

Package ‘rcontrol’

July 23, 2025

Type Package

Title Individual-Based Forest Growth Simulator 'TROLL'

Version 0.1.2

Maintainer Sylvain Schmitt <sylvain.m.schmitt@gmail.com>

URL <https://github.com/sylvainschmitt/rcontrol>,
<https://sylvainschmitt.github.io/rcontrol/>

BugReports <https://github.com/sylvainschmitt/rcontrol/issues>

Description 'TROLL' is coded in C++ and it typically simulates hundreds of thousands of individuals over hundreds of years. The 'rcontrol' R package is a wrapper of 'TROLL'. 'rcontrol' includes functions that generate inputs for simulations and run simulations. Finally, it is possible to analyse the 'TROLL' outputs through tables, figures, and maps taking advantage of other R visualisation packages. 'rcontrol' also offers the possibility to generate a virtual LiDAR point cloud that corresponds to a snapshot of the simulated forest.

License GPL-3

LazyData TRUE

Imports methods (>= 3.5.0), utils (>= 3.5.0), readr (>= 1.3.0), sys (>= 3.2), dplyr (>= 1.0.0), magrittr (>= 2.0.0), reshape2 (>= 1.3.0), ggplot2 (>= 3.2.0), viridis (>= 0.4.0), parallel (>= 3.5.0), doParallel (>= 1.0.7), doSNOW (>= 1.0.10), foreach (>= 1.4.1), iterators (>= 1.0.5), Rcpp (>= 1.0.5), gganimate (>= 1.0.0), vroom (>= 1.0.0), tidyr (>= 1.0.0), tibble (>= 3.0.0), lubridate (>= 1.6.0), terra (>= 1.5-16), lidR (>= 4.0.1)

Suggests markdown, knitr, rmarkdown, testthat, covr

RoxygenNote 7.2.3

Collate 'RcppExports.R' 'TROLLv3_climatedaytime12.R'
'TROLLv3_daytimevar.R' 'TROLLv3_input.R' 'TROLLv3_output.R'
'TROLLv3_pointcloud.R' 'TROLLv3_species.R' 'zzz.R'
'TROLLversion.R' 'trollsim.R' 'load_output.R' 'troll.R'
'autogif.R' 'get_chm.trollsim.R' 'trollstack.R'

'autoplot.troll.R' 'utils-pipe.R' 'generate_climate.R'
 'generate_lidar.R' 'generate_parameters.R'
 'get_forest.trollsim.R' 'get_log.trollsim.R' 'load_stack.R'
 'load_sim.trollsim.R' 'print.trollsim.R' 'rcontroll-package.R'
 'rcontroll.R' 'stack.R' 'update_parameters.trollsim.R'

VignetteBuilder knitr

Depends R (>= 3.6)

LinkingTo Rcpp, RcppGSL

Encoding UTF-8

NeedsCompilation yes

Author Sylvain Schmitt [aut, cre] (ORCID:

<<https://orcid.org/0000-0001-7759-7106>>),

Guillaume Salzet [aut] (ORCID: <<https://orcid.org/0000-0003-4548-5673>>),

Fabian Fischer [aut] (ORCID: <<https://orcid.org/0000-0003-2325-9886>>),

Isabelle Maréchaux [aut] (ORCID:

<<https://orcid.org/0000-0002-5401-0197>>),

Jérôme Chave [aut] (ORCID: <<https://orcid.org/0000-0002-7766-1347>>)

Repository CRAN

Date/Publication 2024-09-30 08:50:02 UTC

Contents

autogif	3
autoplot,trollsim-method	4
generate_climate	6
generate_lidar	9
generate_parameters	10
get_chm	14
get_forest	14
get_log	15
load_output	16
load_sim	17
load_stack	17
option.rcontroll	18
print.trollsim	18
rcontroll	19
stack	20
troll	23
TROLL.version	24
trollC++	25
trollsim	26
trollsim-class	27
trollstack	28
trollstack-class	29
TROLLv3_climatedaytime12	29

TROLLv3_daytimevar	30
TROLLv3_input	31
TROLLv3_output	31
TROLLv3_pointcloud	32
TROLLv3_species	32
update_parameters	33

Index 34

autogif *Create an animation from a TROLL simulation*

Description

autogif() uses ggplot2 and gganimate to render an animation from a TROLL simulation. The animation can include a vertical cut in the forest structure along the X-axis highlighting either tree species (variables = 'species'), either tree height relative to their maximum height (variables = 'height_ct'), or tree carbon acquisition with net over growth primary productivity (variables = 'npp_gpp'). The animation can also include a top view of the canopy representing either canopy trees height (variables = 'height') or total leaf area index per pixel (variables = 'lai').

Usage

```
autogif(
  name = NULL,
  path = NULL,
  variables = c("species", "height_ct", "npp_gpp", "height", "lai"),
  global,
  species,
  climate,
  daily,
  forest = NULL,
  verbose = TRUE,
  overwrite = TRUE,
  thin = NULL
)
```

Arguments

name	char. Model name (if NULL timestamp).
path	char. Path to save the simulation outputs, the default is null corresponding to a simulation in memory without saved intermediary files (based on temporary files from option.rcontrol).
variables	char. Variables to build as a gif among 'species', 'height_ct', 'npp_gpp', 'height', or 'lai'.
global	df. Global parameters (e.g. TROLLv3_input or using generate_parameters()).
species	df. Species parameters (e.g. TROLLv3_species).

climate	df. Climate parameters (e.g. TROLLv3_climatedaytime12).
daily	df. Daily variation parameters (e.g. TROLLv3_daytimevar).
forest	df. TROLL with forest input, if null starts from an empty grid (default NULL) (e.g. using TROLLv3_output with get_forest()).
verbose	bool. Show TROLL outputs in the console.
overwrite	bool. Overwrite previous outputs.
thin	int. Vector of integers corresponding to the iterations to be kept to reduce output size, default is NULL and corresponds to no thinning.

Value

A list of gganimate objects corresponding to chosen outputs.

See Also

[autoplot,trollsim-method](#)

Examples

```
## Not run:
data("TROLLv3_species")
data("TROLLv3_climatedaytime12")
data("TROLLv3_daytimevar")
autogif(
  name = "test", global = generate_parameters(
    cols = 100, rows = 100,
    iterperyear = 12, nbiter = 12 * 100,
    extent_visual = 10
  ),
  species = TROLLv3_species,
  climate = TROLLv3_climatedaytime12,
  daily = TROLLv3_daytimevar,
  verbose = FALSE
)

## End(Not run)
```

Description

autoplot() is a method that takes advantage of ggplot2 to plot TROLL simulations. autoplot() can plot either temporal trajectories of whole ecosystem or species metrics (what = 'temporal'), the initial or final pattern observed in the forest community (what = 'spatial' or what = 'distribution'), or lidar outputs (what = 'lidar'). Metrics includes abundances of individuals above 1cm (N), above 10cm (N10), and above 30cm (N30), aboveground biomass (AGB), basal area of individuals above 1cm (BA), and above 10cm (BA10), gross primary production (GPP), net primary production (NPP), respiration of day (Rday), night (Rnight) and stem (Rstem), and litterfall.

