

Package: LandR (via r-universe)

October 13, 2026

Type Package

Title Landscape Ecosystem Modelling in R

Description Utilities for 'LandR' suite of landscape simulation models. These models simulate forest vegetation dynamics based on LANDIS-II, and incorporate fire and insect disturbance, as well as other important ecological processes. Models are implemented as 'SpaDES' modules.

URL <https://landr.predictiveecology.org>,
<https://github.com/PredictiveEcology/LandR>

Date 2026-09-18

Version 1.2.0.9055

Depends R (>= 4.3)

Imports cli, data.table, Formula, fpCompare, ggplot2, ggspatial, grDevices, httr2, lme4, methods, MuMIn (>= 1.42.5), parallel, patchwork, pemisc (>= 0.0.4.9011), raster, RColorBrewer, Rcpp (>= 1.0.10), reproducible (>= 3.2.1.9037), sf, sp, SpaDES.tools (>= 2.1.1), stats, terra, tidyterra, tools, utils, viridis

LinkingTo Rcpp

Suggests arrow, bbmle, biomod2 (>= 4.2-2), BioSIM (>= 1.1.0), codetools, covr, curl (>= 6.4.0), digest, dplyr, fasterize, future, future.apply, geodata, ggalluvial, ggrepel, googledrive (>= 2.0.0), httr, knitr, lqmm, map, MASS, memuse, parallelly, purrr, qs2, rasterVis, RCurl, RhpcBLASctl, rlang, rmarkdown, roxygen2, R.utils, SpaDES.core, spelling, testthat, withr, XML

Remotes CeresBarros/biomod2@dev_currentCeres, RNCAN/BioSimClient_R, PredictiveEcology/map@development, PredictiveEcology/pemisc@development, PredictiveEcology/reproducible@development, PredictiveEcology/SpaDES.core@development, PredictiveEcology/SpaDES.tools@development

Encoding UTF-8

VignetteBuilder knitr, rmarkdown

Language en-CA

License GPL-3

BugReports <https://github.com/PredictiveEcology/LandR/issues>

ByteCompile yes

LazyData true

Collate 'assertions.R' 'biomodWrappers.R' 'biosim.R' 'chooseName.R'
'cohorts.R' 'convertUnwantedLCC2.R' 'dem.R' 'disturbances.R'
'driveFallback.R' 'ecoregions.R' 'ecozones.R' 'fireSeverity.R'
'forestLand.R' 'imports.R' 'LANDIS-II-tables.R'
'LandR-package.R' 'lccMapGenerator.R' 'leadingProportions.R'
'legacyToTo.R' 'light-shade.R' 'makeDummyInputs.R' 'maps.R'
'maxBestimation.R' 'misc.R' 'nullClimate.R' 'options.R'
'overlayLCCs.R' 'partitionBiomass.R' 'plot-summaries.R'
'plot-transitions.R' 'plotting.R' 'prepEcoregions.R'
'prepInputObjects.R' 'prepInputs_NTEMS.R'
'prepInputs_SCANFI_structure.R' 'prepSpeciesLayers.R'
'RcppExports.R' 'scanfi-access.R' 'scanfiMirror.R'
'seedDispersalLANDIS.R' 'serotiny-resprouting.R'
'speciesPresentFromKNN.R' 'speciesTable.R'
'sppEquivalencies_CA.R' 'sppHarmonize.R' 'studyArea.R'
'subsetDataSize.R' 'SummarizeCanadianForestRasters.R'
'terra-raster-utils.R' 'vignette-artifacts.R' 'wetland.R'
'wildfire.R' 'zzz.R'

Roxygen list(markdown = TRUE)

Config/testthat/edition 3

Config/testthat/parallel true

Config/testthat/start-first prep*

Config/roxygen2/version 8.1.0

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev make default-jdk libicu-dev libjpeg-dev libpng-dev libuv1-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

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

Date/Publication 2026-10-13 22:04:07 UTC

RemoteUrl <https://github.com/PredictiveEcology/LandR>

RemoteRef development

RemoteSha 8efd49a194759f375683c1a81524dc187345e4e2

Contents

LandR-package	6
.compareRas	7
.fitNLMwCovariates	7
.getMaxBCoefs	9

.interpolateLightProb	9
.MLLMaxBPartialPlotData	10
.MLLMaxBplotData	11
.NvalueFlag	12
addNoPixel2CohortData	13
addPixels2CohortData	14
adjustAgeToLongevity	14
aggregateRasByDT	15
asInteger	16
assert1	16
assertPostPartialDist	21
assignLightProb	22
biomodEnsembleFrstWrapper	23
biomodEnsembleProjMaps	23
biomodEnsembleWrapper	24
biomodModelingWrapper	24
biomodProjWrapper	25
BioSIM_extractPoints	26
BioSIM_getMPBSLR	27
BioSIM_getWindAnnual	27
BioSIM_getWindMonthly	28
calc_raster_counts	29
calcSeverityB	32
calcSiteShade	33
calculateClimateEffect	34
CASFRItToSpRasts	34
checkSpeciesTraits	35
cohortDefinitionCols	37
Colors	37
columnsForPixelGroups	38
convert_SCANFI_LCC_codes	38
convertUnwantedLCC	39
convertUnwantedLCC2	42
defineFlammable	43
describeCohortData	43
doResprouting	44
doSerotiny	46
dropTerm	48
ecoregionProducer	48
equivalentName	49
extractMaxB	50
fasterizeFromSp	50
FireDisturbance	51
fitNLMModels	57
forestLandMask	59
gadm_canada	60
generatePixelGroups	61
genericExtract	61

genPGsPostDisturbance	62
getSpeciesTable	63
getWildfire_NFI	65
ggplotMLL_maxB	66
imputeBadAgeModelDefault	67
installBioSIM	68
LANDISDisp	68
LandROptions	72
lccMapGenerator	73
leadingByPixelGroup	75
loadCASFRI	76
loadkNNSpeciesLayers	77
loadSCANFISpeciesLayers	78
makeAndCleanInitialCohortData	80
makeBiomassMap	81
makeCohortDataFiles	82
makeDummyEcoregionMap	83
makeEcoregionDT	84
makeEcoregionMap	85
makeMinRelativeB	86
makePickellStack	86
makePixelGroupMap	87
makePixelTable	87
makeSpeciesEcoregion	88
makeVegTypeMap	90
minRelativeBDefaults	92
mixedwoodProp	92
modifySpeciesAndSpeciesEcoregionTable	93
NACover2zero	94
nonForestedPixels	95
overlayLCCs	95
overlayStacks	97
partialggplotMLL_maxB	98
partitionBiomass	99
pixelFate	100
plantNewCohorts	100
plot_raster	101
plotLeadingSpecies	102
plotSpatial	103
plotVTM	103
prepEcoregions	104
prepEcozonesRst	105
prepInputs_CWIM	106
prepInputs_FAO_forest	107
prepInputs_NTEMS_DominantSpecies	108
prepInputs_NTEMS_LCC_FAO	109
prepInputs_NTEMS_Nonforest	110
prepInputs_SCANFI_LCC_FAO	110

prepInputs_SCANFI_structure	111
prepInputsCanDEM	112
prepInputsEcoregion	113
prepInputsFireYear	113
prepInputsLCC	114
prepInputsSpecies	115
prepInputsSpeciesEcoregion	116
prepInputsStandAgeMap	117
prepRasterToMatch	119
prepRawBiomassMap	120
prepSpeciesLayers_KNN	121
randomStudyArea	125
replaceAgeInFires	126
rescale	127
rmMissingCohorts	127
scanfiUrlRemap	128
scheduleDisturbance	128
speciesEcoregionLatestYear	129
speciesEcoregionStack	130
speciesInStudyArea	130
speciesPresentFromKNN	132
speciesPresentFromNTEMS	133
speciesPresentFromSCANFI	134
speciesTableUpdate	135
sppColors	138
sppEquivalencies_CA	139
sppEquivCheck	142
sppHarmonize	143
standAgeMapGenerator	144
statsModel	145
studyAreaEco	145
subsetDataSize	146
subsetDT	147
sumRastersBySpecies	148
uniqueCohortDefinition	148
updateCohortData	149
updateCohortDataPostHarvest	151
updateSpeciesTable	153
vegTypeGenerator	153
vegTypeMapGenerator	155
vtm2conifdecid	156
WardKernel	158
wetlandToLCC	158

LandR-package

LandR *package*

Description

Utilities for 'LandR' suite of landscape simulation models. These models simulate forest vegetation dynamics based on LANDIS-II, and incorporate fire and insect disturbance, as well as other important ecological processes. Models are implemented as 'SpaDES' modules.

Package options

LandR packages use `options()` to configure behaviour. See `LandROptions()` for the options and their defaults.

Author(s)

Maintainer: Eliot J B McIntire <eliot.mcintire@nrcan-rncan.gc.ca> ([ORCID](#))

Authors:

- Eliot J B McIntire <eliot.mcintire@nrcan-rncan.gc.ca> ([ORCID](#))
- Alex M Chubaty <achubaty@for-cast.ca> ([ORCID](#))
- Ceres Barros <ceres.barros@ubc.ca> ([ORCID](#))
- Ian M S Eddy <ian.eddy@nrcan-rncan.gc.ca> ([ORCID](#))

Other contributors:

- His Majesty the King in Right of Canada, as represented by the Minister of Natural Resources Canada [copyright holder]

See Also

Useful links:

- <https://landr.predictiveecology.org>
- <https://github.com/PredictiveEcology/LandR>
- Report bugs at <https://github.com/PredictiveEcology/LandR/issues>

.compareRas	<i>Compare raster properties</i>
-------------	----------------------------------

Description

TODO: Move to reproducible

TODO: Move to reproducible TODO: expand to multiple objects

Usage

```
.compareRas(x, ...)
```

```
.compareCRS(x, y)
```

Arguments

x, y a Raster, SpatRaster, sf, SpatVector, or Spatial object

... additional Raster or SpatRaster objects, and arguments passed to [terra::compareGeom\(\)](#).

Value

the projected extent

Note

this function internally converts Raster to SpatRaster to allow using `compareGeom()`, and benefit from its complexity.

.fitNLMwCovariates	<i>Fit non-linear growth model under various starting conditions</i>
--------------------	--

Description

Uses likelihood parameter estimation to fit non linear models while attempting several starting values.

Usage

```
.fitNLMwCovariates(  
  data,  
  nonLinModelQuoted,  
  linModelQuoted,  
  mllsOuterPrev,  
  model = c("CR", "Logistic"),  
  maxCover = 1L,
```

```

    starts = NULL,
    lower = NULL,
    upper = NULL,
    nbWorkers = 1L
  )

```

Arguments

<code>data</code>	a <code>data.table</code> or <code>data.frame</code> with all covariates and the response variable. Note that incomplete lines are removed.
<code>nonLinModelQuoted</code>	The non-linear equation as a call (quoted expression) passed to <code>bbmle::mle2(minuslog1)</code> . See <code>?mle</code> . Accepts equations with three parameters 'A', 'p' and 'k'.
<code>linModelQuoted</code>	A list of linear equations/modes relating each parameter ('A', 'p' and 'k') with a set of covariates. A call (quoted expression) passed to <code>mle2(..., parameters)</code> . Note that for the purpose of tree growth, the linear equation determining 'A' should include a 'cover' predictor indicating the tree cover or dominance in the stand. Should be scaled between 0 and <code>maxCover</code> .
<code>mllsOuterPrev</code>	the output of a previous <code>fitNLMwCovariates</code> run which is used to extract last best AIC and maximum biomass estimate and judge if new iterations are better.
<code>model</code>	character. Non-linear model form used to estimate average maximum biomass. One of "CR" (Chapman-Richards) or "Logistic". In both cases, maximum biomass is equivalent to the 'A' asymptote parameter, which is estimated using observed mean values of predictors entering its linear equation and <code>cover == maxCover</code> , if this predictor is included (as it should). Passed to <code>extractMaxB</code>
<code>maxCover</code>	numeric. Value indicating maximum cover/dominance.
<code>starts</code>	<code>data.table</code> or <code>data.frame</code> of parameter starting values. Will be coerced to named list with names being parameter names.
<code>lower</code>	passed to bbmle::mle2
<code>upper</code>	passed to bbmle::mle2
<code>nbWorkers</code>	integer. If > 1, the number of workers to use in <code>parallelly::makeClusterPSOCK(nbWorkers = .)</code> , otherwise no parallelisation is done.

Value

a list with entries `mll` (the maximum likelihood-estimated coefficients) and `AICbest` (the AIC of the best models generating these coefficients)

See Also

[bbmle::mle2\(\)](#)

<code>.getMaxBCoefs</code>	<i>Get maximum biomass coefficient names</i>
----------------------------	--

Description

Extracts the names of linear coefficients for the maximum-biomass-equivalent parameter in the non-linear growth equations

Usage

```
.getMaxBCoefs(m11, model = c("CR", "Logistic"))
```

Arguments

<code>m11</code>	the output of an <code>bbmle::mle2</code> call (the fitted non-linear model), from which coefficient values will be extracted
<code>model</code>	character. Non-linear model form used to estimate average maximum biomass. One of "CR" (Chapman-Richards) or "Logistic".

Value

a list of two vectors of parameter names one following coefficient names in `m11` (`'m11CoefNames'`), the other using the original names as in the data used for model fitting (`'origCoefNames'`)

<code>.interpolateLightProb</code>	<i>Find interpolated value of light probability</i>
------------------------------------	---

Description

Find interpolated value of light probability

Usage

```
.interpolateLightProb(x, x0, x1, y0, y1)
```

Arguments

<code>x</code>	the species shade tolerance trait value for which we want to find the interpolated probability.
<code>x0</code>	the floor of <code>x</code> corresponding to a class of shade tolerance in the <code>sufficientLight</code> table.
<code>x1</code>	the ceiling of <code>x</code> corresponding to a class of shade tolerance in the <code>sufficientLight</code> table.
<code>y0</code>	the probability of germination in the <code>sufficientLight</code> table corresponding to <code>x0</code> .
<code>y1</code>	the probability of germination in the <code>sufficientLight</code> table corresponding to <code>x1</code> .

Value

vector of the interpolated value

.MLLMaxBPartialPlotData

Prepare data for model plotting

Description

Prepare data for model plotting

Usage

```
.MLLMaxBPartialPlotData(
  mll,
  nonLinModelQuoted,
  linModelQuoted,
  targetCovar = "cover",
  fixMaxCover = TRUE,
  maxCover = 1,
  data,
  fun = "mean",
  plotCIs = TRUE
)
```

Arguments

<code>mll</code>	outputs of an <code>bbmle::mle2</code> call (the fitted non-linear model), from which coefficient values will be extracted.
<code>nonLinModelQuoted</code>	The non-linear equation as a call (quoted expression) passed to <code>mle2(minuslog1)</code> . See <code>?mle</code> . Accepts equations with three parameters 'A', 'p' and 'k'.
<code>linModelQuoted</code>	A list of linear equations/modes relating each parameter ('A', 'p' and 'k') with a set of covariates. A call (quoted expression) passed to <code>mle2(..., parameters)</code> . Note that for the purpose of tree growth, the linear equation determining 'A' should include a 'cover' predictor indicating the tree cover or dominance in the stand. Should be scaled between 0 and <code>maxCover</code> .
<code>targetCovar</code>	the covariate for which variation in <code>maxB</code> values will be shown. Defaults to showing how <code>maxB</code> values change with "cover". All other covariates except "age" are averaged. Age values are generated as <code>round(seq(min(age), max(age)*1.5, length.out = 100), 0)</code> . When <code>targetCovar != "cover"</code> , "cover" will be fixed at <code>maxCover</code> . See <code>fixMaxCover</code> .
<code>fixMaxCover</code>	logical. If TRUE and <code>targetCovar != "cover"</code> , cover is not averaged and is fixed to <code>maxCover</code> .
<code>maxCover</code>	numeric. Value indicating maximum cover/dominance.

<code>data</code>	data for estimation of maximum biomass. Should contain at least an 'age' column. Note that other covariates will be averaged and 'cover' values will be replaced with the maximum cover value (<code>maxCover</code>).
<code>fun</code>	The function to apply when summarizing other variables. By default, the all other variables except age are averaged ("mean"). Other options are: "median", "min", "max".
<code>plotCIs</code>	should confidence intervals be calculated and plotted?

See Also

`bbmle:mle2()`

<code>.MLLMaxBplotData</code>	<i>Prepare data for model plotting</i>
-------------------------------	--

Description

Prepare data for model plotting

Usage

```
.MLLMaxBplotData(
  mll,
  nonLinModelQuoted,
  linModelQuoted,
  maxCover,
  data,
  averageCovariates = TRUE,
  observedAge = FALSE,
  plotCIs = TRUE
)
```

Arguments

<code>mll</code>	outputs of an <code>bbmle:mle2</code> call (the fitted non-linear model), from which coefficient values will be extracted.
<code>nonLinModelQuoted</code>	The non-linear equation as a call (quoted expression) passed to <code>mle2(minuslog1)</code> . See <code>?mle</code> . Accepts equations with three parameters 'A', 'p' and 'k'.
<code>linModelQuoted</code>	A list of linear equations/modes relating each parameter ('A', 'p' and 'k') with a set of covariates. A call (quoted expression) passed to <code>mle2(..., parameters)</code> . Note that for the purpose of tree growth, the linear equation determining 'A' should include a 'cover' predictor indicating the tree cover or dominance in the stand. Should be scaled between 0 and <code>maxCover</code> .
<code>maxCover</code>	numeric. Value indicating maximum cover/dominance.

<code>data</code>	data for estimation of maximum biomass. Should contain at least an 'age' column. Note that other covariates will be averaged and 'cover' values will be replaced with the maximum cover value (<code>maxCover</code>).
<code>averageCovariates</code>	should covariates other than age/cover be averaged for biomass predictions? If not, for each age (at maximum cover) there will be as many predictions as other covariate values. If <code>observedAge == TRUE</code> and <code>averageCovariates == FALSE</code> then the original data is used, with cover changed to <code>maxCover</code> .
<code>observedAge</code>	should observed age values be used, or should these be generated as <code>round(seq(min(age), max(age)*1.5, length.out = 100), 0)</code> ? If <code>observedAge == TRUE</code> and <code>averageCovariates == FALSE</code> then the original data is used, with cover changed to <code>maxCover</code> .
<code>plotCIs</code>	should confidence intervals be calculated and plotted?

See Also

[bbmle::mle2\(\)](#)

<code>.NAvalueFlag</code>	<i>Set NA values in Raster or SpatRaster</i>
---------------------------	--

Description

These all create a single function that can be used for either Raster or SpatRaster objects.

Usage

```
.NAvalueFlag(ras, NAval)

.stack(rasList)

.projectExtent(ras, crs)

asInt(ras)

isInt(ras)

reclass(ras, tab)
```

Arguments

<code>ras</code>	a Raster, or SpatRaster object
<code>NAval</code>	the value to use as NA
<code>rasList</code>	a list of Raster or SpatRaster objects
<code>crs</code>	passed to <code>raster::projectRaster(..., crs = crs)</code> and <code>terra::project(..., y = crs)</code>
<code>tab</code>	matrix of values to reclassify. See <code>terra::classify</code> and <code>raster::reclassify</code> .

Value

a raster with attributed NA values

a stacked raster

the projected extent

asInt returns a *Raster with values converted to integer, if they weren't already.

isInt returns a logical as per is.integer.

reclass returns a *Raster with values reclassified as per terra::classify and raster::reclassify.

addNoPixel2CohortData	<i>Add number of pixels per pixelGroup and add it has a new column to cohortData</i>
-----------------------	--

Description

Add number of pixels per pixelGroup and add it has a new column to cohortData

Usage

```
addNoPixel2CohortData(
  cohortData,
  pixelGroupMap,
  cohortDefinitionCols = LandR::cohortDefinitionCols(),
  doAssertion = getOption("LandR.assertions", TRUE)
)
```

Arguments

cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPPAct, and sumB.
pixelGroupMap	A RasterLayer with pixel values equal to a pixel group number that corresponds exactly to pixelGroup column in cohortData.
cohortDefinitionCols	the columns in cohortData that define unique cohorts (default: c("pixelGroup", "speciesCode", "age")).
doAssertion	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: getOption("LandR.assertions", TRUE).

Value

An cohortData dat.table with a new noPixels column

`addPixels2CohortData` *Generate cohortData table per pixel:*

Description

Generate cohortData table per pixel:

Usage

```
addPixels2CohortData(
  cohortData,
  pixelGroupMap,
  cohortDefinitionCols = LandR::cohortDefinitionCols(),
  doAssertion = getOption("LandR.assertions", TRUE)
)
```

Arguments

<code>cohortData</code>	A <code>data.table</code> with columns: <code>pixelGroup</code> , <code>ecoregionGroup</code> , <code>speciesCode</code> , and optionally <code>age</code> , <code>B</code> , <code>mortality</code> , <code>aNPPAct</code> , and <code>sumB</code> .
<code>pixelGroupMap</code>	A <code>RasterLayer</code> with pixel values equal to a pixel group number that corresponds exactly to <code>pixelGroup</code> column in <code>cohortData</code> .
<code>cohortDefinitionCols</code>	the columns in <code>cohortData</code> that define unique cohorts (default: <code>c("pixelGroup", "speciesCode", "age")</code>).
<code>doAssertion</code>	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: <code>getOption("LandR.assertions", TRUE)</code> .

Value

An expanded `cohortData` `data.table` with a new `pixelIndex` column.

`adjustAgeToLongevity` *Reduces the age of cohorts that exceed their longevity x adjustmentFactor*

Description

Reduces the age of cohorts that exceed their longevity x `adjustmentFactor`

Usage

```
adjustAgeToLongevity(pixelCohortData, longevity, adjustmentFactor)
```

Arguments

pixelCohortData	The full cohortData data.table
longevity	A data.table with the longevity of each species.
adjustmentFactor	A numeric controlling the proportion of species longevity that cohort ages cannot exceed.

Value

A cohortData data.table with corrected ages.

aggregateRasByDT	<i>Aggregate a raster</i>
------------------	---------------------------

Description

Uses **data.table** to perform aggregation calculations, which is faster than `raster::aggregate`.

Usage

```
aggregateRasByDT(ras, newRas, fn = sum)
```

Arguments

ras	RasterLayer to aggregate
newRas	RasterLayer to match
fn	function to use to aggregate pixel values

Value

RasterLayer

asInteger

Convert numeric values to rounded integers

Description

Essentially a wrapper around round, rather than truncate, which is what as.integer does. Internally, this is simply as.integer(floor(x + 0.5)).

Usage

```
asInteger(x)
```

Arguments

x A numeric vector

Value

An integer vector of length x, rounded to zero decimal places prior to as.integer

Note

Values ending in .5 will be rounded up, whether positive or negative. This is different than round.

Examples

```
x <- seq(-2, 2, 0.25)
data.frame(dbl = x, int = asInteger(x))
```

assert1

Assertions

Description

- assert1: Assert that ecoregionCodes that were replaced, were correctly identified;

This is the full pixelCohortData, not the collapsed one.

Usage

```
assert1(
  cohortData34to36,
  cohortData,
  rmZeroBiomassQuote,
  classesToReplace = 34:36,
  doAssertion = getOption("LandR.assertions", TRUE)
)
```

```
assert2(  
  cohortDataNo34to36,  
  classesToReplace = 34:36,  
  doAssertion = getOption("LandR.assertions", TRUE)  
)  
  
assertSppMaxBMaxANPP(  
  speciesEcoregion,  
  doAssertion = getOption("LandR.assertions", TRUE)  
)  
  
assertUniqueCohortData(  
  cohortData,  
  columns,  
  doAssertion = getOption("LandR.assertions", TRUE)  
)  
  
assertERGs(  
  ecoregionMap,  
  cohortData,  
  speciesEcoregion,  
  minRelativeB,  
  doAssertion = getOption("LandR.assertions", TRUE)  
)  
  
assertColumns(  
  obj,  
  colClasses,  
  doAssertion = getOption("LandR.assertions", TRUE)  
)  
  
assertCohortData(  
  cohortData,  
  pixelGroupMap,  
  maxExpectedNumDiverge = 1,  
  message = "",  
  doAssertion = getOption("LandR.assertions", TRUE),  
  verbose = getOption("LandR.verbose", TRUE),  
  cohortDefinitionCols = LandR::cohortDefinitionCols()  
)  
  
assertCohortDataERG(  
  cohortData,  
  doAssertion = getOption("LandR.assertions", TRUE)  
)  
  
assertPixelCohortData(  
  cohortData,
```

```
    pixelCohortData,  
    pixelGroupMap,  
    doAssertion = getOption("LandR.assertions", TRUE)  
  )  
  
  assertSpeciesPlotLabels(  
    speciesNames,  
    sppEquiv,  
    doAssertion = getOption("LandR.assertions", TRUE)  
  )  
  
  assertFireToleranceDif(  
    burnedPixelCohortData,  
    doAssertion = getOption("LandR.assertions", TRUE)  
  )  
  
  assertPostFireDist(  
    cohortDataOrig,  
    pixelGroupMapOrig,  
    cohortDataNew,  
    pixelGroupMapNew,  
    postFirePixelCohortData,  
    burnedPixelCohortData,  
    doAssertion = getOption("LandR.assertions", TRUE)  
  )  
  
  assertSpeciesLayers(  
    speciesLayers,  
    thresh,  
    doAssertion = getOption("LandR.assertions", TRUE)  
  )  
  
  assertRstLCChange(  
    rstLCChange,  
    rasterToMatch,  
    doAssertion = getOption("LandR.assertions", TRUE)  
  )  
  
  assertSpeciesEcoregionCohortDataMatch(  
    cohortData,  
    speciesEcoregion,  
    doAssertion = getOption("LandR.assertions", TRUE)  
  )  
  
  assertPixelCohortDataValid(  
    standCohortData,  
    doAssertion = getOption("LandR.assertions", TRUE)  
  )
```

```

assertRepsAllCohortData(
  allCohortData,
  reps,
  years,
  doAssertion = getOption("LandR.assertions", TRUE)
)

assertStandAgeMapAttr(
  standAgeMap,
  doAssertion = getOption("LandR.assertions", TRUE)
)

assertCohortDataAttr(
  cohortData,
  doAssertion = getOption("LandR.assertions", TRUE)
)

assertSppVectors(
  sppEquiv = NULL,
  sppNameVector = NULL,
  sppColorVect = NULL,
  sppEquivCol = NULL,
  doAssertion = getOption("LandR.assertions", TRUE)
)

assertSpeciesTableRaw(
  speciesTableRaw,
  doAssertion = getOption("LandR.assertions", TRUE)
)

assertSpeciesTable(
  speciesTable,
  doAssertion = getOption("LandR.assertions", TRUE)
)

```

Arguments

cohortData34to36	A cohortData data.table with only the pixels what were LCC 34:36
cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPPAct, and sumB.
rmZeroBiomassQuote	An expression to evaluate, in the form of quote(B>0), used to select cohorts with biomass.
classesToReplace	Integer vector of classes that are to be replaced, e.g., 34, 35, 36 on LCC2005, which are burned young, burned 10 year, and cities.

<code>doAssertion</code>	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: <code>getOption("LandR.assertions", TRUE)</code> .
<code>cohortDataNo34to36</code>	A <code>cohortData</code> <code>data.table</code> with only the pixels what were LCC 34:36
<code>speciesEcoregion</code>	A <code>data.table</code> with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: <code>speciesCode</code> and <code>ecoregionGroup</code> , character representation of species and ecoregion groups respectively, <code>maxB</code> the maximum biomass for the species in a given 'ecoregion', <code>maxANPP</code> the maximum aboveground net primary productivity and <code>SEP</code> the species establishment probability. May contain columns <code>inflationFactor</code> (used to adjust <code>maxB</code>) and <code>maxANPPproportion</code> (used to calculate <code>maxANPP</code>).
<code>columns</code>	Vector of column names on which to test for unique <code>cohortData</code>
<code>ecoregionMap</code>	The <code>ecoregionMap</code> , a raster of all the unique ecoregion groupings, which group together pixels of similar biophysical characteristics.
<code>minRelativeB</code>	a <code>data.frame</code> with the cut points to classify stand shadiness.
<code>obj</code>	A <code>data.frame</code> - or <code>data.table</code> -like object
<code>colClasses</code>	A named vector of column classes, where the names are the column names
<code>pixelGroupMap</code>	A <code>RasterLayer</code> with pixel values equal to a pixel group number that corresponds exactly to <code>pixelGroup</code> column in <code>cohortData</code> .
<code>maxExpectedNumDiverge</code>	A numeric, length 1, indicating by how many they can diverge. Default 1.
<code>message</code>	An optional message to print. This may help identify where this function was called.
<code>verbose</code>	Controls message output. Defaults to <code>getOption("LandR.verbose")</code>
<code>cohortDefinitionCols</code>	the columns in <code>cohortData</code> that define unique cohorts (default: <code>c("pixelGroup", "speciesCode", "age")</code>).
<code>pixelCohortData</code>	The full <code>cohortData</code> <code>data.table</code>
<code>speciesNames</code>	A vector of species names for which the labels will be checked
<code>sppEquiv</code>	an table with a column containing species names
<code>burnedPixelCohortData</code>	<code>cohortData</code> -like object containing all dead, surviving and new cohorts (i.e. activated by serotiny/resprouting)
<code>cohortDataOrig</code>	object used as <code>updateCohortData(cohortData = .)</code>
<code>pixelGroupMapOrig</code>	object used as <code>updateCohortData(pixelGroupMap = .)</code>
<code>cohortDataNew</code>	<code>cohortData</code> output from <code>updateCohortData</code>
<code>pixelGroupMapNew</code>	<code>pixelGroupMap</code> output from <code>updateCohortData</code>

postFirePixelCohortData	object used as <code>updateCohortData(newPixelCohortData = .)</code>
speciesLayers	A <code>RasterStack</code> or <code>RasterLayer</code> that should contain species cover data in the study area
thresh	the minimum number of pixels where the species must have biomass > 0 to be considered present in the study area. Defaults to 1.
rstLCChange	a raster layer indicating pixels where land-use change occurred as 1s
rasterToMatch	A <code>SpatRaster</code> object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). <code>rasterToMatch</code> should correspond to <code>studyArea</code> if used.
standCohortData	A <code>data.table</code> with simulated and observed stand data for validation
allCohortData	A <code>data.table</code> with all simulated cohortData to use for validation
reps	repetition ids
years	years
standAgeMap	A <code>rasterLayer</code> of stand ages with attribute "imputedPixID"
sppNameVector	A character vector of species to use. These species must all be from one naming convention, i.e., from one column in the <code>sppEquiv</code> .
sppColorVect	A named vector of colours to use for plotting. The names must conform with species name convention used (see sppEquivalencies_CA for possible naming conventions) and should also contain a colour for 'Mixed', when a mixed forest type is supported (see <code>vegLeadingProportion</code> parameter in vegTypeGenerator() for details on mixed forest types).
sppEquivCol	the column name to use from <code>sppEquiv</code> .
speciesTableRaw	raw species traits data.table (see getSpeciesTable())
speciesTable	species traits data.table (see prepSpeciesTable())

assertPostPartialDist *Assert post-fire disturbance mortality and regeneration*

Description

Assert post-fire disturbance mortality and regeneration

Usage

```
assertPostPartialDist(
  cohortDataOrig,
  pixelGroupMapOrig,
  cohortDataNew,
  pixelGroupMapNew,
  postDistPixelCohortData,
  distrbdPixelCohortData,
  doAssertion = getOption("LandR.assertions", TRUE)
)
```

Arguments

cohortDataOrig	original cohortData (prior to any modification)
pixelGroupMapOrig	original pixelGroupMap (prior to any modification)
cohortDataNew	modified cohortData output from updateCohortData()
pixelGroupMapNew	modified pixelGroupMap output from updateCohortData()
postDistPixelCohortData	modified cohortData output from updateCohortData()
distrbdPixelCohortData	cohortData-like object containing all dead, surviving and new cohorts (i.e. activated by serotiny/resprouting)
doAssertion	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: <code>getOption("LandR.assertions", TRUE)</code> .

assignLightProb	<i>Assign light probability</i>
-----------------	---------------------------------

Description

Assign light probability

Usage

```
assignLightProb(
  sufficientLight,
  newCohortData,
  interpolate = TRUE,
  doAssertion = getOption("LandR.assertions", TRUE)
)
```

Arguments

sufficientLight	a <code>data.frame</code> (not <code>data.table</code> !) containing probability of establishment, given a site's light conditions (X0-X5) for each level of a species shade tolerance (1-5).
newCohortData	a modified version of <code>cohortData</code> that contains new cohorts.
interpolate	Logical. Activates interpolation of probabilities of establishment between any two values of shade tolerance in the sufficient light table, allowing species shade tolerance trait values to take any decimal value between 1 and 5 (inclusively). If <code>FALSE</code> , species shade tolerances can only take integer values between 1 and 5 (inclusively).
doAssertion	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: <code>getOption("LandR.assertions", TRUE)</code> .

