Package: naturdata (via r-universe)

May 29, 2026

Title Tools For Working With Naturdata_kalo

Version 0.4.16

Description This R package provides tools for working with the naturdata_kalo database

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Imports cli, DBI, dbplyr, dplyr, glue, lubridate, odbc, rlang, RSQLite, stringi, stringr, tibble, wk

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 3.5)

LazyData true

Config/pak/sysreqs make libicu-dev unixodbc-dev

Repository https://ldalby.r-universe.dev

Date/Publication 2025-03-13 13:03:15 UTC

RemoteUrl https://gitlab.au.dk/ecos/tools/r-pkgs/naturdata

RemoteRef HEAD

RemoteSha 7eb6030d7ec66cf428ef0a3c3b169ede3f3d0c6f

Contents

`get_condition` *Generate a condition string*

Description

Generate a condition string based on the input arguments. This function is only to be used in `replace_na_if()`.

Usage

```
get_condition(column, progIds, naturtypeIds = NULL, year = NULL, year_cond)
```

Arguments

`column`	name The name of the column to replace values in
`progIds`	int The progIds to modify values within
`naturtypeIds`	int The naturtypeIds to modify values within
`year`	int The year to use in the condition. See `year_cond`
`year_cond`	chr The condition to apply on `year`

`ndb_aggregate_pinpoint` *Calculate number of pins hit by an aggregated species group*

Description

Calculate number of pins hit by an aggregated species group as the unique number of pins hit by the group.

Usage

```
ndb_aggregate_pinpoint(pp)
```

Arguments

`pp`	chr The pinpoint hits as character string of the format "[1,2,4,16]"

Value

int The number of pins

Examples

```
tibble::tibble(
aktId = 1,
artId = 1:2,
pinpoint = c("[3,7,9,11,13,14,16]", "[1,2,3]")
) |>
 dplyr::summarize(hyppighed = ndb_aggregate_pinpoint(pinpoint),
                  .by = aktId)
```

`ndb_append_coords` *Extract x and y from WKT*

Description

Coordinates are stored as WKT on the database in the column `geomText` This function will extract the x and y from `geomText` and add them as columns to `data`.

Usage

```
ndb_append_coords(data)
```

Arguments

`data`	data.frame containing a WKT string in `geomText`

Details

Coordinates in `geomText` are in UTM32 (EPSG 25832)

Value

`data` with augmented with `utm_x` and `utm_y`

`ndb_calc_avg` *Calculate the average across multiple columns*

Description

Calculate the average of e.g. vegetationheight which is recorded by four measurements each in it own column. The average is only calculated where there are at least `min_n_values` non-na values.

Usage

```
ndb_calc_avg(tbl, prefix, min_n_values = 3)
```

Arguments

`tbl`	The table with the columns to average over.
`prefix`	chr The string with which the columns are selected (using `dplyr::starts_with()`).
`min_n_values`	int Number of required non-na values (Default 3).

Value

tibble Table with `aktId` and the calculated average as prefix_avg

---

`ndb_calc_cover_ratio` *Calculate the cover ratio*

Description

Calculate the cover ratio as the ratio between species A to the sum of species A and species B The cover ratio is defined as

$$\frac{\text{species A}}{\text{sum(species A, species B)}}$$

Usage

```
ndb_calc_cover_ratio(species_A, species_B)
```

Arguments

`species_A`	int The number of pins hitting species A.
`species_B`	int The number of pins hitting species B.

Value

The ratio of species A to the sum of species A and species B.

Examples

```
ndb_calc_cover_ratio(10L, 5L)
```

`ndb_calc_jord_cn` *Calculate jord_CN*

Description

The function will calculate the ratio between jord_C and jord_N while performing checks for whether both are within the detection limit. Lastly the function will ensure that the calculated ratio is within 10 and 60.

Usage

```
ndb_calc_jord_cn(tbl)
```

Arguments

`tbl`	A table containing the columns jord_C, jord_N, jord_C_under_detektionsgraensen and jord_N_under_detektionsgraensen.

