Package: growthTrendR (via r-universe)

May 31, 2026

Type Package

Title Toolkit for Data Processing, Quality, and Statistical Models

Version 0.3.0.9000

Date 2026-04-19

Maintainer Xiao Jing Guo <xiaojing.guo@nrcan-rncan.gc.ca>

Description Offers tools for data formatting, anomaly detection, and classification of tree-ring data using spatial comparisons and cross-correlation. Supports flexible detrending and climate–growth modeling via generalized additive mixed models (Wood 2017, ISBN:978-1498728331) and the 'mgcv' package (<https://CRAN.R-project.org/package=mgcv>), enabling robust analysis of non-linear trends and autocorrelated data. Provides standardized visual reporting, including summaries, diagnostics, and model performance. Compatible with '.rwl' files and tailored for the Canadian Forest Service Tree-Ring Data (CFS-TRenD) repository (Girardin et al. (2021) <doi:10.1139/er-2020-0099>), offering a comprehensive and adaptable framework for dendrochronologists working with large and complex datasets.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Imports data.table (>= 1.17.8), stringr (>= 1.4.0), stats (>= 4.3.2), mgcv (>= 1.8.0), ggplot2 (>= 3.0.0), nlme(>= 3.0.0), patchwork(>= 1.0.0), terra (>= 1.7), sf(>= 1.0-15), raster(>= 3.6-26), furrr(>= 0.3.1), future(>= 1.34.0), htmltools(>= 0.5.9), curl(>= 7.0.0), dplyr(>= 1.1.4)

Suggests knitr, geosphere, ggeffects (>= 2.3.0), ggforce(>= 0.4.2), gstat(>= 2.1-3), scales(>= 1.3.0), purrr(>= 1.0.0), parallel(>= 4.3.0), MuMIn(>= 1.48.2), rmarkdown(>= 2.29), rstudioapi(>= 0.16), MASS(>= 7.3-60), sp(>= 2.1.0), spdep(>= 1.3), magick(>= 2.8.6), gt(>= 1.2.0), kableExtra(>= 1.4.0), httr(>= 1.4.7), gridExtra(>= 2.3), rnaturalearth(>= 1.2.0)

Depends R (>= 4.3.0)

VignetteBuilder knitr

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libicu-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

Repository https://cfs-treering.r-universe.dev

Date/Publication 2026-05-31 12:05:39 UTC

RemoteUrl https://github.com/cfs-treering/growthtrendr

RemoteRef HEAD

RemoteSha eb81e4c090f272c420962b2087527864236750bd

Contents

`bam_spatial` *spatial growth model for large dataset or vast geographical coverage*

Description

To address the computational limitations of GAMMs for large datasets, this function offers a hybrid solution combining the efficiency of the mgcv::bam() function

Usage

```
bam_spatial(data, resp_scale = "resp_gamma", m.candidates, sp.option = "AICc")
```

Arguments

`data`	data containing all necessary columns to run the model
`resp_scale`	Character. Specifies how the response variable is treated in the model. • **"resp_gaussian"**: Uses the response on its original scale, assuming a Gaussian distribution with an identity link (no transformation applied). • **"resp_log"**: Log-transforms the response before modelling. The transformed response is then assumed to follow a Gaussian distribution with an identity link. • **"resp_gamma"**: Keeps the response on its original scale, fitted under a Gamma distribution with a log link. Suitable for strictly positive and right-skewed data.
`m.candidates`	the list of candidate equations.
`sp.option`	Character. Spatial autocorrelation selection method. • **"AICc"**: selects spatial structure using AICc comparison. • **"Moran"**: selects spatial structure based on Moran's I diagnostics.

Details

This function accounts for within-site variability and temporal autocorrelation by including series identity as random effects and a first-order autoregressive (AR1) correlation structures, respectively. Among-site variability and spatial effects are captured by incorporating site identity as random effects. The model is refitted automatically by introducing a smooth term for latitude and longitude using the thin plate ("tp") basis if significant spatial autocorrelation persists. "Normalized" residuals are provided for future analysis.

This function supports parallel computation for the large-scale, geographically distributed datasets.

If users specify multiple candidate models through the m.candidates argument, the function will fit each candidate model using the maximum likelihood (ML) method. The corrected Akaike Information Criterion (AICc) will then be compared to determine the best-fitting model. Once the optimal model is identified, it will be refitted using the restricted maximum likelihood (REML) method and output the results.

If users specify only 1 candidate model through the m.candidates argument, the model is fitted with "REML" method.

