Package 'papros'

February 20, 2019

Title PAthogene PROgnosis System

Version 1.0

Description
The package can be used to prognose severe infection events with fungi on agricultural plants.

License GPL-3

ByteCompile true

Encoding UTF-8

LazyData true

Imports sp,
raster,
stats,
utils,
graphics,
dplyr,
ggplot2,
magrittr,
R.utils,
RCurl,
gstat,
automap,
C50,
randomForest,
zoo,
animation

RoxygenNote 6.1.1

R topics documented:

`aggregate_interpolate_points` *Interpolate and temporal aggregate DWD values*

Description

This functions interpolates points based on a dataframe containing coordinates, an aim variable, an aim dataset and a parameter such as a date, by which the interpolations are divided

Usage

```
aggregate_interpolate_points(dataframe, coords, epsg, DateTime, infection,
  incubation, aim_variable, outputfile, trans_epsg = FALSE,
  co_variables = FALSE, procedure = c("ked", "ok", "idw"),
  progressbar = TRUE)
```

Arguments

`dataframe`	dataframe containing the aim variable and if "ked" should be applied the covariables
`coords`	vector containing names of the columns containing x and y coordinate values
`epsg`	number of the EPSG code of the "coords" information
`DateTime`	name of the DateTime column
`infection`	duration of the assumed infection
`incubation`	duration of the assumed incubation
`aim_variable`	Character string with the name of the aim variable
`outputfile`	SpatialPointsDataframe, SpatialGridDataFrame or raster which should be filled with predictions; requires covariables for "ked"
`trans_epsg`	default = FALSE, number of the EPSG code the "coords" information should be transformed to
`co_variables`	default = FALSE, vector of covariables if needed

procedure	default = c("ked","ok","idw"); vector containing the interpolation technic to be used; the first method is used and if this does not work out, the second, and so on
progressbar	default = TRUE; should a progressbar be generated?

Value

a dataframe or a SpatialPointsDataFrame containing information about DWD locations in Germany

Author(s)

Wolfgang Hamer

Examples

```
# Download example data
shdat <- download_statewide_hourly_station_data(state = "Schleswig-Holstein", coord = TRUE)

shdat2 <- shdat %>%
    filter(DateTime < sort(unique(shdat$DateTime))[50])

example <- aggregate_interpolate_points(dataframe = shdat2,
                                        coords = c("lon","lat"),
                                        DateTime = "DateTime",
                                        infection=2,
                                        incubation=8,
                                        aim_variable ="Temperature",
                                        outputfile=c(1000,1000),
                                        co_variables = FALSE,
                                        procedure = c("ked","ok","idw"),
                                        epsg = 4326,
                                        trans_epsg = 25832)
plot(example[[1]])
```

---

ctu *Calculate Cumulative Thermal Unit*

Description

This functions calculates cumulative thermal unit using the daily thermal unit (dtu) (Based on https://www.researchgate.net/publication/281674392_Modeling_physiology_of_crop_development_growth_and_yield)

Usage

```
ctu(TMP, TBD = 0, TP1D = 25, TP2D = 28, TCD = 40)
```

Arguments

TMP	temperature in °C (vector or raster file)
TBD	thermal base temperature; default = 0 for wheat
TP1D	lower optimum temperature; default = 25 for wheat
TP2D	upper optimum temperature; default = 28 for wheat
TCD	thermal ceiling temperature; default = 40 for wheat

Value

a vector or raster file (depending on input) with the relative development rate based on temperature

Author(s)

Wolfgang Hamer

Examples

```
ctu(10)
ctu(c(23,12,23))
```

---

download_alltime_hourly_station_data

Download alltime hourly weather data of the DWD stations

Description

Downloads historical and recent hourly data of the DWD stations in Germany

Usage

```
download_alltime_hourly_station_data(station,
  parameter = c("temperature", "precipitation", "windspeed"),
  astbl = FALSE)
```

Arguments

station	the station ID
parameter	one ore multiple paramters of c("temperature", "humidity", "precipitation", "windspeed", "winddirection")
astbl	default = FALSE; should the explort be a dataframe or a tibble

