Package: Recocrop (via r-universe)

October 27, 2026

Type Package

Title Estimating Environmental Suitability for Plants

Description The ecocrop model estimates environmental suitability for plants using a limiting factor approach for plant growth following Hackett (1991) <doi:10.1007/BF00045728>. The implementation in this package is fast and flexible: it allows for the use of any (environmental) predictor variable. Predictors can be either static (for example, soil pH) or dynamic (for example, monthly precipitation).

Version 0.4-0

Date 2021-05-04

LinkingTo Rcpp

SystemRequirements C++11

Imports meteor, terra (>= 1.1-4), methods (>= 0.2-2), Rcpp (>= 0.12.4)

Depends R (>= 3.5.0)

License GPL (>=3)

BugReports https://github.com/cropmodels/Recocrop/issues

URL https://github.com/cropmodels/Recocrop/

Config/pak/sysreqs libgdal-dev gdal-bin libgeos-dev libproj-dev libsqlite3-dev

Repository https://cropmodels.r-universe.dev

Date/Publication 2026-08-01 12:32:34 UTC

RemoteUrl https://github.com/cropmodels/Recocrop

RemoteRef HEAD

RemoteSha 118a558f90767783f71ab2a354b21394f0126fc4

Contents

Recocrop-package 2
crop 3

ecocrop_model 3
ecocropPars 5
plot-ecocrop 7
predict 8
predictors 9
removeParameters 11

Index **12**

`Recocrop-package` *Estimating Environmental Suitability for Plants*

Description

Recocrop implements the ecocrop model. This is a simple limiting-factor based model of plant adaptation to environmental conditions. Plateau-shaped curves are used to quantify plant response (expressed as 0-1) to the environment. The environment can be represented by "dynamic" variables that vary with time, such as precipitation and temperature and "static" variables such as soil properties. The response is zero below a minimum and above a maximum threshold, and one between a minimum and maximum optimal value, and linearly interpolated in between these thresholds. Overall response across environmental variables follows the Sprengel-Liebig Law of the Minimum (Van der Ploeg et al., 1999), that is, the most limiting factor is used, and interactions are not considered.

The dynamic variables must have a monthly time-step. The values are interpolated to go from 12 to 24 observations per year, and model scores are calculated for seasons starting in any of the 24 time-periods. See the `predictors` help page for details.

The *model* with this name was first implemented in DIVA-GIS (Hijmans et al., 2005) and later in the R package 'dismo' and elsewhere. The modeling approach is based on the 'Plantgro' approach (Hackett, 1991).

Default *parameters* used to be available from the UN FAO.

References

Hackett, C, 1991. Mobilising environmental information about lesser-known plants: the value of two neglected levels of description. Agroforestery Systems 14: 131-143. https://doi.org/10.1007/BF00045728

Hijmans RJ., L. Guarino, C Bussink, P Mathur, M Cruz, I Barrentes, E Rojas, 2005. DIVA-GIS. A geographic information system for the analysis of species distribution data. https://diva-gis.org

Van der Ploeg, RR, W Bohm, MB Kirkham, 1999. On the origin of the theory of mineral nutrition of plants and the law of the minimum. Soil Science Society of America Journal 63: 1055-1062

crop *Set crop paramters*

Description

Set crop parameters to a ecocrop model.

Usage

```
crop(x) <- value
```

Arguments

x	EcocropModel object
value	matrix with crop parameters. Each column represents parameters for an environmental variable, and must have a name. The matrix must have four rows that represent the x-coordinates for a table-mountain shaped relative response function (y-coordinates: 0,1,1,0; see the plot in the examples). The only exception is the "duration" parameter with has a single value (in the first row). For functions where only "one tail" is relevant, the values for the other tail can be set to very high or low values, including Inf and -Inf. See ecocropPars for default parameters for many species.

Value

none

Examples

```
x <- ecocropPars("potato")
potato <- x$parameters

m <- ecocrop()
crop(m) <- potato

plot(m)
```

ecocrop_model *EcoCrop model*

Description

Create and run an EcoCrop model to asses the environmental suitability of a location for a (plant) species.

First create a model object with the `ecocrop` method. Then set parameters describing the environmental requirements of a species or other taxon. The `ecocropPars` method provides default parameters for 1710 taxa.