Value

list including model, fitting statistics, ptable, stable and prediction table

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# climate
```

```
dt.clim <- data.table::fread(system.file("extdata", "dt.clim.csv", package = "growthTrendR"))
# pre-data for model
dt.samples_clim <- prepare_samples_clim(dt.samples_trt, dt.clim)
dt.m <- dt.samples_clim[ageC >1]
# bam_spatial model
m.sp <-bam_spatial(data = dt.m, resp_scale = "resp_log",
       m.candidates = c(
       "bai_cm2 ~ log(ba_cm2_t_1) + s(ageC) + s(FFD)",
       "bai_cm2 ~ log(ba_cm2_t_1) + s(ageC) + FFD",
       "bai_cm2 ~ log(ba_cm2_t_1) + s(ageC)"))
```

`calc_bai` *calculate annual basal area increment*

Description

calculate basal area (cm2) and basal area increment (cm2)

Usage

```
calc_bai(data)
```

Arguments

`data`	data in long format containing at least 3 columns: id, year, rw

Value

add 3 columns to the original input data, ageC for cambial age, ba_cm2_t_1 for basal area of the previous year in cm2, and bai_cm2 for annual basal area increment in cm2

Examples

```
 # generate data
dt.rw <- data.table::data.table(
  uid_radius = rep(paste0("R", 1:3), each = 5),
  year      = rep(2001:2005, times = 3),
  rw_mm     = round(runif(15, 0.5, 3.5),2)
)
data.table::setorder(dt.rw, uid_radius, year)
 # calculate bai
dt.rw <- calc_bai(dt.rw)
```

CFS_format *Convert tree-ring data into CFS-TRenD format*

Description

converts tree-ring data from various formats into a format compatible with hierarchical structure of the CFS-TRenD data collection (Girardin et al., 2021).

Usage

```
CFS_format(data, usage, out.csv = NULL)
```

Arguments

data	a list, first is input data in wide format; second is a flat sequence referring to the column indices of ring measurement variables
usage	1: for submission data, 2: for cfstrend id structure to perform the analyse
out.csv	output csv file (default is NULL, otherwise to specify the directory to output)

Value

A list of 3 elements: 1) A list containing seven tables compatible with CFS-TRenD data structure; 2) A data table containing all the meta-data and ring width measurement in wide format; 3) A data table for the percentage of completeness of each variable.

References

Girardin, M.P., Guo, X.J., Metsaranta, J., Gervais, D., Campbell, E., Arsenault, A., Isaac-Rentone, M., Harvey, J.E., Bhatti, J., Hogg, E.A. 2021. A national tree-ring repository for Canadian forests (CFS-TRenD): structure, synthesis and applications. Environmental Reviews, 29 (999), 1-17. https://doi.org/10.1139/er-2020-0099

Examples

```
# ring measurement
dt.samples <- data.table::fread(
system.file("extdata", "dt.samples.csv", package = "growthTrendR"))
# formatting the users' data conformed to CFS-TRenD data structure
dt.samples_trt <- CFS_format(data = list(dt.samples, 39:68), usage = 1, out.csv = NULL)
# save it to extdata for further use
# saveRDS(dt.samples_trt, file = "inst/extdata/dt.samples_trt.rds")
```

CFS_freq *frequency distributions by geo-location per species*

Description

frequency distributions by geo-location per species

Usage

```
CFS_freq(
  data,
  freq.label_data = "",
  freq.uid_level = "uid_radius",
  freq.cutoff_year = -999,
  freq.geo_resolution = NULL
)
```

Arguments

data	meta table from function CFS_format()
freq.label_data	description of data
freq.uid_level	which uid level to count(uid_project, uid_site, uid_tree, uid_meas, uid_sample, uid_radius)
freq.cutoff_year	cut-off year for a subset which series were recorded on or after
freq.geo_resolution	Numeric vector (lon, lat) giving spatial resolution in degrees (e.g., c(5, 3)). If NULL, each value defaults to one quarter of the corresponding coordinate range.

Value

a data table of counts of uid by latitude-longitude per species

Examples

```
# treated ring measurement
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# Compute frequency statistics at radius level
dt.freq <- CFS_freq(
  dt.samples_trt$tr_all_wide,
  freq.label_data = "demo-samples",
  freq.uid_level  = "uid_radius"
)
class(dt.freq)
```

CFS_mapping *Interpolate and Map Tree-Ring Data*

Description

This function performs inverse distance weighting (IDW) interpolation of tree-ring data across a spatial grid, either for all species combined or by individual species. It generates yearly interpolated raster maps over a user-defined extent or the extent of the input data.

Usage

```
CFS_mapping(
  data,
  year.span = c(1801, 2017),
  extent.lim = NULL,
  grid.step = 0.1,
  by.spc = FALSE
)
```

Arguments

data	input in wide format.
year.span	Numeric vector of length 2 giving the range of years to include.
extent.lim	Optional numeric vector defining the spatial extent (c(xmin, xmax, ymin, ymax)). If NULL, the extent is determined from the input data.
grid.step	Numeric value specifying the grid spacing in degrees.
by.spc	Logical; if TRUE, maps are generated by species; if FALSE, all species are combined.

Value

An object of class cfs_map, a list of interpolated raster layers by species and year.

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
cols.meta = c("uid_tree", "uid_site", "longitude", "latitude", "species")
dt.mapping <- dt.samples_trt$tr_all_wide[
  , c(..cols.meta, as.character(1991:1995)), with = FALSE]
results_mapping <- CFS_mapping(dt.mapping, year.span = c(1991,1993))
```

CFS_qa *Quality assessment: radius alignment*

Description

Performs comprehensive quality assurance analysis for tree-ring crossdating using pairwise cross-correlation functions and iterative chronologies refinement with automatic memory-efficient batch processing.