Value

a p value of the comparison between the selected and random points

Author(s)

Wolfgang Hamer

Examples

```
# Select Location
mapview::mapview(get_all_dwd_locations(TRUE))

Fehmarn <- download_alltime_hourly_station_data(5516)
head(Fehmarn)

LeuchtturmKiel <- download_alltime_hourly_station_data(02961, parameter = "windspeed")
head(LeuchtturmKiel)
```

---

download_dwd_raster *Download multi-annual DWD rasters*

Description

This functions download multi-annual DWD rasters and crops them if desired

Usage

```
download_dwd_raster(parameter = "air_temperature_mean", period = "",
  month = "", crop = FALSE, savepath = FALSE)
```

Arguments

parameter	a character string defining the parameter to be downloaded (e.g.: "air_temperature_mean", "drought_index", "evapo_p", "frost_days", "hot_days", "ice_days", "precipitation", "snowcover_days", "soil_moist", "soil_temperature_5cm", "summer_days", "sunshine_duration", "vegetation_begin", "vegetation_end", "water_balance")
period	years which are combined in the mult annual datasets (e.g.: "1961-1990", "1981-2010","1991-2010", "1992-2015")
month	the month which should be downloaded (e.g.: 1,2,3,...,12 or 13 for spring (March, April, May), 14 for summer (June, July, August), ..., or 17 for the whole year)
crop	Spatial Dataset of which an extent can be created which is used to crop the germany wide DWD dataset
savepath	defalut = FALSE; path to folder where files should be stored

Value

a raster dataset

Author(s)

Wolfgang Hamer

`download_hourly_station_data` *Download hourly weather data of the DWD stations*

Description

Downloads hourly data of the DWD stations in Germany

Usage

```
download_hourly_station_data(station, parameter = c("temperature",
  "precipitation", "windspeed"), time = "recent", astbl = FALSE)
```

Arguments

`station`	the station ID
`parameter`	one ore multiple paramters of c("temperature", "humidity", "precipitation", "windspeed", "winddirection")
`time`	either "recent" often updated data or "historical" data which go longer in the past
`astbl`	default = FALSE; should the explort be a dataframe or a tibble

Value

a p value of the comparison between the selected and random points

Author(s)

Wolfgang Hamer

Examples

```
# Select Location
mapview::mapview(get_all_dwd_locations(TRUE))

Fehmarn <- download_hourly_station_data(5516)
head(Fehmarn)

LeuchtturmKiel <- download_hourly_station_data(02961, parameter = "windspeed")
head(LeuchtturmKiel)
```

download_statewide_hourly_station_data

Download hourly weather data of the DWD stations of federal states

Description

Downloads hourly data of the DWD stations in one federal state in Germany

Usage

```
download_statewide_hourly_station_data(state,
  parameter = c("temperature", "precipitation", "windspeed"),
  time = "recent", coord = FALSE, savefile = FALSE)
```

Arguments

state	the Federal State (e.g. "Schleswig-Holstein")
parameter	one ore multiple paramters of c("temperature","humidity","precipitation","windspeed","winddirectio
time	either "recent" often updated data or "historical" data which go longer in the past
coord	default = FALSE; should the explort contain coordinates or not
savefile	default = FALSE; where should the file be saved as .csv file?

Value

a p value of the comparison between the selected and random points

Author(s)

Wolfgang Hamer

Examples

```
shdat <- download_statewide_hourly_station_data(state = "Schleswig-Holstein", coord = TRUE)
head(shdat)
```

dtu

Calculate Daily Thermal Unit

Description

This functions calculates the Daily Thermal Unit (Based on https://www.researchgate.net/publication/281674392_Modeling_physiology_of_crop_development_growth_and_yield)

Usage

```
dtu(TMP, TBD = 0, TP1D = 25, TP2D = 28, TCD = 40)
```

Arguments

TMP	temperature in °C (vector or raster file)
TBD	thermal base temperature; default = 0 for wheat
TP1D	lower optimum temperature; default = 25 for wheat
TP2D	upper optimum temperature; default = 28 for wheat
TCD	thermal ceiling temperature; default = 40 for wheat

