Package: datefixR (via r-universe)

October 8, 2026

Title Standardize Dates in Different Formats or with Missing Data

Version 2.0.1.9000

Maintainer Nathan Constantine-Cooke

<nathan.constantine-cooke@ed.ac.uk>

Description There are many different formats dates are commonly represented with: the order of day, month, or year can differ, different separators (``-'', ``/'', or whitespace) can be used, months can be numerical, names, or abbreviations and year given as two digits or four. 'datefixR' takes dates in all these different formats and converts them to R's built-in date class. If 'datefixR' cannot standardize a date, such as because it is too malformed, then the user is told which date cannot be standardized and the corresponding ID for the row. 'datefixR' also allows the imputation of missing days and months with user-controlled behavior.

License GPL (>= 3)

URL https://docs.ropensci.org/datefixR/, https://github.com/ropensci/datefixR

BugReports https://github.com/ropensci/datefixR/issues

Depends R (>= 4.2)

Imports lifecycle, rlang

Suggests anytime, DT, future, future.apply, htmltools, knitr, parsedate, pkgbuild, png, readr, readxl, rmarkdown, shiny, shinytest2, spelling, testthat (>= 3.0.0), withr

VignetteBuilder knitr

Config/testthat/edition 3

Config/testthat/parallel true

Encoding UTF-8

Language en-US

LazyData true

Roxygen list(markdown = TRUE)

Config/rextendr/version 0.5.0

SystemRequirements Cargo (Rust's package manager), rustc, xz

Config/roxygen2/version 8.1.0

Config/pak/sysreqs xz-utils libclang-dev

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

Date/Publication 2026-10-08 16:27:15 UTC

RemoteUrl https://github.com/ropensci/datefixR

RemoteRef main

RemoteSha 48712960ae2acdd1abeebf10d476de950e570b79

Contents

exampledates 2
fix_date_app 3
fix_date_char 3
fix_date_df 6

Index **9**

`exampledates` *Example dataset of dates in different formats*

Description

A toy dataset to use with datefixR functions.

Usage

```
exampledates
```

Format

A data frame with 5 rows and 3 variables:

id Row ID (numeric).

some.dates Dates in different formats (character).

some.more.dates Additional dates in different formats (character).

fix_date_app *Shiny application standardizing date data in csv or excel files*

Description

A shiny application which allows users to standardize dates using a graphical user interface (GUI). Most features of `datefixR` are supported including imputing missing date data. Data can be provided as CSV (comma-separated value) or XLSX (Excel) files. Processed datasets can be downloaded as CSV files. Please note, the dependencies for this app (`DT`, `htmltools`, `readxl`, and `shiny`) are not installed alongside `datefixR`. This allows `datefixR` to be installed on secure systems where these packages may not be allowed. If one of these dependencies is not installed on the system when this function is called, then the user will be given the option of installing them.

Usage

```
fix_date_app(theme = "datefixR")
```

Arguments

`theme`	Color theme for shiny app. Either `"datefixR"` (`datefixR` colors) or `"none"`(default shiny app styling).

Value

A shiny app.

See Also

The `shiny` package.

Examples

```
## Not run:
fix_date_app()

## End(Not run)
```

fix_date_char *Convert non-standardized dates to R's Date class*

Description

Converts a character vector (or single character object) from inconsistently formatted dates to R's `Date` class. Supports numerous separators including /, -, ., or space. Supports numeric, abbreviation, or long-hand month notation in multiple languages (English, French, German, Spanish, Portuguese, Russian, Czech, Slovak, Indonesian). Where day of the month has not been supplied, the first day of the month is imputed by default. Either DMY or YMD is assumed by default. However, the US system of MDY is supported via the `format` argument.

