Package ‘scenfire’

February 3, 2026

Type Package

Title Post-Processing Algorithm for Integrating Wildfire Simulations

Version 0.1.0

Description A specialized selection algorithm designed to align simulated fire perimeters with specific fire size distribution scenarios. The foundation of this approach lies in generating a vast collection of plausible simulated fires across a wide range of conditions, assuming a random pattern of ignition. The algorithm then assembles individual fire perimeters based on their specific probabilities of occurrence, e.g., determined by (i) the likelihood of ignition and (ii) the probability of particular fire-weather scenarios, including wind speed and direction. Implements the method presented in Rodrigues (2025a) <doi:10.5194/egusphere-egu25-8974>. Demo data and code examples can be found in Rodrigues (2025b) <doi:10.5281/zenodo.15282605>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports dplyr, ggplot2, foreach, methods, readr, sf, terra, tidyr, rlang, stringr, doParallel, tidyselect

Depends R (>= 4.1.0)

Suggests rmarkdown, knitr

VignetteBuilder knitr

NeedsCompilation no

Author Marcos Rodrigues [aut, cre]

Maintainer Marcos Rodrigues `<rmarcos@unizar.es>`

Repository CRAN

Date/Publication 2026-02-03 13:10:01 UTC

Contents

`build_target_hist` *Function to obtain the target histogram from a vector of fire size data*

Description

This function builds a histogram (specifically, its density) from a set of fire sizes, which is intended to represent a target distribution (e.g., from historical fire events). It allows for an optional logarithmic transformation of the fire sizes before binning and determines bin breakpoints dynamically.

Usage

```
build_target_hist(num_bins = 20, logaritmic = TRUE, sizes, event_surfaces)
```

Arguments

`num_bins`	Integer. The desired number of bins for the histogram (default: 20).
`logaritmic`	Logical. If ‘TRUE‘, a logarithmic transformation (‘log(sizes + 1e-6)‘) is applied to ‘sizes‘. If ‘FALSE‘, the original scale is used. This parameter influences both the distribution transformation and the binning.
`sizes`	Numeric vector. The fire sizes (e.g., area in hectares) from the data used to build the target distribution.
`event_surfaces`	Numeric vector. The fire sizes (e.g., area in hectares) from the simulated fire perimeters.

Value

A list containing two elements:

target_hist A numeric vector representing the density values of the target histogram.

bins A numeric vector specifying the breakpoints (edges) of the bins used in the histogram.

Examples

```
# Dummy data for demonstration (replace with your actual data)
set.seed(123)
historical_data_for_target <- floor(fit_powerlaw(n = 500, alpha = 2, xmin = 10, xmax = 10000))
event_surfaces <- fit_powerlaw(n = 10000, alpha = 2, xmin = 10, xmax = 10000)

# Discard simulated fires that are too large (below 110% max historical size)
event_surfaces <- event_surfaces[event_surfaces < max(historical_data_for_target) * 1.1]

# Default configuration: logaritmic transformation on fire size
target_info_example <- build_target_hist(num_bins = 10, logaritmic = TRUE,
                                         sizes = historical_data_for_target,
                                         event_surfaces = event_surfaces)

# Print results
target_hist <- target_info_example$target_hist
print(target_hist)
bins <- target_info_example$bins
print(bins)

# Alternate configuration: original frequency distribution on fire size
target_info_example <- build_target_hist(num_bins = 10, logaritmic = FALSE,
                                         sizes = historical_data_for_target,
                                         event_surfaces = event_surfaces)

# Print results
target_hist <- target_info_example$target_hist
print(target_hist)
bins <- target_info_example$bins
print(bins)
```

---

calculate_discrepancy *Function to calculate the relative distance between the target distribution and the selected perimeters*

Description

Calculates the relative absolute discrepancy between the density histogram of a set of ‘selected_surfaces‘ and a ‘target_hist‘. This metric quantifies how well the distribution of the selected surfaces matches the target distribution. It can optionally apply a logarithmic transformation to the selected surfaces.

Usage

```
calculate_discrepancy(selected_surfaces, target_hist, bins, logaritmic = TRUE)
```

Arguments

`selected_surfaces`
: Numeric vector of surface values for the selected events.

