Package 'rainerosr'

March 19, 2026

Type Package

Title Calculate Rainfall Intensity and Erosivity Indices

Version 0.1.1

Description Calculates I30 (maximum 30-minute rainfall intensity) and EI30 (erosivity index) from rainfall breakpoint data. Supports multiple storm events, rainfall validation, and visualization for soil erosion modeling and hydrological analysis. Methods are based on Brown and Foster (1987) <doi:10.13031/2013.30422>, Wischmeier and Smith (1978) ``Predicting Rainfall Erosion Losses: A Guide to Conservation Planning" <doi:10.22004/ag.econ.171903>, and Renard et al. (1997) ``Predicting Soil Erosion by Water: A Guide to Conservation Planning with the Revised Universal Soil Loss Equation (RUSLE)" (USDA Agriculture Handbook No. 703).

License MIT + file LICENSE

Encoding UTF-8

Language en-US

LazyData true

LazyDataCompression bzip2

Depends R (>= 3.5.0)

Imports dplyr (>= 1.1.0), ggplot2 (>= 3.4.0), lubridate (>= 1.9.0)

Suggests testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

RoxygenNote 7.3.3

NeedsCompilation no

Author Sadikul Islam [aut, cre] (ORCID: <https://orcid.org/0000-0003-2924-7122>)

Maintainer Sadikul Islam <sadikul.islamiasri@gmail.com>

Repository CRAN

Date/Publication 2026-03-19 14:50:10 UTC

Contents

`calculate_ei30` *Calculate EI30 (Rainfall Erosivity Index)*

Description

Calculates the EI30 erosivity index, which is the product of total storm kinetic energy (E) and the maximum 30-minute intensity (I30). This is a key parameter in the USLE/RUSLE erosion prediction equations.

Usage

```
calculate_ei30(
  data,
  time_col,
  depth_col,
  interval_col = NULL,
  ke_equation = "brown_foster",
  validate = TRUE
)
```

Arguments

`data`	A data frame containing rainfall breakpoint data
`time_col`	Character string specifying the name of the time/datetime column
`depth_col`	Character string specifying the name of the rainfall depth column (in mm)
`interval_col`	Optional character string specifying the name of the time interval column (in minutes). If NULL, intervals will be calculated from consecutive time differences.
`ke_equation`	Character string specifying which kinetic energy equation to use. Options: "brown_foster" (default), "wischmeier", "mcgregor_mutchler"
`validate`	Logical indicating whether to validate data before calculation (default: TRUE)

Details

The function calculates kinetic energy for each rainfall increment using one of three equations:

Brown & Foster (1987): `e = 0.29 * (1 - 0.72 * exp(-0.05 * i))` where e is unit energy (MJ ha^-1 mm^-1) and i is intensity (mm hr^-1)

Wischmeier & Smith (1978): e = 0.119 + 0.0873log10(i)

McGregor & Mutchler (1976): e = 0.273 + 0.2168i - 0.0083i^2 (for i < 76 mm/hr)

Value

A list with components:

`ei30`	Erosivity index (MJ mm ha^-1 hr^-1)
`total_energy`	Total kinetic energy (MJ ha^-1)
`i30`	Maximum 30-minute intensity (mm hr^-1)
`total_rainfall`	Total rainfall depth (mm)
`duration`	Storm duration (minutes)

Examples

```
data <- data.frame(
  time = as.POSIXct(c("2024-01-01 10:00", "2024-01-01 10:15",
                      "2024-01-01 10:30", "2024-01-01 10:45")),
  depth_mm = c(5.2, 8.3, 4.1, 2.5)
)
result <- calculate_ei30(data, "time", "depth_mm")
print(paste("EI30 =", result$ei30, "MJ mm ha^-1 hr^-1"))
```

`calculate_i30` *Calculate I30 (Maximum 30-minute Rainfall Intensity)*

Description

Calculates the maximum 30-minute rainfall intensity from breakpoint data. This is commonly used in erosion prediction equations like USLE and RUSLE.

Usage

```
calculate_i30(data, time_col, depth_col, interval_col = NULL, validate = TRUE)
```

Arguments

`data`	A data frame containing rainfall breakpoint data
`time_col`	Character string specifying the name of the time/datetime column
`depth_col`	Character string specifying the name of the rainfall depth column (in mm)
`interval_col`	Optional character string specifying the name of the time interval column (in minutes). If NULL, intervals will be calculated from consecutive time differences. For equal-interval data, this is automatically detected and handled correctly.
`validate`	Logical indicating whether to validate data before calculation (default: TRUE)

Details

The function uses a sliding 30-minute window to find the period with maximum rainfall intensity. The intensity is expressed in mm/hr as:

`I30 = (max_rainfall_in_30min / 30) * 60`

For equal-interval data (e.g., every 5, 10, 15, or 30 min), the function automatically detects the fixed interval from timestamps and applies it consistently to all records including the first.

