_locations_hierarchy_from_to(
  from = c("municip", "county"),
  to = "georegion",
  include_to_name = TRUE
)[1:5]

# baregion is accepted but the bundled hierarchy holds no BA-region codes
nrow(csdata::nor_locations_hierarchy_from_to(from = "municip", to = "baregion"))
```

`nor_locations_names` *Location codes and names for Norwegian geographic units*

Description

Returns a `data.table` of Norwegian geographic units with their location codes, display names, and presentation metadata. Coverage includes nation, counties, municipalities, city districts (Oslo, Bergen, Stavanger, Trondheim), BA-regions, and lab regions, plus placeholder codes for not-mainland and unknown locations. The returned table has no key set.

Usage

```
nor_locations_names(border = csdata::config$border_nor)
```

Arguments

border Integer. The geographic border year that selects which administrative boundaries the function uses. Valid values: 2024. Defaults to `csdata::config$border_nor`.

Value

A `data.table` with one row per geographic unit and the columns:

location_code Location code (e.g. "nation_nor", "county_nor03").

location_name Full location name.

location_name_short Abbreviated name, NA on most rows. The nation and the 5 georegions carry one letter. The 15 counties carry three letters. The 15 Oslo city districts and the 2 extra Oslo wards carry a display name of 4 to 13 characters. On some rows that display name is identical to `location_name`. It is NA for municipalities, BA-regions, lab regions, and the Bergen, Stavanger and Trondheim city districts.

location_name_description_nb Location name with a parenthetical description of geographic level (Norwegian Bokmal).

location_name_file_nb_utf Name suitable for use in file names, retaining Norwegian characters.

location_name_file_nb_ascii Name suitable for use in file names, with Norwegian characters replaced by ASCII equivalents.

location_order Integer giving the preferred presentation order.

granularity_geo Geographic granularity: one of "nation", "georegion", "mtregion", "county", "municip", "baregion", "wardoslo", "wardbergen", "wardstavanger", "wardtrondheim", "extrawardoslo", "lab", "notmainlandcounty", "notmainlandmunicip", "missingcounty", "missingmunicip", "missingwardoslo", "missingwardbergen", "missingwardstavanger", "missingwardtrondheim".

Source

https://no.wikipedia.org/wiki/Liste_over_norske_kommunenummer

See Also

`vignette("locations_norway", package = "csdata")` prints the `location_code` and `location_name_description_nb` columns of this table in full. `vignette("csdata", package = "csdata")` summarises the table by `granularity_geo` and states which name column to use where.

Examples

```
d <- nor_locations_names()
head(d)
d[granularity_geo == "county"]
```

`nor_locations_redistricting`

Redistricting weights for Norwegian geographic units

Description

Returns a `data.table` of weighting factors used to convert historical data recorded under old administrative boundaries to the 2024 borders. Each row maps an original location code (as of a given calendar year) to the current location code, with a proportional weighting.

Usage

```
nor_locations_redistricting(border = csdata::config$border_nor)
```

Arguments

border Integer. The target geographic border year. Valid values: 2024. Defaults to `csdata::config$border_nor`.

Value

A `data.table` with columns:

location_code_current Location code under the target border year. Each of the 421 distinct values also appears in `nor_locations_names()`.

location_code_original Location code as it existed in calyear. Of the 941 distinct values, 520 are retired codes that `nor_locations_names()` no longer lists.

calyear The calendar year to which `location_code_original` applies.

weighting Proportional weight to apply when you aggregate from the original location to the current location. Values sum to 1 within each `location_code_original` / `calyear` group, on all 40,819 of those groups. A row spread across several current locations is therefore split, not duplicated. Values need not sum to 1 within a `location_code_current` / `calyear` group: 4,334 of those 35,160 groups do not.

granularity_geo Geographic granularity: one of "nation", "county", "municip", "wardbergen", "wardoslo", "wardstavanger", "wardtrondheim", "extrawardoslo", "missingwardbergen", "missingwardoslo", "missingwardstavanger", "missingwardtrondheim", "notmainlandcounty", "notmainlandmunicip", "missingcounty", "missingmunicip".

Source

Statistics Norway (SSB) municipal reform documentation.

See Also

No vignette covers this function. `vignette("locations_norway", package = "csdata")` tabulates the `location_code` values returned by `nor_locations_names()`, which are the values `location_code_current` is drawn from.

Examples

