Working with LuccME

A user's guide

Version 3.1 | September 2017

Working with LuccME

A user's guide

Version 3.1 | September 2017

Ciência para sustentabilidade

Lead authors: LuccME Team

Contents

1 Introducing the LuccME framework

1.1 What is LuccME

LuccME (*Land Use and Cover Change Modelling Environment*) is an open source framework for spatially explicit Land Use and Cover Change (LUCC) modeling developed by the Earth System Science Center (CCST) and collaborators, built on top of the Terra-ME (Carneiro et al., 2013), as an extension. Using LuccME the modeler can easily create deforestation, agricultural expansion, desertification, forest degradation, urban sprawl spatial models and other land use change models at different scales and areas of study, combining existing model components and/or creating new ones. These land-use models basically quantify over time and space the relationships between different driving factors and land use patters (Verburg et al., 2004).

1.2 The philosophy of LuccME

There are many different types of LUCC models which can be classified according to model goal, scale, technical approach or underlying theory. In spite this diversity of modeling approaches, a common structure can be identified in several spatially explicit models (Verburg et al., 2006; Eastman et al, 2005), as Figure 1 illustrates. In general, three main components can be defined: i) *Demand* – responsible by the calculation of the magnitude or quantity of change; ii) *Potential* – responsible by the calculation of the suitability or propensity of change for each cell; iii) *Allocation* – responsible by the spatial distribution of changes based on land demand and potential of change of each cell. These components are in general organized in a top-down manner, in which a demand for land is spatially allocated according to cell suitability. Several well-known LUCC models follow this structure, including CLUE family (Veldkamp and Fresco, 1996; Verbug et al., 1999; Verburg et al., 2002), Dinamica EGO (Soares-Filho et al. 2002), and GEOMOD (Pontius et al., 2001), using a range of different approaches and techniques for the three components (Eastman et al., 2009; Lesschen et al., 2007). However, these models are implemented in different computing platforms, their code is in general not open, and they cannot be easily modified or combined.

In this sense, LuccME allows the construction of new models, combining elements of *Demand*, *Potential* and *Allocation* components, designed according to the original idea of the main models found in the literature.

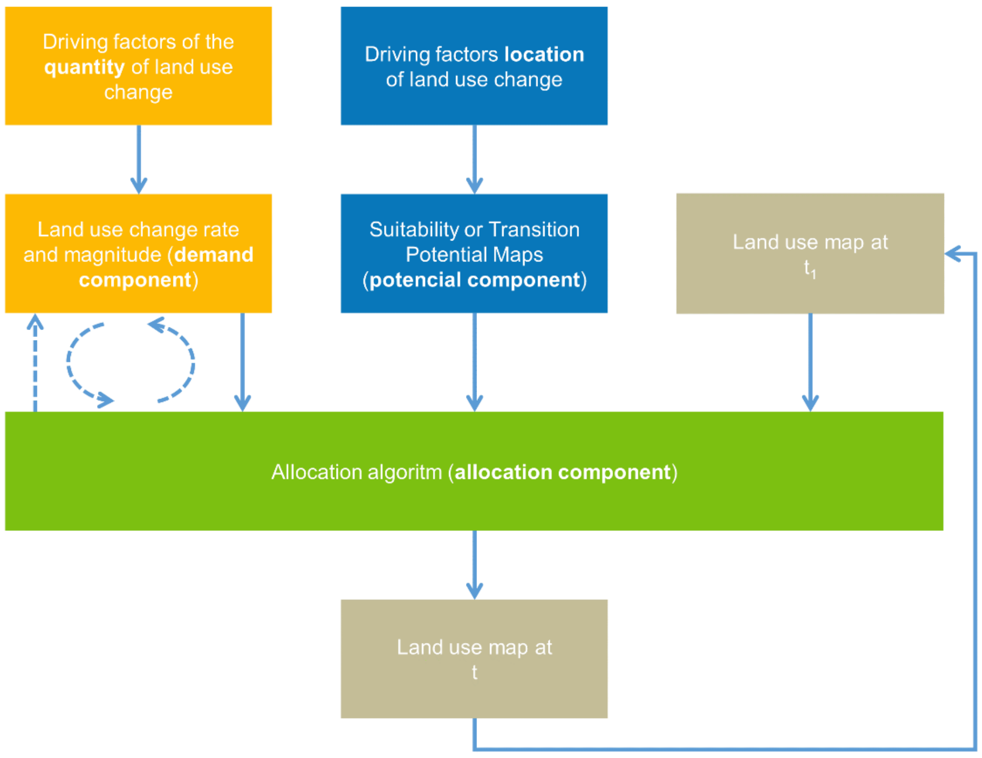

Driving factors of the **quantity** of land use change
Driving factors **location** of land use change
Land use change rate and magnitude (**demand component**)
Suitability or Transition Potential Maps (**potencial component**)
Land use map at t_1
Allocation algoritm (**allocation component**)
Land use map at t

Figure 1 - Generic structure of the main spatially explicitly LUCC Models available in the literature (adapted from Verburg et al. 2006).

An important feature of LuccME is its modular set-up. This allows tailoring of the model structure to the user needs. In LuccME, the components can be chosen and modified according to the needs of a given application and scale of analysis, and easily parameterized through a simple user interface. LuccME is built on top of TerraME, a general programming environment for spatial dynamical modeling, designed to support models in several domains, including hydrology, biodiversity, land cover change, and many others. Its modeling language has in-built functions that make it easier to develop multi-scale and multi-paradigm models for environmental applications. TerraME provides an interface to TerraLib geographical library (Camara et al., 2008), allowing models direct access to geospatial data. LuccME, TerraME and Terralib are technological products developed by the Brazil's National Institute for Space Research (INPE) and its partners.

Besides providing an easy way to use and create new LUCC models, LuccME was designed in order to support the development of LUCC models which can be integrated to more complex multi-scale environmental models. In this sense, models created using LuccME are encapsulated in Environments, a core concept in Terra-ME, as presented in Carneiro (2006), so later they may be easily coupled to other models, for instance Earth System models and/or multi-scale LUCC models.

1.3 For whom is this manual?

This manual provides an explanation of the components and a user guide to build and run a model with LuccME framework. However, using LuccME requires, at a minimum, an understanding of how land use models work in general (Demand, Potential and Allocation) and the ability to manipulate a spatial database. Therefore, in the following LuccME documentation, neither land use modeling nor LUA programming language or TerraView GIS are discussed.

This manual describes how to get started with LuccME framework by creating a model from scratch. Some examples of models and database are also provided along with the LuccME package, as described in Chapter 5. This guide also describes the main steps for running a model built with LuccME and how to combine different model components.

2 Getting started with LuccME

This chapter describes how to get started with LuccME 3.1.

2.1 Installing LuccME

There are two ways to interact with LuccME, via Graphic Interface or text editor with command prompt.

1. LuccME with graphic interface (recommended):
   The only need is to execute the installer file (64 bits).

2. LuccME without graphic interface:
   To use LuccME without graphic interface there are two options:

   Using a pre-configured IDE:

   1. Download ZeroBrane Studio IDE configured to run LuccME models.
   2. Unzip the files in C:.
   3. Open the ZeroBrane IDE and select LuccME option on *Project* > *Lua Interpreter* menu.
   4. To execute the model clik on the green arrow icon (F5).

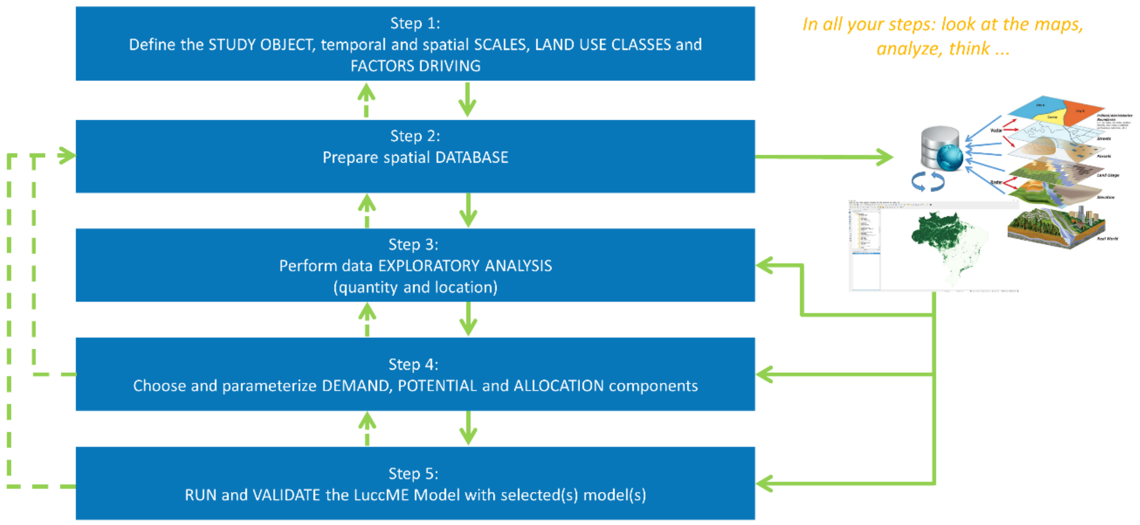
   Using the text editor of your choice:

   1. Download LuccME (source code).
   2. Install TerraME 2.0.
   3. Unzip the luccme.zip file into “luccme" folder, inside "bin/packages" folder on TerraME.
      All files are available in http://luccme.ccst.inpe.br.

