Modeling Environmental Dynamics with Dinamica EGO

Britaldo S. Soares-Filho, Hermann O. Rodrigues, William L. Costa

TEXT EDITING
Britaldo Silveira Soares-Filho
TEXT REVISING
Silvia Weel
Peter Schlesinger
COVER PAGE
Britaldo Silveira Soares-Filho
Flávio de Castro Oliveira
GRAFIC PRODUCTION
Britaldo Silveira Soares-Filho
Flávio de Castro Oliveira
FORMATTING
Britaldo Silveira Soares-Filho
Flávio de Castro Oliveira
Letícia de Barros Viana Hissa

Britaldo Silveira Soares-Filho
Av. Antônio Carlos, 6627 – Universidade Federal de Minas Gerais – Campus Pampulha
Instituto de Geociências – Centro de Sensoriamento Remoto, suite 206
31270-901 - Belo Horizonte/ MG - Brazil
Tel + 55 (31) 3409-5449 - Fax + 55 31 3409-5410
Email: dinamica@csr.ufmg.br
www.csr.ufmr.br/dinamica

Soares-Filho, Britaldo Silveira; Rodrigues, Hermann Oliveira; Costa, William Leles Souza

Modeling Environmental Dynamics with Dinamica EGO/ Britaldo Silveira Soares-Filho, Hermann Oliveira Rodrigues, William Leles Souza Costa. 2ed (2010). - Belo Horizonte: Britaldo Silveira Soares-Filho, 2009.
120 p.

ISBN: 978-85-910119-0-2

1. Environmental Modeling. 2. Landscape Dynamics. I Title.

CDD: 000

Contents

1. Introduction

What will you learn?

- What is Dinamica EGO
- What is a functor
- What is this guidebook about
- What's new in Dinamica EGO version 1.4 (October 2009)

Welcome to Dinamica EGO. EGO stands for Environment for Geoprocessing Objects. The previous version of Dinamica was totally reengineered to become a modeling environment, now with outstanding possibilities for the design from the very simple static spatial model to very complex dynamic ones, which can ultimately involve nested iterations, dynamic feedbacks, multi-region approach, manipulation and algebraic combination of data in several formats, such as maps, tables, matrices and constants, decision processes for bifurcating and joining execution pipelines, and a series of complex spatial algorithms for the analysis and simulation of space-time phenomena (fig.1).

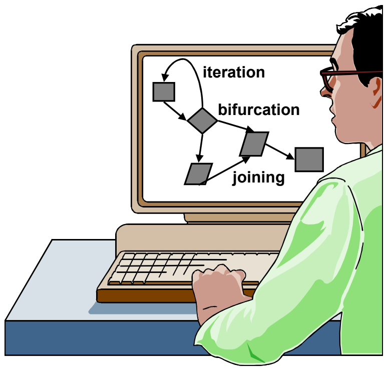


Fig. 1 - A data flow chain showing iteration, bifurcation, and joining.

The software environment, written in C++ and Java, holds a series of algorithms called functors. Each functor performs an operation. To date, we have implemented the most common spatial analysis algorithms available in commercial GIS (Geographic Information System), plus a series of algorithms especially designed for spatial simulations, including transition functions, calibration and validation methods.

These functors are sequenced to establish data flow in the form of graphs. Through the Dinamica EGO graphical interface one can create models by simply dragging and connecting functors via their ports, which represent connectors to types of data, such as maps, tables, matrices, mathematical expressions and constants. Functors can be enveloped by "containers", a special type of functor that is used, for example, to execute iterations or process data from specific regions of a map. Thus models can be designed as a diagram and execution follows a data flow chain. This friendly interface permits the design of simple to very complex spatial models that are saved in a script language in XML format or EGO programming language.

In sum, Dinamica EGO software favors simplicity, flexibility and good performance, optimizing speed and computer resources, such as virtual memory and parallel processing. Most of its algorithms are designed to take advantage of dual or higher processor architecture. In addition, Dinamica EGO handles large raster format, up to 64000x64000 cells, using disk paging. On the other hand, if memory is available, it can load all the input maps at the beginning of a model execution and keep them in memory only while they are needed. In this way, the software only accesses the disk at the end of an execution to write the final outputs or, if a user specifies, at the end of an iteration to save the output maps from each time step.

The aim of this guidebook is to introduce the user to the innumerous possibilities of Dinamica EGO for the design of models that can fully represent the complexity of various geographic phenomena.

What's new in current Version 1.6

The development of version 1.6 has highly prioritized performance improvement as follows:

* 64 bit support: Dinamica EGO now can be installed as a native 64-bit program on Windows Seven 64 and Vista 64, taking advantage of expanded memory addressing and performance optimizations.

*Better memory management: Dinamica EGO now handles larger and more sophisticated models even if running in 32 bit version or on the graphical interface. Several large models that failed on the graphical interface now run properly.

* Similar to the console version, Dinamica EGO graphical interface now also takes advantage of multiple available processors to run a model.

* New console launcher allows for queuing models and running them sequentially.

* Native expression compilation: Calculate Map expressions can be compiled to native code, improving, as result, model performance. The compilation takes place automatically without user intervention. Users just need to install the optional package for native expression support!

* The precision used to store values on lookup tables has been increased, enabling storage of large numbers.

* The precision for map, value and table calculations has been increased, preventing the occurrences of spurious rounding errors. The operators *CalculateCategoricalMap*, *CalculateMap*, *CalculateValue* and *CalculateLookupTable* have been updated.

* Model optimization tool based on Genetic Algorithm. Dinamica EGO has incorporated an *Algorithm Genetic Tool* that performs heuristic calibration of models. This generic tool allows a set of model parameters to be assembled into tables and input as a gene to the GA tool. The GA tool contains several options for the optimization process, such as number of individuals, generations, exit condition to stop optimization process as evolution becomes asymptotical, as well as controls for overspecialization. A text and examples are included to illustrate the potential of the GA tool for improving transition probability maps obtained from Weights of Evidence method.

* New EGO script configuration options: it is now possible to customize the way Dinamica EGO writes EGO scripts, making easier to write and to understand models in textual format.

* Dinamica now uses a private installation of Java runtime environment. So, it no longer needs to replace preexisting versions of Java.

* New support to read inputs and write outputs straightly from/into zip files using the new *Workdir* operator. Also, the user can use this operator to define a root folder, making easier to share and transport models across computers with different folder structure.

* New operators: several new operators are available to facilitate the manipulation of tables and maps.

* New enhanced graphical editor for lookup tables and matrices.

* New graphical interface docking framework with improved panel management and possibility for undocking panels. This enables to enlarge the model sketch, keeping at the same time some key panels afloat, such as the Bird View, thus facilitating model visualization.

* As always, we fixed several minor bugs and improved the performance of several functors.

*Sad news: the VENSIM support is deprecated and will be removed in future releases together with other deprecated operators.

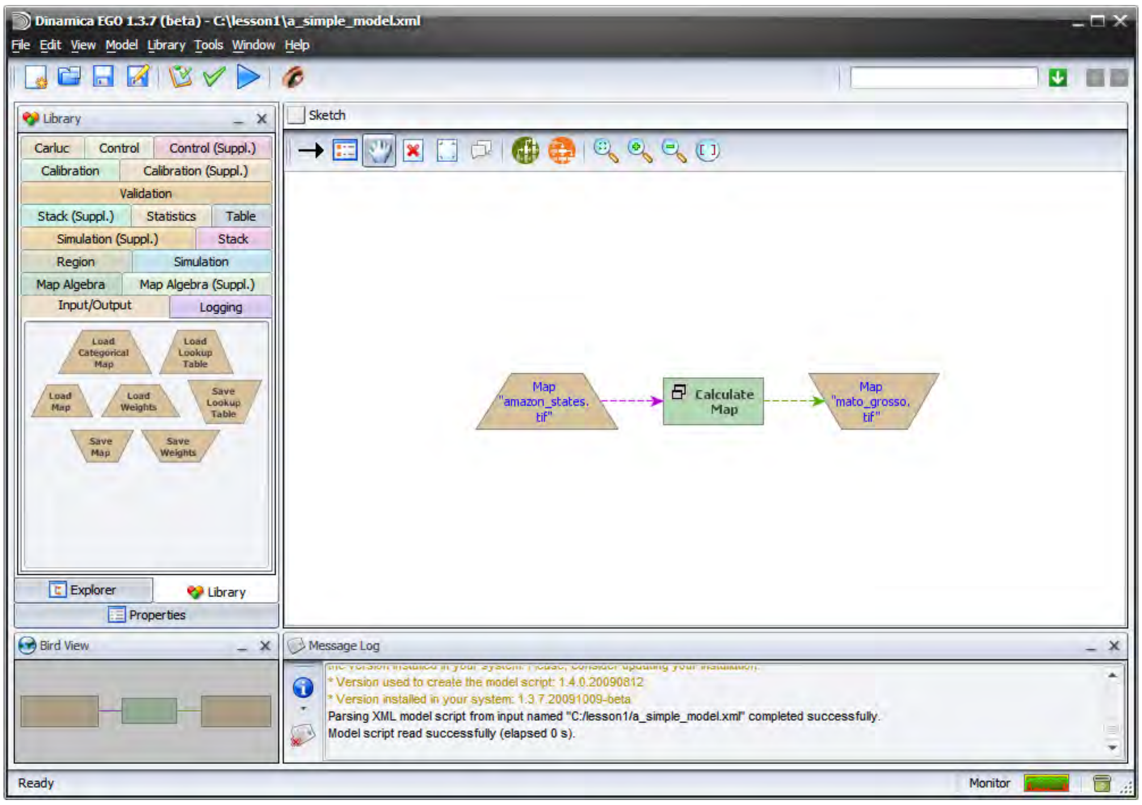
1.1 Dinamica EGO graphical interface

What will you learn?

- Dinamica EGO graphical interface
- Interface tools

Models are expressed in Dinamica EGO as a sequence of functors connected via compatible inputs and outputs. Hence data flow through these operators to produce a desirable outcome that represents the solution to a query on some environmental topic.
The Dinamica EGO graphical interface is divided into 6 windows. From the top left clockwise they are 1) the library, 2) the sketch, 3) the message log, 4) the bird view, 5) functor properties and 6) explorer tabs (fig. 2).

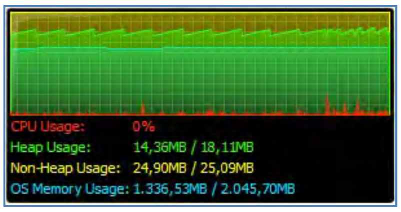
A model is designed simply by dragging functors from the library window and placing them on the sketch. The bird view provides a synoptic view of the model and is especially useful for large models that do not fit entirely on the sketch when shown at the default size. The shaded area on the bird view corresponds to the sketch window. Move the shade to zoom at a particular part of the model. The tree view provides a hierarchical view of the model allowing one to select or locate a functor and subsequently to edit its properties through the functor property window. In the lower part of the interface is the message log window, a space reserved for reporting textual messages showing results, errors, warnings, information and debug information. By default, the log text will output the level of information, but the user can change this just by pressing the button for the desired information level. **TIP**: You can increase the performance of some models reducing the level of the log message. Another way to do this is using the *log policy* functor. The sketch window has a window tool along its top border. **TIP:** You can change the interface perspective, by clicking on the Window drop-down option on the menu bar and choosing one of the three available perspectives.



```
* Version used to create the model script: 1.4.0.20090812
* Version installed in your system: 1.3.7.20091009-beta
Parsing XML model script from input named "C:/lesson1/a_simple_model.xml" completed successfully.
Model script read successfully (elapsed 0 s).
```

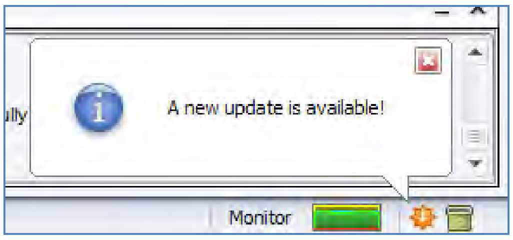
Fig. 2 – Dinamica EGO graphical interface

TIP: you can resize or close windows. This may become useful as the model increases in number of functors. You just need to increase the sketch size at the expense of the other windows' sizes.


