Biomass_borealDataPrep

Yong Luo, Eliot McIntire, Ceres Barros

24 September 2019

Overview

This module converts open datasets that are available for all of Canada's forests, into the input requirements for LandR Biomass, a forest landscape succession model derived from the Landis-II Biomass Succession Model. This has been partially tested for some parts of the Western Boreal Forest.

Specifically, it takes the stand biomass, stand age, ecozone, ecoprovince, and ecoregion maps of Canada, as well as species specific biomass maps of Canada (defaulting to the kNN biomass maps) and derives Land-R Biomass parameter estimates from these data. Species traits are taken from those used by Dominic Cyr for LANDIS-II simulations.

Keeping data preparation outside of the LandR Biomass_core module maintains the modularity of the LandR modules.

Functioning

The module defaults to using kNN species cover maps and a study area in Alberta. After downloading all the necessary data (see above), the module proceeds to prepare the necessary objects and parameters for the simulation.

These are: * `cohortData` - a `data.table` contaning cohort-level information per pixelGroup (a group of pixels with identical species composition, ecoregion, age structure and biomass) * `pixelGroupMap` - a map of the study area with pixelGroup IDs * `speciesEcoregion` - a table of species parameters by ecoregion * `sufficientLight` - a table of light requirements corresponding to each shade tolerance trait value 1:5) and each shading amount in a pixel (1:5).
* `ecoregion` - a table with ecoregions coded as "active" or "inactive * `biomassMap` - a map stand biomasses used to initialise the simulation * `ecoregionMap` - a map of ecoregion IDs - note that ecoregions are a combination of land cover class and ecodistrict. * `minRelativeB` - a parameter defining the cut points to classify stand shadeness

Depending on the objects, some are parametrised using empirical models, others using expert knowledge (`sufficientLight`)

NOTE: all raster *inputs* should be at the scale of `rasterToMatchLarge` and all raster *outputs* will be at the scale of `rasterToMatch`.

Filling data gaps

- mismatches between stand age, stand biomass and species cover are dealt with by trusting species cover first. If `cover > 0` and `age == 0`, `age` is empirically estimated using a linear mixed effect model:

  ```
  age ~ B * speciesCode + (1 | initialEcoregionCode) + cover
  ```

  If `cover` and `age == 0` and `biomass > 0`, `biomass` is set to `0`.

- species `longevity` parameters are taken from published numbers from closest regions and adjusted using expert opinion.
- the user can choose to replace certain land cover classes with others. For example, in a "Natural Range of Variation" project, urban pixels can be changed to forested. Since the currently available simulation modules do not drive land cover change, species-ecoregion traits will be spatially non-varying. This means that initial land cover classes that are "transient", such as "recent burn" are likely not appropriate to have "permanent" traits associated with them. This module offers the ability to estimate the parameters for these pixels *as if they were a neighbouring land cover class*. In these two cases, the module can replace these classes with a neighbouring forest type (probabilistically chosen in proportion to its neighbourhood abundance).

Parametrisation

LandR-Biomass, like LANDIS-II Biomass Succession, has species-level traits and species-ecoregion level traits. The difference between these is the the species-ecoregion traits can vary spatially. During the steps here, we estimate these traits by *species* X *land cover class* X *ecodistrict* (Canadian, federal definition). This can mean that there are some species-ecoregion traits that are unique at the pixel level as there may be rare combinations of, say, a land cover type, for a given species, in a given ecodistrict.

Species ecoregion parameters

- establishment probabilites: species establishment probabilities (or sometimes `SEP` in LANDIS) by ecoregion are empirically estimated using species cover data (converted to presence/absence) and a GLMEM defined as: `prob. presence ~ species + (1|ecoregion)`
- `maxB` is estimated empirically, using stand age, stand biomass per species, per ecodistrict data and a LMEM defined as: `B ~ logAge * speciesCode + cover * speciesCode + (logAge + cover + speciesCode | ecoregionGroup)` Because this equation represents a curve whose positive slope is decreasing with age (via `log(age)`), we estimate `maxB` as the expected `B` when `cover = 100` and `logAge = log(longevity)` for a given species in a given `ecoregionGroup` (which is combination of `land cover class` X `ecodistrict`. This equation is a parameter in the module so can be changed by the user.
- `maxANPP`: is defined as maxB/30 following LANDIS-II. But see below for updates.

Species parameters

- `growthcurve` and `mortalityshape` are, at the moment, taken from LANDIS-II parameters. This is changing soon (see "in development" section below)

All empirically estimated parameters *can* be estimated using data from a larger study area (`studyAreaLarge`) than the one used to run the simulation (`studyArea`), if the user provides such a polygon.