Value

newCohortData with a lightProb column

biomodEnsembleFrcstWrapper

Simple wrapper around [biomod2::BIOMOD_EnsembleForecasting\(\)](#)

Description

Simple wrapper around [biomod2::BIOMOD_EnsembleForecasting\(\)](#)

Usage

```
biomodEnsembleFrcstWrapper(
  bm.em,
  bm.proj = NULL,
  proj.name = NULL,
  new.env = NULL,
  new.env.xy = NULL,
  keep.in.memory = TRUE,
  ...
)
```

Arguments

bm.em	output of biomod2::BIOMOD_EnsembleModeling
bm.proj	output of biomod2::BIOMOD_Projection
proj.name	passed to biomod2::BIOMOD_Projection
new.env	passed to biomod2::BIOMOD_EnsembleForecasting
new.env.xy	passed to biomod2::BIOMOD_EnsembleForecasting
keep.in.memory	passed to biomod2::BIOMOD_EnsembleForecasting
...	passed to reproducible::Cache

biomodEnsembleProjMaps

Make maps from [BIOMOD_EnsembleForecasting\(\)](#) results

Description

Make maps from [BIOMOD_EnsembleForecasting\(\)](#) results

Usage

```
biomodEnsembleProjMaps(bm.em.proj, predModel, rasTemplate, origCRS)
```

Arguments

<code>bm.em.proj</code>	output of <code>biomod2::BIOMOD_EnsembleForecasting()</code> . Note that <code>x</code> and <code>y</code> coordinates extracted from <code>bm.em.proj@coord</code> must be in the same projection as <code>rasTemplate</code> .
<code>predModel</code>	character. Which model predictions should be used. Choose one of <code>bm.em.proj@models.projected</code> .
<code>rasTemplate</code>	a template <code>RasterLayer</code> or <code>SpatRaster</code> that can be used to map the projections
<code>origCRS</code>	character. The original CRS projection of <code>bm.em.proj@coord</code> .

Value

a `SpatRaster` object

`biomodEnsembleWrapper` *BIOMOD_EnsembleModeling wrapper*

Description

`BIOMOD_EnsembleModeling` wrapper

Usage

```
biomodEnsembleWrapper(bm.mod, metric.select.thresh = NULL, ...)
```

Arguments

<code>bm.mod</code>	output of biomod2::BIOMOD_Modeling
<code>metric.select.thresh</code>	passed biomod2::BIOMOD_EnsembleModeling . By default no thresholding is applied.
<code>...</code>	passed to reproducible::Cache

`biomodModelingWrapper` *BIOMOD_Modeling wrapper*

Description

`BIOMOD_Modeling` wrapper

Usage

```

biomodModelingWrapper(
  sp,
  responseVar,
  responseVarData,
  predictorVars,
  predictorVarsData,
  dir.name,
  BIOMOD_ModelingArgs = list(models = c("GLM", "MARS"), bm.options = NULL, CV.k = 5,
    CV.perc = 100, metric.eval = c("TSS", "ROC"), modeling.id = "test"),
  ...
)

```

Arguments

sp	character of species name to subset responseVarData table
responseVar	character vector, or list of character vectors, of response variable (column) to subset responseVarData table. If a list, it should be named according to sp, for subsetting.
responseVarData	a data.table or list of data.tables with environmental data. If a list, it should be named according to sp, for subsetting.#'
predictorVars	character vector, or list of character vectors, of environmental variables (columns) to subset predictorVarsData table. If a list, it should be named according to sp, for subsetting.
predictorVarsData	environmental data.
dir.name	passed to biomod2::BIOMOD_FormatingData
BIOMOD_ModelingArgs	a named list of arguments passed to biomod2::BIOMOD_Modeling
...	further arguments passed to reproducible::Cache

biomodProjWrapper	<i>BIOMOD_Projection wrapper</i>
-------------------	----------------------------------

Description

BIOMOD_Projection wrapper

Usage

```

biomodProjWrapper(
  bm.mod,
  proj.name = "testProj",
  new.env = NULL,

```

```

    new.env.xy = NULL,
    ...
)

```

Arguments

<code>bm.mod</code>	passed to biomod2::BIOMOD_Projection . If not supplied the data used to fit the model will be used.
<code>proj.name</code>	passed to biomod2::BIOMOD_Projection
<code>new.env</code>	passed to biomod2::BIOMOD_Projection
<code>new.env.xy</code>	passed to biomod2::BIOMOD_Projection
<code>...</code>	passed to reproducible::Cache

BioSIM_extractPoints *Extract point locations from DEM raster to pass to BioSIM functions*

Description

Extract point locations from DEM raster to pass to BioSIM functions

Usage

```
BioSIM_extractPoints(x)
```

Arguments

<code>x</code>	A digital elevation model (DEM) RasterLayer.
----------------	--

Value

`data.table` with columns Name, Long, Lat, Elev.

See Also

[prepInputsCanDEM\(\)](#)

BioSIM_getMPBSLR	<i>Get annual historic and projected MPB climate suitability maps from BioSIM</i>
------------------	---

Description

Raster stacks for all 9 MPB climate indices. See `BioSIM::getModelHelp("MPB_SLR")`.

Usage

```
BioSIM_getMPBSLR(dem, years, SLR = "R", climModel = "GCM4", rcp = "RCP45")
```

Arguments

dem	RasterLayer of elevation data (m).
years	numeric vector corresponding to the years to retrieve.
SLR	character. Specifies which climate suitability index to extract. Currently, one of "S", "L", "R", or "G", corresponding to Safranyik-P3P4, Logan-2b, Régnière Cold Tolerance Survival, or their Geometric product (S*L*R), respectively.
climModel	climate model to use. one of "GCM4" or "RCM4".
rcp	RCP scenario to use. one of "RCP45" or "RCP85".

Value

RasterStack

Note

Although the BioSIM MPB_SLR model provides several other indices (see `BioSIM::getModelHelp("MPB_SLR")`), only 4 are currently used here.

BioSIM_getWindAnnual	<i>Get annual historic and projected wind maps from BioSIM</i>
----------------------	--

Description

Get annual historic and projected wind maps from BioSIM

Usage

```
BioSIM_getWindAnnual(dem, years, climModel = "GCM4", rcp = "RCP45")
```

Arguments

dem	RasterLayer of elevation data (m).
years	numeric vector corresponding to the years to retrieve.
climModel	climate model to use. one of "GCM4" or "RCM4".
rcp	RCP scenario to use. one of "RCP45" or "RCP85".

Value

RasterStack

BioSIM_getWindMonthly *Get monthly historic and projected wind maps from BioSIM*

Description

Get monthly historic and projected wind maps from BioSIM

Usage

```
BioSIM_getWindMonthly(dem, years, months, climModel = "GCM4", rcp = "RCP45")
```

Arguments

dem	RasterLayer of elevation data (m).
years	numeric vector corresponding to the years to retrieve.
months	numeric vector corresponding to the months to retrieve (e.g., 6 : 8 for June through August).
climModel	climate model to use. one of "GCM4" or "RCM4".
rcp	RCP scenario to use. one of "RCP45" or "RCP85".

Value

RasterStack

calc_raster_counts	<i>Raster summary statistics and maps by polygon</i>
--------------------	--

Description

calc_raster_stats iteratively calculates the frequency of raster values within each polygon. Supports user-provided polygons or automatic retrieval of ecoregions/ecozones.

Usage

```
calc_raster_counts(  
  raster,  
  polygons = NULL,  
  polygon_id = NULL,  
  filter_ids = NULL  
)  
  
calc_raster_stats(  
  raster,  
  polygons = NULL,  
  polygon_id = NULL,  
  filter_ids = NULL,  
  counts_df = NULL  
)  
  
plot_raster_stats(  
  raster,  
  polygons = NULL,  
  polygon_id = NULL,  
  filter_ids = NULL,  
  counts_df = NULL,  
  stats_df = NULL,  
  aggregate_factor = 1,  
  remove_zeros = FALSE,  
  raster_label = NULL,  
  inset_canada = TRUE,  
  inset_polygons = NULL,  
  bin_width = 10,  
  output_dir = ".",  
  csv_file = NULL,  
  fig_width = 12,  
  fig_height = 7.5,  
  fig_dpi = 300  
)
```

Arguments

raster	A SpatRaster object.
--------	----------------------

<code>polygons</code>	A <code>SpatVector</code> or <code>sf</code> object of polygons.
<code>polygon_id</code>	Name of the column in polygons to use as polygon ID.
<code>filter_ids</code>	Optional. Polygon IDs to filter.
<code>counts_df</code>	optional <code>data.frame</code> of frequency counts output by <code>calc_raster_counts</code> . If not provided, that function will be called internally.
<code>stats_df</code>	optional <code>data.frame</code> of raster statistics output by <code>calc_raster_stats</code> . If not provided, that function will be called internally.
<code>aggregate_factor</code>	Optional. Integer factor to aggregate raster for plotting. Default is 1 (no aggregation).
<code>remove_zeros</code>	Logical. Removes zeroes from raster and counts for plotting. Useful to eliminate non-forested areas from forest rasters.
<code>raster_label</code>	Optional. Character label for raster used in plots. Defaults to raster layer name.
<code>inset_canada</code>	Logical. Generates a small inset map of plotted polygons within Canada.
<code>inset_polygons</code>	Optional <code>sf</code> or <code>SpatVector</code> outline to draw as the backdrop of the inset map. When <code>NULL</code> (the default) the Canada boundary is downloaded via <code>gadm_canada()</code> . Supply it to keep the call offline – e.g. <code>sf::st_union()</code> of the polygons being summarized. Ignored when <code>inset_canada = FALSE</code> .
<code>bin_width</code>	Numeric. Bin size for histogram plots.
<code>output_dir</code>	Directory to save figures and optional csv file.
<code>csv_file</code>	Optional. Base filename to save the output CSV file. If provided, a suffix will be added to this filename to denote the 'counts' and 'stats' tables (e.g., if <code>csv_file = "ecozones.csv"</code> , the output files saved to <code>output_dir</code> will be <code>ecozones_counts.csv</code> and <code>ecozones_stats.csv</code>).
<code>fig_width, fig_height</code>	Numeric. Size of each saved figure, in inches.
<code>fig_dpi</code>	Numeric. Resolution of each saved figure. The defaults suit print; drop <code>fig_dpi</code> (and the dimensions) well below them when the figures are destined for a web page or a vignette, where a 300 dpi 12 x 7.5 in PNG is about 20x larger than it needs to be.

Details

`plot_raster_stats` calculates and returns statistics (min, max, median, mean, 25th percentile, 75th percentile, and proportion of zeroes) for a raster, and creates and saves figures for each polygon in a vector layer, including a histogram of raster values, a map of raster values within the polygon, and a summary of statistics. Optionally, zero values can be removed from the analysis, and the raster can be aggregated to a lower resolution to speed up plotting.

Value

A `data.frame` with columns: ID, value, count.

A `data.frame` of computed statistics (via `calc_raster_stats()`), with side effects of saving summary plots (and optionally a '.csv') to disk.

Note

Large rasters should be processed on disk wherever possible (see `terra::terraOptions()`). Cap the RAM terra uses per operation with `memmax` (in GB) and force results to disk with `todisk = TRUE`, e.g. `terraOptions(memmax = 16, todisk = TRUE, tempdir = <fast local dir>)`. Prefer `memmax` over `memfrac`: `memfrac` is a *fraction of free RAM* (not a hard cap), and `memfrac = 0.0` crashes (terra's block size collapses to 0). Point `tempdir` at fast *local* storage (NVMe) – never NFS, and never a `tmpfs` such as `/tmp` on some systems, which is RAM-backed (so a raster "copied to disk" there really sits in RAM, e.g. via `withr::local_tempdir()`).

For an integer raster, `calc_raster_stats()` can be derived from the value-frequency table produced by `calc_raster_counts()` (every statistic is an exact function of it), which avoids reading every pixel into memory.

Examples

```
if (interactive()) {
  if (requireNamespace("dplyr", quietly = TRUE) &&
      requireNamespace("geodata", quietly = TRUE) &&
      requireNamespace("withr", quietly = TRUE)) {
    tmp_pth <- withr::local_tempdir()

    ## process the raster on disk, with bounded RAM (see @note)
    terraOptions(memmax = 16, todisk = TRUE, tempdir = tmp_pth)

    age <- prepInputsStandAgeMap(
      dataSource = "KNN",
      dataYear = 2011,
      destinationPath = tmp_pth
    )

    ecozones <- reproducible::prepInputs(
      url = "https://sis.agr.gc.ca/cansis/nsdb/ecostrat/zone/ecozone_shp.zip",
      destinationPath = tmp_pth
    ) |>
      sf::st_make_valid()

    id_col <- "ZONE_NAME"

    ## WARNING: don't use Cache() below -- excessive RAM usage

    ecozone_counts <- calc_raster_counts(
      raster = age,
      polygons = ecozones,
      polygon_id = id_col
    )

    ## reuse the counts to avoid recomputing the frequency table
    ecozone_stats <- calc_raster_stats(
      raster = age,
      polygons = ecozones,
      polygon_id = id_col,
      counts_df = ecozone_counts
    )
  }
}
```

```

)

gc()

## below will call calc_raster_counts and calc_raster_stats internally,
## so we shortcut this by passing the previously-calculated tables
stats_ecozone <- plot_raster_stats(
  raster = age,
  polygons = ecozones,
  polygon_id = id_col,
  filter_ids = NULL,
  counts_df = ecozone_counts,
  stats_df = ecozone_stats,
  aggregate_factor = 1,
  remove_zeros = FALSE,
  raster_label = "age",
  inset_canada = TRUE,
  bin_width = 10,
  output_dir = tmp_pth,
  csv_file = NULL
)

gc()

## review the plots produced in tmp_pth
list.files(tmp_pth, pattern = "[.]png", full.names = TRUE)

identical(ecozone_stats, stats_ecozone) ## TRUE
}
}

```

calcSeverityB

Calculate fire severity

Description

Calculates fire severity as the loss of pre-fire to post-fire biomass (in absolute and percentual terms).

Usage

```
calcSeverityB(cohortData, burnedPixelCohortData)
```

Arguments

cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPPAct, and sumB.
------------	--

burnedPixelCohortData

An expanded cohortData data.table with pixel-level cohort information on burnt pixels and the following (optional) columns: severity - fire severity in that pixel calculated based on fire behaviour properties; firetolerance - species-level fire tolerance; severityToleranceDif - the difference between severity and firetolerance.

Value

data.table with columns pixelIndex, pixelGroup and severityB (absolute biomass lost) and severityPropB (proportion of biomass lost)

Note

if burnedPixelCohortData does not have a B column, the fire is assumed to be stand replacing (i.e. we assume B to be 0 across all pixels/cohorts in burnedPixelCohortData)

calcSiteShade	<i>Calculate site shade</i>
---------------	-----------------------------

Description

Calculate site shade

Usage

```
calcSiteShade(currentTime, cohortData, speciesEcoregion, minRelativeB)
```

Arguments

currentTime	The current simulation time e.g., time(sim).
cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPPAct, and sumB.
speciesEcoregion	A data.table with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: speciesCode and ecoregionGroup, character representation of species and ecoregion groups respectively, maxB the maximum biomass for the species in a given 'ecoregion', maxANPP the maximum aboveground net primary productivity and SEP the species establishment probability. May contain columns inflationFactor (used to adjust maxB) and mANPPproportion (used to calculate maxANPP).
minRelativeB	a data.frame with the cut points to classify stand shadiness.

Value

cohortData table with a siteShade column

See Also

[makeMinRelativeB\(\)](#)

calculateClimateEffect
<i>Null climate effect</i>

Description

Default climate effects function in the case where no climate effect is simulated

Usage

```
calculateClimateEffect(cohortData, ...)
```

Arguments

cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPPAct, and sumB.
...	additional arguments that are passed to LandR.CS

Details

the cohortData object is used to calculate the \reduction/increase of mortality and growth biomasses per cohort.

Value

data.table with pixelGroup, age and speciesCode, as well as mortPred and growthPred columns with \ growth biomasses resulting from a climate effect. These percentages are later multiplied by the by baseline biomasses of mortality and growth (e.g. 0\ This default, no climate effect, function outputs 100\ mortPred and growthPred.

CASFRItoSpRasts	CASFRItoSpRasts
-----------------	-----------------

Description

Extract CASFRI data.table values and create species layer rasters

Usage

```
CASFRItoSpRasts(
  CASFRIRas,
  CASFRIattrLong,
  CASFRIdt,
  sppEquiv,
  sppEquivCol,
  destinationPath
)
```

Arguments

CASFRIRas	RasterLayer object
CASFRIattrLong	data.table of CASFRI attributes (long format) (i.e., from loadCASFRI)
CASFRIdt	data.table of CASFRI raster values (i.e., from loadCASFRI)
sppEquiv	table with species name equivalencies between naming conventions (see sppEquivalencies_CA , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
sppEquivCol	the column name to use from sppEquiv (see sppEquivalencies_CA). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.
destinationPath	path to data directory where objects will be downloaded or saved to

Value

RasterStack (or equivalent)

checkSpeciesTraits	<i>Check if all species in have trait values</i>
--------------------	--

Description

Check if all species in have trait values

Usage

```
checkSpeciesTraits(speciesLayers, species, sppColorVect)
```

Arguments

speciesLayers	A RasterStack or RasterLayer that should contain species cover data in the study area
species	a data.table with species traits such as longevity, shade tolerance, etc. Must have column speciesCode, with species names/IDs. The following is a list of default trait columns: • "species" same as "speciesCode" – species ID name • "speciesCode" • "Area" – inherited from LANDIS-II default table, the Canadian ecoregion from which traits were derived. Not used during the simulation • "firetolerance" – module <i>relative</i> (to other species) fire tolerance • "growthcurve" and "mortalityshape" – growth curve shape parameters. • "longevity" – maximum species age • "postfireregen" – post-fire regeneration strategy ("serotiny", "resprout" or "none") • "resproutprob" – probability of resprouting • "resproutage_min" – minimum age at which species is capable of resprouting • "resproutage_max" – maximum age at which species is capable of resprouting • "seeddistance_eff" – effective dispersal distance • "seeddistance_max" – maximum dispersal distance • "shadetolerance" – <i>relative</i> (to other species) shade tolerance • "sexualmature" – age at sexual maturity Known optional parameters added/needed by some modules (the user may add others for their own modules): • "inflationFactor" – Biomass_speciesParameters module: inflation factor for maxB • "growthCurveSource" – Biomass_speciesParameters module: how "growthcurve" was estimated • "mANPPproportion" – Biomass_speciesParameters module: multiplication factor to calculate maxANPP from maxB • "thermokarsttol" – Biomass_disturbances module: proportion of biomass surviving after thermokarst (i.e. permafrost thaw). Applied equally across cohorts. Parameters inherited from LANDIS-II default table, but not used in LandR at the moment: • "leaflongevity" • "wooddecayrate" • "leafLignin" • "hardsoft" Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.
sppColorVect	A named vector of colours to use for plotting. The names must conform with species name convention used (see sppEquivalencies_CA for possible naming conventions) and should also contain a colour for 'Mixed', when a mixed forest type is supported (see vegLeadingProportion parameter in vegTypeGenerator() for details on mixed forest types).

Value

A list with the speciesLayers and sppColorVect containing only the species that have trait values in species

cohortDefinitionCols	<i>Default columns that define cohorts</i>
----------------------	--

Description

Default columns that define cohorts

Usage

```
cohortDefinitionCols()
```

Note

because the name cohortDefinitionCols is also used as a function argument, be sure to use `LandR::cohortDefinitionCols()` in those functions or you'll get a "promise already under evaluation" error.

Colors	<i>Helper for setting Raster or SpatRaster colors</i>
--------	---

Description

This is a wrapper to help with migration to **terra**. Currently can only be used for a single layer SpatRaster or a RasterLayer.

Usage

```
Colors(ras, cols, n = NULL)
```

Arguments

ras	A Raster* or SpatRaster class object.
cols	a character vector of colours. See examples. Can also be a data.frame, see terra::coltab .
n	A numeric scalar giving the number of colours to create. For a RasterLayer, the number of colours the colour table is interpolated to (for a categorical RasterLayer, the number of categories). If missing, then n will be length(cols).

Examples

```
cols <- colorRampPalette(c("blue", "red"))(12)
ras <- terra::rast(matrix(1:100, 10, 10))
ras <- Colors(ras, cols)
terra::plot(ras)

ras <- raster::raster(matrix(1:100, 10, 10))
ras <- Colors(ras, cols)
raster::plot(ras)
```

columnsForPixelGroups *Default columns that define pixel groups*

Description

Default columns that define pixel groups

Usage

```
columnsForPixelGroups()
```

Note

because the name columnsForPixelGroups is also used as a function argument, be sure to use `LandR::columnsForPixelGroups()` in those functions or you'll get a "promise already under evaluation" error.

convert_SCANFI_LCC_codes

Convert SCANFI Landcover layers from 1-8 codes to typical Canada LCC codes (0-230)

Description

This function has been run previously to create layers with the correct codes which are available on Google Drive. These corrected layers are available here: <https://drive.google.com/drive/folders/1zLYV-wcDjJfSf1H1VkJG6sosqZZF4SYc> In the respective folder for each year, under the file name SCANFI_att_nfILandCover_CanadaLCCclassCodes_S_<YEAR>_v1_1.tif

Usage

```
convert_SCANFI_LCC_codes(year = 2000, dataVersion = "V2", writeTo = NULL, ...)
```

Arguments

year	data year for SCANFI landcover data. 2000, 2010, and 2020 possible for V1. 1985, 1990, 1995, 2000, 2005, 2010, 2015, 2020, 2025 possible for V2.
dataVersion	Character. SCANFI product version for data. Default is currently V2. V1 also available.
writeTo	Optional character. If supplied, the converted layer is written to this file as a tiled, LZW-compressed INT1U GeoTIFF (NAflag = 255). All output codes lie in 20-230, so a single unsigned byte is sufficient; leaving terra to its Float32 default roughly doubles the file and, because the default layout is full-width strips, makes windowed reads of these national (178400 x 119100) rasters needlessly expensive.
...	additional args passed to <code>reproducible::prepInputs()</code>

Value

a SpatRaster with corrected classification codes

convertUnwantedLCC	<i>Convert Land Cover Classes (LCC) to another value in its neighbourhood</i>
--------------------	---

Description

This searches around the pixels on rstLCC that have classesToReplace for the nearest neighbouring Land Cover Class other than those indicated in classesToReplace, and assigns the cohorts in those pixels the corresponding new ecoregionCode. Replacement codes must already exist in availableERC_by_Sp, i.e., no new ecoregionCode values are created. See Details.

Usage

```
convertUnwantedLCC(
  classesToReplace = 34:36,
  rstLCC,
  availableERC_by_Sp,
  theUnwantedPixels,
  ecoregionGroupVec,
  speciesEcoregion,
  pixelClassesToReplace,
  doAssertion = getOption("LandR.assertions", TRUE),
  method = c("nearestWeighted", "nearestRandom")
)
```

Arguments

classesToReplace	Integer vector of classes that are to be replaced, e.g., 34, 35, 36 on LCC2005, which are burned young, burned 10 year, and cities.
rstLCC	LCC raster, e.g., LCC2005
availableERC_by_Sp	A <code>data.table</code> or <code>data.frame</code> with 3 columns: <code>speciesCode</code> , <code>initialEcoregionCode</code> and <code>pixelIndex</code> . <code>pixelIndex</code> is the pixel id for each line in the <code>data.table</code> ; <code>speciesCode</code> is the species name in the pixel (can have more than one species per pixel, so multiple rows per pixel); and, <code>initialEcoregionCode</code> is the unique codes that are "available" to be used as a replacement for <code>classesToReplace</code> . <code>initialEcoregionCode</code> must be a character vector, with one or no "" <i>used as a separator, with the last component being the Land Cover Class that matches classesToReplace, e.g., "242_18". If there is no "" in this code, then the codes must match the classesToReplace exactly, e.g., "11".</i> If <code>pixelIndex</code> is missing, the function will fill it with <code>seq(ncell(rstLCC))</code> . If <code>speciesCode</code> is missing, the function will replace it with a dummy value ("allSpecies").
theUnwantedPixels	An optional vector of pixel IDs that need to be changed. If not provided, then pixels to change will be taken from the match between <code>availableERC_by_Sp</code> and <code>classesToReplace</code> . Supplying this allows the user to only replace some of the pixels with a given class.
ecoregionGroupVec	Deprecated. Use <code>availableERC_by_Sp</code>
speciesEcoregion	Deprecated. Use <code>availableERC_by_Sp</code>
pixelClassesToReplace	Deprecated. Use <code>classesToReplace</code>
doAssertion	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: <code>getOption("LandR.assertions", TRUE)</code> .
method	Character; where the draw among an unwanted pixel's available land-cover classes comes from. Both options weight the classes by their abundance in the pixel's neighbourhood and differ only in reproducibility: "nearestWeighted" (default) keys the draw on the pixel's ground position, so it is deterministic and seed-free, while "nearestRandom" draws from the RNG and so varies with <code>set.seed()</code> . See Details.

Details

This function is designed to be used in highly constrained situations, where it is not just replacing a Land Cover Class by a neighbouring Land Cover Class. But it can be used for the simpler cases of simply replacing a Land Cover Class.

Each pixel whose value is in `classesToReplace` is assigned an available `initialEcoregionCode` taken from its neighbourhood. For each candidate land-cover class present in `rstLCC`, the distance from every pixel to the nearest cell of that class is computed with a single vectorized `terra::distance()` call, and each unwanted pixel keeps only those classes whose implied `initialEcoregionCode` is

available for that pixel's ecoregion (and all of its speciesCodes). One of the surviving classes is then drawn with probability proportional to how many cells of that class fall inside the pixel's neighbourhood – the smallest window reaching that pixel's nearest available class. method decides only where the draw comes from:

- "nearestWeighted" (default) derives it from the pixel's own ground position, so the result is deterministic: it needs no `set.seed()`, is stable under `reproducible::Cache()` (which does not key on RNG state), and, because the key is the cell *centre* rather than the cell *index*, a grid-aligned crop reproduces its parent raster cell for cell – so a small development subset agrees with the full run. Reprojecting or changing resolution changes the pixels, and so changes the result.
- "nearestRandom" draws from the RNG instead, so it varies with `set.seed()`. Use it when replicates should differ; note that `Cache()` will replay a single draw unless the seed is part of the cache key.

Both are $O(nClasses)$ distance transforms, independent of the geometry of the `classesToReplace` blobs. They replace an earlier iterative `spread2()`-based search whose run time grew with the square of the radius of the largest contiguous block of `classesToReplace`, which could take hours – or never finish – on large study areas with big lakes or burns, or an irregular masked boundary.

The abundance weighting restores the behaviour of that former search, which sampled among all valid cells within the radius at which it first found one and so favoured classes in proportion to their local abundance. The window is rectangular, matching the Chebyshev neighbourhood of `spread2(directions = 8)`. What it does *not* reproduce is that implementation's exact output: the radius here comes from a Euclidean distance transform rather than from counting spread iterations, and the draws differ. Runs prior to LandR 1.2.0.9004 cannot be reproduced bit-for-bit by any current method.

LandR 1.2.0.9004's deterministic nearest-class rule, which broke ties to the lowest land-cover class, has been removed. Ties are common – 35-41% of unwanted pixels on real landscapes – so always taking the lowest code biased the result toward low-numbered classes, which in the Canada LCC coding means toward sparse, non-forest cover: it assigned shrubs 1.69x as often as the previous implementation, and broadleaf 0.58x and mixedwood 0.28x as often.

Value

A data.table with three columns, `newPossLCC`, `pixelIndex` and `ecoregionGroup`. This represents the new codes to be used in the `pixelIndex` locations. These should have no values overlapping with `classesToReplace`. `newPossLCC` is the assigned land-cover class itself, i.e. `ecoregionGroup` without its ecoregion prefix; it is NA for pixels that could not be assigned.

Author(s)

Eliot McIntire

convertUnwantedLCC2	<i>Convert Land Cover Classes (LCC) to another value in its neighbourhood</i>
---------------------	---

Description

This will search around the pixels on rstLCC that have classesToReplace, and search in iteratively increasing radii outwards for other Land Cover Classes than the those indicated in classesToReplace.

Usage

```
convertUnwantedLCC2(
  classesToReplace = 34:36,
  rstLCC,
  nIterations = 6,
  defaultNewValue = NA,
  invalidClasses = NA
)
```

Arguments

classesToReplace	Integer vector of classes that are to be replaced, e.g., 34, 35, 36 on LCC2005, which are burned young, burned 10 year, and cities.
rstLCC	raster of land-cover class, e.g., LCC2005
nIterations	the number of iterations to perform
defaultNewValue	the value to assign a pixel in classesToReplace if no valid pixel is closer after nIterations
invalidClasses	classes that are not valid options

Value

A rasterLayer with values in classesToReplace converted to adjacent values or NA.