Usage

```
CFS_qa(
  dt.input,
  qa.label_data = "",
  qa.label_trt = "",
  qa.max_lag = 10,
  qa.max_iter = 100,
  qa.min_nseries = 100,
  qa.blcrit = 0.1,
  qa.mem_target = 0.6
)
```

Arguments

dt.input	A data.table containing tree-ring measurements with required columns: **species** Character. Species identifier (must be single species) **SampleID** Character/Integer. Unique sample identifier **Year** Integer. Year of measurement **RawRing** Numeric. Raw ring-width measurement **RW_trt** Numeric. Treated/detrended ring-width series
qa.label_data	Character. Label identifier for the dataset (required)
qa.label_trt	Character. Label identifier for the treatment method (required)
qa.max_lag	Integer. Maximum lag for cross-correlation analysis (default: 10)
qa.max_iter	Integer. Maximum iterations for chronologies refinement (default: 100)
qa.min_nseries	Integer. Minimum number of series required (default: 100)
qa.blcrit	Numeric. Borderline threshold criterion for quasi-pass classification (default: 0.1)
qa.mem_target	Numeric. Proportion of free memory to use for batch processing (0-1, default: 0.6). Higher values use more memory but may be faster.

Details

The function performs a two-step quality assurance process:

Step 1: Pairwise Cross-Correlation

- Computes CCF for all pairs of treated series
- Uses auto-batching to manage memory efficiently
- Identifies initial qualified samples with max CCF at lag 0

Step 2: Iterative chronologies Refinement

- Computes chronologies from qualified samples
- Evaluates each sample by running CCF with the chronologies
- Iteratively refines the qualified samples until convergence

QA Code Classification:

- `pass`: Maximum correlation at lag 0
- `borderline`: Second highest correlation at lag 0 (within threshold)
- `pm1`: Maximum correlation at lag +/- 1 (slight misalignment)
- `highpeak`: Maximum at non-zero lag, >2x second highest
- `fail`: All other cases

Auto-batching: The function automatically determines optimal batch size based on:

- Available system memory
- Number of CPU cores
- Estimated memory per CCF operation
- Safety margins to prevent out-of-memory errors

Value

An object of class `cfs_qa` containing:

dt.ccf data.table with CCF results and QA codes (`qa_code`) for each series

dt.chron data.table with chronologies statistics

dt.stats data.table with summary statistics per radius

dt.plots List of data.tables formatted for plotting (raw and treated series, CCF plots)

qa.parms List of parameters used in the analysis

See Also

`ccf` for cross-correlation function

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# data processing
dt.samples_long <- prepare_samples_clim(dt.samples_trt, calbai = FALSE)
# rename to the reserved column name
data.table::setnames(dt.samples_long,
c("sample_id", "year", "rw_mm"), c("SampleID", "Year" ,"RawRing"))
# assign treated series
# users can decide their own treated series
# for rhub::rhub_check() on macos VECTOR_ELT issues
data.table::setorder(dt.samples_long, SampleID, Year)
dt.samples_long$RW_trt <-
  ave(
  as.numeric(dt.samples_long$RawRing),
  dt.samples_long$SampleID,
  FUN = function(x)
  if (length(x) > 1L) c(NA_real_, diff(x)) else NA_real_
  )
# quality check on radius alignment based on the treated series
dt.qa <-CFS_qa(dt.input = dt.samples_long, qa.label_data = "demo-samples",
qa.label_trt = "difference", qa.min_nseries = 5)
```

`CFS_scale` *Quality assessment: ring-width measurement at plot level*

Description

Apply a k-nearest neighbors (k-NN) method based on geographic location for the same species. It compares the median tree-ring measurements of a specific site to those of nearby sites

Usage

```
CFS_scale(
  target_site,
  ref_sites,
  scale.label_data_ref = "",
  scale.max_dist_km = 20,
  scale.N_nbs = 10
)
```

Arguments

`target_site`	data.table with columns uid_site, site_id, species, longitude and latitude
`ref_sites`	reference sites including species, uid_site, latitude, longitude, uid_radius, year, rw_mm in long format

scale.label_data_ref
: description of ref_sites

scale.max_dist_km
: maximum distance to search the neighbors in km

scale.N_nbs
: number of nearest-neighbors (maximum)

Value

A data table containing the median ring-width measurements of the involved sites, along with the distances from the specific site

Examples

