Package 'climwin'

May 8, 2026

Type Package

Title Climate Window Analysis

Version 1.2.33

Maintainer Liam D. Bailey <liam.bailey@liamdbailey.com>

URL https://github.com/LiamDBailey/climwin

BugReports https://github.com/LiamDBailey/climwin/issues

Description Contains functions to detect and visualise periods of climate sensitivity (climate windows) for a given biological response. Please see van de Pol et al. (2016) <doi:10.1111/2041-210X.12590> and Bailey and van de Pol (2016) <doi:10.1371/journal.pone.0167980> for details.

Depends R (>= 3.5.0)

Imports ggplot2 (>= 3.5.2), gridExtra, Matrix, evd, lubridate, lme4, MuMIn, reshape, numDeriv, RcppRoll, nlme

Suggests testthat, knitr, rmarkdown

License GPL-2

Repository CRAN

LazyData True

VignetteBuilder knitr

RoxygenNote 7.3.2

Encoding UTF-8

NeedsCompilation no

Author Liam D. Bailey [aut, cre], Martijn van de Pol [aut]

Date/Publication 2026-03-01 23:40:02 UTC

Contents

`autowin` *Test for auto-correlation in climate.*

Description

Tests the correlation between the climate in a specified climate window and other fitted climate windows.

Usage

```
autowin(
  reference,
  xvar,
  cdate,
  bdate,
  baseline,
```

```
  range,
  stat,
  func,
  type,
  refday,
  cmissing = FALSE,
  cinterval = "day",
  upper = NA,
  lower = NA,
  binary = FALSE,
  centre = list(NULL, "both"),
  cohort = NULL,
  spatial = NULL,
  cutoff.day = NULL,
  cutoff.month = NULL,
  furthest = NULL,
  closest = NULL,
  thresh = NULL
)
```

Arguments

reference	Reference climate data to be compared. Generated by functions singlewin or slidingwin.
xvar	The climate variable of interest. Please specify the parent environment and variable name (e.g. Climate$Temp).
cdate	The climate date variable (dd/mm/yyyy). Please specify the parent environment and variable name (e.g. Climate$Date).
bdate	The biological date variable (dd/mm/yyyy). Please specify the parent environment and variable name (e.g. Biol$Date).
baseline	The baseline model used to fit climate windows. These will be correlated with the reference climate window.
range	Two values signifying respectively the furthest and closest number of time intervals (set by cinterval) back from the cutoff date or biological record to include in the climate window search.
stat	The aggregate statistic used to analyse the climate data. Can currently use basic R statistics (e.g. mean, min), as well as slope. Additional aggregate statistics can be created using the format function(x) (...). See parameter FUN in apply for more detail.
func	The function used to fit the climate variable. Can be linear ("lin"), quadratic ("quad"), cubic ("cub"), inverse ("inv") or log ("log"). Not required when a variable is provided for parameter 'centre'.
type	"absolute" or "relative", whether you wish the climate window to be relative (e.g. the number of days before each biological record is measured) or absolute (e.g. number of days before a set point in time).
refday	If type is "absolute", the day and month respectively of the year from which the absolute window analysis will start.

`cmissing` cmissing Determines what should be done if there are missing climate data. Three approaches are possible: - FALSE; the function will not run if missing climate data is encountered. An object 'missing' will be returned containing the dates of missing climate. - "method1"; missing climate data will be replaced with the mean climate of the preceding and following 2 days. - "method2"; missing climate data will be replaced with the mean climate of all records on the same date.

`cinterval` The resolution at which climate window analysis will be conducted. May be days ("day"), weeks ("week"), or months ("month"). Note the units of parameter 'range' will differ with the choice of cinterval.

`upper` Cut-off value used to determine growing degree days or positive climate thresholds (depending on parameter thresh). Note that when values of lower and upper are both provided, autowin will instead calculate an optimal climate zone.

`lower` Cut-off value used to determine chill days or negative climate thresholds (determined by parameter thresh). Note that when values of lower and upper are both provided, autowin will instead calculate an optimal climate zone.

`binary` TRUE or FALSE. Determines whether to use values of upper and lower to calculate binary climate data (binary = TRUE), or to use for growing degree days (binary = FALSE).

`centre` A list item containing: 1. The variable used for mean centring (e.g. Year, Site, Individual). Please specify the parent environment and variable name (e.g. Biol$Year). 2. Whether the model should include both within-group means and variance ("both"), only within-group means ("mean"), or only within-group variance ("dev").

`cohort` A variable used to group biological records that occur in the same biological season but cover multiple years (e.g. southern hemisphere breeding season). By default, autowin will use year (extracted from parameter bdate) as the cohort variable. The cohort variable should be in the same dataset as the variable bdate.

`spatial` A list item containing: 1. A factor that defines which spatial group (i.e. population) each biological record is taken from. The length of this factor should correspond to the length of the biological dataset. 2. A factor that defines which spatial group (i.e. population) climate data corresponds to. The length of this factor should correspond to the length of the climate dataset.

`cutoff.day`, `cutoff.month`
Redundant parameters. Now replaced by refday.

`furthest`, `closest`
Redundant parameters. Now replaced by range.

`thresh` Redundant parameter. Now replaced by binary.

Value

Will return a data frame showing the correlation between the climate in each fitted window and the chosen reference window.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
#Simple test example
#Create data from a subset of our test dataset
#Just use two years
biol_data <- Mass[1:2, ]
clim_data <- MassClimate[grep(pattern = "1979|1986", x = MassClimate$Date), ]

single <- singlewin(xvar = list(Temp = clim_data$Temp),
                    cdate = clim_data$Date,
                    bdate = biol_data$Date,
                    baseline = lm(Mass ~ 1, data = biol_data),
                    range = c(1, 0),
                    type = "relative", stat = "mean",
                    func = c("lin"), cmissing = FALSE, cinterval = "day")

auto <- autowin(reference = single,
                xvar  = list(Temp = clim_data$Temp),
                cdate = clim_data$Date, bdate = biol_data$Date,
                baseline = lm(Mass ~ 1, data = biol_data), range = c(1, 0),
                stat = "mean", func = "lin",
                type = "relative",
                cmissing = FALSE, cinterval = "day")

## Not run:

# Full example
# Test for auto-correlation using 'Mass' and 'MassClimate' data frames

data(Mass)
data(MassClimate)

# Fit a single climate window using the datasets Mass and MassClimate.

single <- singlewin(xvar = list(Temp = MassClimate$Temp),
                    cdate = MassClimate$Date, bdate = Mass$Date,
                    baseline = lm(Mass ~ 1, data = Mass),
                    range = c(72, 15),
                    stat = "mean", func = "lin", type = "absolute",
                    refday = c(20, 5),
                    cmissing = FALSE, cinterval = "day")

# Test the autocorrelation between the climate in this single window and other climate windows.

auto <- autowin(reference = single,
           xvar  = list(Temp = MassClimate$Temp), cdate = MassClimate$Date, bdate = Mass$Date,
                baseline = lm(Mass ~ 1, data = Mass), range = c(365, 0),
                stat = "mean", func = "lin",
                type = "absolute", refday = c(20, 5),
                cmissing = FALSE, cinterval = "day")

# View the output
head(auto)
```

```
# Plot the output
plotcor(auto, type = "A")

## End(Not run)
```

`Chaff` *Annual laying date of breeding common chaffinch (Fringilla coelebs).*

Description

Average annual laying date of common chaffinch (Fringilla coelebs) measured over 47 years.

Format

A data frame with 47 rows and 3 variables.

Year Year of laying date measurement.

Date Average date of measurement.

Laydate Average annual laying date in days after January 1st.

`ChaffClim` *Daily climate data from 1965 to 2012.*

Description

Maximum daily temperature and average rainfall data from 1965 to 2012. Coincides with biological data from `Chaff`.

Format

A data frame with 17,520 rows and 3 variables.

Date Date when climate was recorded (dd/mm/yyyy).

Rain Average daily rainfall data in mm.

Temp Maximum daily temperature in degrees centigrade.

`crosswin` *Test the correlation between two climate variables.*

Description

Test the correlation between two climate variables.

Usage

