MEDFIRE

User's guide

2020-03-20

Structure of the model

MEDFIRE is a **Landscape Dynamic Model** that integrates the main factors of change in Mediterranean forest landscapes. It allows exploring the spatial and temporal interactions between land-cover changes, fire regime, forest managment, fire management and vegetation dynamics.

MEDFIRE is a spatially explicit model implemented in R and structured in the following folders:
- *inputfiles* contains the text files with the current version of model's parameters.
- *inputlyrs* contains the initial state in raster and Rdata format of the model.
- *mld* contains the R-scripts files of the model, sub-modules and reporting functions.
- *output* contains one sub-folder for each scenario, and for each scenario **scn.def.r** and **scn.custom.def** describe the initialization of model's parameters.
- *rscripts* contains auxiliar R-scripts to run the model, build model's inputs and analyze model's outputs.

The current version of the model is implemented thought the following functions (in alphabetic order):
- age.by.spp.r reports the age class abundance per vegetation type.
- buffer.mig4.r determines the presence of species within a buffer around target cells. It is called from the regeneration/succession section of the model.
- define.scenario.r initializes the scenario parameters and the global variables of the model.
- disturbance.cc.r is the clear-cutting sub-module. It calculates sustained-yield levels and simulates clear cuts harvesting for each management unit.
- disturbance.fire2.r is the wildfire disturbance sub-module. It simulates fire events in four different fire regime homogeneous zones.
- fire.spread3.r simulates fire spread from ignitions points according to a spread probability.
- forest.transitions2.r assigns a new tree species according to the post-disturbance regeneration or forest succession hypotheses. It is called from the regeneration/succession section of the model.
- landscape.dyn5.r is the Quebec Landscape Dynamics Model. It loads the spatial state variables, the initialization of model's parameters, creates the scenario output sub-folder and schedules the processes, e.g. clear-cuts, wildfires, post-disturbance regeneration, aging and forest succession.
- neighbour.spp.r is a fnction that returns the forest composition of a circular neighbourhood of radius R around the target cells.
- suitabilty.r determines the environmental suitability of a subset of cells for given climatic and soil (surficial deposits) conditions. It is called from the regeneration/succession section of the model.

To run a test scenario

1. Unzip the supplementary file **QLandscapeDynamics.zip** in a local folder, e.g. *C:/WORK/QLandscapeDynamics*.
2. In R, clean the working space.

```
rm(list = ls())
```

3. Set the working directory to the previous local folder.

```
setwd("C:/WORK/QLandscapeDynamics")
```

4. Load the model.

```
source("mdl/landscape.dyn5.r")
```

5. As the scenarios are already defined and initialized (in the **scn.def.r** and **scn.custom.def** of the corresponding sub-folder), to re-run them, call the **landscape.dyn5** function with the name of the scenario as argument.

```
landscape.dyn("feuTcouTbufCpersT")
landscape.dyn("feuTcouTbufCpersF")
landscape.dyn("feuTcouTbufLpersT")
landscape.dyn("feuTcouTbufLpersF")
landscape.dyn("feuTcouFbufCpersT")
landscape.dyn("feuTcouFbufCpersF")
landscape.dyn("feuTcouFbufLpersT")
landscape.dyn("feuTcouFbufLpersF")
landscape.dyn("feuFcouTbufCpersT")
landscape.dyn("feuFcouTbufCpersF")
landscape.dyn("feuFcouTbufLpersT")
landscape.dyn("feuFcouTbufLpersF")
landscape.dyn("feuFcouFbufCpersT"))
landscape.dyn("feuFcouFbufCpersF")
landscape.dyn("feuFcouFbufLpersT")
landscape.dyn("feuFcouFbufLpersF")
landscape.dyn("barrier")
```

To run a new scenario

1. Do steps 2. to 4. and load the function to set the default scenario's parameters.

```
rm(list = ls())
setwd("C:/WORK/QLandscapeDynamics")
source("mdl/landscape.dyn5.r")
source("mdl/define.scenario.r")
```

2. Clean all data from the workspace, but not loaded functions.

```
fun <- unclass(lsf.str())
toremove <- setdiff(ls(), fun)
rm(list = c(toremove, 'toremove'))
```

3. Give a name to the new scenario, e.g. *"test"*, and call the **define.scenario** fucntion to load model's parameters with the default initialization.

```
scn.name <- "test"
define.scenario(scn.name)
```

4. Customize the initialization of some parameters

```
is.climate.change <- TRUE
time.horizon <- 90
nrun <- 3
```

5. Write the name of all the above updated parameters in the following call of the **dump** function. It copies these R objects into the file **outputs/test/scn.custom.def.r** (do not change the name of this file).

```
dump(c("is.climate.change", "time.horizon", "nrun"),
     paste0("outputs/", scn.name, "/scn.custom.def.r"))
```

6. Run ths scenario calling the **landscape.dyn** function. A sub-folder named *"test"* will be created in the folder *outputs*. Model's output files, **scn.def.r** and **scn.custom.def.r** are saved in it.

```
landscape.dyn(scn.name)
```