Usage

```
## S4 method for signature 'trollsim'
autoplot(
  object,
  what = "temporal",
  variables = NULL,
  species = NULL,
  iter = NULL
)
```

Arguments

object	TROLL simulation or stack (see troll() , stack() , trollsim() and trollstack()).
what	char. What to plot: "temporal", "spatial" "distribution", or "lidar". "temporal" is for temporal trajectories of the whole ecosystem or defined species. "spatial" is for spatial patterns in the initial or final forest. "distribution" is for metrics distribution in the initial or final forest. "lidar" is for canopy height model plot.
variables	char. Which variable(s) to plot. Only one variable is accepted when plotting "spatial".
species	char. Which species to plot. NULL indicates the whole ecosystem level. "all" can be used to use all species.
iter	char. Which iteration(s) to plot, for temporal thinning or to specify which forest to plot. "initial" or "final" can be used. NULL is converted to "final".

Value

A ggplot2 object.

See Also

[autogif\(\)](#), [summary,trollsim-method](#)

Examples

```
data("TROLLv3_output")
autoplot(TROLLv3_output)
```

generate_climate *Generate climate dataset*

Description

TROLL forest simulator relies on climate tables with half-hourly variations of a typical day and monthly variations of a typical year which are recycled through simulation days and years. Initially, TROLL climate tables were computed from the Nourflux dataset. Variations in quantities of interests (temperatures, ...) were averaged to the target resolution (half-hour for daily variation or month for monthly variation). The purpose of climate generation functions is to compute equivalent climate tables from the ERA5 land reanalysis dataset (Muñoz-Sabater et al. 2021). With these functions, `rcontroll` users only need inventories and associated functional traits to run TROLL simulations. See the corresponding vignette `vignette("climate", package = "rcontroll")` for further details.

Usage

```
generate_climate(
  x,
  y,
  tz,
  era5land_hour,
  era5land_month,
  daytime_start = 7,
  daytime_end = 19
)
```

Arguments

<code>x</code>	num. Longitude in UTM. Can be obtained from the location name with <code>nominatimlite::geo_lite_sf()</code> .
<code>y</code>	num. Latitude in UTM. Can be obtained from the location name with <code>nominatimlite::geo_lite_sf()</code> .
<code>tz</code>	num. Time zone. Can be obtained from the coordinates with <code>lutz::tz_lookup_coords()</code> .
<code>era5land_hour</code>	str. Path to ERA5 land data monthly averaged reanalysis by hour of day in netCDF. See the corresponding vignette <code>vignette("climate", package = "rcontroll")</code> to download corresponding data from Copernicus in R.
<code>era5land_month</code>	str. Path to ERA5 land data monthly averaged reanalysis in netCDF. See the corresponding vignette <code>vignette("climate", package = "rcontroll")</code> to download corresponding data from Copernicus in R.
<code>daytime_start</code>	int. Daytime starting hour to compute nigh and day variables (default 7).
<code>daytime_end</code>	int. Daytime ending hour to compute nigh and day variables (default 19).

Details

The TROLL forest model simulates tree growth based on ecophysiological processes, with an external climate forcing. Input climatic conditions are provided in the form of climate tables with (i) half-hourly standardised variation of a typical day, and (ii) monthly average values of a typical year,

which are currently recycled through simulation. Initially, TROLL climate tables were computed from the Nourflux dataset (Poncy et al., 1998). The variation in quantities of interest (irradiance, temperature, vapour pressure, rainfall, and wind speed) were averaged to the target resolution (half-hour for daily variation or month for monthly variation).

The purpose of the climate generation function is to compute equivalent climate tables from a global climatic reanalysis dataset. With `generate_climate`, `rcontrol1` users no longer need to format complex climate input fields, but can generate them from global and carefully documented climate distributions to run TROLL simulations. The selected input climate product for this version of `rcontrol1` is ERA5-Land (Muñoz-Sabater et al. 2021). The ERA5-Land climate reanalysis has two main advantages over other climate reanalysis products: (1) the data are at a spatial resolution of 9km and have been available at hourly temporal resolution since 1950, and (2) daily or monthly averages are available and their uncertainties are reported.

Hypotheses:

The following assumptions are made in the generation of climate data:

1. The temperature at 2m and its derivatives (d2m) from ERA5-land corresponds to air temperature measurement used as an input in TROLL;
2. We can calculate the vapour pressure deficit using the Buck equation;
3. The extraction of standardised half-hourly values of an average day and of monthly average values of a year for the climate variables of interest is based on the decomposition of the raw time series into: (i) an overall trend over the study period, (ii) seasonal or daily variation across months or hours depending on the study level, and (iii) the remaining variation;
4. Half-hourly values are not available for ERA5-land data. Spline functions are used to interpolate hourly values for downscaling to half-hourly resolution.

Quantities of interest:

TROLL variables:

TROLL climate tables summarise temporal variation of quantities of interest. These variations are called **seasonal pattern** and are computed from time series under the additive assumption : $X(t) = Trend_X(t) + Seasonal_{x,Period}(t \bmod p[modPeriod]) + Irregular(t)$

With :

- $Trend_X(t)$ a moving average covering one period;
- $Seasonal_{x,Period}(t \bmod p[modPeriod])$ the seasonal pattern contribution at t time modulo the period;
- $Irregular(t)$ the remainder part.

The de-seasonally average of X is defined as :

$$mean(X) = mean(Trend_X(t))$$

These values of seasonal pattern and de-seasonally average are used to compute the climate table `TROLLv3_climatedaytime12` and `TROLLv3_daytimevar`.