```
crosswin(
  xvar,
  xvar2,
  cdate,
  bdate,
  range,
  stat,
  stat2,
  type,
  refday,
  cinterval = "day",
  cmissing = FALSE,
  spatial = NULL,
  cohort = NULL,
  cutoff.day = NULL,
  cutoff.month = NULL,
  furthest = NULL,
  closest = NULL
)
```

Arguments

`xvar`	The first climate variable of interest. Please specify the parent environment and variable name (e.g. Climate$Temp).
`xvar2`	The second climate variable of interest. Please specify the parent environment and variable name (e.g. Climate$Temp).
`cdate`	The climate date variable (dd/mm/yyyy). Please specify the parent environment and variable name (e.g. Climate$Date).
`bdate`	The biological date variable (dd/mm/yyyy). Please specify the parent environment and variable name (e.g. Biol$Date).
`range`	Two values signifying respectively the furthest and closest number of time intervals (set by cinterval) back from the cutoff date or biological record to include in the climate window search.
`stat`	The aggregate statistic used to analyse the climate data. Can currently use basic R statistics (e.g. mean, min), as well as slope. Additional aggregate statistics can be created using the format function(x) (...). See FUN in `apply` for more detail.

`stat2` Second aggregate statistic used to analyse climate data (xvar2). Can currently use basic R statistics (e.g. mean, min), as well as slope. Additional aggregate statistics can be created using the format function(x) (...). See FUN in `apply` for more detail.

`type` "absolute" or "relative", whether you wish the climate window to be relative (e.g. the number of days before each biological record is measured) or absolute (e.g. number of days before a set point in time).

`refday` If type is absolute, the day and month respectively of the year from which the absolute window analysis will start.

`cinterval` The resolution at which climate window analysis will be conducted. May be days ("day"), weeks ("week"), or months ("month"). Note the units of parameter 'range' will differ depending on the choice of cinterval

`cmissing` cmissing Determines what should be done if there are missing climate data. Three approaches are possible: - FALSE; the function will not run if missing climate data is encountered. An object 'missing' will be returned containing the dates of missing climate. - "method1"; missing climate data will be replaced with the mean climate of the preceding and following 2 days. - "method2"; missing climate data will be replaced with the mean climate of all records on the same date.

`spatial` A list item containing: 1. A factor that defines which spatial group (i.e. population) each biological record is taken from. The length of this factor should correspond to the length of the biological dataset. 2. A factor that defines which spatial group (i.e. population) climate data corresponds to. This length of this factor should correspond to the length of the climate dataset.

`cohort` A variable used to group biological records that occur in the same biological season but cover multiple years (e.g. southern hemisphere breeding season). By default, autowin will use year (extracted from parameter bdate) as the cohort variable. The cohort variable should be in the same dataset as the variable bdate.

`cutoff.day`, `cutoff.month`
Redundant parameters. Now replaced by refday.

`furthest`, `closest`
Redundant parameters. Now replaced by range.

Value

Will return a dataframe containing the correlation between the two climate variables.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
#Simple test example
#Create data from a subset of our test dataset
#Just use two years
biol_data <- Mass[1:2, ]
```

```
clim_data <- MassClimate[grep(pattern = "1979|1986", x = MassClimate$Date), ]

cross <- crosswin(xvar  = list(Temp = clim_data$Temp),
                  xvar2 = list(Rain = clim_data$Rain),
                  cdate = clim_data$Date, bdate = biol_data$Date,
                  range = c(1, 0),
                  stat = "mean", stat2 = "mean",
                  type = "relative",
                  cmissing = FALSE, cinterval = "day")

## Not run:
# Full working example
# Test correlation between temperature and rainfall in the MassClimate dataset.

data(Mass)
data(MassClimate)

cross <- crosswin(xvar = list(Temp = MassClimate$Temp),
                  xvar2 = list(Rain = MassClimate$Rain),
                  cdate = MassClimate$Date, bdate = Mass$Date,
                  range = c(365, 0),
                  stat = "mean", stat2 = "mean", type = "relative",
                  cmissing = FALSE, cinterval = "day")

# View the output
head(cross)

# Plot the output
plotcor(cross, type = "C")


## End(Not run)
```

`explore` *Visualise the weight distribution for given parameter values*

Description

Create a plot of the Weibull or Generalised Extreme Values (GEV) distribution for given values of shape, scale and location parameters. Used to determine initial parameter values for `weightwin`.

Usage

```
explore(shape = 1, scale = 1, loc = 0, weightfunc = "W")
```

Arguments

`shape`	A parameter that determines the shape of the distribution. Should be greater than 0.

scale	A parameter that determines the scale of the distribution. Should be greater than 0.
loc	A parameter that determines the location of the distribution. Should be less than or equal to 0.
weightfunc	Choose whether to use a weibull ("W") or GEV ("G") distribution.

Value

explore will return an example plot of the distribution using given parameter values. This can be used to select the initial parameter values for weightwin

Author(s)

Martijn van de Pol and Liam D. Bailey

Examples

```
# Test a weibull distribution

explore(shape = 3, scale = 0.2, loc = 0, weightfunc = "W")

# Test a GEV distribution

explore(shape = 3, scale = 5, loc = -5, weightfunc = "G")
```

Mass *Chick body mass data since 1979.*

Description

Artificially generated data representing average body mass of bird chicks since 1979.

Format

A data frame with 47 rows and 2 variables

Date Date of mass measurements (dd/mm/yyyy).

Mass Annual average body mass in grams.

Age Annual average age of mother in years.

MassClimate *Daily climate data since 1979.*

Description

Daily temperature and rainfall data since 1979.

Format

A data frame with 17,532 rows and 3 variables.

Date Date when climate data was recorded (dd/mm/yyyy).

Rain Daily rainfall data in mm.

Temp Daily temperature data in degrees centigrade.

MassOutput *Example output dataframe from function slidingwin.*

Description

Output file from `slidingwin` using temperature and body mass data. Generated with `Mass` and `MassClimate` dataframes.

Format

A data frame with 5,151 rows and 19 variables.

deltaAICc Difference between model AICc of fitted climate window and a null model containing no climate.

WindowOpen The start day of each tested climate window. Furthest from the biological record.

WindowClose The end day of each tested climate window. Closest to the biological record.

ModelBeta Beta estimate of the relationship between temperature and mass.

Std.Error Standard error term for linear model betas.

ModelBetaQ Quadratic beta estimate of the relationship between temperature and mass.

ModelBetaC Cubic beta estimate of the relationship between temperature and mass.

ModelInt Model intercept.

Function The function used to fit climate (e.g. linear ("lin"), quadratic ("quad"))

Furthest Furthest day back considered in slidingwin.

Closest Closest day back considered in slidingwin.

Statistics The aggregate statistic used to analyse climate (e.g. mean, max, slope).

Type Whether "absolute" or "relative" climate windows were tested.

K Number of folds used for k-fold cross validation.

ModWeight Model weight of each fitted climate window.

sample.size Sample size (i.e. number of years or sites) used for climate window analysis.

Reference.day,Reference.month If type is "absolute", the date from which the climate window was tested.

Randomised Whether the data was generated using `slidingwin` or `randwin`.

`MassRand` *Example output dataframe from function randwin.*

Description

Output file from function `randwin` using temperature and mass data. Generated with `Mass` and `MassClimate` dataframes.

Format

A data frame with 5 rows and 21 variables.

deltaAICc Difference between model AICc of fitted climate window and a null model containing no climate.

WindowOpen The start day of each tested climate window. Furthest from the biological record.

WindowClose The end day of each tested climate window. Closest to the biological record.

ModelBeta Beta estimate of the relationship between temperature and mass.

Std.Error Standard error term for linear model betas.

ModelBetaQ Quadratic beta estimate of the relationship between temperature and mass.

ModelBetaC Cubic beta estimate of the relationship between temperature and mass.