Next, provide environmental data with the `staticPredictors` and/or `dynamicPredictors` method. Static predictors, such as soil pH, do not change throughout the year. In contrast, dynamic predictors, such as temperature and rainfall, vary over time. In the current implementation the time-step of the input data is months. Therefore, dynamic variables must have 12 values, one for much month of the year, or multiples of 12 values, to represent multiple years or locations. The computations are done in half-month time steps, by interpolating the monthly values.

The names of the predictors much match the names in the parameters, but not vice versa. That is, parameters that are not matched by a predictor are ignored.

The main purpose of implementing the model is to support making spatial predictions with `predict`.

Usage

```
ecocrop(crop)
## S4 method for signature 'Rcpp_EcocropModel'
control(x, get_max=FALSE, which_max=FALSE, count_max=FALSE, lim_fact=FALSE, ...)
## S4 method for signature 'Rcpp_EcocropModel'
run(x, ...)
```

Arguments

`crop`	list with ecocrop parameters. See `link[ecocropPars]` and `link[crop]`
`x`	EcocropModel object
`get_max`	logical. If `TRUE`, the maximum value (across the time periods of the year) is returned.
`which_max`	logical. If `TRUE`, the first month with the maximum value is returned.
`count_max`	logical. If `TRUE`, the number of months with the maximum value is returned.
`lim_fact`	logical. If `TRUE`, the options above are ignored, the most-limiting factor for each time period (or the one that is reached first if there are ties) is returned.
`...`	additional arguments. None implemented

Details

The model computes a score for each variable for the 1st and 15th day of each month. It then takes the lowest (most limiting) score for each time period. After that, the minimum score for the time periods that follow (the growing season) is computed. The lenght of the growing season is by the `duration` parameter (see `ecocropPars`).

You can set the output variables with `options`. If all options are `FALSE`, the 24 bi-monthly scores are returned.

Value

vector

Examples

```
# Get parameters
potato <- ecocropPars("potato")

# create a model
m <- ecocrop(potato)

# add parameters
crop(m) <- cbind(clay=c(0,0,10,20))

# inspect
plot(m)

# add predictors
dp <- cbind(tavg=c(10,12,14,16,18,20,22,20,18,16,14,12), prec=seq(50,182,12))
t(dp)
dynamicPredictors(m) <- dp

staticPredictors(m) <- cbind(clay=12)

# run model
x <- run(m)
x

y <- matrix(round(x, 1), nrow=2)
colnames(y) <- month.abb
rownames(y) <- c("day1", "day15")
y

dates <- as.Date(paste0("2000-", rep(1:12, each=2), "-", rep(c(1,15), 12)))
plot(dates, x, las=1, ylab="suitability", xlab="")
lines(dates, x, col="red")


control(m, get_max=TRUE)
run(m)
```

---

ecocropPars *Crop parameters*

Description

Default ecocrop parameters for 1710 taxa. The values are derived from values that used to be available on the website of the Food and Agricultural Organization (FAO) of the United Nations.

User beware: These parameters are expert opinion and not necessarily optimal.

Usage

```
ecocropPars(name, ...)
```

Arguments

name	character. Common or scientific name of a plant (typically a crop plant). If missing, a data.frame with available names is returned
...	additional arguments. None implemented

Details

The parameters returned are returned in a matrix of four rows.

duration: this is the only parameter that is represented by a single number, the length of the growing season in days. This number has to be in the first row of the matrix. The other rows are ignored. The ecocrop model is very sensitive to this parameter. This number is computed by adding one month to the minimum length of the growing season (GMIN in the FAO database), unless the maximum length (GMAX) was the same as GMIN (for example in the case of a 12 month growing season). GMIN and GMAX are provided, for reference, in row 3 and 4. duration was computed this way, because the difference between GMIN and GMAX was sometimes very large, and because it is the shorter growing season that is more relevant for determining suitability. (If a long season is possible, a short season should also be possible, but not vice versa). For example, for maize, GMIN is 65 (extremely short) and GMAX is 365 (extremely long). The returned value is 3 (months). That is still short, and 4 months may be more realistic.

prec: The minimum, lower optimum, higher optimum, and maximum threshold for monthly precipitation. The FAO database has these numbers for annual precipitation. The monthly numbers were derived by dividing the annual precipitation by the (growth duration (in months, see below) plus one).

tavg: The minimum, lower optimum, higher optimum, and maximum threshold for average temperature.

ktmp: The minimum and lower optimum threshold for extreme minimum temperature (killing temperature). Only these two parameters are relevant (the temperature below which the plant dies, and above which there is no frost damage.) The were estimated from KTMP in the FAO database, as KTMP-1 and KTMP+1. The higher optimum, and maximum temperature are set to Inf.