Value

tbl agumented with the jord_CN column

`ndb_calc_n_arter` *Calculate the number of unique species observations*

Description

Art, Hybrid, Sektion, Superart are counted as unique observations. Form and Variant are counted as their parent species.

Usage

```
ndb_calc_n_arter(artsliste, artsnavne)
```

Arguments

`artsliste`	data.frame with `artId`, `aktId` and `taxonomi`
`artsnavne`	data.frame returned by `ndb_get_artsnavne()`

Value

tibble with `artId` & `n_arter`

See Also

`ndb_get_artsnavne()`

`ndb_calc_n_store_stammer`

Calculate the number of large trunks

Description

Function to calculate the number of large trunks. What constitutes a large trunk is dependent on the naturetype. Since the naturetype can vary over time and is sometimes interpreted from previous registrations n_store_stammer needs to be re-calculated (in some situations).

Usage

```
ndb_calc_n_store_stammer(nt, traeregistrering, traediameter)
```

Arguments

`nt`	A table with (at least) `aktId` & `naturtypeId`
`traeregistrering`	The table `data.skov_traeregistrering` from naturdata_kalo. This is a list of the species measured in the plot.
`traediameter`	The list `lists.trae_diameter_naturtype`. This is a table with the criteria for when a certain tree species is considered large (growth is naturetype dependent).

Details

The circle diameter has varied over time between 10 and 15m. To accomodate for that observations from the 10m circle are scaled up by the ratio of the areas of two circles with radii 15 and 10 is given by: $\frac{\pi \cdot 15^2}{\pi \cdot 10^2} = \frac{225}{100} = \frac{9}{4} \approx 2.25$

Value

Table with `aktId` and `n_store_stammer_15m`

`ndb_calc_weighted_assessment`

Calculate an area weighted assessment

Description

Calculate a weighted assessment where the area within and outside the NATURA 2000 sites is used as weights.

Usage

```
ndb_calc_weighted_assessment(Ai, Au, ni, nu, vi, vu)
```

Arguments

`Ai`	Area inside NATURA 2000 sites
`Au`	Area outside the NATURA 2000 sites
`ni`	Number of plots inside NATURA 2000
`nu`	Number of plots outside NATURA 2000
`vi`	Proportion of plots that meet the criteria inside the NATURA 2000 sites
`vu`	Proportion of plots that meet the criteria outside the NATURA 2000 sites

Details

The weighted assessment is defined as: $v = \frac{A_i\sqrt{n_i}v_i + A_u\sqrt{n_u}v_u}{A_i\sqrt{n_i} + A_u\sqrt{n_u}}$

Value

The weighted assesment

`ndb_clean_artsliste` *Function to clean the artsliste table*

Description

Cases with `hyppighed` > 16 are set to NA. Cases where `hyppighed` == 0 & `isPinpoint` == 1 are changed to `isPinpoint` == 0 and `isProevefelt` == 1

Usage

```
ndb_clean_artsliste(tbl)
```

Arguments

`tbl`	tibble The raw.artsliste table

Value

`tbl` cleaned

`ndb_clean_kemi` *Function to clean the raw.kemi table*

Description

The function will insert a value of half the detection limit for values less than the detection limit (0.05 for `jord_C` and 0.01 for `jord_N`). For pH values outside 2.5 - 8.5 will be replaced with `NA`. In case `vand_ph` is missing `vand_ph2` is used instead. The `vand_ph2` is not included in the returned table.

Usage

```
ndb_clean_kemi(tbl)
```

Arguments

`tbl`	tibble The raw.kemi table

Details

The ratio between `jord_C` and `jord_N` is also calculated and added to the returned table.

Value

`tbl` cleaned

`ndb_clean_lysaaben_struktur`
Function to clean the raw.lysaaben_struktur table

Description

This function will perform several clean-up steps.

