

Package: PEcAn.LPJGUESS (via r-universe)

October 19, 2026

Type Package

Title PEcAn Package for Integration of the LPJ-GUESS Model

Version 1.9.0.9000

Description This module provides functions to link LPJ-GUESS to PEcAn.

URL <https://pecanproject.github.io>

BugReports <https://github.com/PecanProject/pecan/issues>

Imports PEcAn.logger, PEcAn.remote, PEcAn.utils, lubridate (>= 1.6.0),
ncdf4 (>= 1.15), Rcpp (>= 1.0.1), tibble, stringr, utils

Suggests testthat (>= 1.0.2)

SystemRequirements LPJ-GUESS model

OS_type unix

License BSD_3_clause + file LICENSE

Copyright Authors

LazyLoad yes

LazyData FALSE

Encoding UTF-8

RoxygenNote 7.3.3

LinkingTo Rcpp

X-schema.org-keywords LPJ-GUESS, ecosystem-modeling

Config/pak/sysreqs libicu-dev libnetcdf-dev libssl-dev libudunits2-dev

Repository <https://pecanproject.r-universe.dev>

Date/Publication 2026-10-19 19:47:56 UTC

RemoteUrl <https://github.com/PecanProject/pecan>

RemoteRef HEAD

RemoteSha c15c509358ff9bad115d5f2ff54cf5ec02234899

RemoteSubdir models/lpjguess

Contents

calculateGridcellSOC	2
calculateGridcellVariablePerPFT	3
find_closing	4
met2model.LPJGUESS	4
model2netcdf.LPJGUESS	5
pecan2lpjguess	6
read_binary_LPJGUESS	6
read_restart.LPJGUESS	7
read_state	8
readStateBinary	8
split_inputs.LPJGUESS	9
update_state_both_LPJGUESS	10
update_state_LPJGUESS	11
write.config.LPJGUESS	13
write.insfile.LPJGUESS	14
write_binary_LPJGUESS	15
write_binary_LPJGUESS_flexible	15
write_restart.LPJGUESS	16
Index	18

calculateGridcellSOC	<i>calculateGridcellSOC</i>
----------------------	-----------------------------

Description

Calculates soil carbon This should be put into the SDA procedure.

Usage

```
calculateGridcellSOC(  
  model.state,  
  pools = c("SOILSTRUCT", "SOILMETA", "SOILMICRO", "SLOWSOM", "PASSIVESOM")  
)
```

Arguments

model.state	A large multiply-nested list containing the entire LPJ-GUESS state as read by function readStateBinary.LPJGUESS
pools	"SOILSTRUCT","SOILMETA","SOILMICRO","SLOWSOM","PASSIVESOM" (see somdynam.cpp)

Value

The total SOC in Gridcell. The unit is consistent with cmass in state, usually kg C m-2

Author(s)

Yinghao Sun

```
calculateGridcellVariablePerPFT
      calculateGridcellVariablePerPFT
```

Description

Calculates a per-PFT, gridcell-summed quantity from the LPJ-GUESS state, correctly averaging over patches. This should be put into the SDA procedure.

Usage

```
calculateGridcellVariablePerPFT(
  model.state,
  variable,
  pft.params,
  min.diam = 5
)
```

Arguments

<code>model.state</code>	A large multiply-nested list containing the entire LPJ-GUESS state as read by function <code>readStateBinary.LPJGUESS</code>
<code>variable</code>	A character string specifying what variable to extract. This can be chosen based on the LPJ-GUESS variable name as recorded in the big list of list (that represents describes the model state in R). Once special case is "biomass" which returns the sum of "cmass_leaf", "cmass_root", "cmass_sap" and "cmass_heart"
<code>pft.params</code>	A data frame containing PFT parameters such as allometric coefficients.
<code>min.diam</code>	Minimum tree diameter (in cm) required for inclusion in calculations.

Value

A numeric vector, with one entry per PFT

Author(s)

Matthew Forrest

find_closing	<i>Find Closing Bracket</i>
--------------	-----------------------------

Description

Identifies the line number of the matching closing bracket for a given opening bracket.

