Package 'SOfireA'

March 28, 2018

Title Satellite Observations for Fire Activity

Version 1.1

Date 2018-03-28

Author Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

Maintainer Matthias Forkel <matthias.forkel@geo.tuwien.ac.at>

Description SOFIA (Satellite Observations for Fire Activity) is an empirical modelling concept to predict burned area based on satellite and climate data. The package implements the basic SOFIA model structure, and functions to optimize and plot SOFIA models.

Depends R (>= 3.2.3), ModelDataComp, plyr, parallel

Imports rgenoud

License GPL-2

URL

LazyLoad yes

R topics documented:

SOfireA-package *Satellite Observations for Fire Activity*

Description

SOFIA (Satellite Observations for Fire Activity) is an empirical modelling concept to predict burned area based on satellite and climate data. The package implements the basic SOFIA model structure, and functions to optimize and plot SOFIA models.

Details

The DESCRIPTION file: This package was not yet installed at build time.

Index: This package was not yet installed at build time.

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

firedata *Data on burned area, land cover, climate, vegetation, and socioeconomics*

Description

This data.frame contains several datasets on burned area, climate, soil moisture, vegetation, land cover, and socioeconomic variables. The data was extracted for sampled 0.25° grid cells of the globe and has monthly time steps. The data includes the following variables:

- cellid: identifier of grid cells
- lon: longitude
- lat: latitude
- regid: identifier for regions
- time: time (months between 1997 and 2011)
- area: area of grid cells (ha)
- train: indicator if data is used for training (train=1) or evaluation (train=4)
- GFED burned area version 4 (Giglio et al., 2013), http://www.globalfiredata.org
  - GFED.BA.obs: Fractional burned area of a 0.25° grid cell, used for optimization of SOFIA models
  - GFED.BA.unc: uncertainty of burned area
- ESA land cover_cci version 1.6.1, http://maps.elie.ucl.ac.be/CCI/viewer/index.php

  - CCI.LC.Tree.BE: broadleaved evergreen tree
  - CCI.LC.Tree.BD: Broadleaved deciduous trees
  - CCI.LC.Tree.NE: Needle-leaved evergreen trees
  - CCI.LC.Tree.ND: Needle-leaved deciduous trees
  - CCI.LC.Shrub.BE: Broadleaved evergreen shrubs
  - CCI.LC.Shrub.BD: Broadleaved deciduous shrubs
  - CCI.LC.Shrub.NE: Needle-leaved evergreen shrubs
  - CCI.LC.Herb: Natural grass and herbaceous vegetation
  - CCI.LC.Crop: Cropland and managed grass
  - CCI.LC.HrbCrp: Natural and managed grass and croplands = Herb + Crop
  - CCI.LC.Tree: Coverage of trees = Tree.BE + Tree.BD + Tree.NE + Tree.ND
  - CCI.LC.Shrub: Coverage of shrubs = Shrub.BE + Shrub.BD + Shrub.NE
  - CCI.LC.Broadleaf: Coverage of broadleaved vegetation = Tree.BE + Tree.BD + Shrub.BE + Shrub.BD
  - CCI.LC.Needleleaf: Coverage of needle-leaved vegetation = Tree.NE + Tree.ND + Shrub.NE
- CRU TS3.23 climate data (Harris et al., 2014), https://crudata.uea.ac.uk/cru/data/hrg/cru_ts_3.23
  - CRU.T.orig: Mean monthly air temperature (degC)
  - CRU.T.annual: Mean air temperature in the actual month and the 12 months before a fire
  - CRU.WET.orig: Monthly number of wet day
  - CRU.WET.annual: Mean number of wet days in the actual month and the 12 months before a fire
  - CRU.DTR.orig: Mean monthly diurnal temperature range (K)
- GPCC precipitation version 7, http://dx.doi.org/10.5676/DWD_GPCC/FD_M_V7_050
  - GPCC.P.orig: Monthly total precipitation (mm)
  - GPCC.P.annual: Total precipitation in the actual month and the 12 months before a fire
- ESA soil moisture_cci version 2.3 (Dorigo et al.) , http://cci.esa.int/data
  - CCI.SM.orig Mean monthly surface soil moisture
  - CCI.SM.annual Mean monthly surface soil moisture
- GIMMS fraction of absorbed photosynthetic active radiation version 3g (Zhu et al., 2013), http://cliveg.bu.edu/modismisr/lai3g-fpar3g.html
  - GIMMS.FAPAR.orig: Mean monthly FAPAR
  - GIMMS.FAPAR.pre: FAPAR in the month before a fire
  - GIMMS.FAPAR.annual: Mean FAPAR in the 12 months before a fire
- Multi-sensor harmonized vegetation optical depth (Liu et al., 2011b, 2015)
  - Liu.VOD.orig: Mean monthly VOD
  - Liu.VOD.pre: VOD in the month before a fire
  - Liu.VOD.annual: Mean VOD in the 12 months before a fire
- GRUMP population density version 1 (years 1990, 1995, 2000) (Balk et al., 2006), http://dx.doi.org/10.7927/H4R20
  - PD.med: Population density (individuals km-2), median estimate of three methods for temporal inter- and extrapolation (spline interpolation, linear interpolation, interpolation with last value as constant)
- Night light development index (year 2006) (Elvidge et al., 2012), http://ngdc.noaa.gov/eog/dmsp/download_nldi.htr
  - NLDI: Night light development index, but grid cells without night lights or population set to 1.01

