Package 'burnr'

January 3, 2018

Title Fire-History Analysis in R

Version 0.2.2

Description Basic tools to analyze forest fire history data (e.g. FHX) in R.

URL https://github.com/ltrr-arizona-edu/burnr/

BugReports https://github.com/ltrr-arizona-edu/burnr/issues

Depends R (>= 3.2)

License GPL (>= 3)

LazyData true

Suggests testthat, knitr, rmarkdown

Imports MASS, stats, ggplot2, reshape2, plyr

RoxygenNote 6.0.1

VignetteBuilder knitr

NeedsCompilation no

Author Steven Malevich [aut, cre],
Christopher Guiterman [ctb],
Ellis Margolis [ctb]

Maintainer Steven Malevich <malevich@email.arizona.edu>

Repository CRAN

Date/Publication 2018-01-03 00:59:45 UTC

R topics documented:

`+.fhx` *Concatenate or combine two fhx objects.*

Description

Concatenate or combine two fhx objects.

Usage

```
## S3 method for class 'fhx'
a + b
```

Arguments

`a`	An fhx object.
`b`	The fhx object to be append.

Value

An fhx object with the series from `a` and `b`.

Examples

```
data(lgr2)
data(pgm)
plot(lgr2 + pgm)
```

`check_duplicates` *Check for duplicate observations in an fhx object.*

Description

Check for duplicate observations in an fhx object.

Usage

```
check_duplicates(x)
```

Arguments

`x`	An fhx object.

Value

A `x` or stop() is thrown.

Examples

```
data(lgr2)
data(pgm)
burnr:::check_duplicates(lgr2 + pgm)
```

`composite` *Composite fire events in fhx object returning composited object with prominent fires.*

Description

Composite fire events in fhx object returning composited object with prominent fires.

Usage

```
composite(x, filter_prop = 0.25, filter_min_rec = 2,
  filter_min_events = 1, injury_event = FALSE, comp_name = "COMP")
```

Arguments

`x`	An fhx instance.
`filter_prop`	The proportion of fire events to recording series needed in order to be considered. Default is 0.25.
`filter_min_rec`	The minimum number of recording series needed to be considered a fire event. Default is 2 recording series.
`filter_min_events`	The minimum number of fire scars needed to be considered a fire event. Default is 1. This includes injuries if injury_event=TRUE.
`injury_event`	Boolean indicating whether injuries should be considered events. Default is FALSE.
`comp_name`	Character vector of the series name for the returned fhx object composite series. Default is 'COMP'.

Value

An fhx object representing the composited series.

Examples

```
data(lgr2)
composite(lgr2)

# Use with composite to get composite years:
comp <- composite(pgm, comp_name = 'pgm')
event_yrs <- get_event_years(comp)[['pgm']]
print(event_yrs)
```

count_event_position *Count of different events*

Description

Count of different events

Usage

```
count_event_position(x, injury_event = FALSE, position, groupby)
```

Arguments

x	An fhx object.
injury_event	Optional boolean indicating whether injuries should be considered event. Default is FALSE.
position	Optional character vector giving the types of event positions to include in the count. Can any combination of the following: "unknown", "dormant", "early", "middle", "late", "latewd". The default counts all event positions.
groupby	Optional named list containing character vectors that are used to count the total number of different event types. The names given to each character vector give the group's name in the output data.frame.

Value

A data.frame with a columns giving the event and corresponding number of events for each event type.

Examples

```
data(pgm)
count_event_position(pgm)

# As above, but considering injuries to be a type of event.
count_event_position(pgm, injury_event = TRUE)

# Count only events of a certain position, in this case, "unknown", "early", and "middle".
count_event_position(pgm, injury_event = TRUE, position = c("unknown", "early", "middle"))

# Using custom `groupby` args.
grplist <- list(foo = c("dormant_fs", "early_fs"), bar = c("middle_fs", "late_fs"))
count_event_position(pgm, groupby = grplist)
```

count_injury *Number of injury events in an fhx series.*

Description

Number of injury events in an fhx series.

Usage

```
count_injury(x)
```

Arguments

x	An fhx object.

Value

The number of injury events observed in the series.

count_recording *Number of recording years in an fhx series.*

Description

Number of recording years in an fhx series.

Usage

```
count_recording(x, injury_event = FALSE)
```

Arguments

x	An fhx object.
injury_event	Boolean indicating whether injuries should be considered event.

Value

The number of recording events observed in the series.

count_scar *Number of scar events in an fhx series.*

Description

Number of scar events in an fhx series.

Usage

```
count_scar(x)
```

Arguments

x	An fhx object.