```
# loading formatted
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
all.sites <- dt.samples_trt$tr_all_wide[,.N, by = c("species", "uid_site", "site_id")][, N:=NULL]
dupes <- all.sites[, .N, by = .(species, site_id)][N > 1]
# e.g. taking the target sites
target_site <- all.sites[c(1,2), -"uid_site"]
ref.sites <- merge(
dt.samples_trt$tr_all_wide[,c("species", "uid_site", "site_id",
 "latitude","longitude", "uid_radius")],
dt.samples_trt$tr_all_long$tr_7_ring_widths, by = c("uid_radius"))
dt.scale <- CFS_scale( target_site = target_site, ref_sites = ref.sites,
scale.label_data_ref = "demo-samples", scale.max_dist_km = 200, scale.N_nbs = 2)
```

---

ci_resp *Compute prediction and confidence intervals of smooth terms on response scale*

Description

This function computes predicted values and confidence intervals from a fitted GAM model with a log-link (or other link) and optionally back-transforms predictions to the response scale. Five methods are supported: 1. **delta_link**: classic delta method on the linear predictor (link) scale; back-transformed CI is asymmetric. 2. **delta_resp**: delta method applied directly on the response scale using $\mathrm{Var}[\exp(\eta)] \approx \exp(2\eta)\,\mathrm{Var}(\eta)$. 3. **bootstrap_link**: parametric bootstrap on the linear predictor; quantiles back-transformed. 4. **bootstrap_resp**: parametric bootstrap on the response scale; quantiles computed after exponentiating. 5. **posterior**: Bayesian posterior simulation using the model covariance matrix; quantiles on response scale.

Usage

```
ci_resp(
  model,
  newdata,
  nboot = 100,
  model.ci_method = c("posterior", "delta_link", "delta_resp", "bootstrap_link",
    "bootstrap_resp"),
  level = 0.95
)
```

Arguments

model	A fitted GAM object from mgcv::gam.
newdata	A data.frame containing values at which to predict.
nboot	Integer. Number of bootstrap or posterior samples. Default 1000.
model.ci_method	Character. One of "delta_link", "delta_resp", "bootstrap_link", "bootstrap_resp", "posterior".
level	Numeric. Confidence level, default 0.95.

Details

References: - Wood, S.N. (2017) *Generalized Additive Models: An Introduction with R, 2nd Edition*. CRC Press. - Efron, B., & Tibshirani, R. (1993) *An Introduction to the Bootstrap*. Chapman & Hall. - Gelman, A., et al. (2013) *Bayesian Data Analysis, 3rd Edition*. CRC Press.

Value

A data.table with columns: fit, lwr, upr.

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# climate
dt.clim <- data.table::fread(system.file("extdata", "dt.clim.csv", package = "growthTrendR"))
# pre-data for model
dt.samples_clim <- prepare_samples_clim(dt.samples_trt, dt.clim)
dt.m <- dt.samples_clim[ageC >1]
# using gamm_spatial model as an example
m.sp <-gamm_spatial(data = dt.m, resp_scale = "resp_log",
       m.candidates = "bai_cm2 ~ log(ba_cm2_t_1) + s(ageC) + s(FFD)")

dt.m[, uid_site.fac:= as.factor(as.character(uid_site))]
dt.ci <- ci_resp(m.sp$model$gam, newdata = dt.m)
```

gam_mod *generalized additive model*

Description

generalized additive model that captures non-linear relationships between a response variable and predictors using smooth functions, while allowing inclusion of random effects or complex structures.

Usage

```
gam_mod(data, resp_scale = "", m.candidates, sp.option = "AICc")
```

Arguments

data	data containing all necessary columns to run the model
resp_scale	Character. Specifies how the response variable is treated in the model. • **"resp_gaussian"**: Uses the response on its original scale, assuming a Gaussian distribution with an identity link (no transformation applied). • **"resp_log"**: Log-transforms the response before modelling. The transformed response is then assumed to follow a Gaussian distribution with an identity link. • **"resp_gamma"**: Keeps the response on its original scale, fitted under a Gamma distribution with a log link. Suitable for strictly positive and right-skewed data.
m.candidates	the list of candidate equations.
sp.option	Character. Spatial autocorrelation selection method. • **"AICc"**: selects spatial structure using AICc comparison. • **"Moran"**: selects spatial structure based on Moran's I diagnostics.

Details

This function models the generalized additive model using mcgv::gam.

If users specify multiple candidate models through the m.candidates argument, the function will fit each candidate model using the maximum likelihood (ML) method. The corrected Akaike Information Criterion (AICc) will then be compared to determine the best-fitting model. Once the optimal model is identified, it will be refitted using the restricted maximum likelihood (REML) method and output the results.

If users specify only 1 candidate model through the m.candidates argument, the model is fitted with "REML" method.

Value

list including model, fitting statistics, ptable, stable and prediction table

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# pre-data for model
dt.samples_long <- prepare_samples_clim(dt.samples_trt)
dt.samples_long$uid_site.fac <- as.factor(as.character(dt.samples_long$uid_site))
dt.m <- dt.samples_long

# gam_mod
m.gam <-gam_mod(data = dt.m, resp_scale = "resp_log",
                                  m.candidates = c( "rw_mm ~ s(year, by = uid_site.fac)",
                                                    "rw_mm ~ year:uid_site.fac"))
```

`gamm_radius` *detrending model on tree-ring width series*

Description

models the biological growth trends in individual tree-ring width series using mgcv::gamm

Usage