Author(s)

Eliot McIntire Ian Eddy

defineFlammable	<i>Define flammability map</i>
-----------------	--------------------------------

Description

Define flammability map

Usage

```
defineFlammable(
  LandCoverClassifiedMap = NULL,
  nonFlammClasses = c(0L, 25L, 30L, 33L, 36L, 37L, 38L, 39L),
  mask = NULL,
  to = NULL,
  writeTo = NULL,
  ...
)
```

Arguments

LandCoverClassifiedMap	A Raster that represents land cover (e.g., Land Cover Classified map from 2005 or 2010 from the Canadian Forest Service).
nonFlammClasses	numeric vector defining which classes in LandCoverClassifiedMap.
mask	A raster to use as a mask (see raster::mask()).
to	Passed to <code>postProcessTo(..., to = to)</code> and to the mask arg here, if mask is not supplied.
writeTo	See reproducible::postProcess() . Default NULL.
...	additional args (not used)

describeCohortData	<i>Summary for cohortData</i>
--------------------	-------------------------------

Description

Summary for cohortData

Usage

```
describeCohortData(cohortData)
```

Arguments

cohortData	A <code>data.table</code> with columns: <code>pixelGroup</code> , <code>ecoregionGroup</code> , <code>speciesCode</code> , and optionally <code>age</code> , <code>B</code> , <code>mortality</code> , <code>aNPAct</code> , and <code>sumB</code> .
------------	--

doResprouting	<i>Activate resprouting after a (fire) disturbance</i>
---------------	--

Description

Activate resprouting after a (fire) disturbance

Usage

```
doResprouting(
  burnedPixelCohortData,
  postFirePixelCohortData,
  postFireRegenSummary = NULL,
  serotinyPixel,
  treedFirePixelTableSinceLastDisp,
  currentTime,
  species,
  sufficientLight,
  calibrate = FALSE
)
```

Arguments

burnedPixelCohortData	An expanded cohortData data.table with pixel-level cohort information on burnt pixels and the following (optional) columns: severity - fire severity in that pixel calculated based on fire behaviour properties; firetolerance - species-level fire tolerance; severityToleranceDif - the difference between severity and firetolerance.
postFirePixelCohortData	an empty cohortData-like table with columns: age, B, mortality, aNPPAct, and sumB removed; and pixelIndex added.
postFireRegenSummary	a data.table summarizing for which species serotiny/resprouting were activated and in how many pixels, for each year. Only necessary if calibrate = TRUE.
serotinyPixel	a vector of pixels where serotiny was activated;
treedFirePixelTableSinceLastDisp	a vector of pixels that burnt and were forested in the previous time step.
currentTime	integer. The current simulation time obtained with time(sim)
species	a data.table with species traits such as longevity, shade tolerance, etc. Must have column speciesCode, with species names/IDs. The following is a list of default trait columns: • "species" same as "speciesCode" – species ID name • "speciesCode"

- "Area" – inherited from LANDIS-II default table, the Canadian ecoregion from which traits were derived. Not used during the simulation
- "firetolerance" – modular *relative* (to other species) fire tolerance
- "growthcurve" and "mortalityshape" – growth curve shape parameters.
- "longevity" – maximum species age
- "postfireregen" – post-fire regeneration strategy ("serotiny", "resprout" or "none")
- "resproutprob" – probability of resprouting
- "resproutage_min" – minimum age at which species is capable of resprouting
- "resproutage_max" – maximum age at which species is capable of resprouting
- "seeddistance_eff" – effective dispersal distance
- "seeddistance_max" – maximum dispersal distance
- "shadetolerance" – *relative* (to other species) shade tolerance
- "sexualmature" – age at sexual maturity Known optional parameters added/needed by some modules (the user may add others for their own modules):
- "inflationFactor" – Biomass_speciesParameters module: inflation factor for maxB
- "growthCurveSource" – Biomass_speciesParameters module: how "growthcurve" was estimated
- "mANPPproportion" – Biomass_speciesParameters module: multiplication factor to calculate maxANPP from maxB
- "thermokarsttol" – Biomass_disturbances module: proportion of biomass surviving after thermokarst (i.e. permafrost thaw). Applied equally across cohorts. Parameters inherited from LANDIS-II default table, but not used in LandR at the moment:
- "leaflongevity"
- "wooddecayrate"
- "leafLignin"
- "hardsoft" Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.

sufficientLight

a data.frame (not data.table!) containing probability of establishment, given a site's light conditions (X0-X5) for each level of a species shade tolerance (1-5).

calibrate

logical. Determines whether to output postFirePixelCohortData. Defaults to FALSE

Value

A list of objects: postFirePixelCohortData, a data.table with the cohorts that undergo serotiny; serotinyPixel, a vector of pixels where serotiny was activated; postFireRegenSummary, the updated postFireRegenSummary, if calibrate = TRUE.

doSerotiny

*Activate serotiny after a (fire) disturbance***Description**

Activate serotiny after a (fire) disturbance

Usage

```
doSerotiny(
  burnedPixelCohortData,
  postFirePixelCohortData,
  postFireRegenSummary = NULL,
  species,
  sufficientLight,
  speciesEcoregion,
  currentTime,
  treedFirePixelTableSinceLastDisp,
  calibrate = FALSE
)
```

Arguments

burnedPixelCohortData

An expanded cohortData data.table with pixel-level cohort information on burnt pixels and the following (optional) columns: severity - fire severity in that pixel calculated based on fire behaviour properties; firetolerance - species-level fire tolerance; severityToleranceDif - the difference between severity and firetolerance.

postFirePixelCohortData

an empty cohortData-like table with columns: age, B, mortality, aNPPAct, and sumB removed; and pixelIndex added.

postFireRegenSummary

a data.table summarizing for which species serotiny/resprouting were activated and in how many pixels, for each year. Only necessary if calibrate = TRUE.

species

a data.table with species traits such as longevity, shade tolerance, etc. Must have column speciesCode, with species names/IDs. The following is a list of default trait columns:

- "species" same as "speciesCode" – species ID name
- "speciesCode"
- "Area" – inherited from LANDIS-II default table, the Canadian ecoregion from which traits were derived. Not used during the simulation
- "firetolerance" – modular *relative* (to other species) fire tolerance
- "growthcurve" and "mortalityshape" – growth curve shape parameters.
- "longevity" – maximum species age

- "postfireregen" – post-fire regeneration strategy ("serotiny", "resprout" or "none")
- "resproutprob" – probability of resprouting
- "resproutage_min" – minimum age at which species is capable of resprouting
- "resproutage_max" – maximum age at which species is capable of resprouting
- "seeddistance_eff" – effective dispersal distance
- "seeddistance_max" – maximum dispersal distance
- "shadetolerance" – *relative* (to other species) shade tolerance
- "sexualmature" – age at sexual maturity Known optional parameters added/needed by some modules (the user may add others for their own modules):
- "inflationFactor" – Biomass_speciesParameters module: inflation factor for maxB
- "growthCurveSource" – Biomass_speciesParameters module: how "growthcurve" was estimated
- "mANPPproportion" – Biomass_speciesParameters module: multiplication factor to calculate maxANPP from maxB
- "thermokarsttol" – Biomass_disturbances module: proportion of biomass surviving after thermokarst (i.e. permafrost thaw). Applied equally across cohorts. Parameters inherited from LANDIS-II default table, but not used in LandR at the moment:
- "leaflongevity"
- "wooddecayrate"
- "leafLignin"
- "hardsoft" Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.

sufficientLight

a data.frame (not data.table!) containing probability of establishment, given a site's light conditions (X0-X5) for each level of a species shade tolerance (1-5).

speciesEcoregion

A data.table with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: speciesCode and ecoregionGroup, character representation of species and ecoregion groups respectively, maxB the maximum biomass for the species in a given 'ecoregion', maxANPP the maximum aboveground net primary productivity and SEP the species establishment probability. May contain columns inflationFactor (used to adjust maxB) and mANPPproportion (used to calculate maxANPP).

currentTime integer. The current simulation time obtained with time(sim)

treedFirePixelTableSinceLastDisp

a vector of pixels that burnt and were forested in the previous time step.

calibrate logical. Determines whether to output postFirePixelCohortData. Defaults to FALSE

Value

A list of objects: postFirePixelCohortData, a data.table with the cohorts that undergo serotiny; serotinyPixel, a vector of pixels where serotiny was activated; postFireRegenSummary, the updated postFireRegenSummary, if calibrate = TRUE.

dropTerm	<i>Drop factor term including interactions from a model formula</i>
----------	---

Description

Based on <https://stackoverflow.com/a/23382097/1380598>.

Usage

```
dropTerm(form, term, dropRanEff = TRUE)
```

Arguments

form	A model formula.
term	Character vector giving the name of the term to drop.
dropRanEff	Logical. If TRUE (the default), then the term to drop will also be dropped from the random effects terms. If FALSE, it will only be dropped from the fixed terms.

Value

An updated model formula.

ecoregionProducer	<i>Make ecoregionMap and ecoregion table</i>
-------------------	--

Description

This function combines an ecoregion map and a land cover map (e.g. ecodistricts and LCC) and creates a map and table of containing their combined values and pixel IDs. Used internally in LandR modules to prepare maps for to make cohortData.

Usage

```
ecoregionProducer(
  ecoregionMaps,
  ecoregionName = NULL,
  rasterToMatch,
  ecoregionTable
)
```

Arguments

- ecoregionMaps a list with two rasters, one with ecoregions (e.g. ecodistricts) and another with land cover (e.g. LCC).
- ecoregionName the name describing the type of ecoregions in first map (e.g. "ecoDistrict") if passing a polygon file.
- rasterToMatch A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
- ecoregionTable A data.table that has 2 columns, ecoregionName (a factor) and ID a factor of paddedFloatToChar(1:length(unique(ecoregionName))), padL = max(nchar(length(unique(ecoregionName)))) This represents all the possible values that are available; this will be joined to ecoregionMaps[[1]] values

Value

A list with two objects: the ecoregionMap and a table summarizing its information per pixelID

equivalentName	<i>Return equivalent name from a data.frame of equivalencies</i>
----------------	--

Description

This is simply a wrapper around match or %in% for a specific data.frame of values.

Usage

```
equivalentName(value, df, column, multi = FALSE, searchColumn = NULL)

equivalentNameColumn(value, df, column, multi = FALSE, searchColumn = NULL)
```

Arguments

- value Vector of values to match in df.
- df A data.frame where every row is a set of equivalent names.
- column A character string or numeric of length 1, indicating the column in df to return names from.
- multi Logical. If TRUE, then all matches will be returned. Default FALSE for backwards compatibility. This may result in more elements than were in value, as each value may be matched by more than one entry, returning more than one result each from column. If a species is found in the df, but there is no corresponding value in column, it will return NA.
- searchColumn Optionally, provide the name of a column in df that results must be found in. The return value will still be from column

See Also

[sppEquivalencies_CA](#) for the sppEquiv table.

extractMaxB	<i>Maximum biomass estimator</i>
-------------	----------------------------------

Description

Estimation of maximum biomass as the A parameter in the Chapman-Richards and Logistic growth equations. Since A is modelled as linear term, it is a matrix product of its linear coefficients. It is assumed that all coefficients are related additively.

Usage

```
extractMaxB(m11, newdata, average = FALSE, model = c("CR", "Logistic"))
```

Arguments

m11	the output of an <code>bbmle::mle2</code> call (the fitted non-linear model), from which coefficient values will be extracted
newdata	data for estimation of 'A'
average	should 'A' be estimated for average values of its predictors.
model	character. Non-linear model form used to estimate average maximum biomass. One of "CR" (Chapman-Richards) or "Logistic".

fasterizeFromSp	<i>Rasterize polygons using fasterize</i>
-----------------	---

Description

Rasterize polygons using fasterize

Usage

```
fasterizeFromSp(sp, raster, fieldName)
```

Arguments

sp	a shapefile to rasterize
raster	the template raster to use
fieldName	the field to use (will be ignored if the shapefile has no fields)

Value

RasterLayer

FireDisturbance	<i>Disturbance functions</i>
-----------------	------------------------------

Description

Disturbance functions

Usage

```
FireDisturbance(
  cohortData = copy(sim$cohortData),
  cohortDefinitionCols = c("pixelGroup", "age", "speciesCode"),
  calibrate = FALSE,
  postFireRegenSummary = copy(sim$postFireRegenSummary),
  treedFirePixelTableSinceLastDisp = sim$treedFirePixelTableSinceLastDisp,
  rstCurrentBurn = sim$rstCurrentBurn,
  inactivePixelIndex = sim$inactivePixelIndex,
  pixelGroupMap = sim$pixelGroupMap,
  currentTime = NULL,
  rasterToMatch = sim$rasterToMatch,
  species = copy(sim$species),
  sufficientLight = copy(sim$sufficientLight),
  speciesEcoregion = copy(sim$speciesEcoregion),
  initialB = 10,
  successionTimestep = 10L,
  doAssertion = getOption("LandR.assertions", TRUE),
  verbose = getOption("LandR.verbose", TRUE)
)

FireDisturbancePM(
  cohortData = copy(sim$cohortData),
  cohortDefinitionCols = c("pixelGroup", "age", "speciesCode"),
  colsForPixelGroups = LandR::columnsForPixelGroups(),
  calibrate = FALSE,
  LANDISPM = TRUE,
  postFireRegenSummary = copy(sim$postFireRegenSummary),
  treedFirePixelTableSinceLastDisp = copy(sim$treedFirePixelTableSinceLastDisp),
  rstCurrentBurn = sim$rstCurrentBurn,
  inactivePixelIndex = sim$inactivePixelIndex,
  pixelGroupMap = sim$pixelGroupMap,
  currentTime = NULL,
  rasterToMatch = sim$rasterToMatch,
  fireDamageTable = copy(sim$fireDamageTable),
  fireRSORas = sim$fireRSORas,
  fireROSRas = sim$fireROSRas,
  fireCFBRas = sim$fireCFBRas,
  species = copy(sim$species),
```

```

sufficientLight = copy(sim$sufficientLight),
speciesEcoregion = copy(sim$speciesEcoregion),
initialB = 10,
minRelativeB = copy(sim$minRelativeB),
successionTimestep = 10L,
doAssertion = getOption("LandR.assertions", TRUE),
verbose = getOption("LandR.verbose", TRUE)
)

PeatlandThermokarst(
  thawedPixIDs = copy(sim$thawedPixIDs),
  treedThawedPixelTableSinceLastDisp = copy(sim$treedThawedPixelTableSinceLastDisp),
  wetlands = sim$wetlands,
  cohortData = copy(sim$cohortData),
  pixelGroupMap = sim$pixelGroupMap,
  rasterToMatch = sim$rasterToMatch,
  species = copy(sim$species),
  speciesEcoregion = copy(sim$speciesEcoregion),
  cohortDefinitionCols = LandR::cohortDefinitionCols(),
  initialB = 10L,
  inactivePixelIndex = sim$inactivePixelIndex,
  currentTime = NULL,
  successionTimestep = 10L
)

```

Arguments

cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPPAct, and sumB.
cohortDefinitionCols	the columns in cohortData that define unique cohorts (default: c("pixelGroup", "speciesCode", "age")).
calibrate	logical. Determines whether to output postFirePixelCohortData. Defaults to FALSE
postFireRegenSummary	a data.table summarizing for which species serotiny/resprouting were activated and in how many pixels, for each year. Only necessary if calibrate = TRUE.
treedFirePixelTableSinceLastDisp	data.table with 3 columns: pixelIndex, pixelGroup, and burnTime. Each row represents a forested pixel that was burned up to and including this year, since last dispersal event, with its corresponding pixelGroup and time it occurred. Pixel group IDs correspond to the last year's pixelGroupMap and not necessarily the pixelGroupMap of the burnTime year.
rstCurrentBurn	raster layer with burnt pixels in currentTime. Usually generated by a fire spread module (or collection of modules) like fireSense. Values in the raster should contain 0's for unburnt pixels and values > 0 for burn pixels. NA's are assumed to be outside of the study area.

inactivePixelIndex	an internal object to Biomass_core that track of inactive pixels.
pixelGroupMap	A RasterLayer with pixel values equal to a pixel group number that corresponds exactly to pixelGroup column in cohortData.
currentTime	numeric. Current simulation time step (as in time(sim))
rasterToMatch	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
species	a data.table with species traits such as longevity, shade tolerance, etc. Must have column speciesCode, with species names/IDs. The following is a list of default trait columns: • "species" same as "speciesCode" – species ID name • "speciesCode" • "Area" – inherited from LANDIS-II default table, the Canadian ecoregion from which traits where derived. Not used during the simulation • "firetolerance" – modular<i>relative</i> (to other species) fire tolerance • "growthcurve" and "mortalityshape" – growth curve shape parameters. • "longevity" – maximum species age • "postfireregen" – post-fire regeneration strategy ("serotiny", "resprout" or "none") • "resproutprob" – probability of resprouting • "resproutage_min" – minimum age at which species is capable of resprouting • "resproutage_max" – maximum age at which species is capable of resprouting • "seeddistance_eff" – effective dispersal distance • "seeddistance_max" – maximum dispersal distance • "shadetolerance" – <i>relative</i> (to other species) shade tolerance • "sexualmature" – age at sexual maturity Known optional parameters added/needed by some modules (the user may add others for their own modules): • "inflationFactor" – Biomass_speciesParameters module: inflation factor for maxB • "growthCurveSource" – Biomass_speciesParameters module: how "growthcurve" was estimated • "mANPPproportion" – Biomass_speciesParameters module: multiplication factor to calculate maxANPP from maxB • "thermocarsttol" – Biomass_disturbances module: proportion of biomass surviving after thermokarst (i.e. permafrost thaw). Applied equally across cohorts. Parameters inherited from LANDIS-II default table, but not used in LandR at the moment: • "leaflongevity" • "wooddecayrate" • "leafLignin"

- "hardsoft" Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.

sufficientLight

a `data.frame` (not `data.table`!) containing probability of establishment, given a site's light conditions (X0-X5) for each level of a species shade tolerance (1-5).

speciesEcoregion

A `data.table` with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: `speciesCode` and `ecoregionGroup`, character representation of species and ecoregion groups respectively, `maxB` the maximum biomass for the species in a given 'ecoregion', `maxANPP` the maximum aboveground net primary productivity and `SEP` the species establishment probability. May contain columns `inflationFactor` (used to adjust `maxB`) and `maxANPPproportion` (used to calculate `maxANPP`).

initialB

the initial biomass of new cohorts. Defaults to 10. If NULL or NA, initial cohort biomass is calculated as in LANDIS-II Biomass Succession Extension v3.2.1 (Scheller & Miranda, 2015):

```
initialB = asInteger(pmin(maxANPP, asInteger(pmax(1, maxANPP * exp(-1.6 * sumB / maxB_e
```

where `maxANPP` and `maxB_eco` are the maximum ANPP and B parameters of the species in question within the pixel's ecolocation, and `sumB` is the total stand biomass excluding cohorts with ages less than `successionTimestep`.

successionTimestep

The time between successive seed dispersal events. In LANDIS-II, this is called "Succession Timestep".

doAssertion

A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: `getOption("LandR.assertions", TRUE)`.

verbose

Controls message output. Defaults to `getOption("LandR.verbose")`

colsForPixelGroups

character. `cohortData` columns used to find identical pixels in terms of cohort composition, and group them into `pixelGroups` (default: `c("ecoregionGroup", "speciesCode", "age", "B")`).

LANDISPM

logical. Should partial mortality be calculated as in LANDIS-II Dynamic Fire System v3.0. Must be TRUE for the time being.

fireDamageTable

`data.table` defining upper age limit of cohorts killed by fire. See LANDIS-II Dynamic Fire System v3.0 Manual for an example.

fireRSORas

raster layer of critical spread rate for crowning (m/min) for each burned pixel. If not provided dummy values are used.

fireROSas

raster layer of equilibrium rate of spread (m/min) for each burned pixel. If not provided dummy values are used.

fireCFBRas

raster layer of crown fraction burnt for each burned pixel. If not provided dummy values are used.

minRelativeB

a `data.frame` with the cut points to classify stand shadiness.

thawedPixIDs	integer. Vector of pixel IDs than underwent thermokarst (and were converted to wetlands) in the current year (not the last thermokarst event).
treedThawedPixelTableSinceLastDisp	data.table with 3 columns: pixelIndex, pixelGroup, and burnTime. Each row represents a forested pixel that was burned up to and including this year, since last dispersal event, with its corresponding pixelGroup and time it occurred. Pixel group IDs correspond to the last year's pixelGroupMap and not necessarily the pixelGroupMap of the burnTime year.
wetlands	binary SpatRaster with current wetland pixels.

Value

a list of objects to be exported to sim:

- cohortData
- pixelGroupMap
- lastFireYear
- treedFirePixelTableSinceLastDisp
- serotinyResproutSuccessPixels
- severityBMap
- severityData
- postFireRegenSummary (if calibrate == TRUE and !is.null(postFireRegenSummary))

Stand-replacing fire disturbances

FireDisturbance() simulates post-fire mortality, serotiny and regeneration sequentially occurring after a fire. Post-fire mortality is assumed to be 100% (stand-replacement). The serotiny and regeneration algorithms are based on those in LANDIS-II Biomass Succession extension, v3.2.1, with modifications. Requires the following objects in sim (passed as sim\$*): @template FireDisturbance. For any given burnt pixel, the function begins by killing all cohorts (i.e. removing them from cohortData). Then it activates serotiny for serotinous species that had been present pre-fire, and resprouting for resprouter species. Whether a species successfully regenerates via serotiny or resprouting depends on its traits (whether it is serotinous, resprouter, both or none; species table), pre-fire cohort age (in cohortData table), its light/shade requirements (species table) and its local maxB (speciesEcoregion) table (see Scheller & Miranda, 2015 and Scheller & Mladenoff, 2004 for details). Unlike in LANDIS-II Biomass Succession extension, v3.2.1, here we allow serotiny and resprouting to occur in the same pixel to reflect the competitive advantage of resprouters. However, for a given species only serotiny (takes precedence) or resprouting can be activated. The species table must contain the columns:

- sexualmature – age at sexual maturity
- postfireregen – post-fire regeneration strategy ("serotiny", "resprout" or "none")
- shadetolerance – shade tolerance value relative to *other* species.
- resproutage_min, resproutage_max – minimum and maximum age at which species can resprout
- resproutprob – probability of resprouting success (before light/shade suitability is assessed)

Partial severity (i.e. mortality) fire disturbances

FireDisturbancePM() simulates partial post-fire mortality, serotiny and regeneration sequentially after a fire. The level of mortality depends of fire severity, and, by default, follows the mechanisms in LANDIS-II Dynamic Fire System v3.0. Serotiny and regeneration algorithms are based on those in LANDIS-II Biomass Succession extension, v3.2.1, with modifications (see FireDisturbance()). Requires the following objects in sim (and passed as sim\$*): @template FireDisturbance

- fireDamageTable
- fireRSORas (Critical spread rate for crowning)
- fireROSRas (Equilibrium Head Fire Rate of Spread)
- fireCFBRas (Crown Fraction Burnt)
- minRelativeB Rasters of fire behaviour properties (fireRSORas, fireROSRas and fireCFBRas) can be calculated using the cffdrs package. The species table must contain the columns:
- firetolerance – fire tolerance value relative to *other* species.
- longevity – maximum species age
- sexualmature – age at sexual maturity
- postfireregen – post-fire regeneration strategy ("serotiny", "resprout" or "none")
- shadetolerance – shade tolerance value relative to *other* species.
- resproutage_min, resproutage_max – minimum and maximum age at which species can resprout
- resproutprob – probability of resprouting success (before light/shade suitability is assessed)

Peatland permafrost degradation (thermokarst) disturbances

PeatlandThermokarst() simulates tree cohort survival/mortality after peatland permafrost thermokarst. The level of mortality depends on species tolerance to thermokarst, determined by the thermokarsttol trait column in the species traits table. At the moment, this level of tolerance is used as the proportion of a cohort biomass that survives (is kept) when a pixel undergoes thermokarst. This is similar to the partial disturbance effects used in LANDIS-II Biomass Harvest v4.0. Requires the following objects in sim (and passed as sim\$*):

- thawedPixIDs
- treedThawedPixelTableSinceLastDisp
- wetlands
- cohortData
- pixelGroupMap
- rasterToMatch
- species
- speciesEcoregion
- inactivePixelIndex The species table must contain the columns:
- thermokarsttol – proportion of cohort biomass (B) that survives thermokarst

References

- Scheller, R.M. & Miranda, B.R. (2015). LANDIS-II Biomass Succession v3.2 Extension – User Guide.
- Scheller, R.M. & Mladenoff, D.J. (2004). A forest growth and biomass module for a landscape simulation model, LANDIS: design, validation, and application. *Ecological Modelling*, 180, 211–229.
- Sturtevant, B.R., Miranda, B.R., Scheller, R.M. & Shinneman, D. (2018). LANDIS-II Dynamic Fire System Extension v3.0 – User Guide.
- Scheller, R.M. & Domingo, J.B. (2021). LANDIS-II Biomass Harvest v4.4 Extension – User Guide

fitNLMModels

FUNCTIONS TO FIT NON-LINEAR MODELS TO ESTIMATE MAXB
Wrapper function to fit non-linear growth model per species.

Description

The function will create sensible parameter ranges for A, k and p parameters of the Chapman-Richards and Logistic growth curves and attempt to run a forward step-AIC procedure to add covariates to the linear component of the model (on the A parameter – the asymptote). The maximum number of covariates to add is determined by maxNoCoefs.

Usage

```
fitNLMModels(
  sp = NULL,
  predictorVarsData,
  sppVarsB,
  predictorVars,
  predictorVarsCombos = NULL,
  maxNoCoefs = 4,
  doFwdSelection = FALSE,
  sampleSize = 3000,
  Ntries = 2000,
  maxCover = 1L,
  models = c("CR", "Logistic"),
  modelOutputsPrev = NULL,
  randomStarts = FALSE,
  lowerBounds = TRUE,
  upperBounds = TRUE,
  nbWorkers = 1L
)
```

Arguments

sp species name – only used for messaging.

predictorVarsData	a data.table of predictor variables including those in predictorVars and age, as well as pixelIndex. Note that age should be in the original scale (e.g., not logged).
sppVarsB	s data.table of species biomass (B) and pixelIndex.
predictorVars	character vector of predictor variables to be included in the linear component of the model affecting the asymptote (need to correspond to names(predictorVarsData[[sp]])) the same predictors will be considered for all species.
predictorVarsCombos	a list of sets of covariates in predictorVars to add to the fitted models. If this list has several entries with sets of covariates, each will be fitted as part of the model selection process.
maxNoCoefs	how many covariates from predictorVars should be added to the linear component of the model affecting the asymptote? Note that the more covariates are added the longer the model takes to fit, as all combinations are attempted. For 2 or more covariates, only combinations with "cover" are attempted.
doFwdSelection	should covariates be added one at a time to the linear component of the model? If TRUE, and is.null(predictorVarsCombos), then each entry in predictorVarsCombos is used as the set of covariates to test. Otherwise, predictorVarsCombos will be created from combinations of predictorVars, with maxNoCoefs determining the maximum number of covariates to add. If FALSE the full model is fitted.
sampleSize	how many data points should be randomly sampled to fit the model? If NA the full dataset will be used. Note that this may result in long computation times. Biomass data will be binned into 10 regular bins before sampling points in number equal to sampleSize.
Ntries	how many times should the models be fit with new randomly generated starting values? Only used if randomStarts == TRUE.
maxCover	numeric. Value indicating maximum cover/dominance.
models	character vector of models to fit. Only Chapman-Richards ('CR') and 'Logistic' can be chosen at the moment.
modelOutputsPrev	previous outputs of fitNLMmodels. The model will try refitting and comparing AIC with the last results.
randomStarts	logical. Should random starting values of A, k and p non-linear parameters be picked from a range sensible values, or should all combinations of values within this range be used? If FALSE, the default, the starting values are spaced at regular intervals within an acceptable range for each parameter – 20 values for A, 10 for k and p – and all combinations are used (2000 starting values in total). Parameter ranges are estimated from data following Fekedulegn et al. (1999) as follows: range of A starting values (B0 parameter in Fekedulegn et al. 1999) varies between $ObsMaxB \times 0.3$ and $ObsMaxB \times 0.9$, where $ObsMaxB$ is the maximum observed B across the full dataset (not the sampled data for fitting)

- k (CR model) and p (Logistic model; both are B2 parameter in Fekedulegn et al. 1999) are estimated as a constant rate to get to $\$ObsMaxB\$\$$, calculated as $\frac{B_{obs2}-B_{obs1}}{age2-age1}$, where B1/B2 and age1/age2 are observed values at two points in time. We draw 100 samples of two age values, and corresponding B, to calculate a sample of rates. After excluding rates ≤ 0 , we take the minimum and maximum values as the range of k (CR model) or p (Logistic model) parameters of the growth model
 - the p parameter (CR model) (related to B3 parameter in Fekedulegn et al. 1999) should be > 1 . Here we use a range of values between 1.1 and 80 which provided suitable fitting using data from the Northwest Territories, Canada
 - the k parameter (Logistic model; B1 parameter in Fekedulegn et al. 1999) is estimated as $B_0 = \frac{ObsMaxB}{1+k}$, using a small positive number for B_0 , e.g. 2. Here, we estimate k values for B_0 values 1 to 5, and use the minimum and maximum to determine the range from where to draw starting values.
- lowerBounds a named vector of lower parameter boundaries. If FALSE, no lower boundaries are applied. If TRUE, coefficients of the linear model on the A parameter (intercept, cover, k and p) are bound (intercept = observed maximum B * 0.5, cover = 0, k = 0.05 and p = 1). Alternatively, pass a named vector of parameter boundaries.
- upperBounds a named vector of upper parameter boundaries. If FALSE, no lower boundaries are applied. If TRUE, coefficient of the linear model on the A parameter (intercept and k) are bound (intercept = observed maximum B * 1.5, k = 0.2). Alternatively, pass a named vector of parameter boundaries.
- nbWorkers integer. If > 1 , the number of workers to use in `future::future_apply`, otherwise no parallelisation is done.

See Also

`bbmle::mle2()`

forestLandMask	<i>Is a pixel forest land?</i>
----------------	--------------------------------

Description

Forest land is a land *use*: ground that grows trees, including ground that has none right now because it burned or was harvested. The land-cover maps answer a different question – what covered the ground in one year – so a recent burn is "shrubs" there. Telling the temporarily open forest apart from land that is permanently open (a bog, a rock barren) needs a second source, and there are two to choose from.

Usage

```
forestLandMask(  
  lccList = list(),  
  faoRas = NULL,  
  treedClasses = .treedLCCClasses,  
  faoForestCodes = c(1L, 2L)  
)
```

Arguments

- lccList A list of land-cover SpatRasters, all aligned. May be empty.
- faoRas An FAO forest SpatRaster aligned with lccList, or NULL.
- treedClasses Land-cover classes that count as treed.
- faoForestCodes FAO codes that count as forest land. Both 1 and 2 are forest land; code 2 alone means "still an opening in the FAO layer's year".

Details

"fao" uses the NFIS FAO forest layer: code 1 is treed in the layer's own year, code 2 is not treed then but with a fire or harvest recorded since 1984. It is one small layer. Its limit is its fixed year: a stand that was open in 1995 and had grown back by then is code 1, which the old rule (code 2 only) discarded, and openings whose disturbance predates 1984 are code 0.

"lccYears" calls a pixel forest land if it is treed in *any* of several land-cover years. That sees the recovered stands and the pre-1984 openings, at the cost of reading one land-cover layer per year.

"both" takes the union, and is the default.

Value

A SpatRaster of 1 (forest land) and 0, with no NAs.

gadm_canada	<i>Canada administrative boundaries</i>
-------------	---

Description

Canada administrative boundaries

Usage

```
gadm_canada(src = "stats_can", dst_path = tempdir())
```

Arguments

- src Character. One of "stats_can" or (if geodata package is installed) "geodata".
- dst_path Character specifying the path to download to.

generatePixelGroups *Add the correct pixelGroups to a pixelDataTable object*

Description

Generates unique groupings of a `data.table` object where one or more rows can all belong to the same `pixelIndex`. Pixel groups will be identical pixels based on unique combinations of columns.

Usage

```
generatePixelGroups(
  pixelDataTable,
  maxPixelGroup,
  columns = c("ecoregionGroup", "speciesCode", "age", "B")
)
```

Arguments

`pixelDataTable` A `data.table` with column-based descriptions. Must have a column called `pixelIndex`, which allows for multiple rows to be associated with a single pixel.

`maxPixelGroup` A length 1 numeric indicating the current maximum `pixelGroup` value; the `pixelGroup` numbers returned will start at `maxPixelGroup + 1`.

`columns` A character vector of column names to use as part of the generation of unique combinations of features. Default is `c("ecoregionGroup", "speciesCode", "age", "B")`

Value

Returns a vector of `pixelGroup` in the original order of the input `pixelDataTable`. This should likely be added to the `pixelDataTable` object immediately.

genericExtract *Generic extract points*

Description

Extracts points from raster layers using the original raster layer projection.

Usage

```
genericExtract(x, y, field = NULL, ...)
```

Arguments

x	a raster or polygon object (sp, raster or terra)
y	a points or polygons spatial object (sp, sf, or terra)
field	character. The field(s) to extract when x is a polygon. If NULL, all fields are extracted and returned. IDs of y are always returned (ID column).
...	passed to <code>terra::extract()</code>

Details

If x and y are both polygons, extract often outputs NA due to misalignments (this can happen even when $x == y$), even after `snap(y, x)`. To circumvent this problem, `intersect` is used internally and, if the `extract` argument `fun` is passed, it is applied to values of y per polygon ID of x.

Value

a `data.table` with extracted values and an ID column of y point IDs

genPGsPostDisturbance *Re-generate new pixelGroups in partially disturbed pixels.*

Description

Re-generate new pixelGroups in partially disturbed pixels.

Usage

```
genPGsPostDisturbance(
  cohortData,
  pixelGroupMap,
  disturbedPixelTable,
  disturbedPixelCohortData,
  colsForPixelGroups = LandR::columnsForPixelGroups(),
  doAssertion = getOption("LandR.assertions", TRUE)
)
```

Arguments

cohortData	<code>data.table</code> . The pre-disturbance cohortData table
pixelGroupMap	<code>SpatRaster</code> . The pre-disturbance pixelGroupMap.
disturbedPixelTable	<code>data.table</code> . A table with at least the <code>pixelIndex</code> of all disturbed pixels. Additional columns are ignored.
disturbedPixelCohortData	a cohortData-like table with information of dead, and surviving, but <i>NOT</i> regenerating cohorts (cohorts for whom regeneration via, e.g., serotiny or resprouting was successfully activated), in <i>disturbed pixels only</i> . Dead cohorts should <code>age B == 0</code> , surviving cohorts <code>B > 0</code> .

colsForPixelGroups	character. cohortData columns used to find identical pixels in terms of cohort composition, and group them into pixelGroups (default: c("ecoregionGroup", "speciesCode", "age", "B")).
doAssertion	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: getOption("LandR.assertions", TRUE).

Details

This function regenerates pixelGroups in situations where disturbances are not stand-replacing and create survivor and dead cohorts in some, but potentially not all, pixels of a pixelGroup. This is necessary to prevent reintroducing dead cohorts that were not affected in the other pixels of the same, original, pixelGroup. **ATTENTION** This function alone will not generate final pixelGroups, and will likely need to be followed by an updateCohortData run. The function:

1. creates a table of undisturbed pixels, and disturbed pixels with dead and surviving cohorts but not new cohorts (e.g. from serotiny/resprouting) – these are added by updateCohortData;
2. generates pixelGroups considering these dead and surviving cohorts – this ensures that we account for cohorts that died in some but not all pixels of a given pixelGroup;
3. then removes dead cohorts (as they should not be in tables for downstream functions like updateCohortData) and recalculates pixelGroups – this ensures that pixels that became similar/dissimilar after the death of some cohorts can form new pixelGroups.