Value

a vector or raster file (depending on input) with the relative development rate based on temperature

Author(s)

Wolfgang Hamer

Examples

```
dtu(10)
dtu(c(23,12,23))
```

---

dwd_add_date_time — *Replaces DateTime column by Date (date) and Time (numeric) columns*

Description

This functions replaces DateTime column by Date (date) and Time (numeric) columns

Usage

```
dwd_add_date_time(dataframe, columnname = "DateTime")
```

Arguments

dataframe	a dataframe
columnname	default = "DateTime"; should contain values in the format "2017072602" for 2 oclock at the 26 th of Julya in 2017

Value

a dataframe like dataframe with two new columns

Author(s)

Wolfgang Hamer

Examples

```
locs <- get_dwd_locations(sp = TRUE)
mapview::mapview(locs)
```

get_all_dwd_locations *Download all available DWD location data from the CDC Server*

Description

This functions downloads DWD location data from the CDC Server

Usage

```
get_all_dwd_locations(sp = FALSE)
```

Arguments

sp	default = FALSE; if TRUE returns not the plain data frame but a spatialised version

Value

a dataframe or a SpatialPointsDataFrame containing information about DWD locations in Germany

Author(s)

Wolfgang Hamer

Examples

```
mapview::mapview(get_all_dwd_locations(TRUE))
```

get_dwd_locations *Download DWD location data from the CDC Server*

Description

This functions downloads DWD location data from the CDC Server

Usage

```
get_dwd_locations(sp = FALSE, parameter = "temperature")
```

Arguments

sp	default = FALSE; if TRUE returns not the plain data frame but a spatialised version
parameter	default = "temperature"; should the "temperature" (and humidity), "precipitation" or "wind" station network be downloaded

Value

a dataframe or a SpatialPointsDataFrame containing information about DWD locations in Germany

Author(s)

Wolfgang Hamer

Examples

```
locs <- get_dwd_locations(sp = TRUE)
mapview::mapview(locs)
```

---

interpolate_points *Interpolates point values*

Description

This function tries to interpolate points based on given parameters

Usage

```
interpolate_points(sp_points, aim_variable, outputfile,
  co_variables = FALSE, procedure = c("ked", "ok", "idw"))
```

Arguments

sp_points	SpatialPointsDataframe containing the aim variable and if "ked" should be applied the covariables
aim_variable	Character string with the name of the aim variable
outputfile	SpatialPointsDataframe, SpatialGridDataFrame or raster which should be filled with predictions; requires covariables for "ked" (if two values are given in a vector a raster is created with the resolution given by the values)
co_variables	default = FALSE, vector of covariables if needed
procedure	default = c("ked","ok","idw"); vector containing the interpolation technic to be used; the first method is used and if this does not work out, the second, and so on

Value

a dataframe or a SpatialPointsDataFrame containing information about DWD locations in Germany

Author(s)