```
gamm_radius(data, resp_scale = "resp_gamma", m.candidates)
```

Arguments

`data`	data containing all necessary columns to run the model
`resp_scale`	Character. Specifies how the response variable is treated in the model. • **"resp_gaussian"**: Uses the response on its original scale, assuming a Gaussian distribution with an identity link (no transformation applied). • **"resp_log"**: Log-transforms the response before modelling. The transformed response is then assumed to follow a Gaussian distribution with an identity link. • **"resp_gamma"**: Keeps the response on its original scale, fitted under a Gamma distribution with a log link. Suitable for strictly positive and right-skewed data.
`m.candidates`	the list of candidate equations.

Details

This function models the biological growth trends in individual tree-ring width series using mcgv::gamm. By integrating a first-order autoregressive (AR1) component, it accounts for temporal autocorrelation. This method can provide “normalized” residuals, which are adjusted to reflect deviations after considering the AR1 correlation structure. ’Normalized’ residuals are valuable for further analyses, such as investigating relationships with climatic variables. If users specify multiple candidate models through the m.candidates argument, the function will fit each candidate model using the

maximum likelihood (ML) method. The corrected Akaike Information Criterion (AICc) will then be compared to determine the best-fitting model. Once the optimal model is identified, it will be refitted using the restricted maximum likelihood (REML) method and output the results.

If users specify only 1 candidate model through the m.candidates argument, the model is fitted with "REML" method.

Value

list including model, fitting statistics, ptable, stable and prediction table

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# pre-data for model
dt.samples_long <- prepare_samples_clim(dt.samples_trt)

dt.m <- dt.samples_long[uid_site == 1][ageC >1]

# gamm_radius model
m.radius <-gamm_radius(data = dt.m, resp_scale = "resp_log",
                                        m.candidates = c( "rw_mm ~ s(year)",
                                                          "rw_mm ~ s(year)"))
```

---

gamm_site *growth model at site-level*

Description

models the growth trend or climate-growth relationship per site.

Usage

```
gamm_site(data, resp_scale = "resp_gamma", m.candidates)
```

Arguments

`data`	data containing all necessary columns to run the model
`resp_scale`	Character. Specifies how the response variable is treated in the model. • **"resp_gaussian"**: Uses the response on its original scale, assuming a Gaussian distribution with an identity link (no transformation applied). • **"resp_log"**: Log-transforms the response before modelling. The transformed response is then assumed to follow a Gaussian distribution with an identity link. • **"resp_gamma"**: Keeps the response on its original scale, fitted under a Gamma distribution with a log link. Suitable for strictly positive and right-skewed data.
`m.candidates`	the list of candidate equations.

Details

This function accounts for within-site variability and temporal autocorrelation by including series identity as random effects and a first-order autoregressive (AR1) correlation structures, respectively. using mgcv::gamm()

If users specify multiple candidate models through the m.candidates argument, the function will fit each candidate model using the maximum likelihood (ML) method. The corrected Akaike Information Criterion (AICc) will then be compared to determine the best-fitting model. Once the optimal model is identified, it will be refitted using the restricted maximum likelihood (REML) method and output the results.

If users specify only 1 candidate model through the m.candidates argument, the model is fitted with "REML" method.

Value

list including model, fitting statistics, ptable, stable and prediction table

Examples

```
#' @examples
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# climate
dt.clim <- data.table::fread(system.file("extdata", "dt.clim.csv", package = "growthTrendR"))
# pre-data for model
dt.samples_clim <- prepare_samples_clim(dt.samples_trt, dt.clim)

dt.m <- dt.samples_clim[uid_site == 1][ageC >1]

# gamm_site model
m.site <-gamm_site(data = dt.m, resp_scale = "resp_log",
         m.candidates = c(
         "bai_cm2 ~ log(ba_cm2_t_1) + s(ageC) + s(FFD)",
         "bai_cm2 ~ log(ba_cm2_t_1) + s(ageC) + FFD",
         "bai_cm2 ~ log(ba_cm2_t_1) + s(ageC)"))
```

---

gamm_spatial *spatial growth model at regional-level (multiple sites)*

Description

models the growth trend or climate-growth relationship at regional-level with multiple sites

Usage

```
gamm_spatial(data, resp_scale = "resp_gamma", m.candidates, sp.option = "AICc")
```

Arguments

`data`	data containing all necessary columns to run the model
`resp_scale`	Character. Specifies how the response variable is treated in the model. • **"resp_gaussian"**: Uses the response on its original scale, assuming a Gaussian distribution with an identity link (no transformation applied). • **"resp_log"**: Log-transforms the response before modelling. The transformed response is then assumed to follow a Gaussian distribution with an identity link. • **"resp_gamma"**: Keeps the response on its original scale, fitted under a Gamma distribution with a log link. Suitable for strictly positive and right-skewed data.
`m.candidates`	the list of candidate equations.
`sp.option`	Character. Spatial autocorrelation selection method. • **"AICc"**: selects spatial structure using AICc comparison. • **"Moran"**: selects spatial structure based on Moran's I diagnostics.

Details

This function accounts for within-site variability and temporal autocorrelation by including series identity as random effects and a first-order autoregressive (AR1) correlation structures, respectively. Among-site variability and spatial effects are captured by incorporating site identity as random effects. The model is refitted automatically by introducing a smooth term for latitude and longitude using the thin plate ("tp") basis if significant spatial autocorrelation persists. "Normalized" residuals are provided for future analysis.