3 Building a model with LuccME

3.1 Introduction

The construction of a land use model in LuccME can be split over 5 steps as illustrated in Figure 2. The first step is to define the purpose of the application such as agricultural expansion, urban expansion, desertification, deforestation. In this stage, the spatial and temporal scale of work is usually defined, as well as, the land use types and the determinants of change for the intended land use model.



Step 1:
Define the STUDY OBJECT, temporal and spatial SCALES, LAND USE CLASSES and FACTORS DRIVING
In all your steps: look at the maps, analyze, think ...
Step 2:
Prepare spatial DATABASE
Step 3:
Perform data EXPLORATORY ANALYSIS (quantity and location)
Step 4:
Choose and parameterize DEMAND, POTENTIAL and ALLOCATION components
Step 5:
RUN and VALIDATE the LuccME Model with selected(s) model(s)

Figure 2 - Steps for building a land use model in LuccME 3.1.

The second step comprises two activities: i) preparing the spatial database and ii) filling the cell space. Cell space (Figure 3) is a generalized matrix structure where each cell is associated with various types of attributes, such as land use/land cover classes (e.g. forest, agriculture, pasture, etc.) or driving factors (e.g. roads, conservation units, agrarian structure, etc.). The objective is to homogenize information from different sources, in different formats (vector, matrix and other cell spaces), by aggregating them in the same time- space basis. The cell space can be built directly into TerraME 2.0 (as described below) or from Geographic Information Systems (GIS) of the user's preference (e.g. TerraView, ArcGIS or QGIS). Table 1 presents an overview of the main operators available to fill cells via TerraME.

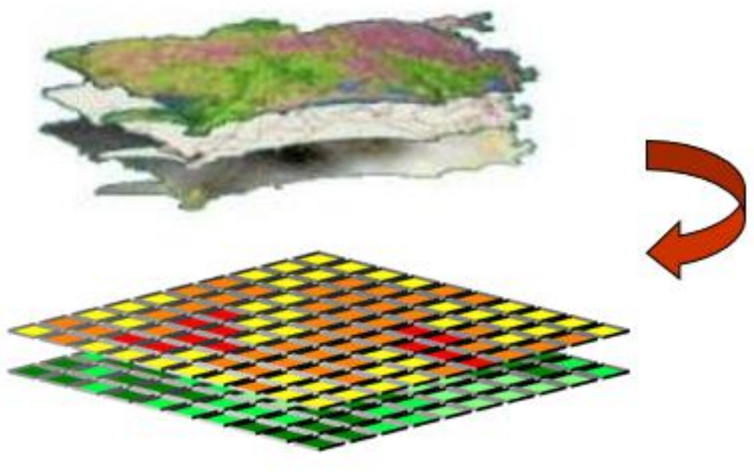


Figure 3 - Cell space - process of homogenization of different databases in cell space.

The FillCell script manager is a tool offered to the user free of charge and is intended to assist in the construction and filling of a cell space compatible with LuccME 3.1. Through an intuitive interface, the user provides the information inherent to their application without having to worry about programming syntax. However, this tool does not limit the user who wants to build their scripts directly on TerraME, through the editor of their choice. Below is the main window of FillCell, where you can access the features of this manager, such as:

- Create a New Script.
- Opening a model generated by this tool.
- Select the display language (Portuguese or English).

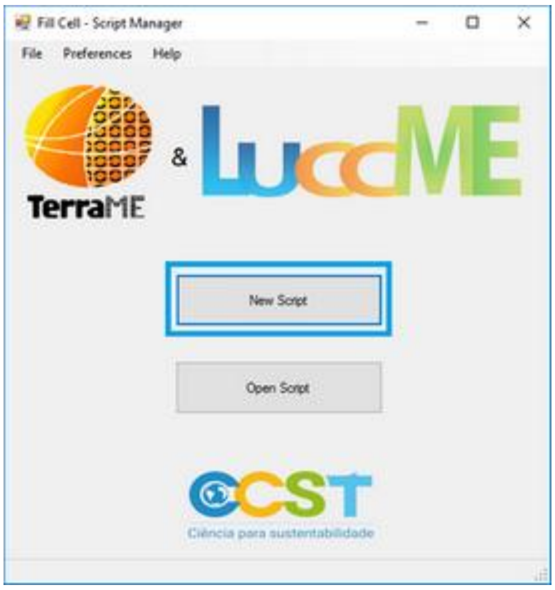


Figure 4. FillCell Main Window.

Another way to access the form for creating a new script is through the main menu, as shown in Figure 5.

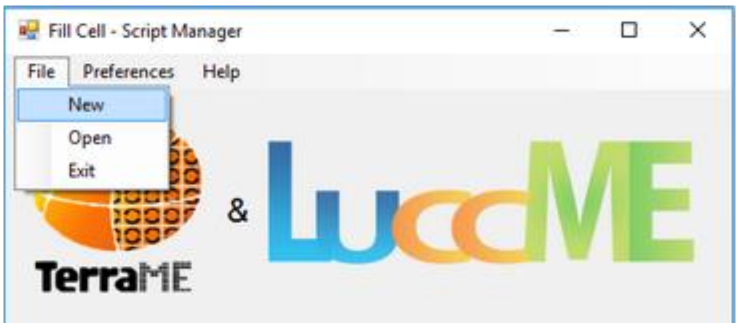


Figure 5 - Main Menu – File -> New.

Once selected, the script creation window is opened, as shown in Figure 6. In this step the user must first indicate the directory where the script will be saved, using the "folder to save the scripts" option. After selecting the folder where the file will be saved, the selected address will be displayed to the user. Second, the user must indicate an output name for the script that will be created using the "script name" option.

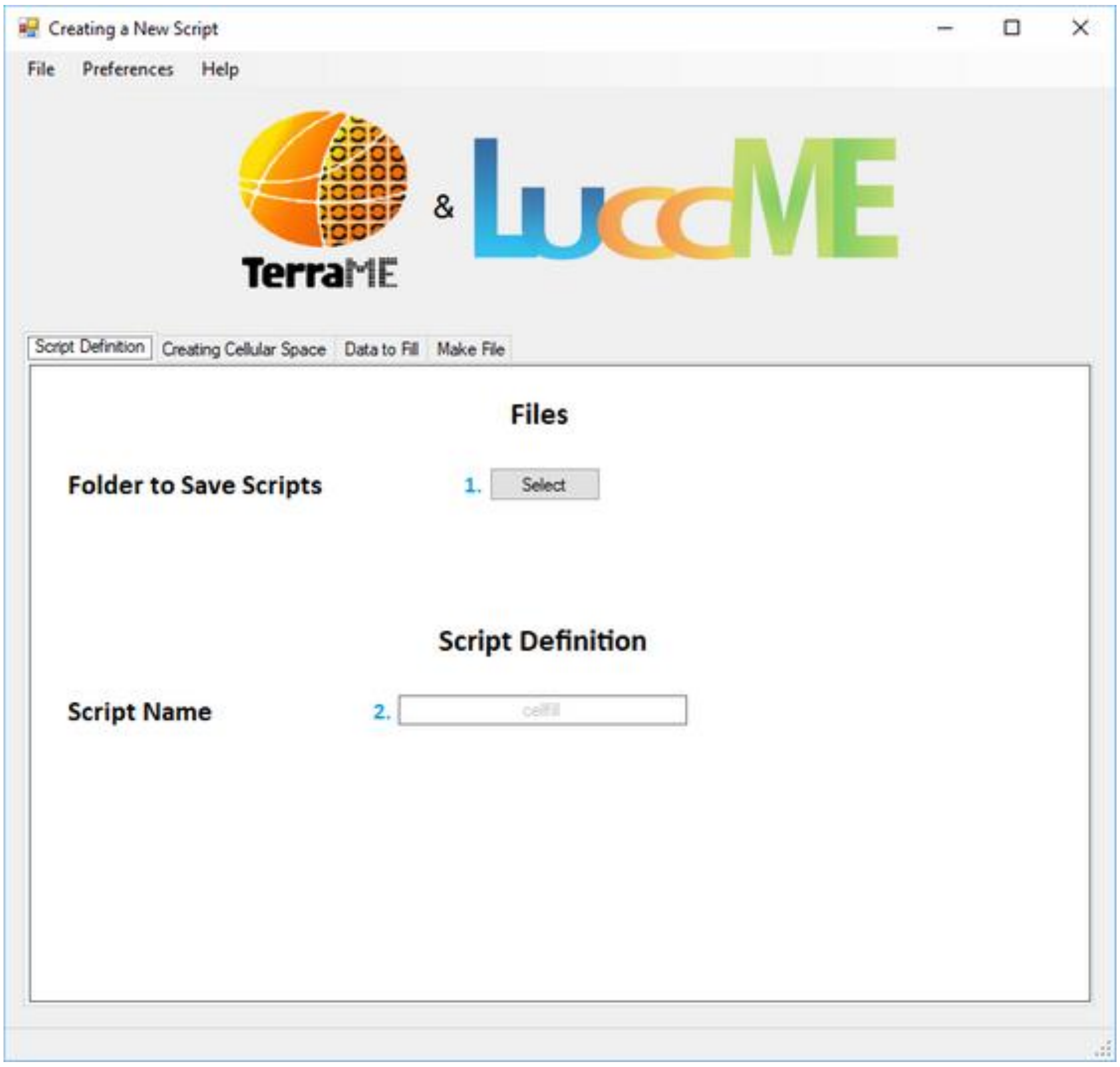


Figure 6 - Script Definition.