CPU Usage: 0%
Heap Usage: 14,36MB / 18,11MB
Non-Heap Usage: 24,90MB / 25,09MB
OS Memory Usage: 1.336,53MB / 2.045,70MB

On the bar at the bottom of the application window, you will also find the monitor window and the garbage collector tool that allows the user to reduce the memory dynamically allocated by Dinamica EGO. Allow the mouse to hover over the monitor window to amplify it.

When a new version is available, a message will pop up on the interface corner window. Go to the Dinamica EGO website (www.csr.ufmg.br/dinamica) to update your version.


A new update is available!

The sketch toolset permits connecting, editing, selecting, moving functors, editing the functor ports, writing comments to a functor, organizing of the model layout, and exhibiting the model layout in different zooms (fig.3). **TIP:** you can move the toolset and place it vertically onto the left side of the sketch window. You can also access the sketch toolset by going to Edit on the menu bar or through shortcuts displayed on Edit drop-down menu.

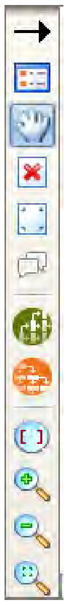

Connect functor via their ports
Edit functors
Move/Select functors
Delete functors
Edit functor ports
Write comments to a functor
Organize model layout left to right
Organize model layout left to bottom
Fit model layout to the sketch window
Zoom in
Zoom out
Zoom to default size

Fig. 3 - The sketch toolset.

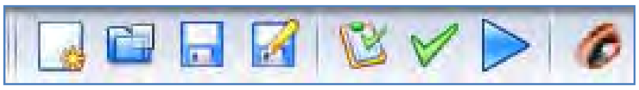
TIP: you can delete functors by selecting them with the hand tool and then pressing delete.

In the library window you will find the available functors organized in subsets according to their nature and application. The subsets are:

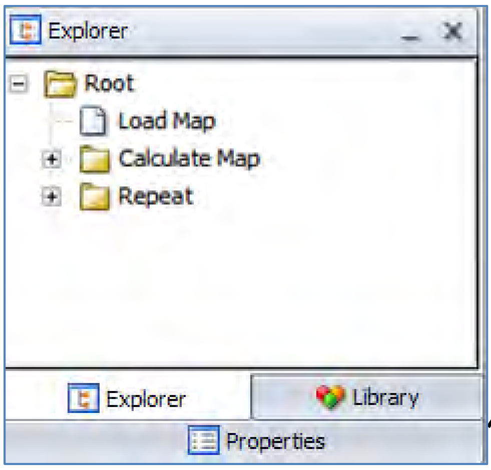
1) Map Algebra: a set of algorithms for spatial analyses.
2) Map Algebra supplementary: auxiliary functors to be used in conjunction with map algebra functors.

3) Region: these functors permits division of a map into several regional maps, each one to be used separately in a particular submodel or iteration.
4) Simulation: set of functors especially designed to develop space-time models.
5) Simulation supplementary: auxiliary functors to be used in conjunction with map simulation functors
6) Stack: these functors allow for the selection of a specific parameter according to a model step.
7) Stack supplementary: to be used in conjunction with stack functors.
8) Statistics: functors for performing spatial statistics.
9) Table: a set of functors for manipulating lookup tables.
10) Validation: metrics for map comparison.
11) Calibration: a set of tools for calibrating space-time models.
12) Calibration supplementary: to be used in conjunction with calibration functors.
13) Carluc: algorithms of CARLUC, Carbon and land use change model (Hirsh et al., 2004). This subset is now deprecated since the CARLUC model is now directly implemented using Dinamica EGO functors.
14) Control: functors that control the data flow chain.
15) Control supplementary: to be used in conjunction with Control functors.
16) Input/output: tools for reading and saving maps, tables and weight of evidence data structure.
17) Logging: specifically for spatial modeling of logging (Merry *et al*, 2009).

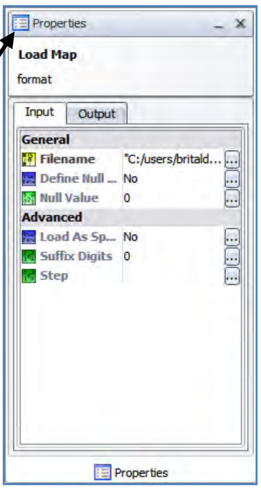
Finally, the icons on the menu bar are shortcuts to create, open, save a model, save model to a new file, check model script operator integrity, check model script connection integrity, run a model, and open a map viewer. **TIP:** It is possible to open more than one instance of Dinamica EGO. This can be useful for example for comparing the structure of two models. You can also open a series of map viewer windows to show multiple maps. Just click the eye icon again.



Once you start building a model, Dinamica EGO provides some features to help you navigate through the model. You can use the Explorer window to see a tree like representation of your model.



Clicking on a tree element will select it and enable editing via the property window:

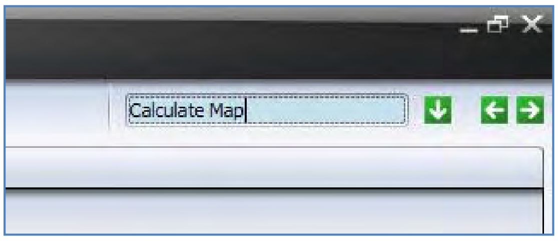

Properties

Load Map
format

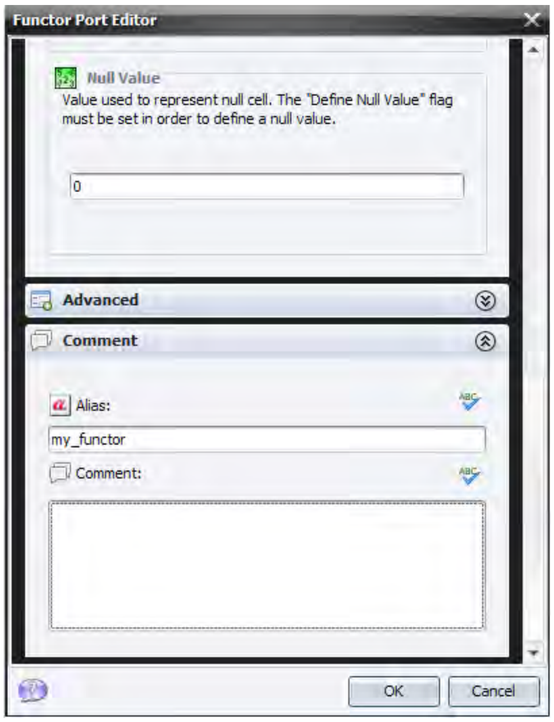
Input | Output

General	
Filename	"C:/users/britald...
Define Null ...	No
Null Value	0
Advanced	
Load As Sp...	No
Suffix Digits	0
Step	

Properties



TIP: You can easily find any functor in the model, typing its name in a text box located at the right top of the interface. You can also perform a search by functor alias or comments.


Functor Port Editor

Null Value
Value used to represent null cell. The "Define Null Value" flag must be set in order to define a null value.

0

Advanced

Comment

Alias:

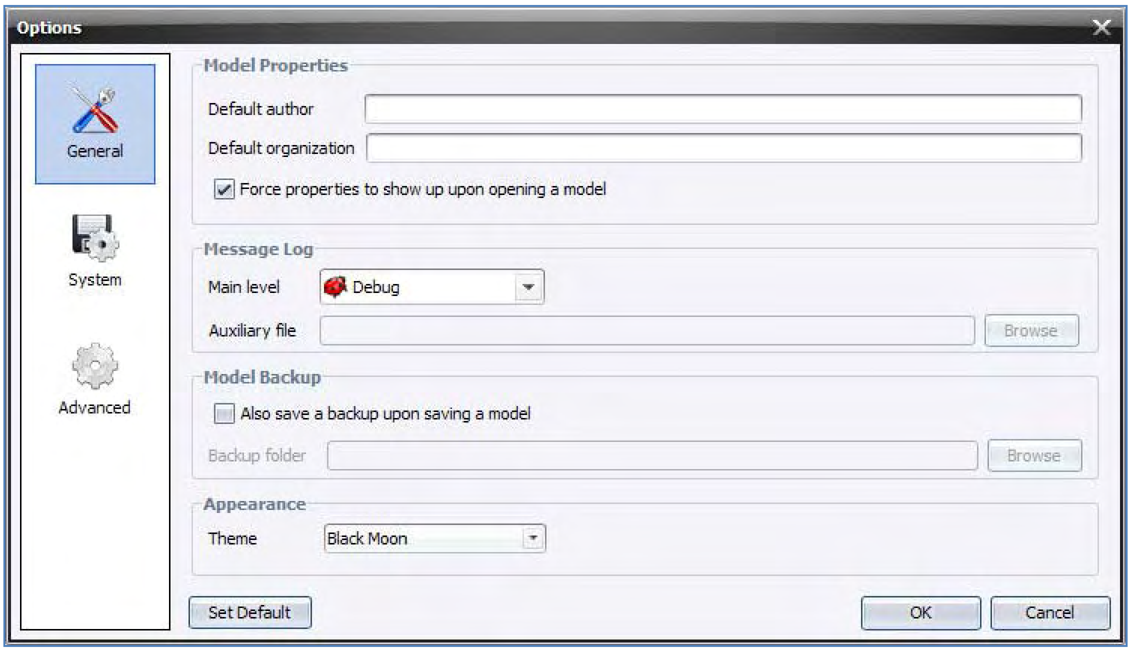
my_functor

Comment:

OK Cancel

Now it is possible to give an alias to a functor in order to facilitate model understanding. Click on any functor with the Edit Functor tool, go to comment tab and type in the alias.

A new option window is available to enter model information and to set Dinamica EGO environment parameters. Go to Main menu Tools – Options:


Options

General | System | Advanced

Model Properties

Default author

Default organization

☑ Force properties to show up upon opening a model

Message Log

Main level: Debug

Auxiliary file — Browse

Model Backup

☐ Also save a backup upon saving a model

Backup folder — Browse

Appearance

Theme: Black Moon

Set Default | OK | Cancel

The option window is divided into 3 tabs. The first tab is for filling in information on the model, setting message log, model backup, and look & feel. The second is for setting environment parameters, such as system backup, auto-recovery and temporary file folder. In advanced options, raster map swapping can be disabled. **TIP:** If you turn off map swapping, raster maps will be kept in virtual memory. Beware that large data models may cause "Allocation failed" and crash thereof.

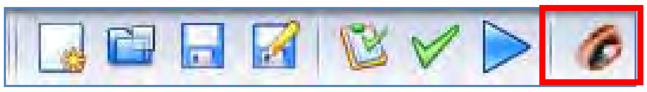
To see the list of functors available in Dinamica EGO, use the Dump Functor list, located at the Help menu. A list of functor is displayed inside the log window. Go also to Help Contents to get a detailed description of each functor and its algorithms.