If users specify multiple candidate models through the m.candidates argument, the function will fit each candidate model using the maximum likelihood (ML) method. The corrected Akaike Information Criterion (AICc) will then be compared to determine the best-fitting model. Once the optimal model is identified, it will be refitted using the restricted maximum likelihood (REML) method and output the results.

If users specify only 1 candidate model through the m.candidates argument, the model is fitted with "REML" method.

Value

list including model, fitting statistics, ptable, stable, prediction table and spatial effect(moranI)

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# climate
dt.clim <- data.table::fread(system.file("extdata", "dt.clim.csv", package = "growthTrendR"))
# pre-data for model
dt.samples_clim <- prepare_samples_clim(dt.samples_trt, dt.clim)
dt.m <- dt.samples_clim[ageC >1]
# gamm_spatial model
m.sp <-gamm_spatial(data = dt.m, resp_scale = "resp_gamma",
```

```
m.candidates = c(
"bai_cm2 ~ log(ba_cm2_t_1) + s(ageC) + s(FFD)",
"bai_cm2 ~ log(ba_cm2_t_1) + s(ageC) + FFD",
"bai_cm2 ~ log(ba_cm2_t_1) + s(ageC)"))
```

`generate_report` *Generate Automated Reports*

Description

Creates HTML reports from various analysis objects using predefined R Markdown templates. The function automatically selects the appropriate template based on the input object's class and renders a comprehensive report with visualizations and analysis results.

Usage

```
generate_report(
  robj,
  data_report.reports_sel = c(1, 2, 3, 4),
  output_file = NULL,
  ...
)
```

Arguments

`robj`	An R object containing analysis results from functions in this package. The object's class determines which report template is used. Supported classes depend on available templates in the package.
`data_report.reports_sel`	Numeric vector. Specifies which sections of the data report to include in the output: 1 = project level; 2 = species level; 3 = site level; 4 = radius level. The default is 'c(1, 2, 3, 4)', which includes all sections.
`output_file`	Character string. Optional path and filename for the output HTML file. If NULL (default), the report is generated with an automatic filename and opened in RStudio viewer.
`...`	Additional parameters passed to the R Markdown template. Available parameters vary by template type and are filtered to only include those recognized by the selected template.

Value

Invisibly returns the file path of the generated report. The function is primarily called for its side effect of generating the report file.

Examples

```
# loading processed data

dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# genereate data summary report at project level
outfile_data <- tempfile(fileext = ".html")
generate_report(robj = dt.samples_trt, data_report.reports_sel = c(1), output_file = outfile_data)
```

---

plot_freq *plot frequency distribution by geo-location per species*

Description

This function plots the frequency distribution by geo-location for each species

Usage

```
plot_freq(dt.freq, out.species = "all")
```

Arguments

dt.freq	a table with class "cfs_freq", resulting from function CFS_freq()
out.species	species list, default is 'all' to output the frequency distribution for all species

Value

A list of ggplot objects corresponding to the species requested via out.species. Each element of the list contains a faceted spatial plot of tree locations, with point size proportional to the number of samples.

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
dt.freq <- CFS_freq(
dt.samples_trt$tr_all_wide,
freq.label_data = "demo-samples",
freq.uid_level = "uid_radius")

plots.lst <- plot_freq(dt.freq)
```

plot_mapping *Plot spatially interpolated tree-ring growth maps*

Description

Visualizes spatial interpolation results produced by CFS_mapping and optionally exports raster files, static maps, and animations.

Usage

```
plot_mapping(
  mapping_results,
  crs.src = "EPSG:4326",
  parms.out = c("csv", "tif", "png", "gif"),
  dir.shp = NULL,
  dir.out = NULL,
  animation_fps = 1,
  ...
)
```

Arguments

mapping_results	A list returned by CFS_mapping containing spatially interpolated rasters by species and year.
crs.src	Coordinate Reference System of the input data, specified as a string in the format 'EPSG:<ID>' (for example, 'EPSG:4326'). The function will stop with an error if 'crs.src' is not provided in this format.
parms.out	Character vector indicating output formats to generate. Supported values are "csv", "tif", "png", and "gif".
dir.shp	Character or NULL. Path to the folder containing shapefiles for cropping data to the Canadian boreal regions. Only used for specific research purposes; if NULL (default), no cropping is applied and all data are included.
dir.out	Output directory used to save generated files. Required when parms.out is not empty.
animation_fps	Frames per second used when creating GIF animations.
...	Additional arguments passed to plot_tree_ring_map, such as png.text.

Details

This function assumes that spatial interpolation has already been performed using CFS_mapping. The input object is iterated by species and year to generate maps and optional exports.

When "gif" is requested in parms.out, yearly PNG images are combined into animated GIFs.

Value

A magick-image object representing an animated GIF composed of the generated frames.