If you wanted to consider damage from high temperatures, you would need to create a new parameter. These cannot be combined, as ktmp requires extreme minimum temperature data, whereas for heat damage you would need extreme maximum temperatures.

ph: The minimum, lower optimum, higher optimum, and maximum threshold for soil pH (presumably ph(H2O), that is, measured in water).

Value

list

See Also

crop

Examples

```
potato <- ecocropPars("potato")
str(potato)

ecocropPars("patato")

p <- ecocropPars("Solanum tuberosum L.")
p

pp <- ecocropPars()
head(pp)

# to see the entire database, including parameters
# that are not automatically extracted do
fname <- system.file("parameters/ecocrop.rds", package="Recocrop")
d <- readRDS(fname)
head(d)
```

`plot-ecocrop` *Ecocrop plot*

Description

Plot the parameters of an ecocrop model

Usage

```
## S4 method for signature 'Rcpp_EcocropModel,ANY'
plot(x, nr, nc, col="red", ...)
```

Arguments

`x`	EcocropModel object
`nr`	number of rows in the plot
`nc`	number of columns in the plot
`col`	color of the lines
`...`	additional arguments. None implemented

Value

none

Examples

```
potato <- ecocropPars("potato")
m <- ecocrop(potato)
plot(m)
```

predict *Spatial EcoCrop model predictions*

Description

Make spatial predictions with an EcoCrop model. First create a model, then provide that to the predict methods, together with SpatRaster objects as predictors.

Usage

```
## S4 method for signature 'Rcpp_EcocropModel'
predict(object, ..., filename="", overwrite=FALSE, wopt=list())
```

Arguments

object	EcocropModel
...	SpatRaster objects with environmental data
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster

Value

SpatRaster

Examples

```
## Predictors
library(terra)
fta <- system.file("ex/ta.tif", package="Recocrop")
fpr <- system.file("ex/pr.tif", package="Recocrop")
fph <- system.file("ex/ph.tif", package="Recocrop")

# monthly average temperature
ta <- rast(fta)
# monthly precipitation
pr <- rast(fpr)
# pH
ph <- rast(fph)

# just for plotting
preds <- c(mean(ta), sum(pr), ph)
names(preds) <- c("tavg", "prec", "ph")
plot(preds)


## make ecocrop model
crop <- ecocropPars("maize")
```

```
m <- ecocrop(crop)
control(m, get_max=TRUE)

# run the model, make sure to match
# the predictor and parameter names
mz <- predict(m, tavg=ta, prec=pr, ph=ph, wopt=list(names="maize"))
plot(mz)

# output by 15 day period
control(m, get_max=FALSE)
mz12 <- predict(m, tavg=ta, prec=pr, ph=ph)
plot(mz12[[14:15]])

control(m, lim_fact=TRUE)
mzlim <- predict(m, tavg=ta, prec=pr+50, ph=ph)
plot(mzlim[[14]])

control(m, get_max=TRUE)
# make the soil more acidic
ph2 <- ph - 1.25
control(m, get_max=TRUE)
mzph <- predict(m, tavg=ta, prec=pr, ph=ph2)
plot(mzph)

# more rainfall
control(m, get_max=TRUE)
pr2 <- pr + 30 # for each month
mzpr <- predict(m, tavg=ta, prec=pr2, ph=ph)
plot(mzpr)

s <- c(mz, mzph, mzpr)
names(s) <- c("base", "ph", "prec")
plot(s)

## another crop
crop <- ecocropPars("pearl millet")
m <- ecocrop(crop)
control(m, get_max=TRUE)
pm <- predict(m, prec=pr, tavg=ta, wopt=list(names="Pearl millet"))

crops <- c(mz, pm)
plot(crops)
```

`predictors` *Environmental predictors*

Description

Set environmental predictors to an ecocrop model. Use `staticPredictors` to set values for "static predictors", such as soil pH, that do not change throughout the year.