Value

The number of fire events observed in the series.

count_year_span *Number of years of an fhx series.*

Description

Number of years of an fhx series.

Usage

```
count_year_span(x)
```

Arguments

x	An fhx object.

Value

The difference between the first and last observations in the series. 'NA' will be returned if 'NA' is in 'x$year'.

delete *Remove series or years from an fhx object.*

Description

Remove series or years from an fhx object.

Usage

```
delete(x, s, yr)
```

Arguments

x	An fhx object.
s	Character vector of series to erase from x.
yr	Integer vector of years to erase from x.

Details

You can combine s and yr to specify years within select series to remove.

Value

An fhx object with observations erased.

Examples

```
data(lgr2)
plot(delete(lgr2, s = 'LGR46'))

plot(delete(lgr2, yr = 1300:1550))
```

fhx *Constructor for S3 fhx class.*

Description

Constructor for S3 fhx class.

Usage

```
fhx(year, series, rec_type, metalist = list())
```

Arguments

year	A numeric vector of observation years for each series and rec_type argument.
series	A factor of series names for each year and rec_type argument.
rec_type	A factor of ring types for each element in year and series.
metalist	An option list of arbitrary metadata to be included in the fhx instance.

Value

An fhx instance.

---

find_recording *Subset 'rings' data.frame to years that are considered recording.*

Description

Subset 'rings' data.frame to years that are considered recording.

Usage

```
find_recording(x, injury_event)
```

Arguments

x	A an fhx object dataframe.
injury_event	Boolean indicating whether injuries should be considered event.

Value

A dataframe with a column of each year which is 'recording'.

Examples

```
require(plyr)
data(lgr2)
ddply(lgr2$rings, 'series', burnr:::find_recording, injury_event = TRUE)
```

first_year *First (earliest) year of an fhx series.*

Description

First (earliest) year of an fhx series.

Usage

```
first_year(x)
```

Arguments

x	An fhx object.

Value

The minimum or first year of series in 'x'.

get_event_years *Get years with events for an fhx object.*

Description

Get years with events for an fhx object.

Usage

```
get_event_years(x, scar_event = TRUE, injury_event = FALSE,
  custom_grep_str = NULL)
```

Arguments

x	An fhx object.
scar_event	Boolean indicating whether years with scar events should be returned. Default is TRUE.
injury_event	Boolean indicating weather years with injury events should be returned. Default is FALSE.
custom_grep_str	Character string to pass a custom grep search pattern to search rec_type column for. Undefined by default.

Value

A list. Elements of the list are integer vectors giving the years with events for each fhx series. Each element's name reflects the series name.

Examples

```
data(pgm)
get_event_years(pgm, scar_event = TRUE, injury_event = TRUE)

# Passing a custom string to grep. This one identified recorder years:
get_event_years(pgm, custom_grep_str = 'recorder_')

# Use with composite to get composite years:
comp <- composite(pgm, comp_name = 'pgm')
event_yrs <- get_event_years(comp)[['pgm']]
print(event_yrs)
```

---

get_ggplot *Create a ggplot2 object for plotting.*

Description

This function is depreciated. Please use 'plot_demograph()'.

Usage

```
get_ggplot(...)
```

Arguments

...	Arguments passed on to plot_demograph.

---

get_series *Extract fhx observations for given series.*

Description

Extract fhx observations for given series.

Usage

```
get_series(x, s)
```

Arguments

x	An fhx object.
s	Character vector of series you would like extracted from x.

Value

A dataframe with extracted observations.

Examples

```
data(lgr2)
get_series(lgr2, 'LGR46')

get_series(lgr2, c('LGR41', 'LGR46'))
```

`get_year` *Extract fhx observations for given years.*

Description

Extract fhx observations for given years.

Usage

```
get_year(x, yr)
```

Arguments

x	An fhx object.
yr	Integer vector of year(s) you would like extracted from x.

Value

A dataframe with extracted observations.

Examples

```
data(lgr2)
get_year(lgr2, 1806)

get_year(lgr2, 1805:1807)
```

`inner_type` *Type of observation in the first (earliest) year of an fhx series.*

Description

Type of observation in the first (earliest) year of an fhx series.

Usage

```
inner_type(x)
```

Arguments

x	An fhx object.

Value

The a factor giving the type of observation in the first observation of the series.

---

intervals *Constructor for S3 intervals class.*

Description

Constructor for S3 intervals class.

Usage

```
intervals(comp, densfun = "weibull")
```

Arguments

comp	A composite fhx instance. Should have only one series in it.
densfun	String giving desired distribution to fit. Suggest "weibull" or "lognormal". Default is "weibull".