ERA5-Land variables:

There is a restricted set of variables needed to generate the TROLL climate files:

- t2m: Temperature at 2m in K
- d2m: Dew point temperature at 2m in K
- tp: Cumulative rainfall in m
- sp: Atmospheric pressure at the surface in Pa
- ssrd: Cumulative Net solar irradiation in J/m2

- u10: Zonal wind component at 10m in m/s
- v10: Meridional wind component at 10m in m/s

Calculation of variables:

The transition from ERA-Land data to TROLL data requires several transformations. The TROLL climate files correspond to seasonal components, either daily from 7am to 7pm, or monthly. The extraction of these seasonal components is possible by analysing the time series of the data. An additive decomposition of the variables allows one to obtain the pattern of interest at the original resolution (hourly or monthly). Interpolation of the pattern using spline functions of the periodic type, ensuring the boundary conditions (i.e. value at 0 am is the same as 12 pm), except for the ssrd which is not a continuous periodic function and which requires natural type spline interpolation. An evaluation of the quality of the pattern extraction is possible by measuring the standard deviation of the error to the original time series. This error can be calculated for each unit of the pattern (for each hour for example).

Wind speed:

The wind speed is the norm of the vector generated by the u10 and v10 components of the wind. We can therefore deduce that the wind speed corresponds to:

$$WindSpeed = \sqrt{u10^2 + v10^2}$$

Vapour pressure deficit:

The calculation of the vapour pressure deficit can be done according to three variables (t2m, d2m and sp) using the formula of (Buck 1981):

$$VPD = e_{sat}(d2m, sp) - e_{obs}(2m, sp)$$

$$e_*(t2m|d2m, sp) = 611.21x f(t2m|d2m, sp)x(1)$$

$$(1) = 18.678 - (t2m|d2m - 273.15)/234.5x(t2m|d2m - 273.15)/(240.97 + t2m|d2m - 273.15)$$

$$f(t2m|d2m, sp) = 1.0007 + 10^{-7}xspx0.032 + 5.9x10^{-6}x(t2m|d2m - 273.15)^2$$

Instantaneous irradiance:

The measurement of irradiance in the ERA5 data corresponds to the accumulation either at hourly or daily intervals. The instantaneous measurement can be obtained by calculating the variation of this accumulation:

$$\Delta ssrd(t) = ssrd(t) - ssrd(t - 1)$$

Conclusion:

In conclusion, despite some discrepancies between climate input generated from local meteorological station data and the generate_climate function that should be investigated further, the generate_climate function allows rcontrol1 users to easily obtain relevant climate data for their study. The discrepancies may be partly due to the unconventional situation of the Nourflux station, which should not be considered as the true climate.

Warning: As TROLL is under constant development, some of the variables presented here may not be used in the current version (v 3.1.7) and may be left over from previous versions or may be intended for future versions. Furthermore, this supplementary information corresponds to the version 3.1.7 of TROLL and the climate variables used by the model may change as new versions of TROLL are released. We plan to include future major developments of TROLL in rcontrol1 to keep the advances of the model accessible to the community, including the development of the generate_climate function. We thus invite the reader to check the corresponding updated vignette on GitHub (<https://sylvainschmitt.github.io/rcontrol1/articles/climate.html>) according to the version of TROLL they are using in rcontrol1 (check with `TROLL.version()`).

References:

Buck, Arden L. (1981) New equations for computing vapor pressure and enhancement factor. *Journal of Applied Meteorology and Climatology*, 1981, vol. 20, no 12, p. 1527-1532.

Muñoz-Sabater, J., Dutra, E., Agustí-Panareda, A., Albergel, C., Arduini, G., Balsamo, G., ... Thépaut, J. N. (2021). ERA5-Land: A state-of-the-art global reanalysis dataset for land applications. *Earth System Science Data*, 13(9), 4349–4383. <https://doi.org/10.5194/essd-13-4349-2021>

Poncy, O., Riéra, B., Larpin, D., Belbenoit, P., Jullien, M., Hoff, M., & Charles-Dominique, P. (1998). The permanent field research station “Les Nouragues” in the tropical rainforest of French Guiana: current projects and preliminary results on tree diversity, structure, and dynamics. *Forest Biodiversity in North, Central and South America, and the Caribbean: Research and Monitoring*, 385–410.

Value

A list with two `data.frame()`: daytimevar and climatedaytime12.

generate_lidar	<i>Generate lidar parameters</i>
----------------	----------------------------------

Description

generate_lidar generate the lidar parameters used in TROLL lidar simulation. All parameters have a default value from literature.

Usage

```
generate_lidar(
  mean_beam_pc = 10,
  sd_beam_pc = 5,
  klaser_pc = 0.63,
  transmittance_laser = 0.4,
  iter_pointcloud_generation = NULL
)
```

Arguments

mean_beam_pc	num. Mean pulse density (pulses per m2).
sd_beam_pc	num. Standard deviation of pulse density (per m2).
klaser_pc	num. laser attenuation factor.
transmittance_laser	num. Percentage of pulses that continue through the canopy after a hit.
iter_pointcloud_generation	num. Number of iteration for point cloud generation.

Value

A [data.frame]) of lidar simulation parameters.
[data.frame]): R:data.frame)

See Also

[troll\(\)](#), [stack\(\)](#)

Examples

```
generate_lidar(iter_pointcloud_generation = 3600)
```

generate_parameters *Generate global parameters*

Description

generate_parameters() generate the global parameters used in the TROLL simulation. All parameters have a default value used in French Guiana simulations.

Usage

```
generate_parameters(  
  cols = 200,  
  rows = 200,  
  HEIGHT = 70,  
  length_dcell = 25,  
  nbiter,  
  iterperyear = 12,  
  NV = 1,  
  NH = 1,  
  nbout = 4,  
  nbspp = 45,  
  SWtoPPFD = 2.27,  
  p_nonvert = 0.05,  
  klight = 0.63,  
  phi = 0.093,  
  absorptance_leaves = 0.9,  
  theta = 0.7,  
  g1 = 3.77,  
  vC = 0.021,  
  DBH0 = 0.005,  
  H0 = 0.95,  
  CR_min = 0.3,  
  CR_a = 2.13,  
  CR_b = 0.63,
```

```

CD_a = 0,
CD_b = 0.2,
CD0 = 0.3,
shape_crown = 0.72,
dens = 1,
fallocwood = 0.35,
falloccanopy = 0.25,
Cseedrain = 50000,
nbs0 = 10,
sigma_height = 0,
sigma_CR = 0,
sigma_CD = 0,
sigma_P = 0,
sigma_N = 0,
sigma_LMA = 0,
sigma_wsg = 0,
sigma_dbhmax = 0,
corr_CR_height = 0,
corr_N_P = 0,
corr_N_LMA = 0,
corr_P_LMA = 0,
leafdem_resolution = 30,
p_tfsecondary = 1,
hurt_decay = 0,
crown_gap_fraction = 0.15,
m = 0.013,
m1 = 0.013,
Cair = 400,
LL_parameterization = 1,
LA_regulation = 2,
sapwood = 1,
seedsadditional = 0,
NONRANDOM = 1,
GPPcrown = 0,
BASICTREEFALL = 1,
SEEDTRADEOFF = 0,
CROWN_MM = 0,
OUTPUT_extended = 1,
extent_visual = 0
)