In development

updating species parameters

1. We run ~41,000,000 hypothetical species with full factorial combinations of longevity, ratio of `maxANPP` to `maxBiomass`, `growthcurve`, `mortalityshape`
2. We take the closest permanent and temporary sample plots in or near the study area and find the hypothetical species in previous step that most closely matches the growth dynamics in the PSPs. This gives us the growthcurve, mortalityshape, and ratio of maximum biomass to maximum ANPP for each species in our study area
3. We introduce a new parameter, `actualMaxBiomass`. We recognize that `maxB`, as obtained empirically above, cannot be easily reached in simulations because all reasonable values of `growthcurve`, `mortalityshape` and `longevity` prevent the equation from reaching `maxB` (it acts as an asymptote that

is never approached). The `actualMaxBiomass` is then obtained by multiplying the empirically estiamted `maxB` by the ratio between the maxBiomass parameter used for the simulations in step 1 (just above) and the maximum simulated biomass actually achieved in the simulations (of point 1). We use this `actualMaxBiomass` so that the resulting non-linear growth curves will hit the the empirically estimated `maxB`. This adjustment effectively allows the growth equation's "maximum biomass" parameter to actually be the empirically estimated maximum biomass.

4. Species-specific `maxANPP` is estimated by multiplying the empirically estimated maxB (spatial) above and the ratio of the simulated maxANPP parameter (point 1) to the maximum simulated biomass (point 1) at the species level.

Get the module

See SpaDES-modules repository to see how to download this and other SpaDES modules. Alternatively, it can be forked or cloned from github.com directly. # Load libraries

```
library(magrittr) # for %>% pipe
library(SpaDES)
```

Set up paths

```
moduleName <- "Biomass_borealDataPrep"
spadesModulesDirectory <- ".." # where the module will be located -- this is correct, if this module
                               # is an Rstudio project, i.e., one up from the project

inputPath <- file.path(dirname(spadesModulesDirectory), "inputs") %>% checkPath(create = TRUE)
outputPath <- file.path(dirname(spadesModulesDirectory), "outputs")
cachePath = file.path(outputPath, "cache")

setPaths(cachePath = cachePath,
         modulePath = spadesModulesDirectory,
         inputPath = inputPath,
         outputPath = outputPath)
paths <- getPaths()
```

Choose a study area

```
library(raster)
# modulePath <- Cache(readline, paste0("Where is the module path? (e.g., ~/module, with no quotes).\n",
#                                      "Press Enter to accept the path in getPaths()$modulePath: "),
#                     cacheRepo = cachePath)
# setPaths(cachePath = cachePath, modulePath = modulePath)

## do you want to hand-draw a map or use defaults?
# - note that large areas will take longer to compute
handDrawMap <- FALSE

if (handDrawMap) {
  dev()
  clearPlot()
  canadaMap <- Cache(getData, 'GADM', country = 'CAN', level = 1, path = Paths$inputPath,
                     cacheRepo = getPaths()$cachePath, quick = FALSE)
  Plot(canadaMap, speedup = 5, visualSqueeze = 0.9) # 5 seemed optimal

  ## hand-drawn study area
  if (!exists("studyAreaLarge")) {
    message("Since there is no object called 'studyAreaLarge', please draw a study area with 10 points")
    severalrandompoints <- Cache(clickCoordinates, 10)
    # if(startsWith(attr(severalrandompoints, "tags"), "cache")) message("Taking studyAreaLarge from Ca
    studyAreaLarge <- SpatialPolygons(list(Polygons(list(Polygon(severalrandompoints$coords)), ID = 1))
                                      proj4string = crs(canadaMap))
  }
  Plot(studyAreaLarge, addTo = "canadaMap", col = "red")
} else {
  studyArea <- randomStudyArea(seed = 1234, size = (250^2)*100)
}

times <- list(start = 0, end = 10)
modules <- list("Biomass_borealDataPrep")
objects <- if (handDrawMap) list("studyAreaLarge" = studyAreaLarge,
                                 "studyArea" = studyAreaLarge) else list()

mySim <- simInit(times = times, #params = parameters,
                 modules = modules, #, "Biomass_core"),
                 objects = objects, paths = getPaths())
```

Run spades

This module is about data preparation, so there is no stochastic elements. The `spades` call will only cause one event to occur (the `init` event)

```
simOut <- spades(mySim, debug = TRUE)
```

Visualize

The `Plot` function will visualize all known .quickPlot type objects, which includes `Raster*` and `SpatialPolygons*` objects. After running this module, these are the outputs, which would likely be used as inputs to `Biomass_core`.

```
dev()
clearPlot()

Plot(simOut)
```

Downloads

During the `simInit` call, if the user does not provide alternatives for the expected inputs, the module will download 3 large `.tar` files (~2 GB each) and 1 `.zip` file (~45 MB) from the internet.

Inputs

This module has several input requirements. One is a study area, which should be provided as a SpatialPolygonsDataFrame, and named `studyAreaLarge`. This should be inside the boundaries of the boreal forest of

Canada. When first running the code in this `.Rmd` file, you will be prompted to draw a polygon if none is provided as an input.

Creates Inputs

Most of the inputs will be created automatically, if they are not provided by the user. The automatic creation will work in the boreal forest of Canada. These are zip files and tar files that are available from various Natural Resources Canada web pages. Also, this module gets its Species Traits table from dcyr/LANDIS-II_IA__generalUseFiles.

Outputs

This will show the outputs of this module, which can be used directly as the inputs for Biomass_core:

```r
# List all objects
ls(simOut)

# Examine a few tables a visuals
simOut$speciesTable
Plot(simOut$biomassMap)
simOut$studyAreaLarge <- spTransform(simOut$studyAreaLarge, crs(simOut$biomassMap))
Plot(simOut$studyAreaLarge, addTo = "simOut$biomassMap")
```

References