```
d <- csdata::nor_locations_redistricting()
head(d)
d[calyear == 2019 & granularity_geo == "municip"]
```

`nor_population_by_age_cats`

Norwegian population aggregated into custom age categories

Description

Aggregates the bundled Norwegian population dataset (from Statistics Norway) into caller-defined age bands. The underlying data covers every integer age from 0 to 105 at national, georegion, county, municipality, and city-district level. Not-mainland and unknown locations carry the single placeholder age -99 instead. `include_total` counts that age, and a band such as `0:105` does not.

Usage

```
nor_population_by_age_cats(
  cats = NULL,
  include_total = TRUE,
  include_9999 = FALSE,
  border = csdata::config$border_nor
)
```

Arguments

<code>cats</code>	A named or unnamed list of integer vectors that give the age values to include in each category. Each vector element defines one age band. If a list element is named, that name becomes the age label. If it is not named, the label is "LLL_HHH": the zero-padded lower and upper bounds. Defaults to <code>NULL</code> (no custom bands; only "total" is returned when <code>include_total = TRUE</code>).
<code>include_total</code>	Logical. If <code>TRUE</code> (default), the function appends an extra row group with age = "total" that covers all ages.
<code>include_9999</code>	Logical. If <code>TRUE</code> , the function duplicates the most recent calendar year and adds it with <code>calyear = 9999</code> . This follows the <code>cs tidy</code> convention for <code>granularity_time == "event_*</code> ". Default <code>FALSE</code> .
<code>border</code>	Integer. The geographic border year. Valid values: 2024. Defaults to <code>csdata::config\$border_nor</code> .

Value

A data.table with columns:

granularity_geo Geographic granularity level.

location_code Location code.

age Age category label, as derived from cats names or auto-generated, plus "total" if include_total = TRUE.

sex Sex. Always "total", because this function filters the bundled dataset to the sex total. The dataset itself also holds "male" and "female"; reach those with [nor_population_by_sex_age_cats\(\)](#).

calyear Calendar year.

pop_jan1_n Population count as of 1 January of calyear.

imputed Logical. TRUE if the value was imputed.

See Also

vignette("population_norway", package = "csdata") plots the national series and tabulates pop_jan1_n by location and calendar year from this function. vignette("csdata", package = "csdata") gives the csverse coding rules for the age and sex columns.

Other population data: [nor_population_by_sex_age_cats\(\)](#)

Examples

```
# data.table's default multi-threading pushes this example past CRAN's
# CPU-to-elapsed limit, so pin it to one thread and restore afterwards.
old_threads <- data.table::getDTthreads()
data.table::setDTthreads(1)

# Default: return total population only
d <- nor_population_by_age_cats()
print(head(d[granularity_geo == "nation"]))

# Named age bands
d2 <- nor_population_by_age_cats(
  cats = list("children" = 0:17, "working_age" = 18:66, "elderly" = 67:105),
  include_total = TRUE
)
print(d2[granularity_geo == "nation" & calyear == 2024])

data.table::setDTthreads(old_threads)
```

nor_population_by_sex_age_cats

Population in Norway by sex and age categories

Description

Aggregates the Norwegian population into custom age categories, split by sex (male/female) and optionally the total of both sexes.

Usage