The next step is to define the parameters for creating the cell space, as shown in Figure 7. At this moment the user is asked to indicate:

1. File containing the cellular space limit (eg study area boundary). This file must be a polygon type vector in the shapefile format;
2. Name of the cellular space to be created;
3. Spatial resolution of cells (side x side);

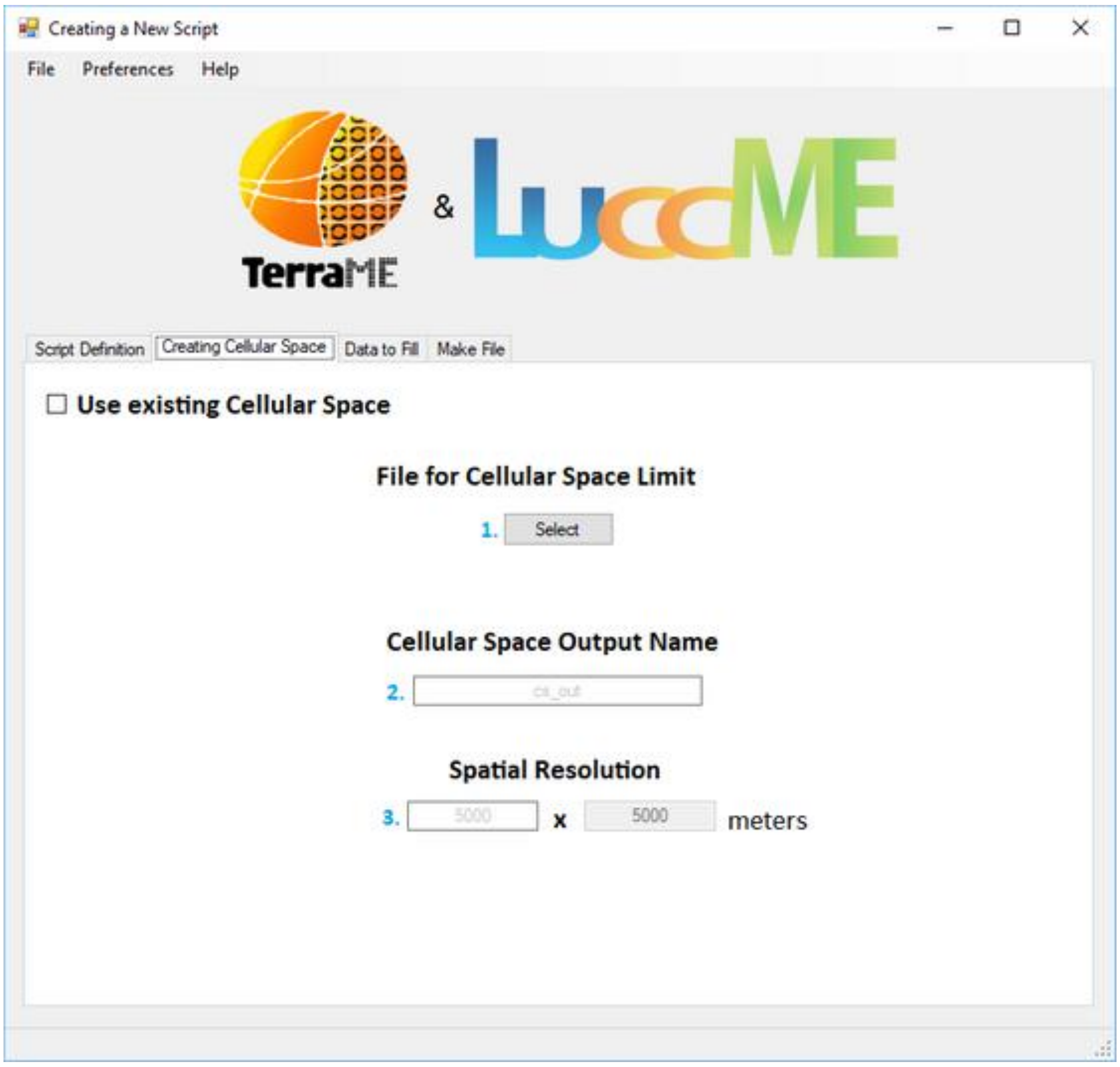


Figure 7 - Creating a cellular space.

Once the cellular space parameters have been defined, the user can start filling them in through the "fill attributes" menu. In this section the user must indicate what data will be used to fill the cells (eg land uses), as shown in figure 8. To add the files containing the data, the user must click the "+" button located in the lower corner left of the window. After clicking this button, the file selection window will appear. The file formats currently supported in the interface are shapefile for vector data (lines, points, and polygons) and GeoTIFF for matrix data.

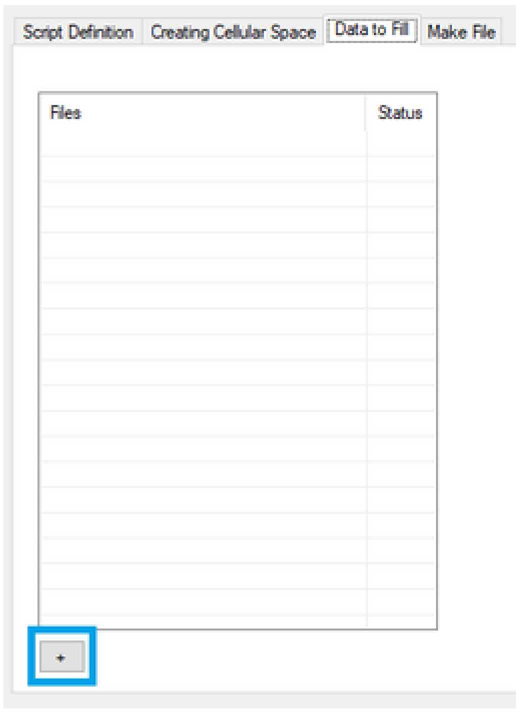
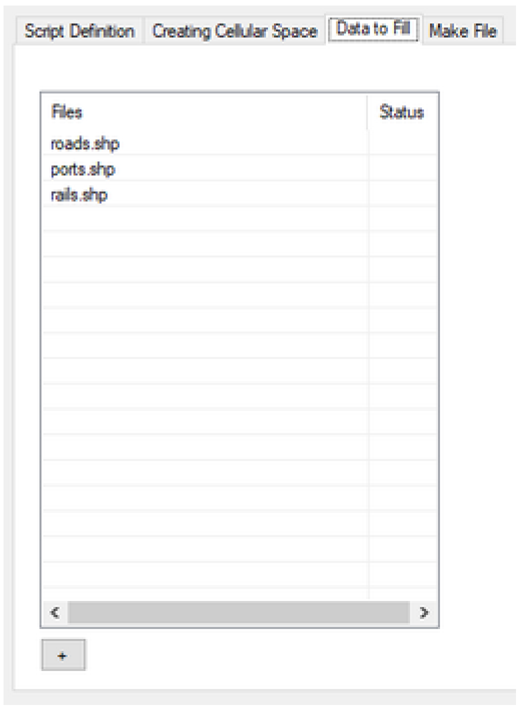


Figure 8 - Adding attributes to fill.

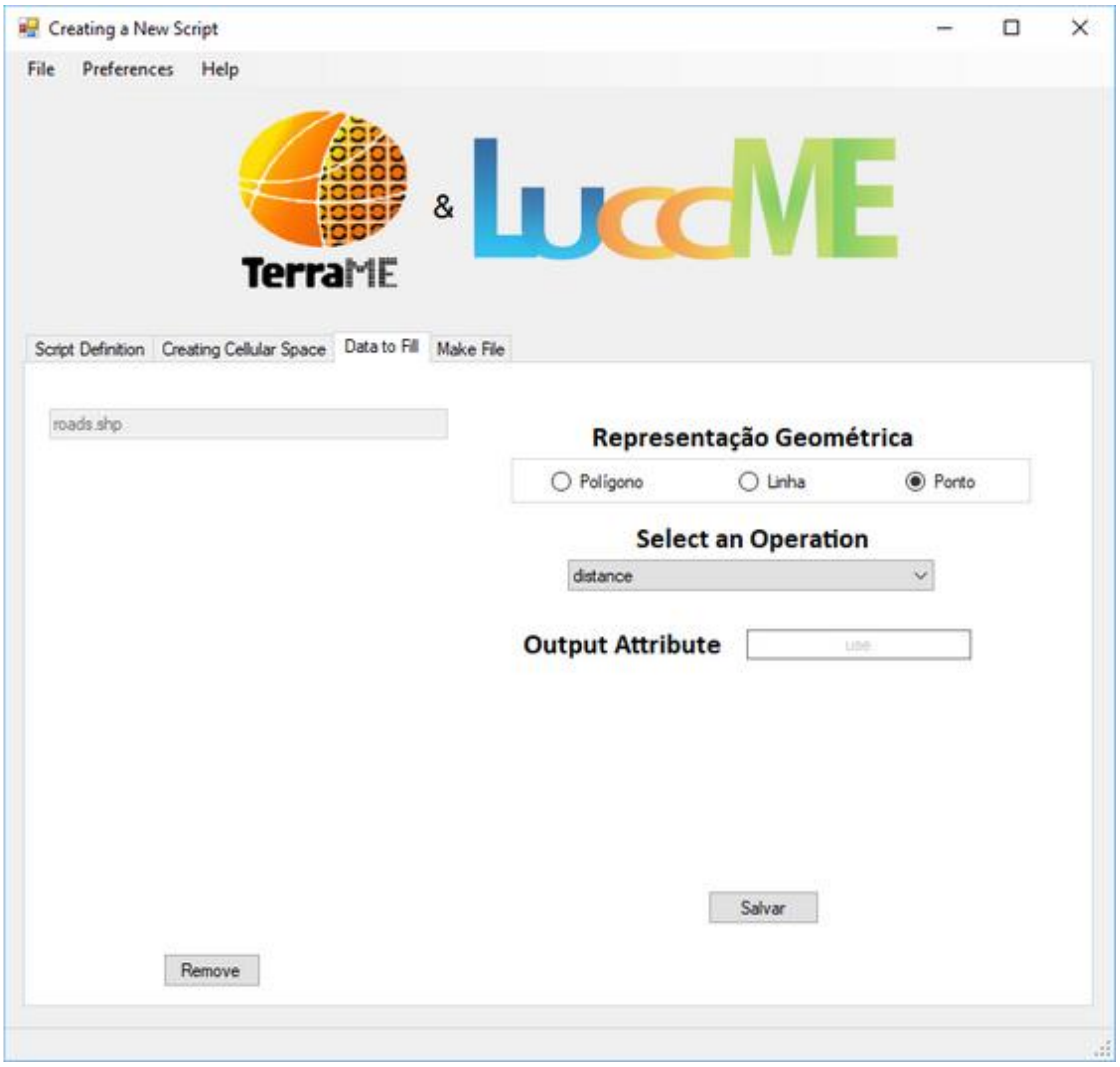
When the selection of the data to be filled is complete, the file names (.tif and .shp) will appear in the attributes column, as shown in Figure 9.



roads.shp
ports.shp
rails.shp

Figure 9 - Added attributes to fill.