- Replace vegetationshojde > 300 with NA
- Replace areas over 78.5 and <= 80 with 78.5
- Replace areas over 80 with NA
- Replace negative values with NA

Usage

```
ndb_clean_lysaaben_struktur(tbl)
```

Arguments

`tbl`	tibble The raw.lysaaben_struktur table

Details

In addition NA's are replaced with 0 under these conditions #TODO

hoeljer_5m_observeret is not written to data as this is the raw interval strings which are not used currently.

Value

tbl cleaned

`ndb_clean_plantekemi` *Function to clean the plantekemi table*

Description

The function will ensure that all species have `artId` and that all `naturtypeId`'s are valid (i.e. not marine or aquatic). The function will also insert a value of half the detection limit for values less than the detection limit (0.03 for plante_N and 0.01 for plante_P) and add julian date as a column. Cases where samples have been taken from species that was not part of the protocol are removed (Dunet dueurt, Søblomme, Ager-padderok, Uomsat dødt organisk materiale). Lastly a few cases where multiple `artId` for the same `artsNavn` are set to the correct artID.

Usage

```
ndb_clean_plantekemi(tbl, nogo_naturtyper)
```

Arguments

`tbl`	tibble The raw.plantekemi table
`nogo_naturtyper`	integer Vector of illegal naturtypeId's.

Details

The julianDate column is calculated using `lubridate::yday()`.

`ndb_clean_skov_doedtved` *Function to clean the skov_doedtved table*

Description

To have valid column names when pivoting to wider formats values with å in `retning` are replaced with aa. Cases where `nedbrydning` != 1:5 are set to NA. The `comment` column is dropped.

Usage

```
ndb_clean_skov_doedtved(tbl)
```

Arguments

`tbl`	tibble The raw.skov_doedtved table

Value

`tbl` cleaned

`ndb_clean_skov_struktur` *Function to clean the skov_struktur table*

Description

Columns with coverage in the 5 m circle have areas adjusted to 78.5 m2 when values are between 78.5 - 80, coverage over 80 are set to NA. In all cases negative values are replaced with NA. For `progId` 133, 134 with `naturtypeId` 9999 NA are replaced with 0. Count and area columns are converted to integer. For for all coverage variables NAs are replaced with 0. For `n_hulheder_15m` and `n_raad_15m` NAs are only replaced from 2011 and onwards.

Usage

```
ndb_clean_skov_struktur(tbl)
```

Arguments

`tbl`	tibble The raw.skov_struktur table

Details

The sum of `hulheder_15m` and `raad_15m` is also calculated by this function as `n_hulheder_raad_samlet_15m`

`ndb_clean_stamdata` *Function to clean the raw.stamdata table*

Description

Currently just sets `naturtypeId` to `NA` when `status` is not "Godkendt data" (`statusId` = 100).

Usage

```
ndb_clean_stamdata(tbl)
```

Arguments

`tbl`	tibble The raw.stamdata table

Value

`tbl` cleaned

`ndb_connect` *Connect to naturdata_kalo*

Description

The function assumes that `server`, `database`, `uid` & `pwd` are stored in the .Renviron file using the variable names NATURDATA_KALO_SERVER, NATURDATA_KALO_DB, NATURDATA_KALO_UID & NATURDATA_KALO_PWD

Usage

```
ndb_connect()
```

Details

Note that the ODBC Driver 17 for SQL Server is required.

Value

OdbcConnection

`ndb_create_tbl` *Create and write table*

Description

This wraps `DBI::dbWriteTable()` to create a new table

Usage

```
ndb_create_tbl(con, obj, schema = "data", table)
```

Arguments

`con`	An OdbcConnection
`obj`	Table to write
`schema`	chr Name of the schema
`table`	chr Name of the table to create

Details

`create_tbl` will overwrite any existing table.