ModelInt Model intercept.

Function The function used to fit climate (e.g. linear ("lin"), quadratic ("quad"))

Furthest Furthest day back considered in slidingwin.

Closest Closest day back considered in slidingwin.

Statistics The aggregate statistic used to analyse climate (e.g. mean, max, slope).

Type Whether "fixed" or "variable" climate windows were tested.

K Number of folds used for k-fold cross validation.

ModWeight Model weight of each fitted climate window.

sample.size Sample size (i.e. number of years or sites) used for climate window analysis.

Reference.day,Reference.month If type is "absolute", the date from which the climate window was tested.

Randomised Whether the data was generated using `slidingwin` or `randwin`.

Repeat The number of randomisations carried out.

WeightDist Model spread of 95 percent confidence set of models.

medwin *Determine the median start and end time for climate windows*

Description

Determine the median start and end time for climate windows within a chosen confidence set.

Usage

```
medwin(dataset, cw = 0.95)
```

Arguments

dataset	Output dataframe of function slidingwin.
cw	Cut-off for confidence set (0.95 by default)

Value

Returns two values representing the median start and end time of climate windows within the confidence set.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
# Determine median start and end time of MassOutput from the 95% confidence set

medwin(MassOutput, cw = 0.95)
```

merge_results *Merge two slidingwin analyses.*

Description

Merges outputs of two separate slidingwin analyses.

Usage

```
merge_results(dataset1, dataset2)
```

Arguments

dataset1, dataset2	The slidingwin outputs to be merged. Note that all elements (i.e. Dataset, BestModel, BestModelData) will be merged and do not need to be specified.

Value

A list object, identical to that produced by slidingwin, containing all records from both outputs.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
#Simple test example
#Create data from a subset of our test dataset
#Just use two years
biol_data <- Mass[1:2, ]
clim_data <- MassClimate[grep(pattern = "1979|1986", x = MassClimate$Date), ]

output <- slidingwin(xvar = list(Temp = clim_data$Temp),
                     cdate = clim_data$Date,
                     bdate = biol_data$Date,
                     baseline = lm(Mass ~ 1, data = biol_data),
                     range = c(1, 0),
                     type = "relative", stat = "mean",
                     func = c("lin"), cmissing = FALSE, cinterval = "day")

#Merge MassOutput
merge_results(output, output)

## Not run:

data(Offspring)
data(OffspringClimate)

# Test a linear functions

OffspringWin_lin <- slidingwin(xvar = list(Temp = OffspringClimate$Temperature),
                               cdate = OffspringClimate$Date,
                               bdate = Offspring$Date,
                          baseline = glm(Offspring ~ 1, data = Offspring, family = poisson),
                               range = c(150, 0),
                               type = "relative", stat = "mean",
                               func = c("lin"), cmissing = FALSE, cinterval = "day")

# Test a quadratic functions

OffspringWin_quad <- slidingwin(xvar = list(Temp = OffspringClimate$Temperature),
                                cdate = OffspringClimate$Date,
                                bdate = Offspring$Date,
                           baseline = glm(Offspring ~ 1, data = Offspring, family = poisson),
                                range = c(150, 0),
                                type = "relative", stat = "mean",
                                func = c("quad"), cmissing = FALSE, cinterval = "day")

# Combine these outputs
```

```
OffspringWin_comb <- merge_results(dataset1 = OffspringWin_lin, dataset2 = OffspringWin_quad)

#View analyses contained in the new output

OffspringWin_comb$combos

#View output from linear analysis

head(OffspringWin_comb[[1]]$Dataset)


## End(Not run)
```

`Monthly_data` *Monthly temperature data*

Description

Artificially generated temperature data at a monthly scale. Used for code testing.

Format

A data frame with 576 rows and 2 variables

Date Date of temperature measurements (dd/mm/yyyy).

Temp Mean monthly temperature

`Offspring` *Reproductive success of birds since 2009.*

Description

Artificially generated data representing reproductive success of birds since 2009.

Format

A data frame with 1,619 rows and 5 variables.

Offspring Total number of offspring produced.

Date Date of hatching (dd/mm/yyyy).

Order Order of nest within each season.

BirdID Individual ID of female.

Cohort Grouping factor designating the breeding season of each record.

OffspringClimate *Daily climate data since 2009.*

Description

Daily temperature and rainfall data since 2009. Coincides with biological data from Offspring.

Format

A data frame with 2,588 rows and 3 variables.

Date Date when climate was recorded (dd/mm/yyyy).

Rain Daily rainfall data in mm.

Temperature Daily temperature data in degrees centigrade.

plotall *Visualise climate window data*

Description

Creates a panel of plots to help visualise climate window data.

Usage

```
plotall(
  dataset,
  datasetrand = NULL,
  bestmodel = NULL,
  bestmodeldata = NULL,
  cw1 = 0.95,
  cw2 = 0.5,
  cw3 = 0.25,
  title = NULL,
  arrow = FALSE,
  verbose = TRUE
)
```

Arguments

dataset	A dataframe containing information on all fitted climate windows. Output from slidingwin.
datasetrand	A dataframe containing information on all fitted climate windows using randomised data. Output from randwin.
bestmodel	A model object. The strongest climate window model. Returned from singlewin or slidingwin.

bestmodeldata	A dataframe containing the biological and climate data used to fit the strongest climate window model. Output from singlewin or slidingwin.
cw1, cw2, cw3	Cumulative weight levels used to visualise model weight distribution. See plotweights for more detail.
title	Title of the plot panel.
arrow	TRUE or FALSE. Add arrows to plots to pinpoint best window.
verbose	TRUE or FALSE. Should messages and warnings be printed while running function? Default: TRUE

Value

Will return a panel of 6-8 plots:

- DeltaAICc: A colour plot of model deltaAICc values (larger negative values indicate stronger models). DeltaAICc is the difference between AICc of each climate window model and the baseline model containing no climate data.
- Model weight: A plot showing the distribution of cumulative model weights. Gradient levels determined by parameters cw1, cw2 and cw3. Darker areas have a higher chance of containing the best climate window. Also returns the percentage of models within the 95
- Model betas: A colour plot of model beta estimates. Where applicable, 2nd order coefficients (quadratic) and 3rd order coefficients (cubic) will be plotted separately.
- Histogram(s): If datasetrand is provided, plotall will return a histogram showing the deltaAICc of randomised data. This can help determine the likelihood of obtaining a deltaAICc value for a fitted climate window model at random. plotall will also use pvalue to return values of Pc and PdeltaAICc.
- Boxplots: Two boxplots showing the start and end time for a subset of best climate windows. Best climate windows make up the cumulative model weight equivalent to the largest value of cw1, cw2 and cw3. Values above boxplots represent the median values.
- Best Model: If bestmodel and bestmodeldata are provided, plotall will create a scatterplot to show the fit of the best model through the data.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
# Visualise a fixed climate window generated for dataframes Mass and MassClimate

data(MassOutput)
data(Mass)
data(MassClimate)

single <- singlewin(xvar = list(Temp = MassClimate$Temp),
                    cdate = MassClimate$Date, bdate = Mass$Date,
                    baseline = lm(Mass ~ 1, data = Mass),
                    range = c(72, 15),
```

```
                    stat = "mean", func = "lin",
                    type = "absolute", refday = c(20, 5),
                    cmissing = FALSE, cinterval = "day")

plotall(dataset = MassOutput, bestmodel = single$BestModel,
       bestmodeldata = single$BestModelData,
       cw1 = 0.95, cw2 = 0.5, cw3 = 0.25, title = "Mass")
```

`plotbest` *Visualise the best climate window*

Description

Create a scatterplot showing the fit of the best climate window model through the biological data.

Usage

```
plotbest(dataset, bestmodel, bestmodeldata)
```

Arguments

`dataset`	A dataframe containing information on all fitted climate windows. Output from `slidingwin`.
`bestmodel`	A model object. The strongest climate window model. Output from `singlewin` or `slidingwin`.
`bestmodeldata`	A dataframe with the data used to fit the strongest climate window model. Output from `singlewin` or `slidingwin`.

Value

Returns a scatterplot with a fitted line to show the fit of the best model through the data.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
# Visualise the best climate window from the datasets Mass and MassClimate

data(MassOutput)
data(Mass)
data(MassClimate)

single <- singlewin(xvar = list(Temp = MassClimate$Temp),
                    cdate = MassClimate$Date, bdate = Mass$Date,
                    baseline = lm(Mass ~ 1, data = Mass),
```

```
                    range = c(72, 15),
                    stat = "mean", func = "lin",
                    type = "absolute", refday = c(20, 5),
                    cmissing = FALSE, cinterval = "day")

plotbest(dataset = MassOutput, bestmodel = single$BestModel,
        bestmodeldata = single$BestModelData)
```

`plotbetas` *Plot model beta estimates*

Description

Create colour plots of model beta estimates. Will include quadratic and cubic beta estimates where appropriate.