Value

a named list with:

- a cohortData table with the updated pixelGroups, as well as survivor cohorts, but not dead cohorts.
- a pixelGroupMap with the updated pixelGroups in disturbed pixels

getSpeciesTable	<i>Download and prepare a species traits table for use with Biomass_core module</i>
-----------------	---

Description

prepSpeciesTable

Usage

```
getSpeciesTable(url = NULL, dPath = tempdir(), cacheTags = NULL)

prepSpeciesTable(
  speciesTable,
  speciesLayers = NULL,
  sppEquiv = NULL,
```

```
sppEquivCol = "LandR",
areas = c("BSW", "BP", "MC")
)
```

Arguments

url	If NULL (the default), uses one from D. Cyr's LANDIS-II files: https://github.com/dcyr/LANDIS-II_IA_generalUseFiles/blob/master/speciesTraits.csv .
dPath	The destination path.
cacheTags	User tags to pass to Cache.
speciesTable	A species traits table, with at least the following columns: • speciesCode an character representation of species; • Area a character of geographical area within a species range for which trait values are relevant; • longevity species longevity in years, sexualmature age in years at sexual maturity; • shadetolerance numeric value between 1-5 of relative shade tolerance (with respect to other species); • seeddistance_eff the "effective" seed dispersal distance; • seeddistance_max a numeric with the maximum seed dispersal distance; • mortalityshape and growthcurve: growth curve shape parameters. Other columns (e.g. fire-related traits) can also be included depending on LandR modules in use. Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.
speciesLayers	Deprecated.
sppEquiv	table with species name equivalencies between naming conventions (see sppEquiv , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR:sppEquivalencies_CA.
sppEquivCol	the column name to use from sppEquiv (see sppEquiv). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR:sppEquivalencies_CA that best matches the species names.
areas	A character vector of areas to use. Can be one or more of c("Acadian", "AM", "NorthShore", "BP", "BSE", "BSW", "LSJ", "MC", "PM", "WestON"). If it is more than one, this function will take the minimum value, within a species. These are short versions of the Canada Ecoprovinces. Currently defaults to c("BSW", "BP", "MC") for historical reasons.

Value

getSpeciesTable() returns a data.table with columns LandisCode, Area, Longevity, Maturity, Shade, Fire, SeedEffDist, SeedMaxDist, VegProb, MinAgeVeg, MaxAgeVeg, PostFireRegen, LeafLongevity, WoodDecayRate, MortalityCurve, GrowthCurve, LeafLignin, HardSoft. See [assertSpeciesTableRaw\(\)](#) for expected column data types.

prepSpeciesTable() returns a data.table with columns species, Area, longevity, sexualmature, shadetolerance, firetolerance, seeddistance_eff, seeddistance_max, resproutprob, resproutage_min, resproutage_max, postfireregen, leaflongevity, wooddecayrate, mortalityshape, growthcurve, leafLignin, hardsoft. See [assertSpeciesTable\(\)](#) for expected column data types.

Note

This one is tailored to Canadian forests

See Also

[assertSpeciesTableRaw\(\)](#)

[assertSpeciesTable\(\)](#)

getWildfire_NFI

Download and prepare raster fire data from NFI

Description

Download and prepare raster fire data from NFI

Usage

```
getWildfire_NFI(dPath, rasterToMatch, url = NULL)
```

Arguments

dPath	destination path for archive
rasterToMatch	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
url	location from which to download the wildfire raster layer(s)

Value

a raster with values representing fire year 1985-2015

ggplotMLL_maxB

*Plot estimated maximum biomass by age***Description**

Plots a maximum biomass estimated at maximum 'cover' (or dominance) levels as a function of age.

Usage

```
ggplotMLL_maxB(
  m11,
  data,
  maxCover = 1L,
  xCovar = "age",
  plotTitle = NULL,
  nonLinModelQuoted,
  linModelQuoted,
  averageCovariates = TRUE,
  observedAge = FALSE,
  plotCIs = TRUE
)
```

Arguments

m11	a named list with outputs of an <code>bbmle::mle2</code> call (the fitted non-linear model), from which coefficient values will be extracted. If several model outputs are provided all fitted models will be plotted, with plot labels corresponding to list names.
data	data for estimation of maximum biomass. Should contain at least an 'age' column. Note that other covariates will be averaged and 'cover' values will be replaced with the maximum cover value (<code>maxCover</code>). If <code>m11</code> is a list data is assumed to be the same for the two models.
maxCover	numeric. Value indicating maximum cover/dominance.
xCovar	the variable shown in the x axis. Defaults to age.
plotTitle	character. Passed to title in <code>ggplot2::labs()</code> .
nonLinModelQuoted	a named list of non-linear equations as a call (quoted expression) passed to <code>mle2(minuslog1)</code> . See <code>bbmle::mle2()</code> . Accepts equations with three parameters 'A', 'p' and 'k'. List names and length must the same as in <code>m11</code> .
linModelQuoted	A named list of lists of linear equations/modes relating each parameter ('A', 'p' and 'k') with a set of covariates. A call (quoted expression) passed to <code>mle2(..., parameters)</code> . Note that for the purpose of tree growth, the linear equation determining 'A' should include a 'cover' predictor indicating the tree cover or dominance in the stand. Should be scaled between 0 and <code>maxCover</code> . List names and length must the same as in <code>m11</code> .

averageCovariates	should covariates other than age/cover be averaged for biomass predictions? If not, for each age (at maximum cover) there will be as many predictions as other covariate values. If observedAge == TRUE and averageCovariates == FALSE then the original data is used, with cover changed to maxCover.
observedAge	should observed age values be used, or should these be generated as <code>round(seq(min(age), max(age)*1.5, length.out = 100), 0)</code> ? If observedAge == TRUE and averageCovariates == FALSE then the original data is used, with cover changed to maxCover.
plotCIs	should confidence intervals be calculated and plotted?

```
imputeBadAgeModelDefault
```

Default model for imputing missing stand ages

Description

The model used by `makeAndCleanInitialCohortData()` (and, by default, by `Biomass_borealDataPrep`'s `imputeBadAgeModel` parameter) to impute the age of stands with missing or unreliable age data. The response is $\log(\text{age})$, not age, so a prediction can never come back negative – an imputed age of 0 with positive biomass, which `CBMutils::cumPoolsCreateAGB()` rejects, is not reachable from a log-scale model. This is the only place the formula is written; `Biomass_borealDataPrep` uses it as its parameter default rather than duplicating it.

Usage

```
imputeBadAgeModelDefault()
```

Details

The model is $\log(\text{age}) \sim \log(\text{totalBiomass}) * \text{cover} * \text{speciesCode} + (\log(\text{totalBiomass}) | \text{initialEcoregionCode})$. `makeAndCleanInitialCohortData()` recognizes a $\log(\text{age})$ response and back-transforms predictions with `exp()`. That gives the median (geometric-mean) age for given predictors, not the mean: with residual variance σ^2 on the log scale, the mean is larger by a factor of about $e^{\sigma^2/2}$. Imputed ages are therefore somewhat younger, on average, than observed ages of stands with the same biomass, cover and species; keep this in mind when comparing imputed and observed age distributions.

Value

A quoted `lme4::lmer()` call.

installBioSIM	<i>Install BioSIM to retrieve climate and other projections using BioSIM</i>
---------------	---

Description

Install **BioSIM** to retrieve climate and other projections using BioSIM

Usage

```
installBioSIM(lib)
```

Arguments

lib	character vector giving the library directories where to install the packages. Recycled as needed. If missing, defaults to the first element of <code>.libPaths()</code> .
-----	--

LANDISDisp	<i>Simulate a LANDIS-II dispersal process on a landscape.</i>
------------	---

Description

Simulate seed dispersal using user defined function. This is a "receiving pixel" focused dispersal approach. It is the "potentially receiving" cell that looks around itself for potential seed sources. If it finds a single seed source, that passes the probability function described by the `dispersalFn`. If this passes a comparison to a uniform random draw, then the receiving cell is deemed to have a "successful" dispersal for that species. This function can therefore only be used for a relatively specific situation where there is a yes/no returned for each potential receiving cell, i.e., not abundance. This function is also not cumulative, i.e., there is no higher abundance of seeds received if a receiving cell has lots of seed sources around it vs. a single seed source. The difference will come with a higher probability of successfully receiving a "seed".

Usage

```
LANDISDisp(
  dtSrc,
  dtRcv,
  pixelGroupMap,
  speciesTable,
  dispersalFn = Ward,
  b = 0.01,
  k = 0.95,
  plot.it = FALSE,
  successionTimestep,
  verbose = getOption("LandR.verbose", TRUE),
  useCpp = getOption("LandR.LANDISDisp.useCpp", TRUE),
```

```

    pgv = NULL,
    ...
)

```

Arguments

dtSrc	data.table
dtRcv	data.table
pixelGroupMap	A RasterLayer with pixel values equal to a pixel group number that corresponds exactly to pixelGroup column in cohortData.
speciesTable	A species traits table, with at least the following columns: • speciesCode an character representation of species; • Area a character of geographical area within a species range for which trait values are relevant; • longevity species longevity in years, sexualmature age in years at sexual maturity; • shadetolerance numeric value between 1-5 of relative shade tolerance (with respect to other species); • seeddistance_eff the "effective" seed dispersal distance; • seeddistance_max a numeric with the maximum seed dispersal distance; • mortalityshape and growthcurve: growth curve shape parameters. Other columns (e.g. fire-related traits) can also be included depending on LandR modules in use. Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.
dispersalFn	An expression that can take a "dis" argument. See details. Default is "Ward" (temporarily unused, as it is hard coded inside Rcpp function)
b	LANDIS Ward seed dispersal calibration coefficient (set to 0.01 in LANDIS)
k	LANDIS Ward seed dispersal the probability that seed will disperse within the effective distance (e.g., 0.95)
plot.it	Deprecated. If TRUE, then plot the raster at every interaction, so one can watch the LANDISDisp event grow.
successionTimestep	The time between successive seed dispersal events. In LANDIS-II, this is called "Succession Timestep".
verbose	Controls message output. Defaults to getOption("LandR.verbose")
useCpp	Logical. If TRUE (the default), the spiral seed-dispersal inner loop runs in C++ via Rcpp . If FALSE, the R reference implementation is used instead. Output is bit-identical between the two under a fixed seed. Defaults to getOption("LandR.LANDISDisp.useCpp", TRUE).
pgv	Optional integer vector of pixelGroup IDs, one per cell of pixelGroupMap (length(pgv) == ncell(pixelGroupMap)), NA for cells that can neither source nor receive. When supplied, the raster values are not read (reading a 9M-cell map costs over a second per call); pixelGroupMap is still used for its dimensions and resolution. When NULL (the default), the values are read from pixelGroupMap.
...	Additional parameters. Currently none.

Details

dispersalFn (temporarily unused as code is converted to Rcpp – the default dispersalFn is hard coded within the spiralSeedDispersal function that uses C++) must be an expression that returns a probability distribution. Because it is a dispersal kernel, it must be a probability distribution. The expression that can take an argument named "dis" (without quotes) as this will be calculated internally and represents the distance from the initial (receiving) pixel and all active pixels within that cluster of active pixels. **LandR** includes the `Ward()` kernel as defined in the LANDIS-II documentation.

The spiral inner loop is implemented in C++ (via **Rcpp**) and is on by default; it produces bit-identical output to the previous R implementation under a fixed seed, but is typically ~3x faster end-to-end on landscape-scale inputs (and uses dramatically less memory by replacing the per-cell-by-species source matrix with a per-pixelGroup species bitmask). To fall back to the R reference (e.g., for debugging), pass `useCcpp = FALSE` or set `options(LandR.LANDISDisp.useCcpp = FALSE)`.

Value

A numeric vector of raster pixel indices, in the same resolution and extent as seedSrc raster.

Author(s)

Eliot McIntire

Examples

```
if (require("googledrive")) {
  seed <- sample(1e6, 1)
  set.seed(seed)
  library(data.table)

  # keep this here for interactive testing with a larger raster
  rasterTemplate <- reproducible::rasterRead(terra::ext(0, 2500, 0, 2500), res = 100)

  # make a pixelGroupMap
  pgs <- 4 # make even just because of approach below requires even
  pixelGroupMap <- SpaDES.tools::randomPolygons(rasterTemplate, numTypes = pgs)
  pixelGroupMap[1:100] <- NA # emulate a mask at the start

  # Make a receive pixels table -- need pixelGroup and species
  nSpecies <- 3
  maxNSpeciesPerPixel <- min(5, nSpecies)
  rcvSpByPG <- lapply(seq_len(pgs / 2), function(pg) {
    data.table(speciesCode = sample(nSpecies, size = sample(maxNSpeciesPerPixel, 1)))
  })
  seedReceive <- rbindlist(rcvSpByPG, idcol = "pixelGroup")

  # Make a source pixels table -- need pixelGroup and species
  srcSpByPG <- lapply(seq_len(pgs / 2), function(pg) {
    data.table(speciesCode = sample(nSpecies, size = sample(maxNSpeciesPerPixel, 1)))
  })
  seedSource <- rbindlist(srcSpByPG, idcol = "pixelGroup")
}
```

```

# make source pixels not same pixelGroups as receive
seedSource[, pixelGroup := pixelGroup + pgs / 2]

# Get a species table -- if using in Canada, can use this
destPath <- file.path(tempdir(), "ex_LANDISDisp")
speciesTable <- getSpeciesTable(dPath = destPath)
speciesTable <- speciesTable[Area == "BSW"]
speciesTable[, speciesCode := as.factor(LandisCode)]
speciesTable[, seeddistance_eff := SeedEffDist]
speciesTable[, seeddistance_max := SeedMaxDist]

speciesTable <- speciesTable
speciesTable <- data.table(speciesTable)[, speciesCode := seq_along(LandisCode)]
seedReceiveFull <- speciesTable[seedReceive, on = "speciesCode"]
output <- LANDISDisp(
  dtRcv = seedReceiveFull, plot.it = interactive(),
  dtSrc = seedSource,
  speciesTable = speciesTable,
  pixelGroupMap,
  verbose = TRUE,
  successionTimestep = 10
)
# Summarize
output[, .N, by = speciesCode]

## Plot the maps
if (interactive()) {
  spMap <- list()
  spMap$pixelGroupMap <- pixelGroupMap
  for (sppp in unique(output$speciesCode)) {
    spppChar <- paste0("Sp_", sppp)
    spMap[[spppChar]] <- reproducible::rasterRead(pixelGroupMap)
    ss <- unique(seedSource[speciesCode == sppp], on = c("pixelGroup", "speciesCode"))
    spMap[[spppChar]][pixelGroupMap[] %in% ss$pixelGroup] <- 1

    receivable <- reproducible::rasterRead(pixelGroupMap)
    srf <- unique(seedReceiveFull[speciesCode == sppp], on = c("pixelGroup", "speciesCode"))
    receivable[pixelGroupMap[] %in% srf$pixelGroup] <- 1

    forest <- which(!is.na(pixelGroupMap[]))
    src <- which(!is.na(spMap[[spppChar]][]))
    recvable <- which(!is.na(receivable[]))
    rcvd <- output[speciesCode == sppp][["pixelIndex"]]

    spMap[[spppChar]][forest] <- 0
    spMap[[spppChar]][recvable] <- 2
    spMap[[spppChar]][src] <- 1
    spMap[[spppChar]][rcvd] <- 3
    spMap[[spppChar]][intersect(src, rcvd)] <- 4

    levels(spMap[[spppChar]]) <- data.frame(
      ID = 0:4,
      type = c(

```

```

        "OtherForest", "Source", "Didn't receive",
        "Received", "Src&Rcvd"
    )
}
terra::plot(terra::rast(spMap))

# A summary
rr <- apply(rast(spMap)[[-1]][,] + 1, 2, tabulate)
rownames(rr) <- raster::levels(spMap[[2]])[[1]][, "type"][1:NROW(rr)]
# next line only works if there are some places that are both source and potential to receive
# rr <- rbind(rr, propSrcRcvd = round(rr[5,]/ (rr[5,]+rr[2,]), 2))
}
}

```

LandROptions

LandR *options*

Description

These provide top-level settings for LandR functions and the LandR SpaDES modules. To see defaults, run `LandROptions()`. See Details below.

Usage

```
LandROptions()
```

Details

Below are options that can be set with `options("LandR.xxx" = newValue)`, where `xxx` is one of the values below, and `newValue` is a new value to give the option. Sometimes these options can be placed in the user's `.Rprofile` file so they persist between sessions.

`assertions` Default: TRUE. If TRUE, additional code checks are run during function calls; see [assertions](#).

`forestLandUseCache` Default: unset, which means the forest-land inputs of [prepInputs_SCANFI_LCC_FA0\(\)](#) and [prepInputs_NTEMS_LCC_FA0\(\)](#) – the FAO forest layer and each forestLandYears land cover – are cached as `options("reproducible.useCache")` says, like any other `Cache()` call. Set it to "always" to cache them even when that option is FALSE, so a caller that prepares several years of one study area prepares each input once: `fireSense` runs under `spades.useCache = "eventsOnly"`, which sets `reproducible.useCache = FALSE`. Any other value is passed to `Cache(useCache = )`.

`leadingSpeciesProp` Default: unset, which means it takes the value of `mixedwoodProp`. The share ONE species must hold for the stand to be called its own rather than mixed. A user who wants "just a majority" can set it to 0.51 without moving the `mixedwood` definition. Read with [leadingSpeciesProp\(\)](#); used by [vegTypeMapGenerator\(\)](#) and [vegTypeGenerator\(\)](#) for `mixedType` other than 2.

- subsetDataSize** Default: 500L. The maximum number of rows `subsetDT()` keeps per group when subsampling data for model fitting, and the default of the LandR modules' `subsetData*Model` parameters. Read with `subsetDataSize()`. It was 50 for years, set when these fits were expensive; at that size repeated runs of the same simulation gave visibly different maxB.
- mergeHybridSpruce** Default: "engelmann". Which species the hybrid white x Engelmann spruce (`Pice_eng_gla`) is merged into in the `sppEquiv` returned by `speciesInStudyArea()`: "engelmann" (`Pice_eng`), "white" (`Pice_gla`), or NA for no merging.
- mixedwoodProp** Default: 0.75. The share of a stand held by a GROUP – all conifers, or all broadleaves – at or above which the stand stops being mixedwood. This is the definition the national products use: NTEMS/EOSD (Wulder & Nelson 2003) and the NFI photo plot dictionary call a stand coniferous or broadleaf at 75% or more of total basal area (photo plots: total tree volume), mixed wood when neither group reaches it. Deciduous conifers count as conifers (`Larix` is `Type == "Conifer"` in `sppEquivalencies_CA`). Read with `mixedwoodProp()`; used by `vegTypeMapGenerator()` and `vegTypeGenerator()` with `mixedType = 2`, `lccMapGenerator()` and `plotVTM()`.
- scanfiMirror** Default: TRUE. When LandR is loaded and no `reproducible.urlRemap` is set, fetch SCANFI from the PredictiveEcology arbutus mirror instead of Google Drive, so the ids that 404 for anonymous users resolve and no login is needed; see `scanfiUrlRemap()`.
- verbose** Default: 1. The default verbose argument of functions that report their progress, e.g., `updateCohortData()` and `LANDISDisp()`. Higher numbers give more messages.

Value

A named list of the LandR options and their default values, as set by `.onLoad()` when the package is loaded. Options the user has already set are not overwritten.

<code>lccMapGenerator</code>	<i>Calculate landcover classes based on species cohorts</i>
------------------------------	---

Description

Convert `cohortData` into landcover class based on leading species and density.

1. Classify each pixel as "broadleaf", "mixedwood", or "conifer";
2. Define the openness of each pixel as "dense", "open", or "sparse";
3. Combine these into cover classes for forested cells.

Usage

```
lccMapGenerator(
  cohortData,
  pixelGroupMap,
  lccTable = NULL,
  deciduousCoverDiscount = 0.8418911,
  vegLeadingProportion = mixedwoodProp(),
  deciduousSpp = NULL,
  rstLCC
)
```

Arguments

cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPPAct, and sumB.
pixelGroupMap	A RasterLayer with pixel values equal to a pixel group number that corresponds exactly to pixelGroup column in cohortData.
lccTable	data.table (with columns leading and LCC) defining the mappings between leading species and landcover classes.
deciduousCoverDiscount	numeric between 0 and 1 that translates %% cover to %% biomass. It assumes all hardwoods are equivalent; all softwoods are equivalent; and that %% cover of hardwoods will be an overestimate of the %%% biomass of hardwoods. Hardwoods in Canada have a much wider canopy than softwoods. E.g., 30%% cover of hardwoods might translate to 20%% biomass of hardwoods. NB: the default deciduousCoverDiscount value was estimated from NWT data in March 2020, and may not be useful for other study areas.
vegLeadingProportion	Numeric between 0-1, the share of a stand above which it stops being called mixed. Which share depends on what is being asked, and the two questions have their own options (see leadingProportions): with mixedType = 2 it is the whole broadleaf group against the whole conifer group, and defaults to mixedwoodProp() (LandR.mixedwoodProp, the NTEMS/EOSD and NFI value: coniferous or broadleaf at 75% or more of total basal area, mixed wood when neither reaches it); otherwise it is one species' share and defaults to leadingSpeciesProp() (LandR.leadingSpeciesProp, which takes the mixedwood value unless it is set). Deciduous conifers count as conifers.
deciduousSpp	character vector of deciduous species names. If not supplied, will attempt to match deciduous species names in sppEquivalencies_CA to those in cohortData.
rstLCC	landcover raster

Details

By default, landcover classes are coded to match the Land Cover Map of Canada 2005 at 250m (Latifovic & Pouliot, 2005).

Code modified from https://github.com/tati-micheletti/caribouRSF_NT/blob/main/R/makeLCCfromCohortData.R.

Value

raster of the same type as pixelGroupMap, containing LCC values

Note

Sparse classes: Even though the default LCC2005 input does *not* have "mixed sparse" or "broadleaf sparse", these are likely to be produced in simulations, starting from "conifer sparse" sites (i.e. broadleaf trees might grow on these sites, and eventually convert these into mixed or broadleaf sparse sites).

Author(s)

Tati Micheletti and Alex Chubaty

References

Latifovic, R. & Pouliot, D. (2005). Multitemporal land cover mapping for Canada: methodology and products. *Canadian Journal of Remote Sensing*, 31, 347–363.

leadingByPixelGroup	<i>Leading species of each pixelGroup</i>
---------------------	---

Description

The one place where the leading species (or "Mixed") of every pixelGroup in a cohortData table is worked out. [vegTypeGenerator\(\)](#) and [vegTypeMapGenerator\(\)](#) use it, and so can a caller that only needs the result per pixelGroup, which avoids the join back onto the (much longer) cohortData that [vegTypeGenerator\(\)](#) does.

Usage

```
leadingByPixelGroup(
  x,
  vegLeadingProportion = NULL,
  mixedType = 2,
  sppEquiv = NULL,
  sppEquivCol,
  pixelGroupColName = "pixelGroup",
  leadingBasedOn = if ("B" %in% colnames(x)) "B" else "cover"
)
```

Arguments

x	A cohortData object, with B (or cover) and speciesCode.
vegLeadingProportion	Numeric between 0-1, the share of a stand above which it stops being called mixed. Which share depends on what is being asked, and the two questions have their own options (see leadingProportions): with mixedType = 2 it is the whole broadleaf group against the whole conifer group, and defaults to mixedwoodProp() (LandR.mixedwoodProp, the NTEMS/EOSD and NFI value: coniferous or broadleaf at 75% or more of total basal area, mixed wood when neither reaches it); otherwise it is one species' share and defaults to leadingSpeciesProp() (LandR.leadingSpeciesProp, which takes the mixedwood value unless it is set). Deciduous conifers count as conifers.
mixedType	An integer defining whether mixed stands are: not differentiated (0), any kind of species admixture (1), or deciduous mixed with conifer (2; default).

sppEquiv	table with species name equivalencies between naming conventions (see sppEquiv , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
sppEquivCol	the column name to use from sppEquiv (see sppEquiv). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.
pixelGroupName	Name of the column in x identifying the pixelGroup.
leadingBasedOn	"B" (default if x has a B column) or "cover".

Details

With mixedType = 2, B-based leading type and a factor speciesCode (what cohortData normally has), the species sums per pixelGroup are made in one pass on an integer key and the broadleaf proportion of the (few) pixelGroups with two or more deciduous species is summed per group with `base::sum()`, exactly as `.isMixedwood()` does. Any other case (mixedType 0 or 1, cover, a character speciesCode, NA in B, a species with two Types in sppEquiv) takes the original, slower algorithm. Both give identical results.

Value

A data.table with one row per pixelGroup: the pixelGroupName column, leading (a factor, or speciesCode when mixedType = 0) and, for mixedType 1 and 2, mixed.

Author(s)

Eliot McIntire, Ceres Barros, Alex Chubaty

Examples

```
library(data.table)
x <- data.table(
  pixelGroup = rep(1:2, each = 2), B = c(100, 200, 20, 400),
  speciesCode = factor(rep(c("Pice_gla", "Popu_tre"), 2))
)
sppEquiv <- data.table(LandR = c("Pice_gla", "Popu_tre"), Type = c("Conifer", "Deciduous"))
leadingByPixelGroup(x, sppEquiv = sppEquiv, sppEquivCol = "LandR")
```

loadCASFRI	<i>Load CASFRI data</i>
------------	-------------------------

Description

Loads CASFRIv4 data specifically supplied for the LandWeb project

Usage

```
loadCASFRI(
  CASFRIras,
  attrFile,
  headerFile,
  sppEquiv,
  sppEquivCol,
  type = c("cover", "age")
)
```

Arguments

CASFRIras	RasterLayer object
attrFile	character string specifying a CASFRI attribute filename
headerFile	character string specifying a CASFRI header filename
sppEquiv	table with species name equivalencies between naming conventions (see sppEquiv , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
sppEquivCol	the column name to use from sppEquiv (see sppEquivalencies_CA). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.
type	Character string. Either "cover" or "age".

Value

list with named elements: CASFRIattrLong and CASFRIIdt (both data.tables)

loadkNNSpeciesLayers	<i>Load kNN species layers from online data repository</i>
----------------------	--

Description

TODO: description needed

Usage

```
loadkNNSpeciesLayers(
  dPath,
  rasterToMatch = NULL,
  studyArea = NULL,
  sppEquiv,
  year = 2001,
  knnNamesCol = "KNN",
```

```

    sppEquivCol = "Boreal",
    thresh = 10,
    url = NULL,
    ...
)

```

Arguments

dPath	path to the data directory
rasterToMatch	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
studyArea	A SpatVector or sf polygons object used as the principle study region, passed to <code>reproducible::prepInputs()</code> .
sppEquiv	table with species name equivalencies between naming conventions (see sppEquivalencies_CA , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
year	which year's layers should be retrieved? One of 2001 (default) or 2011.
knnNamesCol	character string indicating the column in sppEquiv containing kNN species names. Default "KNN" for when sppEquivalencies_CA is used.
sppEquivCol	the column name to use from sppEquiv (see sppEquivalencies_CA). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.
thresh	the minimum percent cover a species must have (per pixel) to be considered present in the study area. Defaults to 10.
url	the source url for the data, default is KNN 2011 dataset (https://ftp.maps.canada.ca/pub/nrcan_rncan/Forests_Foret/canada-forests-attributes_attributes-forests-canada/2011-attributes_attributes-2011/)
...	Additional arguments passed to <code>reproducible::Cache()</code> and <code>equivalentName()</code> . Also valid: outputPath, and studyAreaName.

Value

A raster stack of percent cover layers by species.

loadSCANFISpeciesLayers

Load SCANFI species layers from online data repository

Description

Load SCANFI species layers from online data repository

Usage

```
loadSCANFISpeciesLayers(
  dPath,
  to = NULL,
  cropTo = NULL,
  projectTo = NULL,
  maskTo = NULL,
  sppEquiv,
  year = 2020,
  dataVersion = "V2",
  SCANFINamesCol = "SCANFI",
  sppEquivCol = "SCANFI",
  thresh = 10,
  url = NULL,
  ...
)
```

Arguments

dPath	path to the data directory
to, cropTo, projectTo, maskTo	passed to reproducible::prepInputs() ; see reproducible::postProcessTo() . The legacy rasterToMatch and studyArea are still accepted through ... and are translated onto these by .legacyToTo(), following the table in reproducible::postProcess() : a rasterToMatch on its own is to; a studyArea on its own crops and masks but does not reproject (unless useSAcrs); and when both are supplied the raster gives extent, resolution, projection and alignment while the polygon gives the mask. An argument passed here explicitly always wins.
sppEquiv	table with species name equivalencies between naming conventions (see sppEquiv , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
year	data year for SCANFI data. 2000, 2010, and 2020 possible for V1. 1985, 1990, 1995, 2000, 2005, 2010, 2015, 2020 (default), 2025 possible for V2.
dataVersion	Character. SCANFI product version for data. Default is currently V2. V1 also available.
SCANFINamesCol	character string indicating the column in sppEquiv containing SCANFI species names. Default "SCANFI" for when sppEquivalencies_CA is used.
sppEquivCol	the column name to use from sppEquiv (see sppEquivalencies_CA). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.
thresh	the minimum percent cover a species must have (per pixel) to be considered present in the study area. Defaults to 10.
url	a source url for the data
...	Additional arguments passed to reproducible::Cache() and equivalentName() . Also valid: outputPath, and studyAreaName.

Value

A raster stack of percent cover layers by species.

```
makeAndCleanInitialCohortData
```

Generate initial cohortData table

Description

Takes a single `data.table` input, which has the following columns in addition to others that will be labelled with species name, and contain percent cover of each:

Usage

```
makeAndCleanInitialCohortData(
  inputDataTable,
  sppColumns,
  imputeBadAgeModel = imputeBadAgeModelDefault(),
  minCoverThreshold,
  doAssertion = getOption("LandR.assertions", TRUE),
  doSubset = TRUE,
  minSpeciesEcoregionShare = 0
)
```

Arguments

<code>inputDataTable</code>	A <code>data.table</code> with columns described above.
<code>sppColumns</code>	A vector of the names of the columns in <code>inputDataTable</code> that represent percent cover by species, rescaled to sum up to 100%%.
<code>imputeBadAgeModel</code>	statistical model used to impute ages in pixels with missing data or with cover == 0. If set to NULL no imputation will be attempted, and pixels with missing age are excluded.
<code>minCoverThreshold</code>	minimum total cover percentage necessary to consider the pixel vegetated, or a cohort present in a pixel.
<code>doAssertion</code>	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: <code>getOption("LandR.assertions", TRUE)</code> .
<code>doSubset</code>	Turns on/off subsetting. Defaults to TRUE.
<code>minSpeciesEcoregionShare</code>	Minimum share (0-1) of an ecoregion's pixels in which a species must have cover above <code>minCoverThreshold</code> to be kept in that ecoregion. Below it, the species' cohorts are removed from every pixel of the ecoregion (their cover goes to the other species when cover is rescaled), so the species is also absent when <code>establishprob</code> , <code>maxB</code> and <code>maxANPP</code> are estimated there. The ecoregion is <code>initialEcoregionCode</code> without its land-cover suffix. Default 0: no species is removed.

Details

- pixelIndex (integer)
- age (integer)
- logAge (numeric)
- initialEcoregionCode (factor)
- totalBiomass (integer)
- lcc (integer)
- rasterToMatch (integer)
- speciesCode (factor)
- cover (integer)
- coverOrig (integer)
- B (integer)

Several data correction/imputation operations are also performed. Namely, age is imputed in pixels where age data is missing (but not cover) and where cover == 0 but age > 0, total biomass is zeroed if age == 0, and age is zeroed if biomass == 0.

Value

a cohortData data.table with attribute "imputedPixID" (a vector of pixel IDs that suffered imputation).

Author(s)

Eliot McIntire

makeBiomassMap

Create biomassMap

Description

This is a function that creates the biomassMap raster used for simulations in Biomass_core module, using estimated data based on rawBiomassMap contained in pixelCohortData.

Usage

```
makeBiomassMap(pixelCohortData, rasterToMatch)
```

Arguments

pixelCohortData

The full cohortData data.table

rasterToMatch

A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.