Value

TRUE, invisibly

`ndb_disconnect` *Disconnect from database*

Description

Simple wrapper around `DBI::dbDisconnect()`

Usage

```
ndb_disconnect(con)
```

Arguments

`con`	OdbcConnection

`ndb_filter_artsliste` *Filter the artsliste table*

Description

Filter artslisten to only have observations where `isPinpoint`, `isProvefelt` or `is5m` has the value of 1.

Usage

```
ndb_filter_artsliste(artsliste)
```

Arguments

`artsliste`	The artsliste table

See Also

`ndb_get_artsliste_pf()`

`ndb_filter_lysaaben_prog` *Filter to only program IDs for lysåben*

Description

Filter on `progId` to only 117,119,131:132,142,144

Usage

```
ndb_filter_lysaaben_prog(tbl)
```

Arguments

`tbl`	A table that contains the column `progId`

Value

tbl filtered

`ndb_filter_skov_prog` *Filter to only skov programs*

Description

Filter on `progId` to only 133, 134 or 140

Usage

```
ndb_filter_skov_prog(tbl)
```

Arguments

`tbl`	A table that contains the column `progId`

Value

tbl filtered

`ndb_get` *Get a specified table from the database*

Description

Get a specified table from the database

Usage

```
ndb_get(con, schema, tbl)
```

Arguments

`con`	Connection to the database
`schema`	chr Name of the schema where `tbl` lives
`tbl`	chr Name of table

`ndb_get_artsgruppe` *Get the artsgruppe_planter table*

Description

Get the artsgruppe_planter table

Usage

```
ndb_get_artsgruppe(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table

Source

lists.artsgruppe_planter

`ndb_get_artsliste_pf` *Get the artsliste table*

Description

Get the artsliste table and filter it to only have observations where `isPinpoint`, `isProvefelt` or `is5m` has the value of 1.

Usage

```
ndb_get_artsliste_pf(con)
```

Arguments

`con`	Connection to the database

Source

data.artsliste

See Also

`ndb_filter_artsliste()`

`ndb_get_artsnavne` *Get the list of species names*

Description

Get the list of species names and filter to only include either plants (when `taxon` = 'plants') or only bryophtyes when `taxon` = 'bryophytes'. 'Plants' is definded as species from the phyla *Tracheophyta*, *Magnoliophyta*, *Pinophyta* and *Pteridophyta*. 'Bryophytes' is defined as the classes *Bryopsida* and *Polytrichopsida*.

Usage

```
ndb_get_artsnavne(con, taxon)
```

Arguments

`con`	Connection to the database
`taxon`	chr Either "bryophytes" or "plants".

Details

In addition to filtering, the function will also ensure that Form and Variant are treated as their parent species by `ndb_calc_n_arter()` whereas Art, Underart, Hybrid, Sektion and Superart are treated as unique observations.

Source: lists.artsNavne

Value

A tibble with species names.

See Also

`ndb_calc_n_arter()`

`ndb_get_artsscorer` *Get the list of artsscorer per species*

Description

Get the list of artsscorer per species

Usage

```
ndb_get_artsscorer(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table

Source

lists.artsScore

`ndb_get_arttraits` *Get the list of traits*

Description

Get the list of traits

Usage

```
ndb_get_arttraits(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table

Source

lists.arttraits

`ndb_get_doedtved` *Get the skov doedtved table*

Description

Get the skov doedtved table

Usage

```
ndb_get_doedtved(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table

Source

data.skov_doedtved

`ndb_get_ellenberg` *Get the Ellenberg values*

Description

Get the list of Ellenberg values (the English version) plus `T_EllTysk`. It is somewhat misleading that `T_EllTysk` is in lists.ellenberg_engelsk, but it is not a mistake.