Value

An intervals instance.

Examples

```
data(pgm)
interv <- intervals(composite(pgm))
print(interv)

mean(interv)  # Mean interval

# Now fit log-normal distribution instead of Weibull.
intervals(composite(pgm), densfun = "lognormal")

## Not run:
# Boxplot of fire interval distribution.
boxplot(intervals(composite(pgm))$intervals)

## End(Not run)
```

is.fhx *Check if object is fhx.*

Description

Check if object is fhx.

Usage

```
is.fhx(x)
```

Arguments

x	Any R object.

Value

Boolean indicating whether ‘x‘ is an fhx object.

Examples

```
data(lgr2)
is.fhx(lgr2)
```

is.intervals *Check if object is intervals.*

Description

Check if object is intervals.

Usage

```
is.intervals(x)
```

Arguments

x	An R object.

Value

Boolean indicating whether ‘x‘ is an intervals object.

is.sea *Check if object is sea.*

Description

Check if object is sea.

Usage

```
is.sea(x)
```

Arguments

x	An R object.

Value

Boolean indicating whether ‘x‘ is an sea object.

last_year *Last (most recent) year of an fhx series.*

Description

Last (most recent) year of an fhx series.

Usage

```
last_year(x)
```

Arguments

x	An fhx object.

Value

The maximum or last year of series in ’x’. ’NA’ will be returned if ’NA’ is in x$year.

lgr2 *Los Griegos Peak plot2 fire-history data*

Description

An fhx object with fire-history data from Los Griegos Peak, New Mexico.

Usage

```
lgr2
```

Format

An fhx object with 26 series from 1366 to 2012 CE. Each series is a string with 5 characters.

lgr2_meta *Metadata for the Los Griegos Peak fire-history dataset*

Description

A dataset with species information for the Los Griegos Peak plot2 fire-history dataset (‘lgr2‘).

Usage

```
lgr2_meta
```

Format

A data frame with 26 rows and 2 variables:

- TreeID: name of tree series
- SpeciesID: abbreviated tree species

list_filestrings *List of character strings to write to FHX file.*

Description

List of character strings to write to FHX file.

Usage

```
list_filestrings(x)
```

Arguments

x	An fhx object.

Value

A list with four members containing vectors: "head_line", "subhead_line", "series_heading", and "body". Each referring to a portion of an FHX file that the strings are dumped into.

See Also

write_fhx

make_rec_type *Turn character vector into factor with proper fhx levels.*

Description

Turn character vector into factor with proper fhx levels.

Usage

```
make_rec_type(x)
```

Arguments

x	A character vector containing one or more rec_type-like strings.

Value

A factor with appropriate fhx levels.

Examples

```
make_rec_type('null_year')

make_rec_type(c('null_year', 'late_fs'))
```

max.intervals *Maximum interval.*

Description

Maximum interval.

Usage

```
## S3 method for class 'intervals'
max(x, ...)
```

Arguments

x	An intervals object.
...	Additional arguments passed to max.

Value

Numeric or NA.

mean.intervals *Interval arithmetic mean.*

Description

Interval arithmetic mean.

Usage

```
## S3 method for class 'intervals'
mean(x, ...)
```

Arguments

x	An intervals object.
...	Additional arguments passed to mean.

Value

Numeric or NA.

median.intervals *Interval median.*

Description

Interval median.

Usage

```
## S3 method for class 'intervals'
median(x, ...)
```

Arguments

x	An intervals object.
...	Additional arguments passed to median.

Value

Numeric or NA.

min.intervals *Minimum interval.*

Description

Minimum interval.

Usage

```
## S3 method for class 'intervals'
min(x, ...)
```

Arguments

x	An intervals object.
...	Additional arguments passed to min.

Value

Numeric or NA.

outer_type *Type of observation in the last (most recent) year of an fhx series.*

Description

Type of observation in the last (most recent) year of an fhx series.

Usage

```
outer_type(x)
```

Arguments

x	An fhx object.

Value

The a factor giving the type of observation in the last observation of the series.

pgm *Peggy Mesa fire-history data*

Description

An fhx object with fire-history data from Peggy Mesa. See dataset 'pgm_meta' for metadata. Data from Guiterman, Christopher H., Ellis Q. Margolis, and Thomas W. Swetnam. 2015. "Dendroecological Methods For Reconstructing High-Severity Fire In Pine-Oak Forests." Tree-Ring Research 71 (2): 67-77. doi:10.3959/1536-1098-71.2.67.