```

Arguments

cols	num. Number of columns.
rows	num. Number of rows.
HEIGHT	num. Vertical extent of simulation.
length_dcell	num. Linear size of a dcell.
nbiter	num. Total number of timesteps.

iterperyear	num. Number of iterations per year.
NV	num. Vertical number of cells (per m).
NH	num. Horizontal number of cells (per m).
nbout	num. Number of outputs.
nbspp	num. Number of species
SWtoPPFD	num. Convert shortwave irradiance to PAR photons.
p_nonvert	num. Light incidence parameter (difference through turbid medium).
klight	num. Light attenuation in the canopy following a Beer-Lambert law.
phi	num. Quantum yield (in micromol C/micromol photon).
absorptance_leaves	num. Absorptance of individual leaves.
theta	num. Parameter of the Farquhar model.
g1	num. Parameter g1 of Medlyn et al stomatal conductance model.
vC	num. Variance of the flexion moment.
DBH0	num. Initial diameter at breast height (m).
H0	num. Initial height (m).
CR_min	num. Minimum crown radius (in m).
CR_a	num. Crown radius log intercept or Michaelis Menten initial growth.
CR_b	num. Crown radius log slope or Michaelis Menten asymptotic CR.
CD_a	num. Crown depth intercept (absolute value).
CD_b	num. Crown depth slope (as fraction of tree height).
CD0	num. Initial crown depth (in m).
shape_crown	num. Crown shape parameter.
dens	num. Initial leaf density (m ² /m ²).
fallocwood	num. Fraction of biomass allocated to above ground wood (branch turnover+stem).
falloccanopy	num. Fraction of biomass allocated to canopy (leaves + reproductive organs + twigs).
Cseedrain	num. Constant used to scale total seed rain per hectare across species.
nbs0	num. Number of seeds produced and dispersed by each mature tree when SEED-TRADEOFF is not defined.
sigma_height	num. Intraspecific variation in tree height (lognormal).
sigma_CR	num. Intraspecific variation in crown radius (lognormal).
sigma_CD	num. Intraspecific variation in crown depth (lognormal).
sigma_P	num. Intraspecific variation in leaf phosphorus (lognormal).
sigma_N	num. Intraspecific variation in leaf nitrogen (lognormal).
sigma_LMA	num. Intraspecific variation in leaf mass per area (lognormal).
sigma_wsg	num. Intraspecific variation in wood specific gravity.
sigma_dbhmax	num. Intraspecific variation in maximum diameter.

corr_CR_height	num. Correlation coefficient between crown radius and tree height.
corr_N_P	num. Correlation coefficient between leaf nitrogen and leaf phosphorus.
corr_N_LMA	num. Correlation coefficient between leaf nitrogen and leaf mass per area
corr_P_LMA	num. Correlation coefficient between leaf phosphorus and leaf mass per area
leafdem_resolution	num. Resolution of leaf demography model.
p_tfsecondary	num. Probability of secondary treefall.
hurt_decay	num. Parameter determining how tree damages are repaired.
crown_gap_fraction	num. Fraction of gaps in the crown.
m	num. Minimal death rate.
m1	num. Slope of death rate m1.
Cair	num. Atmospheric CO2 concentration in micromol/mol.
LL_parameterization	num. Leaf lifespan parameterizations: Reich empirical, Kikuzawa model, and Kikuzawa model with leaf plasticity (0,1,2).
LA_regulation	num. Dynamic LA regulation: off, 1.0, 0.75, or 0.5 (0,1,2,3).
sapwood	num. Sapwood parameterizations: constant thickness (0.04), Fyllas percentage, Fyllas lower limit (0,1,2).
seedsadditional	num. Excess biomass into seeds after maturation (0,1).
NONRANDOM	num. If <code>_NONRANDOM</code> \geq 1, the seeds for the random number generators will be set using fixed seed in R, default for bug fixing (0,1).
GPPcrown	num. This defines an option to compute only GPP from the topmost value of PPFD and GPP, instead of looping within the crown (0,1).
BASICTREEFALL	num. If defined: treefall is a source of tree death (0,1).
SEEDTRADEOFF	num. if defined: the number of seeds produced is determined by NPP allocated to reproduction and seed mass, otherwise the number of seeds is fixed (0,1).
CROWN_MM	num. Michaelis Menten allometry for crowns instead of power law, parameters have to be changed in other input sheets accordingly (0,1).
OUTPUT_extended	num. extended set of output files (0,1).
extent_visual	num. extent for visualization output. Unactivated when equal 0.

Value

A data frame of global parameters.

See Also

[troll\(\)](#), [stack\(\)](#), [update_parameters\(\)](#)

Examples

```
generate_parameters(nbiter = 12)
```

get_chm	<i>Extract canopy height model</i>
---------	------------------------------------

Description

get_chm() extract the canopy height model from TROLL outputs with lidar option.

Usage

```
get_chm(sim, method = "smoothed", ...)  
  
## S4 method for signature 'trollsim'  
get_chm(sim, method = "smoothed", ...)
```

Arguments

sim	trollsim.
method	char. method to extract the canopy height model from the point cloud in las, either 'filled' (replacing NA by 0) or 'smoothed' (local means, default value).
...	unused argument.

Value

[data.frame\(\)](#)

See Also

[trollsim\(\)](#), [troll\(\)](#), [stack\(\)](#)

get_forest	<i>Extract forest inventory</i>
------------	---------------------------------

Description

get_forest() extract the forest inventory from TROLL outputs.

Usage

```
get_forest(sim, ...)  
  
## S4 method for signature 'trollsim'  
get_forest(sim, ...)
```

Arguments

sim trollsim or trollstack.
... unused argument.

Value

`data.frame()`

See Also

`trollsim()`, `trollstack()`, `troll()`, `stack()`

Examples

```
data("TROLLv3_output")  
head(get_forest(TROLLv3_output))
```

get_log

Extract simulation log

Description

`get_forest()` extract the simulation log TROLL outputs.

Usage

```
get_log(sim, ...)  
  
## S4 method for signature 'trollsim'  
get_log(sim, ...)
```

Arguments

sim trollsim or trollstack.
... unused argument.

Value

the log in the console

See Also

`trollsim()`, `trollstack()`, `troll()`, `stack()`

Examples

```
data("TROLLv3_output")
get_log(TROLLv3_output)
```

load_output	<i>Load outputs from simulation</i>
-------------	-------------------------------------

Description

load_output load outputs from TROLL simulation files using TROLL simulation name and path.