Usage

```
ndb_get_ellenberg(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table

Source

lists.ellenberg_engelsk

`ndb_get_laver` *Get the list of lichen species names*

Description

Get the list of lichen species names (i.e. species in class *Lecanoromycetes*). The returned species list will have a column `isLav` with all 1L's.

Usage

```
ndb_get_laver(con)
```

Arguments

`con`	Connection to the database

Value

A tibble with lichen species names.

Source

lists.artsNavne

`ndb_get_lysaaben_struktur` *Get the lysaaben struktur table*

Description

Get the lysaaben struktur table

Usage

```
ndb_get_lysaaben_struktur(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table

Source

data.lysaaben_struktur

`ndb_get_mosser` *Get the artsnavne table only of bryophytes*

Description

Get the list of bryophyte species names (i.e. species in phylum *Bryophyta*).

Usage

```
ndb_get_mosser(con)
```

Arguments

`con`	Connection to the database

Value

A tibble with bryophyte species names.

Source

lists.artsNavne

`ndb_get_novana_aktids` *Get the list of aktId's from NOVANA*

Description

Get a list of aktId's where `formaal` == `"NOVANA"` and `ansvarligMyndighed` == `"Milj\u00f8styrelsen"`

Usage

```
ndb_get_novana_aktids(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table with `progId`, `aktId` & `dato`

Source

data.stamdata

`ndb_get_plantekemi` *Get the plantekemi table*

Description

Return the plantekemi table

Usage

```
ndb_get_plantekemi(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table

Source

data.plantekemi

`ndb_get_schema` *Get the database schema*

Description

Query the database for schemas, tables and their columns. Useful when only given a column name and need to find out where it lives on the database.

Usage

```
ndb_get_schema(con)
```

Arguments

`con`	OdbcConnection

Details

The rapportering scheme is rarely relevant and is not returned. The same goes for two tables trend_p4 and tilstand_p4#'

Value

A dataframe with

schema chr, Schema name

table chr, Table name

column chr, Column name

`ndb_get_skov_struktur` *Get the skov struktur table*

Description

Get the skov struktur table

Usage

```
ndb_get_skov_struktur(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table

Source

data.skov_struktur

`ndb_get_skov_trae_diam_naturtype` *Get the trae diameter per naturtype table*

Description

This table holds information about the diameter a trunk should have for it to be considered "big".

Usage

```
ndb_get_skov_trae_diam_naturtype(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table

Source

list.trae_diameter_naturtype

`ndb_get_skov_traeregistrering` *Get the skov traeregistrering table*

Description

Get the skov traeregistrering table

Usage

```
ndb_get_skov_traeregistrering(con)
```

Arguments

`con`	Connection to the database

Value

A lazy table

Source

data.skov_traeregistrering

`ndb_get_stamdata` *Get the stamdata table*

Description

Get the stamdata table

Usage

```
ndb_get_stamdata(con)
```

Arguments

`con`	Connection to the database

`ndb_get_tbl` *Get table from the database and filter*

Description

Get table from the database and filter to only rows with formaal == "NOVANA" and ansvarligMyndighed == "Miljøstyrelsen". Joins (both the semijoin on novana and the left_join on kv) are done on columns `progId` and `aktId`.

Usage

```
ndb_get_tbl(con, tbl, novana, kv = NULL)
```

Arguments

`con`	Connection to the database
`tbl`	chr Name of table
`novana`	lazy table Table with the progId & aktId that are from the NOVANA monitoring program.
`kv`	lazy table An optional table with status ("Godkendt data", "Indtastning påbegyndt") for the data.

Details

The filtering is done via the `dplyr::semi_join()` on `novana` which is a table of `progId` and `aktId` where only the entries relevant for NOVANA are included.

Value

lazy table

`ndb_label` *Given the variable name, return the label*

Description

The names of the variables (aka. indikators) used on Naturdata_kalo are not well suited for use on a webpage or figure. This function will return a more meaningful, but verbose string.