Usage

```
plotbetas(dataset, arrow = FALSE, plotallenv, plotall = FALSE)
```

Arguments

`dataset`	A dataframe containing information on all fitted climate windows. Output from `slidingwin`.
`arrow`	TRUE or FALSE. Add arrows to plots to pinpoint best window.
`plotallenv`	Used in conjunction with function `plotall`. Should not be changed manually.
`plotall`	Used in conjunction with function `plotall`. Should not be changed manually.

Value

Returns colour plots of model beta estimates. Where applicable, 2nd order coefficients (quadratic) and 3rd order coefficients (cubic) will be plotted separately.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
# Plot model beta estimates for linear models in the Mass dataset

data(MassOutput)

plotbetas(dataset = MassOutput)
```

plotcor *Visualise climate cross correlation or autocorrelation.*

Description

Create a colour plot to visualise the results of autowin or crosswin. Displays correlation across all desired climate windows.

Usage

```
plotcor(cor.output, type, arrow = FALSE)
```

Arguments

cor.output	Output of autowin or crosswin
type	Should be either "A" for data generated by autowin or "C" for data generated by crosswin.
arrow	TRUE or FALSE. Add arrows to plots to pinpoint best window.

Value

Will generate a colour plot to visualise the correlation data.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
#Simple test example
#Create data from a subset of our test dataset
#Just use two years
biol_data <- Mass[1:2, ]
clim_data <- MassClimate[grep(pattern = "1979|1986", x = MassClimate$Date), ]

single <- singlewin(xvar = list(Temp = clim_data$Temp),
                    cdate = clim_data$Date,
                    bdate = biol_data$Date,
                    baseline = lm(Mass ~ 1, data = biol_data),
                    range = c(1, 0),
                    type = "relative", stat = "mean",
                    func = c("lin"), cmissing = FALSE, cinterval = "day")

auto <- autowin(reference = single,
                xvar  = list(Temp = clim_data$Temp),
                cdate = clim_data$Date, bdate = biol_data$Date,
                baseline = lm(Mass ~ 1, data = biol_data), range = c(1, 0),
                stat = "mean", func = "lin",
                type = "relative",
```

```
                cmissing = FALSE, cinterval = "day")

plotcor(auto, type = "A")

## Not run:
# Full working example
# Visualise climate autocorrelation

data(Mass)
data(MassClimate)

# Fit a single climate window using the datasets Mass and MassClimate.

single <- singlewin(xvar = list(Temp = MassClimate$Temp),
                    cdate = MassClimate$Date, bdate = Mass$Date,
                    baseline = lm(Mass ~ 1, data = Mass),
                    range = c(72, 15),
                    stat = "mean", func = "lin",
                    type = "absolute", refday = c(20, 5),
                    cmissing = FALSE, cinterval = "day")

# Test the autocorrelation between the climate in this single window and other climate windows.

auto <- autowin(reference = single,
                xvar  = list(Temp = MassClimate$Temp),
                cdate = MassClimate$Date, bdate = Mass$Date,
                baseline = lm(Mass ~ 1, data = Mass),
                range = c(365, 0),
                stat = "mean", func = "lin",
                type = "absolute", refday = c(20, 5),
                cmissing = FALSE, cinterval = "day")

# Plot the auto-correlation data

plotcor(auto, type = "A")

## End(Not run)
```

`plotdelta` *Plot deltaAICc of models*

Description

Create a colour plot of model deltaAICc values.

Usage

```
plotdelta(dataset, arrow = FALSE, plotall = FALSE, plotallenv, ThreeD = FALSE)
```

Arguments

dataset	A dataframe containing information on all fitted climate windows. Output from slidingwin.
arrow	TRUE or FALSE. Add arrows to plots to pinpoint best window.
plotall	Used in conjunction with function plotall. Should not be changed manually.
plotallenv	Used in conjunction with function plotall. Should not be changed manually.
ThreeD	TRUE or FALSE. Generate a 3-dimensional plot of the deltaAICc landscape.

Value

Returns a colour plot of model deltaAICc values (larger negative values indicate stronger models). DeltaAICc is the difference between AICc of each climate window and a null model.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
# Plot deltaAICc estimates for climate windows in the Mass dataset

data(MassOutput)

plotdelta(dataset = MassOutput)
```

---

plothist *Create a histogram of randomised deltaAICc values*

Description

Create a histogram of deltaAICc values from randomised data.

Usage

```
plothist(dataset, datasetrand, verbose = TRUE)
```

Arguments

dataset	A dataframe containing information on all fitted climate windows from observed data. Output from slidingwin.
datasetrand	A dataframe containing information on all fitted climate windows using randomised data. Output from randwin.
verbose	TRUE or FALSE. Should messages and warnings be printed while running function? Default: TRUE

Value

plothist will return a histograms of deltaAICc values from randomised data. Values of PdeltaAICc and Pc will be provided to help determine the likelihood that an observed deltaAICc value would occur by chance.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
# Plot randomised data for the Mass dataset

data(MassOutput)
data(MassRand)

plothist(datasetrand = MassRand, dataset = MassOutput)
```

---

plotweights *Plot distribution of model weights*

Description

Create a plot showing the distribution of cumulative model weights for all fitted climate windows.

Usage

```
plotweights(
  dataset,
  cw1 = 0.95,
  cw2 = 0.5,
  cw3 = 0.25,
  arrow = FALSE,
  plotall = FALSE,
  plotallenv,
  ThreeD = FALSE
)
```

Arguments

dataset	A dataframe containing information on all fitted climate windows. Output from slidingwin.
cw1, cw2, cw3	Cumulative weight levels used to visualise model weight distribution. Cumulative weights represent the chance that the best model is contained within a set. For example, there is a 95 percent chance that the best climate window model is contained within the cumulative weight level of 0.95. Parameter values must <= 1.

arrow	TRUE or FALSE. Add arrows to plots to pinpoint best window.
plotall	Used in conjunction with function plotall. Should not be changed manually.
plotallenv	Used in conjunction with function plotall. Should not be changed manually.
ThreeD	TRUE or FALSE. Generate a 3-dimensional plot of the model weight landscape.

Value

Returns a plot showing the distribution of cumulative model weights. Levels determined by parameters cw1,cw2 and cw3.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
# Plot distribution of model weights for Mass dataset

data(MassOutput)

plotweights(dataset = MassOutput, cw1 = 0.95, cw2 = 0.75, cw3 = 0.25)
```

---

plotwin *Plot the start and end time of best climate windows*

Description

Visualise the start and end time for a subset of best climate windows.

Usage

```
plotwin(dataset, cw = 0.95)
```

Arguments

dataset	A dataframe containing information on all fitted climate windows. Output from slidingwin.
cw	Cumulative model weight used to subset the group of best models.

Value

Creates two boxplots showing the start and end time for a subset of best climate windows. Best climate windows make up the cumulative model weight equivalent to the value of cw.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
# View window limits for climate windows in the top 95% of model weights.

data(MassOutput)

plotwin(dataset = MassOutput, cw = 0.95)
```

`pvalue` *Determine the probability that a given climate signal is 'true'.*

Description

Calculate probability that a given climate signal is 'true' using either PDAICc or Pc.