`target_hist`	Numeric vector representing the density values of the target histogram.
`bins`	Numeric vector of bin breakpoints used for both histograms.
`logaritmic`	Logical. If ‘TRUE‘, a logarithmic transformation (‘log(selected_surfaces + 1e-6)‘) is applied to ‘selected_surfaces‘ before calculating its histogram density.

Value

A numeric value representing the relative discrepancy. A value of 0 indicates a perfect match.

Examples

```
# Example target histogram and bins (from build_target_hist)
historical_sizes_ex <- c(10, 50, 100, 200, 500, 1000)
# Dummy 'event_surfaces' for example context (replace with actual data)
event_surfaces <- c(5, 15, 25, 35, 45, 55)
target_info_ex <- build_target_hist(event_surfaces = event_surfaces,
                                    num_bins = 5,
                                    logaritmic = TRUE,
                                    sizes = historical_sizes_ex)
target_hist_ex <- target_info_ex$target_hist
bins_ex <- target_info_ex$bins

# Example selected surfaces
simulated_surfaces_ex <- c(20, 80, 450, 900)

# Calculate discrepancy with logarithmic transformation
discrepancy_log <- calculate_discrepancy(selected_surfaces = simulated_surfaces_ex,
                                         target_hist = target_hist_ex,
                                         bins = bins_ex,
                                         logaritmic = TRUE)
print(paste("Discrepancy (log):", discrepancy_log))

# Example selected surfaces for linear transformation
simulated_surfaces_linear_ex <- c(15, 60, 110, 210, 480, 950)
target_info_linear_ex <- build_target_hist(event_surfaces = event_surfaces,
                                           num_bins = 5,
                                           logaritmic = FALSE,
                                           sizes = historical_sizes_ex)
target_hist_linear_ex <- target_info_linear_ex$target_hist
bins_linear_ex <- target_info_linear_ex$bins

# Calculate discrepancy with linear transformation
discrepancy_linear <- calculate_discrepancy(selected_surfaces = simulated_surfaces_linear_ex,
                                            target_hist = target_hist_linear_ex,
                                            bins = bins_linear_ex,
                                            logaritmic = FALSE)
print(paste("Discrepancy (linear):", discrepancy_linear))
```

calc_abp — *Calculate annual burned probability (BP) from perimeters*

Description

This function calculates the burned probability raster from a set of simulated fire perimeters. It first converts the perimeters into a 'terra' vector object, ensures their validity, and then rasterizes them onto a template raster. The resulting raster indicates the number of times each pixel has been burned, which is then normalized by a scaling factor derived from the total simulated area and a reference surface.

Usage

```
calc_abp(template_raster, candidate_surfaces, reference_surface)
```

Arguments

`template_raster`	A 'terra::SpatRaster' object that defines the extent, resolution, and CRS for the output burned probability raster.
`candidate_surfaces`	An 'sf' object (simple features) representing the simulated fire perimeters. It is assumed to have a column named 'size' containing the area of each perimeter.
`reference_surface`	Numeric value. A reference total surface area used for normalization. This typically represents the target total burned area (e.g., historical average annual burned area).

Value

A 'terra::SpatRaster' object representing the burned probability. Each cell value indicates the proportion of times that pixel was burned, normalized by the ratio of total simulated area to the reference surface.

Examples

```
# Create a dummy template raster
library(terra)
template_raster_ex <- rast(nrows=10, ncols=10, xmin=0, xmax=100, ymin=0, ymax=100,
                           crs="EPSG:25830")

# Create dummy candidate_surfaces (sf object with polygons)
# Ensure 'size' column exists
library(sf)
poly1 <- st_polygon(list(cbind(c(10,10,30,30,10), c(10,30,30,10,10))))
poly2 <- st_polygon(list(cbind(c(50,50,70,70,50), c(50,70,70,50,50))))
poly3 <- st_polygon(list(cbind(c(20,20,40,40,20), c(20,40,40,20,20))))
```

```
candidate_surfaces_ex <- st_sf(
  id = 1:3,
  size = c(400, 400, 400), # Dummy sizes for example
  geometry = st_sfc(poly1, poly2, poly3),
  crs = 25830
)

# Example reference surface
reference_surface_ex <- 1000

# Calculate burned probability
bp_raster <- calc_abp(template_raster = template_raster_ex,
                      candidate_surfaces = candidate_surfaces_ex,
                      reference_surface = reference_surface_ex)

plot(bp_raster, main = "Burned Probability Raster")
```