Once a data is added to the attribute list, the user can select the desired fill operation so that this variable is available in the cellular space. To configure a fill operation, simply click on the name of the file in the list, then the window will enter the configuration mode, as shown in Figure 10.



roads.shp

Representação Geométrica: Polígono Linha [✓ Ponto]

Select an Operation: distance

Output Attribute: use

Figure 10 - Selecting an operation to fill an attribute.

The list of operators available for completion is generated according to the format of the files used (.shp or .tif). When the desired fill operation is selected, the operation setting parameters will appear.

Table 1 presents an overview of the main operators available to fill cells via TerraME.

The script example shown below can be used to create and fill a cellular space directly in TerraME (generated with the FillCell interface). Each type of land use as well, the different determining factors will be represented by a column in the cell space.

```
----------------------------------------------------------------
--         This file contains a Fill Cell Script           --
--            Compatible with LuccME 3.1                   --
--      Generated with Fill Cell Script Configurator  --
--            19/06/2017 at 09:24:50                        --
----------------------------------------------------------------

local x = os.clock()
import("terralib")

local projFile = File("t3mp.tview")
if(projFile:exists()) then
		projFile:delete()
end

-- CREATING PROJECT --
print("-- Creating Project --\n")

proj = Project {
		file = "t3mp.tview",
		clean = true
}

-- ADDING LAYERS --
print("-- Adding Layers to the Project --")

l1 = Layer{
		project = proj,
		name = "limit",
		file = "C:\\Treinamento\\preenchimentoDeCelulas\\limitePA_src29101_pol.shp"
}
print("Added Limit: limitePA_src29101_pol.shp")

l2 = Layer{
		project = proj,
		name = "layer2",
		file = "C:\\Treinamento\\preenchimentoDeCelulas\\rodoviasPA_src29101_lin.shp"
}
print("Added Layer2: rodoviasPA_src29101_lin.shp")

-- Checking EPSGs --
print("\n-- Checking EPSGs--")
local epsgVector = {l1.epsg, l2.epsg}
local fileVector = {"limitePA_src29101_pol.shp", "rodoviasPA_src29101_lin.shp"}
local checkEPSG = true

for i = 1, #epsgVector, 1 do
		if (epsgVector[i] ~= l1.epsg) then
				print("Error: EPSG does not math - limit : "..l1.epsg.." "..fileVector[i].." : "..epsgVector[i])
				checkEPSG = false
		end

		if checkEPSG then print("EPSG - limit : "..l1.epsg.."\t"..fileVector[i]..": "..epsgVector[i]) end
end

if not checkEPSG then os.exit() end

-- CREATING CELLULAR SPACE --
print("\n-- Creating Cellular Space --\n")

local cs = Layer{
		project = proj,
		source = "shp",
		name = "cs",
		input = l1.name,
		resolution = 50000,
		file = "csExemplo.shp",
		clean = true
}

-- FILLING CELLULAR SPACE --
print("Filling rodoviasPA_src29101_lin.shp into Cellular Space using distance operation")
cs:fill{
		layer = "layer2",
		operation = "distance",
		attribute = "rod",
		dataType = "line"
}

projFile = File("t3mp.tview")
if(projFile:exists()) then
		projFile:delete()
end
print(string.format("\nElapsed time : %.2fs\n", os.clock() - x))
print("\nEnd of Script")
```

Table 1 - Synthesis of the main operators and functions available to fill cells via TerraME.

"area"	Total overlay area between the cell and a layer of polygons. The created values will range from zero to one, indicating its area of coverage.
"average"	Average of quantitative values from the objects that have some intersection with the cell, without taking into account their geometric properties. When using argument area, it computes the average weighted by the proportions of the respective intersection areas. Useful to distribute attributes that represent averages, such as per capita income.
"count"	Number of objects that have some overlay with the cell.
"distance"	Distance to the nearest object. The distance is computed from the centroid of the cell to the closest point, line, or border of a polygon.
"mode"	More common qualitative value from the objects that have some intersection with the cell, without taking into account their geometric properties. This operation converts the output to string. Whenever there are two or more values with the same count, the resulting value will contain all them separated by comma. When using argument area, it uses the value of the object that has larger coverage.
"maximum"	Maximum quantitative value among the objects that have some intersection with the cell, without taking into account their geometric properties.
"minimum"	Minimum quantitative value among the objects that have some intersection with the cell, without taking into account their geometric properties.
"coverage"	Percentage of each qualitative value covering the cell, using polygons or raster data. It creates one new attribute for each available value, in the form attribute.."_"..value, where attribute is the value passed as argument to fill and value represent the different values in the input data. The sum of the created attribute values for a given cell will range from zero to one hundred, according to the coverage of the cell. When using shapefiles, keep in mind the total limit of ten characters, as it removes the characters after the tenth in the name. This function will stop with an error if two attribute names in the output are the same.
"presence"	Boolean value pointing out whether some object has an overlay with the cell.
"stdev"	Standard deviation of quantitative values from objects that have some intersection with the cell, without taking into account their geometric properties.
"sum"	Sum of quantitative values from objects that have some intersection with the cell, without taking into account their geometric properties. When using argument area, it computes the sum based on the proportions of intersection area. Useful to preserve the total sum in both layers, such as population size.

Source: Adapted from Andrade e Avancini, 2016 (https://github.com/TerraME/terrame/wiki/Fill).

For continuous models, each cell (line) must be filled with the percentage occupied by each type of land use. The sum of all land use/land cover classes should total 1 in all cells. The total area occupied by each land use class must be equal to the value reported in the first time-step in the demand component. For discrete models, each cell must be filled with the type of majority land use where 1 should indicate presence and 0 should indicate the absence of each type of land use in the cell, as shown in Figure 4.

The names used to identify the attributes (columns with classes of land use/cover and driving factors) in the cell space fill should also be used to parameterize the model developed from the LuccME platform. If the user has chosen to use some transformation of the explanatory variables (e.g. logarithmic, radiciation, etc.) during the statistical analyzes, the same should be done with the respective columns of the cell space, if they are not yet transformed. Dependent variables (land use/land cover types) can only be transformed with Log10, which can be indicated in the LuccME. For the other variables (independent variables) any transformation can be used.

The third step is basically to support the parameterization of the Potential component, but also the components of Demand and Allocation based on the analysis of the rhythm and agglutination of changes in time and space. This stage is intended for visual analysis of land use data (extent and difference), analysis of the amount of change and visual analysis of the driving factors (qualitative relation with land uses). The third step also included the exploratory analysis of the data: correlations, histograms and transformations necessary to establish potential relationships from different methods (linear, logistic or spatial regression). These analyzes are done outside of LuccME, usually in software such as R, GeoDA, SPSS and others.

Based on the application and analysis of the different classes of land use/cover and driving factors, the user can start the fourth step, which consists of the parameterization of the LuccME model. First, the user must select the most appropriate Potential, Demand and Allocation components for the application. If necessary, the user can still implement new components. The LuccME components (described in the following sections) follow a structural approach (Verburg et al., 2004), where spatial determinants are related to different land use patterns, based on a single map. That is, the land use models constructed in LuccME did not work with observed transitions between classes of land use (e.g. forest to pasture). In other words, the land use variable is not the change that occurred between two classes of land use in a given period, but the accumulated historical pattern. In addition, the LuccME components are organized into two groups (directories) that cannot be combined: i) Continuous and ii) Discrete. In practice, Continuous algorithms should be used when the land use variables are provided as a percentage of each cell in the cell space (Figure 4). On the other hand, discrete algorithms should be used when land use types are provided as categorical variables, representing the presence of each class in a cell. In each of these groups (Continuous and Discrete), all components can be used interchangeably. New components implemented by LuccME users must respect this rule. For more information on creating new components, please check the LuccME Creating Components Guide available on the LuccME website.