```
nor_population_by_sex_age_cats(
  cats = NULL,
  include_total_age = TRUE,
  include_total_sex = TRUE,
  include_9999 = FALSE,
  border = cldata::config$border_nor
)
```

Arguments

<code>cats</code>	A list of vectors that give the age values to include in each category.
<code>include_total_age</code>	Logical. If TRUE, the function includes "total" as an age category.
<code>include_total_sex</code>	Logical. If TRUE, the function includes "total" (both sexes combined) as a sex, alongside "male" and "female".
<code>include_9999</code>	Logical. If TRUE, the function duplicates the current calendar year and adds it with <code>calyear = 9999</code> . This follows the <code>cstidy</code> convention for <code>granularity_time == "event_*</code> ".
<code>border</code>	Integer. The Norwegian geographic border year. Valid values: 2024.

Details

Statistics Norway provides no sex breakdown for Svalbard and Jan Mayen (`notmainlandcounty_nor21`, `notmainlandcounty_nor22`, `notmainlandmunicip_nor2100`, `notmainlandmunicip_nor2200`). On those four codes `pop_jan1_n` is NA on the male and female rows, and the total row holds the real count. For the unknown-location codes (`missingcounty_nor99`, `missingmunicip_nor9999`) `pop_jan1_n` is NA on all three sex rows, total included.

When `include_total_sex = TRUE`, the output holds male, female and total in long format. A sum of `pop_jan1_n` across all sex values therefore double-counts. Filter to a single sex, or sum only the male/female components.

Value

A `data.table` containing the following columns:

- `granularity_geo`
- `location_code`
- age (as specified in the argument "cats")
- sex ("male", "female", and "total" if `include_total_sex`)
- `calyear`

- `pop_jan1_n`
- `imputed`

See Also

No vignette covers this function. `vignette("population_norway", package = "csdata")` uses [nor_population_by_age_cats\(\)](#), which returns the sex total only. `vignette("csdata", package = "csdata")` gives the csverse coding rules for the age and sex columns.

Other population data: [nor_population_by_age_cats\(\)](#)

Examples

```
# data.table's default multi-threading pushes this example past CRAN's
# CPU-to-elapsed limit, so pin it to one thread and restore afterwards.
old_threads <- data.table::getDTthreads()
data.table::setDTthreads(1)

d <- nor_population_by_sex_age_cats(cats = list(c(1:10), c(11:20)))
print(d[location_code == "nation_nor" & calyear == 2024])

data.table::setDTthreads(old_threads)
```

se

Swedish characters in unicode

Description

A named list of Swedish special characters as unicode strings, for use where literal non-ASCII characters are inconvenient. Elements: `OE/oe` (`Oe/oe`), `AE/ae` (`Ae/ae`). Each element is a single character. These four names are spelled the same way in [nb](#), so code that indexes `se` by name also works against `nb`.

Usage

`se`

See Also

No vignette covers this object.

Other unicode character lists: [nb](#)

Examples

```
print(csdata::se)
csdata::se$oe # lowercase oe
```

set_config	<i>Set package configuration options</i>
------------	--

Description

Updates `border_nor` in the `config` environment. Call this at the start of a script to change the default `border` argument of `nor_locations_names()`, `nor_locations_redistricting()`, `nor_locations_hierarchy_from_to`, `nor_population_by_age_cats()` and `nor_population_by_sex_age_cats()`.

Usage

```
set_config(border_nor = NULL)
```

Arguments

<code>border_nor</code>	Integer. The Norwegian geographic border year to use as the default. Valid values: 2024. Pass NULL to leave unchanged.
-------------------------	--

Value

Invisibly, the value assigned to `config$border_nor`, or NULL when `border_nor` is NULL. The function exists for its side effect on `config`.

See Also

No vignette covers this function.

Examples

```
old <- csdata::config$border_nor
csdata::set_config(border_nor = 2024)
csdata::config$border_nor # 2024
csdata::set_config(border_nor = old) # restore
```

Index

- * **data set column adders**
 - add_granularity_geo_to_data_set, 2
 - add_iso3_to_data_set, 3
- * **location code converters**
 - location_code_to_granularity_geo, 5
 - location_code_to_iso3, 6
- * **population data**
 - nor_population_by_age_cats, 11
 - nor_population_by_sex_age_cats, 12
- * **unicode character lists**
 - nb, 7
 - se, 14

add_granularity_geo_to_data_set, 2

add_granularity_geo_to_data_set(), 3

add_iso3_to_data_set, 3

add_iso3_to_data_set(), 3

config, 4, 15

location_code_to_granularity_geo, 5

location_code_to_granularity_geo(), 3, 6

location_code_to_iso3, 6

location_code_to_iso3(), 3, 5

nb, 7, 14

nor_locations_hierarchy_from_to, 7

nor_locations_hierarchy_from_to(), 4, 15

nor_locations_names, 9

nor_locations_names(), 2, 4–6, 8, 10, 11, 15

nor_locations_redistricting, 10

nor_locations_redistricting(), 4, 15

nor_population_by_age_cats, 11

nor_population_by_age_cats(), 4, 14, 15

nor_population_by_sex_age_cats, 12

nor_population_by_sex_age_cats(), 4, 12, 15

se, 7, 14

set_config, 15

set_config(), 4