1.2 Map Viewer

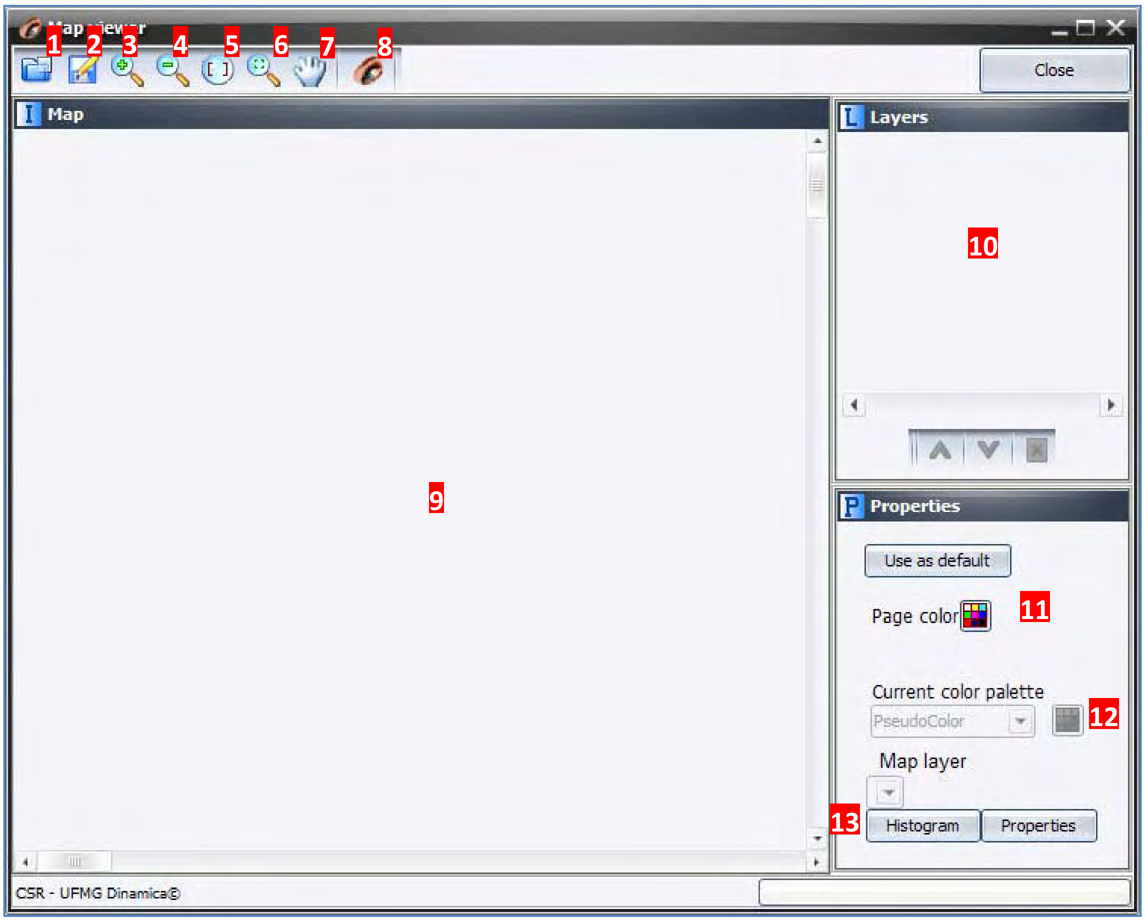
What will you learn?

- Using Map Viewer

Dinamica EGO provides a tool to visualize the input and output maps. Click on Map Viewer on the main tool bar.

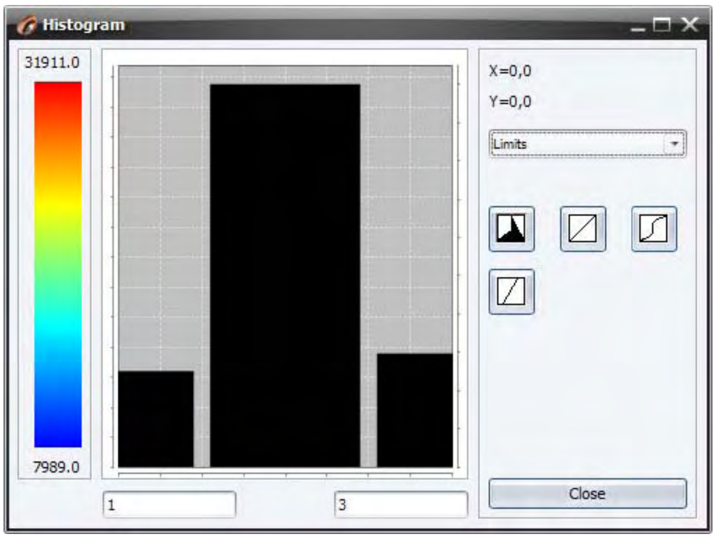


This is what Map Viewer looks like:



1) Open Map
2) Save Map as a New File
3) Zoom in
4) Zoom out

5) Zoom 1:1
6) Fit Map
7) Move Map
8) Open New Map Viewer
9) Map window.
10) Layers: List of layers for cube raster.
11) Change background page color
12) Choose color palette to display a map. **TIP:** you can edit a color palette.
13) Manipulate histogram and show map properties.



For better display, Map Viewer permits a stretch of a map histogram using linear and equalizing transformations.

1.3 Data structures and formats

What will you learn?

- Dinamica EGO data structures
- Dinamica EGO file formats

Dinamica EGO handles data in several formats, including raster maps or images, tables, matrices, and a Weights of Evidence coefficient file.

For spatial data, **Dinamica only supports raster datasets**. Therefore you will need to prepare your dataset in a GIS package and then export the maps in one of the three formats specified below. Although geo-referencing is supported and needed, all raster dataset in a model must have the same number of columns and rows. Moreover they must be tied to the same coordinate space and registration point (fig. 4).

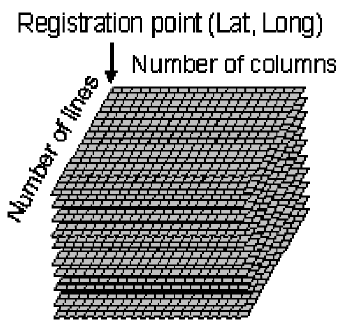
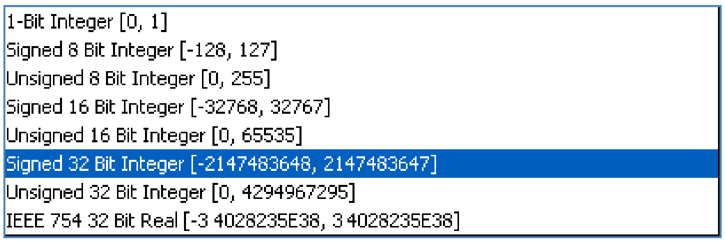


Fig.4 a cube raster dataset

Dinamica EGO reads and writes raster data in three formats: ER Mapper format, Geotiff, and ArcView ASCII. Coordinate systems supported for translation into ERMapper format are Geodetic (WGS 84, SAD69, Corrego Alegre) and UTM (WGS84, SAD 69, Corrego Alegre). When a coordinate system is not supported, LOCAL projection and datum WGS84 are assumed. For ASCII/ArcView, LOCAL projection and datum WGS84 are always assumed. All Geotiff coordinate systems are supported if translation is not needed. When GeoTiff cell dimension is not found, 100 meter resolution is assumed. Geotiff tiling format is also supported, thus you won't find a problem importing and exporting your dataset from and to most common GIS, such as IDRISI, SPRING or ARCGIS 9.*.
TIP: You can find more information on GeoTiff in http://trac.osgeo.org/geotiff/

In map algebra, the null concept is very important to obtain an intelligible result from a model execution. Null means absence of data. Thus a map containing an irregular geographic area of interest, which does not completely cover the geographic plane, must contain a representation for null cell. The value reserved for null cell representation may vary depending on the data cell type, i.e. the size in terms of bits used to store the cell values of a map. Dinamica EGO supports data cell types as follows:


1-Bit Integer [0, 1]
Signed 8 Bit Integer [-128, 127]
Unsigned 8 Bit Integer [0, 255]
Signed 16 Bit Integer [-32768, 32767]
Unsigned 16 Bit Integer [0, 65535]
Signed 32 Bit Integer [-2147483648, 2147483647]
Unsigned 32 Bit Integer [0, 4294967295]
IEEE 754 32 Bit Real [-3 4028235E38, 3 4028235E38]

Usually, the lowest negative value is used to represent the null value. For example -32768 for Signed 16 Bit Integer. **TIP**: Always choose a data cell type able to embrace the range of values contained in a variable. For example: elevation, varying from -10 meters to 4000 meters, must be represented as Signed 16 Bit Integer or IEEE 754 32 Bit Real.
TIP: You may need (and must) in some cases to define the null value when loading a Geotiff dataset, which lacks this definition (highly advised). Learn in lesson 1 how to do this.
Tables are a convenient way to represent attribute data, usually pertaining to a certain geographic zone, for example: country, state or counties. Dinamica EGO can read data in Comma Separated Value format, in which the first column represents the key and the second, the value, as follows.
TIP: The first line of the table must contain the columns titles "Key" and the variable name, such as population, countries, etc.

Key	Value
1	10
2	30
3	15

Transition matrix is also stored using this format; the only difference is that the key employs a composite algorism to represent a transition as follows:

Key	Value
1.002	0.223567
1.003	0.379618
2.001	0.024841
2.003	0.030573
3.002	0.000348

Thus the previous table is equivalent to the following transition matrix: Note that the diagonal values do not need to be filled in, nor are necessary the transitions equal to zero.

	1	2	3
1	-	0.223567	0.379618
2	0.024841	-	0.030573
3	0	0	-

Another supported format is the Weights of Evidence file - a text file containing the Weights of Evidence coefficients. This file is obtained through the Weights of Evidence method used in the calibration process. **TIP:** You can edit this file directly on a text editor or using the Weights of Evidence graphical editor linked to *Load Weights* Functor.

2. Lesson 1: Building a simple model

What will you learn?

- Functors and ports
- How to connect functors
- Writing algebraic and logical expressions
-
- Functors:
  - *Load Map*
  - *Save Map*
  - *Calculate Map*
  - *Number Map*

A solution for a spatial model involves recovering maps from a cartographic model - that is a cube of registered raster maps, in which each map depicts a particular geographic feature -, processing them in an ordered way, and then storing the intermediate results for subsequent processing (fig. 5). To improve performance, the Dinamica EGO processing system only accesses the disk once to read the input maps, if enough memory is available, and then to write the final outputs (fig. 6). Thus a model in Dinamica EGO is represented by a sequence of functors, whose execution takes place from left to right. A simple model consists of a functor that loads the data, for example, a map, another functor that performs some calculation as well as one that writes the result into a file. Note that the functors are connected to each other through arrows. In order to do so, they must exchange compatible data via their outputs and inputs. Dinamica EGO input parameters and data, as well as the outputs from a functor, are called ports. Hence functors are connected via compatible input and output ports. Let's start designing your first model in Dinamica EGO.

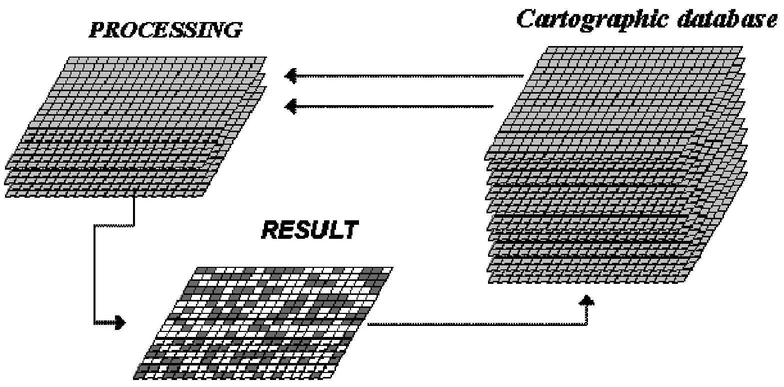


Fig. 5. A traditional sequence of map processing.