Usage

```
pgm
```

Format

An fhx object with 41 series from 1555 to 2013 CE. Each series is a string with 5 characters.

pgm_meta *Metadata for the Peggy Mesa fire-history dataset*

Description

A dataset with species and location information for the Peggy Mesa fire-history dataset (‘pgm‘). Data from Guiterman, Christopher H., Ellis Q. Margolis, and Thomas W. Swetnam. 2015. "Dendroecological Methods For Reconstructing High-Severity Fire In Pine-Oak Forests." Tree-Ring Research 71 (2): 67-77. doi:10.3959/1536-1098-71.2.67.

Usage

```
pgm_meta
```

Format

A data frame with 41 rows and 5 variables:

- TreeID: name of tree series
- SpeciesID: abbreviated tree species
- Latitude: latitude of tree in decimal degrees
- Longitude: longitude of tree in decimal degrees
- Elevation: tree elevation in meters

pgm_pdsi *Reconstructed PDSI time series for the Peggy Mesa fire-history dataset*

Description

A tree-ring reconstructed Palmer Drought-Severity Index time series corresponding to the Peggy Mesa fire-history dataset (‘pgm‘) – specifically, the Jemez Mountains area (gridpoint 133). The reconstruction is from The North American Drought Atlas (Cook and Krusic 2004).

Usage

```
pgm_pdsi
```

Format

A data frame with 2004 rows and 1 variables. Row names give the year for the reconstructed value:

- RECON: The reconstructed PDSI series.

plot.fhx *Plot an fhx object.*

Description

Plot an fhx object.

Usage

```
## S3 method for class 'fhx'
plot(...)
```

Arguments

...	Arguments passed on to plot_demograph.

Examples

```
data(lgr2)
plot(lgr2)

plot(lgr2, ylabels = FALSE, plot_legend = TRUE)

data(lgr2_meta)
# With color showing species.
plot(lgr2,
     color_group = lgr2_meta$SpeciesID,
     color_id = lgr2_meta$TreeID,
     plot_legend = TRUE)
# With facets for each species.
plot(lgr2,
     facet_group = lgr2_meta$SpeciesID,
     facet_id = lgr2_meta$TreeID,
     plot_legend = TRUE)

# Append annotation onto a ggplot object.
require(ggplot2)
p <- plot_demograph(lgr2,
                color_group = lgr2_meta$SpeciesID,
                color_id = lgr2_meta$TreeID)
# Add transparent box as annotation to plot.
p + annotate('rect',
             xmin = 1750, xmax = 1805,
             ymin = 3.5, ymax = 13.5, alpha = 0.2)
```

plot.intervals *Plot an intervals object.*

Description

Plot an intervals object.

Usage

```
## S3 method for class 'intervals'
plot(...)
```

Arguments

...	Arguments passed on to plot_intervals_dist.

Examples

```
data(pgm)
interv <- intervals(composite(pgm))

plot(interv, binwidth = 5)
```

plot.sea *Plot a sea object.*

Description

Plot a sea object.

Usage

```
## S3 method for class 'sea'
plot(...)
```

Arguments

...	Arguments passed on to plot_sealags.

Examples

```
## Not run:
# Read in the Cook and Krusic (2004; The North American Drought Atlas) reconstruction
# of Palmer Drought Severity Index (PDSI) for the Jemez Mountains area (gridpoint 133).

data(pgm_pdsi)

# Run SEA on Peggy Mesa (pgm) data
data(pgm)
pgm_comp <- composite(pgm)

pgm_sea <- sea(pgm_pdsi, pgm_comp)

plot(pgm_sea)

## End(Not run)
```

`plot_demograph` *Create an ggplot2 object for plotting fhx demographics.*

Description

Create an ggplot2 object for plotting fhx demographics.

Usage

```
plot_demograph(x, color_group, color_id, facet_group, facet_id,
  facet_type = "grid", ylabels = TRUE, yearlims = FALSE,
  composite_rug = FALSE, filter_prop = 0.25, filter_min_rec = 2,
  filter_min_events = 1, injury_event = FALSE, plot_legend = FALSE,
  event_size = c(Scar = 4, Injury = 2, `Pith/Bark` = 1.5),
  rugbuffer_size = 2, rugdivide_pos = 2)
```

Arguments

`x`	An `fhx` instance.
`color_group`	Option to plot series with colors. This is a character vector or factor which corresponds to the series names given in `color_id`. Both `color_group` and `color_id` need to be specified. Default plot gives no color.
`color_id`	Option to plot series with colors. A character vector of series names corresponding to groups given in `color_group`. Every unique value in `x` series.names needs to have a corresponding color_group value. Both `color_group` and `color_id` need to be specified. Default plot gives no species colors.
`facet_group`	Option to plot series with faceted by a factor. A vector of factors or character vector which corresponds to the series names given in `facet_id`. Both `facet_group` and `facet_id` need to be specified. Default plot is not faceted.