Wolfgang Hamer

Examples

```
# Download example data
shdat <- download_statewide_hourly_station_data(state = "Schleswig-Holstein", coord = TRUE)

# Select data of specific Time / Date
da_sel <- shdat %>% filter(DateTime == sort(unique(shdat$DateTime))[5])
```

```
# Create spatial dataset
da_sel_sp <- SpatialPointsDataFrame(da_sel[,c("lon", "lat")],
                                    da_sel,
                                    proj4string = CRS("+init=epsg:4326"))

# Transform to projected (m based!) system
da_sel_sp <- spTransform(da_sel_sp, CRS("+init=epsg:25832"))

# Manually creating points of interest
preds <- SpatialPointsDataFrame(data.frame(x=c(9.5,9.0,10.4,10.5,9.9,10.5),
                                           y=c(53.5, 54.69,54.44,53.93,53.65, 54.55)),
                                data.frame(lat=c(9.5,9.0,10.4,10.5,9.9,10.5),
                                           lon=c(53.5, 54.69,54.44,53.93,53.65, 54.55)),
                                proj4string = CRS("+init=epsg:4326"))
outputpoints <- spTransform(preds, CRS("+init=epsg:25832"))

# Application of function for point data result
myintpoints <- interpolate_points(sp_points = da_sel_sp,
                                  aim_variable = "Temperature",
                                  outputfile = outputpoints,
                                  co_variables = c("lat","lon"),
                                  procedure = c("ked","ok","idw"))

# Create raster of interest
outputraster <- raster(ncol=100, nrow=100)
extent(outputraster) <- extent(outputpoints)
crs(outputraster) <- CRS("+init=epsg:25832")
outputraster[]<- rep(1,length(outputraster$layer[]))

# Application of function for raster data result
myintraster <- interpolate_points(sp_points = da_sel_sp,
                                  aim_variable = "Temperature",
                                  outputfile = outputraster,
                                  co_variables = c("lat","lon"),
                                  procedure = c("ked","ok","idw"))

# Application of function for raster data result
myintraster2 <- interpolate_points(sp_points = da_sel_sp,
                                   aim_variable = "Temperature",
                                   outputfile = c(500,500),
                                   co_variables = c("lat","lon"),
                                   procedure = c("ked","ok","idw"))
```

`large_ctu` *Apply ctu on large dataset*

Description

This functions calculates Cumulative Thermal Units for large datasets using the daily thermal unit (dtu) (Based on https://www.researchgate.net/publication/281674392_Modeling_physiology_of_crop_development_growth_and_yield)

Usage

```
large_ctu(dataset, temp_column, date_column, start_date = "10-01",
  location_column = FALSE, vector = TRUE, TBD = 0, TP1D = 25,
  TP2D = 28, TCD = 40)
```

Arguments

dataset	a dataset
temp_column	name of the temperature column
date_column	name of the date column
start_date	start date of the growing plant; defalut = "10-01" for October the 10th
location_column	name of the location column; defalut = FALSE
vector	default = TRUE; boolean operator defining if a dataset with additional column or only the new column should be given out
TBD	thermal base temperature; default = 0 for wheat
TP1D	lower optimum temperature; default = 25 for wheat
TP2D	upper optimum temperature; default = 28 for wheat
TCD	thermal ceiling temperature; default = 40 for wheat

Value

a vector or raster file (depending on input) with the relative development rate based on temperature

Author(s)

Wolfgang Hamer

Examples

```
# Download example data
shdat <- download_statewide_hourly_station_data(state = "Schleswig-Holstein", coord = TRUE)
shdat2 <- shdat %>% filter(DateTime > 1995093023)

shdat2 %<>% dplyr::mutate(Date = as.Date(substr(DateTime,1,8),"%Y%m%d"))

shdat2 %<>% dplyr::mutate( CTU = large_ctu(dataset = shdat2,
                                           temp_column = "Temperature",
                                           date_column = "Date",
                                           start_date = "10-01",
                                           location_column = "ID"))
```

list_files_in_CDC_folder *List files in CDC FTP folder*

Description

List files in CDC FTP folder

Usage

```
list_files_in_CDC_folder(path)
```

Arguments

path	the path to be explored

Value

a vector with files stored in specific path

Author(s)

Wolfgang Hamer

Examples

```
list_files_in_CDC_folder("ftp://ftp-cdc.dwd.de/pub/CDC/observations_germany/climate/hourly/air_temperature/
```

loocv_machine_learner *Apply loocv machine learning functions*

Description

This functions applies one or several machine learning methods on a given dataset and checks for the validity of the prediction

Usage

```
loocv_machine_learner(dataframe, aim_variable, co_variables,
  location = FALSE, method = c("DT", "BDT", "RF"))
```

Arguments

dataframe	dataframe containing variables of interest
aim_variable	Character string with the name of the aim variable
co_variables	Character string with the name of the co-variables
location	defalut = FALSE; Character string with the name of the location. If FALSE each observation is treated as unique location
method	default = c("DT","BDT","RF"); which method should be used: DecisionTree, BoostedDecisionTree and/or RandomForest?