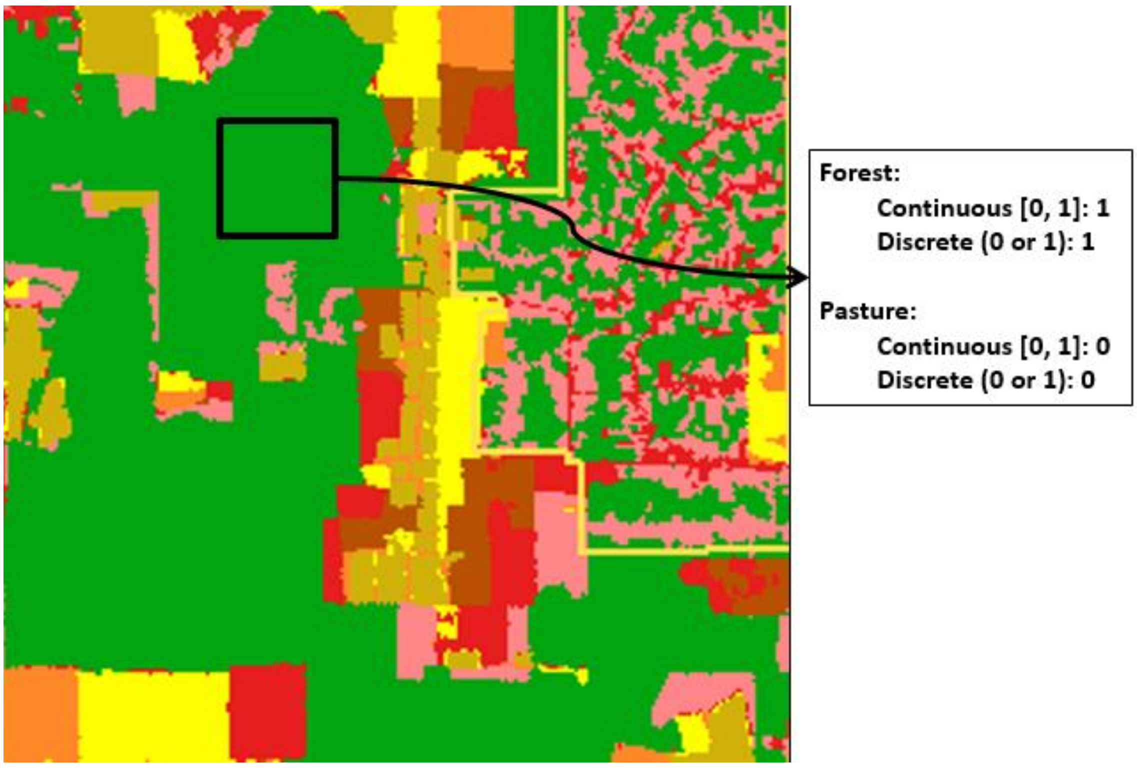
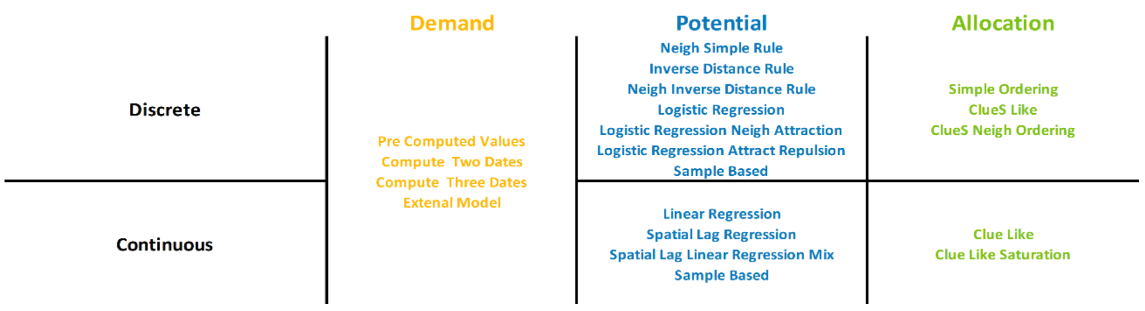


Figure 11 - Difference between models with "continuous" and "discrete" land use data.

New users are invited to explore LuccME by combining different components of Potential, Allocation and Demand already implemented (Figure 5); In this way, the user has only to parameterize an initial model from the graphical interface taking into account the guidelines presented in section 3.2.



	Demand	Potential	Allocation
Discrete	Pre Computed Values Compute Two Dates Compute Three Dates Extenal Model	Neigh Simple Rule Inverse Distance Rule Neigh Inverse Distance Rule Logistic Regression Logistic Regression Neigh Attraction Logistic Regression Attract Repulsion Sample Based	Simple Ordering ClueS Like ClueS Neigh Ordering
Continuous		Linear Regression Spatial Lag Regression Spatial Lag Linear Regression Mix Sample Based	Clue Like Clue Like Saturation

Figure 12 - LuccME components

The fifth and last step for constructing a model in LuccME is to run the model after its parameterization and, above all, in the calibration and validation of the proposed model. This step will be described in detail in the last section of this chapter.

3.2 Setting Model parameters

With the creation of the LuccME graphical interface, the process of creating a new model became simplified; the user just simply set the model parameters. In the coming sessions will be explained all the parameters of a LuccME 3.1 model using the structure: the interface name (*name in Lua code*): explanation. For advanced users it is recommended that they create a first version in the graphical interface and then modify the generated files.

3.2.1 Model Definition

The parameters related to the application model are:

Folder to Save the Model (*dofile*): Place where the generated files will be saved.

Model Name (*name*): Name for the LuccME model.

Start Time (*startTime*): The initial year considered in the model.

End Time (*endTime*): The last year of simulation.

3.2.2 Spatial Definition

The parameters related to the spatial data are:

File (*file*): Inform the name of the file that has the cellular space, it can be a project file (Terraview) or a vector file (Shape).
Layer Name (*layer*): the name of the Layer where the cellular space is placed on the Terraview project (not used in Shape file).
Cell Area (*cellArea*): Informs the cells spatial resolution (cells size).

3.2.3 Land Use Types

The parameters related to the land use types are:

Land Use Types (*landUseTypes*): Informs the name of each land use type as informed in the spatial database.
Land Use No Data (*landUseNoData*): Informs the land use type on which eventual no data generated by the system will be added, for instance, edge effects (normally static land use types).

3.2.1 Components

The user has to choose the components according to his application/model.
The explanation about which component and its parameters can be found in section 3.3.

3.2.2 Parameters to Save

The parameters related to model output, what will be saved, are:

Output Name (*outputTheme*): If a Terraview project is used, the name will be the name of the output layer, otherwise if a Shape is used, the name will be name of the output file.

Yearly Save (*yearly*): If selected on the interface (true in source code) each simulation year will be saved.

Years to Save (*saveYears*): Informs the output years to be saved.

Atributes to Save (*saveAttrs*): Informs the land use types to be saved.

To each land use type 3 different outputs will be saved:

_out: The allocation patter of each land use type.

_chtot: Change between initial time and final simulation time.

_chpast: Change between saved year and previous simulation year.

_pot: The land use change potential surface.

3.3 Setting Components

Another step to build a new spatial model on LuccME 3.1 is to configure the *Components* used on the application/model (see the labs folder on LuccME package for an example). The Potential, Allocation and Demand components are defined and parameterized by the user on the Components section and are recorded on the sub-model file. The algorithms and parameters available for each component on LuccME 3.1 are described from section 3.3.1 to section 3.3.3.

3.3.1 Potential Component

This component indicates the suitability or potential of change to each cell for a given land use type. Such potential is based on spatial driving factors (e.g. distance to roads, distance to urban centers, presence of protected areas and/or neighborhood characteristics), indicated by the user to be determinant for the location of land use changes. Driving factors selection and their relative weights definition is normally based on empirical analyses, which should be consist with the modeling approach running on LuccME. For example, if someone runs an algorithm based on linear regression to project cell's potential of change, then driving factors selection and weights should be defined based on the same approach. The process of driving factors selection and its coefficients (e.g. R^2) are inputs to the Potential component, therefore defined outside LuccME. The algorithms of Potential available on LuccME 3.1 for *Continuous* and *Discrete* models are summarized below.

3.3.1.1 Discrete Potential Components

The algorithms of Potential available on LuccME 3.1 for *Discrete* models are:

PotentialDNeighSimpleRule : Simple model developed as teaching material. Not to be used in real applications. Estimates cell potential for a given land use according to the percentage of cells of the same type in a Moore neighborhood;

Specific Allocation parameters are not available for this algorithm.

PotentialDInverseDistanceRule : Simple model developed as teaching material. Not to be used in real applications. Estimate the potential of change to each cell for a given type of land use according to the sum of the inverse of the value of a given attribute, usually weighted by a distance attribute).

For each land use type:

Const (*const*): Related to the importance of the multipliers (0-1).

Betas (*betas*): Informs the attribute name and its weight.

PotentialDNeighInverseDistanceRule : Simple model developed as teaching material. Not to be used in real applications. Estimates cell potential combining the two methods above;

For each land use type:

Weight (*const*): Related to the importance of the multipliers (0-1).

Attribute (*betas*): Informs the attribute name and its weight.

PotentialDLogisticRegression: Uses logistic regression techniques to compute cell probability for each land use type. Based on the original CLUE-S framework potential calculation (Verburg et al., 2002). For each land use, an elasticity is also defined, which will indicate how difficult it is to remove that type of land use from the cell (varies from 0-1, being 1 very easy and 0 very difficult). This elasticity is added to the probability estimated by logistic regression for the current land use of the cell (area as an "anchor" - the larger, the more difficult it is to change to another type of land use).

For each land use type:

Const (*const*): Informs the regression constant.

Elasticity (*elasticity*): Informs the regression elasticity (0-1). (Increase (closer to 1) or decrease (closer to 0) the potential of change).

Betas (*betas*): Informs the name and weights (betas) of each independent variable.