Usage

```
fix_date_char(
  dates,
  day.impute = 1,
  month.impute = 7,
  format = "dmy",
  excel = FALSE,
  roman.numeral = FALSE
)
```

Arguments

dates	Character vector to be converted to R's date class.
day.impute	Integer between 1 and 31, or NA, or NULL. Day of the month to be imputed when missing. Defaults to 1. If day.impute = NA, then NA will be imputed for the date and a warning will be raised. If day.impute = NULL, the function will fail with an error when day is missing.
month.impute	Integer between 1 and 12, or NA, or NULL. Month to be imputed when missing. Defaults to 7 (July). If month.impute = NA, then NA will be imputed for the entire date and a warning will be raised. If month.impute = NULL, the function will fail with an error when month is missing.
format	Character string specifying date interpretation preference. Either "dmy" (day-month-year, default) or "mdy" (month-day-year, US format). This setting only affects ambiguous numeric dates like "01/02/2023". When month names are present or year appears first, the format is auto-detected regardless of this parameter. Note that unambiguous dates (e.g., "25/12/2023") are parsed correctly regardless of the format setting.
excel	Logical: Assumes FALSE by default. If TRUE, treats numeric-only dates with more than four digits as Excel serial dates with 1900-01-01 origin, correcting for known Excel date discrepancies.
roman.numeral	**[Experimental]** Logical: Defaults to FALSE. When TRUE, attempts to interpret Roman numeral month indications within datasets. This feature may not handle all cases correctly.

Details

This function intelligently parses dates by:

- Handling mixed separators within the same dataset
- Recognizing month names in multiple languages
- Converting Roman numeral months (experimental)
- Processing Excel serial date numbers
- Automatically detecting YMD format when year appears first
- Smart imputation of missing date components with user control

For comprehensive examples and advanced usage, see browseVignettes("datefixR") or the package README at https://docs.ropensci.org/datefixR/.

Value

A vector of elements belonging to R's built in `Date` class with the following format `yyyy-mm-dd`.

See Also

`fix_date_df` for data frame columns with date data.

For detailed examples and usage patterns, see:

- Package vignette: `browseVignettes("datefixR")`
- Online documentation: https://docs.ropensci.org/datefixR/articles/datefixR.html
- Package README: https://docs.ropensci.org/datefixR/

Examples

```
# Basic usage
bad.date <- "02 03 2021"
fix_date_char(bad.date)

# Multiple formats with different separators
mixed_dates <- c(
  "02/05/92", # slash separator, 2-digit year
  "2020-may-01", # hyphen separator, text month
  "1996.05.01", # dot separator
  "02 04 96", # space separator
  "le 3 mars 2013" # French format
)
fix_date_char(mixed_dates)

# Text months in different languages
text_months <- c(
  "15 January 2020", # English
  "15 janvier 2020", # French
  "15 Januar 2020", # German
  "15 enero 2020", # Spanish
  "15 de janeiro de 2020" # Portuguese
)
fix_date_char(text_months)

# Roman numeral months (experimental)
roman_dates <- c("15.VII.2023", "3.XII.1999", "1.I.2000")
fix_date_char(roman_dates, roman.numeral = TRUE)

# Excel serial numbers
excel_serials <- c("44197", "44927") # Excel dates
fix_date_char(excel_serials, excel = TRUE)

# Two-digit years (automatic century detection)
two_digit_years <- c("15/03/99", "15/03/25", "15/03/50")
fix_date_char(two_digit_years) # 1999, 2025, 1950

# MDY format (US style)
```

```
us_dates <- c("12/25/2023", "07/04/1776", "02/29/2020")
fix_date_char(us_dates, format = "mdy")

# Incomplete dates with custom imputation
incomplete <- c("2023", "March 2022", "June 2021")
fix_date_char(incomplete, day.impute = 15, month.impute = 6)
```

`fix_date_df` *Clean up messy date columns*

Description

Tidies a `dataframe` or `tibble` object with date columns entered via a free-text interface, addressing non-standardized formats. Supports diverse separators including /, -, ., and spaces. Handles all-numeric, abbreviated, or full-length month names in languages such as English, French, German, Spanish, Portuguese, Russian, Czech, Slovak, and Indonesian. Imputes missing day data by default, with flexibility for custom imputation strategies.

Usage

```
fix_date_df(
  df,
  col.names,
  day.impute = 1,
  month.impute = 7,
  id = NULL,
  format = "dmy",
  excel = FALSE,
  roman.numeral = FALSE,
  cores = getOption("Ncpus", 1)
)
```

Arguments

`df`	A `dataframe` or `tibble` object containing messy date column(s).
`col.names`	Character vector specifying column names of date data to be cleaned.
`day.impute`	Integer between 1 and 31, or NA, or NULL. Day of the month to be imputed when missing. Defaults to 1. If `day.impute` is greater than the number of days in a given month, the last day of that month will be imputed (accounting for leap years). If `day.impute = NA`, then `NA` will be imputed for the entire date and a warning will be raised. If `day.impute = NULL`, the function will fail with an error when day is missing.
`month.impute`	Integer between 1 and 12, or NA, or NULL. Month to be imputed when missing. Defaults to 7 (July). If `month.impute = NA`, then `NA` will be imputed for the entire date and a warning will be raised. If `month.impute = NULL`, the function will fail with an error when month is missing.