facet_id	Option to plot series with faceted by a factor. A vector of series names corresponding to species names given in facet_group. Every unique values in x series.names needs to have a corresponding facet_group value. Both facet_group and facet_id need to be specified. Default plot is not faceted. Note that composite_rug and facet_group, facet_id cannot be used in the same plot. You must choose facets or a composite rug.
facet_type	Type of ggplot2 facet to use, if faceting. Must be either "grid" or "wrap". Default is "grid". Note that composite_rug and facet_group, facet_id cannot be used in the same plot. You must choose facets or a composite rug.
ylabels	Optional boolean to remove y-axis (series name) labels and tick marks. Default is TRUE.
yearlims	Option to limit the plot to a range of years. This is a vector with two integers. The first integer gives the lower year for the range while the second integer gives the upper year. The default is to plot the full range of data given by x.
composite_rug	A boolean option to plot a rug on the bottom of the plot. Default is FALSE. Note that composite_rug and facet_group, facet_id cannot be used in the same plot. You must choose facets or a composite rug.
filter_prop	An optional argument if the user chooses to include a composite rug in their plot. This is passed to composite. See this function for details.
filter_min_rec	An optional argument if the user chooses to include a composite rug in their plot. This is passed to composite. See this function for details.
filter_min_events	An optional argument if the user chooses to include a composite rug in their plot. This is passed to composite. See this function for details.
injury_event	Boolean indicating whether injuries should be considered recorders. This is passed to composite. See this function for details.
plot_legend	A boolean option allowing the user to choose whether a legend is included in the plot or not. Default is FALSE.
event_size	An optional numeric vector that adjusts the size of fire event symbols on the plot. Default is c("Scar" = 4, "Injury" = 2, "Pith/Bark" = 1.5).
rugbuffer_size	An optional integer. If the user plots a rug, this controls the amount of buffer whitespace along the y-axis between the rug and the main plot. Must be >= 2.
rugdivide_pos	Optional integer if plotting a rug. Adjust the placement of the rug divider along the y-axis. Default is 2.

Value

A ggplot object for plotting or manipulation.

Examples

```
data(lgr2)
plot(lgr2)

plot(lgr2, ylabels = FALSE, plot_legend = TRUE)
```

```
data(lgr2_meta)
# With color showing species.
plot(lgr2,
     color_group = lgr2_meta$SpeciesID,
     color_id = lgr2_meta$TreeID,
     plot_legend = TRUE)
# With facets for each species.
plot(lgr2,
     facet_group = lgr2_meta$SpeciesID,
     facet_id = lgr2_meta$TreeID,
     plot_legend = TRUE)

# Append annotation onto a ggplot object.
require(ggplot2)
p <- plot_demograph(lgr2,
                color_group = lgr2_meta$SpeciesID,
                color_id = lgr2_meta$TreeID)
# Add transparent box as annotation to plot.
p + annotate('rect',
             xmin = 1750, xmax = 1805,
             ymin = 3.5, ymax = 13.5, alpha = 0.2)
```

`plot_intervals_dist` *Basic intervals distribution plot.*

Description

Basic intervals distribution plot.

Usage

```
plot_intervals_dist(x, binwidth = NULL)
```

Arguments

`x`	An intervals object.
`binwidth`	A sea object.

Value

A ggplot object.

plot_sealags *Basic SEA lag plot.*

Description

Basic SEA lag plot.

Usage

```
plot_sealags(x)
```

Arguments

x	A sea object.

Value

A ggplot object.

print.intervals *Print an intervals objects.*

Description

Print an intervals objects.

Usage

```
## S3 method for class 'intervals'
print(x, ...)
```

Arguments

x	An intervals object.
...	Additional arguments that are tossed.

print.sea *Print an sea objects.*

Description

Print an sea objects.

Usage

```
## S3 method for class 'sea'
print(x, ...)
```

Arguments

x	An intervals object.
...	Additional arguments that are tossed.

quantile.intervals *Fit distribution quantiles.*

Description

Fit distribution quantiles.

Usage

```
## S3 method for class 'intervals'
quantile(x, q = c(0.125, 0.5, 0.875), ...)
```

Arguments

x	An intervals object.
q	Vector giving the desired quantiles.
...	Additional arguments passed to the quantile function of the fit distribution.