Value

list containing the models

Author(s)

Wolfgang Hamer

`machine_learner` *Apply machine learning functions*

Description

This functions applies one or several machine learning methods on a given dataset

Usage

```
machine_learner(dataframe, aim_variable, co_variables, method = c("DT",
  "BDT", "RF"))
```

Arguments

`dataframe`	dataframe containing variables of interest
`aim_variable`	Character string with the name of the aim variable
`co_variables`	Character string with the name of the co-variables
`method`	default = c("DT","BDT","RF"); which method should be used: DecisionTree, BoostedDecisionTree and/or RandomForest?

Value

list containing the models

Author(s)

Wolfgang Hamer

`machine_predictor` *Predict by raster stack and machine learning model*

Description

This functions applies one machine learning model on a given rasterstack

Usage

```
machine_predictor(rstack, mmodel, additionalRaster = FALSE,
  type = FALSE, index = FALSE)
```

Arguments

rstack	list containing an raster stack with covariables for prediction based on the mmodel
mmodel	machine learning model
additionalRaster	rasters that are identical in each time step and should be added to each rasterstack
type	character string containing type of prediction (e.g. "prob" for probability); default to FALSE
index	in case of type = "prob" the index of the parameter of which the probability shold be returned

Value

stack with one prediction for each element of the input list

Author(s)

Wolfgang Hamer

---

machine_predictor_lineplot

Lineplot of predict by raster stack and machine learning model

---

Description

This functions applies one machine learning model on a given rasterstack

Usage

```
machine_predictor_lineplot(rstack, location, yname, ylim = c(0, 100),
  rollingaverage = 1, threshold = FALSE, aggregate_x_ticks = 5)
```

Arguments

rstack	list containing an raster stack of predictions as created by ´machine_predictor´. The names of the rasters are expected to be in the format "X20180515" for the date 2018-05-15
location	sp object containing location information
yname	name for the y axis
ylim	default = c(0,100); limits of the y axis
rollingaverage	default = 1; how many points should be averaged for the line
threshold	default = FALSE; numeric which indicates a red threshold line on the y axis
aggregate_x_ticks	default = 5; how many ticks should the x axis have?

Value

plot

Author(s)

Wolfgang Hamer

`multiple_interpolate_points` *Interpolates DWD values*

Description

This functions interpolates points based on a dataframe containing coordinates, an aim variable, an aim dataset and a parameter such as a date, by which the interpolations are divided

Usage

```
multiple_interpolate_points(dataframe, coords, epsg, splitter,
  aim_variable, outputfile, trans_epsg = FALSE, co_variables = FALSE,
  procedure = c("ked", "ok", "idw"), progressbar = TRUE)
```

Arguments

`dataframe`	dataframe containing the aim variable and if "ked" should be applied the covariables
`coords`	vector containing names of the columns containing x and y coordinate values
`epsg`	number of the EPSG code of the "coords" information
`splitter`	name of the splitter column
`aim_variable`	Character string with the name of the aim variable
`outputfile`	SpatialPointsDataframe, SpatialGridDataFrame or raster which should be filled with predictions; requires covariables for "ked"
`trans_epsg`	default = FALSE, number of the EPSG code the "coords" information should be transformed to
`co_variables`	default = FALSE, vector of covariables if needed
`procedure`	default = c("ked","ok","idw"); vector containing the interpolation technic to be used; the first method is used and if this does not work out, the second, and so on
`progressbar`	default = TRUE; should a progressbar be generated?