Value

The biomassMap, a raster of total stand biomass per pixel.

makeCohortDataFiles	<i>Make the cohortData table, while modifying the temporary pixelCohortData that will be used to prepare other files.</i>
---------------------	---

Description

Takes a pixelCohortData table (see makeAndCleanInitialCohortData), the speciesEcoregion list and returns a modified pixelCohortData and the cohortData tables to be used in the simulation. This function mainly removes unnecessary columns from pixelCohortData, subsets pixels with biomass > 0, generates pixelGroups, and adds ecoregionGroup and totalBiomass columns to pixelCohortData. cohortData is then created by subsetting unique combinations of pixelGroup and whatever columns are listed in columnsForPixelGroups. The resulting cohortData table has the following columns:

- speciesCode (factor)
- ecoregionGroup (factor)
- pixelGroup (integer)
- age (integer)
- B (integer)

Usage

```
makeCohortDataFiles(
  pixelCohortData,
  columnsForPixelGroups,
  speciesEcoregion,
  pixelGroupBiomassClass,
  pixelGroupAgeClass,
  minAgeForGrouping = 0,
  rmImputedPix = FALSE,
  imputedPixID,
  pixelFateDT
)
```

Arguments

pixelCohortData	The full cohortData data.table
columnsForPixelGroups	Default columns that define pixel groups

speciesEcoregion	A data.table with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: speciesCode and ecoregionGroup, character representation of species and ecoregion groups respectively, maxB the maximum biomass for the species in a given 'ecoregion', maxANPP the maximum aboveground net primary productivity and SEP the species establishment probability. May contain columns inflationFactor (used to adjust maxB) and mANPPproportion (used to calculate maxANPP).
pixelGroupBiomassClass	Integer. When assigning pixelGroup membership, this defines the resolution of biomass that will be considered 'the same pixelGroup', e.g., if it is 100, then 5160 and 5240 will be the same
pixelGroupAgeClass	Integer. When assigning pixelGroup membership, this defines the resolution of ages that will be considered 'the same pixelGroup', e.g., if it is 10, then 6 and 14 will be the same.
minAgeForGrouping	Minimum age for regrouping. This may be because there is a source of ages for young stands/trees that is very reliable, such as a fire database. Ages below this will not be grouped together. Defaults to -1, meaning treat all ages equally. If this is related to known ages from a high quality database, then use age of the oldest trees in that database.
rmImputedPix	Should imputed pixels be removed
imputedPixID	a vector of IDs of pixels that suffered data imputation.
pixelFateDT	A data.table of pixelFateDT; if none provided, will make an empty one.

Value

A list with a modified pixelCohortData, cohortData, and pixelFateDT data.tables.

makeDummyEcoregionMap *Create dummy inputs for test simulations*

Description

ecoregionMap is a raster of all the unique groupings.

Usage

```
makeDummyEcoregionMap(rasterToMatch)
```

```
makeDummyRawBiomassMap(rasterToMatch)
```

```
makeDummyStandAgeMap(rawBiomassMap)
```

```
makeDummyRstLCC(rasterToMatch)

makeDummyEcoregionFiles(ecoregionMap, rstLCC, rasterToMatch)
```

Arguments

- rasterToMatch A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
- rawBiomassMap a rawBiomassMap (e.g. the one used throughout the simulation)
- ecoregionMap The ecoregionMap, a raster of all the unique ecoregion groupings, which group together pixels of similar biophysical characteristics.
- rstLCC raster of land-cover class, e.g., LCC2005

Details

rawBiomassMap is a raster of "raw" total stand biomass per pixel, with values between 100 and 20000 g/m^2.

standAgeMap is a raster of stand age per pixel (where biomass exists) with values between 1 and 300 years.

rstLCC is a raster land-cover class per pixel, with values between 1 and 5 that have no correspondence to any real land-cover classes.

ecoregionFiles uses dummy versions of ecoregionMap and rstLCC to create a list with two objects: the ecoregionMap and a table summarizing its information per pixelID. See ecoregionProducer (it uses ecoregionProducer internally).

Value

a RasterLayer object or, in the case of makeDummyEcoregionFiles, a list.

makeEcoregionDT	<i>Make the ecoregion table</i>
-----------------	---------------------------------

Description

This function creates a table containing pixel-wise ecoregion codes and whether they are "active" (have biomass > 0) or not for simulation. Unlike ecoregionProducer, this function creates the ecoregion table from pixel information contained in pixelCohortData

Usage

```
makeEcoregionDT(pixelCohortData, speciesEcoregion)
```

Arguments

- pixelCohortData The full cohortData data.table
- speciesEcoregion A data.table with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: speciesCode and ecoregionGroup, character representation of species and ecoregion groups respectively, maxB the maximum biomass for the species in a given 'ecoregion', maxANPP the maximum aboveground net primary productivity and SEP the species establishment probability. May contain columns inflationFactor (used to adjust maxB) and mANPPproportion (used to calculate maxANPP).

Value

A data.table with ecoregion codes and their active status per pixelID.

makeEcoregionMap	<i>Make the ecoregionMap raster</i>
------------------	-------------------------------------

Description

Creates a raster of ecoregion codes per pixel. Unlike ecoregionProducer, this fills the raster with pixel information contained in pixelCohortData.

Usage

makeEcoregionMap(ecoregionFiles, pixelCohortData)

Arguments

- ecoregionFiles A list with two objects: the ecoregionMap and a table summarizing its information per pixelID.
- pixelCohortData The full cohortData data.table

Value

A raster with ecoregion codes.

makeMinRelativeB	Create minRelativeB table
------------------	---------------------------

Description

The table contains expert-based values for minimum relative biomass of each shade tolerance class (the minimum relative biomass a cohort with a given shade tolerance should have to be able to germinate), in each unique ecoregion group. All ecoregion groups currently have the same values.

Usage

```
makeMinRelativeB(pixelCohortData)
```

Arguments

pixelCohortData
The full cohortData data.table

Value

a data.frame of min relative biomass values per ecoregion group.

makePickellStack	makePickellStack
------------------	------------------

Description

TODO: description and title needed

Usage

```
makePickellStack(PickellRaster, sppEquiv, sppEquivCol, destinationPath)
```

Arguments

PickellRaster TODO: description needed

sppEquiv table with species name equivalencies between naming conventions (see [sppEquivalencies_CA](#), also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.

sppEquivCol the column name to use from sppEquiv (see [sppEquivalencies_CA](#)). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.

destinationPath path to data directory where objects will be downloaded or saved to

Value

TODO: description needed

makePixelGroupMap	<i>Create makePixelGroupMap</i>
-------------------	---------------------------------

Description

Create the makePixelGroupMap raster containing pixelGroups in pixelCohortData.

Usage

```
makePixelGroupMap(pixelCohortData, rasterToMatch)
```

Arguments

pixelCohortData	The full cohortData data.table
rasterToMatch	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.

Value

a raster with pixel groups

makePixelTable	<i>Make pixelTable from biomass, age, land-cover and species cover data</i>
----------------	---

Description

Make pixelTable from biomass, age, land-cover and species cover data

Usage

```
makePixelTable(
  speciesLayers,
  standAgeMap,
  ecoregionFiles,
  biomassMap,
  rasterToMatch,
  rstLCC,
  printSummary = TRUE,
  doAssertion = getOption("LandR.assertions", TRUE)
)
```

Arguments

speciesLayers	A RasterStack or RasterLayer that should contain species cover data in the study area
standAgeMap	A rasterLayer of stand ages with attribute "imputedPixID"
ecoregionFiles	A list with two objects: the ecoregionMap and a table summarizing its information per pixelID. See ecoregionProducer.
biomassMap	raster of total stand biomass in t/ha. Biomass units are converted to g/m ² .
rasterToMatch	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
rstLCC	raster of land-cover class, e.g., LCC2005
printSummary	Logical. If TRUE, the default, a print out of the summary(pixelTable) will occur.
doAssertion	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: getOption("LandR.assertions", TRUE).

Value

A data.table as many rows as non-NA pixels in rasterToMatch and the columns containing pixel data from the input raster layers, with biomass in g/m².

makeSpeciesEcoregion *Create speciesEcoregion*

Description

Use statistically estimated maxB, maxANPP and establishment probabilities to generate speciesEcoregion table.

Usage

```
makeSpeciesEcoregion(
  cohortDataBiomass,
  cohortDataShort,
  cohortDataShortNoCover,
  species,
  modelCover,
  modelBiomass,
  successionTimestep,
  currentYear
)
```

Arguments

cohortDataBiomass	a subset of cohortData object
cohortDataShort	a subset of cohortData
cohortDataShortNoCover	a subset of cohortData
species	a data.table with species traits such as longevity, shade tolerance, etc. Must have column speciesCode, with species names/IDs. The following is a list of default trait columns: • "species" same as "speciesCode" – species ID name • "speciesCode" • "Area" – inherited from LANDIS-II default table, the Canadian ecoregion from which traits were derived. Not used during the simulation • "firetolerance" – modular <i>relative</i> (to other species) fire tolerance • "growthcurve" and "mortalityshape" – growth curve shape parameters. • "longevity" – maximum species age • "postfireregen" – post-fire regeneration strategy ("serotiny", "resprout" or "none") • "resproutprob" – probability of resprouting • "resproutage_min" – minimum age at which species is capable of resprouting • "resproutage_max" – maximum age at which species is capable of resprouting • "seeddistance_eff" – effective dispersal distance • "seeddistance_max" – maximum dispersal distance • "shadetolerance" – <i>relative</i> (to other species) shade tolerance • "sexualmature" – age at sexual maturity Known optional parameters added/needed by some modules (the user may add others for their own modules): • "inflationFactor" – Biomass_speciesParameters module: inflation factor for maxB • "growthCurveSource" – Biomass_speciesParameters module: how "growthcurve" was estimated • "mANPPproportion" – Biomass_speciesParameters module: multiplication factor to calculate maxANPP from maxB • "thermokarsttol" – Biomass_disturbances module: proportion of biomass surviving after thermokarst (i.e. permafrost thaw). Applied equally across cohorts. Parameters inherited from LANDIS-II default table, but not used in LandR at the moment: • "leaflongevity" • "wooddecayrate" • "leafLignin" • "hardsoft" Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.

modelCover	statistical model of species presence/absence
modelBiomass	statistical model of species biomass
successionTimestep	The time between successive seed dispersal events. In LANDIS-II, this is called "Succession Timestep".
currentYear	time(sim)

Details

See Details.

establishprob

This section takes the cover as estimated from the mature tree cover and partitions it between resprouting and seeds. Unfortunately, establishment by seed is not independent of resprouting, i.e., some pixels would have both. Since we don't know the level of independence, we can't correctly assess how much to discount the two. If there is resprouting > 0, then this is the partitioning: $\text{establishprob} = f(\text{establishprob} + \text{resproutprob} + \text{jointEstablishProbResproutProb})$. If $\text{jointEstablishProbResproutProb}$ is 0, then these are independent events and the total cover probability can be partitioned easily between seeds and resprout. This is unlikely ever to be the case. We are picking 50% overlap as a number that is better than 0 (totally independent probabilities, meaning no pixel has both seeds and resprout potential) and 100% overlap (totally dependent probabilities, i.e., every pixel where there is seeds will also be a pixel with resprouting). This is expressed with the `"* 0.5"` in the code.

#' @return A speciesEcoregion data.table with added columns for parameters `maxB`, `maxANPP` and `establishprob`

makeVegTypeMap	<i>Make a vegetation type map from a stack of species abundances</i>
----------------	--

Description

makeVegTypeMap is a wrapper around vegTypeMapGenerator that works from a species stack of percent cover. These do not have to sum to 100%.

Downloads the 2011 kNN species cover layers from the Canadian Forestry Service, National Inventory System, for validation.

Usage

```
makeVegTypeMap(speciesStack, vegLeadingProportion, mixed, ...)
```

```
loadkNNSpeciesLayersValidation(
  dPath,
  rasterToMatch,
  studyArea,
  sppEquiv,
```

```

    knnNamesCol = "KNN",
    sppEquivCol,
    thresh = 1,
    url,
    ...
)

prepSpeciesLayers_KNN2011(
  destinationPath,
  outputPath,
  url = NULL,
  studyArea,
  rasterToMatch,
  sppEquiv,
  sppEquivCol,
  thresh = 10,
  ...
)

```

Arguments

speciesStack	A RasterStack of species abundances. This must be one RasterLayer per species.
vegLeadingProportion	Numeric between 0-1, the share of a stand above which it stops being called mixed. Which share depends on what is being asked, and the two questions have their own options (see leadingProportions): with mixedType = 2 it is the whole broadleaf group against the whole conifer group, and defaults to mixedwoodProp() (LandR.mixedwoodProp, the NTEMS/EOSD and NFI value: coniferous or broadleaf at 75% or more of total basal area, mixed wood when neither reaches it); otherwise it is one species' share and defaults to leadingSpeciesProp() (LandR.leadingSpeciesProp, which takes the mixedwood value unless it is set). Deciduous conifers count as conifers.
mixed	Deprecated. See mixedType argument to vegTypeMapGenerator .
...	Additional arguments passed to reproducible::Cache() and equivalentName() .
dPath	path to the data directory
rasterToMatch	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
studyArea	A SpatVector or sf polygons object used as the principle study region, passed to reproducible::prepInputs() .
sppEquiv	table with species name equivalencies between naming conventions (see sppEquiv , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
knnNamesCol	character string indicating the column in sppEquiv containing kNN species names. Default "KNN" for when sppEquivalencies_CA is used.

sppEquivCol	the column name to use from sppEquiv (see sppEquivalencies_CA). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR : sppEquivalencies_CA that best matches the species names.
thresh	the minimum number of pixels where the species must have biomass > 0 to be considered present in the study area. Defaults to 1.
url	the source url for the data, passed to reproducible::prepInputs()
destinationPath	path to data directory where objects will be downloaded or saved to
outputPath	path to output directory

Value

A factor raster
A raster stack of percent cover layers by species.

minRelativeBDefaults	<i>minRelativeB defaults for Western Boreal Forest Canada</i>
----------------------	---

Description

minRelativeB defaults for Western Boreal Forest Canada

Usage

minRelativeBDefaults()

mixedwoodProp	<i>Leading-species and mixedwood proportion thresholds</i>
---------------	--

Description

Two proportions decide how a stand’s composition is labelled, and they answer different questions. Both are read from options so a project sets them once.

Usage

mixedwoodProp()

leadingSpeciesProp()

Details

mixedwoodProp() is the **group** threshold: the share of the stand held by all conifers, or by all broadleaves, at or above which the stand stops being mixedwood. It is the definition the national products are written on – NTEMS/EOSD (Wulder & Nelson 2003) and the NFI photo plot dictionary call a stand coniferous or broadleaf at 75% or more of total basal area (NFI photo plots: total tree volume), and mixed wood when neither group reaches it.

leadingSpeciesProp() is the **single-species** threshold: the share one species must hold to be called leading. It is a different question and a user may reasonably want a different answer – 0.51, "just a majority", is a sensible setting – so it has its own option. When it is not set it takes the mixedwood value, which is what makes the two agree by default.

Set them with, e.g., options(LandR.mixedwoodProp = 0.7) or options(LandR.leadingSpeciesProp = 0.51). The shipped default for LandR.mixedwoodProp lives in [LandROptions\(\)](#) and is the only place the number itself is written down, so changing it later changes every function at once.

Deciduous conifers are conifers here. Larix is Type == "Conifer" in [sppEquivalencies_CA](#), so it never counts toward the broadleaf share.

Value

A numeric proportion between 0 and 1.

See Also

[vegTypeMapGenerator\(\)](#), [lccMapGenerator\(\)](#)

modifySpeciesAndSpeciesEcoregionTable

Update species and speciesEcoregion tables

Description

Update species and speciesEcoregion tables

Usage

```
modifySpeciesAndSpeciesEcoregionTable(speciesEcoregion, speciesTable)
```

Arguments

speciesEcoregion

A data.table with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: speciesCode and ecoregionGroup, character representation of species and ecoregion groups respectively, maxB the maximum biomass for the species in a given 'ecoregion', maxANPP the maximum aboveground net primary productivity and SEP the species establishment probability. May contain columns

inflationFactor (used to adjust maxB) and mANPPproportion (used to calculate maxANPP).

speciesTable A species traits table, with **at least** the following columns:

- speciesCode an character representation of species;
- Area a character of geographical area within a species range for which trait values are relevant;
- longevity species longevity in years, sexualmature age in years at sexual maturity;
- shadetolerance numeric value between 1-5 of relative shade tolerance (with respect to other species);
- seeddistance_eff the "effective" seed dispersal distance;
- seeddistance_max a numeric with the maximum seed dispersal distance;
- mortalityshape and growthcurve: growth curve shape parameters. Other columns (e.g. fire-related traits) can also be included depending on LandR modules in use. Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.

Value

a list with modified speciesEcoregion and speciesTable, under the names "newSpeciesEcoregion" and "newSpeciesTable", respectively.

NAcover2zero	<i>Convert NA values in speciesLayers to zeros</i>
--------------	--

Description

Pixels that are NA but are inside rasterToMatch may need to be converted to 0, as they can could still potentially be forested

Usage

NAcover2zero(speciesLayers, rasterToMatch)

Arguments

speciesLayers A RasterStack or RasterLayer that should contain species cover data in the study area

rasterToMatch A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.

Value

the speciesLayers with 0 in pixels that had NA

nonForestedPixels	<i>Assess non-forested pixels based on species cover data and land-cover</i>
-------------------	--

Description

Assess non-forested pixels based on species cover data and land-cover

Usage

```
nonForestedPixels(
  speciesLayers,
  omitNonTreedPixels,
  forestedLCCClasses,
  rstLCC
)
```

Arguments

speciesLayers	A RasterStack or RasterLayer that should contain species cover data in the study area
omitNonTreedPixels	logical. Should pixels with classes in forestedLCCClasses be included as non-forested?
forestedLCCClasses	vector of forested land-cover classes in rstLCC
rstLCC	raster of land-cover class, e.g., LCC2005

Value

a logical vector of length ncell(rstLCC) where TRUE indicates non-forested pixels where there is no species cover data, or a non-forested land-cover class

overlayLCCs	<i>Overlay different LCC data sources</i>
-------------	---

Description

Overlay different LCC data sources

Usage

```

overlayLCCs(
  LCCs,
  forestedList,
  outputLayer,
  NAcondition,
  NNcondition,
  remapTable = NULL,
  classesToReplace,
  availableERC_by_Sp,
  forestEquivalencies = NULL,
  method = c("nearestWeighted", "nearestRandom")
)

```

Arguments

LCCs	A named list or named RasterStack of layers whose content is Land Cover Class.
forestedList	A named list of same length and names as LCCs indicating which classes in each LCC raster are 'forested', either permanent or transient
outputLayer	A character string that matches one of the named elements in LCCs. This will be the classification system returned.
NAcondition	The condition when a pixel is deemed to be NA. Given as a character string of a vectorized logical statement that will be within the forestEquivalencies table. It should be a set of conditions with == 0, i.e., non-forested. Examples:, e.g., "LCC2005 == 0" or "CC == 0 LCC2005 == 0", where 0 is the non-forested pixels based on converting LCCs and forestedList to 1 and 0.
NNcondition	The 'nearest-neighbour' condition; i.e., the condition when a nearest-neighbour search is done to fill in the pixel with forested type. Given as a character string of a vectorized logical statement that will be parsed within the forestEquivalencies table. It should be a set of conditions with == 0, i.e., non-forested. Examples:, e.g., "LCC2005 == 0" or "CC == 0 LCC2005 == 0", where 0 is the non-forested pixels based on converting LCCs and forestedList to 1 and 0.
remapTable	data.table. This would be for a situation where 2 LCC layers are provided, one has information in a pixel, but not the one which is outputLayer, so this needs a reclassify or remap.
classesToReplace	Passed to convertUnwantedLCC() , for the pixels where NNcondition is TRUE
availableERC_by_Sp	Passed to convertUnwantedLCC() , for the pixels where NNcondition is TRUE. If this is NULL, then it will be created internally with all pixels with: <code>data.table(initialEcoregionCode = LCCs[[outputLayer]][])</code>
forestEquivalencies	A data.frame or NULL. If NULL, this function will derive this table automatically from the other arguments. Otherwise, the user must provide a data.frame with <code>length(LCCs) + 1</code> columns, and <code>2 ^ length(LCCs)</code> rows. Currently not used.

method Character; where the draw among an unwanted pixel's available land-cover classes comes from. Both options weight the classes by their abundance in the pixel's neighbourhood and differ only in reproducibility: "nearestWeighted" (default) keys the draw on the pixel's ground position, so it is deterministic and seed-free, while "nearestRandom" draws from the RNG and so varies with `set.seed()`. See Details.

Author(s)

Eliot McIntire and Alex Chubaty

overlayStacks	<i>Overlay layers within raster stacks</i>
---------------	--

Description

Overlays rasters of different data resolution by filling in gaps in the highest resolution raster with data available in lowest resolution one. If only high or low resolution data are available, it will use it without attempting to overlay.

Usage

```
overlayStacks(
  highQualityStack,
  lowQualityStack,
  outputFilenameSuffix = "overlay",
  destinationPath
)
```

Arguments

`highQualityStack`
high quality list/stack of rasters (will be used preferentially)

`lowQualityStack`
low quality list/stack of rasters (will be used to fill NAs in `highQualityStack`)

`outputFilenameSuffix`
file suffix to save raster if there was overlaying. Defaults to "overlay".

`destinationPath`
path to data directory where objects will be downloaded or saved to

partialggplotMLL_maxB *Partial effect plots of maximum biomass estimates by age*

Description

Plots the maximum biomass estimates along an age gradient as a function of another target covariate, with all others held at average values.

Usage

```
partialggplotMLL_maxB(
  mll,
  data,
  targetCovar = "cover",
  maxCover = 1,
  fixMaxCover = TRUE,
  xCovar = "age",
  showQuantiles = "allQuantiles",
  plotTitle = NULL,
  nonLinModelQuoted,
  linModelQuoted,
  fun = "mean",
  plotCIs = TRUE
)
```

Arguments

mll	a named list with outputs of an <code>bbmle::mle2</code> call (the fitted non-linear model), from which coefficient values will be extracted. If several model outputs are provided all fitted models will be plotted, with plot labels corresponding to list names.
data	data for estimation of maximum biomass. Should contain at least an 'age' column. Note that other covariates will be averaged and 'cover' values will be replaced with the maximum cover value (<code>maxCover</code>). If <code>mll</code> is a list, data is assumed to be the same for the two models.
targetCovar	the covariate for which variation in <code>maxB</code> values will be shown per age value. Defaults to showing how <code>maxB</code> values change with "cover" at any given age. Age values are generated as <code>round(seq(min(age), max(age)*1.5, length.out = 100), 0)</code> . When <code>targetCovar != "cover"</code> , "cover" may be fixed at <code>maxCover</code> . See <code>fixMaxCover</code> .
maxCover	numeric. Value indicating maximum cover/dominance.
fixMaxCover	logical. If <code>TRUE</code> and <code>targetCovar != "cover"</code> , cover is not averaged and is fixed to <code>maxCover</code> .
xCovar	the variable shown in the x axis. Defaults to "age". When <code>xCovar == "age"</code> the output plots are not true "Partial Effects" plots. Instead, they show the variation in <code>B</code> values across values of <code>targetCovar</code> for each value of age.

showQuantiles	controls whether quantile predictions will be shown. If "allQuantiles", quantile values (5%, 25%, 50%, 75%, 95%) of B will be plotted as blue lines, with their respective asymptote values (quantile values at maximum age) as dashed lines. If "maximum" the 100% quantile will be plotted. If "none", only the 50% quantile line (average prediction) is plotted. Quantiles are always calculated at max(age).
plotTitle	character. Passed to title in <code>ggplot2::labs()</code>
nonLinModelQuoted	a named list of non-linear equations as a call (quoted expression) passed to <code>mle2(minuslog1)</code> . See <code>?mle</code> . Accepts equations with three parameters 'A', 'p' and 'k'. List names and length must the same as in <code>m11</code> .
linModelQuoted	A named list of lists of linear equations/modes relating each parameter ('A', 'p' and 'k') with a set of covariates. A call (quoted expression) passed to <code>mle2(..., parameters)</code> . Note that for the purpose of tree growth, the linear equation determining 'A' should include a 'cover' predictor indicating the tree cover or dominance in the stand. Should be scaled between 0 and maxCover. List names and length must the same as in <code>m11</code> .
fun	passed to <code>.MLLMaxBPartialPlotData</code> .
plotCIs	should confidence intervals be calculated and plotted?

Details

Note that the original data, not the predicted values is shown.

See Also

`bbmle::mle2()`

partitionBiomass	<i>Partition biomass according to cover estimates</i>
------------------	---

Description

This function will partition totalBiomass into each cohort. It will discount deciduous cover, if $x < 1$.

Usage

```
partitionBiomass(x = 1, pixelCohortData)
```

Arguments

x The ratio for deciduous cover:biomass, where conifer cover:biomass = 1

pixelCohortData A full pixelCohortData object (i.e., not cohortData)

pixelFate	<i>Create or amend data to a pixelFateDT object</i>
-----------	---

Description

Create or amend data to a pixelFateDT object

Usage

```
pixelFate(
  pixelFateDT,
  fate = NA_character_,
  pixelsRemoved = 0,
  runningPixelTotal = NA_integer_
)
```

Arguments

pixelFateDT	A pixelFateDT data.table with 3 columns: fate, pixelsRemoted, and runningPixelTotal.
fate	A character string (length 1) describing in words the change
pixelsRemoved	A numeric indicating how many pixels were removed due to the fate.
runningPixelTotal	an optional numeric with new, running total. If not supplied, it will be calculated from the last row of pixelFateDT runningTotal minus the pixelsRemoved

Value

A pixelFateDT object, updated with one extra row.

plantNewCohorts	<i>Create new cohorts based on provenance table with unique pixelGroup and add to cohortData</i>
-----------------	--

Description

Create new cohorts based on provenance table with unique pixelGroup and add to cohortData

Usage

```
plantNewCohorts(
  newPixelCohortData,
  cohortData,
  pixelGroupMap,
  initialB = 10,
  currentTime,
```

```

    successionTimestep,
    trackPlanting = FALSE
  )

```

Arguments

newPixelCohortData	the cohorts that were harvested
cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPPAct, and sumB.
pixelGroupMap	A RasterLayer with pixel values equal to a pixel group number that corresponds exactly to pixelGroup column in cohortData.
initialB	the initial biomass of new cohorts. Defaults to ten.
currentTime	The current simulation time e.g., time(sim).
successionTimestep	The time between successive seed dispersal events. In LANDIS-II, this is called "Succession Timestep".
trackPlanting	adds column that tracks planted cohorts if TRUE

Value

A data.table with a new cohortData

plot_raster	<i>Plot raster objects using ggplot</i>
-------------	---

Description

Plot raster objects using ggplot

Usage

```
plot_raster(x, title = NULL, subtitle = NULL)
```

Arguments

x	A SpatRaster object
title	character, the plot title
subtitle	character, the plot subtitle

Value

ggplot object

plotLeadingSpecies	<i>Leading species plots</i>
--------------------	------------------------------

Description

Plot effects on conifer-to-deciduous or deciduous-to-conifer conversions.

Usage

```
plotLeadingSpecies(
  studyAreaName,
  climateScenario,
  Nreps,
  years,
  outputDir,
  treeSpecies,
  defineLeading = .defineLeading,
  leadingPercentage = 0.8,
  treeType = NULL,
  rasterToMatch,
  figurePath = file.path(outputDir, "figures")
)
```

Arguments

studyAreaName	character string giving the study area name
climateScenario	character string specifying the name of a CIMP6 climate scenario, including SSP, formatted as in ClimateNA, using underscores as separator (e.g., 'CanESM5_SSP370').
Nreps	the number of simulation replicates/run used to produce summary figures. NOTE: mclapply is used internally, so you should set options(mc.cores = nReps) to take advantage of parallel processing.
years	TODO
outputDir	Path specifying the directory to which outputs figures/objects should be saved.
treeSpecies	TODO
defineLeading	TODO
leadingPercentage	TODO
treeType	TODO
rasterToMatch	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
figurePath	Directory to which the .png figure is saved. Defaults to figures/ under outputDir; a SpaDES module would pass SpaDES.core::figurePath(sim).

Value

list of file names corresponding to the figures and/or objects written to disk
 list of filepaths corresponding to the images and/or objects written to disk

plotSpatial	Create a ggplot of a raster or sf object.
-------------	---

Description

Can be used with SpaDES.core::Plots.

Usage

```
plotSpatial(x, plotTitle, limits = NULL, field = NULL)
```

Arguments

x	SpatRaster, RasterLayer, SpatVector or sf object
plotTitle	character. A title for the plot passed to ggplot::labs(title = plotTitle).
limits	TODO
field	character. If x is sf or SpatVector, a field to plot.

plotVTM	Summary plots of leading vegetation types
---------	---

Description

Create raster of leading vegetation types, and build a bar chart summary beside a vegetation type map. NOTE: plot order will follow colors order.

Usage

```
plotVTM(
  speciesStack = NULL,
  vtm = NULL,
  vegLeadingProportion = mixedwoodProp(),
  sppEquiv,
  sppEquivCol,
  colors,
  title = "Leading vegetation types"
)
```

Arguments

speciesStack	A SpatRaster, RasterStack or RasterBrick of percent-cover-by-species layers.
vtm	An optional vegetation type map (RasterLayer or SpatRaster). If not supplied, will be produced internally by makeVegTypeMap.
vegLeadingProportion	Numeric between 0-1, the share of a stand above which it stops being called mixed. Which share depends on what is being asked, and the two questions have their own options (see leadingProportions): with mixedType = 2 it is the whole broadleaf group against the whole conifer group, and defaults to mixedwoodProp() (LandR.mixedwoodProp, the NTEMS/EOSD and NFI value: coniferous or broadleaf at 75% or more of total basal area, mixed wood when neither reaches it); otherwise it is one species' share and defaults to leadingSpeciesProp() (LandR.leadingSpeciesProp, which takes the mixedwood value unless it is set). Deciduous conifers count as conifers.
sppEquiv	table with species name equivalencies between naming conventions (see sppEquivalencies_CA , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
sppEquivCol	the column name to use from sppEquiv (see sppEquivalencies_CA). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.
colors	Named vector of colour codes, named using species names. NOTE: plot order will follow this order.
title	The title to use for the generated plots.

Value

A patchwork object (a ggplot) with the two panels side by side. Draw it with print(), save it with ggplot2::ggsave(), or pass it to SpaDES.core::Plots().

Author(s)

Eliot McIntire

prepEcoregions	<i>Prepare ecoregions objects</i>
----------------	-----------------------------------

Description

Prepare ecoregions objects

Usage

```

prepEcoregions(
  ecoregionRst = NULL,
  ecoregionLayer,
  ecoregionLayerField = NULL,
  rasterToMatchLarge,
  rstLCCAdj,
  pixelsToRm,
  cacheTags
)

```

Arguments

ecoregionRst an optional raster object that could be passed to sim, representing ecoregions

ecoregionLayer an sf polygons object representing ecoregions

ecoregionLayerField
optional. The field in ecoregionLayer that represents ecoregions.

rasterToMatchLarge
A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatchLarge should correspond to studyAreaLarge if used.

rstLCCAdj RasterLayer representing land cover adjusted for non-forest classes

pixelsToRm a vector of pixels to remove

cacheTags UserTags to pass to cache

prepEcozonesRst	<i>Prepare ecozones raster</i>
-----------------	--------------------------------

Description

Download, rasterize, crop, mask, and reproject Canadian national ecozones shapefile.

Usage

```

prepEcozonesRst(
  url = NULL,
  destinationPath,
  studyArea = NULL,
  rasterToMatch = NULL
)

```

Arguments

url	character. ecozones shapefile url. Default: http://sis.agr.gc.ca/cansis/nsdb/ecostrat/zone/ecozone_shp.zip .
destinationPath	path to data directory where objects will be downloaded or saved to
studyArea	A <code>SpatVector</code> or <code>sf</code> polygons object used as the principle study region, passed to <code>reproducible::prepInputs()</code> .
rasterToMatch	A <code>SpatRaster</code> object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). <code>rasterToMatch</code> should correspond to <code>studyArea</code> if used.

Value

`SpatRaster`

prepInputs_CWIM	<i>Wetland (site) layer from the Canadian Wetland Inventory Map</i>
-----------------	---

Description

SCANFI's land cover has no wetland classes, so a land-cover map built from it cannot say which forest stands on wet ground – the distinction NTEMS draws with classes 80 (wetland) and 81 (treed wetland). This builds that site layer from the Canadian Wetland Inventory Map v3A (CWIM3A; Mahdianpari et al. 2021): 10 m, national, 2016-2020 imagery.

Usage

```
prepInputs_CWIM(  
  to,  
  url = .cwimURL,  
  wetClasses = 1:4,  
  wetThreshold = 0.5,  
  writeTo = NULL  
)
```

Arguments

to	A <code>SpatRaster</code> giving the grid to return, and the study window. Its NA cells are NA in the result.
url	The CWIM3A source. A URL is read through GDAL's <code>/vsicurl/</code> ; a local path is read directly (useful for a pre-cut regional copy).
wetClasses	CWIM3A codes counted as wet. Default: Bog, Fen, Marsh, Swamp.
wetThreshold	Minimum wet fraction of a target cell for it to be wet.
writeTo	Optional file to write the result to.

