Package 'tidyweather'

January 17, 2026

Title Analysis the Weather Data for Agriculture

Version 0.1.0

Description Functions are collected to analyse weather data for agriculture purposes including to read weather records in multiple formats, calculate extreme climate index. Demonstration data are included the SILO daily climate data (licensed under CC BY 4.0, <https://www.longpaddock.qld.gov.au/silo/>).

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports stringr, tibble, dplyr, settings

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://tidyweather.bangyou.me/, https://github.com/byzheng/tidyweather

BugReports https://github.com/byzheng/tidyweather/issues

NeedsCompilation no

Author Bangou Zheng [aut, cre]

Maintainer Bangou Zheng <bangyou.zheng@csiro.au>

Repository CRAN

Date/Publication 2026-01-17 11:20:02 UTC

Contents

`check_weather` *Check weather records for data quality issues*

Description

This function validates weather records for:

- Continuous weather data (no gaps in dates)
- No missing values in key columns (mint, maxt, radn, rain)
- No extreme values (e.g., less than -100 or above 100 for temperature, less than 0 for radiation and rain)
- Latitude and longitude columns exist and contain a single non-NA value for all records

Usage

```
check_weather(
  data,
  key_cols = c("mint", "maxt", "radn", "rain"),
  temp_range = c(-100, 100),
  radn_range = c(0, 50),
  rain_range = c(0, 500),
  stop_on_error = FALSE
)
```

Arguments

`data`	A data.frame or tibble containing weather records with at minimum a date column, latitude, longitude, and key weather variables (mint, maxt, radn, rain).
`key_cols`	A character vector of column names to check for missing values and extreme values. Default is c("mint", "maxt", "radn", "rain").
`temp_range`	A numeric vector of length 2 specifying the acceptable range for temperature values (mint, maxt). Default is c(-100, 100).
`radn_range`	A numeric vector of length 2 specifying the acceptable range for radiation values. Default is c(0, 50).
`rain_range`	A numeric vector of length 2 specifying the acceptable range for rainfall values. Default is c(0, 500).
`stop_on_error`	Logical. If TRUE, the function will stop with an error when issues are found. If FALSE, it will return a list of issues. Default is FALSE.

Value

If stop_on_error is FALSE, returns a list with the following components:

`is_valid`	Logical indicating if all checks passed
`date_gaps`	Data frame of date gaps found, or NULL if none
`missing_values`	Data frame summarizing missing values, or NULL if none
`extreme_values`	Data frame of rows with extreme values, or NULL if none

If stop_on_error is TRUE and issues are found, the function stops with an error message.

Examples

```
file <- system.file("extdata/ppd_72150.met", package = "tidyweather")
records <- read_weather(file)
result <- check_weather(records)
if (result$is_valid) {
  print("Weather data passed all quality checks")
} else {
  print(result)
}
```

`get_weather_option` *Get weather package option*

Description

Retrieves the value of a specified option from the weather package configuration. This function provides access to package-level settings and preferences.

Usage

```
get_weather_option(name)
```

Arguments

`name`	A character string specifying the name of the option to retrieve.

Value

The value of the specified option, or NULL if the option does not exist.

Examples

```
# Get the default frost threshold
get_weather_option("extreme.frost_threshold")
```

last_frost_day *Calculate the last frost day*

Description

This function calculates the last frost day from a numeric vector of daily minimum temperatures using tidyverse principles.

Usage

```
last_frost_day(
  .data,
  threshold = get_weather_option("extreme.frost_threshold"),
  hemisphere = "south",
  require_full_year = get_weather_option("require_full_year")
)
```

Arguments

.data	A data frame or tibble containing daily minimum temperatures in a column named "mint".
threshold	The stress temperature threshold for frost (default: 0)
hemisphere	Hemisphere indicator: "south" or "north" (default: "south"). If latitude information is available in the data, it will be used to determine the hemisphere.
require_full_year	Logical. If TRUE, requires exactly 365 or 366 days (default: TRUE)

Value

An data.frame or tibble representing the day of year for the last frost, or NA if no frost occurs

Examples

```
file <- system.file("extdata/ppd_72150.met", package = "tidyweather")
records <- read_weather(file)
records |>
    dplyr::group_by(year) |>
    last_frost_day(require_full_year = FALSE)
```

number_frost_day *Calculate the number of frost days*

Description

This function calculates the number of frost days from a numeric vector of daily minimum temperatures using tidyverse principles.