Usage

```
load_output(name, path, thin = NULL)
```

Arguments

name	char. Name given to the model output.
path	char. Path where the model is saved.
thin	int. Vector of integers corresponding to the iterations to be kept to reduce output size, default is NULL and corresponds to no thinning.

Value

An S4 `trollsim()` class object.

See Also

`trollsim()`, `trollstack()`, `load_sim()`, `load_stack()`

Examples

```
## Not run:
load_output("test", "./")

## End(Not run)
```

load_sim	<i>Load outputs from simulation or stack of simulations</i>
----------	---

Description

load_sim is a method of `trollsim()` or `trollstack()` with wirtten files not in R memory to load them into R memory taking advantage of `load_output()` and `load_stack()`.

Usage

```
load_sim(sim, ...)

## S4 method for signature 'trollsim'
load_sim(sim, ...)
```

Arguments

sim	trollsim or trollstack.
...	unused argument.

Value

An S4 `trollsim()` or `trollstack()` class object.

See Also

`trollsim()`, `trollstack()`, `load_sim()`, `load_stack()`

load_stack	<i>Load outputs from a stack of simulations</i>
------------	---

Description

load_stack load outputs from a stack of TROLL simulation files using TROLL stack of simulation name and path.

Usage

```
load_stack(name, path, thin = NULL)
```

Arguments

name	char. Name given to the stack output.
path	char. Path where the stack is saved.
thin	int. Vector of integers corresponding to the iterations to be kept to reduce output size, default is NULL and corresponds to no thinning.

Value

An S4 `trollstack()` class object.

See Also

`trollsim()`, `trollstack()`, `load_sim()`, `load_stack()`

Examples

```
## Not run:
load_stack("test", ".")

## End(Not run)
```

option.rcontroll	<i>Options</i>
------------------	----------------

Description

rcontroll package global options including temporary files location and 'TROLL version.

Arguments

rcontroll.tmp char. Path to temporary files folder used by `troll()` and `stack()`.
 rcontroll.troll char. TROLLversion number accessible with `TROLL.version()`.

print.trollsim	<i>Print a summary</i>
----------------	------------------------

Description

print() prints a summary of TROLL simulation or stack of simulations outputs.

Usage

```
## S4 method for signature 'trollsim'
print(x, ...)

## S4 method for signature 'trollsim'
show(object)

## S4 method for signature 'trollsim'
summary(object, ...)
```

Arguments

x	trollsim or trollstack.
...	unused argument.
object	trollsim or trollstack.

Value

Print or show in console.

See Also

[troll\(\)](#), [stack\(\)](#), [trollsim\(\)](#), [trollstack\(\)](#)

Examples

```
data("TROLLv3_output")
print(TROLLv3_output)
```

rcontrol1

rcontrol1: *individual-based forest growth simulator TROLL*

Description

TROLL is coded in C++ and it typically simulates hundreds of thousands of individuals over hundreds of years. The rcontrol1 R package is a wrapper of TROLL. rcontrol1 includes functions that generate inputs for simulations and run simulations. Finally, it is possible to analyse the TROLL outputs through tables, figures, and maps taking advantage of other R visualisation packages. rcontrol1 also offers the possibility to generate a virtual LIDAR point cloud that corresponds to a snapshot of the simulated forest.

Construction and manipulation of input files

As stated above, three types of input data are needed for a typical TROLL simulation: (i) climate data, (ii) plant functional traits, (iii) global model parameters. Pre-simulation functions include global parameters definition (`generate_parameters` function) and climate data generation (`generate_climate` function). rcontrol1 also includes default data for species and climate inputs for a typical French Guiana rainforest site. The purpose of the `generate_climate` function with the help of the corresponding vignette is to create TROLL climate inputs from ERA5-Land (Muñoz-Sabater et al. 2021), a global climatic reanalysis dataset that is freely available. The ERA5-Land climate reanalysis is available at 9 km spatial resolution and hourly temporal resolution since 1950, and daily or monthly means are available and their uncertainties reported. Therefore, rcontrol1 users only need to input the species-specific trait data to run TROLL simulations, irrespective of the site. TROLL was originally developed for tropical and subtropical forests, so certain assumptions must be critically examined when applying it outside the tropics. The input files can be used to start a TROLL simulation run within the rcontrol1 environment (see below), or saved so that the TROLL simulation can be started as a command line tool.

Simulations

The default option is to run a TROLL simulation using the `troll` function of the `rcontroll` package, which currently calls version 3.1.7 of TROLL using the `Rcpp` package (Eddelbuettel & François 2011). The output is stored in a `trollsim` R class. For multiple runs, users can rely on the `stack` function, and the output is stored in the `trollstack` class. Both `trollsim` and `trollstack` values can be accessed using object attributes in the form of simple R objects (with `@` in R). They consist of eight simulation attributes: (1) name, (2) path to saved files, (3) parameters, (4) inputs, (5) log, (6) initial and final state, (7) ecosystem output metrics, and (8) species output metrics. The initial and final states are represented by a table with the spatial position, size and other relevant traits of all trees at the start and end of the simulation. The ecosystem and species metrics are summaries of ecosystem processes and states, such as net primary production and aboveground biomass, and they are documented at species level and aggregated over the entire stand. Simulations can be saved using a user-defined path when run and later loaded as a simple simulation (`load_output` function) or a stack of simulations (`load_stack` function).

Simulated airborne lidar scanning option

TROLL also has the capacity of generating point clouds from virtual aerial lidar scanings of simulated forest scenes. Within each cubic metre voxel of the simulated stand, points are generated probabilistically, with the probability depending both on the amount of light reaching the particular voxel and the amount of leaf matter intercepting light within the voxel. Extinction and interception of light are based on the Beer-Lambert law, but an effective extinction factor is used to account for differences between the near-infrared and visible light. The definition of the lidar parameters (`generate_lidar` function) is optional but allows the user to add a virtual aerial lidar scan for a time step of the TROLL simulation. When this option is enabled, the cloud of points from simulated aerial lidar scans are stored as LAS using the R package `lidR` (Roussel et al., 2020) as a ninth attribute of the `trollsim` and `trollstack` objects.

Manipulation of simulation outputs

`rcontroll` includes functions to manipulate simulation outputs. Simulation outputs can be retrieved directly from the `trollsim` or `trollstack` objects and summarised or plotted in the R environment with the `print`, `summary` and `autoplot` functions. The `get_chm` function allows users to retrieve canopy height models from aerial lidar point clouds. In addition, a `rcontroll` function is available to visualise TROLL simulations as an animated figure (`autogif` function).