Usage

```
pvalue(dataset, datasetrand, metric, sample.size)
```

Arguments

`dataset`	A dataframe containing information on all fitted climate windows. Output from `slidingwin`.
`datasetrand`	A dataframe containing information on all fitted climate windows using randomised data. Output from `randwin`.
`metric`	"AIC" or "C". Determine whether a value of PDAICc or Pc will be returned.
`sample.size`	Sample size of analysis.

Value

Returns a value representing the probability that a given climate window result is a false positive.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
# Calculate PDAICc for the Mass dataset

pvalue(datasetrand = MassRand, dataset = MassOutput,
      metric = "AIC", sample.size = 47)

# Calculate Pc for the Mass dataset

pvalue(datasetrand = MassRand, dataset = MassOutput,
      metric = "C", sample.size = 47)
```

randwin *Climate window analysis for randomised data*

Description

Randomises biological data and carries out a climate window analysis. Used to help determine the chance of obtaining an observed result at random.

Usage

```
randwin(
  exclude = NA,
  repeats = 5,
  window = "sliding",
  xvar,
  cdate,
  bdate,
  baseline,
  stat,
  range,
  func,
  type,
  refday,
  cmissing = FALSE,
  cinterval = "day",
  spatial = NULL,
  cohort = NULL,
  upper = NA,
  lower = NA,
  binary = FALSE,
  centre = list(NULL, "both"),
  k = 0,
  weightfunc = "W",
  par = c(3, 0.2, 0),
  control = list(ndeps = c(0.01, 0.01, 0.01)),
  method = "L-BFGS-B",
  cutoff.day = NULL,
  cutoff.month = NULL,
  furthest = NULL,
  closest = NULL,
  thresh = NULL,
  cvk = NULL
)
```

Arguments

exclude	Two values (distance and duration) which allow users to exclude short-duration long-lag climate windows from analysis (e.g., windows with a duration of 10

days which occur over a month ago). These windows are often considered to be biologically implausible.

`repeats` The number of times that data will be randomised and analysed for climate windows.

`window` Whether randomisations are carried out for a sliding window ("sliding") or weighted window ("weighted") approach.

`xvar` A list object containing all climate variables of interest. Please specify the parent environment and variable name (e.g. Climate$Temp).

`cdate` The climate date variable (dd/mm/yyyy). Please specify the parent environment and variable name (e.g. Climate$Date).

`bdate` The biological date variable (dd/mm/yyyy). Please specify the parent environment and variable name (e.g. Biol$Date).

`baseline` The baseline model structure used for testing correlation. Currently known to support lm, glm, lmer and glmer objects.

`stat` If window = "sliding"; The aggregate statistic used to analyse the climate data. Can currently use basic R statistics (e.g. mean, min), as well as slope. Additional aggregate statistics can be created using the format function(x) (...). See FUN in `apply` for more detail.

`range` Two values signifying respectively the furthest and closest number of time intervals (set by cinterval) back from the cutoff date or biological record to include in the climate window search.

`func` The functions used to fit the climate variable. Can be linear ("lin"), quadratic ("quad"), cubic ("cub"), inverse ("inv") or log ("log").

`type` "absolute" or "relative", whether you wish the climate window to be relative (e.g. the number of days before each biological record is measured) or absolute (e.g. number of days before a set point in time).

`refday` If type is absolute, the day and month respectively of the year from which the absolute window analysis will start.

`cmissing` cmissing Determines what should be done if there are missing climate data. Three approaches are possible: - FALSE; the function will not run if missing climate data is encountered. An object 'missing' will be returned containing the dates of missing climate. - "method1"; missing climate data will be replaced with the mean climate of the preceding and following 2 days. - "method2"; missing climate data will be replaced with the mean climate of all records on the same date.

`cinterval` The resolution at which climate window analysis will be conducted. May be days ("day"), weeks ("week"), or months ("month"). Note the units of parameter 'range' will differ depending on the choice of cinterval.

`spatial` A list item containing: 1. A factor that defines which spatial group (i.e. population) each biological record is taken from. The length of this factor should correspond to the length of the biological dataset. 2. A factor that defines which spatial group (i.e. population) climate data corresponds to. This length of this factor should correspond to the length of the climate dataset.

`cohort` A variable used to group biological records that occur in the same biological season but cover multiple years (e.g. southern hemisphere breeding season). Only required when type is "absolute". The cohort variable should be in the same dataset as the variable bdate.

`upper` Cut-off values used to determine growing degree days or positive climate thresholds (depending on parameter thresh). Note that when values of lower and upper are both provided, climatewin will instead calculate an optimal climate zone.

`lower` Cut-off values used to determine chill days or negative climate thresholds (depending on parameter thresh). Note that when values of lower and upper are both provided, climatewin will instead calculate an optimal climate zone.

`binary` TRUE or FALSE. Determines whether to use values of upper and lower to calculate binary climate data (thresh = TRUE), or to use for growing degree days (thresh = FALSE).

`centre` A list item containing: 1. The variable used for mean centring (e.g. Year, Site, Individual). Please specify the parent environment and variable name (e.g. Biol$Year). 2. Whether the model should include both within-group means and variance ("both"), only within-group means ("mean"), or only within-group variance ("dev").

`k` If window = "sliding"; the number of folds used for k-fold cross validation. By default this value is set to 0, so no cross validation occurs. Value should be a minimum of 2 for cross validation to occur.

`weightfunc` If window = "weighted"; the distribution to be used for optimisation. Can be either a Weibull ("W") or Generalised Extreme Value distribution ("G").

`par` If window = "weighted"; the shape, scale and location parameters of the Weibull or GEV weight function used as start weight function. For Weibull : Shape and scale parameters must be greater than 0, while location parameter must be less than or equal to 0. For GEV : Scale parameter must be greater than 0.

`control` If window = "weighted"; parameters used to determine step size for the optimisation function. Please see `optim` for more detail.

`method` If window = "weighted"; the method used for the optimisation function. Please see `optim` for more detail.

`cutoff.day`, `cutoff.month`
Redundant parameters. Now replaced by refday.

`furthest`, `closest`
Redundant parameters. Now replaced by range.

`thresh` Redundant parameter. Now replaced by binary.

`cvk` Redundant parameter. Now replaced by k.

Value

Returns a dataframe containing information on the best climate window from each randomisation. See `MassRand` as an example.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
#Simple test example
#Create data from a subset of our test dataset
#Just use two years
biol_data <- Mass[1:2, ]
clim_data <- MassClimate[grep(pattern = "1979|1986", x = MassClimate$Date), ]

rand <- randwin(repeats = 1, xvar = list(Temp = clim_data$Temp),
                cdate = clim_data$Date,
                bdate = biol_data$Date,
                baseline = lm(Mass ~ 1, data = biol_data),
                range = c(1, 0),
                type = "relative", stat = "mean",
                func = c("lin"), cmissing = FALSE, cinterval = "day")

## Not run:

# Full working examples

## EXAMPLE 1 ##

# Test climate windows in randomised data using a sliding window approach.

data(Mass)
data(MassClimate)

# Randomise data twice
# Note all other parameters are fitted in the same way as the climatewin function.

rand <- randwin(repeats = 2, window = "sliding",
                xvar = list(Temp = MassClimate$Temp),
                cdate = MassClimate$Date, bdate = Mass$Date,
                baseline = lm(Mass ~ 1, data = Mass),
                range = c(100, 0),
                stat = "mean", func = "lin", type = "absolute",
                refday = c(20, 5),
                cmissing = FALSE, cinterval = "day")

# View output #

head(rand)

## EXAMPLE 2 ##

# Test climate windows in randomised data using a weighted window approach.

data(Offspring)
data(OffspringClimate)

# Randomise data twice
# Note all other parameters are fitted in the same way as the weightwin function.
```

```
weightrand <- randwin(repeats = 2, window = "weighted",
                      xvar = list(Temp = OffspringClimate$Temperature),
                      cdate = OffspringClimate$Date,
                      bdate = Offspring$Date,
                      baseline = glm(Offspring ~ 1, family = poisson, data = Offspring),
                      range = c(365, 0), func = "quad",
                      type = "relative", weightfunc = "W", cinterval = "day",
                      par = c(3, 0.2, 0), control = list(ndeps = c(0.01, 0.01, 0.01)),
                      method = "L-BFGS-B")

# View output

head(weightrand)


## End(Not run)
```

`singlewin` *Fit a single climate window*

Description

Fit a single climate window with a known start and end time.