Format

A `data.frame`.

References

Forkel, M., et al. (2016): in prep.

Examples

```
require(ModelDataComp)

data(firedata)
colnames(firedata)
dim(firedata)

# spatial distribution of the data
ScatterPlot(firedata$lon, firedata$lat, fit.global=FALSE, plot.type="points")

# latitudinal pattern of burned area
fit <- ScatterPlot(firedata$lat, firedata$GFED.BA.obs)
```

`FitSofia` *Fit a Sofia model to a data set*

Description

The function fits a SOFIA model to a dataset.

Usage

```
FitSofia(x, y, unc = NULL, per.group = rep(FALSE, ncol(x)), nodes = 4,
    sofiapar, restart = 0, cost = NULL, ...)
```

Arguments

`x`	data.frame with independent variables
`y`	dependent variable (observation)
`unc`	uncertainty of dependent variable
`per.group`	a boolean vector that indicates if a column in x acts per group (e.g. PFTs)
`nodes`	number of nodes for parallel compuation during genetic optimization
`sofiapar`	SofiaPar object with prior parameters
`restart`	restart previous Sofia optimization? 0 = start new, 1 = continue with previous, 2 = do post-processing
`cost`	cost function to be used
`...`	further arguments

Details

No details.

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

References

No reference.

See Also

`Sofia`, `SofiaOpt`

`MakeFig` *fig positions for graphics*

Description

Calculates positions for figure that consist of multiple panels

Usage

```
MakeFig(nfig, border = c(0, 1, 0, 1), nrow = NULL, ncol = NULL)
```

Arguments

`nfig`	number of figures
`border`	relative graphic borders in which the figures should be placed
`nrow`	number of rows to arrange the figures
`ncol`	number of cols to arrange the figures

Value

A list with positions for each figure and number of rows and columns

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

`plot.Sofia` *plot a Sofia object*

Description

Plots a `Sofia` object.

Usage

```
## S3 method for class 'Sofia'
plot(x, ylab = "y", mfrow = NULL, names = NULL, main = NULL,
    plot.order = NULL, labels = paste0("(", letters, ")"), ...)
```

Arguments

`x`	a 'Sofia' object
`ylab`	label for response variable
`mfrow`	number of rows and columns for the plot
`names`	names of the variables in the response functions
`main`	title of the plot
`plot.order`	Order for plotting of factors
`labels`	Labels for subplots. Set to NULL to avoid labels.
`...`	further arguments (not used)

Details

No details.

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

References

No reference.