See Also

CFS_mapping

Examples

```
# Load processed demo data
dt.samples_trt <- readRDS(
  system.file("extdata", "dt.samples_trt.rds",
              package = "growthTrendR")
)
# prepare data for IDW model
cols.meta = c("uid_tree", "uid_site", "longitude", "latitude", "species")
dt.mapping <- dt.samples_trt$tr_all_wide[
  , c(..cols.meta, as.character(1991:1995)), with = FALSE]

# Run spatial interpolation
mapping_results <- CFS_mapping(
  dt.mapping,
  year.span = c(1991, 1993)
)

# generate png plots
img_ani <- plot_mapping(
mapping_results = mapping_results,
parms.out = NULL,
dir.shp = NULL,
dir.out = NULL,
png.text = list(
  text_top  = "Ring width measurement - ",
  text_bott = "Source: demo-samples",
  text_side = "ring width (mm)"
)
)
```

---

plot_qa *plotting quality assessment per radius*

Description

Plotting the time series and cross-correlation to contrast each radius with the chronologies, using both raw ring-width measurements and treated series.

Usage

```
plot_qa(qa.trt, qa.out_series = "all")
```

Arguments

qa.trt	object "cfs_qa" from CFS_qa() function.
qa.out_series	series_id list to be plotted, default "all" for plotting the graphs for all.

Value

A named list of tree-ring series, where each element corresponds to a series requested via `qa.out_series`. Each element is itself a list containing four `ggplot` objects:

plot.raw.series Raw tree-ring series vs year.

plot.trt.series Treated (detrended/standardized) series vs year.

plot.raw.ccf Cross-correlation function of the raw series.

plot.trt.ccf Cross-correlation function of the treated series.

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# data processing
dt.samples_long <- prepare_samples_clim(
dt.samples_trt = dt.samples_trt, calbai = FALSE )
# rename to the reserved column name
data.table::setnames(
dt.samples_long,
c("sample_id", "year", "rw_mm"),
c("SampleID", "Year" ,"RawRing"))
# assign treated series
# users can decide their own treated series

# for rhub::rhub_check() on macos VECTOR_ELT issues
data.table::setorder(dt.samples_long, SampleID, Year)
dt.samples_long$RW_trt <-
  ave(
  as.numeric(dt.samples_long$RawRing),
  dt.samples_long$SampleID,
  FUN = function(x)
  if (length(x) > 1L) c(NA_real_, diff(x)) else NA_real_
  )
# quality check on radius alignment based on the treated series
dt.qa <-CFS_qa(dt.input = dt.samples_long, qa.label_data = "demo-samples",
qa.label_trt = "difference", qa.min_nseries = 5)
plots.lst <- plot_qa(dt.qa, qa.out_series = "X003_101_005")
```

plot_resp *Plot smooth terms prediction with confidence intervals*

Description

This function generates ggplot objects for each smooth term in a GAM model. Predictions vary one smooth term at a time, keeping all other terms fixed at reference values. Confidence intervals are computed using `ci_resp()`, supporting multiple methods: "delta_link", "delta_resp", "bootstrap_link", "bootstrap_resp", and "posterior". Small samples automatically trigger the "posterior" method for more robust CIs.

Usage

```
plot_resp(robj, model.ci_method = "posterior", ...)
```

Arguments

`robj`	R object from the modelling functions.
`model.ci_method`	Character. One of `"delta_link"`, `"delta_resp"`, `"bootstrap_link"`, `"bootstrap_resp"`, `"posterior"`.
`...`	Additional arguments passed to `ci_resp()`.

Value

A list of ggplot objects, one per smooth term.

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# climate
dt.clim <- data.table::fread(system.file("extdata", "dt.clim.csv", package = "growthTrendR"))
# pre-data for model
dt.samples_clim <- prepare_samples_clim(dt.samples_trt, dt.clim)
dt.m <- dt.samples_clim[ageC >1]
# gamm_site model
m.spatial <-gamm_spatial(
  data = dt.m, resp_scale = "resp_log",
  m.candidates = c( "bai_cm2 ~ log(ba_cm2_t_1) + s(ageC) + s(FFD)"))
plots.lst <- plot_resp(m.spatial, model.ci_resp = "posterior")
```

plot_scale *plot median of ring width of the target site and its neighbours*

Description

This function plots the time series and the geographical distribution of the median ring-width measurements to contrast the target site with its neighbouring sites.

Usage

```
plot_scale(dt.scale)
```

Arguments

dt.scale	a table with class "cfs_scale", resulting from function CFS_scale()

Value

A named list with two ggplot objects showing the contrast between the target site and its neighboring sites:

plot.year A time-series plot of median ring width by year.

plot.ll A spatial plot showing the geographic location of the sites, with point size proportional to the magnitude of site-level ring-width measurements.