Usage

```
number_frost_day(
  .data,
  threshold = get_weather_option("extreme.frost_threshold"),
  require_full_year = get_weather_option("require_full_year")
)
```

Arguments

.data	A data frame or tibble containing daily minimum temperatures in a column named "mint".
threshold	The stress temperature threshold for frost (default: 0)
require_full_year	Logical. If TRUE, requires exactly 365 or 366 days (default: TRUE)

Value

An data.frame or tibble representing the number of frost days, or 0 if no frost occurs

Examples

```
file <- system.file("extdata/ppd_72150.met", package = "tidyweather")
records <- read_weather(file)
records |>
    dplyr::group_by(year) |>
    number_frost_day(require_full_year = FALSE)
```

read_weather *Read weather records from a file list and/or a folder list*

Description

Read weather records from a file list and/or a folder list

Usage

```
read_weather(file, format = "APSIM", ...)
```

Arguments

file	A character string to specify weather filename.
format	A character string to specify the format of weather file.
...	Other arguments

Value

A data.frame which contains all weather data.

Examples

```
file <- system.file("extdata/ppd_72150.met", package = "tidyweather")
records <- read_weather(file)
head(records)
```

---

summarise_weather *Summarise Weather Extremes and Key Indicators*

Description

This function calculates summary metrics for weather data, including the number of frost days and the last frost day, grouped by one or more grouping variables. The function uses package-wide options for thresholds and year completeness.

Usage

```
summarise_weather(.data)
```

Arguments

.data	A tibble or data frame containing daily weather data. Must include at least a mint column for daily minimum temperatures. A day column is recommended if require_full_year = TRUE.

Details

The function retrieves thresholds and settings from the global tidyweather options via weather_options(). The default frost threshold is get_weather_option("extreme.frost_threshold") and require_full_year is get_weather_option("require_full_year"). These can be changed using weather_options().

This function is designed to work with grouped tibbles (e.g., after dplyr::group_by()), applying the summary per group.

Value

A tibble with one row per group, containing the following columns:

number_frost_days Number of days where minimum temperature is below the frost threshold.

last_frost_day The day of year of the last frost (or NA if none).

Examples

```
library(dplyr)

# Example weather data (daily minimum temperatures)
weather_data <- read_weather(system.file("extdata/ppd_72150.met", package = "tidyweather"))

# Summarise without grouping
weather_options(require_full_year = FALSE)
summarise_weather(weather_data)

# Summarise by group (e.g., year)
weather_data_grouped <- weather_data %>% group_by(year)
summarise_weather(weather_data_grouped)
```

---

weather_options *Set or get options for tidyweather*

Description

This function allows users to get or set configuration options for the tidyweather package.

Usage

```
weather_options(...)
```

Arguments

...	Option names to retrieve or key-value pairs to set.

Details

The options are managed via a nested structure that distinguishes between the tidyweather package.

Value

If called with no arguments, returns all current options. If called with named arguments, updates and returns the modified options.

Supported options

`extreme.frost_threshold` Frost threshold for extreme weather events.

`require_full_year` Whether to require a full year of data for calculations.

Examples

```
# Get all options
weather_options()

# Set frost_threshold
weather_options(extreme.frost_threshold = 2)
```

`weather_reset` *Reset all tidyweather options to defaults*

Description

This restores all package settings to their initial defaults as defined when tidyweather was loaded. Use this if you want to undo all customizations.

Usage

```
weather_reset()
```

Value

Invisibly returns the default settings after reset.

Examples

```
weather_options(extreme.frost_threshold = -2)
weather_reset()
weather_options()
```

`write_weather` *Write weather data to file*

Description

Exports weather records to a file in the specified format. Currently supports APSIM format for agricultural modeling applications.

Usage

```
write_weather(records, file, format = "APSIM", overwrite = FALSE)
```

Arguments

records	A data frame containing weather data with columns for date, temperature, precipitation, and other meteorological variables
file	Character string specifying the output file path
format	Character string specifying the output format. Currently supports "APSIM" (default)
overwrite	Logical indicating whether to overwrite existing files. Default is FALSE

Value

Invisibly returns the file path of the written file

Examples

```
# Read sample weather data from package
file <- system.file("extdata/ppd_72150.met", package = "tidyweather")
records <- read_weather(file)

# Write to temporary file
temp_file <- tempfile(fileext = ".met")

# Write to APSIM format
write_weather(records, temp_file, format = "APSIM")

# Overwrite existing file
write_weather(records, temp_file, format = "APSIM", overwrite = TRUE)
```

Index