Details

Only the study window is read: the source is a cloud-optimised GeoTIFF, so terra fetches the tiles it needs rather than the national raster.

CWIM3A codes are 1 Bog, 2 Fen, 3 Marsh, 4 Swamp, 5 Shallow water; 15 is NoData, and CWIM3A maps nothing but those five classes, so NoData means "not wetland". Shallow water is not counted as wet by default: it is open water, and it coincides with land-cover water (class 20) almost exactly – on a 60 km boreal test window, 6,593 of 6,764 CWIM shallow-water cells were VLCE2 water.

The 10 m cells are summarised onto to as the *fraction* of each target cell that is wet; a cell is wet when that fraction is at least wetThreshold. For a 0/1 layer that is the majority rule, without "mode"'s arbitrary tie-breaking.

Value

A single-layer SpatRaster on to's grid, named "wetland": 1 wet, 0 not wet.

References

Mahdianpari, M., et al. (2021). The Third Generation of Pan-Canadian Wetland Map at 10 m Resolution Using Multisource Earth Observation Data on Cloud Computing Platform. IEEE JSTARS 14, 8789-8803.

See Also

[wetlandToLCC\(\)](#)

prepInputs_FA0_forest *Obtain the NFIS FAO forest layer*

Description

Obtain the NFIS FAO forest layer

Usage

```
prepInputs_FA0_forest(
  year = 2022,
  to = NULL,
  destinationPath = NULL,
  method = "near",
  ...
)
```

Arguments

year	One of 2019, 2022.
to	Passed to prepInputs, to align the layer with the land cover.
destinationPath	Passed to prepInputs.
method	Resampling method passed to prepInputs.
...	Other arguments passed to prepInputs, e.g. maskTo = NA to align without masking.

Value

a SpatRaster of FAO forest codes: 0 non-forest, 1 forest, 2 forest land whose trees were removed by fire or harvest since 1984.

```
prepInputs_NTEMS_DominantSpecies
```

Obtain an Dominant species layer for a given year from NTEMS

Description

Obtain an Dominant species layer for a given year from NTEMS

Usage

```
prepInputs_NTEMS_DominantSpecies(
  year = 2011,
  destinationPath,
  sppEquiv = LandR::sppEquivalencies_CA,
  sppEquivCol = "LandR",
  ...
)
```

Arguments

year	stack of species layers rasters
destinationPath	path to data directory where objects will be downloaded or saved to
sppEquiv	table with species name equivalencies between naming conventions (see sppEquivalencies_CA , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
sppEquivCol	the column name to use from sppEquiv (see sppEquivalencies_CA). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.
...	passed to prepInputs

Value

a SpatRaster with dominant species

```
prepInputs_NTEMS_LCC_FAO
```

Obtain an LCC layer for a given year from NTEMS, with forest matching the FAO definition

Description

Pixels that are forest land but carry no trees in year are given disturbedCode, so that a recent burn or cutblock is still simulated as forest rather than dropped as shrubland. [forestLandMask\(\)](#) describes how forest land is decided, and [.applyForestLand\(\)](#) which pixels are relabelled.

Usage

```
prepInputs_NTEMS_LCC_FAO(
  year = 2010,
  disturbedCode = 240,
  resampleMethod = "near",
  forestLandFrom = c("both", "fao", "lccYears"),
  forestLandYears = NULL,
  faoYear = 2022,
  convertibleClasses = NULL,
  ...
)
```

Arguments

year	1984 to 2022 possible.
disturbedCode	value assigned to pixels that are forest land but not treed in year
resampleMethod	method used when resampling LCC layers to match rasterToMatch
forestLandFrom	how forest land is decided: "fao" (the FAO forest layer), "lccYears" (treed in any of forestLandYears), or "both" (the default union). See forestLandMask() .
forestLandYears	the land-cover years scanned when forestLandFrom is "both" or "lccYears". NULL (default) uses .defaultForestLandYears() , roughly one year per decade plus the most recent. Each year is one more layer to read.
faoYear	the year of the FAO forest layer: 2019, 2022.
convertibleClasses	the classes that may become disturbedCode; NULL (default) means every class that is not treed. See .applyForestLand() .
...	passed to prepInputs

Value

a SpatRaster with corrected forest pixels

prepInputs_NTEMS_Nonforest	<i>Reclassify non-flammable pixels that become flammable - herbaceous or shrubby - vegetation</i>
----------------------------	---

Description

Reclassify non-flammable pixels that become flammable - herbaceous or shrubby - vegetation

Usage

```
prepInputs_NTEMS_Nonforest(  
  rstLCC,  
  endYear = 2019,  
  lccToAdjust = 33,  
  nonforestLCC = c(50, 100),  
  ...  
)
```

Arguments

rstLCC	input lcc layer with bare soil class that may become vegetated
endYear	NTEMS LCC year to use for correcting transition from bare to non-forest
lccToAdjust	lcc values of the bare class
nonforestLCC	allowable lcc values for bare to become
...	non-spatial arguments passed to prepInputs e.g. destinationPath

Value

a SpatRaster with non-flammable pixels corrected if they become flammable non-forest

prepInputs_SCANFI_LCC_FAO	<i>Obtain an LCC layer for a given year from SCANFI, with forest matching the FAO definition</i>
---------------------------	--

Description

Pixels that are forest land but carry no trees in year are given disturbedCode, so that a recent burn or cutblock is still simulated as forest rather than dropped as shrubland. [forestLandMask\(\)](#) describes how forest land is decided, and [.applyForestLand\(\)](#) which pixels are relabelled.

Usage

```

prepInputs_SCANFI_LCC_FAO(
  year = 2020,
  dataVersion = "V2",
  disturbedCode = 240,
  resampleMethod = "near",
  forestLandFrom = c("both", "fao", "lccYears"),
  forestLandYears = NULL,
  faoYear = 2022,
  convertibleClasses = NULL,
  ...
)

```

Arguments

year	data year for SCANFI landcover data. 2000, 2010, 2020 possible for V1. 1985, 1990, 1995, 2000, 2005, 2010, 2015, 2020, 2025 possible for V2.
dataVersion	Character. SCANFI product version for data. Default is currently "V2". "V1" also available.
disturbedCode	value assigned to pixels that are forest land but not treed in year
resampleMethod	method used when resampling LCC layers to match rasterToMatch
forestLandFrom	how forest land is decided: "fao" (the FAO forest layer), "lccYears" (treed in any of forestLandYears), or "both" (the default union). See forestLandMask() .
forestLandYears	the SCANFI years scanned when forestLandFrom is "both" or "lccYears". NULL (default) uses .defaultForestLandYears() , roughly one year per decade plus the most recent. Each year is one more layer to read.
faoYear	the year of the FAO forest layer: 2019, 2022.
convertibleClasses	the classes that may become disturbedCode; NULL (default) means every class that is not treed. See .applyForestLand() .
...	passed to prepInputs

Value

a SpatRaster with corrected forest pixels

```
prepInputs_SCANFI_structure
```

Obtain a SCANFI structural attribute layer

Description

SCANFI publishes canopy height and canopy closure alongside the biomass layer that `prepRawBiomassMap()` fetches. They describe how much structure a pixel carries, independent of what species carry it, which is what makes them useful as controls: two pixels of the same height and closure hold comparable stands, so a difference in biomass between them is a difference in composition rather than in site quality.

Usage

```
prepInputs_SCANFI_structure(  
  attribute = c("height", "closure"),  
  year = 2020,  
  dataVersion = "V2",  
  ...  
)
```

Arguments

attribute	"height" (metres) or "closure" (percent crown closure).
year	one of 1985, 1990, 1995, 2000, 2005, 2010, 2015, 2020, 2025.
dataVersion	SCANFI product version. Only "V2" publishes these layers.
...	passed to <code>reproducible::prepInputs()</code> and <code>reproducible::Cache()</code> ; pass one of <code>to</code> , <code>cropTo</code> or <code>rasterToMatch</code> – the national rasters are large.

Value

a `SpatRaster`

prepInputsCanDEM	<i>Get digital elevation map of Canada</i>
------------------	--

Description

Defaults to using 3 arcmin DEM of Canada in lonlat.

Usage

```
prepInputsCanDEM(studyArea, rasterToMatch, destinationPath)
```

Arguments

studyArea	A <code>SpatVector</code> or <code>sf</code> polygons object used as the principle study region, passed to <code>reproducible::prepInputs()</code> .
rasterToMatch	A <code>SpatRaster</code> object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). <code>rasterToMatch</code> should correspond to <code>studyArea</code> if used.
destinationPath	path to data directory where objects will be downloaded or saved to

Value

SpatRaster

```
prepInputsEcoregion
```

Prepare ecoregion table

Description

Get the dummy ecoregion table from LANDIS-II examples.

Usage

```
prepInputsEcoregion(url = NULL, dPath, cacheTags = NULL)
```

Arguments

url	If NULL (the default), uses one from the LANDIS-II project: https://github.com/LANDIS-II-Foundation/Extensions-Succession/blob/master/biomass-succession-archi/trunk/tests/v6.0-2.0/ecoregion.txt).
dPath	The destination path.
cacheTags	User tags to pass to Cache.

Value

A data.table

```
prepInputsFireYear
```

Create a raster of fire perimeters

Description

Create a raster of fire perimeters

Usage

```
prepInputsFireYear(..., rasterToMatch, fireField = "YEAR", earliestYear = 1950)
```

Arguments

...	Additional arguments passed to <code>reproducible::prepInputs()</code>
rasterToMatch	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
fireField	field used to rasterize fire polys
earliestYear	the earliest fire date to allow

Value

a SpatRaster layer of fire perimeters with fire year values.

Examples

```
## NOTE: runs locally, skipped on CI -- downloads NFDB_poly.zip from
## cwfis.cfs.nrcan.gc.ca (same predicate as testthat::skip_on_ci())
withr::local_options(list(
  reproducible.useTerra = TRUE,
  reproducible.rasterRead = "terra::rast"
))

randomPoly <- LandR::randomStudyArea(
  size = 1e+8, seed = 5
)

ras2match <- terra::rast(
  randomPoly,
  vals = 1,
  res = 100,
)
ras2match <- terra::mask(ras2match, randomPoly)

firePerimeters <- prepInputsFireYear(
  url = paste0(
    "https://cwfis.cfs.nrcan.gc.ca/downloads/",
    "nfdb/fire_poly/current_version/NFDB_poly.zip"
  ),
  destinationPath = file.path(tempdir(), "ex_prepInputsFireYear"),
  rasterToMatch = ras2match,
  earliestYear = 1950
)

if (interactive()) {
  terra::plot(firePerimeters)
  terra::plot(randomPoly, add = TRUE)
}

withr::deferred_run()
```

```
prepInputsLCC
```

Simple prepInputs for Canadian LCC data

Description

A wrapper around prepInputs for the Canadian Land Cover Classification product(s).

Usage

```

prepInputsLCC(
  year = 2010,
  destinationPath = asPath("."),
  method = c("ngb", "near"),
  writeTo = NULL,
  ...
)

```

Arguments

year	Numeric, either 2010 or 2015. See note re: backwards compatibility for 2005.
destinationPath	Character string of a directory in which to download and save the file that comes from url and is also where the function will look for archive or targetFile. To prevent repeated downloads in different locations, the user can also set options("reproducible.destinationPaths") to one or more local file paths to search for the file before attempting to download. Default for that option is NULL meaning do not search locally. The previous name options("reproducible.inputPaths") is still accepted as a backwards-compatible alias.
method	passed to terra::intersect or raster::intersect , and reproducible::prepInputs
writeTo	passed to reproducible::prepInputs
...	Arguments passed to terra::mask (for maskTo), terra::project (for projectTo) or terra::writeRaster (for writeTo) and not used for cropTo, as well postProcess's rasterToMatch and studyArea arguments (see below). Commonly used arguments might be method, touches, and datatype. If filename is passed, it will be ignored; use writeTo = .

Note

As of May 2021, NRCAN no longer provides/hosts the LCC2005 data. A privately hosted version of the data is available to maintain backwards compatibility, but new users/projects should use the 2010 (or newer) data.

prepInputsSpecies	<i>Download and prepare a species traits table for use with Biomass_core module</i>
-------------------	---

Description

TODO: add detailed description

Usage

```

prepInputsSpecies(url = NULL, dPath, cacheTags = NULL)

prepInputsMainInput(url = NULL, dPath = tempdir(), cacheTags = NULL)

```

Arguments

url	If NULL (the default), uses one from the LANDIS-II project: https://github.com/LANDIS-II-Foundation/Extensions-Succession/blob/master/biomass-succession-archi/trunk/tests/v6.0-2.0/biomass-succession_test.txt ").
dPath	The destination path.
cacheTags	User tags to pass to Cache.

Value

A data.table with columns ... TODO

Note

This one is tailored to Canadian forests (?)

prepInputsSpeciesEcoregion
<i>Prepare species ecoregion table</i>

Description

Get the dummy ecoregion table from LANDIS-II examples.

Usage

prepInputsSpeciesEcoregion(url = NULL, dPath, cacheTags = NULL)

Arguments

url	If NULL (the default), uses one from the LANDIS-II project: https://github.com/LANDIS-II-Foundation/Extensions-Succession/blob/master/biomass-succession-archi/trunk/tests/v6.0-2.0/biomass-succession-dynamic-inputs_test.txt ").
dPath	The destination path.
cacheTags	User tags to pass to Cache.

Value

A data.table

```
prepInputsStandAgeMap Create standAgeMap
```

Description

Create the standAgeMap raster containing age estimates for pixelCohortData. A separate `reproducible::prepInputs()` call will source Canadian National Fire Data Base data to update ages of recently burned pixels. To suppress this, pass NULL/NA fireURL

Usage

```
prepInputsStandAgeMap(
  rasterToMatch = NULL,
  dataSource = "SCANFI",
  dataYear = 2020,
  dataVersion = "V2",
  ageURL = NULL,
  ageFun = "terra::rast",
  maskWithRTM = TRUE,
  method = "bilinear",
  datatype = "INT2U",
  destinationPath = NULL,
  writeTo = NULL,
  firePerimeters = NULL,
  fireURL = paste0("https://cwfis.cfs.nrcan.gc.ca/downloads/nfdb/",
    "fire_poly/current_version/NFDB_poly.zip"),
  fireFun = "terra::vect",
  fireField = "YEAR",
  ...
)
```

Arguments

- | | |
|---------------|---|
| rasterToMatch | A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used. |
| dataSource | <p>Character. One of KNN, NTEMS, or SCANFI. Defaults to SCANFI for dataYear 2020. Also available:</p> <ul style="list-style-type: none"> • KNN for dataYear 2011; 2001 When dataSource = "SCANFI": • For dataYear = 2020, the SCANFI stand age map for 2020 is used directly. • For dataYear != 2020, the 2020 SCANFI map is back-adjusted to the requested dataYear using NTEMS fire and harvest disturbance layers (1985–2020) and the 2001 kNN stand age map. Disturbed pixels are assigned ages based on year of disturbance and kNN-based estimates where applicable, while undisturbed areas are aged by subtracting the difference between dataYear and 2020. Negative ages are set to zero. This process fills missing or |

inconsistent SCANFI values and records pixel IDs that were imputed or adjusted. A separate `reproducible::prepInputs()` call can be used to source Canadian National Fire Database (NFDB) fire polygons, allowing further stand age correction for burned areas. To suppress this, set `fireURL = NULL` or `fireURL = NA`.

<code>dataYear</code>	Numeric. Year for which data is obtained. Can be 2001 or 2011 for KNN, 2020 for SCANFI V1, or 1985-2025 (5 year intervals) for SCANFI V2.
<code>dataVersion</code>	Character. SCANFI product version for data. Default is currently V2. V1 also available.
<code>ageURL</code>	URL for age map download. Will be supplied based on <code>dataSource</code> and <code>dataYear</code>
<code>ageFun</code>	passed to 'fun' arg of <code>reproducible::prepInputs()</code> of stand age map
<code>maskWithRTM</code>	passed to <code>reproducible::prepInputs()</code> of stand age map
<code>method</code>	passed to <code>reproducible::prepInputs()</code> for reprojecting the stand age map
<code>datatype</code>	passed to <code>reproducible::prepInputs()</code> of stand age map
<code>destinationPath</code>	path to data directory where objects will be downloaded or saved to
<code>writeTo</code>	passed to <code>reproducible::prepInputs()</code> of stand age map
<code>firePerimeters</code>	fire raster layer fire year values.
<code>fireURL</code>	url to download fire polygons used to update age map. If NULL or NA age imputation is bypassed. Requires passing <code>rasterToMatch</code> . Only used if <code>firePerimeters</code> is missing.
<code>fireFun</code>	passed to <code>reproducible::prepInputs()</code> of fire data. Only used if <code>firePerimeters</code> is missing.
<code>fireField</code>	field used to rasterize fire polys. Only used if <code>firePerimeters</code> is missing.
<code>...</code>	additional arguments passed to <code>reproducible::prepInputs()</code>

Value

a raster layer stand age map corrected for fires, with an attribute vector of pixel IDs for which ages were corrected. If no corrections were applied the attribute vector is `integer(0)`.

Examples

```
## Not run:
library(SpaDES.tools)
library(terra)
library(reproducible)
randomPoly <- randomStudyArea(size = 1e7)
randomPoly
ras2match <- rast(res = 250, ext = ext(randomPoly), crs = crs(randomPoly))
ras2match <- rasterize(randomPoly, ras2match)
tempDir <- file.path(tempdir(), "ex_prepInputsStandAgeMap")

## NOT USING FIRE PERIMETERS TO CORRECT STAND AGE
## rasterToMatch does not need to be provided, but can be for masking/cropping.
standAge <- prepInputsStandAgeMap(
```

```

    destinationPath = tempDir,
    rasterToMatch = ras2match,
    fireURL = NA
) ## or NULL
attr(standAge, "imputedPixID")

## USING FIRE PERIMETERS TO CORRECT STAND AGE
## ideally, get the firePerimeters layer first
firePerimeters <- Cache(prepareInputsFireYear,
  url = paste0(
    "https://cwfis.cfs.nrcan.gc.ca/downloads",
    "/nfdb/fire_poly/current_version/NFDB_poly.zip"
  ),
  destinationPath = tempDir,
  rasterToMatch = ras2match
)

## Example adjusting SCANFI to 2000 using NTEMS and kNN
standAge2000 <- prepareInputsStandAgeMap(
  destinationPath = tempdir(),
  dataSource = "SCANFI",
  dataYear = 2000,
  rasterToMatch = rast(res = 250, ext = ext(vect(randomStudyArea(size = 1e7))))
)
attr(standAge2000, "imputedPixID")

## End(Not run)

```

```
prepRasterToMatch
```

```
Create rasterToMatch and rasterToMatchLarge
```

Description

rasterToMatch and rasterToMatchLarge raster layers are created from studyArea and studyAreaLarge polygons (respectively) using a template raster (often rawBiomassMap)

Usage

```

prepRasterToMatch(
  studyArea,
  studyAreaLarge,
  rasterToMatch,
  rasterToMatchLarge,
  destinationPath,
  templateRas,
  studyAreaName,
  cacheTags = NULL
)

```

Arguments

studyArea	A SpatVector or sf polygons object used as the principle study region, passed to <code>reproducible::prepInputs()</code> .
studyAreaLarge	same as studyArea, but larger and completely covering it.
rasterToMatch	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
rasterToMatchLarge	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatchLarge should correspond to studyAreaLarge if used.
destinationPath	path to data directory where objects will be downloaded or saved to
templateRas	a template raster used to make rasterToMatch and/or rasterToMatchLarge. Must match studyAreaLarge.
studyAreaName	character string giving the study area name.
cacheTags	a character vector of strings to pass to Cache(userTags)

```
prepRawBiomassMap      Create rawBiomassMap
```

Description

Create the rawBiomassMap raster containing biomass estimates for pixelCohortData.

Usage

```
prepRawBiomassMap(
  dataSource = "SCANFI",
  dataYear = "2020",
  dataVersion = "V2",
  ...
)
```

Arguments

dataSource	Character. One of KNN, NTEMS, or SCANFI. Defaults to KNN for dataYear 2001. Also available: • KNN for dataYear 2011; • NTEMS for dataYear 2015; • SCANFI for dataYear 2000, 2010, or 2020.
dataYear	Numeric. Year for which data is obtained. Can be 2001 or 2011 for KNN, 2015 for NTEMS, 2000, 2010, or 2020 for SCANFI V1, or 1985, 1990, 1995, 2000, 2005, 2010, 2015, 2020 (default), 2025 possible for V2.

dataVersion	Character. SCANFI product version for data. Default is currently V2. V1 also available.
...	arguments passed to reproducible::prepInputs() and reproducible::Cache(). If the following arguments are not provided, the following values will be used: • url: by default, the 2020 SCANFI stand biomass map is downloaded from a private google drive • useSAcrs and projectTo: FALSE and NA • method: "bilinear" • datatype: "INT2U" • overwrite: TRUE • omitArgs: c("destinationPath", "targetFile", "userTags", "stable")

Value

a rawBiomassMap raster

prepSpeciesLayers_KNN *Prepare species layers*

Description

Download and prepare species layers for a given study area, from one of several sources:

- [prepSpeciesLayers_CASFRI\(\)](#) uses CASFRiv4 described in Cosco (2011);
- [prepSpeciesLayers_ForestInventory\(\)](#) uses data prepared for Western Canada as part of the LandWeb project;
- [prepSpeciesLayers_KNN\(\)](#) uses the layers from Beaudoin *et al.* (2014, 2017);
- [prepSpeciesLayers_MBFRI\(\)](#) uses data derived from Manitoba Forest Resource Inventories;
- [prepSpeciesLayers_NTEMS\(\)](#) uses the National Terrestrial Ecosystem Monitoring System for Canada (NTEMS) tree species data described in Hermosilla *et al.* (2024);
- [prepSpeciesLayers_ONFRI\(\)](#) uses data derived from Ontario Forest Resource Inventories;
- [prepSpeciesLayers_Pickell\(\)](#) uses data prepared for Western Canada by Pickell & Coops (2016) as part of the LandWeb project;
- [prepSpeciesLayers_SCANFI\(\)](#) uses SCANFI data described in Guidon *et al.* (2023, 2024);

Usage

```
prepSpeciesLayers_KNN(
  destinationPath,
  outputPath,
  url = NULL,
  studyArea,
  rasterToMatch,
  sppEquiv,
```

```
sppEquivCol,  
thresh = 10,  
...  
)  
  
prepSpeciesLayers_CASFRI(  
  destinationPath,  
  outputPath,  
  url = NULL,  
  studyArea,  
  rasterToMatch,  
  sppEquiv,  
  sppEquivCol,  
  ...  
)  
  
prepSpeciesLayers_Pickell(  
  destinationPath,  
  outputPath,  
  url = NULL,  
  studyArea,  
  rasterToMatch,  
  sppEquiv,  
  sppEquivCol,  
  ...  
)  
  
prepSpeciesLayers_ForestInventory(  
  destinationPath,  
  outputPath,  
  url = NULL,  
  studyArea,  
  rasterToMatch,  
  sppEquiv,  
  sppEquivCol,  
  ...  
)  
  
prepSpeciesLayers_NTMS(  
  destinationPath,  
  outputPath,  
  url = NULL,  
  dataYear = 2020,  
  studyArea,  
  rasterToMatch,  
  sppEquiv,  
  sppEquivCol,  
  thresh = 10,
```

```

    ...
)

prepSpeciesLayers_SCANFI(
  destinationPath,
  outputPath,
  url = NULL,
  dataYear = 2020,
  dataVersion = "V2",
  to = NULL,
  cropTo = NULL,
  projectTo = NULL,
  maskTo = NULL,
  sppEquiv,
  sppEquivCol,
  thresh = 10,
  ...
)

prepSpeciesLayers_MBFRI(
  destinationPath,
  outputPath,
  url = NULL,
  studyArea,
  rasterToMatch,
  sppEquiv,
  sppEquivCol,
  ...
)

prepSpeciesLayers_ONFRI(
  destinationPath,
  outputPath,
  url = NULL,
  studyArea,
  rasterToMatch,
  sppEquiv,
  sppEquivCol,
  ...
)

```

Arguments

<code>destinationPath</code>	path to data directory where objects will be downloaded or saved to
<code>outputPath</code>	character, specifying the output directory to use
<code>url</code>	if NULL, the default, use the default source url
<code>studyArea</code>	A <code>SpatVector</code> or <code>sf</code> polygons object used as the principle study region, passed

	to <code>reproducible::prepInputs()</code> .
<code>rasterToMatch</code>	A <code>SpatRaster</code> object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). <code>rasterToMatch</code> should correspond to <code>studyArea</code> if used.
<code>sppEquiv</code>	table with species name equivalencies between naming conventions (see sppEquivalencies_CA , also <code>?sppEquiv</code>). For functions that have <code>mixedType</code> , this only necessary if <code>mixedType == 2</code> . If not provided and <code>mixedType == 2</code> , will attempt to use <code>LandR::sppEquivalencies_CA</code> .
<code>sppEquivCol</code>	the column name to use from <code>sppEquiv</code> (see sppEquivalencies_CA). For functions that have <code>mixedType</code> , only necessary if <code>mixedType == 2</code> . If <code>sppEquiv</code> is not provided and <code>mixedType == 2</code> , will use the column of <code>LandR::sppEquivalencies_CA</code> that best matches the species names.
<code>thresh</code>	threshold \ If at least one pixel has cover $\geq$ <code>thresh</code> , the species is considered "present". Otherwise the raster is excluded from the output. Defaults to 10.
<code>...</code>	other arguments, used for compatibility with other <code>prepSpeciesLayers</code> functions.
<code>dataYear</code>	Year for the data obtained. 2000, 2010, or 2020 (default) possible.
<code>dataVersion</code>	character. Data version for SCANFI. V2 is default.
<code>to</code> , <code>cropTo</code> , <code>projectTo</code> , <code>maskTo</code>	passed to <code>reproducible::prepInputs()</code> ; see <code>reproducible::postProcessTo()</code> . <code>prepSpeciesLayers_SCANFI()</code> only – the other <code>prepSpeciesLayers_*</code> () functions still take <code>studyArea</code> and <code>rasterToMatch</code> as formals. That function accepts the legacy pair through <code>...</code> and translates it onto these following the table in <code>reproducible::postProcessTo()</code> : a <code>rasterToMatch</code> on its own is <code>to</code> ; a <code>studyArea</code> on its own crops and masks but does not reproject (unless <code>useSAcrs</code>); and when both are given, the raster supplies extent, resolution, projection and alignment while the polygon supplies the mask. An argument passed explicitly wins.

Value

multilayer `SpatRaster` ("stack")

References

- Beaudoin, A., Bernier, P.Y., Guindon, L., Villemaire, P., Guo, X.J., Stinson, G., et al. (2014). Mapping attributes of Canada's forests at moderate resolution through kNN and MODIS imagery. *Canadian Journal of Forest Research*, 44, 521–532.
- Beaudoin, A., Bernier, P.Y., Villemaire, P., Guindon, L. & Guo, X.J. (2017). Species composition, forest properties and land cover types across Canada's forests at 250m resolution for 2001 and 2011. <https://doi.org/10.23687/EC9E2659-1C29-4DDB-87A2-6ACED147A990>
- Cosco, J.A. (2011). Common Attribute Schema (CAS) for Forest Inventories Across Canada. Timberline Natural Resource Group for Boreal Avian Modelling Project and Canadian BEACONS Project.

Guindon L., Villemaire P., Correia D.L.P., Manka F., Lacarte S., Smiley B. (2023). SCANFI: Spatialized Canadian National Forest Inventory data product. Natural Resources Canada, Canadian Forest Service, Laurentian Forestry Centre, Quebec, Canada. <https://doi.org/10.23687/18e6a919-53fd-41ce-b4e2-44a9707c52dc>

Guindon L., Manka F, Correia L.P. D., Villemaire P., Smiley B., Bernier P., Gauthier S., Beaudoin A., Boucher J., Boulanger Y. (2024). A new approach for Spatializing the Canadian National Forest Inventory (SCANFI) using Landsat dense time series. Canadian Journal of Forest Research. <https://doi.org/10.1139/cjfr-2023-0118>

Hermosilla, T., Wulder, M.A., White, J.C., Coops, N.C., Bater, C.W., Hobart, G.W. (2024). Characterizing long-term tree species dynamics in Canada’s forested ecosystems using annual time series remote sensing data. Forest Ecology and Management, 122313. <https://doi.org/10.1016/j.foreco.2024.122313>

Pickell, P.D. & Coops, N.C. (2016). Development of historical forest attribute layers using Landsat time series and kNN imputation for the western Canadian boreal forest. University of British Columbia.

randomStudyArea	Create default study areas for use with LandR modules
-----------------	---

Description

This simply re-exports `SpaDES.tools::randomStudyArea`

Usage

```
randomStudyArea(center = NULL, size = 10000, seed = NULL)
```

Arguments

center	SpatialPoints object specifying a set of coordinates and a projection. Default is an area in southern Alberta, Canada.
size	Numeric specifying the approximate size of the area in m^2. Default 1e4.
seed	Numeric indicating the random seed to set internally (useful for ensuring the same study area is produced each time).

replaceAgeInFires	<i>Replace stand age with time since last fire</i>
-------------------	--

Description

Replace stand age with time since last fire

Usage

```
replaceAgeInFires(standAgeMap, firePerimeters, startTime)
```

Arguments

standAgeMap	a raster layer stand age map
firePerimeters	the earliest fire date to allow
startTime	date of the last fire year to be considered (e.g., start of fire period counting backwards). If missing, last fire year available will be used.

Value

a raster layer stand age map corrected for fires, with an attribute vector of pixel IDs for which ages were corrected. If no corrections were applied the attribute vector is `integer(0)`.

Examples

```
## Not run:
randomPoly <- terra::vect(SpaDES.tools::randomStudyArea(size = 1e7))
randomPoly
ras2match <- terra::rast(resolution = 250,
                        ext = terra::ext(randomPoly),
                        crs = terra::crs(randomPoly))
ras2match <- terra::rasterize(randomPoly, ras2match)
tempDir <- file.path(tempdir(), "ex_replaceAgeInFires")

standAge <- reproducible::prepInputsStandAgeMap(
  destinationPath = tempDir,
  rasterToMatch = ras2match,
  fireURL = NA
) ## or NULL
attr(standAge, "imputedPixID")

firePerimeters <- reproducible::Cache(prepareInputsFireYear,
  url = paste0(
    "https://cwfis.cfs.nrcan.gc.ca/downloads",
    "/nfdb/fire_poly/current_version/NFDB_poly.zip"
  ),
  destinationPath = tempDir,
  rasterToMatch = ras2match
)
```

```
standAge <- replaceAgeInFires(standAge, firePerimeters)
attr(standAge, "imputedPixID")

## End(Not run)
```

rescale	<i>Rescale function (as in scales::rescale)</i>
---------	---

Description

This is a simple function copied from the scales package (almost the same).

Usage

```
rescale(x, to)
```

Arguments

x	a numeric vector
to	a numeric vector of length 2. The new range of values.

rmMissingCohorts	<i>Remove missing cohorts from cohortData based on pixelGroupMap</i>
------------------	--

Description

Remove missing cohorts from cohortData based on pixelGroupMap

Usage

```
rmMissingCohorts(
  cohortData,
  pixelGroupMap,
  cohortDefinitionCols = LandR::cohortDefinitionCols(),
  doAssertion = getOption("LandR.assertions", TRUE)
)
```

Arguments

cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPAct, and sumB.
pixelGroupMap	A RasterLayer with pixel values equal to a pixel group number that corresponds exactly to pixelGroup column in cohortData.
cohortDefinitionCols	the columns in cohortData that define unique cohorts (default: c("pixelGroup", "speciesCode", "age")).
doAssertion	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: getOption("LandR.assertions", TRUE).

Value

A list with 2 `data.table` objects, `cohortData` and `pixelGroupMap`, each updated based on missing `pixelGroups` in the other.

<code>scanfiUrlRemap</code>	<i>Resolve SCANFI Google Drive ids to the PredictiveEcology mirror</i>
-----------------------------	--

Description

LandR addresses SCANFI v2 files by their Google Drive ids, and some of those ids no longer resolve for an anonymous user – the 2020 land cover and 2020 stand age return 404, which stops the default SCANFI path of `Biomass_borealDataPrep`. The same files are mirrored on the PredictiveEcology object store on arbutus (Digital Research Alliance of Canada). This builds a `reproducible::makeUrlRemap()` hook from the manifest shipped with LandR, so a Drive id – or a SCANFI species *folder* – is fetched from the mirror instead, with no Google login and no change at any call site.