For irregular-interval data, actual elapsed time between timestamps is used directly. When a rainfall interval spans more than 30 minutes, linear interpolation is applied to estimate rainfall at the exact 30-minute boundary.

Value

A list with components:

`i30`	Maximum 30-minute rainfall intensity in mm/hr
`total_rainfall`	Total rainfall depth in mm
`duration`	Total storm duration in minutes
`start_time`	Start time of the maximum intensity period
`end_time`	End time of the maximum intensity period
`interval_type`	Whether intervals were "equal", "irregular", or "explicit"

Examples

```
# Equal interval example (15-min)
data <- data.frame(
  time = as.POSIXct(c("2024-01-01 10:00", "2024-01-01 10:15",
                      "2024-01-01 10:30", "2024-01-01 10:45")),
  depth_mm = c(5.2, 8.3, 4.1, 2.5)
)
result <- calculate_i30(data, "time", "depth_mm")
print(paste("I30 =", result$i30, "mm/hr"))

# Irregular interval example
data2 <- data.frame(
  time = as.POSIXct(c("2024-01-01 10:00", "2024-01-01 10:10",
```

```
                       "2024-01-01 10:35", "2024-01-01 11:00")),
  depth_mm = c(3, 12, 8, 2)
)
result2 <- calculate_i30(data2, "time", "depth_mm")
print(paste("I30 =", result2$i30, "mm/hr"))
```

`plot_intensity_profile` *Plot Rainfall Intensity Profile*

Description

Creates a bar plot showing rainfall intensity over time, highlighting the maximum 30-minute intensity period.

Usage

```
plot_intensity_profile(
  data,
  time_col,
  depth_col,
  interval_col = NULL,
  highlight_i30 = TRUE,
  title = NULL
)
```

Arguments

`data`	A data frame containing rainfall breakpoint data
`time_col`	Character string specifying the name of the time/datetime column
`depth_col`	Character string specifying the name of the rainfall depth column
`interval_col`	Optional character string specifying the name of the time interval column (in minutes)
`highlight_i30`	Logical indicating whether to highlight the maximum 30-minute intensity period (default: TRUE)
`title`	Optional plot title

Value

A ggplot2 object

Examples

```
data <- data.frame(
  time = as.POSIXct(c("2024-01-01 10:00", "2024-01-01 10:15",
                      "2024-01-01 10:30")),
  depth_mm = c(5.2, 8.3, 4.1)
)

plot_intensity_profile(data, "time", "depth_mm")
```

---

plot_rainfall_pattern *Plot Rainfall Pattern*

Description

Creates a visualization of the rainfall pattern over time, showing cumulative rainfall and rainfall intensity.

Usage

```
plot_rainfall_pattern(
  data,
  time_col,
  depth_col,
  interval_col = NULL,
  plot_type = "both",
  title = NULL
)
```

Arguments

data	A data frame containing rainfall breakpoint data
time_col	Character string specifying the name of the time/datetime column
depth_col	Character string specifying the name of the rainfall depth column
interval_col	Optional character string specifying the name of the time interval column (in minutes)
plot_type	Character string specifying plot type: "cumulative", "incremental", or "both" (default: "both")
title	Optional plot title

Value

A ggplot2 object

Examples

```
data <- data.frame(
  time = as.POSIXct(c("2024-01-01 10:00", "2024-01-01 10:15",
                      "2024-01-01 10:30")),
  depth_mm = c(5.2, 8.3, 4.1)
)

plot_rainfall_pattern(data, "time", "depth_mm")
```

`process_storm_events` *Process Multiple Storm Events*

Description

Processes rainfall data containing multiple storm events, calculating I30 and EI30 for each event separately.

Usage

```
process_storm_events(
  data,
  time_col,
  depth_col,
  interval_col = NULL,
  event_col = NULL,
  min_gap_hours = 6,
  min_rainfall_mm = 12.7,
  calculate_ei30 = TRUE,
  ke_equation = "brown_foster"
)
```

Arguments

`data`	A data frame containing rainfall breakpoint data
`time_col`	Character string specifying the name of the time/datetime column
`depth_col`	Character string specifying the name of the rainfall depth column (in mm)
`interval_col`	Optional character string specifying the name of the time interval column (in minutes)
`event_col`	Optional character string specifying a column that identifies different storm events. If NULL, events will be separated automatically.
`min_gap_hours`	Minimum gap in hours between events for automatic separation (default: 6). Only used if event_col is NULL.
`min_rainfall_mm`	Minimum total rainfall (mm) for a period to be considered an event (default: 12.7, which is 0.5 inches)

`calculate_ei30` Logical indicating whether to calculate EI30 in addition to I30 (default: TRUE)