Value

a dataframe or a SpatialPointsDataFrame containing information about DWD locations in Germany

Author(s)

Wolfgang Hamer

Examples

```
# Download example data
shdat <- download_statewide_hourly_station_data(state = "Schleswig-Holstein", coord = TRUE)

shdat2 <- shdat %>%
    filter(DateTime < sort(unique(shdat$DateTime))[5])
```

```
example <- multiple_interpolate_points(dataframe = shdat2,
                                       coords = c("lon","lat"),
                                       splitter = "DateTime",
                                       aim_variable ="Temperature",
                                       outputfile=c(1000,1000),
                                       co_variables = FALSE,
                                       procedure = c("ked","ok","idw"),
                                       epsg = 4326,
                                       trans_epsg = 25832)
plot(stack(example))
```

`one_year` *Give out one year*

Description

This function gives the sequence of one year following the date given

Usage

```
one_year(fromdate)
```

Arguments

`fromdate`	name of the temperature column

Value

a vector of one year following the date given

Author(s)

Wolfgang Hamer

Examples

```
one_year(as.Date("1996-10-01","%Y-%m-%d"))
```

`read_rasterstacklist` *Reads a list of raster stacks*

Description

This function reads a list of raster stacks as stored by store_rasterstacklist

Usage

```
read_rasterstacklist(pathfolder)
```

Arguments

pathfolder	folder to which the lists rasterstacks should be exported

Author(s)

Wolfgang Hamer

---

reduce_input *Reduce the size of an given dataset*

Description

This functions reduces the size of an given dataset in respect to the weather data of "infection" days relevant for an infestation "incubation" days later

Usage

```
reduce_input(dataframe, DateTime, infection, incubation, event_dates)
```

Arguments

dataframe	dataframe containing variables of interest which should be reduced
DateTime	name of the DateTime column
infection	duration of the assumed infection
incubation	duration of the assumed incubation
event_dates	Character string with the name of the aim variable

Value

reduced dataframe input

Author(s)

Wolfgang Hamer

---

store_rasterstacklist *Stores a list of raster stacks*

Description

This function stores a list of raster stacks

Usage

```
store_rasterstacklist(rstacklist, pathfolder)
```

Arguments

rstacklist	a list of raster stacks
pathfolder	folder to which the lists rasterstacks should be exported

Author(s)

Wolfgang Hamer

---

tempfun *Calculate response of relative development rate to temperature*

Description

This functions calculates response of relative development rate to temperature depending on crop parameters (Based on https://www.researchgate.net/publication/281674392_Modeling_physiology_of_crop_development_growth_and_yield)

Usage

```
tempfun(TMP, TBD = 0, TP1D = 25, TP2D = 28, TCD = 40)
```

Arguments

TMP	temperature in °C (vector or raster file)
TBD	thermal base temperature; default = 0 for wheat
TP1D	lower optimum temperature; default = 25 for wheat
TP2D	upper optimum temperature; default = 28 for wheat
TCD	thermal ceiling temperature; default = 40 for wheat

Value

a vector or raster file (depending on input) with the relative development rate based on temperature

Author(s)

Wolfgang Hamer

Examples

```
tempfun(10)
tempfun(c(23,12,23))
```

`videoplot_rasterstack` *Videoplot predicted raster stack*

Description

This function creates a video of the raster stack

Usage

```
videoplot_rasterstack(rstack, ffmpeg_path, storefile,
  other.opts = "-pix_fmt yuv420p -b 500k -s:v 720x720",
  main = "default", col = colorRampPalette(c("green", "yellow",
  "red"))(8), breaks = c(0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875,
  1), sub = "", cex.axis = 1.3, cex.main = 1.8, cex.sub = 1.6,
  cex.lab = 1.4, legend.width = 2, legend.shrink = 0.8,
  axis.args = list(cex.axis = 1.3))
```

Arguments

`rstack`	list containing an raster stack of predictions as created by ´machine_predictor´. The names of the rasters are expected to be in the format "X20180515" for the date 2018-05-15
`ffmpeg_path`	path of the ´ffmpeg.exe´as available by https://www.ffmpeg.org/download.html
`storefile`	File to which the mp4 file should be stored
`other.opts`	Further options of the ´saveVideo´ function of the animation library
`main`	character string containing the main for the raster plot. default = "default" which creates a date of the raster name as mentioned above
`...`	Further options of the raster plot, such as col, breaks, sub, cex.axis, ...

Value

a video file stored at the specified location

Author(s)

Wolfgang Hamer

Index