`id`	Optional parameter specifying the name of the column containing row IDs. Defaults to using the first column for IDs.
`format`	Character string specifying date interpretation preference. Either `"dmy"` (day-month-year, default) or `"mdy"` (month-day-year, US format). This setting only affects ambiguous numeric dates like "01/02/2023". When month names are present or year appears first, the format is auto-detected regardless of this parameter. Note that unambiguous dates (e.g., "25/12/2023") are parsed correctly regardless of the format setting.
`excel`	Logical: Assumes `FALSE` by default. If `TRUE`, treats numeric-only dates with more than four digits as Excel serial dates with 1900-01-01 origin, correcting for known Excel date discrepancies.
`roman.numeral`	**[Experimental]** Logical: Defaults to `FALSE`. When `TRUE`, attempts to interpret Roman numeral month indications within datasets. This feature may not handle all cases correctly.
`cores`	Integer: Number of CPU cores to use for parallel processing. Defaults to `getOption("Ncpus", 1)`. When `cores > 1`, processes multiple date columns in parallel using the `future` framework. Requires the `future` and `future.apply` packages to be installed. The actual number of workers used will be the minimum of `cores` and the number of columns to process.

Details

This function processes messy date data by:

- Supporting mixed format data entries
- Recognizing multilingual month names and Roman numeral inputs
- Interpreting Excel-style serial date numbers if specified
- Providing warnings and controls for missing day/month imputation

For further details and advanced usage, refer to the vignette via `browseVignettes("datefixR")` or visit the online documentation at https://docs.ropensci.org/datefixR/.

Value

A revised `dataframe` or `tibble` structure, maintaining input type. Date columns will be formatted with `Date` class and display as `yyyy-mm-dd`.

See Also

`fix_date_char` for similar functionality on character vectors.

For comprehensive examples and usage practices, consult:

- Vignette: `browseVignettes("datefixR")`
- Documentation: https://docs.ropensci.org/datefixR/articles/datefixR.html
- README Overview: https://docs.ropensci.org/datefixR/

Examples

```
# Basic cleanup
data(exampledates)
fix_date_df(exampledates, c("some.dates", "some.more.dates"))

# Usage with metadata
messy_dates_df <- data.frame(
  id = seq(1, 3),
  dates = c("1992", "April 1990", "Mar 19")
)
fix_date_df(messy_dates_df, "dates", day.impute = 15, month.impute = 12)

# Diverse format normalization
df_formats <- data.frame(
  mixed.dates = c("02/05/92", "2020-may-01", "1996.05.01", "October 2022"),
  european.dates = c("22.07.1977", "05.06.2023")
)
fix_date_df(df_formats, c("mixed.dates", "european.dates"))

# Excel serial examples
serial_df <- data.frame(serial.dates = c("44197", "44927"))
fix_date_df(serial_df, "serial.dates", excel = TRUE)

# Handling Roman numerals
roman_df <- data.frame(roman.dates = c("15.I.2023", "03.XII.2019"))
fix_date_df(roman_df, "roman.dates", roman.numeral = TRUE)

# Parallel processing (requires 'future' and 'future.apply' packages)
## Not run:
large_df <- data.frame(
  dates1 = c("01/02/2020", "15/03/2021", "22/12/2019"),
  dates2 = c("2020-01-01", "March 2021", "Dec 2019"),
  dates3 = c("01.01.20", "15.03.21", "22.12.19")
)
# Use 4 cores for parallel processing
fix_date_df(large_df, c("dates1", "dates2", "dates3"), cores = 4)

# Use all available cores (respects getOption("Ncpus"))
options(Ncpus = parallel::detectCores())
fix_date_df(large_df, c("dates1", "dates2", "dates3"))

## End(Not run)
```

Index

∗ **datasets**
exampledates, 2

exampledates, 2

fix_date_app, 3
fix_date_char, 3, *7*
fix_date_df, *5*, 6

shiny, *3*