`check_fire_data` *Check Consistency of Historical vs. Simulated Fire Data*

Description

This function compares historical fire sizes with simulated fire sizes to assess whether the simulations are sufficient in terms of maximum burned area and total burned area. If the simulated fires are not adequate, the function prints guidance messages suggesting additional simulations or adjustments.

Usage

```
check_fire_data(fires_hist_size, sim_perimeters_size, n_years)
```

Arguments

`fires_hist_size`	Numeric vector of historical fire sizes.
`sim_perimeters_size`	Numeric vector of simulated fire sizes.
`n_years`	Integer. Number of years represented in the simulation.

Details

The function checks:

- Maximum fire size in historical vs. simulated data.
- Total burned area in historical vs. simulated data.

If both criteria are satisfied, it calculates a recommended threshold using `get_select_params`.

Value

If the simulated fires are sufficient, returns an integer representing the recommended surface threshold (in number of seasons). If the simulations are not sufficient, prints diagnostic messages and returns nothing.

See Also

`get_select_params`

Examples

```
# Example with toy data
set.seed(123)
hist_sizes <- rgamma(100, shape = 2, scale = 5)
sim_sizes  <- rgamma(200, shape = 2, scale = 4)

check_fire_data(fires_hist_size = hist_sizes,
                sim_perimeters_size = sim_sizes,
                n_years = 10)
```

`cleanse_duplicates` *Remove Duplicate Polygons from Candidate Set*

Description

This function checks a set of candidate polygons and removes duplicates based on spatial equality. It iteratively removes duplicated polygons until none remain.

Usage

```
cleanse_duplicates(candidates)
```

Arguments

`candidates`	An `sf` object containing candidate polygons.

Details

Duplicates are detected using `sf::st_equals()`, which checks for geometric equality between polygons. The function continues removing duplicates until no further matches are found.

Value

An `sf` object with duplicate polygons removed. The function also prints messages to the console indicating whether duplicates were found and their indices.

See Also

st_equals

Examples

```
library(sf)

# Create a simple sf object with duplicate polygons
poly1 <- st_polygon(list(rbind(c(0,0), c(1,0), c(1,1), c(0,1), c(0,0))))
poly2 <- st_polygon(list(rbind(c(0,0), c(1,0), c(1,1), c(0,1), c(0,0)))) # duplicate
poly3 <- st_polygon(list(rbind(c(2,2), c(3,2), c(3,3), c(2,3), c(2,2))))

candidates <- st_sf(geometry = st_sfc(poly1, poly2, poly3))

cleansed <- cleanse_duplicates(candidates)
```

`fit_powerlaw` *Function to build power law distribution to fit a theortical target histogram*

Description

Generates random numbers that follow a truncated power-law distribution. This function uses the inverse cumulative distribution function (CDF) to sample from the specified power-law within given minimum and maximum bounds.

Usage

```
fit_powerlaw(n = 100, alpha = 1.5, xmin, xmax)
```

Arguments

n	The number of random variates (points) to generate (default 100).
alpha	The exponent (alpha parameter) of the power-law distribution (default 1.5).
xmin	The lower bound (minimum value) of the truncated distribution.
xmax	The upper bound (maximum value) of the truncated distribution.

Value

A numeric vector of ‘n‘ random variates sampled from the truncated power-law distribution.

Examples

```
# Generate 100 random numbers from a power-law distribution
# with alpha=2.5, xmin=10, xmax=1000
set.seed(123)
powerlaw_samples <- fit_powerlaw(n = 100, alpha = 2.5, xmin = 10, xmax = 1000)
# hist(powerlaw_samples, main = "Power-Law Distribution Samples")
```

`flp20_to_bp_df` *Process multiple FLP20 files into a combined data frame*

Description

Applies ‘flp20_to_df‘ to a list of file paths and combines the resulting data frames into a single, large data frame. Each file represents individual fire events.