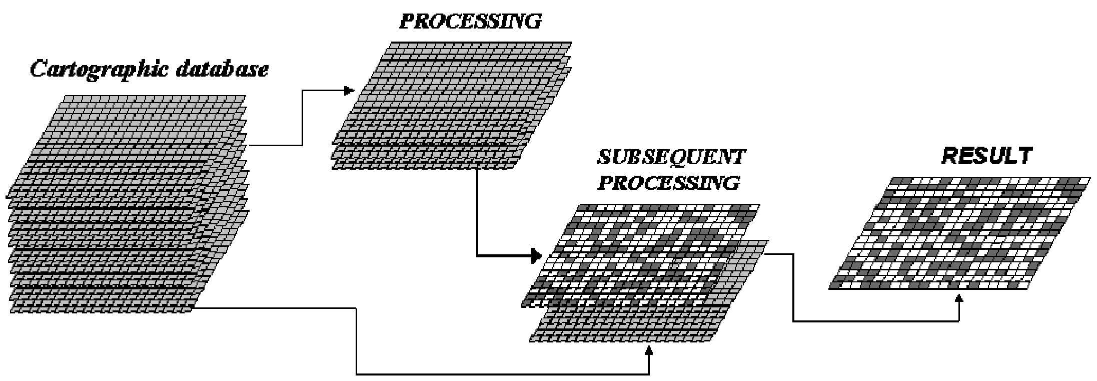
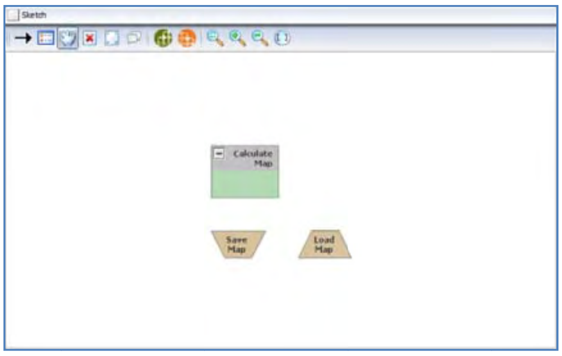
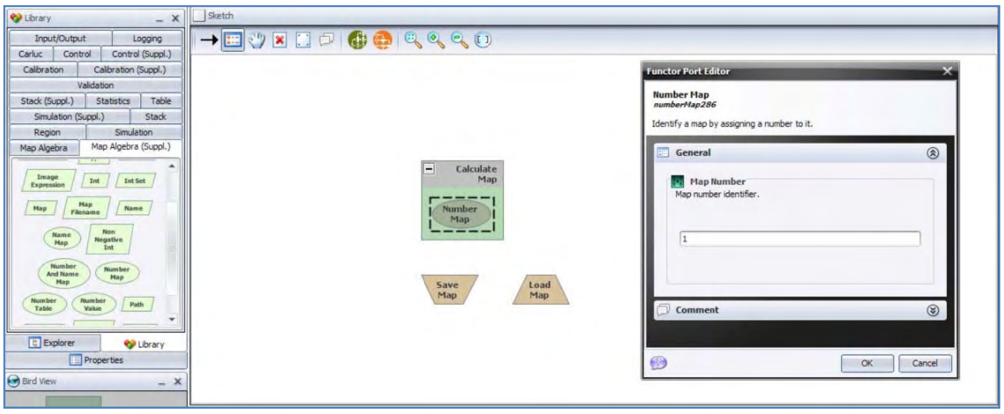


Fig. 6. Dinamica EGO processing system, in this case there is no need to save intermediate results into hard disk files.

From the library window grab and place on the sketch the *Load Map*[1] and *Save Map* functors, which are located in the Input/Output tab. Now grab the container *Calculate Map* from the Map Algebra tab. Remember that a container is a special type of functor that can envelope supplementary functors or even a sequence of functors. The containers differ from a functor by its title bar. **TIP:** By clicking on the left top of the container icon, you can close and open it. You should have something like this:



Now you need to connect the functors to establish a model. First, let's understand what a *Calculate Map* does. This container is a calculator used for map algebra to combine and process maps, tables and constants. As a container, it does not function by itself. There is a need to add supplementary functors to it. In this case, each map processed by this container will be represented by the functor *Number Map* available in the Map Algebra Supplementary tab. Grab *Number Map* and place it inside the container *Calculate Map*. The container will resize to accommodate the *Number Map*.



Functor Port Editor

Number Map
numberMap286

Identify a map by assigning a number to it.

General

Map Number
Map number identifier.

1

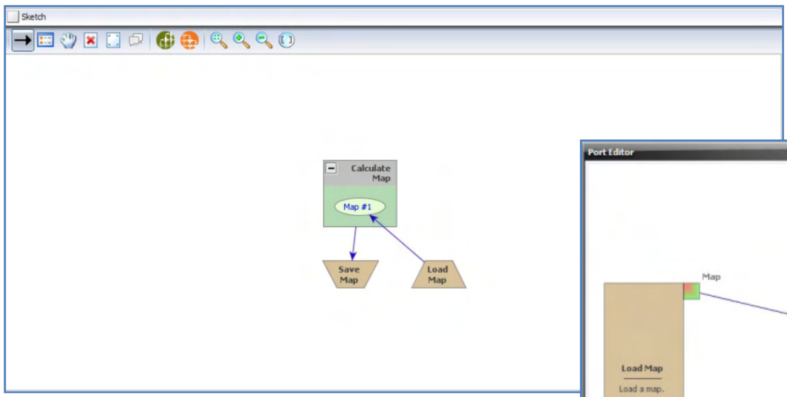
Comment

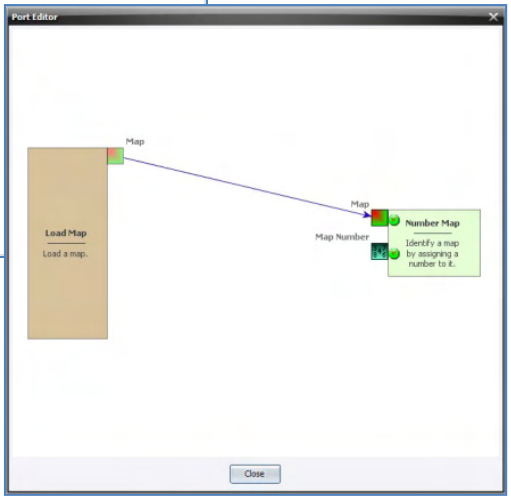
OK Cancel

[1] All functor and containers names are represented in *italic* in this guidebook. Model options and internal parameters are represented in **bold** and data to be entered in quotation marks.

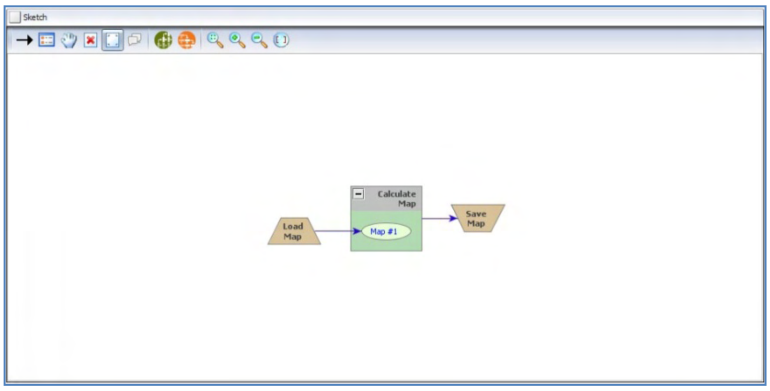
Click on the *Number Map* with the Functor Editor tool and enter “1” (do not write the quotation marks). This is a number identifier for a map and will be represented within the equation box as **i1** (input 1). You can enter other maps by adding more *Number Map* functors, but each one must have a unique number identifier. Now you can connect the functor *Load Map* to the *Number Map* and the container *Calculate Map* to *Save Map*. Use the connect tool (the arrow icon) to establish the connections. Observe that the connection is set automatically because there is only one option for compatible ports in the respective functors and container.

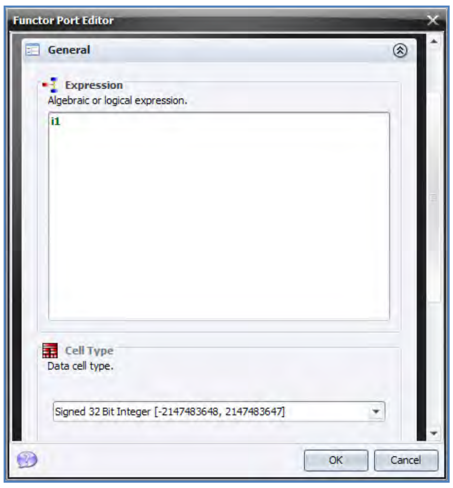
Clicking on the arrow with the Edit Functor Ports tool allows us to visualize the links between a pair of functors.





Now that the functors are linked, you can rearrange the model by clicking on the layout tool. Note that the model will be organized from left to right or left to bottom according to its execution sequence.





Expression
Algebraic or logical expression.
i1

Cell Type
Data cell type.
Signed 32 Bit Integer [-2147483648, 2147483647]

In the next step, open the *Calculate Map* container by grabbing the Edit Functor and clicking on it. Note that map # 1 is represented by **i1**. The *Calculate Map* enables the formulation of various algebraic and logic equations containing maps, tables and constants. The following table presents examples of operators that can be applied to process data within this container. The table is divided into 4 groups: <u>logical</u>, <u>mathematical</u>, <u>table operators</u> and <u>neighborhood operators</u>. Besides writing the equation, there are two parameters that must be set. The data cell type and the null value. You will always find these parameters in functors that produce maps as output. The default is **Signed 32 Bit Integer**, but you can use **IEEE 32 Bit Real** to represent fractional numbers. **TIP:** Try to use always the most economical representation for data cell type to save memory. If you are not sure about the numeric range of the output, use real number representation.

Logical Operators

Operator	Description	Symbol	Usage Example
Boolean Or		or ||	not isNull(i1) or isNull(i2) not isNull(i1) || isNull(i2)
Boolean And		and &&	not isNull(i1) and isNull(i2) not isNull(i1) && isNull(i2)
Equal		= ==	i1 = 2 i1 == 2
Not Equal		!= /= <>	i1 != 2 i1 /= 2 i1 <> 2
Greater Than		>	i1 > 2
Greater Than Or Equal		>=	i1 >= 2
Less Than		<	i1 < i2
Less Than Or Equal		<=	i1 <= i2

Conditional	Execute the second or third term of the equation conditionally to the first	if then else	if not isNull(i1) and isNull(i2) then
			i3
			else if isNull(i1) then
			i1 – i1 / i2
			else
			(i1 / i2) ? (i1 – i2)

Mathematical Operators

Operator	Description	Symbol	Usage Example
Add		+	i1 + i2
Subtract		–	i1 – i1 / i2
Times		*	i1 * i2
Divide		/	i1 / i2
Mod		%	i1 % 100
Power		^	i1 ^ 3
Catch Error	Catch an algebraic error and replace it by the result of another expression	?	(i1 / i2) ? (i1 – i2)
Value			2 + i1 / -3.5e-2
Get Variable Value	Return the variable value	vX where X is an integer value from 1 to 100	v1 + t1[v2 + 4]
Get Image Value		iX where X is an integer value from 1 to 100	i2

Get Image Value At Location	Return the image value on the specified cell coordinate	iX[,] where X is an integer value from 1 to 100	i1[line – 1, column – 2]

Get Image Null Value	Return the null value of the current image	null
		
		null(iX)
			if i1 > 2 then
			  i1
			else
			  null
			
			if null(i2) > 2 then
			  1
			else
			  null

Is Null		isNull(iX)
		where X is an integer value from 1 to 100
			if not isNull(i1) then
			  i1
			else
			  i2

Boolean Not		! not	not isNull(i1) ! isNull(i1)
Get Line Number	Return the line number of the current cell	line	line + 1

Get Column Number	Return the column number of the current cell	column	if column / 2 > 50 then
			  1
			else
			null

Random	Generate a random value using the uniform probability distribution	rand	if rand > 0.5 then
			  i1
			else
			  i2

Negate		–	– ceil(i1 + i2)
Squared Root		sqrt()	sqrt(i1 / i4)
Sin		sin()	sin(i1 / i4)
Cos		cos()	cos(i1 + i2)
Tan		tan()	tan(i1 * i5 + 6)
Acos		acos()	acos(i1 + i2)
Asin		asin()	asin(i1 + i2)
Atan		atan()	atan(i1 + i2)
Ceil		ceil()	ceil(i1 + i2)
Exp		exp()	exp(i1[i1 + i2])
Floor		floor()	floor(i1 + i2)
Round		round()	round(i1 / i4)
Abs		abs()	abs(i1 + i2)
Ln		ln()	ln(i1 / i4)
Log		log()	log(i1 / i4)
Max		max()	max(i1, i2)
Min		min()	min(i1, i4)
Signal	Return +1, if the expression is positive, -1, if it is negative, and 0, otherwise.	signal	signal(i1 - 4)

Abort	Abort the model execution	abort	if i1 > 0 then
			  i1 * i2 + 4
			else
			  abort