Use dynamicPredictors to set dynamic predictors, such as temperature and rainfall, that vary throughout the year. The time step is months. Therefore, dynamic variables normally must have 12 values, one for much month of the year, or multiples of 12 values, to represent multiple years or locations. In this case the values are condsidered to represent climate, and time to be circular: that is, January follows December. If you want to supply weather data (monthly data for a particular year, you have to provide values for that year, and the next year — that is, 24 values (and set parameter nyears to 2.

The 12 dynamic predictor values are interpolated to create 24 time periods for each year, centered around the 1st and 15 day (14th for February) of the month. For example, if the temperature in January is 10, and in Febrary it is 20, the value for January 15 is 10, for Feburary 1 (10+20)/2 = 15, and for February 15 it is 20. This approach works for average values, but not for totals. Precipitation (rainfall) is typically expressed as monthly totals; and if the numbers above were rainfall, the value for January 15 would be 10/2=5, for February 1 it would be (10+20)/4 = 7.5, and for March 15 it would be 20/2 = 10.

It is thus important to recognize variables like this. The model keeps track of that based on the variable name. If the variable is "prec" or "rain" it is assumed to be a total, and otherwise it assumed to be an average.

The names of the predictors much match the names in the parameters, but not vice versa. That is, parameters that are not matched by a predictor are ignored.

Usage

```
dynamicPredictors(x) <- value
staticPredictors(x) <- value
```

Arguments

x	EcocropModel object
value	matrix with environmental predictor variables. Each column represents an environmental variable, and must have a name that matches a predictor variable. For dynamicPredictors, the matrix must have 12 rows, or a multiple of 12 rows. For staticPredictors, the matrix can have any number of rows. The number of rows in staticPredictors must be 1/12 of the number of rows in the dynamicPredictors.

Value

None

Examples

```
# Get parameters
potato <- ecocropPars("potato")
# create a model
m <- ecocrop(potato)

# add predictors
dp <- cbind(tavg=c(10,12,14,16,18,20,22,20,18,16,14,12), prec=seq(50,182,12))
t(dp)
```

```
dynamicPredictors(m) <- dp
staticPredictors(m) <- cbind(clay=12)
m
```

`removeParameters` *Add or remove parameters and predictors*

Description

Remove parameters or predictors from an EcoCrop model

Usage

```
removeParameters(x, name)
removePredictor(x, name)
```

Arguments

`x`	EcocropModel object
`name`	character. name of the variable to remove; or "ALL" to remove all parameters or predictors.

Value

Nothing is returned

Examples

```
pot <- ecocropPars("potato")
m <- ecocrop(pot)

crop(m) <- cbind(clay = c(0, 10, 30, 40))
removeParameters(m, "ph")

dynamicPredictors(m) <- cbind(tavg=rep(15,12), prec=rep(100, 12))
removePredictor(m, "tavg")
```

Index

control *(ecocrop_model)*, 3
control,Rcpp_EcocropModel-method *(ecocrop_model)*, 3
crop, 3, *6*
crop<- *(crop)*, 3
crop<-,Rcpp_EcocropModel,matrix-method *(crop)*, 3

dynamicPredictors, *4*
dynamicPredictors<- *(predictors)*, 9
dynamicPredictors<-,Rcpp_EcocropModel,matrix-method *(predictors)*, 9

ecocrop *(ecocrop_model)*, 3
ecocrop,Rcpp_EcocropModel-method *(ecocrop_model)*, 3
ecocrop_model, 3
ecocropPars, *3*, *4*, 5

plot *(plot-ecocrop)*, 7
plot,Rcpp_EcocropModel,ANY-method *(plot-ecocrop)*, 7
plot-ecocrop, 7
predict, *4*, 8
predict,Rcpp_EcocropModel-method *(predict)*, 8
predictors, *2*, 9

Rcpp_EcocropModel-class *(ecocrop_model)*, 3
Recocrop *(Recocrop-package)*, 2
Recocrop-package, 2
removeParameters, 11
removePredictor *(removeParameters)*, 11
run *(ecocrop_model)*, 3
run,Rcpp_EcocropModel-method *(ecocrop_model)*, 3

show,Rcpp_EcocropModel-method *(ecocrop_model)*, 3
staticPredictors, *4*
staticPredictors<- *(predictors)*, 9
staticPredictors<-,Rcpp_EcocropModel,matrix-method *(predictors)*, 9

writeRaster, *8*