Examples

```
# loading formatted
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
all.sites <- dt.samples_trt$tr_all_wide[,.N, by = c("species", "uid_site", "site_id")][, N:=NULL]
# e.g. taking the target sites
target_site <- all.sites[c(1,2), -"uid_site"]
ref.sites <- merge(
dt.samples_trt$tr_all_wide[,c("species", "uid_site", "site_id",
 "latitude","longitude", "uid_radius")],
dt.samples_trt$tr_all_long$tr_7_ring_widths, by = c("uid_radius"))
dt.scale <- CFS_scale( target_site = target_site, ref_sites = ref.sites,
scale.label_data_ref = "demo-samples", scale.max_dist_km = 200, scale.N_nbs = 2)
plots.lst <- plot_scale(dt.scale[[1]])
```

prepare_samples_clim *Prepare tree-ring data for downstream analysis*

Description

This function prepares tree-ring data including ring-width measurements, and optionally computes basal area increment (BAI) and merges climate variables.

Usage

```
prepare_samples_clim(dt.samples_trt, dt.clim = NULL, calbai = TRUE)
```

Arguments

dt.samples_trt	A list of tree-ring data formatted by CFS_format().
dt.clim	An optional data frame containing climate variables, joined by site_id and year. Default is NULL.
calbai	Logical. If TRUE, basal area increment (BAI) is computed. Default is TRUE.

Value

A data frame or data.table containing tree-ring measurements, optionally including basal area increment and merged climate variables.

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# climate
dt.clim <- data.table::fread(system.file("extdata", "dt.clim.csv", package = "growthTrendR"))
# pre-data for model
dt.samples_clim <- prepare_samples_clim(dt.samples_trt, dt.clim)
```

read_rwl *Import raw ring-width measurements from the ITRDB*

Description

Import raw ring-width measurements from the ITRDB

Usage

```
read_rwl(dir.src, rwl)
```

Arguments

`dir.src`	Character string specifying the directory path or URL where the RWL file is located. Users typically provide a local directory containing a user-downloaded ITRDB file, but an online source may also be used.
`rwl`	Character string giving the file name of the RWL file.

Details

The International Tree-Ring Data Bank (ITRDB) is maintained by the NOAA National Centers for Environmental Information (NCEI) as part of the World Data Service for Paleoclimatology. See: https://www.ncei.noaa.gov/products/paleoclimatology/tree-ring

Raw ring-width measurement files ('.rwl') distributed through the ITRDB follow the *Tucson fixed-width format*. In this convention, header records encode site-level metadata (e.g., site identifier, site name, species code, geographic information, and temporal coverage), followed by ring-width measurements stored in a decadal layout (ten annual values per line).

Record 1

- Columns 1–6: Site ID
- Columns 10–61: Site name
- Columns 62–65: Species code
- Optional ID fields

Record 2

- Columns 1–6: Site ID
- Columns 10–22: State / country
- Columns 23–30: Species
- Columns 41–45: Elevation
- Columns 48–57: Latitude–longitude
- Columns 62–63: Measurement type code
- Columns 68–76: First and last year

Latitude–longitude values are expressed in degrees and minutes ('ddmm' or 'dddmm').

Record 3

- Columns 1–6: Site ID
- Columns 10–72: Investigators
- Columns 73–80: Optional completion date

Value

A list with two elements:

- `header`: a table containing site-level metadata
- `rwl`: a table containing raw ring-width measurements

Examples

```
## Online example (not run to avoid timeout and internet dependency)

dir.src <- "https://www.ncei.noaa.gov/pub/data/paleo/treering/measurements/northamerica/canada"
rwl <- "cana615.rwl"
dt.rwl <- read_rwl(dir.src, rwl)


## Local example using packaged data
file <- system.file("extdata", "cana615.rwl", package = "growthTrendR")
stopifnot(file != "")
dir.src <- dirname(file)
rwl <- basename(file)
dt.rwl <- read_rwl(dir.src, rwl)
```

---

sterm_imp *variable importance of smooth terms in a GAM model*

Description

Evaluates the relative influence of each smooth term in a GAM model by computing its contribution to the fitted values using the linear predictor matrix (`type = "lpmatrix"`). Three summary methods are available: sum of squares, variance, and mean absolute value across all observations. #'

Usage

```
sterm_imp(gam_model, method = c("ssq", "var", "meanabs"))
```

Arguments

`gam_model`	A GAM model object.
`method`	A character string specifying the method to compute importance. One of `"ssq"`, `"var"`, or `"meanabs"`.

Value

A `data.table` with columns:

var Name of the smooth term.

importance_pct Relative importance as a percentage.

method The method used for calculating the importance.

Examples

```
# loading processed data
dt.samples_trt <- readRDS(system.file("extdata", "dt.samples_trt.rds", package = "growthTrendR"))
# climate
dt.clim <- data.table::fread(system.file("extdata", "dt.clim.csv", package = "growthTrendR"))
# pre-data for model
dt.samples_clim <- prepare_samples_clim(dt.samples_trt, dt.clim)
dt.m <- dt.samples_clim[ageC >1]
# using gamm_spatial model as an example
m.sp <-gamm_spatial(data = dt.m, resp_scale = "resp_log",
       m.candidates = "bai_cm2 ~ log(ba_cm2_t_1) + s(ageC) + s(FFD)")

dt.m[, uid_site.fac:= as.factor(as.character(uid_site))]
dt.imp <- sterm_imp(m.sp$model$gam)
```

Index