Table Operators

Operator	Description	Symbol	Usage Example
Table Operators	Table operators return the value corresponding to a given key according to a rule-operator. It uses the following syntax: tX[N] where: X is a table identifier; N is the rule-operator.		
Get Table Value	Return the table value in the X key position of the table	tX[ ] where X is an integer value from 1 to 100	t2[i1 + 2]
Get Table Equal Lower Bound Value	Return the table value in the greater key lesser than or equal to the X key position of the table.	tX[<=] tX{ } where X is an integer value from 1 to 100	t2[<= 14] t2{14}
Get Table Lower Bound Value	Return the table value in the greater key lesser than the X key position of the table.	tX[<] where X is an integer value from 1 to 100	t2[< i1 + 2]
Get Table Equal Upper Bound Value	Return the table value in the lesser key greater than or equal to the X key position of the table.	tX[>=] where X is an integer value from 1 to 100	t2[>= i1 + i3]
Get Table Upper Bound Value	Return the table value in the lesser key greater than the X key position of the table.	tX[>] where X is an integer value from 1 to 100	t2[> i7]
Get Table Closest Value	Return the table value in the key closest to the X key position of the table.	tX[><] where X is an integer value from 1 to 100	t2[>< 3 + i7]
Get Table Interpoled Value	Return a linear interpoled value drawn through the neighbor keys of the X key position of the table.	tX[/] where X is an integer value from 1 to 100	t2[/ i2]

Neighborhood Operators

Operator	Description	Symbol	Usage Example
Neighborhood Operators	Neighborhood operators return the value of an operation within a defined neighborhood window. It uses the following syntax: nbN(iX, h, w, y, x) where: N is the operator name; X is an image identifier; h is the number of window lines; w is the number of window columns; y is the line where the window center is anchored at the image;		

	x is the column where the window center is anchored at the image. The calculation usually includes the center of the window. Even-sided windows have the center displaced toward the top left corner. The window anchor (y and x) can be omitted when the window center is anchored at the current line and column. The shortened syntax is: nbN(iX, h, w)		
Neighborhood Min	Returns the minimum value of the neighbor non-null cells.	nbMin()	nbMin(i4, 2, 3, line-1, column)
Neighborhood Max	Returns the maximum value of the neighbor non-null cells.	nbMax()	nbMax(i1, 4, 4) - 1
Neighborhood Sum	Returns the sum of the neighbor non-null cells.	nbSum()	nbSum(i3, 5, 5) + 7

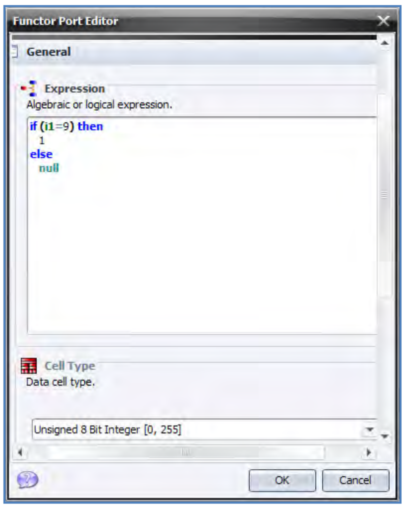
Neighborhood Product	Returns the product of the neighbor non-null cells.	nbProd()	if not isNull(i1) then
			nbProd(i1, 2, 2, 0, column)
			else 0

Neighborhood Count	Returns the number of neighbor non-null cells.	nbCount()	nbCount(i2, 3, 3) + nbCount(i1, 3, 3, line, column+3)
Neighborhood Average	Returns the arithmetic mean of the neighbor non-null cells.	nbAverage()	round(nbAverage(i1, 7, 7))
Neighborhood Median	Returns the median value of the neighbor non-null cells. For an even number of values, the greater of the two median values is returned.	nbMedian()	nbMedian(i1, 5, 5)
Neighborhood Mode	Returns the mode of the neighbor non-null cells. If a mode does not exist, null is returned. If there is more than one, the lesser one is returned.	nbMode()	nbMode(i1, 5, 5)
Neighborhood Variance	Returns the variance of the values of the neighbor non-null cells according to the expression: $s^2 = E_{1,n} (X_i - X')^2/(n-1)$ where $X_1, X_2, ..., X_n$ are the neighbor cells; X' is the mean of the neighbor cells.	nbVar()	nbVar(i4, 7, 7) / 25
Neighborhood Standard Deviation	Returns the standard deviation of the neighbor non-null cells according to the expression: $s = (E_{1,n} (X_i - X')^2/(n-1))^{1/2}$ where $X_1, X_2, ..., X_n$ are the neighbor cells; X' is the mean of the neighbor cells.	nbStdDev()	nbStdDev(i2, 3, 3, line-1, column) + nbStdDev(i2, 3, 3) + nbStdDev(i2, 3, 3, line+1, column)

Now that you have learned the vast possibilities of this container, let's write a simple equation:

if i1 = 9 then 1 else null

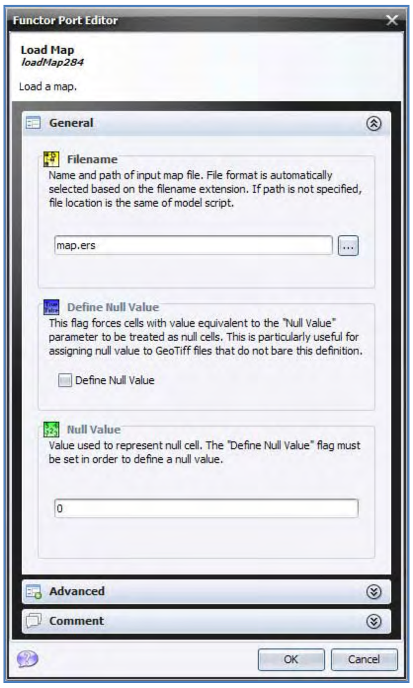
This means: Find the map class equal to 9, and set all others to null. Set the **Data cell type** to **Unsigned 8 Bit Integer** and the **Null Value** to "0". Now you can close this container and open a map file to use as input for this model.



```
if (i1=9) then
  1
else
  null
```

Unsigned 8 Bit Integer [0, 255]

Open the *Load Map* functor and load the file "amazon_states.tif**"** from folder **lesson1** (do not write the quotation marks). You do not need to worry about the other options, they are not needed for now.



Name and path of input map file. File format is automatically selected based on the filename extension. If path is not specified, file location is the same of model script.

map.ers

This flag forces cells with value equivalent to the "Null Value" parameter to be treated as null cells. This is particularly useful for assigning null value to GeoTiff files that do not bare this definition.

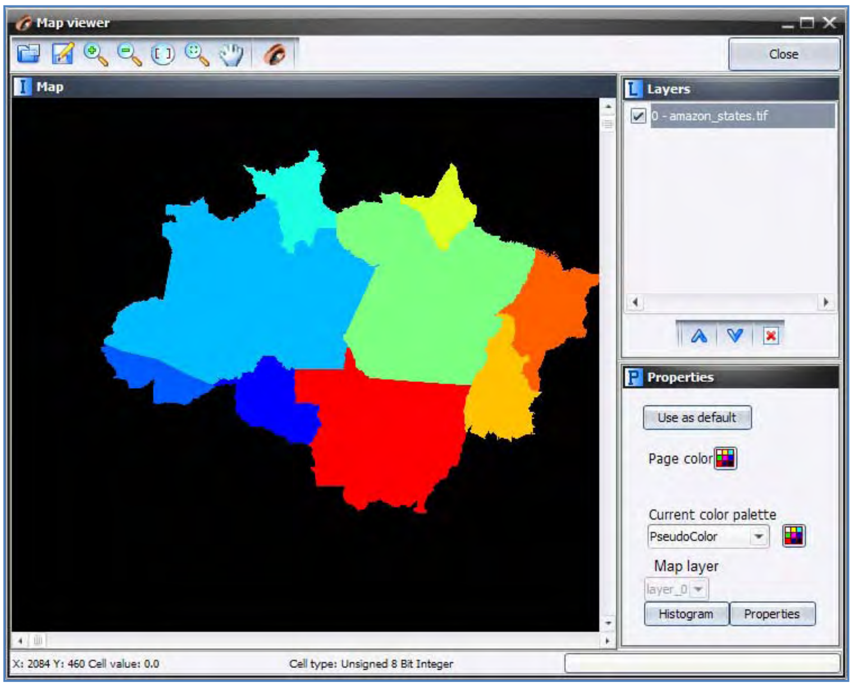
☐ Define Null Value

Value used to represent null cell. The "Define Null Value" flag must be set in order to define a null value.

0

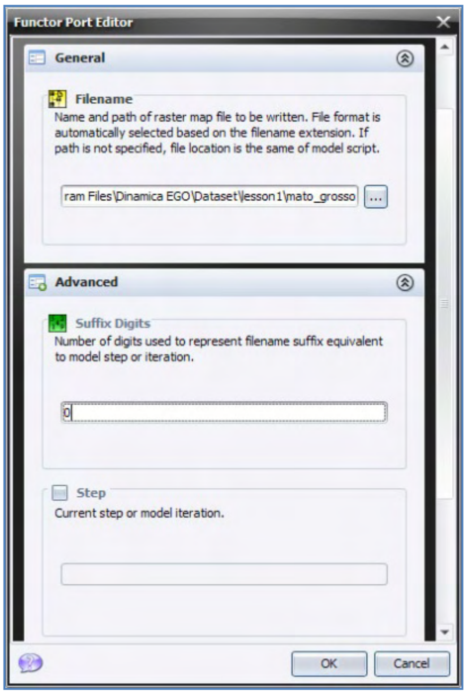
TIP: Although you won't need it in this lesson, you might set the null value by first turning on the **Define Null Value** flag and then setting its value on **Null Value** field to "0".

Open the Map Viewer to visualize this map. Press Histogram and then choose limits to actual and make sure that the current color palette is **PseudoColor**. The map will look like this:



TIP: Limits to actual stretches the map histogram throughout the display range, i.e. from 0 to 255. Use preferentially **PseudoColor** or **GrayScale** for quantitative data, and **Categorical** for categorical data.

Now you need to save the result to a file. Open *Save Map* and write “Mato_Grosso_state”.


Functor Port Editor

General

Filename

Name and path of raster map file to be written. File format is automatically selected based on the filename extension. If path is not specified, file location is the same of model script.

ram Files\Dinamica EGO\Dataset\lesson1\mato_grosso

Advanced

Suffix Digits

Number of digits used to represent filename suffix equivalent to model step or iteration.

0

Step

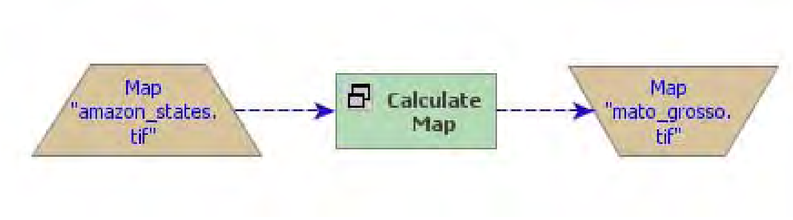
Current step or model iteration.

In the Advanced tab there is an option to set the file name **suffix digits.** Although the default is 2, zero will be assigned since there is only one iteration. Use **Compression is** only applicable for geotiff format.

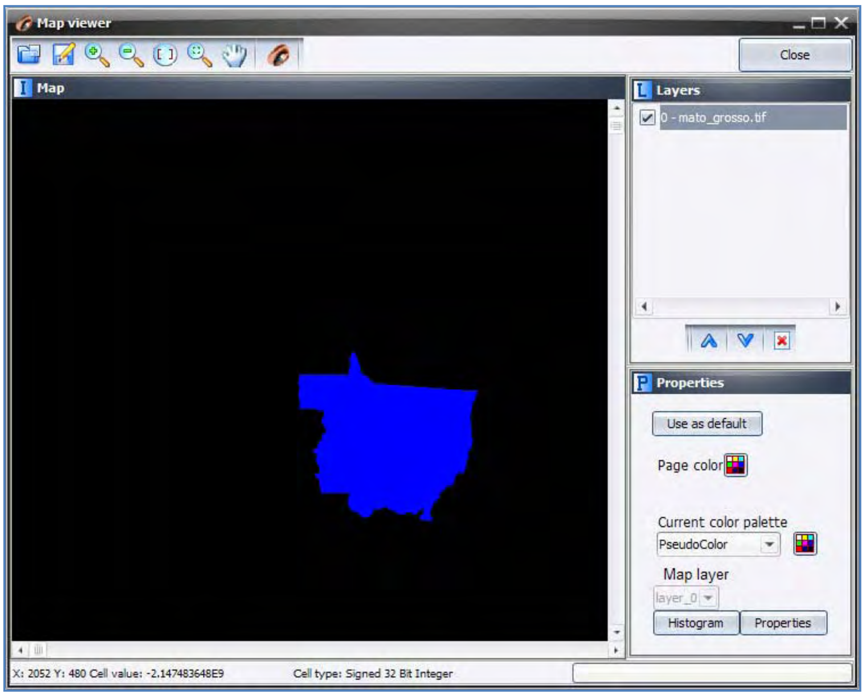
The input map is a categorical map; its values do not represent quantities, but are identifiers to map classes, thus representing categorical data as follows:

Key	State
1	Rondonia
2	Acre
3	Amazonas
4	Roraima
5	Para
6	Amapa
7	Tocantins
8	Maranhao
9	Mato Grosso

Close the *Calculate Map* container by clicking on its bar. The model should look like this:



Save the model as “my_model” in **lesson1** folder, click on **check model script integrity** to check out whether the model is ready to run, and if it is, run the model by clicking on the **run model script** button on the top toolbar and open the result on the Map Viewer. The output map depicts only the Mato Grosso state.