Usage

```
flp20_to_bp_df(files)
```

Arguments

`files`	A character vector of paths to FLP20 .csv files.

Value

A single, combined data frame containing processed data from all input files.

Examples

```
# In a real scenario, 'files' would be paths to your actual FLP20 CSV files.
# For this example, we'll create a few dummy files representing individual fires.

temp_dir <- tempdir()
file1 <- file.path(temp_dir, "flp20_fire1.csv")
file2 <- file.path(temp_dir, "flp20_fire2.csv")

# Create dummy data for file1 (one fire, FL bin 10)
dummy_data1 <- data.frame(
  XPos = 50, YPos = 50, PBurn = 1,
  FIL1 = 0, FIL2 = 0, FIL3 = 0, FIL4 = 0,
  FIL5 = 0, FIL6 = 0, FIL7 = 0, FIL8 = 0,
  FIL9 = 0,
  FIL10 = 1, # This fire is in FL bin 10
  FIL11 = 0, FIL12 = 0, FIL13 = 0, FIL14 = 0,
  FIL15 = 0, FIL16 = 0, FIL17 = 0, FIL18 = 0,
  FIL19 = 0, FIL20 = 0
)
write.csv(dummy_data1, file1, row.names = FALSE)

# Create dummy data for file2 (one fire, FL bin 15)
dummy_data2 <- data.frame(
  XPos = 70, YPos = 80, PBurn = 1,
  FIL1 = 0, FIL2 = 0, FIL3 = 0, FIL4 = 0,
  FIL5 = 0, FIL6 = 0, FIL7 = 0, FIL8 = 0,
  FIL9 = 0,
  FIL10 = 0, FIL11 = 0, FIL12 = 0, FIL13 = 0,
  FIL14 = 0, FIL15 = 1, # This fire is in FL bin 15
  FIL16 = 0, FIL17 = 0, FIL18 = 0, FIL19 = 0, FIL20 = 0
)
write.csv(dummy_data2, file2, row.names = FALSE)

# List of files to process
my_files <- c(file1, file2)

# Process the files
combined_df <- flp20_to_bp_df(my_files)
print(head(combined_df))

# Clean up dummy files
unlink(my_files)
```

`flp20_to_df` *Convert FLP20 file to a data frame*

Description

Reads a .csv file, typically an FLP20 output from FConstMTT, for single fire event simulation, and reshapes the data to long format, calculating flame length (FL) from ‘FIL‘ columns. ‘PBurn‘ is expected to be 1 for individual fires, and ‘FIL‘ columns are binary (1 or 0).

Usage

```
flp20_to_df(file)
```

Arguments

`file`	A character string specifying the path to the FLP20 .csv file.

Value

A data frame with reshaped data, including ‘name‘ (original FIL column index) and ‘FL‘ (calculated flame length).

Examples

```
## Not run:
# Process an FLP20 file
df_result <- flp20_to_df("path/to/your/file.csv")

## End(Not run)
```

`flp20_to_raster` *Convert processed FLP data to raster format and calculate conditional annual burn probability*

Description

Filters a data frame of fire data based on a flame length threshold, groups by spatial position (‘XPos‘, ‘YPos‘), and summarizes it into conditional annual burning probability (CBP) estimates and mean flame length (FL_mean).

Usage

```
flp20_to_raster(df, fl_threshold, selected_surf, reference_surface, r_ref)
```

Arguments

`df`	A data frame processed by ‘flp20_to_df‘ or ‘flp20_to_bp_df‘.
`fl_threshold`	A numeric value representing the minimum flame length (FL).
`selected_surf`	A data frame or list containing a ‘size‘ column.
`reference_surface`	A numeric value representing a reference surface area.
`r_ref`	A terra SpatRaster object.

Value

A list containing two raster objects: ‘CBP‘, ‘CFL‘ and ‘ID_fires‘.

`get_select_params` *Get Selection Parameters for Fire Simulations*

Description

This function calculates a selection parameter that represents how many times the simulated burned area covers, on average, the historical annual burned area.