Usage

```
find_closing(find = "}", line_no, file_in, if_else_check = FALSE)
```

Arguments

find	A character string of the opening bracket.
line_no	A numeric value indicating the line number to start the search.
file_in	A character vector of the file content.
if_else_check	Optional. A logical value indicating whether to check for if/else blocks (default is FALSE).

Value

A numeric value indicating the line number of the matching closing bracket.

met2model.LPJGUESS	<i>met2model.LPJGUESS</i>
--------------------	---------------------------

Description

met2model wrapper for LPJ-GUESS

Usage

```
met2model.LPJGUESS(
  in.path,
  in.prefix,
  outfolder,
  start_date,
  end_date,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>in.path</code>	location on disk where inputs are stored
<code>in.prefix</code>	prefix of input and output files
<code>outfolder</code>	location on disk where outputs will be stored
<code>start_date</code>	the start date of the data to be downloaded (will only use the year part of the date)
<code>end_date</code>	the end date of the data to be downloaded (will only use the year part of the date)
<code>overwrite</code>	should existing files be overwritten
<code>verbose</code>	should the function be very verbose
<code>...</code>	additional arguments, currently ignored

Author(s)

Istem Fer

`model2netcdf.LPJGUESS` *Function to convert LPJ-GUESS model output to standard netCDF format*

Description

Convert LPJ-GUESS output to netCDF

Usage

```
model2netcdf.LPJGUESS(outdir, sitelat, sitelon, start_date, end_date)
```

Arguments

<code>outdir</code>	Location of LPJ-GUESS model output
<code>sitelat</code>	Latitude of the site
<code>sitelon</code>	Longitude of the site
<code>start_date</code>	Start time of the simulation
<code>end_date</code>	End time of the simulation

Author(s)

Istem Fer

pecan2lpjguess	<i>Function to translate pecan param names and units to lpjguess names and units</i>
----------------	--

Description

Function to translate pecan param names and units to lpjguess names and units

Usage

```
pecan2lpjguess(trait.values)
```

Arguments

trait.values trait.values, list

Value

translated list

Author(s)

Istem Fer

read_binary_LPJGUESS	<i>Read Binary File for LPJ-GUESS</i>
----------------------	---------------------------------------

Description

Reads a binary file formatted for LPJ-GUESS and extracts relevant data.

Usage

```
read_binary_LPJGUESS(outdir, rundir, version = "PaleON", bak = FALSE)
```

Arguments

outdir	The directory where the .state file is stored (e.g., .../state/1961)
rundir	The directory where the corresponding params.ins is stored (e.g., .../run/ENS-00001)
version	A character string specifying the LPJ-GUESS version (default is "PaleON").
bak	whether read bak.state or not (default is FALSE)

Value

A matrix or list containing the extracted data.

Author(s)

Istem Fer, Yinghao Sun

read_restart.LPJGUESS *Read Restart for LPJGUESS*

Description

Read Restart for LPJGUESS

Usage

```
read_restart.LPJGUESS(outdir, runid, stop.time, settings, var.names, params)
```

Arguments

outdir	output directory
runid	run ID
stop.time	year that is being read
settings	PEcAn settings object
var.names	var.names to be extracted
params	passed on to return value

Value

X_tmp vector of forecasts

Author(s)

Istem Fer, Yinghao Sun

Examples

```
## Not run:
settings <- PEcAn.settings::read.settings("pecan.xml")
params <- list()
rx <- read_restart.LPJGUESS(
  outdir = "/projectnb/. . . /LPJ_output",
  runid = "123456",
  stop.time = as.POSIXct("2001-12-31 23:59:59", tz = "UTC"),
  settings = settings,
  var.names = c("AGB.pft"),
  params = params)