X: 2052 Y: 480 Cell value: -2.147483648E9 Cell type: Signed 32 Bit Integer

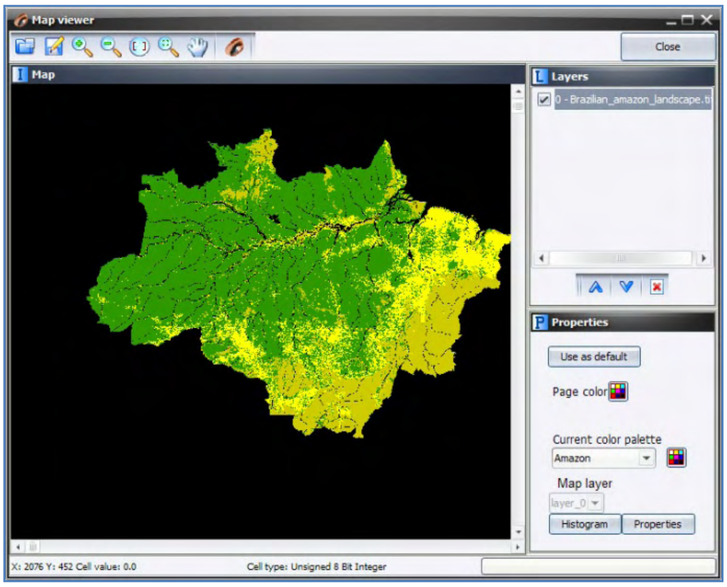
Congratulations, you have successfully completed the first lesson. Now let’s move to lesson 2.

3. Lesson 2: Incorporating iterations into a model

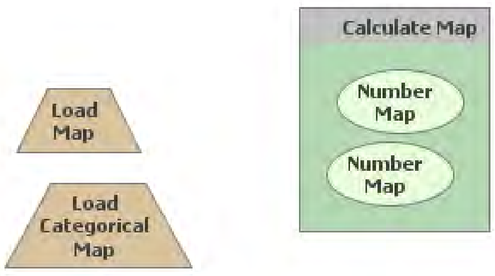
What will you learn?

- Iterations
- Using Register Viewer
- Functors:
    - *Repeat*
    - *Step*
    - *Extract Map Attributes*
    - *Load Categorical Map*
    - *Calculate Value*
    - *Mux Lookup Table*
    - *Set Lookup Table Values*

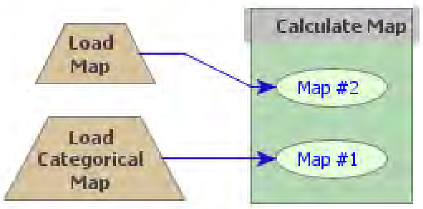
Open the file "brazilian_amazon_ landscape.tif" located in the folder **lesson2** on the Map Viewer. Use **Amazon** as the current color palette. This is a land cover map of the Brazilian Amazon. The map has the following classes: 1 – deforested (yellow), 2 – forest (green), and 3 – non-forest (light brown). The intent of this exercise is to calculate the extent of remaining forest within each state.



Begin placing one *Load Map* and one *Load Categorical Map* on the sketch. This latter functor will categorize a map if it is not categorized, i.e. cell values are identifiers to map classes, so when a map is loaded, it browses the map to identify all unique cell values that represent different categories or map classes, producing as a result a list of classes, which becomes embedded in the map header. Place one *Calculate Map* and two *Number Map* functors inside it.

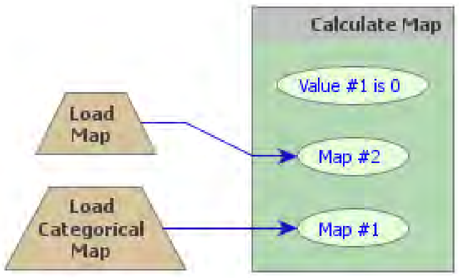


Now assign a number to each *Number Map* and then connect *Load Map* and *Load Categorical Map* to them.

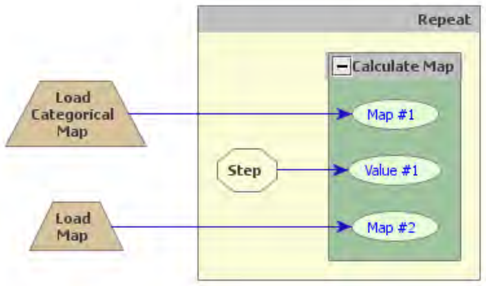


Load "amazon_states.tif" using the *Load Categorical Map* and "Brazilian_amazon_landscape.tif" with *Load Map*. Add a *Number Value* from Map Algebra Supplementary tab inside the *Calculate Map*, assign "1" to it and write the following equation: **if i1 = v1 and i2 = 2 then 1 else null**. **TIP:** you can copy an equation from a text editor and paste it in the equation box.

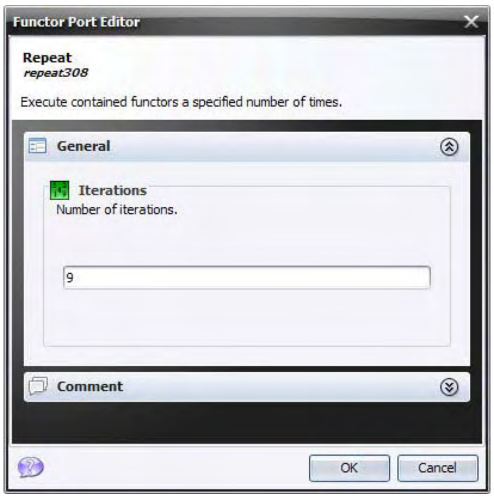
A value is represented by **v# (v1, v2, ..vn)**. Close the container. Do not forget to set the **Data cell type** to "Unsigned 8 Bit Integer" and null value to "0".



Grab the container *Repeat* from the Control tab and place it on the sketch. Drag *Calculate Map* into it. It will automatically resize to envelop *Calculate Map*. Then add the functor *Step* into *Repeat*. Open *Calculate Map* by clicking on its top left icon and connect *Step* to **Value** port of *Number Value*.



Open *Repeat* with the Edit Functor tool and insert "9".



Repeat
repeat308

Execute contained functors a specified number of times.

General

Iterations
Number of iterations.

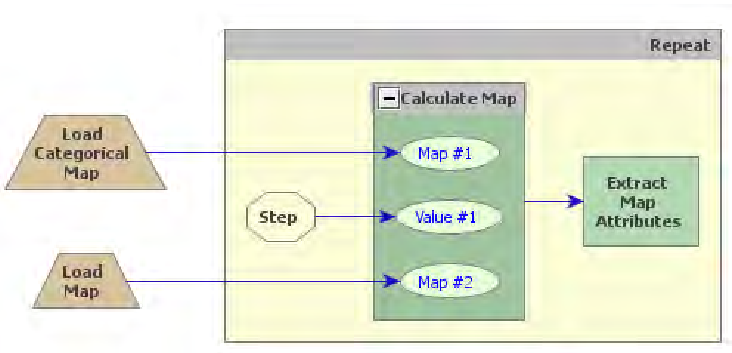
9

Comment

This implies that the model will iterate 9 times. The *Step* functor self-associates to the enveloping container and passes to *Calculate Map* the current step. Thus for each iteration *Calculate Map* produces a map containing the remaining forest (value 2) for each state.

Now you need to sum all cells that represent forest. Note that the output is a binary map with only ones and nulls, the latter is represented by zero. Next, use *Extract Map Attribute* (Map Algebra tab) to extract the sum of non-null cells. This functor produces as output a table with the following map attributes:

Key	**Description**
1	number of lines
2	number of columns
3	number of cells (number of lines multiplied by the number of columns).
4	number of layers
5	cell height (in meters)
6	cell width (in meters)
7	cell area (in hectares)
Dynamic attributes:	
8	number of null cells
9	number of non null cells
10	minimum value (excluding null cells)
11	maximum value (excluding null cells)
12	sum of the values (excluding null cells)
Statistical attributes:	
13	average (excluding null cells)
14	variance (excluding null cells)
15	standard deviation (excluding null cells)

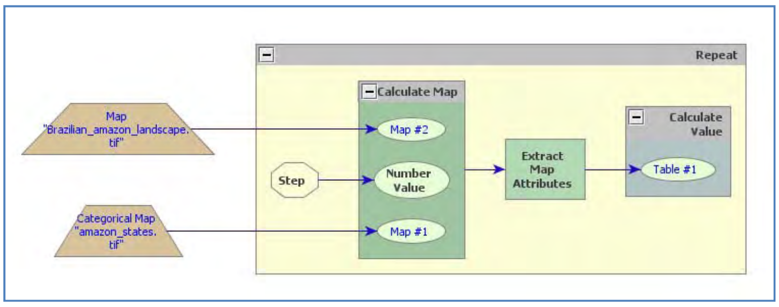


The areal extent is the sum of non-null cells (field 9) multiplied by the cell area in hectares (field 7). You need to add one *Calculate Value,* located in the Table tab, to perform this calculation. Also, drag one *Number Table* into it (from Map Algebra Supplementary tab), which will receive the attribute table output from *Extract Map Attributes*. You need to enter "1" to assign an identifier for this table. Finally, write:

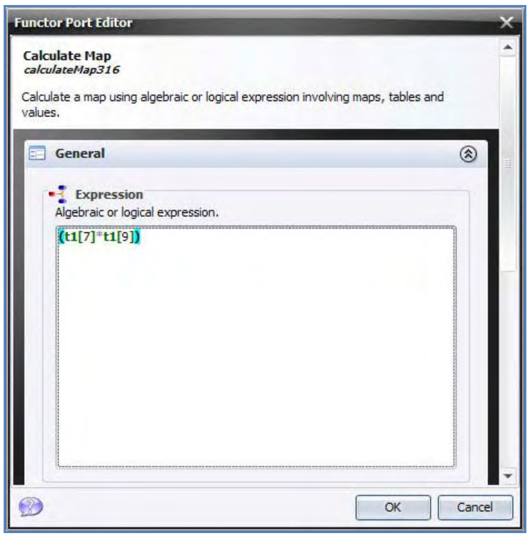
t1[7]*t1[9]

(cell area (in hectares) * sum of the values (excluding null cells))

Note that **t1** means table #1. In this case, the brackets are needed to represent the value associated to a table key.



Now you need to fill in a table in order to store the area calculated for each state. The functor *Set Lookup Table Value* updates a lookup table placing a value to a position defined by a key. To fill in the entire lookup table, you need to develop a loop that enables this functor to browse through the lookup table. To close this loop, you will need a functor that is key to the development of dynamic models. Here we introduce the concept of *Mux* functor.