TROLL

version 3.1.6

Description

`stack()` run a stack of TROLL simulation. The minimal set of input files required for a TROLL run include (i) climate data for the focal location (`climate` and `daily`), (ii) functional traits for the list of species at the focal location (`species`), and (iii) global parameters (`global`), i.e. parameters that do not depend on species identity.

Usage

```
stack(
  name = NULL,
  simulations,
  path = NULL,
  global,
  species,
  climate,
  daily,
  lidar = NULL,
  forest = NULL,
  load = TRUE,
  cores = NULL,
  verbose = TRUE,
  overwrite = TRUE,
  thin = NULL
)
```

Arguments

<code>name</code>	char. Stack name (if NULL the timestamp will be used).
<code>simulations</code>	char. Simulation names (corresponding to simulation indexes in corresponding tables, see example below).
<code>path</code>	char. Path to save the stack of simulation outputs (parent folder), the default is null corresponding to a simulation in memory without saved intermediary files (based on temporary files from option.rcontrol).
<code>global</code>	df. Global parameters (e.g. TROLLv3_input or using generate_parameters()).
<code>species</code>	df. Species parameters (e.g. TROLLv3_species).
<code>climate</code>	df. Climate parameters (e.g. TROLLv3_climatedaytime12).
<code>daily</code>	df. Daily variation parameters (e.g. TROLLv3_daytimevar).
<code>lidar</code>	df. Lidar simulation parameters (e.g. using generate_lidar()), if null not computed (default NULL).
<code>forest</code>	df. TROLL with forest input, if null starts from an empty grid (default NULL) (e.g. using TROLLv3_output with get_forest()).
<code>load</code>	bool. TROLL outputs are loaded in R memory, if not only the path and name of the stack of simulations is kept in the resulting trollstack() object but the content can be accessed later using the load_sim() method.

cores	int. Number of cores for parallelization, if NULL available cores - 1 (default NULL). You can use <code>parallel::detectCores()</code> to know available cores on your machine.
verbose	bool. Show TROLL log in the console.
overwrite	bool. Overwrite previous outputs folder and files.
thin	int. Vector of integers corresponding to the iterations to be kept to reduce output size, default is NULL and corresponds to no thinning.

Value

A `trollstack()` object.

See Also

`troll()`

Examples

```
## Not run:
data("TROLLv3_species")
data("TROLLv3_climatedaytime12")
data("TROLLv3_daytimevar")
data("TROLLv3_output")
TROLLv3_input_stack <- generate_parameters(
  cols = 100, rows = 100,
  iterperyear = 12, nbiter = 12 * 1
) %>%
  mutate(simulation = list(c("seed50000", "seed500"))) %>%
  unnest(simulation)
TROLLv3_input_stack[62, 2] <- 500 # Cseedrain
stack(
  name = "teststack",
  simulations = c("seed50000", "seed500"),
  global = TROLLv3_input_stack,
  species = TROLLv3_species,
  climate = TROLLv3_climatedaytime12,
  daily = TROLLv3_daytimevar,
  load = TRUE,
  cores = 2,
  verbose = FALSE,
  thin = c(1, 5, 10)
)

## End(Not run)
```

troll	<i>Run a TROLL simulation</i>
-------	-------------------------------

Description

`troll()` run a TROLL simulation. The minimal set of input files required for a TROLL run include (i) climate data for the focal location (`climate` and `daily`), (ii) functional traits for the list of species at the focal location (`species`), and (iii) global parameters (`global`), i.e. parameters that do not depend on species identity.

Usage

```
troll(
  name = NULL,
  path = NULL,
  global,
  species,
  climate,
  daily,
  lidar = NULL,
  forest = NULL,
  load = TRUE,
  verbose = TRUE,
  overwrite = TRUE,
  thin = NULL
)
```

Arguments

<code>name</code>	char. Model name (if NULL the timestamp will be used).
<code>path</code>	char. Path to save the simulation outputs, the default is null corresponding to a simulation in memory without saved intermediary files (based on temporary files from option.rcontrol).
<code>global</code>	df. Global parameters (e.g. TROLLv3_input or using generate_parameters()).
<code>species</code>	df. Species parameters (e.g. TROLLv3_species).
<code>climate</code>	df. Climate parameters (e.g. TROLLv3_climatedaytime12).
<code>daily</code>	df. Daily variation parameters (e.g. TROLLv3_daytimevar).
<code>lidar</code>	df. Lidar simulation parameters (e.g. using generate_lidar()), if null not computed (default NULL).
<code>forest</code>	df. TROLL with forest input, if null starts from an empty grid (default NULL) (e.g. using TROLLv3_output with get_forest()).
<code>load</code>	bool. TROLL outputs are loaded in R memory, if not only the path and name of the simulations is kept in the resulting <code>trollsim()</code> object but the content can be accessed later using the load_sim() method.

verbose	bool. Show TROLL log in the console.
overwrite	bool. Overwrite previous outputs folder and files.
thin	int. Vector of integers corresponding to the iterations to be kept to reduce output size, default is NULL and corresponds to no thinning.

Value

A `trollsim()` object.

See Also

[stack\(\)](#)

Examples

```
data("TROLLv3_species")
data("TROLLv3_climatedaytime12")
data("TROLLv3_daytimevar")
troll(
  name = "test",
  global = generate_parameters(
    cols = 100, rows = 100,
    iterperyear = 12, nbiter = 12 * 1
  ),
  species = TROLLv3_species,
  climate = TROLLv3_climatedaytime12,
  daily = TROLLv3_daytimevar
)
```

TROLL.version

TROLL version

Description

`TROLL.version()` prints TROLL version.

Usage

```
TROLL.version()
```

Value

TROLL version in console.

See Also

[option.rcontroll](#)

Examples

```
TROLL.version()
```

trollCpp

TROLL simulator

Description

Wrapper of the TROLL C++ simulator with Rcpp.

Usage

```
trollCpp(  
  global_file,  
  climate_file,  
  species_file,  
  day_file,  
  lidar_file,  
  forest_file,  
  output_file  
)
```

Arguments

global_file	char. Path to the global parameters file.
climate_file	char. Path to the climate file.
species_file	char. Path to the species file.
day_file	char. Path to the daytime file.
lidar_file	char. Path to the lidar file.
forest_file	char. Path to the forest file.
output_file	char. Path to the output folder.

Value

Void with outputs files written in the defined folder.