## End(Not run)
```

read_state	<i>Read State File</i>
------------	------------------------

Description

Reads a state file and processes its contents for further use.

Usage

read_state(file_path)

Arguments

file_path A character string specifying the path to the state file.

Value

A list containing the processed state information.

readStateBinary	<i>readStateBinary.LPJGUESS</i>
-----------------	---------------------------------

Description

read binary state file of LPJ-GUESS

Usage

readStateBinary(out.path, npft)

Arguments

out.path location on disk where model run outputs are stored
npft number of pfts specified in instruction file

Value

Patchpft_list state variables common to all individuals of a particular PFT

Author(s)

Istem Fer

`split_inputs.LPJGUESS` *Split inputs for LPJ-GUESS model*

Description

Splits climate met for LPJGUESS

Usage

```
split_inputs.LPJGUESS(  
  settings,  
  start.time,  
  stop.time,  
  inputs,  
  overwrite = FALSE,  
  outpath = NULL  
)
```

Arguments

<code>settings</code>	PEcAn settings object
<code>start.time</code>	start date and time for each SDA ensemble
<code>stop.time</code>	stop date and time for each SDA ensemble
<code>inputs</code>	list of model inputs to use in <code>write.configs.LPJGUESS</code>
<code>overwrite</code>	Default FALSE
<code>outpath</code>	if specified, write output to a new directory. Default NULL writes back to the directory being read

Value

name of the split met file

Author(s)

Istem Fer

```
update_state_both_LPJGUESS
      updateState.LPJGUESS
```

Description

Adjust LPJ-GUESS state variables based on input parameters by nudging density ONLY, with a robust bisection fallback to ensure physical viability.

Usage

```
update_state_both_LPJGUESS(
  model.state,
  pft.params,
  dens.initial,
  dens.target,
  AbvGrndWood.initial,
  AbvGrndWood.target,
  AbvGrndWood.epsilon,
  trace,
  min.diam,
  HEIGHT_MAX = 150
)
```

Arguments

<code>model.state</code>	A large multiply-nested list containing the entire LPJ-GUESS state as read by function <code>readStateBinary.LPJGUESS</code>
<code>pft.params</code>	A data.frame containing the parameters for each PFT. Each row represents one PFT (ordering must be consistent with the vectors below. The names of the columns describe the per-PFT parameter and must include: <code>wooddens</code> , <code>crownarea_max</code> , <code>lifeform</code> (1 = tree, 2 = grass), <code>k_latosa</code> , <code>k_rp</code> , <code>k_allom1</code> , <code>k_allom2</code> , <code>k_allom3</code> , <code>crownarea_max</code> and <code>sla</code> .
<code>dens.initial</code>	A numeric vector of the initial stand-level stem densities (indiv/m^2) as named numeric vector with one entry per PFT/species, with the names being the PFT/species codes.
<code>dens.target</code>	A numeric vector of the target stand-level stem densities (indiv/m^2) as named numeric vector with one entry per PFT/species, with the names being the PFT/species codes.
<code>AbvGrndWood.initial</code>	A numeric vector of the initial stand-level above ground wood (kgC/m^2) as named numeric vector with one entry per PFT/species, with the names being the PFT/species codes.
<code>AbvGrndWood.target</code>	A numeric vector of the target stand-level above ground wood (kgC/m^2) as named numeric vector with one entry per PFT/species, with the names being the PFT/species codes.

AbvGrndWood.epsilon	A single numeric specifying how close the final above ground wood needs to be to the target above ground stem biomass for each individual. eg. 0.05 requires that the final above ground wood is within 5 of the target above ground wood
trace	Logical; if TRUE, prints detailed adjustment process information.
min.diam	Minimum tree diameter (in cm) for inclusion in adjustments.
HEIGHT_MAX	Maximum allowed height of an individual. This is the maximum height that a tree can have. This is hard-coded in LPJ-GUESS to 150 m, but for SDA that might be unrealistically big, so this argument allows adjustment.

Details

Adjust LPJ-GUESS state (both biomass and density)

Value

And updated model state (as a big old list of lists)

Author(s)

Yinghao Sun

update_state_LPJGUESS *updateState.LPJGUESS*

Description

Adjust LPJ-GUESS state variables based on input parameters.

Usage