Examples

```
data(pgm)
intervs <- intervals(composite(pgm))
quantile(intervs)

# Or you can pass in your own quantiles:
quantile(intervs, q = c(0.25, 0.5, 0.75))
```

read_fhx *Read FHX2 file and return an* fhx *object.*

Description

Read FHX2 file and return an fhx object.

Usage

```
read_fhx(fname, encoding, text)
```

Arguments

fname	Name of target FHX file. Needs to be in format version 2.
encoding	Encoding to use when reading the FHX file. The default is to use the system.
text	Character string. If fname is not provided and text is, then data is read from text using a text connection.

Value

An fhx object.

Examples

```
## Not run:
d <- read_fhx('afile.fhx')

## End(Not run)
```

run_sea *Perform superposed epoch analysis.*

Description

Perform superposed epoch analysis.

Usage

```
run_sea(x, key, years_before = 6, years_after = 4, key_period = TRUE,
  n_iter = 1000)
```

Arguments

x	A data.frame climate reconstruction or tree-ring series with row names as years.
key	A vector of event years for superposed epoch, such as fire years, or an fhx object with a single series as produced by composite
years_before	The number of lag years prior to the event year
years_after	The number of lag years following the event year
key_period	Logical. Constrains the time series to the time period of key events within the range of the x climate series. False uses the entire climate series, ignoring the period of key events. time series
n_iter	The number of iterations for bootstrap resampling

Details

Superposed epoch analysis (SEA) helps to evaluate fire-climate relationships in studies of tree-ring fire history. It works by compositing the values of an annual time series or climate reconstruction for the fire years provided (key) and both positive and negative lag years. Bootstrap resampling of the timeseries is performed to evaluate the statistical significance of each year's mean value. Users interpret the departure of the actual event year means from the simulated event year means.

The significance of lag-year departures from the average climate condition was first noted by Baisan and Swetnam (1990) and used in an organized SEA by Swetnam (1993). Since then, the procedure has been commonly applied in fire history studies. The FORTRAN program EVENT.exe was written by Richard Holmes and Thomas Swetnam (Holmes and Swetnam 1994) to perform SEA for fire history specifically. EVENT was incorporated in the FHX2 software by Henri Grissino-Mayer.

run_sea was designed to replicate EVENT as closely as possible. We have tried to stay true to their implementation of SEA, although multiple versions of the analysis exist in the climate literature and for fire history (e.g., FHAES implements a different procedure). The outcome of EVENT and run_sea should only differ slightly in the values of the simulated events and the departures, because random draws are used. The event year and lag significance levels should match, at least in the general pattern.

We note that our implementation of run_sea borrows from the dplR::sea function in how it performs the bootstrap procedure, but differs in the kind of output provided for the user.

Value

A list of three data frames, following the output of EVENT. (1) the actual events table, (2) the simulated events table, and (3) departures of actual from simulated

References

Baisan and Swetnam 1990, Fire history on desert mountain range: Rincon Mountain Wilderness, Arizona, U.S.A. Canadian Journal of Forest Research 20:1559-1569.

Bunn 2008, A dendrochronology program library in R (dplR), Dendrochronologia 26:115-124

Holmes and Swetnam 1994, EVENT program description

Swetnam 1993, Fire history and climate change in giant sequoia groves, Science 262:885-889.

Examples

```
## Not run:
# Read in the Cook and Krusic (2004; The North American Drought Atlas) reconstruction
# of Palmer Drought Severity Index (PDSI) for the Jemez Mountains area (gridpoint 133).
target_url <- paste0('http://iridl.ldeo.columbia.edu',
                     '/SOURCES/.LDEO/.TRL/.NADA2004'
                     '/pdsiatlashtml/pdsiwebdata/1050w_350n_133.txt')
pdsi <- read.table(target_url, header = TRUE, row.names = 1)
pdsi <- subset(pdsi, select = "RECON")

# Run SEA on Peggy Mesa (pgm) data
data(pgm)
(pgm.comp <- composite(pgm))

(pgm.sea <- run_sea(pdsi, pgm.comp))

# Make a bargraph with confidence intervals
par(mar=c(2, 3, 1, 1), oma=c(3, 3, 1, 1))
bp <- barplot(pgm.sea[[3]]$mean_value,
              col=c(rep("grey75", 3),"grey45", "grey30",
                    "grey75", "grey30", rep("grey75", 4)),
              ylab = '', las=1, cex.axis=1.3, cex=1.3, ylim=c(-2, 2))
axis(1, at=bp, labels = -6:4, tick=FALSE, cex.axis=1.3)
lines(bp, pgm.sea[[3]]$lower_95_perc, lwd=2, lty=2)
lines(bp, pgm.sea[[3]]$upper_95_perc, lwd=2, lty=2)
lines(bp, pgm.sea[[3]]$lower_99_perc, lwd=2, lty=3)
lines(bp, pgm.sea[[3]]$upper_99_perc, lwd=2, lty=3)
mtext(expression(bold('PDSI departure')), side=2, line=2.2, cex=1.5)
mtext(expression(bold('Lag year')), side=1, line=3.3, cex=1.5)

## End(Not run)
## Not run:
# For users who want to perform SEA very near to EVENT.exe and/or have reproducable draws from
# the bootstrap procedure, consider including the \code{set.seed} function prior to \code{run_sea}.
# Convention is to provide a long integer, such as a birthday (e.g. 3191982).
# In the EVENT.exe program, Richard Holmes used the number of days since 1 January 1935.
days <- as.numeric(Sys.Date() - as.Date("1jan1935", "%d%b%Y"))
set.seed(days)

## End(Not run)
```