PotentialDLogisticRegressionNeighAttract: Modification of the *LogisticRegression* component combining cellular automata based models ideas. Cell potential is modified according to the attractiveness of the same class in a given (generic) neighborhood.

filename: Inform the neighborhood file (gal). This is an optional parameter, if not informed a Moore neighborhood will be built. *(Not available on the graphic interface)*

For each land use type:

Const (*const*): Informs the regression constant.

Elasticity (*elasticity*): Informs the regression elasticity (0-1). (Increase (closer to 1) or decrease (closer to 0) the potential of change).

percNeighborsUse (*percNeighborsUse*): Informs to what neighborhood extent such attraction is valid.

Betas (*betas*): Informs the name and weights (betas) of each independent variable.

PotentialDLogisticRegressionNeighAttractRepulsion : Modification of the *LogisticRegression* component combining cellular automata based models ideas. Cell potential is modified according to the affinity matrix in a given (generic) neighborhood.

filename: Inform the neighborhood file (gal). This is an optional parameter, if not informed a Moore neighborhood will be built. *(Not available on the graphic interface)*

For each land use type:

Const (*const*): Informs the regression constant.

Elasticity (*elasticity*): Informs the regression elasticity (0-1). (Increase (closer to 1) or decrease (closer to 0) the potential of change).

percNeighborsUse (*percNeighborsUse*): Informs to what neighborhood extent such attraction is valid.

Betas (*betas*): Informs the name and weights (betas) of each independent variable.

Affinity Matrix (*affinityMatrix*): Informs if the uses attracts one to other (values between 0.1 to 1, where 1 it's most attractive), repels themselves (values between -0.1 to -1, where -1 it's most repels) or indifferent (value equal to 0).

PotentialDSampleBased: Component based on the concepts of maximum entropy. The algorithm uses attributes informed on the model to select a set of samples representing each use. Using the data sample, it searches into the cellular space cells that has potential for that use, making a series of weights to determine the potential of the cell. For all the informed attributes how near the value of the attribute to the average of the sample data, greater will be the potential.

For each land use type:

Percentage to consider use (*cellUsePercentage*): On discrete models the value must be 100 (percentage value).

Range Attributes (*attributesPerc*): Informs range attributes, such as temperature, distance, so on.

Categorical Attributes (*attributesClass*): Informs categorical attributes, such as presence or not, classification, so on.

3.3.1.2 Continuous Potential Components

The potential components available in LuccME 3.1 for *Continuous* models (*LinearRegression, SpatialLagRegression, SpatialLagLinearRoads, SampleBased*) are described below:

PotentialCLinearRegression: Uses linear regression techniques to estimate the cell potential for change. Based on the original continuous CLUE model proposal, the change potential for each land use (increase or decrease the percentage in the cell at a given time is estimated as the difference between the regression cover and the actual land use percentage at a given time (see Verburg et al. 1999 for a detailed description);

For each land use type:

isLog (*isLog*): Informs if the land use type considered is Log transformed.

Const (*const*): Informs the regression constant.

Betas (*betas*): Informs the name and weights (betas) of each independent variable.

PotentialCSpatialLagRegression: Similar to the *LinearRegression* approach, but relies spatial regression techniques to estimate the regression cover (considers the spatial dependence of the land use);

For each land use type:

isLog (*isLog*): Informs if the land use type considered is Log transformed.

Const (*const*): Informs the regression constant.

minReg (*minReg*): Informs the minimum value of the regression (0-1) that should be considered to avoid spreading a certain use in cells with low potential. It should be used for unidirectional use classes and always have demand for increase from one year to another (eg deforestation). If not suitable, use value 0.

maxReg (*maxReg*): Informs the maximum value of the regression (0-1) that should be considered to avoid decreasing use in cells with high potential. It should be used for unidirectional use classes and have always demand for a decrease from one year to another (for example, primary forest). If not suitable, use value 1.

ro (*ro*): Informs the correlation coefficient of the regression (0-1).

Betas (*betas*): Informs the name and weights (betas) of each independent variable.

PotentialCSpatialLagLinearRegressionMix: Modification of the *SpatialLagRegression* component to allow the use of simple linear regression in specific cases, when roads are created or paved. The component is an example of how the framework can be extended for specific applications. It was created to allow the representation of the creation of new deforestation frontiers in the Brazilian Amazon.

For each land use type:

isLog (*isLog*): Informs if the land use type considered is Log transformed.

Const (*const*): Informs the regression constant.

minReg (*minReg*): Informs the minimum value of the regression (0-1) that should be considered to avoid spreading a certain use in cells with low potential. It should be used for unidirectional use classes and always have demand for increase from one year to another (eg deforestation). If not suitable, use value 0.

maxReg (*maxReg*): Informs the maximum value of the regression (0-1) that should be considered to avoid decreasing use in cells with high potential. It should be used for unidirectional use classes and have always demand for a decrease from one year to another (for example, primary forest). If not suitable, use value 1.

ro (*ro*): Informs the correlation coefficient of the regression (0-1).

Betas (*betas*): Informs the name and weights (betas) of each independent variable.

Roads Model Attributes (*roadsModel*): Informs the regression data taking into account new roads built or paved.

Roads Model Attributes (*attrs*): Informs the variable names related to new roads.

Const (*const*): Informs the linear regression constant.

Change (*change*): Informs if cell attribute value (related to roads) should change after a certain level (e.g. -1.5).

Betas (*betas*): Informs the name and weights (betas) of each independent variable.

PotentialCSampleBased: Component based on the concepts of maximum entropy. The algorithm uses attributes informed on the model to select a set of samples representing each use. Using the data sample, it searches into the cellular space cells that has potential for that use, making a series of weights to determine the potential of the cell. For all the informed attributes how near the value of the attribute to the average of the sample data, greater will be the potential.

For each land use type:

Percentage to consider use (*cellUsePercentage*): Informs a percentage of the use in the cell to consider a sample (0-100).

Range Attributes (*attributesPerc*): Informs range attributes, such as temperature, distance, so on.

Categorical Attributes (*attributesClass*): Informs categorical attributes, such as presence or not, classification, so on.

3.3.2 Allocation Component

The allocation components perform the spatial distribution of land use and cover changes based on the land demand (section 3.3.3) and potential of change of each cell (section 3.3.1). Several allocation rules can also be defined in the allocation component (e.g. amount and speed of change allowed to each time-step), but its parameterization depend on the user needs. The algorithms and parameters of Allocation available on LuccME 3.1 for *Continuous* and *Discrete* models are presented below.

3.3.2.1 Discrete Allocation Components

The algorithms of Allocation available on LuccME 3.1 for *Discrete* models are:

AllocationDClueSLike: Based on the process of competition among classes in the same cell, adjusted iteratively to satisfy the demand when all cells are considered, as described in Verburg et al. (2002), but extended to incorporate new features, such as change in blocks (optional, parameterized by cell);

maxIteration (*maxIteration*): Informs the maximum number of iterations allowed at each time step of the model (must be integer > 0; recommended value = 1000). If this number of interactions is reached without allocation success, then the model stops and return an error message informing land demand allocation incompatibility;

factorIteration (factorIteration): Initial value of the parameter that controls the allocation factor interaction.

maxDifference (*maxDifference*): Informs the maximum difference between the land demand value informed by the user and the amount allocated by the model to each land use type (must be informed at the same unit as cell area);

Transition Matrix (*transitionMatrix*): Informs the allowable (1) and not allowable (0) transition in a land-use x land-use matrix (must have at least one region).

AllocationDSimpleOrdering: Simple model developed as teaching material. Not to be used in real applications. Instead of using the iterative process employed in the CLUE family, the component implements a simple ordering approach.

maxDifference (*maxDifference*): Informs the maximum difference between the land demand value informed by the user and the amount allocated by the model to each land use type (must be informed at the same unit as cell area);

AllocationDClueSNeighOrdering: This component is a modification of *AllocationClueSLike* componente, where, before the allocation execution the potential of each land use is ordered according to neighborhood of each cell.

maxIteration (*maxIteration*): Informs the maximum number of iterations allowed at each time step of the model (must be integer > 0; recommended value = 1000). If this number of interactions is reached without allocation success, then the model stops and return an error message informing land demand allocation incompatibility;
factorIteration (factorIteration): Initial value of the parameter that controls the allocation factor interaction.
maxDifference (*maxDifference*): Informs the maximum difference between the land demand value informed by the user and the amount allocated by the model to each land use type (must be informed at the same unit as cell area);
Transition Matrix (*transitionMatrix*): Informs the allowable (1) and not allowable (0) transition in a land-use x land-use matrix (must have at least one region).

3.3.2.2 Continuous Allocation Components

The algorithms of Allocation available on LuccME 3.1 for *Continuous* models are:

AllocationCClueLike: Based on the process of competition among classes in the same cell, adjusted iteratively to satisfy the demand when all cells are considered, as described in Verburg et al. (1999). Cells receive a percentage of the annual change that must be allocated to the whole area, proportionally to their potential. The INPE version differs from the original CLUE model as it was adapted for the Brazilian context, for instance to represent the Forest Code and law enforcement (Aguiar, 2006). For instance, there are parameters to control the amount and speed of change that can happen in each cell;

maxDifference (*maxDifference*): Informs the maximum difference between the land demand value informed by the user and the amount allocated by the model to each land use type (must be informed at the same unit as cell area);

maxIteration (*maxIteration*): Informs the maximum number of iterations allowed at each time step of the model (must be integer > 0; recommended value = 1000). If this number of interactions is reached without allocation success, then the model stops and return an error message informing land demand allocation incompatibility;

initialElasticity (*initialElasticity*): Informs the initial elasticity value (iterationFactor) (float > 0; recommended value = 0.1).

minElasticity (*minElasticity*): Informs the minimum elasticity value (iterationFactor) (float > 0; recommended value = 0.001).

maxElasticity (*maxElasticity*): Informs the maximum elasticity value (iterationFactor) (float > 1; recommended value = 1.5).

complementarLU (*complementarLU*): Informs the land use type which will be recomputed in the end if the total percentages do not amount exactly 100% (null or a valid land use name).