Usage

```
get_select_params(fires_hist_size, sim_perimeters_size, n_years)
```

Arguments

`fires_hist_size`	Numeric vector of historical fire sizes.
`sim_perimeters_size`	Numeric vector of simulated fire sizes.
`n_years`	Integer. Number of years considered in the historical dataset.

Details

The parameter is calculated as:

$$\text{param} = \left\lfloor \frac{\sum(\text{sim\_perimeters\_size})}{\sum(\text{fires\_hist\_size})/n\_years} \right\rfloor$$

Where:

- $\sum(\text{fires\_hist\_size})/n\_years$ = average historical burned area per year.
- $\sum(\text{sim\_perimeters\_size})$ = total simulated burned area.

Value

An integer value representing the ratio between the total simulated area and the average annual historical burned area. Larger values indicate that the simulation covers several historical fire seasons.

See Also

[check_fire_data]

Examples

```
# Example with toy data
hist_sizes <- c(100, 200, 150, 300)
sim_sizes  <- c(80, 120, 200, 250, 300)

get_select_params(fires_hist_size = hist_sizes,
                  sim_perimeters_size = sim_sizes,
                  n_years = 4)
```

`select_events` *Function to select simulated perimeters*

Description

Selects a subset of simulated fire perimeters by matching their surface distribution to a predefined target histogram. The selection is iterative and probabilistic, aiming to minimize the discrepancy while accumulating a total surface area above a certain threshold (e.g., mean annul burned area).

Usage

```
select_events(
  event_sizes,
  event_probabilities,
  target_hist,
  bins,
  reference_surface,
  surface_threshold,
  tolerance,
  max_it = 5,
  iter_limit = 1e+05,
  logaritmic = TRUE
)
```

Arguments

`event_sizes`	Numeric vector of surface values for all available simulated events.
`event_probabilities`	Numeric vector of probabilities corresponding to each event in ‘event_surfaces‘. These probabilities are used to influence the selection of events within each bin.
`target_hist`	Numeric vector representing the density of the target histogram distribution.
`bins`	Numeric vector of bin breakpoints used for classifying event surfaces and calculating histograms.
`reference_surface`	Numeric value representing the total target surface area that the selected events should approximate.
`surface_threshold`	Numeric value between 0 and 1. The selection process continues until the cumulative surface area of selected events is at least ‘reference_surface * surface_threshold‘.
`tolerance`	Numeric value. A tolerance level for the final discrepancy. (Note: the current implementation finds the best discrepancy, not necessarily stopping once tolerance is met, but aims for the minimum).
`max_it`	Integer for the maximum number of iterations for the inner loop (default: 100).
`iter_limit`	Integer for the maximum number of iterations in selection (default: 100000).
`logaritmic`	Logical. If ‘TRUE‘, a logarithmic transformation is applied to ‘event_surfaces‘ before binning and to ‘selected_surfaces‘ for histogram calculations (default: ‘TRUE‘).

Value

A list containing the best selection found across all iterations:

selected_surfaces Numeric vector of surface values of the events in the best selection.

surface_index Integer vector of the original indices (from ‘event_surfaces‘) of the events in the best selection.

total_surface Numeric value. The sum of surface areas of the events in the best selection.

final_discrepancy Numeric value. The relative discrepancy between the selected events’ distribution and the target histogram for the best selection.

Returns ‘NULL‘ if no valid selection could be made (e.g., no valid results after iterations).

See Also

`Selecting Simulated Fire Events`

`visualize_selected_dist` *Visualize Selected vs. Target Fire Size Distribution*

Description

Visualize Selected vs. Target Fire Size Distribution

Usage

```
visualize_selected_dist(result, logaritmic = TRUE, target_hist, bins)
```

Arguments

`result`	A list returned by a model or selection process.
`logaritmic`	Logical. Defaults to `TRUE`.
`target_hist`	A numeric vector containing the densities of the target distribution.
`bins`	A numeric vector containing the bin limits (breaks) for the intervals.

Value

An object `ggplot`.

Examples

```
## Not run:
# Use example for select_events
visualize_selected_dist(result, logaritmic = TRUE,
                        target_hist = target_hist, bins = bins)

## End(Not run)
```

Index