See Also

`Sofia`

Examples

```
# get data
data(firedata)

# predictor variables
train <- firedata$train == 1 # use training data
xvars.df <- data.frame(
  NLDI = firedata$NLDI[train],
  CRU.WET.orig = firedata$CRU.WET.orig[train],
  Liu.VOD.annual = firedata$Liu.VOD.annual[train],
  GIMMS.FAPAR.pre = firedata$GIMMS.FAPAR.pre[train],
  CRU.DTR.orig = firedata$CRU.DTR.orig[train]
)

# observed data
obs <- firedata$GFED.BA.obs[train]
regid <- firedata$regid[train]

# Which x variable should depend on land cover?
per.group <- c(FALSE, TRUE, TRUE, TRUE, TRUE)

# land cover
area <- data.frame(
  Tree = firedata$CCI.LC.Tree[train],
  Shrub = firedata$CCI.LC.Shrub[train],
  HrbCrp = firedata$CCI.LC.HrbCrp[train]
)

# define parameters (from Forkel et al. 2016, Fig. 1)
sofiapar <- SofiaPar(colnames(xvars.df), colnames(area), per.group=per.group,
                     par.act=c(1.9, 0, 780, 1, # for PopDens
                               0.3, 1.1, -5.3, 0, 0, 0, 8.9, 0.54, -23, -39, 13, -16, # f
                               0.13, 3, 0.53, 0, 0, 0, 0.35, -0.44, 0.36, -1.2, -4.8, -45
                               -0.7, 18, -1.5, 0, 0, 0, 22, 11, -17, -2.3, 0.64, 1,  # fo
                               1.9, 3, -0.36, 0, 0, 0, -21, 68, -38, 0.35, 0.31, 0.11) #
)

# run model
sf <- Sofia(xvars.df, area, per.group=per.group, sofiapar=sofiapar)
plot(sf)
```

`plot.SofiaOpt` *Plot SOFIA optimization results*

Description

Plots the devlopment of the cost and of other performance metrics during the optimization of a SOFIA model. `SofiaOpt` produces files that can be used to restart or monitor an optimization

experiment. These files can be read with `ReadSofiaOpt` and plotted with this function..

Usage

```
## S3 method for class 'SofiaOpt'
plot(x, plot.objfct = c("IoA", "FV", "MEF"), ...)
```

Arguments

`x`	an object of class `SofiaOpt` as returned by `ReadSofiaOpt`
`plot.objfct`	which objective function should be plotted (maximum 3)?
`...`	further arguments (not used)

Details

No details.

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

References

No reference.

See Also

`SofiaOpt`

`predict.Sofia` *Predict values based on a 'Sofia' object*

Description

Make a predicition based on a `Sofia` object and newdata

Usage

```
## S3 method for class 'Sofia'
predict(object, newdata, return.all = FALSE, ...)
```

Arguments

`object`	an object of class 'Sofia', see `Sofia`
`newdata`	a data frame with columns names as in object$group.names for area fractions of groups and as in object$x.names for explantory variables
`return.all`	return all Sofia results? If FALSE, returns only total burned area
`...`	further arguments (not used)

Details

No details.

Value

A vector with predicted values.

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

References

No reference.

See Also

`Sofia`, `SofiaOpt`

`ReadSofiaOpt` *Read results from an SOFIA optimization experiment*

Description

The optimization within `SofiaOpt` produces files that can be used to restart or monitor an optimization experiment. This function reads these files.

Usage

```
ReadSofiaOpt(files, combine = TRUE, ...)
```

Arguments

`files`	vector of file names
`combine`	combine several files in a single file?
`...`	further arguments (not used)

Details

No details.

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

References

No reference.

See Also

SofiaOpt

Sofia *Satellite Observations for Fire Activity*

Description

SOFIA (Satellite Observations for Fire Activity) is an empirical modelling concept to predict burned area based on satellite and climate data. Thereby several logistic functions are multiplicatively combined.