Allocation Data (*allocationData*): Parameters to define specific allocation rules for each land-use type.

static (*static*): Informs if the variable can increase or decrease in each cell, or only change in the direction of the demand. Possible values: -1 (only changes in the direction of the demand); 0 (change in any direction); 1 (does not change).

minValue (*minValue*): Informs the minimum value allowed for the percentage of a given land use in a cell (as a result of new changes - the original value can be out of the limits) (float [0,1]). Defined according to territorial planning restrictions (such as the Forest Code). See that the restrictions have to make sense among different land uses as they must sum 100% (so for instance, you cannot say all land use have a minimum limit of 50% if you have 3 or more classes).

maxValue (*maxValue*): Informs the maximum value allowed for the percentage of a given land use in a cell (as a result of new changes - the original value can be out of the limits) (float [0,1]).

minChange (*minChange*): Informs the minimum change in a given land use in a cell in a time step until (saturation) threshold (float [0,1]).

maxChange (*maxChange*): Informs the maximum change in a given land use allowed in a cell in a time step until (saturation) threshold (float [0,1]).

changeLimiarValue (*changeLimiarValue*): Informs the threshold to a given amount of the land use in each cell. After this threshold, the speed of change of a given land use in the cell is modified (float [0,1]). Must be calibrated according to the amount of change recorded on average from one year to the other in the observed data.

maxChangeAboveLimiar (*maxChangeAboveLimiar*): Informs the maximum change in a given land use allowed in a cell in a time step after (saturation) threshold (float [0,1]).

AllocationCClueLikeSaturation: Modification of the *AllocationClueLike* component. In this case the speed of change in each cell use a spatiotemporal variable, dynamically updated every year, that indicates if the cell is in a more consolidated or in a frontier area. The saturation threshold considers a 10x10 neighborhood, discounting protected areas;

maxDifference (*maxDifference*): Informs the maximum difference between the land demand value informed by the user and the amount allocated by the model to each land use type (must be informed at the same unit as cell area);

maxIteration (*maxIteration*): Informs the maximum number of iterations allowed at each time step of the model (must be integer > 0; recommended value = 1000). If this number of interactions is reached without allocation success, then the model stops and return an error message informing land demand allocation incompatibility;

initialElasticity (*initialElasticity*): Informs the initial elasticity value (iterationFactor) (float > 0; recommended value = 0.1).

minElasticity (*minElasticity*): Informs the minimum elasticity value (iterationFactor) (float > 0; recommended value = 0.001).

maxElasticity (*maxElasticity*): Informs the maximum elasticity value (iterationFactor) (float > 1; recommended value = 1.5).

complementarLU (*complementarLU*): Informs the land use type which will be recomputed in the end if the total percentages do not amount exactly 100% (null or a valid land use name).

saturationIndicator (*saturationIndicator*): Informs the name of a attribute which will be dynamically updated (and can be saved for calibration purposes).

attrProtection (*attrProtection*): Informs the database attribute indicating the percentage of protected areas to be excluded from the saturation level computation).

Allocation Data (*allocationData*): Parameters to define specific allocation rules for each land-use type.

> static (*static*): Informs if the variable can increase or decrease in each cell, or only change in the direction of the demand. Possible values: -1 (only changes in the direction of the demand); 0 (change in any direction); 1 (does not change).
>
> minValue (*minValue*): Informs the minimum value allowed for the percentage of a given land use in a cell (as a result of new changes - the original value can be out of the limits) (float [0,1]). Defined according to territorial planning restrictions (such as the Forest Code). See that the restrictions have to make sense among different land uses as they must sum 100% (so for instance, you cannot say all land use have a minimum limit of 50% if you have 3 or more classes).
>
> maxValue (*maxValue*): Informs the maximum value allowed for the percentage of a given land use in a cell (as a result of new changes - the original value can be out of the limits) (float [0,1]).
>
> minChange (*minChange*): Informs the minimum change in a given land use in a cell in a time step until (saturation) threshold (float [0,1]).
>
> maxChange (*maxChange*): Informs the maximum change in a given land use allowed in a cell in a time step until (saturation) threshold (float [0,1]).

changeLimiarValue (*changeLimiarValue*): Informs the threshold to a given amount of the land use in each cell. After this threshold, the speed of change of a given land use in the cell is modified (float [0,1]). Must be calibrated according to the amount of change recorded on average from one year to the other in the observed data.

maxChangeAboveLimiar (*maxChangeAboveLimiar*): Informs the maximum change in a given land use allowed in a cell in a time step after (saturation) threshold (float [0,1]).

3.3.3 Demand Component

This component defines the quantity or the magnitude of land use changes to be allocated in each time-step. This information is frequently an input to spatial models built on LuccME, especially when applied to scenario purposes. Several methods are used to perform such estimates such as baseline projections, scenario assumptions or land demand requirements derived from economic modeling. Land demand estimates might also be generated internally when two or more past periods of land use data is provided in the spatial database (section 3.2.3.1). All algorithms and parameters of Demand available on LuccME 3.1 are summarized below and might be used for both *Continuous* and *Discrete* models.

DemandPreComputedValues: Uses predetermined values of land demand provided by the user for each time-step.

Demanda Anual (*annualDemand*): Informs the annual land demand of each land use type (must be the same unit as cell area) in the same order as informed in the *Land Use Types*.

DemandComputeTwoDates: Land demand values are computed internally based on two different past periods of land use data provided by the user in the spatial database.

Final Year for Interpolation (*finalYearForInterpolation*): The year to be considered the end of interpolation to calculate the demand.

Columns for Interpolation (*finalLandUseTypesForInterpolation*): Informs the column name with the demand values for the end of the interpolation of each type of use (e.g. Use - F, D, O; Final Columns - F_final, D_final, O_final).

DemandComputeThreeDates: Land demand values are computed internally based on three different past periods of land use data provided by the user in the spatial database.

Middle Year (*middleYearForInterpolation*): The year to be considered the end of the first interpolation to calculate the demand.

Final Year (*finalYearForInterpolation*): The year to be considered the end of the second interpolation to calculate the demand.

Columns for Interpolation (*middleLandUseTypesForInterpolation*, *finalLandUseTypesForInterpolation*): Informs the column name with the demand values for the two interpolation calculations for each type of use (e.g. Use - F_00, D_00, O_00; Columns - F_05, D_ 05, O_ 05 (middle); F_10, D_10, O_10 (final)).

3.4 Calibration and Validation

Once the proposed model has been effectively parameterized and executed in LuccME, it is necessary to adjust and test the model's performance. These steps basically represent the Calibration and Validation process. The choice of the best component of Potential and/or Allocation, as well as, the method used for parameterization of these components is also part of the calibration process. It is an interactive process where normally several preliminary rounds are necessary until the identification of the best components and variables of the model. The different components offer a series of parameters that can be adjusted to better capture the change processes being simulated. However, it should be noted that the time horizon used for model calibration (e.g. 1995-2005) cannot be the same period used for validation. Ideally, the calibrated model should be validated for an independent period (e.g. 2005-2015). Other advanced features such as the use of dynamic variables, as well as, the definition of sub-regions can also contribute significantly to the calibration and validation of the models, as will be discussed in the next chapter.

The results of the simulations can be validated by the validation metric of multiple resolutions, adapted from Costanza (1989), implemented in LuccME. This metric allows to establish the level of similarity between simulated and observed maps at different resolutions, through sampling windows that increase at each time-step, as shown in Figure 6.

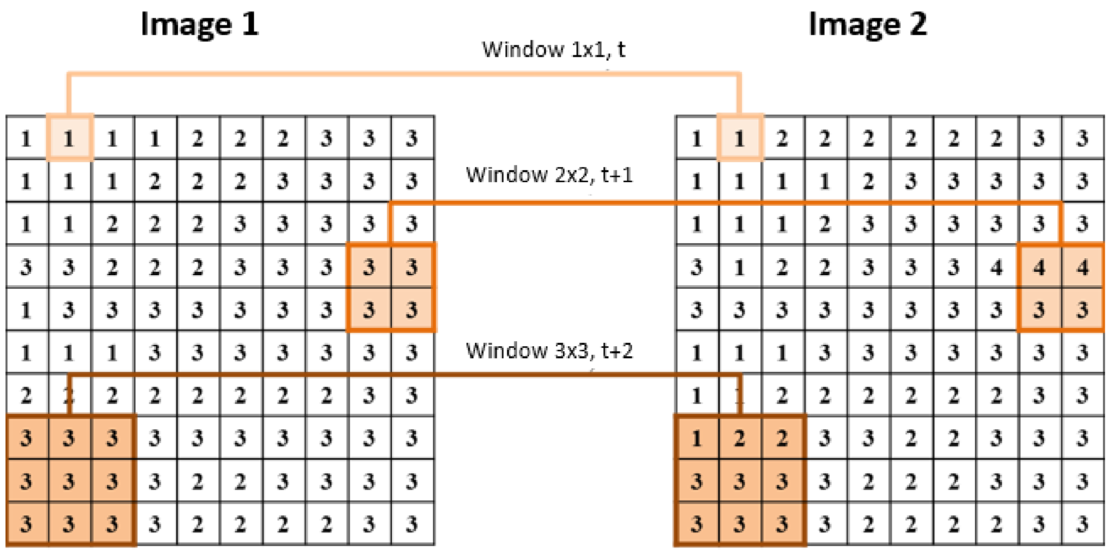


Figure 13 - Example of multiple resolution validation (cell aggregation through sampling windows).