Usage

```
scanfiUrlRemap()
```

Details

LandR sets `options(reproducible.urlRemap = scanfiUrlRemap())` when it is loaded, but only if no remap is set already and `getOption("LandR.scanfiMirror", TRUE)` is TRUE. A remap of your own is never replaced; if you use one, include these rows in it (the manifest is `system.file("extdata", "arbutus_manifest_SCANFI_v2.csv", package = "LandR")`).

Value

A remap function for `reproducible.urlRemap`, or NULL if this reproducible has no `makeUrlRemap()`.

<code>scheduleDisturbance</code>	<i>Test whether disturbance should be scheduled</i>
----------------------------------	---

Description

Test whether disturbance should be scheduled

Usage

```
scheduleDisturbance(disturbanceLayer, currentYear)
```

Arguments

<code>disturbanceLayer</code>	a <code>RasterLayer</code> object
<code>currentYear</code>	time of simulation

Value

Logical indicating whether to schedule a disturbance event

Examples

```
## Not run:
doEvent <- scheduleDisturbance(sim$rstCurrentBurn, time(sim), disturbanceType = "Burn")

## End(Not run)
```

speciesEcoregionLatestYear

Pull out the values from speciesEcoregion table for current time

Description

Pull out the values from speciesEcoregion table for current time

Usage

```
speciesEcoregionLatestYear(speciesEcoregion, currentTime)
```

Arguments

speciesEcoregion

A data.table with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: speciesCode and ecoregionGroup, character representation of species and ecoregion groups respectively, maxB the maximum biomass for the species in a given 'ecoregion', maxANPP the maximum aboveground net primary productivity and SEP the species establishment probability. May contain columns inflationFactor (used to adjust maxB) and mANPPproportion (used to calculate maxANPP).

currentTime The current simulation time e.g., time(sim).

Value

The speciesEcoregion input object, but with data from only one year, the year that is less than or equal to the currentTime

`speciesEcoregionStack` *Create Stacks of the speciesEcoregion content*

Description

Each RasterStack show raster maps of one of the columns listed in columns and each RasterLayer will be one species.

Usage

```
speciesEcoregionStack(
  ecoregionMap,
  speciesEcoregion,
  columns = c("establishprob", "maxB", "maxANPP")
)
```

Arguments

<code>ecoregionMap</code>	The ecoregionMap, a raster of all the unique ecoregion groupings, which group together pixels of similar biophysical characteristics.
<code>speciesEcoregion</code>	A data.table with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: speciesCode and ecoregionGroup, character representation of species and ecoregion groups respectively, maxB the maximum biomass for the species in a given 'ecoregion', maxANPP the maximum aboveground net primary productivity and SEP the species establishment probability. May contain columns inflationFactor (used to adjust maxB) and maxANPPproportion (used to calculate maxANPP).
<code>columns</code>	The columns to use in the speciesEcoregion table. Default is c("establishprob", "maxB", "maxANPP")

Value

list of RasterStack or SpatRaster objects

`speciesInStudyArea` *Get species list in a given study area for a forest in Canada*

Description

speciesInStudyArea defaults to use a URL of a dataset uploaded to Google Drive that is from Canadian Forests, but a different factor raster can be passed e.g., from speciesPresentFromKNN.

Usage

```
speciesInStudyArea(
  studyArea,
  url = NULL,
  speciesPresentRas = NULL,
  sppEquivCol = NULL,
  dataSource = "SCANFI",
  dataYear = 2025,
  dPath = getOption("reproducible.destinationPath"),
  mergeHybridSpruce = getOption("LandR.mergeHybridSpruce", "engelmann")
)
```

Arguments

studyArea	A <code>SpatVector</code> or <code>sf</code> polygons object used as the principle study region, passed to <code>reproducible::prepInputs()</code> .
url	A url to get a speciesPresence raster e.g., from <code>peciesPresentFromKNN</code>
speciesPresentRas	A factor raster where the character string is a string of species names, separated by 2 underscores, sorted alphabetically. Can be produced with <code>speciesPresentFromKNN</code>
sppEquivCol	An optional column from <code>LandR::sppEquivalencies_CA</code> . If passed the KNN species will be returned according to this naming convention. It does not change which rows the returned <code>sppEquiv</code> has; a merged hybrid spruce row reports the target species' name in this column.
dataSource	Character. Either KNN, NTEMS, or SCANFI. Defaults to KNN to obtain species from layer created using species cover data from KNN. Also able to obtain species from SCANFI species cover layers and NTEMS Dominant Species Layer (though less species will be included from the latter).
dataYear	Numeric. Different source data years are available for different data sources. 2011 for KNN, 2011 for NTEMS, and every 5 years from 1985-2025 for SCANFI V2.
dPath	Passed to <code>destinationPath</code> in <code>preProcess</code> .
mergeHybridSpruce	Which species the hybrid white x Engelmann spruce (<code>Pice_eng_gla</code>) is merged into in the returned <code>sppEquiv</code> : "engelmann" (<code>Pice_eng</code>), "white" (<code>Pice_gla</code>), or NA to keep <code>Pice_eng_gla</code> as its own species. Only the hybrid being on the raster triggers the merge. Its rows then take the target species' <code>LandR</code> , <code>LANDIS_traits</code> and <code>sppEquivCol</code> names, and the target's row is added if the target is not on the raster. Defaults to <code>getOption("LandR.mergeHybridSpruce", "engelmann")</code> .

Value

A named list of length 3: `speciesRas` is a factor `RasterLayer`, `speciesList` is a character string containing the unique, sorted species on the `speciesRas`, for convenience, and `sppEquiv` is a `data.table`: the rows of `sppEquivalencies_CA` for those species, matched on the `LandR` column whatever naming the raster uses (so the rows are the same for any `sppEquivCol`), without

_Spp genus entries, keeping only species with LANDIS_traits, and with Pice_eng_gla merged as set by mergeHybridSpruce.

speciesPresentFromKNN *Make a species factor raster from KNN*

Description

This will download all KNN layers in Forests of Canada, and make a factor raster at resolution provided by res (larger is faster).

Usage

```
speciesPresentFromKNN(
  year = 2011,
  dPath = asPath("."),
  res = 2000,
  minPctCover = 10
)
```

Arguments

year	Default (and only implemented) is 2011. This will download the 2011 KNN data layers
dPath	A character string indicating where to download all the KNN layers
res	The resolution (one dimension, in m) for the resulting raster
minPctCover	An integer indicating what percent cover a species must have in a pixel to be considered present in that pixel.

Value

A SpatRaster object with 2 layers: "speciesPresent" is a factor, with a legend (i.e., it is numbers on a map, that correspond to a legend) and "numberSpecies" which represents the number of species in each pixel.

Examples

```
## Not run:
if (requireNamespace("googledrive", quietly = TRUE)) {
  ## Make the dataset
  speciesPresent <- speciesPresentFromKNN(dPath = file.path(tempdir(), "data_KNN"))

  ## To upload this:
  speciesPresentRas <- terra::rast(speciesPresent)[[1]]
  fn <- "SpeciesPresentInCanadianForests.tif"
  writeRaster(speciesPresentRas, file = fn)
  zipFn <- gsub(".tif", ".zip", fn)
  zip(files = dir(pattern = fn), zipFn)
```

```

out <- googledrive::drive_put(zipFn)

## Note: previous file (cropped to boreal forest)
## available at "10j78jJBeha5L6XDBBdWDAfimgNjYc9UD"
driveID <- "1J8fN7c1Zeqjd7yhiDWi13uoCBL80ensF"

## Get species list
sa <- LandR::randomStudyArea(size = 1e11)
species <- LandR::speciesInStudyArea(sa, dataSource = "KNN")
}

## End(Not run)

```

speciesPresentFromNTEMS

Make a species factor raster based on NTEMS Data

Description

This will download NTEMS dominant species layer for 2011 for forests of Canada, and make a factor raster at resolution provided by res (larger is faster).

Usage

```

speciesPresentFromNTEMS(
  dPath = asPath("."),
  res = 2400,
  year = 2011,
  rasterToMatch = NULL,
  studyArea = NULL,
  ...
)

```

Arguments

dPath	A character string indicating where to download all the NTEMS layers
res	The resolution (one dimension, in m) for the resulting raster
year	Default (and only implemented) is 2011. This will download the 2011 NTEMS species layer
rasterToMatch	A SpatRaster object to use as the template for all subsequent raster operations (i.e., the one used throughout the simulation). rasterToMatch should correspond to studyArea if used.
studyArea	A SpatVector or sf polygons object used as the principle study region, passed to reproducible::prepInputs() .
...	Additional arguments passed to reproducible::Cache() and equivalentName() . Also valid: outputPath, and studyAreaName.

Value

A `SpatRaster` object with 2 layers: "speciesPresent" is a factor, with a legend (i.e., it is numbers on a map, that correspond to a legend) and "numberSpecies" which represents the number of species in each pixel.

Examples

```
## Not run:
if (requireNamespace("googledrive", quietly = TRUE)) {
  # Make the dataset
  speciesPresent <- speciesPresentFromNTEMS(dPath = "~/data/NTEMS")

  # To upload this:
  speciesPresentRas <- terra::rast(speciesPresent)[[1]]
  fn <- "SpeciesPresentInCanadianForests_NTEMS.tif"
  writeRaster(speciesPresentRas, file = fn)
  zipFn <- gsub(".tif", ".zip", fn)
  zip(files = dir(pattern = fn), zipFn)
  out <- googledrive::drive_put(zipFn)

  ## Get species list
  sa <- LandR::randomStudyArea(size = 1e11)
  species <- LandR::speciesInStudyArea(sa, dataSource = "NTEMS")
}

## End(Not run)
```

speciesPresentFromSCANFI

Make a species factor raster based on SCANFI Data

Description

This will download SCANFI species layer for forests of Canada, and make a factor raster at resolution provided by res (larger is faster).

Usage

```
speciesPresentFromSCANFI(
  year = 2020,
  dataVersion = "V2",
  dPath = asPath("."),
  res = 2400,
  minPctCover = 10
)
```

Arguments

year	data year for SCANFI data. 2000, 2010, and 2020 possible for V1. 1985, 1990, 1995, 2000, 2005, 2010, 2015, 2020 (default), 2025 possible for V2.
dataVersion	Character. SCANFI product version for data. Default is currently V2. V1 also available.
dPath	A character string indicating where to download all the SCANFI layers
res	The resolution (one dimension, in m) for the resulting raster
minPctCover	An integer indicating what percent cover a species must have in a pixel to be considered present in that pixel.

Value

A `SpatRaster` object with 2 layers: "speciesPresent" is a factor, with a legend (i.e., it is numbers on a map, that correspond to a legend) and "numberSpecies" which represents the number of species in each pixel.

Examples

```
## Not run:
if (requireNamespace("googledrive", quietly = TRUE)) {
  # Make the dataset
  speciesPresent <- speciesPresentFromSCANFI(dPath = "~/data/SCANFI")

  # To upload this:
  speciesPresentRas <- speciesPresent[[1]]
  fn <- "SpeciesPresentInCanadianForests_SCANFI.tif"
  writeRaster(speciesPresentRas, file = fn)
  zipFn <- gsub(".tif", ".zip", fn)
  zip(files = dir(pattern = fn), zipFn)
  out <- googledrive::drive_put(zipFn)

  ## Get species list
  sa <- LandR::randomStudyArea(size = 1e11)
  species <- LandR::speciesInStudyArea(sa, dataSource = "SCANFI")
}

## End(Not run)
```

speciesTableUpdate *Change species table of parameters/traits*

Description

Changes longevity and shade tolerance values in the species table. Longevity values are changed to follow Burton & Cumming (1995) for the following species:

- *Abies balsamea*;

- *Abies lasiocarpa*;
- *Betula papyrifera*;
- *Larix laricina*;
- *Larix occidentalis*;
- *Picea engelmannii*;
- *Picea glauca*;
- *Picea mariana*;
- *Pinus banksiana*;
- *Pinus contorta*;
- *Pinus resinosa*;
- *Pinus strobus*;
- *Populus balsamifera* v. *balsamifera*;
- *Populus tremuloides*;
- *Pseudotsuga menziesii* var. *glauca*;
- *Pseudotsuga menziesii*;
- *Thuja plicata*;
- *Tsuga heterophylla*;
- *Tsuga mertensiana* x *heterophylla*;

Usage

```
speciesTableUpdate(species, speciesTable, sppEquiv = NULL, sppEquivCol = NULL)
```

Arguments

species a data.table with species traits such as longevity, shade tolerance, etc. Must have column speciesCode, with species names/IDs. The following is a list of default trait columns:

- "species" same as "speciesCode" – species ID name
- "speciesCode"
- "Area" – inherited from LANDIS-II default table, the Canadian ecoregion from which traits were derived. Not used during the simulation
- "firetolerance" – modulerelative (to other species) fire tolerance
- "growthcurve" and "mortalityshape" – growth curve shape parameters.
- "longevity" – maximum species age
- "postfireregen" – post-fire regeneration strategy ("serotiny", "resprout" or "none")
- "resproutprob" – probability of resprouting
- "resproutage_min" – minimum age at which species is capable of resprouting
- "resproutage_max" – maximum age at which species is capable of resprouting

- "seeddistance_eff" – effective dispersal distance
- "seeddistance_max" – maximum dispersal distance
- "shadetolerance" – *relative* (to other species) shade tolerance
- "sexualmature" – age at sexual maturity Known optional parameters added/needed by some modules (the user may add others for their own modules):
- "inflationFactor" – Biomass_speciesParameters module: inflation factor for maxB
- "growthCurveSource" – Biomass_speciesParameters module: how "growthcurve" was estimated
- "mANPPproportion" – Biomass_speciesParameters module: multiplication factor to calculate maxANPP from maxB
- "thermokarsttol" – Biomass_disturbances module: proportion of biomass surviving after thermokarst (i.e. permafrost thaw). Applied equally across cohorts. Parameters inherited from LANDIS-II default table, but not used in LandR at the moment:
- "leaflongevity"
- "wooddecayrate"
- "leafLignin"
- "hardsoft" Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.

speciesTable	A species traits table, with at least the following columns: • speciesCode an character representation of species; • Area a character of geographical area within a species range for which trait values are relevant; • longevity species longevity in years, sexualmature age in years at sexual maturity; • shadetolerance numeric value between 1-5 of relative shade tolerance (with respect to other species); • seeddistance_eff the "effective" seed dispersal distance; • seeddistance_max a numeric with the maximum seed dispersal distance; • mortalityshape and growthcurve: growth curve shape parameters. Other columns (e.g. fire-related traits) can also be included depending on LandR modules in use. Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.
sppEquiv	table with species name equivalencies between naming conventions (see sppEquivalencies_CA, also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
sppEquivCol	the column name to use from sppEquiv (see sppEquivalencies_CA). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.

Details

and only for the following speciesTable\$Areas:

- Boreal Shield West (BSW);
- Boreal Plains (BP);
- Montane Cordillera (MC).

Note that BSW and BP areas correspond more closely to the region considered in Table 2 of Burton & Cumming (1995), while MC will correspond to both tables.

Of the above species, shade tolerance values are changed for *Abies spp*, *Picea spp*, and *Tsuga spp* to better reflect the western boreal.

When different longevity/shade tolerance trait values exist for a given species, the minimum value across Area's (BSW, BP, MC) is kept.

ATTENTION: if none of species in species are from BSW, BP or MC area this function will not change any values.

All other species/Area trait values follow Dominic Cyr and Yan Boulanger's trait values (https://raw.githubusercontent.com/dcyr/LANDIS-II_IA_generalUseFiles/master/speciesTraits.csv).

Value

An updated species data. table

sppColors	Create species colour vector from a sppEquiv table
-----------	--

Description

Create species colour vector from a sppEquiv table

Usage

```
sppColors(sppEquiv, sppEquivCol, newVals = NULL, palette = "Accent")
```

Arguments

sppEquiv	table with species name equivalencies between naming conventions (see sppEquiv , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
sppEquivCol	the column name to use from sppEquiv (see sppEquiv). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.
newVals	An optional character vector of extra names to use, e.g., "Mixed".
palette	An RColorBrewer palette, e.g., "Accent". Can get RColorBrewer palette names from rownames(RColorBrewer::brewer.pal.info).

Value

A named vector of colour codes, where the names are the species names plus any extra names passed with newVals.

sppEquivalencies_CA *Species equivalencies table (sppEquiv) and sppEquivalencies_CA*

Description

sppEquiv is the species name equivalencies table that LandR functions and LandR SpaDES modules (e.g., Biomass_borealDataPrep, Biomass_core) use to translate species names between data sources and naming conventions (e.g., kNN, SCANFI and NTEMS species layers, LANDIS-II trait tables, CASFRI, common names). Each row is one taxon (a species, variety, hybrid or genus); each column is one naming convention or attribute.

sppEquivalencies_CA is the default sppEquiv shipped with LandR, covering Canadian tree species. It is used whenever a function's sppEquiv argument is not supplied, and is the template for building a custom sppEquiv: a data.table with the same column names (e.g., a subset of its rows) will work.

Usage

```
sppEquivalencies_CA
```

Format

A data.table with 204 rows and 30 columns (all character unless stated):

LANDIS_test species names from the LANDIS-II test parameter table; source: <https://raw.githubusercontent.com/LANDIS-II-Foundation/Extensions-Succession/master/biomass-succession-arch/trunk/tests/v6.0-2.0/species.txt>

LANDIS_traits species names from the LANDIS-II traits parameter table; source: https://raw.githubusercontent.com/dcyr/LANDIS-II_IA_generalUseFiles/master/speciesTraits.csv

LandR species names used by LandR; the common key for matching rows

KNN species names from the CFS kNN species layers; source: https://ftp.maps.canada.ca/pub/nrcan_rncan/Forests_Foret/canada-forests-attributes_attributs-forests-canada/2011-attributes_attributs-2011/

CASFRI species names from the Common Attribute Schema for Forest Resource Inventories (CASFRI); source: <https://github.com/CASFRI/CASFRI>

Latin_full full Latin names, following The Plant List

EN_generic_short short English common names, used as plot labels

EN_generic_full full English common names

Leading English common names followed by " leading", used as leading-species labels

Notes additional notes

Boreal KNN-style names for a default set of 39 species (42 rows), mostly from the boreal forest; the default sppEquivCol of `sppHarmonize()` and `loadkNNSpeciesLayers()`

Broadleaf logical: whether the species is a broadleaf

Type "Deciduous" or "Conifer"

PSP species names used by the ianmseddy/PSP_Clean module

ApproxFBP approximate Canadian Forest Fire Behaviour Prediction (FBP) System fuel type ("C2", "C3", "C5" or "C7"), for 7 pine, spruce and Douglas-fir rows

FuelClass fuel class used by PredictiveEcology/fireSense modules; one of "BlkSprc", "CedrMpl0ther", "DgFrPoPine", "LdgJkPine", "PopBrch", "RdWhPine" or "SprcFrLrch"

BC_forestry species codes used by the Government of British Columbia

AB_forestry species codes used by the Government of Alberta

SK_forestry species codes used by the Government of Saskatchewan

MB_forestry species codes used by the Government of Manitoba

ON_forestry species codes used by the Government of Ontario

QCPS species codes used in Québec permanent sample plot (PSP) data

NB_forestry species codes used by the Government of New Brunswick

NFI species codes used by the National Forest Inventory

CanfiCode integer: species codes used by Canada's Forest Inventory (CanFI)

CanfiNote notes on CanfiCode, e.g., generic genus codes

NTEMS_Species_Code integer: species codes in the National Terrestrial Ecosystem Monitoring System (NTEMS) dominant species layer

CBM_speciesID integer: species IDs used by the CFS Carbon Budget Model (CBM)

SCANFI species names used by the Spatialized Canadian National Forest Inventory (SCANFI) species layers

colorHex hexadecimal colour codes for plotting, used by `sppColors()`

Naming conventions

The same species (balsam fir) in each species name column:

Column	Format	Example
LandR	Genu_spe	Abie_bal
KNN, Boreal	Genu_Spe	Abie_Bal
NFI, SCANFI	GENU_SPE	ABIE_BAL
LANDIS_traits	GENU.SPE	ABIE.BAL
LANDIS_test	genuspec	abiebals
CASFRI	Genu spec	Abie bals
EN_generic_short	short common name	Bal fir
Leading	common name + " leading"	Balsam fir leading

Varieties and hybrids add a third part (e.g., Pice_eng_gla, PINU_CON_LAT). Genus-level entries use spp (e.g., LandR Pice_spp, KNN Pice_Spp, NFI PICE_SP).

Structure of the table

- A blank string ("") means the source has no equivalent for that taxon. Most columns are blank for most rows: only 115 rows have a KNN name, 37 a SCANFI name, 40 a LANDIS_traits name, and 42 a Boreal name.
- Several rows can share one name in a column. Varieties map onto their species (e.g., the three *Pseudotsuga menziesii* rows all have LandR Pseu_men), and Boreal Pinu_Con covers both lodgepole pine varieties. Choosing such a column as sppEquivCol therefore merges those taxa into one species.
- sppEquivCol (in modules, usually P(sim)\$sppEquivCol) names the column whose names are the species used in the simulation. Rows with a blank or NA in that column are not simulated. `sppHarmonize()` defaults it to "Boreal" when it cannot be determined from sppNameVector.
- To simulate a different set of species, or to merge species, subset the rows of sppEquivalencies_CA, or edit (or add) a naming column, and pass the result as sppEquiv with that column as sppEquivCol.

Helpers

- `equivalentName()`: translate names from one column to another; it finds the column the input names come from. `equivalentNameColumn()` returns that column's name.
- `sppEquivCheck()`: turn a character vector of names into a one-column sppEquiv, and if it lacks any ensureColumns, join it with sppEquivalencies_CA to get all columns.
- `sppHarmonize()`: make sppEquiv, sppEquivCol, sppNameVector and sppColorVect consistent with each other, subsetting sppEquiv to the species being simulated.
- `speciesInStudyArea()`: the species present in a study area, and the rows of sppEquivalencies_CA for them (element sppEquiv); see also the option LandR.mergeHybridSpruce in `LandROptions()`.
- `sppColors()`: a named colour vector for the species in sppEquivCol, using colorHex when every row has one.
- `assertSppVectors()`, `assertSpeciesPlotLabels()`: check sppEquiv against sppNameVector and sppColorVect, and that species plot labels are unique.

Columns used by LandR functions

- LandR: `speciesInStudyArea()` matches rows on it; default sppEquivCol of `prepSpeciesTable()`.
- LANDIS_traits: `prepSpeciesTable()`, `speciesTableUpdate()` and `speciesInStudyArea()`, to find species traits.
- KNN: `loadkNNSpeciesLayers()` (argument knnNamesCol).
- SCANFI: `loadSCANFISpeciesLayers()` (argument SCANFINamesCol).
- NTEMS_Species_Code: `prepInputs_NTEMS_DominantSpecies()` and `speciesInStudyArea()`.
- CASFRI: `loadCASFRI()`.
- EN_generic_short, Leading: plot labels; see `assertSpeciesPlotLabels()`.
- Type, Broadleaf: conifer vs. deciduous, e.g., in `vegTypeMapGenerator()` with mixedType = 2, and in `partitionBiomass()`.
- colorHex: `sppColors()`.
- FuelClass: fuel classes in fireSense modules.

Source

data-raw/sppEquivalencies_CA.csv, built by data-raw/sppEquivalencies_CA.R in the package source.

Examples

```
library(data.table)
sppEquiv <- LandR::sppEquivalencies_CA

## translate names between conventions
equivalentName(c("Abie_Bal", "Pice_Mar"), sppEquiv, column = "LandR")
equivalentName(c("Abie_Bal", "Pice_Mar"), sppEquiv, column = "LANDIS_traits")
equivalentNameColumn(c("ABIE_BAL", "PICE_MAR"), sppEquiv)

## a custom sppEquiv with three species, using the "Boreal" names
mySppEquiv <- sppEquiv[Boreal %in% c("Abie_Bal", "Pice_Gla", "Popu_Tre")]
sppColors(mySppEquiv, sppEquivCol = "Boreal", newVals = "Mixed")

## expand a vector of names into a full sppEquiv
sppEquivCheck(c("Abie_bal", "Pice_gla"), sppEquivCol = "LandR", ensureColumns = "KNN")
```

sppEquivCheck

Check and expand sppEquiv

Description

This will expand a sppEquiv object that is only a vector or only a one-column data.table into a many column data.table, if the columns that are present do not contain ensureColumns.

Usage

```
sppEquivCheck(sppEquiv, ensureColumns = NULL, sppEquivCol = NULL)
```

Arguments

sppEquiv	A character vector or data.table with named column(s). If this data.table does not have columns named ensureColumns, then it will attempt to merge this data.table with sppEquivalencies_CA to get ensureColumns.
ensureColumns	A character vector of column names that must be in sppEquiv. If these are not present, then the function will attempt to merge with sppEquivalencies_CA, so the column name(s) of sppEquiv must match column names in sppEquivalencies_CA.
sppEquivCol	Optional. Column in sppEquivalencies_CA to use for equivalent names when sppEquiv not provided (i.e., when sppEquivalencies_CA is used instead).

Value

A data.table with potentially all columns in sppEquivalencies_CA.

See Also

[sppEquivalencies_CA](#) for the sppEquiv table.

sppHarmonize	<i>Harmonize the three components that bring species into Biomass_** modules</i>
--------------	--

Description

This function will attempt to harmonize many potential issues/conflicts that may arise under different combinations of supplied objects to the three arguments. See manual for details.

Usage

```
sppHarmonize(
  sppEquiv,
  sppNameVector,
  sppEquivCol,
  sppColorVect,
  vegLeadingProportion = 0,
  studyArea,
  dPath = getOption("reproducible.destinationPath")
)
```

Arguments

- | | |
|----------------------|--|
| sppEquiv | table with species name equivalencies between naming conventions (see sppEquivalencies_CA , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA. |
| sppNameVector | A character vector of species to use. These species must all be from one naming convention, i.e., from one column in the sppEquiv. |
| sppEquivCol | A character string normally provided from the P(sim)\$sppEquivCol (see manual). If NA, the default, then this will try to determine which column the sppNameVector used and use that. If sppNameVector is NULL, then it will default to "Boreal". |
| sppColorVect | A named vector of colours to use for plotting. The names must conform with species name convention used (see sppEquivalencies_CA for possible naming conventions) and should also contain a colour for 'Mixed', when a mixed forest type is supported (see vegLeadingProportion parameter in vegTypeGenerator() for details on mixed forest types). |
| vegLeadingProportion | Numeric between 0-1, the share of a stand above which it stops being called mixed. Which share depends on what is being asked, and the two questions have their own options (see leadingProportions): with mixedType = 2 it is the whole broadleaf group against the whole conifer group, and defaults to mixedwoodProp() (LandR.mixedwoodProp, the NTEMS/EOSD and NFI value: coniferous or broadleaf |

	at 75% or more of total basal area, mixed wood when neither reaches it); otherwise it is one species' share and defaults to <code>leadingSpeciesProp()</code> (LandR. <code>leadingSpeciesProp</code> , which takes the <code>mixedwood</code> value unless it is set). Deciduous conifers count as conifers.
<code>studyArea</code>	A <code>SpatVector</code> or <code>sf</code> polygons object used as the principle study region, passed to <code>reproducible::prepInputs()</code> .
<code>dPath</code>	Passed to <code>speciesInStudyArea</code> (which then passes to <code>preProcess</code>)

Value

Returns a named list with the same names as the arguments. These should likely be assigned to the `sim` object in the module following this function call.

Examples

```
## not run. usage example within module
# sppOuts <- sppHarmonize(sim$sppEquiv, sim$sppNameVector, P(sim)$sppEquivCol)
# sim$sppEquiv <- sppOuts$sppEquiv
# sim$sppNameVector <- sppOuts$sppNameVector
# P(sim)$sppEquivCol <- sppOuts$sppEquivCol
```

`standAgeMapGenerator` *Produce stand age map based on cohortData*

Description

Produce stand age map based on `cohortData`

Usage

```
standAgeMapGenerator(
  cohortData,
  pixelGroupMap,
  weight = "biomass",
  doAssertion = getOption("LandR.assertions", FALSE)
)
```

Arguments

<code>cohortData</code>	A <code>data.table</code> with columns: <code>pixelGroup</code> , <code>ecoregionGroup</code> , <code>speciesCode</code> , and optionally <code>age</code> , <code>B</code> , <code>mortality</code> , <code>aNPAct</code> , and <code>sumB</code> .
<code>pixelGroupMap</code>	A <code>RasterLayer</code> with pixel values equal to a pixel group number that corresponds exactly to <code>pixelGroup</code> column in <code>cohortData</code> .
<code>weight</code>	variable by which to weight <code>cohortData</code> ages. one of "biomass" or NA. NA means use max unweighted age.
<code>doAssertion</code>	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: <code>getOption("LandR.assertions", TRUE)</code> .

Value

raster of the same type as pixelGroupMap.

statsModel	<i>The generic statistical model to run (lmer or glmer)</i>
------------	---

Description

This does a few things including R squared, gets the fitted values. It appears that running the models "as is" without this wrapper does not work with Cache. The return of the model in a list solves this problem. For Caching, do NOT omit .specialData via omitArgs. It was once recommended, on the reasoning that uniqueEcoregionGroups identifies the data well enough to stand in for it. It does not: those labels are generic sequential names ("01_NA", "02_NA", ...), so two study areas with the same number of ecoregion groups and the same modelFn produce the same cache key, and the second silently receives the first one's fitted model. That fails loudly only when the two species sets differ in model-matrix width; otherwise it imputes from a foreign model with no error at all. Digesting the data is cheap beside fitting the model.

Usage

```
statsModel(modelFn, uniqueEcoregionGroups, sumResponse, .specialData)
```

Arguments

- | | |
|-----------------------|---|
| modelFn | A quoted expression of type <code>package::model(Y ~ X, ...)</code> , omitting the data argument. E.g., <code>lme4::glmer(Y ~ X + (X G), family = poisson)</code> . |
| uniqueEcoregionGroups | Unique values of ecoregionGroups. This is the basis for the statistics. It is NOT a safe cache key on its own: see the note above about study areas colliding. |
| sumResponse | a sum of all the response variable values. |
| .specialData | The custom dataset required for the model. |

studyAreaEco	<i>Ecological boundaries within a study area</i>
--------------	--

Description

Extend a study area boundary to include the full ecological boundary polygons intersecting it.

Usage

```
studyAreaEco(  
  studyArea = NULL,  
  destinationPath = tempdir(),  
  type = c("ECOZONE", "ECOPROVINCE", "ECOREGION", "ECODISTRICT")  
)
```

Arguments

studyArea A `SpatVector` or `sf` polygons object used as the principle study region, passed to `reproducible::prepInputs()`.

destinationPath path to data directory where objects will be downloaded or saved to

type character. The polygon type to use: one of "ECODISTRICT", "ECOREGION", "ECOPROVINCE", or "ECOZONE".

Value

spatial polygons object of the same class as `studyArea`

Examples

```
## NOTE: runs locally, skipped on CI -- downloads from sis.agr.gc.ca, which throttles
## connections from CI runners (same predicate as testthat::skip_on_ci())
## using SpatVector objects
sa <- randomStudyArea(size = 1e9)
sa_eco <- studyAreaEco(studyArea = sa)

## using sf objects
sa_sf <- sf::st_as_sf(sa)
sa_eco_sf <- studyAreaEco(studyArea = sa_sf)

if (interactive()) {
  ggplot() +
    geom_sf(data = sa_eco_sf, fill = "gray") +
    geom_sf(data = sa_sf, fill = "violet", alpha = 0.3)
}
```

subsetDataSize	<i>How many rows to keep per group when subsampling for model fitting</i>
----------------	---

Description

`subsetDT()` keeps at most this many rows per group (e.g., per `ecoregionGroup` x `speciesCode`) when fitting the statistical models, and the `LandR` modules use it as the default of their `subsetData*Model` parameters. It is a function, and the number behind it lives once in `LandROptions()`, so a project can move it in one place with `options(LandR.subsetDataSize = ...)` and every module follows.

Usage

```
subsetDataSize()
```

Details

The default is 500L. It was 50 for years, chosen when the fits were computationally expensive; that subsample was small enough that repeated runs of the same simulation gave visibly different maxB (a median coefficient of variation of 10% across ecoregion x species, up to 62%, on a 60 km boreal test window). Ten times as many rows is well within what current machines fit comfortably.