---

sample_depth *Calculate the sample depth of an fhx object*

Description

Calculate the sample depth of an fhx object

Usage

```
sample_depth(a)
```

Arguments

a	An fhx object.

Value

A data.frame containing the years and number of trees

---

sea *Perform superposed epoch analysis.*

Description

Perform superposed epoch analysis.

Usage

```
sea(x, event, nbefore = 6, nafter = 4, event_range = TRUE,
  n_iter = 1000)
```

Arguments

x	A data.frame climate reconstruction or tree-ring series with row names as years.
event	A vector of event years for superposed epoch, such as fire years, or an fhx object with a single `series` as produced by `composite`
nbefore	The number of lag years prior to the event year
nafter	The number of lag years following the event year
event_range	Logical. Constrains the time series to the time period of key events within the range of the x climate series. False uses the entire climate series, ignoring the period of key events. time series
n_iter	The number of iterations for bootstrap resampling

Details

Superposed epoch analysis (SEA) helps to evaluate fire-climate relationships in studies of tree-ring fire history. It works by compositing the values of an annual timeseries or climate reconstruction for the fire years provided (`key`) and both positive and negative lag years. Bootstrap resampling of the timeseries is performed to evaluate the statistical significance of each year's mean value. Users interpret the departure of the actual event year means from the simulated event year means.

The significance of lag-year departures from the average climate condition was first noted by Baisan and Swetnam (1990) and used in an organized SEA by Swetnam (1993). Since then, the procedure

has been commonly applied in fire history studies. The FORTRAN program EVENT.exe was written by Richard Holmes and Thomas Swetnam (Holmes and Swetnam 1994) to perform SEA for fire history specifically. EVENT was incorporated in the FHX2 software by Henri Grissino-Mayer.

sea was designed to replicate EVENT as closely as possible. We have tried to stay true to their implementation of SEA, although multiple versions of the analysis exist in the climate literature and for fire history (e.g., FHAES implements a different procedure). The outcome of EVENT and sea should only differ slightly in the values of the simulated events and the departures, because random draws are used. The event year and lag significance levels should match, at least in the general pattern.

We note that our implementation of run_sea borrows from the `dplR::sea` function in how it performs the bootstrap procedure, but differs in the kind of output provided for the user.

Value

A list of three data frames, following the output of EVENT. (1) the actual events table, (2) the simulated events table, and (3) departures of actual from simulated

References

Baisan and Swetnam 1990, Fire history on desert mountain range: Rincon Mountain Wilderness, Arizona, U.S.A. Canadian Journal of Forest Research 20:1559-1569.

Bunn 2008, A dendrochronology program library in R (dplR), Dendrochronologia 26:115-124

Holmes and Swetnam 1994, EVENT program description

Swetnam 1993, Fire history and climate change in giant sequoia groves, Science 262:885-889.

Examples

```
## Not run:
# Read in the Cook and Krusic (2004; The North American Drought Atlas) reconstruction
# of Palmer Drought Severity Index (PDSI) for the Jemez Mountains area (gridpoint 133).

## End(Not run)
## Not run:
# For users who want to perform SEA very near to EVENT.exe and/or have reproducable draws from
# the bootstrap procedure, consider including the \code{set.seed} function prior to \code{run_sea}.
# Convention is to provide a long integer, such as a birthday (e.g. 3191982).
# In the EVENT.exe program, Richard Holmes used the number of days since 1 January 1935.
days <- as.numeric(Sys.Date() - as.Date("1jan1935", "%d%b%Y"))
set.seed(days)

## End(Not run)
```

series_mean_interval *Calculate mean fire interval of a single fhx series.*

Description

Calculate mean fire interval of a single fhx series.