Usage

```
singlewin(
  xvar,
  cdate,
  bdate,
  baseline,
  range,
  stat,
  func,
  type,
  refday,
  cmissing = FALSE,
  cinterval = "day",
  cohort = NULL,
  spatial = NULL,
  upper = NA,
  lower = NA,
  binary = FALSE,
  centre = list(NULL, "both"),
  cutoff.day = NULL,
  cutoff.month = NULL,
  furthest = NULL,
```

```
  closest = NULL,
  thresh = NULL
)
```

Arguments

xvar	A list object containing all climate variables of interest. Please specify the parent environment and variable name (e.g. Climate$Temp).
cdate	The climate date variable (dd/mm/yyyy). Please specify the parent environment and variable name (e.g. Climate$Date).
bdate	The biological date variable (dd/mm/yyyy). Please specify the parent environment and variable name (e.g. Biol$Date).
baseline	The baseline model structure used for testing correlation. Currently known to support lm, glm, lmer and glmer objects.
range	Two values signifying respectively the furthest and closest number of time intervals (set by cinterval) back from the cutoff date or biological record to include in the climate window search.
stat	The aggregate statistic used to analyse the climate data. Can currently use basic R statistics (e.g. mean, min), as well as slope. Additional aggregate statistics can be created using the format function(x) (...). See FUN in apply for more detail.
func	The functions used to fit the climate variable. Can be linear ("lin"), quadratic ("quad"), cubic ("cub"), inverse ("inv") or log ("log").
type	"absolute" or "relative", whether you wish the climate window to be relative (e.g. the number of days before each biological record is measured) or absolute (e.g. number of days before a set point in time).
refday	If type is absolute, the day and month respectively of the year from which the absolute window analysis will start.
cmissing	cmissing Determines what should be done if there are missing climate data. Three approaches are possible: - FALSE; the function will not run if missing climate data is encountered. An object 'missing' will be returned containing the dates of missing climate. - "method1"; missing climate data will be replaced with the mean climate of the preceding and following 2 days. - "method2"; missing climate data will be replaced with the mean climate of all records on the same date.
cinterval	The resolution at which climate window analysis will be conducted. May be days ("day"), weeks ("week"), or months ("month"). Note the units of parameter 'range' will differ depending on the choice of cinterval.
cohort	A variable used to group biological records that occur in the same biological season but cover multiple years (e.g. southern hemisphere breeding season). Only required when type is "absolute". The cohort variable should be in the same dataset as the variable bdate.
spatial	A list item containing: 1. A factor that defines which spatial group (i.e. population) each biological record is taken from. The length of this factor should correspond to the length of the biological dataset. 2. A factor that defines which

spatial group (i.e. population) climate data corresponds to. This length of this factor should correspond to the length of the climate dataset.

`upper` Cut-off values used to determine growing degree days or positive climate thresholds (depending on parameter thresh). Note that when values of lower and upper are both provided, climatewin will instead calculate an optimal climate zone.

`lower` Cut-off values used to determine chill days or negative climate thresholds (depending on parameter thresh). Note that when values of lower and upper are both provided, climatewin will instead calculate an optimal climate zone.

`binary` TRUE or FALSE. Determines whether to use values of upper and lower to calculate binary climate data (thresh = TRUE), or to use for growing degree days (thresh = FALSE).

`centre` A list item containing: 1. The variable used for mean centring (e.g. Year, Site, Individual). Please specify the parent environment and variable name (e.g. Biol$Year). 2. Whether the model should include both within-group means and variance ("both"), only within-group means ("mean"), or only within-group variance ("dev").

`cutoff.day`, `cutoff.month`
Redundant parameters. Now replaced by refday.

`furthest`, `closest`
Redundant parameters. Now replaced by range.

`thresh` Redundant parameter. Now replaced by binary.

Value

Will return a list containing two objects:

- BestModel, a model object of the fitted climate window model.
- BestModelData, a dataframe with the biological and climate data used to fit the climate window model.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
#Simple test example
#Create data from a subset of our test dataset
#Just use two years
biol_data <- Mass[1:2, ]
clim_data <- MassClimate[grep(pattern = "1979|1986", x = MassClimate$Date), ]

single <- singlewin(xvar = list(Temp = clim_data$Temp),
                    cdate = clim_data$Date,
                    bdate = biol_data$Date,
                    baseline = lm(Mass ~ 1, data = biol_data),
                    range = c(1, 0),
                    type = "relative", stat = "mean",
```

```
                    func = c("lin"), cmissing = FALSE, cinterval = "day")

## Not run:
# Full working example
# Fit a known climate window to the datasets Mass and MassClimate

data(Mass)
data(MassClimate)

# Test for a fixed climate window, starting from 20th May
# Fit a climate window starting 72 days ago and ending 15 days ago
# Fit a linear term for the mean climate
# Fit climate windows at the resolution of days

single <- singlewin(xvar = list(Temp = MassClimate$Temp),
                    cdate = MassClimate$Date, bdate = Mass$Date,
                    baseline = lm(Mass ~ 1, data = Mass),
                    range = c(72, 15),
                    stat = "mean", func = "lin",
                    type = "absolute", refday = c(20, 5),
                    cmissing = FALSE, cinterval = "day")

##View data##
single$BestModel
head(single$BestModelData)

## End(Not run)
```

`Size` *Average size of red winged fairy wren (Malurus elegans) chicks.*

Description

Average size (using standardised measures of tarsus length, head-bill length and wing length) in red winged fairy wren (Malurus elegans) chicks. Measured over 7 years.

Format

A data frame with 1,003 rows and 5 variables.

NestID Unique nest identifier.

Cohort Year of breeding season.

Helpers Total number of non-breeding helpers at the nest.

Size Average offspring size.

Date Date when offspring size was recorded (dd/mm/yyyy).

SizeClimate *Daily climate data from 2006 to 2015.*

Description

Average, maximum and minimum daily temperature data, average rainfall data from 2006 to 2015. Coincides with biological data from Size.

Format

A data frame with 3,411 rows and 5 variables.

Date Date when climate was recorded (dd/mm/yyyy).

Rain Average daily rainfall data in mm.

Temperature Average daily temperature data in degrees centigrade.

MaxTemp Maximum daily temperature in degrees centigrade.

MinTemp Minimum daily temperature in degrees centigrade.

slidingwin *Test for a climate windows in data.*

Description

Finds the time period when a biological variable is most strongly affected by climate. Note that climate data and biological data should be loaded as two separate objects. Both objects should contain a date column to designate when the data were recorded (dd/mm/yyyy).

Usage

```
slidingwin(
  exclude = NA,
  xvar,
  cdate,
  bdate,
  baseline,
  type,
  refday,
  stat = "mean",
  func = "lin",
  range,
  cmissing = FALSE,
  cinterval = "day",
  k = 0,
  upper = NA,
```