Examples

```
## Not run:  
trollCpp(  
  global_file = "test/test_input_global.txt",  
  climate_file = "test/test_input_climate.txt",  
  species_file = "test/test_input_species.txt",  
  day_file = "test/test_input_daily.txt",  
  lidar_file = "",
```

```

    forest_file = "",
    output_file = "test"
  )

  ## End(Not run)

```

trollsim

A TROLL *simulations*

Description

trollsim() is an S4 class to represent a TROLL simulation. trollsim values can be accessed using object attributes in the form of simple R objects (with @). They consist of eight simulation attributes: (1) name, (2) path to saved files, (3) parameters, (4) inputs, (5) log, (6) initial and final state, (7) ecosystem output metrics, and (8) species output metrics. The initial and final states are represented by a table with the spatial position, size and other relevant traits of all trees at the start and end of the simulation. The ecosystem and species metrics are summaries of ecosystem processes and states, such as net primary production and aboveground biomass, and they are documented at species level and aggregated over the entire stand.

Usage

```

trollsim(
  name = character(),
  path = character(),
  mem = logical(),
  parameters = numeric(),
  inputs = list(),
  log = character(),
  forest = data.frame(),
  ecosystem = data.frame(),
  species = data.frame(),
  las = list()
)

```

Arguments

name	char. Simulation name.
path	char. File path to the stack of simulation (parent folder).
mem	bool. Is the simulation in memory, see load_sim() .
parameters	numeric. Parameters of the simulation (general inputs).
inputs	list. Simulation inputs (species, climate, daily, forest, lidar, see troll()).
log	chr. Simulation log, see get_log() .
forest	df. Simulation initial and final forest, see get_forest() .
ecosystem	df. Ecosystem metrics.

species df. Species metrics (with OUTPUT_extended option, see [generate_parameters\(\)](#)).
 las list. List with simulated point cloud in LAS from lidar parameters (with lidar option, see [generate_lidar\(\)](#)). The LAS format correspond to [lidR::LAS\(\)](#).

Value

An empty S4 [trollsim\(\)](#) class object.

See Also

[troll\(\)](#), [load_output\(\)](#), [trollstack\(\)](#)

trollsim-class	A TROLL <i>simulations</i>
----------------	----------------------------

Description

[trollsim\(\)](#) is an S4 class to represent a TROLL simulation. [trollsim](#) values can be accessed using object attributes in the form of simple R objects (with @). They consist of eight simulation attributes: (1) name, (2) path to saved files, (3) parameters, (4) inputs, (5) log, (6) initial and final state, (7) ecosystem output metrics, and (8) species output metrics. The initial and final states are represented by a table with the spatial position, size and other relevant traits of all trees at the start and end of the simulation. The ecosystem and species metrics are summaries of ecosystem processes and states, such as net primary production and aboveground biomass, and they are documented at species level and aggregated over the entire stand.

Value

An empty S4 [trollsim\(\)](#) class object.

Slots

name char. Simulation name.
 path char. File path to the simulation.
 mem bool. Is the simulation in memory, see [load_sim\(\)](#).
 parameters numeric. Parameters of the simulation (general inputs).
 inputs list. Simulation inputs (species, climate, daily, forest, lidar, see [stack\(\)](#)).
 log chr. Simulation log, see [get_log\(\)](#).
 forest df. Simulation initial and final forest, see [get_forest\(\)](#).
 ecosystem df. Ecosystem metrics.
 species df. Species metrics (with OUTPUT_extended option, see [generate_parameters\(\)](#)).
 las list. List with simulated point cloud in LAS from lidar parameters (with lidar option, see [generate_lidar\(\)](#)). The LAS format correspond to [lidR::LAS\(\)](#).

See Also

[troll\(\)](#), [load_output\(\)](#), [trollstack\(\)](#)

Description

`trollstack()` is an S4 class to represent a stack of TROLL simulation. `trollstack` values can be accessed using object attributes in the form of simple R objects (with `@`). They consist of eight simulation attributes: (1) name, (2) path to saved files, (3) parameters, (4) inputs, (5) log, (6) initial and final state, (7) ecosystem output metrics, and (8) species output metrics. The initial and final states are represented by a table with the spatial position, size and other relevant traits of all trees at the start and end of the simulation. The ecosystem and species metrics are summaries of ecosystem processes and states, such as net primary production and aboveground biomass, and they are documented at species level and aggregated over the entire stand.

Usage

```
trollstack(
  name = character(),
  path = character(),
  mem = logical(),
  parameters = numeric(),
  inputs = list(),
  log = character(),
  forest = data.frame(),
  ecosystem = data.frame(),
  species = data.frame(),
  las = list()
)
```

Arguments

<code>name</code>	char. Simulation name.
<code>path</code>	char. File path to the stack of simulation (parent folder).
<code>mem</code>	bool. Is the simulation in memory, see load_sim() .
<code>parameters</code>	numeric. Parameters of the simulation (general inputs).
<code>inputs</code>	list. Simulation inputs (species, climate, daily, forest, lidar, see stack()).
<code>log</code>	chr. Simulation log, see get_log() .
<code>forest</code>	df. Simulation initial and final forest, see get_forest() .
<code>ecosystem</code>	df. Ecosystem metrics.
<code>species</code>	df. Species metrics (with <code>OUTPUT_extended</code> option, see generate_parameters()).
<code>las</code>	list. List with simulated point cloud in LAS from lidar parameters (with lidar option, see generate_lidar()). The LAS format correspond to <code>lidR::LAS()</code> .

Value

An empty S4 `trollstack()` class object.

See Also

`stack()`, `load_stack()`, `trollsim()`

trollstack-class	<i>A stack of TROLL simulations</i>
------------------	-------------------------------------

Description

`trollstack()` is an S4 class to represent a stack of TROLL simulation. `trollstack` values can be accessed using object attributes in the form of simple R objects (with @). They consist of eight simulation attributes: (1) name, (2) path to saved files, (3) parameters, (4) inputs, (5) log, (6) initial and final state, (7) ecosystem output metrics, and (8) species output metrics. The initial and final states are represented by a table with the spatial position, size and other relevant traits of all trees at the start and end of the simulation. The ecosystem and species metrics are summaries of ecosystem processes and states, such as net primary production and aboveground biomass, and they are documented at species level and aggregated over the entire stand.

TROLLv3_climatedaytime12	<i>TROLL climate parameters over months</i>
--------------------------	---

Description

Climate parameters used by TROLL model over months.

