LBMR

Jean Marchal (jean.d.marchal@gmail.com) & Ceres Barros (cbarros@mail.ubc.ca)

06 March 2019

Overview

A biomass succession model derived and modified from LANDIS-II Biomass Succession v3.2.1. LBMR simulates forest succession in a spatially explicit manner (per pixel) by reproducing the population dynamics (growth and mortality), dispersal and competition for light resources of tree species. Like in LANDIS-II, dynamics are simulated in terms of their biomass per cohort (a combination of species and age), following a `biomass ~ age` curve that is influenced by species-specific parameters such as growth, mortality, maximum biomass, and species traits suss as longevity and tolerance to shade (see LANDIS-II Biomass Succession v3.2 User Guide and Scheller and Mladenoff (2004). The biggest differences between LBMR and LANDIS-II lie on how the model is parameterised and how different processes have been compartmentalised in order for higher flexibility.

- **Parametrisation**. As of now, LBMR relies on other SpaDES-modules to "automatically" estimate parameters linked to growth and seed germination (such as the maximum biomass per species, per pixel, and the probability of seed germination - *i.e.*, species establishment probability not due to resprouting) from existing data (such as species % cover, or presence/absence, stand age and stand biomass data). Other parameters are currently taken from the LANDIS-II species traits used for version 3.2.1 (which can be found here or here]).
- **Compartmentalisation and modularity**. While in LANDIS-II phenomena like growth and mortality occur as part of the same process (or function from a coding perspective), in LBMR they occur separately and sequentially. This means that they can be more easily upgraded/changed. The same happens with post-disturbance regeneration. Unlike in LANDIS-II, post-disturbance regeneration is not part of LBMR *per se*, but rather belongs to a separate module, that needs to be loaded should the user want to simulate disturbance effect (*i.e.*, fire disturbances). Again, this enables higher flexibility when swapping between different approaches to regeneration.

General flow of LBMR processes

1. Preparation of necessary objects for the simulation - either by accessory prep. data modules, of LBMR itself (using LANDIS-II test parameters)
2. Seed dispersal - see Section 4.5.1 Seeding of the LANDIS-II Model v7.0 Description(https://drive.google.com/file/d/15gSueug-Rj9I2RZqdroDbad-k53Jq7j3/view) for details
3. Growth, ageing and mortality - based on Scheller and Mladenoff (2004)
4. Preparation of visual/saved outputs ... (repeat 2-4) ...

Note that should a post-disturbance regeneration module be used, regeneration will occur after the disturbance, but *before* dispersal and background vegetation growth and mortality. Hence, the disturbance should take place either at the very beginning or at the very end of each simulation time step:

General flow of LBMR processes when disturbances are included (by linking other modules)

1. Preparation of necessary objects for the simulation - either by accessory prep. data modules, of LBMR itself (using LANDIS-II test parameters)
2. Disturbance

3. Post-disturbance regeneration
4. Seed dispersal - see Section 4.5.1 Seeding of the LANDIS-II Model v7.0 Description(https://drive.google.com/file/d/15gSueug-Rj9I2RZqdroDbad-k53Jq7j3/view) for details
5. Growth, ageing and mortality - based on Scheller and Mladenoff (2004)
6. Preparation of visual/saved outputs ... (repeat 2-6) ...

Load SpaDES

```
library(SpaDES)
```

```
## loading reproducible     0.2.7
## loading quickPlot        0.1.6
## loading SpaDES.core      0.2.5
## loading SpaDES.tools     0.3.1.9000
## loading SpaDES.addins    0.1.2
```

```
## Default paths for SpaDES directories set to:
##   cachePath:
##   inputPath:  /tmp/RtmprxgLOk/SpaDES/inputs
##   modulePath: /tmp/RtmprxgLOk/SpaDES/modules
##   outputPath: /tmp/RtmprxgLOk/SpaDES/outputs
## These can be changed using 'setPaths()'. See '?setPaths'.
```

```
moduleName <- "LBMR"
spadesModulesDirectory <- "../" # where the module will be downloaded
```

Get the module

You can either get the module using `downloadModule`, which will download the code, but will disconnect that code from the original GitHub sources, or you can fork, clone and load the repository. Doing this forking will keep the source code connected with the original. The most powerful way is to make a new, empty project of your own name in github, Make a subfolder called "modules" (same as the above name), then add this module as a submodule inside "modules". This module then becomes one of many inside your project.

```
# Simple way to download module
downloadModule(moduleName, path = spadesModulesDirectory)

# OR Advanced for developers is to use GitHub
```

Usage

Module parameters

Name	Default	Description
`growthInitialTime`	`0`	initial time for the growth event to occur.
`.plotInitialTime`	`0`	describes the simulation time at which the first plot event should occur.
`.saveInitialTime`	`0`	describes the simulation time at which the first save event should occur.

Name	Default	Description
spinupMortalityfraction	0.001	defines the mortality loss fraction in spin up-stage simulation.
successionTimestep	10	defines the simulation time step.
cellSize	NA	defines the cell size.
seedingAlgorithm	"wardDispersal"	chooses which seeding algorithm will be used among "noDispersal", "universalDispersal", and "wardDispersal".
useCache	TRUE	use caching for the spinup simulation?
useParallel	TRUE	determines whether the parallel computation will be used in the simulation.

Usage example

```
# inputDir <- file.path(dirname(spadesModulesDirectory), "inputs") %>% checkPath(create = TRUE)
inputDir <- file.path("modules/LBMR/inputs")

# outputDir <- file.path(dirname(spadesModulesDirectory), "outputs")
outputDir <- file.path("modules/LBMR/outputs")

times <- list(start = 0, end = 10)
parameters <- list(
  .globals = list(verbose = FALSE),
  LBMR = list(.plotInitialTime = 0,
              successionTimestep = 5,
              seedingAlgorithm = "wardDispersal")
  #.progress = list(type = "text", interval = 1), # for a progress bar
  ## If there are further modules, each can have its own set of parameters:
  #module1 = list(param1 = value1, param2 = value2),
  #module2 = list(param1 = value1, param2 = value2)
)
modules <- list(moduleName)
objects <- list()

#modulePath changed so I can edit in LBMR project
#spadesModulesDirectory <- dirname(getwd())

setPaths(cachePath = file.path(outputDir, "cache"),
         modulePath = spadesModulesDirectory,
         inputPath = inputDir,
         outputPath = outputDir)
paths <- getPaths()

mySim <- simInit(times = times, params = parameters, modules = modules, objects = objects, paths = paths

dev()
mySimOut <- spades(mySim, debug = TRUE)
```

Events

Events are scheduled as follows:

- Module initiation and spin-up
- Account for fire disturbance if present
- Seed dispersal
- Mortality and growth
- Reclassification of age cohorts
- SummaryRegen
- SummaryBGM
- Plot
- Save

Links to other modules

Intended to be used with other landscape modules, such as LandMine, FireSense, Boreal_LBMRDataPrep, and possibly many others.