Usage

```
ndb_label(data, ...)
```

Arguments

`data`	Vector of variable names
`...`	Not used

Value

data as label

Examples

```
ndb_label("val_vegetationshoejde_avg_5m")
```

`ndb_labels` *Naturdata_kalo labels*

Description

A named vector of variable (aka. indicator) labels. Not particularly useful on its own, but is the basis for the rename function `ndb_label()`

Usage

```
ndb_labels
```

Format

A named vector of length 130

`ndb_names` *Naturdata_kalo names*

Description

A named vector of variable (aka. indicator) names. Not particularly useful on its own, but is the basis for the rename function `ndb_rename()`

Usage

```
ndb_names
```

Format

A named vector of length 130

`ndb_rename` *Rename either a data.frame or vector*

Description

The names of naturdata_kalo where updated in 2024. This function makes it easy to convert old variable names to the new ones.

Usage

```
ndb_rename(data, ...)
```

Arguments

`data`	Vector of names or a data.frame with old names
`...`	Not used

Details

Names in the deprecated Frekvens and Cover tables where not unique. To ensure that the function returns the correct names prefix the old names with "frekvens_" or "cover_". The ambigous names where klokkelyng, hedelyng, blåtop, stjerne-bredribbe and revling.

Value

data renamed

Examples

```
ndb_rename("antal_unik")

tibble::tibble(
 "enebaer_5m" = 1:3,
 "unknown" = "A") |>
 ndb_rename()
```

`ndb_to_percent` *Convert from $m\hat{}2$ cover to percent*

Description

For communication of results it is often desirable to use percent rather than m^2. This function converts measurements in m^2 to percent using the convention that a 5m circle has an area of 78.5 m^2 and a 15m circle an area of 706.9 m^2

Usage

```
ndb_to_percent(value, radius = NA)
```

Arguments

`value`	A measurement in m^2
`radius`	Radius of the circle (either 5 or 15)

`ndb_update_import_job` *Update the import.jobs table*

Description

Utility function to update the status, rundate and version in the import.jobs table

Usage

```
ndb_update_import_job(con, job, status, version)
```

Arguments

`con`	OdbcConnection
`job`	chr JobId e.g. "data.lysaaben_struktur"
`status`	chr The status which can be either running or complete
`version`	chr The current version of the project. This is passed to the script as a an input argument. See naturdata-raw-to-data

Value

TRUE, invisibly

`ndb_write_artsliste` *Write artsliste to database*

Description

This a modified version of `ndb_write_to_db()` where the rows in `name` are written species by species to avoid writing too large chunks.

Usage

```
ndb_write_artsliste(con, tbl, schema = "data", name)
```

Arguments

`con`	OdbcConnection to `naturdata_kalo`
`tbl`	tibble The table to be written
`schema`	chr Name of the schema to write to
`name`	chr Name of table to be written to `schema`

`ndb_write_to_db` *Write table to database*

Description

The function will first call `DELETE FROM` before appending the contents of `tbl`.

Usage

```
ndb_write_to_db(con, tbl, schema = "data", name)
```

Arguments

`con`	OdbcConnection to `naturdata_kalo`
`tbl`	tibble The table to be written
`schema`	chr Name of the schema to write to
`name`	chr Name of table to be written to `schema`

Value

TRUE, invisibly

`nogo_naturtyper` *No-go naturtyper*

Description

A simple vector of marine and aquatic naturtypeId's. These are used for filtering tables to contain only the terrestrial types.

Usage

```
nogo_naturtyper
```

Format

A vector of 21 integer values

`replace_na_if` *Replace NA values based on a set of conditions*

Description

Replace NA in a column with 0 when conditions based on `progId` and optionally `naturtypeId` are true.

Usage

```
replace_na_if(
  tbl,
  column,
  progIds,
  naturtypeIds = NULL,
  year = NULL,
  year_cond
)
```

Arguments

`tbl`	Input table
`column`	name The name of the column to replace values in
`progIds`	int The progIds to modify values within
`naturtypeIds`	int The naturtypeIds to modify values within
`year`	int The year to use in the condition. See `year_cond`
`year_cond`	chr The condition to apply on `year`

Value

`tbl` with `column` modified.

Index