```
  lower = NA,
  binary = FALSE,
  centre = list(NULL, "both"),
  spatial = NULL,
  cohort = NULL
)
```

Arguments

exclude	Two values (duration and distance) which allow users to exclude short-duration long-lag climate windows from analysis (e.g., windows with a duration of 10 days which occur over a month ago). These windows are often considered to be biologically implausible.
xvar	A list object containing all climate variables of interest. Please specify the parent environment and variable name (e.g. climate$Temp).
cdate	The climate date variable (dd/mm/yyyy). Please specify the parent environment and variable name (e.g. climate$Date).
bdate	The biological date variable (dd/mm/yyyy). Please specify the parent environment and variable name (e.g. Biol$Date).
baseline	The baseline model structure used for model testing. Currently known to support lm, glm, lmer, glmer and coxph objects.
type	"absolute" or "relative", whether you wish the climate window to be relative (e.g. the number of days before each biological record is measured) or absolute (e.g. number of days before a set point in time).
refday	If type is "absolute", the day and month respectively of the year from which the absolute window analysis will start.
stat	The aggregate statistics used to analyse the climate data. Can currently use basic R statistics (e.g. mean, min), as well as slope. Additional aggregate statistics can be created using the format function(x) (...). See FUN in apply for more detail.
func	The functions used to fit the climate variable. Can be linear ("lin"), quadratic ("quad"), cubic ("cub"), inverse ("inv") or log ("log").
range	Two values signifying respectively the furthest and closest number of time intervals (set by cinterval) back from the refday or biological record to include in the climate window search.
cmissing	Determines what should be done if there are missing climate data. Three approaches are possible: - FALSE; the function will not run if missing climate data is encountered. An object 'missing' will be returned containing the dates of missing climate. - "method1"; missing climate data will be replaced with the mean climate of the preceding and following 2 records. - "method2"; missing climate data will be replaced with the mean climate of all records on the same date. Note: Other methods are possible. Users should consider those methods most appropriate for their data and apply them manually before using climwin if required.

cinterval	The resolution at which climate window analysis will be conducted. May be days ("day"), weeks ("week"), or months ("month"). Note the units of parameter 'range' will differ depending on the choice of cinterval.
k	The number of folds used for k-fold cross validation. By default this value is set to 0, so no cross validation occurs. Value should be a minimum of 2 for cross validation to occur.
upper	Cut-off values used to determine growing degree days or positive climate thresholds (depending on parameter thresh). Note that when values of lower and upper are both provided, slidingwin will instead calculate an optimal climate zone.
lower	Cut-off values used to determine chill days or negative climate thresholds (depending on parameter thresh). Note that when values of lower and upper are both provided, slidingwin will instead calculate an optimal climate zone.
binary	TRUE or FALSE. Determines whether to use values of upper and lower to calculate binary climate data (binary = TRUE), or to use for growing degree days (binary = FALSE).
centre	A list item containing: 1. The variable used for mean centring (e.g. Year, Site, Individual). Please specify the parent environment and variable name (e.g. Biol$Year). 2. Whether the model should include both within-group means and variance ("both"), only within-group means ("mean"), or only within-group variance ("var").
spatial	A list item containing: 1. A factor that defines which spatial group (i.e. population) each biological record is taken from. The length of this factor should correspond to the length of the biological dataset. 2. A factor that defines which spatial group (i.e. population) climate data corresponds to. This length of this factor should correspond to the length of the climate dataset.
cohort	A variable used to group biological records that occur in the same biological season but cover multiple years (e.g. southern hemisphere breeding season). Only required when type is "absolute". The cohort variable should be in the same dataset as the variable bdate.

Details

Note that slidingwin allows you to test multiple possible parameters with the same code (e.g. func, stat, xvar). See examples for more detail.

Value

Will return a list with an output for each tested set of climate window parameters. Each list item contains three objects:

- BestModel, a model object. The strongest climate window model based on AICc.
- BestModelData, a dataframe used to fit the strongest climate window model.
- Dataset, a dataframe with information on all fitted climate windows. Ordered using deltaAICc, with most negative deltaAICc values first. See `MassOutput` as an example.

In addition, the returned list includes an object 'combos', a summary of all tested sets of climate window parameters.

Author(s)

Liam D. Bailey and Martijn van de Pol

Examples

```
#Simple test example
#Create data from a subset of our test dataset
#Just use two years
biol_data <- Mass[1:2, ]
clim_data <- MassClimate[grep(pattern = "1979|1986", x = MassClimate$Date), ]

output <- slidingwin(xvar = list(Temp = clim_data$Temp),
                     cdate = clim_data$Date,
                     bdate = biol_data$Date,
                     baseline = lm(Mass ~ 1, data = biol_data),
                     range = c(1, 0),
                     type = "relative", stat = "mean",
                     func = c("lin"), cmissing = FALSE, cinterval = "day")

## Not run:

# Full working examples

##EXAMPLE 1##

# Test both a linear and quadratic variable climate window using datasets "Offspring"
# and "OffspringClimate".

# Load data.

data(Offspring)
data(OffspringClimate)

# Test both linear and quadratic functions with climate variable temperature

OffspringWin <- slidingwin(xvar = list(Temp = OffspringClimate$Temperature),
                           cdate = OffspringClimate$Date,
                           bdate = Offspring$Date,
                        baseline = glm(Offspring ~ 1, data = Offspring, family = poisson),
                           range = c(150, 0),
                           type = "relative", stat = "mean",
                           func = c("lin", "quad"), cmissing = FALSE, cinterval = "day")

# Examine tested combinations

OffspringWin$combos

# View output for func = "lin"

head(OffspringWin[[1]]$Dataset)
summary(OffspringWin[[1]]$BestModel)
```

```
# View output for func = "quad"

head(OffspringWin[[2]]$Dataset)
summary(OffspringWin[[2]]$BestModel)

##EXAMPLE 2##

# Test for an absolute climate window with both 'mean' and 'max' aggregate statistics
# using datasets 'Mass' and 'MassClimate'.

# Load data.

data(Mass)
data(MassClimate)

# Test an absolute window, starting 20 May (refday = c(20, 5))
# Test for climate windows between 100 and 0 days ago (range = c(100, 0))
# Test both mean and max aggregate statistics (stat = c("mean", "max"))
# Fit a linear term (func = "lin")
# Test at the resolution of days (cinterval = "day")

MassWin <- slidingwin(xvar = list(Temp = MassClimate$Temp), cdate = MassClimate$Date,
                      bdate = Mass$Date, baseline = lm(Mass ~ 1, data = Mass),
                      range = c(100, 0),
                      stat = c("mean", "max"), func = "lin",
                      type = "absolute", refday = c(20, 5),
                      cmissing = FALSE, cinterval = "day")

# Examine tested combinations

MassWin$combos

# View output for mean temperature

head(MassWin[[1]]$Dataset)
summary(MassWin[[1]]$BestModel)

# View output for max temperature

head(MassWin[[2]]$Dataset)
summary(MassWin[[2]]$BestModel)


## End(Not run)
```

`weightwin` *Find a weighted climate window*

Description

Finds the best weighted average of a weather variable over a period that correlates most strongly with a biological variable. Uses weibull or Generalised Extreme Value (GEV) distribution. See references for a full description.

Usage