`ke_equation` Kinetic energy equation to use if calculating EI30

Value

A data frame with one row per storm event containing:

`event_id` Event identifier

`start_time` Event start time

`end_time` Event end time

`duration_min` Event duration in minutes

`total_rainfall_mm` Total rainfall depth

`i30` Maximum 30-minute intensity

`ei30` Erosivity index (if calculate_ei30 = TRUE)

`n_breakpoints` Number of breakpoints in the event

Examples

```
data <- data.frame(
  datetime = as.POSIXct(c("2024-01-01 10:00", "2024-01-01 10:30",
                          "2024-01-01 11:00", "2024-01-01 20:00",
                          "2024-01-01 20:30")),
  rainfall_mm = c(5, 8, 3, 6, 4)
)
results <- process_storm_events(data, "datetime", "rainfall_mm",
                                min_gap_hours = 6, min_rainfall_mm = 1)
print(results)
```

`rainfall_multi` *Multiple Storm Events Rainfall Data*

Description

Hypothetical rainfall breakpoint data containing three separate storm events recorded during August 2023. Events are separated by gaps greater than six hours and differ in intensity, duration, and recording interval, making this dataset suitable for demonstrating multi-event processing.

Usage

```
rainfall_multi
```

Format

A data frame with 22 rows and 2 columns:

datetime POSIXct. Date and time of each breakpoint observation (UTC).

rainfall_mm numeric. Incremental rainfall depth (mm) recorded in the interval ending at the corresponding datetime.

Details

The three storms differ in character:

- **Storm 1** (2023-08-03): Moderate convective storm, 15-minute intervals, ~34 mm total over 2 hours.
- **Storm 2** (2023-08-11): Short intense burst, 10-minute intervals, ~34 mm total over 50 minutes.
- **Storm 3** (2023-08-22): Gentle frontal rainfall, 30-minute intervals, ~19 mm total over 3 hours.

This dataset is intended for demonstrating `process_storm_events()` and comparing erosivity across events of different types.

Source

Hypothetical data generated for package illustration purposes.

Examples

```
data(rainfall_multi)

# Process all storm events
events <- process_storm_events(rainfall_multi, "datetime", "rainfall_mm",
                               min_gap_hours = 6, min_rainfall_mm = 1)
print(events)

# Compare I30 across events
print(events[, c("event_id", "total_rainfall_mm", "i30", "ei30")])
```

`rainfall_single` *Single Storm Event Rainfall Data*

Description

Hypothetical rainfall breakpoint data for a single convective storm event recorded on 2023-07-15, with observations at 15-minute intervals. The storm represents a moderate-intensity summer convective event with a peak near the middle of the storm and a total depth of approximately 38.5 mm.

Usage

```
rainfall_single
```

Format

A data frame with 12 rows and 2 columns:

datetime POSIXct. Date and time of each breakpoint observation (UTC).

rainfall_mm numeric. Incremental rainfall depth (mm) recorded in the 15-minute interval ending at the corresponding datetime.

Details

This dataset is intended for demonstrating and testing the `calculate_i30()`, `calculate_ei30()`, and `validate_rainfall_data()` functions. The rainfall pattern follows a typical bell-shaped hyetograph with intensity peaking in the 14:45–15:00 UTC window.

Source

Hypothetical data generated for package illustration purposes.

Examples

```
data(rainfall_single)

# Calculate I30
result <- calculate_i30(rainfall_single, "datetime", "rainfall_mm")
print(paste("I30 =", result$i30, "mm/hr"))

# Calculate EI30
ei <- calculate_ei30(rainfall_single, "datetime", "rainfall_mm")
print(paste("EI30 =", ei$ei30, "MJ mm ha^-1 hr^-1"))
```

---

validate_rainfall_data

Validate Rainfall Breakpoint Data

Description

Checks rainfall data for common issues including missing values, negative values, non-monotonic time, and duplicate timestamps.

Usage

```
validate_rainfall_data(data, time_col, depth_col, interval_col = NULL)
```

Arguments

`data`	A data frame containing rainfall breakpoint data
`time_col`	Character string specifying the name of the time/datetime column
`depth_col`	Character string specifying the name of the rainfall depth column
`interval_col`	Optional character string specifying the name of the time interval column (in minutes). If NULL, intervals will be calculated from consecutive time differences.

Value

A list with components:

`valid`	Logical indicating whether data passed all checks
`issues`	Character vector describing any issues found
`warnings`	Character vector of non-critical warnings

Examples

```
data <- data.frame(
  time = as.POSIXct(c("2024-01-01 10:00", "2024-01-01 10:15",
                      "2024-01-01 10:30")),
  depth_mm = c(5.2, 3.1, 2.8)
)
result <- validate_rainfall_data(data, "time", "depth_mm")
print(result$valid)
```

Index