```
update_state_LPJGUESS(
  model.state,
  pft.params,
  dens.initial,
  dens.target,
  AbvGrndWood.initial,
  AbvGrndWood.target,
  AbvGrndWood.epsilon,
  trace,
  min.diam,
  HEIGHT_MAX = 150
)
```

Arguments

<code>model.state</code>	A large multiply-nested list containing the entire LPJ-GUESS state as read by function <code>readStateBinary.LPJGUESS</code>
<code>pft.params</code>	A data.frame containing the parameters for each PFT. Each row represents one PFT (ordering must be consistent with the vectors below. The names of the columns describe the per-PFT parameter and must include: <code>wooddens</code> , <code>crownarea_max</code> , <code>lifeform</code> (1 = tree, 2 = grass), <code>k_latosa</code> , <code>k_rp</code> , <code>k_allom1</code> , <code>k_allom2</code> , <code>k_allom3</code> , <code>crownarea_max</code> and <code>sla</code> . <code>wooddens</code> , <code>crownarea_max</code> , <code>lifeform</code> (1 = tree, 2 = grass), <code>k_latosa</code> , <code>k_rp</code> , <code>k_allom1</code> , <code>k_allom2</code> , <code>k_allom3</code> , <code>crownarea_max</code> and <code>sla</code> .
<code>dens.initial</code>	A numeric vector of the initial stand-level stem densities (indiv/m ²) as named numeric vector with one entry per PFT/species, with the names being the PFT/species codes. These values should be produced using state data assimilation from function XXXXXX.
<code>dens.target</code>	A numeric vector of the target stand-level stem densities (indiv/m ²) as named numeric vector with one entry per PFT/species, with the names being the PFT/species codes. These values should be produced using state data assimilation from function XXXXXX
<code>AbvGrndWood.initial</code>	A numeric vector of the target stand-level above ground wood (kgC/m ²) as named numeric vector with one entry per PFT/species, with the names being the PFT/species codes. These values should be produced using state data assimilation from function XXXXXX
<code>AbvGrndWood.target</code>	A numeric vector of the target stand-level above ground wood (kgC/m ²) as named numeric vector with one entry per PFT/species, with the names being the PFT/species codes. These values should be produced using state data assimilation from function XXXXXX
<code>AbvGrndWood.epsilon</code>	A single numeric specifying how close the final above ground wood needs to be to the target above ground stem biomass for each individual. eg. 0.05 requires that the final above ground wood is within 5 of the target above ground wood
<code>trace</code>	Logical; if TRUE, prints detailed adjustment process information.
<code>min.diam</code>	Minimum tree diameter (in cm) for inclusion in adjustments.
<code>HEIGHT_MAX</code>	Maximum allowed height of an individual. This is the maximum height that a tree can have. This is hard-coded in LPJ-GUESS to 150 m, but for SDA that might be unrealistically big, so this argument allows adjustment.

Details

Adjust LPJ-GUESS state

Value

And updated model state (as a big old list o' lists)

Author(s)

Matthew Forrest, Yinghao Sun

write.config.LPJGUESS *Write LPJ-GUESS configuration files*

Description

Writes a LPJ-GUESS config file.

Usage

```
write.config.LPJGUESS(defaults, trait.values, settings, run.id, restart = NULL)
```

Arguments

defaults	list of defaults to process
trait.values	vector of samples for a given trait
settings	list of settings from pecan settings file
run.id	id of run
restart	Logical, whether to restart the simulation.

Details

Requires a pft xml object, a list of trait values for a single model run, and the name of the file to create

Value

configuration file for LPJ-GUESS for given run

Author(s)

Istem Fer, Tony Gardella

```
write.insfile.LPJGUESS
```

Write LPJ-GUESS instruction script

Description

Write LPJ-GUESS instruction script

Usage