```
weightwin(
  n = 1,
  xvar,
  cdate,
  bdate,
  baseline,
  range,
  k = 0,
  func = "lin",
  type,
  refday,
  nrandom = 0,
  centre = NULL,
  weightfunc = "W",
  cinterval = "day",
  cmissing = FALSE,
  cohort = NULL,
  spatial = NULL,
  par = c(3, 0.2, 0),
  control = list(ndeps = c(0.001, 0.001, 0.001)),
  method = "L-BFGS-B",
  cutoff.day = NULL,
  cutoff.month = NULL,
  furthest = NULL,
  closest = NULL,
  grad = FALSE
)
```

Arguments

n	The number of iterations used to run weightwin. If n > 1, iterations will use randomly generated starting parameters. These are stored in the output data frame 'iterations'.
xvar	A list object containing all climate variables of interest. Please specify the parent environment and variable name (e.g. Climate$Temp).
cdate	The climate date variable. Please specify the parent environment and variable name (e.g. Climate$Date).
bdate	The biological date variable. Please specify the parent environment and variable name (e.g. Biol$Date).
baseline	The baseline model structure used for testing correlation. Currently known to support lm, lme, glm and glmer objects.

range	Two values signifying respectively the furthest and closest number of time intervals (set by cinterval) back from the cutoff date or biological record to include in the climate window search.
k	The number of folds used for k-fold cross validation. By default this value is set to 0, so no cross validation occurs. Value should be a minimum of 2 for cross validation to occur.
func	The function used to fit the climate variable in the model. Can be linear ("lin"), quadratic ("quad"), cubic ("cub"), inverse ("inv") or log ("log").
type	"absolute" or "relative", whether you wish the climate window to be relative (e.g. the number of days before each biological record is measured) or absolute (e.g. number of days before a set point in time).
refday	If type is absolute, the day and month respectively of the year from which the absolute window analysis will start.
nrandom	Used when conducting data randomisation, should not be changed manually.
centre	A list item containing: 1. The variable used for mean centring (e.g. Year, Site, Individual). Please specify the parent environment and variable name (e.g. Biol$Year). 2. Whether the model should include both within-group means and variance ("both"), only within-group means ("mean"), or only within-group variance ("var").
weightfunc	The distribution to be used for optimisation. Can be either a Weibull ("W") or Generalised Extreme Value distribution ("G").
cinterval	The resolution at which the climate window analysis will be conducted. May be days ("day"), weeks ("week"), or months ("month"). Note the units of parameter 'range' will differ depending on the choice of cinterval.
cmissing	Determines what should be done if there are missing climate data. Three approaches are possible: - FALSE; the function will not run if missing climate data is encountered. An object 'missing' will be returned containing the dates of missing climate. - "method1"; missing climate data will be replaced with the mean climate of the preceding and following 2 records. - "method2"; missing climate data will be replaced with the mean climate of all records on the same date. Note: Other methods are possible. Users should consider those methods most appropriate for their data and apply them manually before using climwin if required.
cohort	A variable used to group biological records that occur in the same biological season but cover multiple years (e.g. southern hemisphere breeding season). Only required when type is "absolute". The cohort variable should be in the same dataset as the variable bdate.
spatial	A list item containing: 1. A factor that defines which spatial group (i.e. population) each biological record is taken from. The length of this factor should correspond to the length of the biological dataset. 2. A factor that defines which spatial group (i.e. population) climate data corresponds to. This length of this factor should correspond to the length of the climate dataset.
par	Shape, scale and location parameters of the Weibull or GEV weight function used as start weight function. For Weibull : Shape and scale parameters must

be greater than 0, while location parameter must be less than or equal to 0. For GEV : Scale parameter must be greater than 0.

`control` Parameters used to determine step size for the optimisation function. Please see `optim` for more detail.

`method` The method used for the optimisation function. Please see `optim` for more detail.

`cutoff.day`, `cutoff.month`
Redundant parameters. Now replaced by refday.

`furthest`, `closest`
Redundant parameters. Now replaced by range.

`grad` Run the optimisation procedure with a numerically derived gradient function. This can improve model convergence but will increase computational time.

Value

Produces a constantly updating grid of plots as the optimisation function is running.

- Right panel from top to bottom: The three parameters (shape, scale and location) determining the weight function.
- Left top panel: The resulting weight function.
- Left middle panel: The delta AICc compared to the baseline model.
- Left bottom panel: Plotted relationship between the weighted mean of climate and the biological response variable.

Also returns a list containing three objects:

- BestModel, a model object. The best weighted window model determined by deltaAICc.
- BestModelData, a dataframe. Biological and climate data used to fit the best weighted window model.
- WeightedOutput. Parameter values for the best weighted window.
- iterations. If n > 1, the starting parameters and deltaAICc values from each iteration of weightwin.

Author(s)

Martijn van de Pol and Liam D. Bailey

References

van de Pol & Cockburn 2011 Am Nat 177(5):698-707 (doi: 10.1086/659101) "Identifying the critical climatic time window that affects trait expression"

Examples

```
#Simple test example
#Create data from a subset of our test dataset
biol_data <- Mass[1:5, ]
data(MassClimate)
```

```
weight <- weightwin(xvar = list(Temp = MassClimate$Temp),
                    cdate = MassClimate$Date,
                    bdate = biol_data$Date,
                    baseline = glm(Mass ~ 1, data = biol_data),
                    range = c(100, 0), func = "lin",
                    type = "relative", weightfunc = "W", cinterval = "day",
                    par = c(2.26, 8.45, 0), control = list(ndeps = c(0.01, 0.01, 0.01)),
                    method = "L-BFGS-B")


## Not run:

# Full working example

# Test for a weighted average over a fixed climate window
# using datasets 'Offspring' and 'OffspringClimate'

# N.B. THIS EXAMPLE MAY TAKE A MOMENT TO CONVERGE ON THE BEST MODEL.

# Load data

data(Offspring)
data(OffspringClimate)

# Test for climate windows between 365 and 0 days ago (range = c(365, 0))
# Fit a quadratic term for the mean weighted climate (func="quad")
# in a Poisson regression (offspring number ranges 0-3)
# Test a variable window (type = "absolute")
# Test at the resolution of days (cinterval="day")
# Uses a Weibull weight function (weightfunc="week")

weight <- weightwin(xvar = list(Temp = OffspringClimate$Temperature),
                    cdate = OffspringClimate$Date,
                    bdate = Offspring$Date,
                    baseline = glm(Offspring ~ 1, family = poisson, data = Offspring),
                    range = c(365, 0), func = "quad",
                    type = "relative", weightfunc = "W", cinterval = "day",
                    par = c(3, 0.2, 0), control = list(ndeps = c(0.01, 0.01, 0.01)),
                    method = "L-BFGS-B")

# View output

head(weight[[3]])
summary(weight[[1]])
head(weight[[2]])

## End(Not run)
```

---

wgdev *Calculate within group deviance.*

Description

Calculate within group deviance of a variable. Used to test for 'within individual effects'. See methods outlined in van de Pol and Wright 2009 for more detail.

Usage

```
wgdev(covar, groupvar)
```

Arguments

covar	Continuous variable for which within group deviance will be calculated. Please specify the dataset in which the variable is found (i.e. data$var).
groupvar	Grouping variable within which deviance will be calculated. For example, individual ID or site/plot ID. Please specify the dataset in which the variable is found (i.e. data$var).

Value

Returns a vector containing numeric values. This can be used to differentiate within and between group effects in a model. See function wgmean to calculate within group means. See van de Pol and Wright 2009 for more detail.

Author(s)

Martijn van de Pol and Jonathan Wright

Examples

```
# Calculate within year deviance in temperature from the MassClimate dataset.

data(MassClimate)

#Calculate year column
library(lubridate)
MassClimate$Year <- year(as.Date(MassClimate$Date, format = "%d/%m/%Y"))

#Calculate within year deviance of temperature
within_yr_dev <- wgdev(MassClimate$Temp, MassClimate$Year)

#Add this variable to the original dataset
MassClimate$Within_yr_dev <- within_yr_dev
```

wgmean *Calculate within group means.*

Description

Calculate group means of a variable. Used to test for 'between individual effects'. See methods outlined in van de Pol and Wright 2009 for more detail.

Usage

```
wgmean(covar, groupvar)
```

Arguments

covar	Continuous variable for which group means will be calculated. Please specify the dataset in which the variable is found (i.e. data$var).
groupvar	Grouping variable within which means will be calculated. For example, individual ID or site/plot ID. Please specify the dataset in which the variable is found (i.e. data$var).

Value

Returns a vector containing numeric values. This can be used to differentiate within and between group effects in a model. See function wgdev to calculate within group deviance. See van de Pol and Wright 2009 for more details on the method.

Author(s)

Martijn van de Pol and Jonathan Wright

Examples

```
# Calculate mean temperature within years from the MassClimate dataset.

data(MassClimate)

#Calculate year column
library(lubridate)
MassClimate$Year <- year(as.Date(MassClimate$Date, format = "%d/%m/%Y"))

#Calculate mean temperature within each year
within_yr_mean <- wgmean(MassClimate$Temp, MassClimate$Year)

#Add this variable to the original dataset
MassClimate$Within_yr_mean <- within_yr_mean
```

Index