Usage

TROLLv3_climatedaytime12

Format

A data frame with 12 rows and 12 variables:

Temperature monthly average of temperature (degree C)

DaytimeMeanTemperature the monthly average of daytime temperature (degree C, 7am-7pm)

NightTemperature the monthly average of temperature at night (degree C, 8pm-6am)

Rainfall monthly average of cumulative rainfall (cm)

WindSpeed monthly average of wind speed (m/s)

DaytimeMeanIrradiance monthly average of cumulative solar irradiance averaged over one day (W/m², 7am-7pm)

- MeanIrradiance** monthly average of cumulative solar irradiance over 24 hours (W/m2)
- SaturatedVapourPressure** monthly average of saturation vapour pressure (hPa)
- VapourPressure** monthly average of vapour pressure (hPa)
- VaporPressureDeficit** monthly average of vapour pressure deficit (VPD, hPa)
- DayTimeVapourPressureDeficitVPDbasic** monthly average of daytime vapour pressure deficit (7am-7pm) according to basic formula (hPa)
- DaytimeMeanVapourPressureDeficit** monthly average of daytime vapour pressure deficit (7am-7pm) according to advanced formula (hPa)

See Also

[generate_climate\(\)](#), [TROLLv3_daytimevar](#)

TROLLv3_daytimevar TROLL *daytime variation parameters*

Description

Daytime variation parameters used by TROLLs models.

Usage

TROLLv3_daytimevar

Format

A data frame with 24 rows and 5 variables:

starttime starting time

endtime ending time

vardaytime_light daily variation in irradiance relative to the mean value of the day

vardaytime_vpd daily variation in vapour pressure deficit relative to the mean value of the day

vardaytime_T daily variation in temperature relative to the mean value of the day

See Also

[generate_climate\(\)](#), [TROLLv3_climatedaytime12](#)

TROLLv3_input	TROLL <i>global parameters</i>
---------------	--------------------------------

Description

Global parameters definition used by TROLL model.

Usage

TROLLv3_input

Format

A data frame with 61 rows and 3 variables:

param global parameter

value value of the parameter

description description of the parameter

See Also

[generate_parameters\(\)](#)

TROLLv3_output	TROLL <i>output</i>
----------------	---------------------

Description

TROLL outputs from a 100-year simulation on a 100x100 grid with the other default parameters and using TROLLv3_species, TROLLv3_climatedaytime12, and TROLLv3_daytimevar for use in tests and examples. Ecosystem level output has been thinned and species output has been removed to save disk space.

Usage

TROLLv3_output

Format

A [trollsim\(\)](#) object.

See Also

[TROLLv3_species\(\)](#), [TROLLv3_climatedaytime12\(\)](#), [TROLLv3_daytimevar\(\)](#), [troll\(\)](#)

TROLLv3_pointcloud TROLL *lidar parameters*

Description

Lidar parameters definition used by TROLL for lidar simulations.

Usage

TROLLv3_pointcloud

Format

A data frame with 5 rows and 3 variables:

param lidar parameter

value value of the parameter

description description of the parameter

See Also

[generate_lidar\(\)](#)

TROLLv3_species TROLL *species parameters*

Description

Functional traits used by TROLL model for 45 species in French Guiana.

Usage

TROLLv3_species

Format

A data frame with 45 rows and 12 variables:

s_name Species name genus_species

s_LMA leaf mass per area

s_Nmass leaf nitrogen mass

s_Pmass leaf phosphorus mass

s_wsg wood specific gravity

s_dbhmax maximum diameter

s_hmax maximum height
s_ah height-diameter allometry coefficient
s_regionalfreq regional frequency
s_tlp Turgor loss point
s_drymass leaf dry mass
s_seedmass seed mass

See Also

[troll\(\)](#), [stack\(\)](#)

update_parameters *Update global parameters*

Description

update_parameters() update the global parameters used in the TROLL simulation from a TROLL outputs for a next simulation. All parameters have a default value used in French Guiana simulations.

Usage

```
update_parameters(sim, ...)

## S4 method for signature 'trollsim'
update_parameters(sim, ...)
```

Arguments

sim	trollsim.
...	parameters to update and their values (see generate_parameters() for a complete list).

Value

a `data.frame()`

See Also

[troll\(\)](#), [stack\(\)](#), [generate_parameters\(\)](#)

Examples

```
data("TROLLv3_output")
head(update_parameters(TROLLv3_output, iters = 10))
```

Index

- * **datasets**
 - TROLLv3_climatedaytime12, 29
 - TROLLv3_daytimevar, 30
 - TROLLv3_input, 31
 - TROLLv3_output, 31
 - TROLLv3_pointcloud, 32
 - TROLLv3_species, 32

- autogif, 3
- autogif(), 5
- autoplot, trolsim-method, 4, 4

- data.frame(), 9, 14, 15, 33

- generate_climate, 6
- generate_climate(), 30
- generate_lidar, 9
- generate_lidar(), 21, 23, 27, 28, 32
- generate_parameters, 10
- generate_parameters(), 3, 21, 23, 27, 28, 31, 33

- get_chm, 14
- get_chm, trolsim-method (get_chm), 14
- get_forest, 14
- get_forest(), 4, 21, 23, 26–28
- get_forest, trolsim-method (get_forest), 14

- get_log, 15
- get_log(), 26–28
- get_log, trolsim-method (get_log), 15

- lidR::LAS(), 27, 28
- load_output, 16
- load_output(), 17, 27
- load_sim, 17
- load_sim(), 16–18, 21, 23, 26–28
- load_sim, trolsim-method (load_sim), 17
- load_stack, 17
- load_stack(), 16–18, 29
- lutz::tz_lookup_coords(), 6

- nominatimlite::geo_lite_sf(), 6

- option.rcontrol, 3, 18, 21, 23, 24

- parallel::detectCores(), 22
- print, trolsim-method (print.trolsim), 18
- print.trolsim, 18

- rcontrol, 19

- show, trolsim-method (print.trolsim), 18

- stack, 20
- stack(), 5, 10, 13–15, 18, 19, 24, 27–29, 33
- summary, trolsim-method, 5
- summary, trolsim-method (print.trolsim), 18

- troll, 23
- troll(), 5, 10, 13–15, 18, 19, 22, 26, 27, 31, 33

- TROLL.version, 24
- TROLL.version(), 8, 18
- trollCpp, 25
- trolsim, 26
- trolsim(), 5, 14–19, 23, 24, 27, 29, 31
- trolsim-class, 27
- trollstack, 28
- trollstack(), 5, 15–19, 21, 22, 27, 29
- trollstack-class, 29
- TROLLv3_climatedaytime12, 4, 7, 21, 23, 29, 30
- TROLLv3_climatedaytime12(), 31
- TROLLv3_daytimevar, 4, 7, 21, 23, 30, 30
- TROLLv3_daytimevar(), 31
- TROLLv3_input, 3, 21, 23, 31
- TROLLv3_output, 4, 21, 23, 31
- TROLLv3_pointcloud, 32
- TROLLv3_species, 3, 21, 23, 32
- TROLLv3_species(), 31

update_parameters, [33](#)
update_parameters(), [13](#)
update_parameters, trolsim-method
 (update_parameters), [33](#)