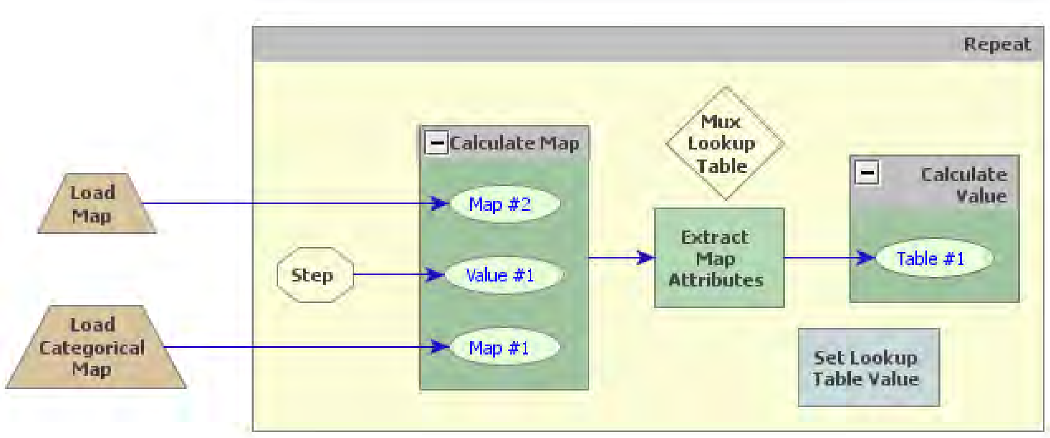
Calculate Map
calculateMap316

Calculate a map using algebraic or logical expression involving maps, tables and values.

Expression
Algebraic or logical expression.

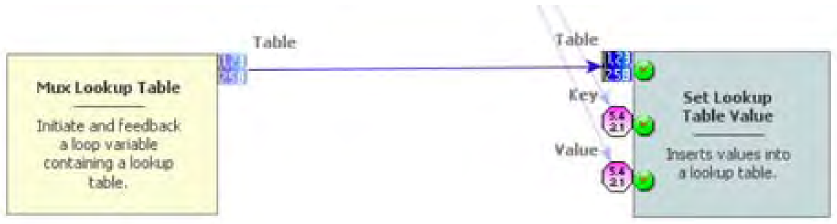
```
(t1[7]*t1[9])
```

A *Mux* functor can be a map, a categorical map, a lookup table, or a value. Look at the Control tab to find *Mux Lookup Table* and drag it into *Repeat*. Also drag *Set Lookup Table Value* from the Table tab.

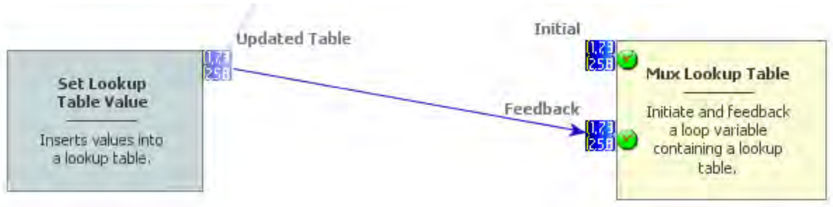


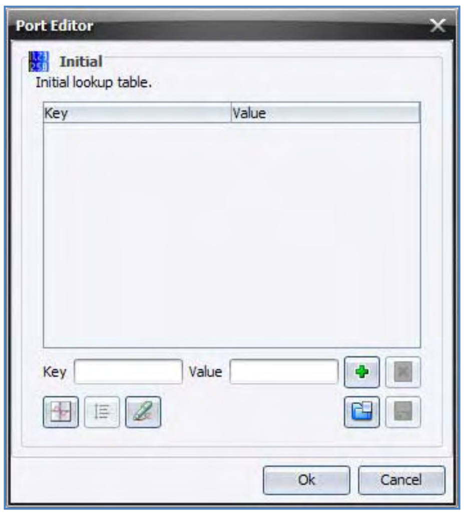
Now, click on the *Mux Lookup Table* with the Edit Functor Ports. Every *Mux* functor has two input ports. In the first iteration, it reads the input from the **Initial** port; thereafter it receives the data from the model step through the port **Feedback**. This process allows data to be updated by the model, thus becoming dynamic. Hence this functor is key to the incorporation of feedback into a dynamic model. Also open *Set Lookup Table Value* with the Edit Functor Ports.

This functor receives a table that will be updated with a value placed in a position defined by a key. So you need to connect table output from the functor *Mux Lookup table* to the input port of *Set Lookup Table Value*.


Mux Lookup Table
Initiate and feedback a loop variable containing a lookup table.
Table
Table
Key
Value
Set Lookup Table Value
Inserts values into a lookup table.

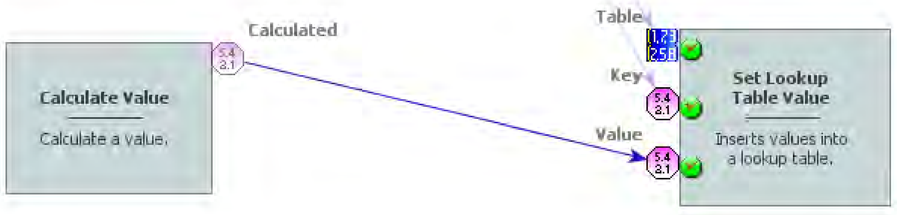
Again, let's connect the output from *Set Lookup Table Value* to *Mux Lookup Table*. When a connection has two or more options, the Edit Functor Ports window opens automatically. You have to choose the port **Feedback**.

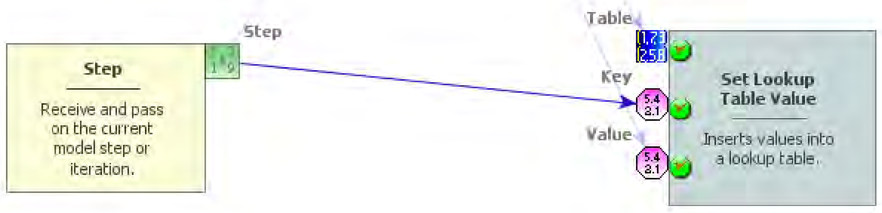

Set Lookup Table Value
Inserts values into a lookup table.
Updated Table
Initial
Feedback
Mux Lookup Table
Initiate and feedback a loop variable containing a lookup table.



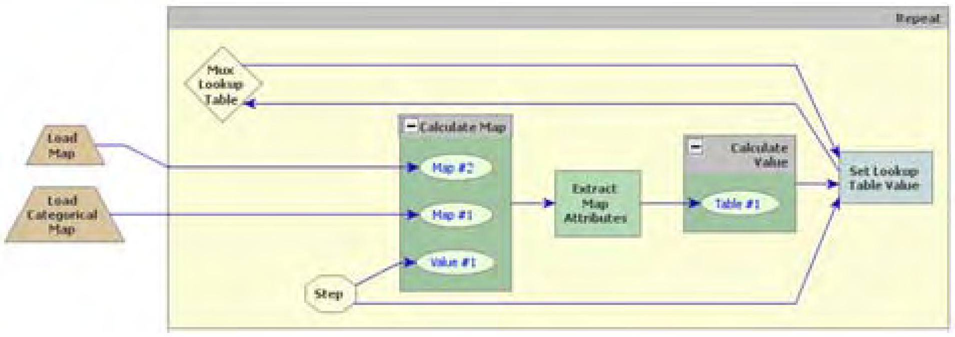
Now click on the port **Initial** with the right mouse button. You will open a table editor. In this case you just need to enter "0, 0" as **Key** and **Value** for the first table record, and then save these inputs using the '+' button.

Still, you need to connect the output from *Calculate Value* to the input port of *Set Lookup Table Value*. The Edit Functor Ports window pops up because there are two options. Connect the arrow to the port **Value**; the key comes from the current model step via the connection of *Step* to *Set Lookup Table Value*.

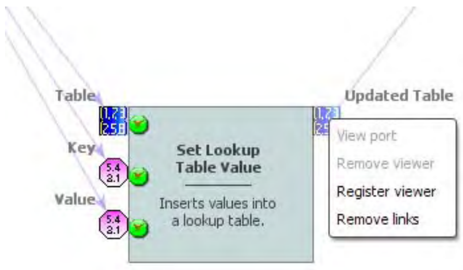



Step
Receive and pass on the current model step or iteration.
Step
Table
Key
Value
Set Lookup Table Value
Inserts values into a lookup table.

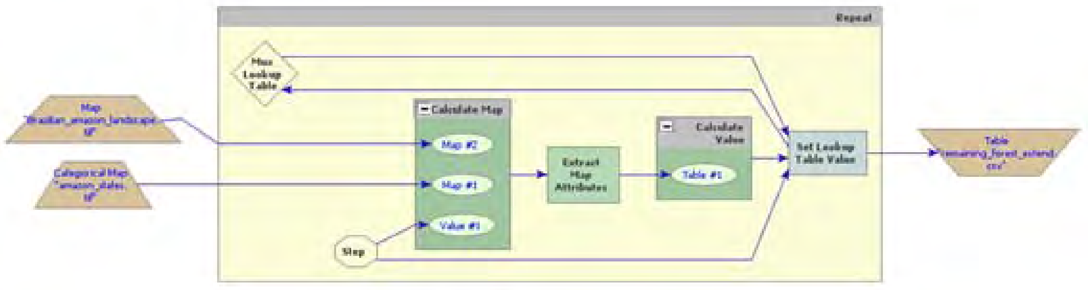
Click on the layout tool Left to Right and your model will look like the one below. Observe that the feedback connection is between *Mux Lookup Table* and *Set Lookup Table Value*. As a last step, you need to save the lookup table into a file. Drag the functor *Save Lookup Table* from the Input/Output tab. Connect *Set Lookup Table* to it and edit the name for the CSV file. Although **Suffix Digits** is “2” by default, the file name won’t have a suffix because it will be saved after *Repeat* is done. **TIP:** If *Save Map* is placed inside *Repeat*, it will save a file per time step and a digit representing the time step will be added to the end of the file name.



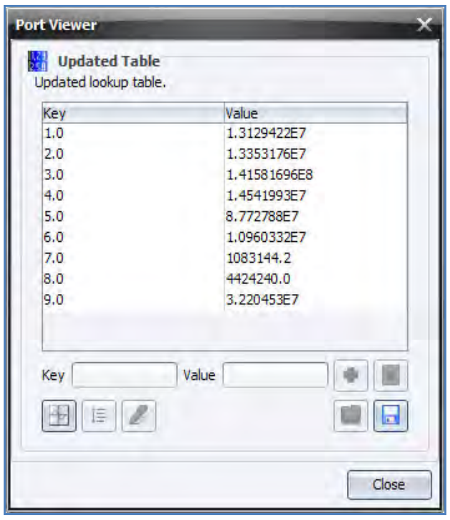
At a last step, open *Set Lookup Table Value* with the Edit Functor Ports. Click on **Updated Table** with the right button and turn on **Register viewer**



Test the model integrity, save it and if everything is O.K., click on the run button. This may take a short while.



Go to *Set Lookup Table Value,* open it with Edit Functor Ports and click with the right button on **Updated Table** to view the result. Also make a chart clicking on the chart button (bottom left).

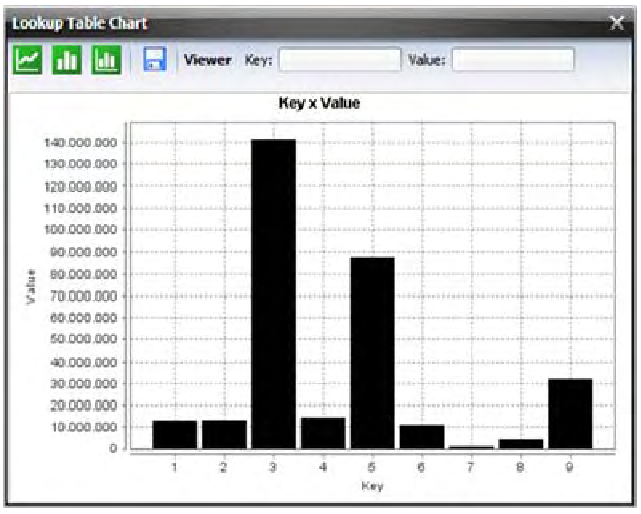

Port Viewer

Updated Table
Updated lookup table.

Key	Value
1.0	1.3129422E7
2.0	1.3353176E7
3.0	1.41581696E8
4.0	1.4541993E7
5.0	8.772788E7
6.0	1.0960332E7
7.0	1083144.2
8.0	4424240.0
9.0	3.220453E7

Key Value

Close



These are the areal extents of the remaining Amazon forest in hectares per state. **TIP:** You can also open the CSV file with a spreadsheet program.