The calculation of the level of similarity can be described according to the equation below:

$$\mathrm{NS_i} = 1 - \left[\frac{\sum_{j=1}^{n} \left(\left| \sum_{c=1}^{k} \mathbf{dif}_{\mathbf{sim,c}} - \sum_{c=1}^{k} \mathbf{dif}_{\mathbf{real,c}} \right| \right)}{\mathbf{2} * \sum_{j=1}^{n} \sum_{c=1}^{k} \mathbf{dif}_{\mathbf{real,c}}} \right]$$

In which ***NS*** corresponds to the level of similarity between observed and simulated maps in a given resolution ***i***; ***j*** is the window considered; ***n*** establishes the number of windows to be considered; ***c*** is the number of cells in a resolution ***k*** (i*i); and ***dif***$_{real}$ = % real$_{tf}$ - % real$_{ti}$ and ***dif***$_{sim}$ = % sim$_{tfinal}$ - % real$_{tinicial}$ being ***ti*** and ***tf*** the initial and real years, respectively, considered in the validation.

In this version of LuccME two validation options are available:

1. Considering all cellular space (Ex): This option takes into account the accumulated historical pattern for the different types of land use;

2. Considering only the modified areas (Dif): This option takes into account only the areas that have undergone some change process, so it is a more specific validation system for each type of land use.

4 Advanced resources

4.1 Dynamic variables

An advanced feature for model building in LuccME 3.1 is the use of dynamic variables. This type of resource is extremely useful to the user when the driving factors that explain the spatial distribution of the land use types change significantly over the period considered. In this case, dynamic variables can be updated in different time-steps to improve the model's performance.

When you want to use dynamic variables, new cell spaces must be created in TerraME 2.0 or another user's preference GIS with the following organization:

<CellSpaceName>_<UpdateYear> (e.g. CS_2015).

This means that in the update year indicated in the model, LuccME will read the updated cell space. For an example of using this feature, please see Lab 6 for Continuous models provided with the LuccME package. Another example of a complete application using this type of resource can be observed in Aguiar et al. (2015).

4.2 Regions

Another advanced resource to build new models on LuccME 3.1 is the use of one or more sub-regions. This type of resource is extremely useful when the driving factors that explain the spatial patterns of land use strongly change over space. In this case, sub-regions definition allows the user to set variables and coefficients more consistent with local processes and improve model's performance. Different sub-regions must be informed by the user in the Potential component.

For an example of usage of this resource, please, see Lab Region for *Continuous* models available with LuccME package. An example of an application built using this type of resource (with previous versions of LuccME) is Coelho (2009) and Pimenta et al. (2010).

4.3 Scenarios

Once the new model developed by the user is able to consistently represent the spatial patters of land cover change observed in the past, LuccME 3.1 provides the opportunity to explore future scenarios of land use change. In this case, the user has to inform the stat-time and the scenario-name, as well as, the update periods for dynamic variables (if applicable).

Years with variables to be Updated (*updateYears*): Informs the years for dynamic variables update.
Scenario Start Time (*scenarioStartTime*) = Informs the scenario start year.
Nome do Cenário (scenarioName): Informs the scenario name.

For an example of using this feature, please see the Lab 7 for Continuous models available along with LuccME package. A complete sample application using this type of resource is Aguiar et Al. (2015).

5 Running a model with LuccME

5.1 Exercises

In this chapter we invite new users to explore LuccME Framework 3.1 through different practical exercises provided at LuccME webpage.

These exercises, called LuccME labs, include both *Continuous* and *Discrete* examples organized into different complexity levels. The users are invited to run each Lab in order to compare model's configuration and outputs. A synthesis of the main differences between Labs available for *Continuous* and *Discrete* models are presented in Tables 2 and 3.

Once the user gets familiar with LuccME and its tools, model examples might be used as the basis to build new land use models. In other words, the user can save time configuring new models based on previous examples as the LuccME labs.

Table 2 - Summary of *Discrete* Lab exercises.

Exercise	Components		
	Potential	**Allocation**	**Demand**
Lab1	*PotentialDNeighSimpleRule*	*AllocationDSimpleOrdering*	*DemandPreComputedValues*
Lab2	*PotentialDInverseDistanceRule*	*AllocationDSimpleOrdering*	*DemandPreComputedValues*
Lab3	*PotentialDNeighInverseDistanceRule*	*AllocationDSimpleOrdering*	*DemandPreComputedValues*
Lab4	*PotentialDLogisticRegression*	*AllocationDSimpleOrderingg*	*DemandPreComputedValues*
Lab5	*PotentialDLogisticRegressionNeighAttract*	*AllocationDClueSLike*	*DemandPreComputedValues*
Lab6	*PotentialDLogisticRegression*	*AllocationDClueSLike*	*DemandPreComputedValues*
Lab7	*PotentialDLogisticRegression*	*AllocationDClueSLike*	*DemandComputeTwoDates*
Lab8	*PotentialDLogisticRegression*	*AllocationDClueSLike*	*DemandComputeThreeDates*
Lab9	*PotentialDSampleBased*	*AllocationDClueSLike*	*DemandPreComputedValues*
Lab10	*PotentialDLogisticRegressionNeighAttractRepulsion*	*AllocationDClueSNeighOrdering*	*DemandPreComputedValues*
Lab11	*PotentialDLogisticRegression*	*AllocationDClueSNeighOrdering*	*DemandComputeTwoDates*
Lab12	*PotentialDLogisticRegressionNeighAttractRepulsion*	*AllocationDClueSLike*	*DemandPreComputedValues*
Lab13	*PotentialDNeighSimpleRule*	*AllocationDSimpleOrdering*	*DemandPreComputedValues*
Lab14	*PotentialDInverseDistanceRule*	*AllocationDSimpleOrdering*	*DemandPreComputedValues*

Table 3 - Summary of *Continuous* Lab exercises.

Exercise	Components		
	Potential	**Allocation**	**Demand**
Lab1	*PotentialCLinearRegression*	*AllocationCClueLike*	*DemandPreComputedValues*
Lab2	*PotentialCSpatialLagRegression*	*AllocationCClueLike*	*DemandPreComputedValues*
Lab3	*PotentialCSpatialLagRegression*	*AllocationCClueLikeSaturation*	*DemandPreComputedValues*
Lab4	*PotentialCSpatialLagRegression*	*AllocationCClueLike*	*DemandComputeTwoDates*
Lab5	*PotentialCSpatialLagRegression*	*AllocationCClueLike*	*DemandComputeThreeDates*
Lab6	*PotentialCSpatialLagRegression*	*AllocationCClueLike*	*DemandPreComputedValues*
Lab7	*PotentialCSpatialLagRegression*	*AllocationCClueLike*	*DemandPreComputedValues*
Lab8	*PotentialCSpatialLagLinearRegressionMix*	*AllocationCClueLike*	*DemandPreComputedValues*
Lab9	*PotentialCSampleBased*	*AllocationCClueLike*	*DemandPreComputedValues*
Lab10	*PotentialCLinearRegression*	*AllocationCClueLike*	*DemandExternalModel*

References

AGUIAR, A.P.D. Modeling land use change in the Amazon: Exploring intra-regional heterogeneity. 2006. 153 p. (INPE- 08.10.18.21-TDI). Thesis (Doctoral in Remote Sensing) - Brazilian National Institute for Space Research (INPE), São José dos Campos, 2006. Retrieved from: http://urlib.net/6qtX3pFwXQZGivnJSY/M7t7e.

CARNEIRO, T. G. S., ANDRADE, P. R., CÂMARA, G., MONTEIRO, A. M. V., & PEREIRA, R. R. (2013). An extensible toolbox for modeling nature–society interactions. Environmental Modelling & Software, 46, 104-117.

VERBURG, P. H.; KOK, K.; PONTIUS JR, R. G.; VELDKAM P, A. Modeling land-Use and land-cover change. In: LAMBIN, E.F.; GEIST, H. (Eds.). Land-use and land-cover change: local processes and global impacts. Berlin: Springer, 2006, p. 117-135.

VELDKAMP, A.; FRESCO, L. CLUE-CR: an integrated multi-scale model to simulate land use change scenarios in Costa Rica. Ecological Modeling, 91: 231-248, 1996.

VERBURG, P.; DE KONING, G.; KOK, K.; VELDKAMP, A.; BOUMA, J. A spatial explicit allocation procedure for modeling the pattern of land use change based upon actual land use. Ecological Modeling, 116: 45-61, 1999.

VERBURG, P.; VELDKAMP, A. The role of spatially explicit models in land-use change research: a case study for cropping patterns in China. Agriculture, Ecosystems and Environment, 85: 177–190, 2001.

SOARES-FILHO, B.; CERQUEIRA,; PENNACHIN, C. DINAMICA – a stochastic cellular automata model designed to simulate the landscape dynamics in an Amazonian colonization frontier. Ecological Modeling, 154 (3): 217 – 235, 2002.

PONTIUS, R. G.; CORNELL, J. D.; HALL, C. A. Modeling the spatial pattern of land-use change with GEOMOD2: application and validation for Costa Rica. Agriculture, Ecosystems & Environment, v. 85, n. 1, p. 191-203, 2001.