Usage

```
Sofia(x, area = rep(1, nrow(x)), per.group = rep(FALSE, ncol(x)),
    sofiapar = NULL, par = NULL, return.all = TRUE, ...)
```

Arguments

x	data.frame with independent variables
area	a vector or data.frame/matrix with fractional coverage of grid cell area. If 'area' is a vector, it represents the maximal fractional burned area of a grid cell (e.g. the maximum vegetated area). If 'area' is a data.frame or matrix, it represents fractional coverage of groups (e.g. PFTs). Columns should represent groups and rows should be observations (grid cells and time steps).
per.group	a boolean vector that indicates if a column in x acts per group (e.g. PFTs)
sofiapar	object of class SofiaPar which is used for the fit. If NULL, the argument 'par' is used to create sofiapar using the function SofiaPar
par	vector of parameters of logistic functions. If NULL, default parameters are used (that are usually physically not plausible)
return.all	return all input and results? The function returns an object of class 'Sofia'. If TRUE, this object includes in the 'data' slot the fitted values, the fits per group, the response functions, the inputs 'x' and 'area'. If FALSE, only the fitted values are included.
...	further arguments

Details

No details.

Value

an object of class 'Sofia' which is actually a list.

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

References

No reference.

See Also

`SofiaOpt`, `SofiaLogistic`

Examples

```
# example based on aritifical data
#---------------------------------

# explanatory variables
sm <- 1:100
temp <- rnorm(100, 12, 10)
x <- cbind(sm, temp)

# fractional coverage of groups, e.g. plant functional types
tree <- runif(100, 0, 0.8)
grass <- 1 - tree
area <- cbind(tree, grass)

# calculate Sofia with some dummy parameters:
sf <- Sofia(x, area, per.group=c(TRUE, FALSE))
sf$eq
summary(sf$data)
plot(sf)


# example based on real data
#---------------------------

# get data
data(firedata)

# predictor variables
train <- firedata$train == 1 # use training data
xvars.df <- data.frame(
   NLDI = firedata$NLDI[train],
   CRU.WET.orig = firedata$CRU.WET.orig[train],
   Liu.VOD.annual = firedata$Liu.VOD.annual[train],
   GIMMS.FAPAR.pre = firedata$GIMMS.FAPAR.pre[train],
   CRU.DTR.orig = firedata$CRU.DTR.orig[train]
)

# observed data
```

```
obs <- firedata$GFED.BA.obs[train]
regid <- firedata$regid[train]

# Which x variable should depend on land cover?
per.group <- c(FALSE, TRUE, TRUE, TRUE, TRUE)

# land cover
area <- data.frame(
   Tree = firedata$CCI.LC.Tree[train],
   Shrub = firedata$CCI.LC.Shrub[train],
   HrbCrp = firedata$CCI.LC.HrbCrp[train]
)

# define parameters (from Forkel et al. 2016, Fig. 1)
sofiapar <- SofiaPar(colnames(xvars.df), colnames(area), per.group=per.group,
   par.act=c(1.9, 0, 780, 1, # for NLDI
   0.3, 1.1, -5.3, 0, 0, 0, 8.9, 0.54, -23, -39, 13, -16, # for CRU.DTR
   0.13, 3, 0.53, 0, 0, 0, 0.35, -0.44, 0.36, -1.2, -4.8, -45, # for CRU.WET
   -0.7, 18, -1.5, 0, 0, 0, 22, 11, -17, -2.3, 0.64, 1,  # for GIMMS.FAPAR
   1.9, 3, -0.36, 0, 0, 0, -21, 68, -38, 0.35, 0.31, 0.11) # for Liu.VOD
   )

# run model
sf <- Sofia(xvars.df, area, per.group=per.group, sofiapar=sofiapar)
plot(sf)
```

`SofiaLogistic` *Logistic function*

Description

Compute values of a logistic function as used in Sofia models.

Usage

```
SofiaLogistic(par, x, ...)
```

Arguments

`par`	parameters of logistic function, a vector of length 3 (upper asymptote, slope, turning point) or length 4 (upper asymptote, slope, turning point, lower asymptote)
`x`	independent variable
`...`	further arguments (not used)

Details

No details.

Value

a vector

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

References

No reference.