Value

A single integer: the maximum rows per group to keep.

See Also

[subsetDT\(\)](#)

subsetDT	<i>Subset a data.table with random subsampling within by groups</i>
----------	---

Description

Subset a data.table with random subsampling within by groups

Usage

```
subsetDT(DT, by, doSubset = TRUE, indices = FALSE)
```

Arguments

DT	A data.table
by	Character vector of column names to use for groups
doSubset	Logical or numeric indicating the number of subsamples to use
indices	Logical. If TRUE, this will return vector of row indices only. Defaults to FALSE, i.e., return the subsampled data.table

Examples

```
library(data.table)
dt <- data.table(Lett = sample(LETTERS, replace = TRUE, size = 1000), Nums = 1:100)
dt1 <- subsetDT(dt, by = "Lett", doSubset = 3)
```

sumRastersBySpecies	<i>Function to sum rasters of species layers</i>
---------------------	--

Description

Function to sum rasters of species layers

Usage

```
sumRastersBySpecies(speciesLayers, layersToSum, filenameToSave, newLayerName)
```

Arguments

- speciesLayers A RasterStack or RasterLayer that should contain species cover data in the study area
- layersToSum names/indices of layers to be summed - optional
- filenameToSave file path to save output raster
- newLayerName name of the output raster layer

uniqueCohortDefinition	<i>The columns in a cohortData that define "unique"</i>
------------------------	---

Description

If two pixels have identical values in all of these columns, they are the same pixelGroup.

Usage

```
uniqueCohortDefinition  
  
uniqueSpeciesEcoregionDefinition
```

updateCohortData	<i>Add cohorts to cohortData and pixelGroupMap</i>
------------------	--

Description

This is a wrapper for generatePixelGroups, initiateNewCohort and updates to pixelGroupMap via assignment to new pixelIndex values in newPixelCohortData. By running these all together, there is less chance that they will diverge. There are some checks internally for consistency.

Calculate new values for B, add age, then rbindlist this with cohortData.

Usage

```
updateCohortData(
  newPixelCohortData,
  cohortData,
  pixelGroupMap,
  currentTime,
  speciesEcoregion,
  treedFirePixelTableSinceLastDisp = NULL,
  successionTimestep,
  cohortDefinitionCols = LandR::cohortDefinitionCols(),
  initialB = 10,
  verbose = getOption("LandR.verbose", TRUE),
  doAssertion = getOption("LandR.assertions", TRUE)
)

.initiateNewCohorts(
  newPixelCohortData,
  cohortData,
  pixelGroupMap,
  currentTime,
  cohortDefinitionCols = LandR::cohortDefinitionCols(),
  speciesEcoregion,
  successionTimestep,
  initialB = 10
)
```

Arguments

newPixelCohortData	must be a complete cohortData object with newly created cohorts. They do not have to have pixelGroup values yet; they can be overlapping with cohortData, (i.e., they can be regenerated on empty pixels or on already occupied pixels). Must contain the columns: pixelIndex, speciesCode, ecoregionGroup. The remaining 4 (see cohortData) will be created with 0s.
cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPPAct, and sumB.

pixelGroupMap	A RasterLayer with pixel values equal to a pixel group number that corresponds exactly to pixelGroup column in cohortData.
currentTime	The current simulation time e.g., time(sim).
speciesEcoregion	A data.table with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: speciesCode and ecoregionGroup, character representation of species and ecoregion groups respectively, maxB the maximum biomass for the species in a given 'ecoregion', maxANPP the maximum aboveground net primary productivity and SEP the species establishment probability. May contain columns inflationFactor (used to adjust maxB) and mANPPproportion (used to calculate maxANPP).
treedFirePixelTableSinceLastDisp	A data.table with at least 2 columns, pixelIndex and pixelGroup. This will be used in conjunction with cohortData and pixelGroupMap to ensure that everything matches correctly.
successionTimestep	The time between successive seed dispersal events. In LANDIS-II, this is called "Succession Timestep".
cohortDefinitionCols	the columns in cohortData that define unique cohorts (default: c("pixelGroup", "speciesCode", "age")).
initialB	the initial biomass of new cohorts. Defaults to 10. If NULL or NA, initial cohort biomass is calculated as in LANDIS-II Biomass Succession Extension v3.2.1 (Scheller & Miranda, 2015): $\text{initialB} = \text{asInteger}(\text{pmin}(\text{maxANPP}, \text{asInteger}(\text{pmax}(1, \text{maxANPP} * \exp(-1.6 * \text{sumB} / \text{maxB_eco}))))$ where maxANPP and maxB_eco are the maximum ANPP and B parameters of the species in question within the pixel's ecolocation, and sumB is the total stand biomass excluding cohorts with ages less than successionTimestep.
verbose	Controls message output. Defaults to getOption("LandR.verbose")
doAssertion	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: getOption("LandR.assertions", TRUE).

Details

Does the following:

1. add *new cohort* (not survivor) data into cohortData;
2. assign initial B and age for new cohort;
3. assign the new pixelGroup to the pixels that have new cohort;
4. update the pixelGroup map.

Note that if newPixelCohortData is generated after a disturbance it must contain a type column indicating the origin of the cohorts (e.g. "survivor", "serotiny", "resprouting"). "Survivor" cohorts will not be added to the output objects, as they are assumed to be accounted for in the input cohortData and, correspondingly, pixelGroupMap.

Value

A list of length 2, cohortData and pixelGroupMap, with newPixelCohortData inserted.

A data.table with a new rbindlisted cohortData

updateCohortDataPostHarvest

Add cohorts to cohortData and pixelGroupMap

Description

This is a wrapper for generatePixelGroups, initiateNewCohort and updates to pixelGroupMap via assignment to new pixelIndex values in newPixelCohortData. By running these all together, there is less chance that they will diverge. There are some checks internally for consistency.

Usage

```
updateCohortDataPostHarvest(
  newPixelCohortData,
  cohortData,
  pixelGroupMap,
  currentTime,
  speciesEcoregion,
  treedHarvestPixelTable = NULL,
  successionTimestep,
  provenanceTable,
  trackPlanting = FALSE,
  initialB = 10,
  cohortDefinitionCols = LandR::cohortDefinitionCols(),
  verbose = getOption("LandR.verbose", TRUE),
  doAssertion = getOption("LandR.assertions", TRUE)
)
```

Arguments

newPixelCohortData	must be a complete cohortData object with newly created cohorts. They do not have to have pixelGroup values yet; they can be overlapping with cohortData, (i.e., they can be regenerated on empty pixels or on already occupied pixels). Must contain the columns: pixelIndex, speciesCode, ecoregionGroup. The remaining 4 (see cohortData) will be created with 0s.
cohortData	A data.table with columns: pixelGroup, ecoregionGroup, speciesCode, and optionally age, B, mortality, aNPAct, and sumB.
pixelGroupMap	A RasterLayer with pixel values equal to a pixel group number that corresponds exactly to pixelGroup column in cohortData.
currentTime	The current simulation time e.g., time(sim).

<code>speciesEcoregion</code>	A <code>data.table</code> with species-ecoregion-specific species trait values. Ecoregion refers to "ecolocation", a categorical variable grouping sites with similar biophysical characteristics. The table should have at least the following columns: <code>speciesCode</code> and <code>ecoregionGroup</code> , character representation of species and ecoregion groups respectively, <code>maxB</code> the maximum biomass for the species in a given 'ecoregion', <code>maxANPP</code> the maximum aboveground net primary productivity and <code>SEP</code> the species establishment probability. May contain columns <code>inflationFactor</code> (used to adjust <code>maxB</code>) and <code>mANPPproportion</code> (used to calculate <code>maxANPP</code>).
<code>treedHarvestPixelTable</code>	A <code>data.table</code> with at least 2 columns, <code>pixelIndex</code> and <code>pixelGroup</code> . This will be used in conjunction with <code>cohortData</code> and <code>pixelGroupMap</code> to ensure that everything matches correctly.
<code>successionTimestep</code>	The time between successive seed dispersal events. In LANDIS-II, this is called "Succession Timestep".
<code>provenanceTable</code>	A <code>data.table</code> with three columns: New cohorts are initiated at the <code>ecoregionGroup</code> <code>speciesEcoregion</code> from the corresponding <code>speciesEcoregion</code> listed in the <code>Provenance</code> column
<code>trackPlanting</code>	if true, planted cohorts in <code>cohortData</code> are tracked with TRUE in column 'planted'
<code>initialB</code>	the initial biomass of new cohorts. Defaults to ten, even if NA/NULL is passed.
<code>cohortDefinitionCols</code>	the columns in <code>cohortData</code> that define unique cohorts (default: <code>c("pixelGroup", "speciesCode", "age")</code>).
<code>verbose</code>	Controls message output. Defaults to <code>getOption("LandR.verbose")</code>
<code>doAssertion</code>	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: <code>getOption("LandR.assertions", TRUE)</code> .

Details

Does the following:

1. add new cohort data into `cohortData`;
2. assign initial B and age for new cohort;
3. assign the new `pixelGroup` to the pixels that have new cohort;
4. update the `pixelGroup` map.

Value

A list of length 2, `cohortData` and `pixelGroupMap`, with `newPixelCohortData` inserted.

updateSpeciesTable	<i>Customize species trait table values</i>
--------------------	---

Description

Customize species trait table values

Usage

```
updateSpeciesTable(speciesTable, params)
```

Arguments

speciesTable	A species traits table, with at least the following columns: • speciesCode an character representation of species; • Area a character of geographical area within a species range for which trait values are relevant; • longevity species longevity in years, sexualmature age in years at sexual maturity; • shadetolerance numeric value between 1-5 of relative shade tolerance (with respect to other species); • seeddistance_eff the "effective" seed dispersal distance; • seeddistance_max a numeric with the maximum seed dispersal distance; • mortalityshape and growthcurve: growth curve shape parameters. Other columns (e.g. fire-related traits) can also be included depending on LandR modules in use. Please see the LANDIS-II Biomass Succession Extension v3.2.1 manual (Scheller and Miranda 2015) for further detail.
params	A named list (of parameters) of named lists (by species), with species traits overrides (e.g., <code>list(seeddistance_eff = list(Abie_sp = 25))</code>).

Author(s)

Alex Chubaty and Ceres Barros

vegTypeGenerator	<i>Generate and add vegetation type column to cohortData</i>
------------------	--

Description

This function is a simplification of `vegTypeMapGenerator` and instead of generating a map, it adds the vegetation type column to the `cohortData` table.

Usage

```
vegTypeGenerator(
  x,
  vegLeadingProportion = NULL,
  mixedType = 2,
  sppEquiv = NULL,
  sppEquivCol,
  pixelGroupColName = "pixelGroup",
  doAssertion = getOption("LandR.assertions", TRUE),
  ...
)
```

Arguments

<code>x</code>	A cohortData object
<code>vegLeadingProportion</code>	Numeric between 0-1, the share of a stand above which it stops being called mixed. Which share depends on what is being asked, and the two questions have their own options (see leadingProportions): with <code>mixedType = 2</code> it is the whole broadleaf group against the whole conifer group, and defaults to <code>mixedwoodProp()</code> (<code>LandR.mixedwoodProp</code> , the NTEMS/EOSD and NFI value: coniferous or broadleaf at 75% or more of total basal area, mixed wood when neither reaches it); otherwise it is one species' share and defaults to <code>leadingSpeciesProp()</code> (<code>LandR.leadingSpeciesProp</code> , which takes the <code>mixedwood</code> value unless it is set). Deciduous conifers count as conifers.
<code>mixedType</code>	An integer defining whether mixed stands are: not differentiated (0), any kind of species admixture (1), or deciduous mixed with conifer (2; default).
<code>sppEquiv</code>	table with species name equivalencies between naming conventions (see sppEquivalencies_CA , also <code>?sppEquiv</code>). For functions that have <code>mixedType</code> , this only necessary if <code>mixedType == 2</code> . If not provided and <code>mixedType == 2</code> , will attempt to use <code>LandR:sppEquivalencies_CA</code> .
<code>sppEquivCol</code>	the column name to use from <code>sppEquiv</code> (see sppEquivalencies_CA). For functions that have <code>mixedType</code> , only necessary if <code>mixedType == 2</code> . If <code>sppEquiv</code> is not provided and <code>mixedType == 2</code> , will use the column of <code>LandR:sppEquivalencies_CA</code> that best matches the species names.
<code>pixelGroupColName</code>	Name of the column in <code>pixelGroup</code> to use.
<code>doAssertion</code>	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: <code>getOption("LandR.assertions", TRUE)</code> .
<code>...</code>	Additional arguments.

Value

`x` with a new column, 'leading', coding the vegetation type of each group defined by `pixelGroupColName`

Author(s)

Eliot McIntire, Ceres Barros, Alex Chubaty

Examples

```
library(data.table)
x <- data.table(
  pixelGroup = rep(1:2, each = 2), B = c(100, 200, 20, 400),
  speciesCode = rep(c("Pice_Gla", "Popu_Tre"), 2)
)
vegTypeGenerator(x)
```

vegTypeMapGenerator	<i>Generate vegetation type map</i>
---------------------	-------------------------------------

Description

Generate vegetation type map

Usage

```
vegTypeMapGenerator(x, ...)

## Default S3 method:
vegTypeMapGenerator(x, ..., doAssertion = getOption("LandR.assertions", FALSE))

## S3 method for class 'data.table'
vegTypeMapGenerator(
  x,
  pixelGroupMap,
  vegLeadingProportion = NULL,
  mixedType = 2,
  sppEquiv = NULL,
  sppEquivCol,
  colors,
  pixelGroupColName = "pixelGroup",
  doAssertion = getOption("LandR.assertions", TRUE),
  ...
)
```

Arguments

x	Either a cohortData object or a speciesCover RasterStack/SpatRaster
...	Additional arguments.
doAssertion	A logical indicating whether some internal tests should be run to ensure the function is running correctly. Default: getOption("LandR.assertions", TRUE).
pixelGroupMap	A RasterLayer with pixel values equal to a pixel group number that corresponds exactly to pixelGroup column in cohortData.

vegLeadingProportion

Numeric between 0-1, the share of a stand above which it stops being called mixed. Which share depends on what is being asked, and the two questions have their own options (see [leadingProportions](#)): with `mixedType = 2` it is the whole broadleaf group against the whole conifer group, and defaults to `mixedwoodProp()` (`LandR.mixedwoodProp`, the NTEMS/EOSD and NFI value: coniferous or broadleaf at 75% or more of total basal area, mixed wood when neither reaches it); otherwise it is one species' share and defaults to `leadingSpeciesProp()` (`LandR.leadingSpeciesProp`, which takes the `mixedwood` value unless it is set). Deciduous conifers count as conifers.

mixedType An integer defining whether mixed stands are: not differentiated (0), any kind of species admixture (1), or deciduous mixed with conifer (2; default).

sppEquiv table with species name equivalencies between naming conventions (see [sppEquivalencies_CA](#), also `?sppEquiv`). For functions that have `mixedType`, this only necessary if `mixedType == 2`. If not provided and `mixedType == 2`, will attempt to use `LandR:sppEquivalencies_CA`.

sppEquivCol the column name to use from `sppEquiv` (see [sppEquivalencies_CA](#)). For functions that have `mixedType`, only necessary if `mixedType == 2`. If `sppEquiv` is not provided and `mixedType == 2`, will use the column of `LandR:sppEquivalencies_CA` that best matches the species names.

colors A named vector of colour codes. The names MUST match the names of species in `cohortData$speciesCode`, plus an optional "Mixed" colour.

pixelGroupName

Name of the column in `pixelGroup` to use.

Author(s)

Eliot McIntire, Ceres Barros, Alex Chubaty

Examples

```
library(data.table)
library(terra)
x <- data.table(
  pixelGroup = rep(1:2, each = 2), B = c(100, 200, 20, 400),
  speciesCode = rep(c("Pice_Gla", "Popu_Tre"), 2)
)
pixelGroupMap <- rast(ext(0, 3, 0, 3), res = 1)
pixelGroupMap[] <- sample(1:2, size = 9, replace = TRUE)
vtm <- vegTypeMapGenerator(x, pixelGroupMap = pixelGroupMap)
```

Description

Create a summaries of vegetation type transitions
 Plot vegetation type transitions

Usage

```
vtm2conifdecid(vtm, sppEquiv = NULL, sppEquivCol = "LandR", zones)

vegTransitions(vtm, zones, field, times, na.rm = FALSE, dest = ".")

plotVegTransitions(transitions_df)
```

Arguments

vtm	character vector of file paths to vegetation type maps (see vegTypeMapGenerator()).
sppEquiv	table with species name equivalencies between naming conventions (see sppEquivalencies_CA , also ?sppEquiv). For functions that have mixedType, this only necessary if mixedType == 2. If not provided and mixedType == 2, will attempt to use LandR::sppEquivalencies_CA.
sppEquivCol	the column name to use from sppEquiv (see sppEquivalencies_CA). For functions that have mixedType, only necessary if mixedType == 2. If sppEquiv is not provided and mixedType == 2, will use the column of LandR::sppEquivalencies_CA that best matches the species names.
zones	sf polygons object delineating the reporting area (zone) boundaries.
field	character string of the column name in zones to use for grouping.
times	numeric vector of years corresponding to vtm.
na.rm	logical. If TRUE, remove rows with NA values in vegType (they won't appear as a stratum in the alluvial diagram). If FALSE, these NA values will be replaced with "_NA_" so transitions between vegetated and non-vegetated pixels can be visualized.
dest	character, specifying a destination directory
transitions_df	A data frame with columns pixelID, zone, vegType, and time. (i.e., output of vegTransitions()).

Value

- vtm2conifdecid() returns a character vector of file paths to the conifer-deciduous maps.
- vegTransitions() returns a arrow dataset with columns pixelID, zone, vegType, and time.
- plotVegTransitions() returns a list of ggplot objects, one for each zone.

Note

creating these plots for large landscapes can be computationally intensive (time and memory use).

WardKernel	<i>Ward Dispersal Kernel – vectorized, optimized for speed</i>
------------	--

Description

A probability distribution used in LANDIS-II.

Usage

```
Ward(dist, cellSize, effDist, maxDist, k, b, algo = 2)
```

Arguments

dist	A vector of distances to evaluate kernel against
cellSize	A numeric, length 1, of the cell resolution (e.g., res(raster))
effDist	A vector of effective distance (parameter in kernel), with same length as dist
maxDist	A vector of maximum distance (parameter in kernel), with same length as dist
k	A parameter in the kernel
b	A parameter in the kernel
algo	Either 1 or 2. 1 is "simpler code" as it uses ifelse; 2 is faster and is default.

Author(s)

Eliot McIntire

wetlandToLCC	<i>Add wetland classes to a land-cover map</i>
--------------	--

Description

Marks the pixels a site layer calls wet with the NTEMS wetland codes: wetTreecode (81) where the land cover is treed, wetCode (80) otherwise. Pixels that are not wet are returned unchanged, and so are water and any pixel already carrying a wetland code.

Usage

```
wetlandToLCC(
  lcc,
  wet,
  treedClasses = c(210, 220, 230, 240),
  keepClasses = 20,
  wetCode = 80L,
  wetTreecode = 81L
)
```

Arguments

<code>lcc</code>	Land cover: a <code>SpatRaster</code> or a numeric vector.
<code>wet</code>	Site layer of the same shape: non-zero means wet, 0 or NA not. Typically prepInputs_CWIM() .
<code>treedClasses</code>	Land-cover codes that become <code>wetTreedCode</code> when wet.
<code>keepClasses</code>	Codes never overwritten (default: water).
<code>wetCode, wetTreedCode</code>	Codes for wet non-treed and wet treed pixels.

Details

"Treed" includes 240 by default: the disturbed-forest code LandR assigns to forest land that is not currently treed, which is still forest ground.

Value

`lcc`, with wet pixels recoded, in the same form it was given.

See Also

[prepInputs_CWIM\(\)](#)

Index

* **datasets**
 sppEquivalencies_CA, 139
 .MLLMaxBPartialPlotData, 10
 .MLLMaxBplotData, 11
 .NAvalueFlag, 12
 .applyForestLand(), 109–111
 .compareCRS (.compareRas), 7
 .compareRas, 7
 .defaultForestLandYears(), 109, 111
 .fitNLMwCovariates, 7
 .getMaxBCoefs, 9
 .initiateNewCohorts (updateCohortData), 149
 .interpolateLightProb, 9
 .libPaths, 68
 .projectExtent (.NAvalueFlag), 12
 .stack (.NAvalueFlag), 12

addNoPixel2CohortData, 13
addPixels2CohortData, 14
adjustAgeToLongevity, 14
aggregateRasByDT, 15
asInt (.NAvalueFlag), 12
asInteger, 16
assert1, 16
assert2 (assert1), 16
assertCohortData (assert1), 16
assertCohortDataAttr (assert1), 16
assertCohortDataERG (assert1), 16
assertColumns (assert1), 16
assertERGs (assert1), 16
assertFireToleranceDif (assert1), 16
assertions, 72
assertions (assert1), 16
assertPixelCohortData (assert1), 16
assertPixelCohortDataValid (assert1), 16
assertPostFireDist (assert1), 16
assertPostPartialDist, 21
assertRepsAllCohortData (assert1), 16
assertRstLCChange (assert1), 16

assertSpeciesEcoregionCohortDataMatch (assert1), 16
assertSpeciesLayers (assert1), 16
assertSpeciesPlotLabels (assert1), 16
assertSpeciesPlotLabels(), 141
assertSpeciesTable (assert1), 16
assertSpeciesTable(), 65
assertSpeciesTableRaw (assert1), 16
assertSpeciesTableRaw(), 64, 65
assertSppMaxBMaxANPP (assert1), 16
assertSppVectors (assert1), 16
assertSppVectors(), 141
assertStandAgeMapAttr (assert1), 16
assertUniqueCohortData (assert1), 16
assignLightProb, 22

base::sum(), 76
bbmle::mle2, 8
bbmle::mle2(), 8, 11, 12, 59, 66, 99
biomod2::BIOMOD_EnsembleForecasting, 23
biomod2::BIOMOD_EnsembleForecasting(), 23
biomod2::BIOMOD_EnsembleModeling, 23, 24
biomod2::BIOMOD_FormatingData, 25
biomod2::BIOMOD_Modeling, 24, 25
biomod2::BIOMOD_Projection, 23, 26
biomodEnsembleFrcstWrapper, 23
biomodEnsembleProjMaps, 23
biomodEnsembleWrapper, 24
biomodModelingWrapper, 24
biomodProjWrapper, 25
BioSIM_extractPoints, 26
BioSIM_getMPBSLR, 27
BioSIM_getWindAnnual, 27
BioSIM_getWindMonthly, 28

calc_raster_counts, 29

- `calc_raster_stats (calc_raster_counts)`, 29
- `calcSeverityB`, 32
- `calcSiteShade`, 33
- `calculateClimateEffect`, 34
- `CASFRItoSpRasts`, 34
- `checkSpeciesTraits`, 35
- `cohortDefinitionCols`, 37
- `Colors`, 37
- `Colours (Colors)`, 37
- `columnsForPixelGroups`, 38
- `convert_SCANFI_LCC_codes`, 38
- `convertUnwantedLCC`, 39
- `convertUnwantedLCC()`, 96
- `convertUnwantedLCC2`, 42
-
- `defineFlammable`, 43
- `describeCohortData`, 43
- `Disturbances (FireDisturbance)`, 51
- `doResprouting`, 44
- `doSerotiny`, 46
- `dropTerm`, 48
- `dummy-inputs (makeDummyEcoregionMap)`, 83
- `dummyInputs (makeDummyEcoregionMap)`, 83
-
- `ecoregionProducer`, 48
- `equivalentName`, 49
- `equivalentName()`, 78, 79, 91, 133, 141
- `equivalentNameColumn (equivalentName)`, 49
- `equivalentNameColumn()`, 141
- `extractMaxB`, 50
-
- `fasterizeFromSp`, 50
- `FireDisturbance`, 51
- `FireDisturbancePM (FireDisturbance)`, 51
- `fitNLMMModels`, 57
- `forestLandMask`, 59
- `forestLandMask()`, 109–111
-
- `gadm_canada`, 60
- `gadm_canada()`, 30
- `generatePixelGroups`, 61
- `genericExtract`, 61
- `genPGsPostDisturbance`, 62
- `getSpeciesTable`, 63
- `getSpeciesTable()`, 21
- `getWildfire_NFI`, 65
- `ggplot2::labs()`, 66, 99
-
- `ggplotMML_maxB`, 66
-
- `imputeBadAgeModelDefault`, 67
- `installBioSIM`, 68
- `isInt (.NAvalueFlag)`, 12
-
- `LANDISDisp`, 68
- `LANDISDisp()`, 73
- `LandR (LandR-package)`, 6
- `LandR-deprecated (makeVegTypeMap)`, 90
- `LandR-package`, 6
- `LandROptions`, 72
- `LandROptions()`, 6, 93, 141, 146
- `lccMapGenerator`, 73
- `lccMapGenerator()`, 73, 93
- `leadingByPixelGroup`, 75
- `leadingProportions`, 74, 75, 91, 104, 143, 154, 156
- `leadingProportions (mixedwoodProp)`, 92
- `leadingSpeciesProp (mixedwoodProp)`, 92
- `leadingSpeciesProp()`, 72
- `loadCASFRI`, 35, 76
- `loadCASFRI()`, 141
- `loadkNNSpeciesLayers`, 77
- `loadkNNSpeciesLayers()`, 140, 141
- `loadkNNSpeciesLayersValidation (makeVegTypeMap)`, 90
- `loadSCANFISpeciesLayers`, 78
- `loadSCANFISpeciesLayers()`, 141
-
- `makeAndCleanInitialCohortData`, 80
- `makeBiomassMap`, 81
- `makeCohortDataFiles`, 82
- `makeDummyEcoregionFiles (makeDummyEcoregionMap)`, 83
- `makeDummyEcoregionMap`, 83
- `makeDummyRawBiomassMap (makeDummyEcoregionMap)`, 83
- `makeDummyRstLCC (makeDummyEcoregionMap)`, 83
- `makeDummyStandAgeMap (makeDummyEcoregionMap)`, 83
- `makeEcoregionDT`, 84
- `makeEcoregionMap`, 85
- `makeMinRelativeB`, 86
- `makeMinRelativeB()`, 34
- `makePickellStack`, 86
- `makePixelGroupMap`, 87
- `makePixelTable`, 87

- makeSpeciesEcoregion, 88
- makeVegTypeMap, 90
- minRelativeBDefaults, 92
- mixedwoodProp, 92
- mixedwoodProp(), 73
- modifySpeciesAndSpeciesEcoregionTable, 93
- NAcover2zero, 94
- nonForestedPixels, 95
- options(), 6
- opts.LandR (LandROptions), 72
- overlayLCCs, 95
- overlayStacks, 97
- partialggplotMLL_maxB, 98
- partitionBiomass, 99
- partitionBiomass(), 141
- PeatlandThermokarst (FireDisturbance), 51
- pixelFate, 100
- plantNewCohorts, 100
- plot_raster, 101
- plot_raster_stats (calc_raster_counts), 29
- plotLeadingSpecies, 102
- plotSpatial, 103
- plotVegTransitions (vtm2conifdecid), 156
- plotVTM, 103
- plotVTM(), 73
- prepEcoregions, 104
- prepEcozonesRst, 105
- prepInputs_CWIM, 106
- prepInputs_CWIM(), 159
- prepInputs_FAO_forest, 107
- prepInputs_NTEMS_DominantSpecies, 108
- prepInputs_NTEMS_DominantSpecies(), 141
- prepInputs_NTEMS_LCC_FAO, 109
- prepInputs_NTEMS_LCC_FAO(), 72
- prepInputs_NTEMS_Nonforest, 110
- prepInputs_SCANFI_LCC_FAO, 110
- prepInputs_SCANFI_LCC_FAO(), 72
- prepInputs_SCANFI_structure, 111
- prepInputsCanDEM, 112
- prepInputsCanDEM(), 26
- prepInputsEcoregion, 113
- prepInputsFireYear, 113
- prepInputsLCC, 114
- prepInputsMainInput (prepInputsSpecies), 115
- prepInputsSpecies, 115
- prepInputsSpeciesEcoregion, 116
- prepInputsStandAgeMap, 117
- prepRasterToMatch, 119
- prepRawBiomassMap, 120
- prepRawBiomassMap(), 112
- prepSpeciesLayers (prepSpeciesLayers_KNN), 121
- prepSpeciesLayers_CASFRI (prepSpeciesLayers_KNN), 121
- prepSpeciesLayers_CASFRI(), 121
- prepSpeciesLayers_ForestInventory (prepSpeciesLayers_KNN), 121
- prepSpeciesLayers_ForestInventory(), 121
- prepSpeciesLayers_KNN, 121
- prepSpeciesLayers_KNN(), 121
- prepSpeciesLayers_KNN2011 (makeVegTypeMap), 90
- prepSpeciesLayers_MBFRI (prepSpeciesLayers_KNN), 121
- prepSpeciesLayers_MBFRI(), 121
- prepSpeciesLayers_NTEMS (prepSpeciesLayers_KNN), 121
- prepSpeciesLayers_NTEMS(), 121
- prepSpeciesLayers_ONFRI (prepSpeciesLayers_KNN), 121
- prepSpeciesLayers_ONFRI(), 121
- prepSpeciesLayers_Pickell (prepSpeciesLayers_KNN), 121
- prepSpeciesLayers_Pickell(), 121
- prepSpeciesLayers_SCANFI (prepSpeciesLayers_KNN), 121
- prepSpeciesLayers_SCANFI(), 121
- prepSpeciesTable (getSpeciesTable), 63
- prepSpeciesTable(), 21, 141
- randomStudyArea, 125
- raster::intersect, 115
- raster::mask(), 43
- raster_stats (calc_raster_counts), 29
- reclass (.NAvalueFlag), 12
- replaceAgeInFires, 126
- reproducible::Cache, 23–26
- reproducible::Cache(), 41, 78, 79, 91, 112, 121, 133

- reproducible::makeUrlRemap(), 128
- reproducible::postProcess(), 43, 79, 124
- reproducible::postProcessTo(), 79, 124
- reproducible::prepInputs, 115
- reproducible::prepInputs(), 39, 78, 79, 91, 92, 106, 112, 113, 117, 118, 120, 121, 124, 131, 133, 144, 146
- rescale, 127
- rmMissingCohorts, 127
- scanfiUrlRemap, 128
- scanfiUrlRemap(), 73
- scheduleDisturbance, 128
- speciesEcoregionLatestYear, 129
- speciesEcoregionStack, 130
- speciesInStudyArea, 130
- speciesInStudyArea(), 73, 141
- speciesPresentFromKNN, 132
- speciesPresentFromNTEMS, 133
- speciesPresentFromSCANFI, 134
- speciesTable (getSpeciesTable), 63
- speciesTableUpdate, 135
- speciesTableUpdate(), 141
- sppColors, 138
- sppColors(), 140, 141
- sppColours (sppColors), 138
- sppEquiv (sppEquivalencies_CA), 139
- sppEquivalencies_CA, 21, 35, 36, 49, 64, 73, 74, 76–79, 86, 91–93, 104, 108, 124, 137, 138, 139, 143, 154, 156, 157
- sppEquivCheck, 142
- sppEquivCheck(), 141
- sppHarmonize, 143
- sppHarmonize(), 140, 141
- standAgeMapGenerator, 144
- statsModel, 145
- studyAreaEco, 145
- subsetDataSize, 146
- subsetDataSize(), 73
- subsetDT, 147
- subsetDT(), 73, 146, 147
- sumRastersBySpecies, 148
- terra::coltab, 37
- terra::compareGeom(), 7
- terra::distance(), 40
- terra::extract(), 62
- terra::intersect, 115
- terra::terraOptions(), 31
- uniqueCohortDefinition, 148
- uniqueDefinitions
 - (uniqueCohortDefinition), 148
- uniqueSpeciesEcoregionDefinition
 - (uniqueCohortDefinition), 148
- updateCohortData, 149
- updateCohortData(), 73
- updateCohortDataPostHarvest, 151
- updateSpeciesTable, 153
- vegetation-transitions
 - (vtm2conifdecid), 156
- vegetationTransitions (vtm2conifdecid), 156
- vegTransitions (vtm2conifdecid), 156
- vegTypeGenerator, 153
- vegTypeGenerator(), 21, 36, 72, 73, 75, 143
- vegTypeMapGenerator, 91, 155
- vegTypeMapGenerator(), 72, 73, 75, 93, 141, 157
- vtm2conifdecid, 156
- Ward (WardKernel), 158
- Ward(), 70
- WardKernel, 158
- wetlandToLCC, 158
- wetlandToLCC(), 107