```
write.insfile.LPJGUESS(  
  settings,  
  trait.values,  
  rundir,  
  outdir,  
  run.id,  
  restart = NULL  
)
```

Arguments

settings	PEcAn settings list
trait.values	trait.values
rundir	rundir
outdir	outdir
run.id	PEcAn run ID
restart	A list object passed from the SDA workflow containing run-specific restart information, or NULL for initial runs.

Value

settings Updated list

Author(s)

Istem Fer, Yinghao Sun

write_binary_LPJGUESS *Write updated variables into a copy of the original LPJ-GUESS .state file*

Description

Write updated variables into a copy of the original LPJ-GUESS .state file

Usage

```
write_binary_LPJGUESS(State_updated, outdir)
```

Arguments

State_updated	A list containing updated state variables, position list and size list (get from read_binary)
outdir	Path to a directory containing the '0.state' and 'meta.bin' files.

Value

No return value. Writes files to disk as side effect.

Author(s)

Yinghao Sun

write_binary_LPJGUESS_flexible
Write LPJ-GUESS binary state allowing cohort count changes

Description

A robust writer for LPJ-GUESS '0.state' that supports changes in the number of cohorts/individuals (i.e., variable-length Individuals blocks). When no cohort count changes are detected, it can fall back to a fast in-place writer (e.g., your existing 'write_binary_LPJGUESS()' that uses stored off-sets).

Usage

```
write_binary_LPJGUESS_flexible(
  State_updated,
  outdir,
  use_fast_inplace_if_possible = TRUE,
  verbose = FALSE
)
```

Arguments

State_updated	A list containing: - 'state': updated Gridcell object (parsed state tree) - 'pos_list': named list/vector of byte offsets for each key - 'siz_list': named list/vector of byte sizes per element for each key
outdir	Character scalar. Directory containing '0.state' and 'meta.bin'.
use_fast_inplace_if_possible	Logical. If 'TRUE', uses the existing in-place writer when no 'number_of_individuals' changes are detected.
verbose	Logical. If 'TRUE', prints minimal progress messages.

Details

If cohort count changes are detected, this function rebuilds a new '0.state' by streaming-replay: it copies unchanged raw bytes from the original state, writes updated fields, and fully rewrites the Individuals blocks for affected stand/patches so downstream offsets remain correct.

Value

Invisibly returns 'TRUE' on success. Side effect: - Creates 'bak.state' as a backup of the original '0.state' - Writes a new '0.state' (possibly via '0.state.tmp' and rename)

Author(s)

Yinghao Sun

write_restart.LPJGUESS

write_restart.LPJGUESS

Description

Write restart files for LPJGUESS new.state includes X (AGB.pft) from Analysis new.params includes LPJGUESS_state

Usage

```
write_restart.LPJGUESS(
  outdir,
  runid,
  start.time,
  stop.time,
  settings,
  new.state,
  new.params,
  inputs = NULL,
  RENAME = TRUE,
  verbose = FALSE
)
```

Arguments

outdir	output directory
runid	run ID
start.time	start date and time for each SDA ensemble
stop.time	stop date and time for each SDA ensemble
settings	PEcAn settings object
new.state	analysis state vector
new.params	list of parameters to convert between different states
inputs	list of model inputs to use in write.configs.LPJGUESS
RENAME	flag to either rename output file or not
verbose	decide if we want to print the runid

Value

NONE

Author(s)

Yinghao Sun

Index

`calculateGridcellSOC`, [2](#)
`calculateGridcellVariablePerPFT`, [3](#)

`find_closing`, [4](#)

`met2model.LPJGUESS`, [4](#)
`model2netcdf.LPJGUESS`, [5](#)

`pecan2lpjguess`, [6](#)

`read_binary_LPJGUESS`, [6](#)
`read_restart.LPJGUESS`, [7](#)
`read_state`, [8](#)
`readStateBinary`, [8](#)

`split_inputs.LPJGUESS`, [9](#)

`update_state_both_LPJGUESS`, [10](#)
`update_state_LPJGUESS`, [11](#)

`write.config.LPJGUESS`, [13](#)
`write.insfile.LPJGUESS`, [14](#)
`write_binary_LPJGUESS`, [15](#)
`write_binary_LPJGUESS_flexible`, [15](#)
`write_restart.LPJGUESS`, [16](#)