See Also

`Sofia`

Examples

```
x <- -20:20
par <- c(1, 0.5, 0)
plot(x, SofiaLogistic(par, x), type="l")

par <- c(1, 0.2, 0)
plot(x, SofiaLogistic(par, x), type="l")

par <- c(10, -1, 0)
plot(x, SofiaLogistic(par, x), type="l")

# define also the lower asymptote as forth parameter
par <- c(1, 0.5, 0, 0.1)
plot(x, SofiaLogistic(par, x), type="l")

par <- c(1, 0.5, 0, -1)
plot(x, SofiaLogistic(par, x), type="l")
```

`SofiaOpt` *Optimize a SOFIA model using genetic optimization*

Description

The function fits a SOFIA model to observations by estimating model parameters with genetic optimization.

Usage

```
SofiaOpt(x, area = rep(1, nrow(x)), per.group = rep(FALSE, ncol(x)),
    sofiapar = NULL, par.init = NULL, obs, unc = NULL, cost = NULL,
    pop.size = 500, max.generations = 30, path = NULL, restart = 0,
    nodes = 5, BFGS = TRUE, BFGSburnin = max.generations - 2,
    ...)
```

Arguments

x	data.frame with independent variables
area	a vector or data.frame/matrix with fractional coverage of grid cell area. If 'area' is a vector, it represents the maximal fractional burned area of a grid cell (e.g. the maximum vegetated area). If 'area' is a data.frame or matrix, it represents fractional coverage of groups (e.g. PFTs). Columns should represent groups and rows should be observations (grid cells and time steps).
per.group	a boolean vector that indicates if a column in x acts per group (e.g. PFTs)
sofiapar	object of class SofiaPar which is used for the fit. If NULL, the argument 'par.init' is used to create sofiapar using the function SofiaPar
par.init	matrix of inital parameters for optimization. If NULL, inital parameter sets will be created randomly based on the parameter ranges in SofiaPar.
obs	a vector of observed values
unc	vector of observation uncertainties, if NULL an uncertainty of 1 is is used for all observations
cost	a function to compute the cost. If NULL, the SSE (sum of squared error) is used.
pop.size	population size, see genoud
max.generations	maximum number of generations, see genoud
path	directory for optimization results
restart	restart: 0 = start with new optimization, 1 = start with best individuals from previous optimization in 'path', 2 = return results
nodes	how many nodes to use for parallel executaion of genoud?
BFGS	Use the L-BFGS-B algorithm? Overrides BFGSburnin if FALSE.
BFGSburnin	The number of generations before the L-BFGS-B algorithm is first used, see genoud
...	further arguments to genoud

Details

No details.

Value

an object of class 'Sofia' which is actually a list.

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

References

No reference.