Usage

```
series_mean_interval(x, injury_event = FALSE)
```

Arguments

x	An fhx object with a single series. For proper fire intervals see 'intervals()'.
injury_event	Boolean indicating whether injuries should be considered event.

Value

The mean fire interval observed in the series.

See Also

intervals()

series_names *Get* fhx *series names.*

Description

Get fhx series names.

Usage

```
series_names(x)
```

Arguments

x	An fhx object.

Value

A character vector or NULL.

Examples

```
data(lgr2)
series_names(lgr2)
```

series_stats *Generate series-level descriptive statistics.*

Description

Generate series-level descriptive statistics.

Usage

```
series_stats(x, func_list = list(first = first_year, last = last_year, years =
  count_year_span, inner_type = inner_type, outer_type = outer_type,
  number_scars = count_scar, number_injuries = count_injury, recording_years =
  count_recording, mean_interval = series_mean_interval))
```

Arguments

x	An fhx object.
func_list	A list of named functions that will be run on each series in the fhx object. The list name for each function is the corresponding column name in the output data.frame.

Value

A data.frame containing series-level statistics.

Examples

```
data(lgr2)
series_stats(lgr2)

# You can create your own list of statistics to output. You can also create
# your own functions:
flist <- list(n = count_year_span,
              xbar_interval = function(x) mean_interval(x, injury_event = TRUE))
sstats <- series_stats(lgr2)
head(sstats)
```

site_stats *Generate site-level summary statistics*

Description

Generate site-level summary statistics

Usage

```
site_stats(x, site_name = "XXX", year_range = NULL, filter_prop = 0.25,
  filter_min_rec = 2, filter_min_events = 1, injury_event = FALSE)
```

Arguments

x	An fhx object
site_name	Three character site code, defaults to "XXX"
year_range	Delimits the analysis period. For example, c(1600, 1900).
filter_prop	An optional argument if the user chooses to include a composite rug in their plot. This is passed to composite. See this function for details.
filter_min_rec	An optional argument if the user chooses to include a composite rug in their plot. This is passed to composite. See this function for details.
filter_min_events	An optional argument if the user chooses to include a composite rug in their plot. This is passed to composite. See this function for details.
injury_event	Boolean indicating whether injuries should be considered recorders. This is passed to composite. See this function for details.

Details

This function produces a summary table for any fhx object. The statistics it includes are shared by other popular fire history software such as FHX2 and FHAES.

Value

A data.frame of summary statistics

---

sort.fhx *Sort the series names of fhx object by the earliest or latest year.*

Description

Sort the series names of fhx object by the earliest or latest year.

Usage

```
## S3 method for class 'fhx'
sort(x, decreasing = FALSE, sort_by = "first_year", ...)
```

Arguments

x	An fhx instance to be sorted.
decreasing	Logical. Decreasing sorting? Defaults to FALSE.
sort_by	Either 'first_year' or 'last_year'. Designates the inner or outer year for sorting. Defaults to 'first_year'
...	Additional arguments that fall off the face of the universe.

Value

A copy of x with reordered series.

Examples

```
data(lgr2)
plot(sort(lgr2, decreasing = TRUE))
plot(sort(lgr2, sort_by = "last_year"))
```

---

summary.fhx *Summary of 'fhx' object*

Description

Summary of 'fhx' object

Usage

```
## S3 method for class 'fhx'
summary(object, ...)
```

Arguments

object	An fhx object.
...	Additional arguments.

Value

A summary.fhx object.

---

write_fhx *Write an fhx object to a new FHX2 file.*

Description

Write an fhx object to a new FHX2 file.

Usage

```
write_fhx(x, fname = "")
```

Arguments

x	An fhx object.
fname	Output filename.

Examples

```
## Not run:
data(lgr2)
write_fhx(lgr2, 'afile.fhx')

## End(Not run)
```

`yearly_recording` *Count the number of recording series for each year in an fhx object.*

Description

Count the number of recording series for each year in an fhx object.

Usage

```
yearly_recording(x, injury_event = FALSE)
```

Arguments

`x`	An fhx object.
`injury_event`	Boolean indicating whether injuries should be considered event. Default is FALSE.

Value

A dataframe with a columns giving the year and corresponding number of recording events for that year.

Examples

```
data(lgr2)
yearly_recording(lgr2)
```

year_range *Range of years for* fhx *object.*

Description

Range of years for fhx object.

Usage

```
year_range(x)
```

Arguments

x	An fhx object.

Value

An integer vector or NULL.

Examples

```
data(lgr2)
year_range(lgr2)
```

Index