See Also

Sofia

Examples

```
# example based on artificial data
#---------------------------------

# some example data
n <- 500
sm <- runif(n, 0, 100) # soil moisture
temp <- rnorm(n, 12, 10) # temperature
tree <- runif(n, 0, 1) # fractional tree cover
grass <- 1 - tree # fractional grass cover
area <- cbind(tree, grass)
x <- cbind(sm, temp)

# create 'observations'
sofiapar <- SofiaPar(colnames(x), colnames(area), per.group=c(TRUE, FALSE))
sofiapar$par <- c(1, 0, 1, 20, 2, 1, 0, 0, -0.2, -0.1, 13, 10) # actual parameters
cbind(sofiapar$name, sofiapar$par)
sf <- Sofia(x, area, per.group=c(TRUE, FALSE), sofiapar=sofiapar)
plot(sf) # fitted values vs. temperature
obs <- sf$data$y # 'observations'

# re-estimate parameters
path <- paste0(here::here(), "/SofiaOpt_test1") # directory for optimization outputs
par.init <- sofiapar$par * 1.5 # some inital parameters for optimization
sfbest <- SofiaOpt(x, area, per.group=c(TRUE, FALSE), obs=obs, sofiapar=sofiapar,
 par.init=par.init, pop.size=10, max.generations=10, BFGS=FALSE, path=path, nodes=1)
str(sfbest)
plot(sfbest)

# plot iterations of optimization
files <- list.files(pattern="SofiaOpt")
fit <- ReadSofiaOpt(files)
plot(fit)
plot(fit, plot.objfct = c("Cor", "Pbias", "RMSE"))

# compare retrieved vs. real
sim <- sfbest$data$y
ScatterPlot(obs, sim, objfct=TRUE)
ObjFct(sim, obs)

# compare real and retrieved response functions
plot(sf$data$x.temp, sf$data$f.temp)
points(sfbest$data$x.temp, sfbest$data$f.temp, col="red")

plot(sf$data$x.sm, sf$data$f.sm.tree)
points(sfbest$data$x.sm, sfbest$data$f.sm.tree, col="red")
```

```
plot(sf$data$x.sm, sf$data$f.sm.grass)
points(sfbest$data$x.sm, sfbest$data$f.sm.grass, col="red")


# example based on real data
# This example is commented because it needs some time.
#-------------------------------------------------------
#
# data(firedata)
#
# # use only training subset
# train <- firedata$train == 1
#
# # predictor variables
# xvars.df <- data.frame(
#   GFED.BA.obs = firedata$GFED.BA.obs[train],
#   Tree = firedata$CCI.LC.Tree[train],
#   Shrub = firedata$CCI.LC.Shrub[train],
#   HrbCrp = firedata$CCI.LC.HrbCrp[train],
#   NLDI = firedata$NLDI[train],
#   CRU.T.orig = firedata$CRU.T.orig[train],
#   CRU.WET.orig = firedata$CRU.WET.orig[train],
#   Liu.VOD.annual = firedata$Liu.VOD.annual[train]
# )
# xvars.df <- na.omit(xvars.df)
# obs <- xvars.df$GFED.BA.obs
# area <- xvars.df[,3:5] # land cover fractions
# xvars.df <- xvars.df[,-(1:4)]
#
# # Which x variable should depend on land cover?
# per.group <- c(FALSE, TRUE, TRUE, TRUE)
#
# # create parameters - with dummy prior parameter values
# sofiapar <- SofiaPar(colnames(xvars.df), colnames(area), per.group=per.group)
# sofiapar
#
# # run prior model
# sf <- Sofia(xvars.df, area, per.group=per.group, sofiapar=sofiapar)
# plot(sf)
#
# # optimize model
# # Note that pop.size should be higher for real applications
# path <- paste0(here::here(), "/SofiaOpt_test2") # directory for optimization outputs
# par.init <- sofiapar$par.act # some inital parameters for optimization
# sfbest <- SofiaOpt(xvars.df, area, per.group=per.group, obs=obs, sofiapar=sofiapar, res
#                    path=path, par.init=par.init, pop.size=10, max.generations=10, BFGS=
# plot(sfbest)
# ScatterPlot(obs, sfbest$data$y, objfct=TRUE)
```

`SofiaPar` *Parameters for SOFIA models*

Description

The function creates an object of class 'SofiaPar' (which is actually a list) which contains information about Sofia model parameters.

Usage

```
SofiaPar(x.names, per.group = rep(FALSE, length(x.names)), group.names = NULL,
    par.act = NULL, par.prior = NULL, par.lower = NULL, par.upper = NULL,
    par.priorsd = NULL, par.optim = NULL, ...)
```

Arguments

`x.names`	names of independent variables
`per.group`	a boolean vector that indicates if a column in x acts per group (e.g. PFTs)
`group.names`	names of groups
`par.act`	
`par.prior`	prior parameters
`par.lower`	lower parameter limits
`par.upper`	upper parameter limits
`par.priorsd`	uncertainty of prior parameters
`par.optim`	(boolean) parameters that shoud be included in optimization
`...`	further arguments

Details

No details.

Value

An object of class 'SofiaPar', which is actually a list.

Author(s)

Matthias Forkel <matthias.forkel@geo.tuwien.ac.at> [aut, cre]

References

No reference.

See Also

`Sofia`, `SofiaLogistic`

Examples

```
# explanatory variables
sm <- 1:100
temp <- rnorm(100, 12, 10)
x <- cbind(sm, temp)

# fractional coverage of groups, e.g. plant functional types
tree <- runif(100, 0, 0.8)
grass <- 1 - tree
area <- cbind(tree, grass)

# parameters for SOFIA models
par <- SofiaPar(colnames(x), per.group=c(TRUE, FALSE), group.names=c("tree", "grass"))
par
```