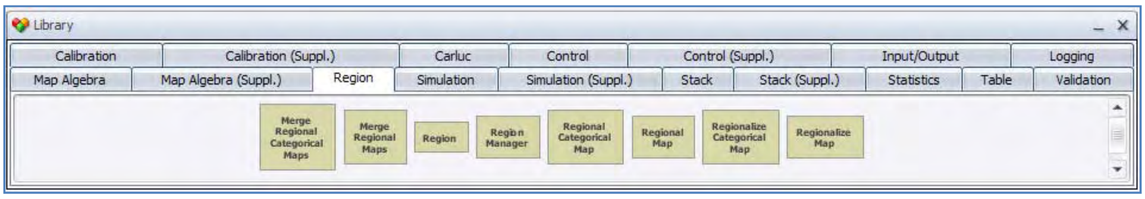
You have successfully completed the two introductory lessons. Another way to solve the query of lesson 2 is through the use of a subset of Region functors. This will be the topic of the next lesson.

4. Lesson 3: Using the concept of region

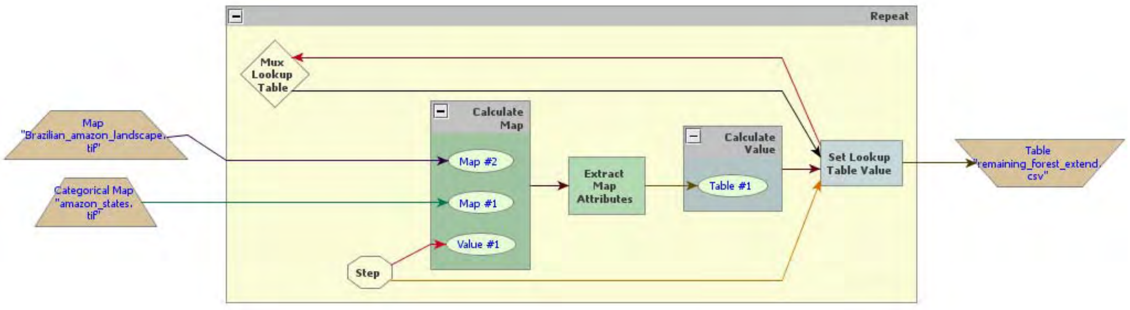
What will you learn?

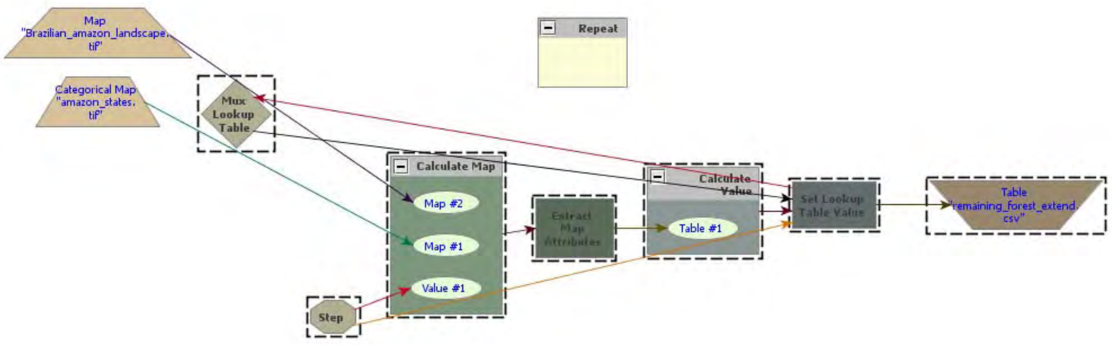
- How to use the concept of Regions
- Functors:
    - o *Region manager*
    - o *For Each Category*

An alternate way to solve the exercise of **lesson 2** is by means of Region. This subset of functors permits division of a map into several regional maps, which can be input for separate models or submodels, whose parameters are customized for each map's region. Let's adapt the previous model to use the concept of regions. First open the Region Tab in the library window.

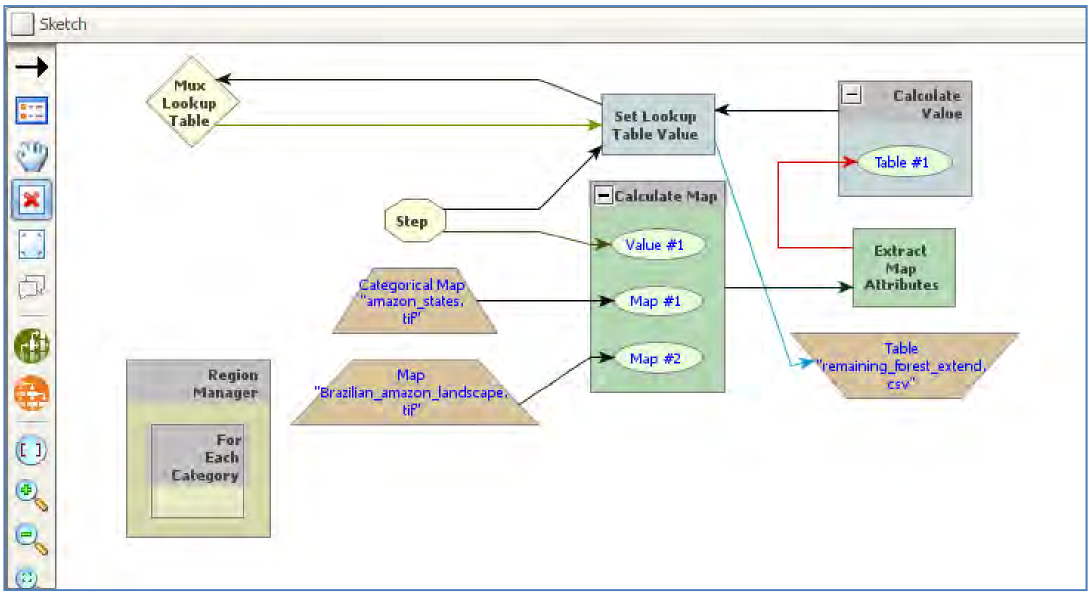


These functors can be combined to produce regional maps and submodels, as well as to merge the regional maps into a single map after a processing is performed. In this lesson you will learn how to retrieve the area of the remaining forest for each state; as a result each regional map will correspond to a state area. Open the model completed in **lesson 2**. Select the functors within the container *Repeat* with the Hand tool and then drag them out of *Repeat*. **TIP:** Use **Crtl + right button** to keep continually selecting the functors, this will avoid selecting *Repeat* as well. The functor *Repeat* will shrink.





Delete the container *Repeat* and place the containers *Region Manager* from the Region tab and *For Each Category* from the Control tab. Place the latter within the first.

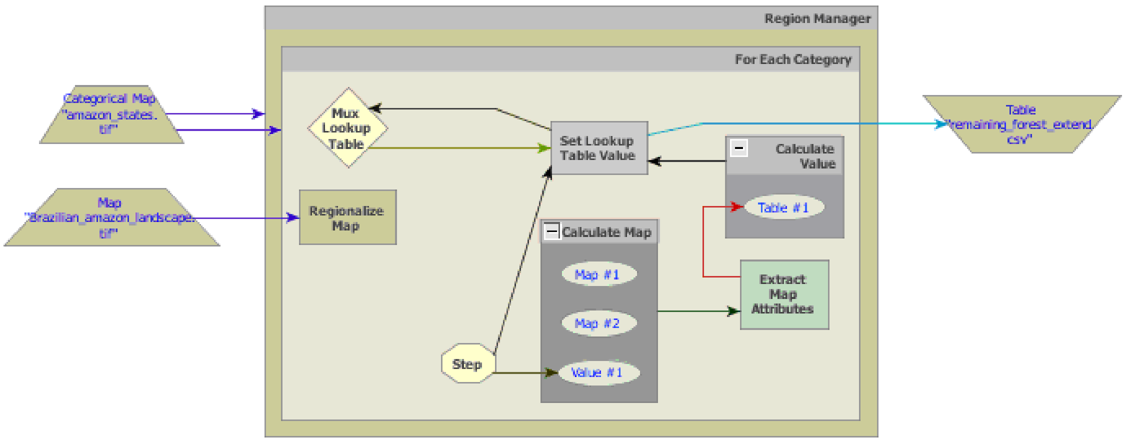


Region Manager manages the regionalization process, while *For Each Category* replaces the *Repeat* making the model iterate for each category present in the categorical map used to define the regions, in this case the map of the Brazilian Amazon states.

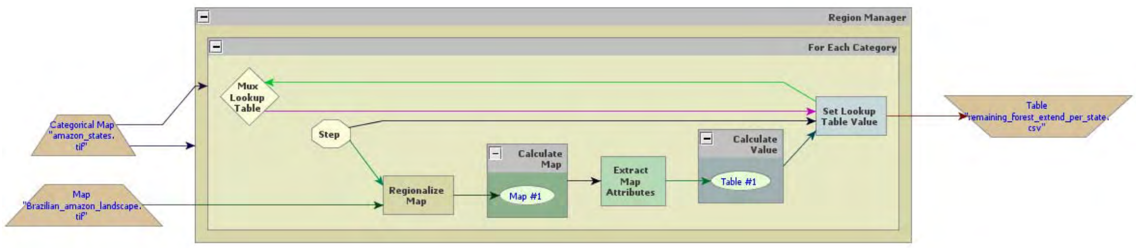
TIP: While Repeat iterates sequentially, *For Each Category* iterates according to the map categories, which do not need to be sequential.

Now select all functors, except the input and output maps, and place them within *For Each Category.* **TIP:** it is easier to select everything to be placed within *For Each Category* and then to take only *Load Map* and *Save Map* out of it.

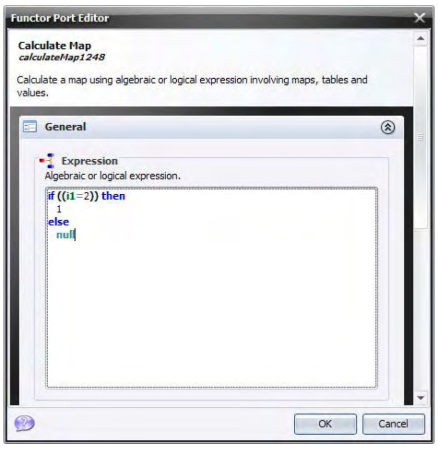
Now break the link between *Categorical Map* and *Calculate Map* (use the Remove Items tool and click on the arrow connecting both functors); next, connect the first to *Region Manager* and to *For Each Category* too. This map will control the regionalization process splitting the other maps into several regions according to its cell classes or categories through the functor *Regionalize Map*. Drag this functor and place it within *For Each Category,* break the link between *Map* “brazilian_amazon_lanscape.tif” and *Calculate Map* and finally connect the first to *Regionalize Map*.



Note that there are two *Number Map* functors disconnected within *Calculate Map*. You do not need the *Map # 2* and *Value # 1* anymore, so delete them. Now connect *Regionalize Map* to *Map # 1.* Open the *Table* functor and change the file name to “remaining_forest_extent_per_state”. Make sure it will be saved into folder **lesson3.** You will need to connect *Step* to *Regionalize Map* allowing it to pass to the first the current region identification coming from *For Each Category*.



At a last step, you will need to modify the *Calculate Map*. Open it with Edit Functor. Write: **if i1 = 2 then 1 else null**. Remember that “2” represents forest.



Calculate Map
calculateMap1248

Calculate a map using algebraic or logical expression involving maps, tables and values.

Expression
Algebraic or logical expression.

```
if ((i1=2)) then
  1
else
  null
```

Finally save the model as a new file "my_calc_forest_remaining_per_state_using_subregions.xml" into folder **lesson3.**

Test the model integrity, and if everything is O.K., click on the run button. Compare this model's output with the result from the previous lesson. Do they match? Can you explain how this model works? Observe that you did not need to segregate the information per state in *Calculate Map* because *Regionalize Map* already did it. **TIP:** the region approach is a helpful way to break the map into several subsets either to customize a submodel, for example to run with different parameters per country, state or county, or to optimize the memory usage as the model does not need to handle all the map cells for a specific calculation or processing, but only the cells selected from a region per time. Now let's move on to more advanced spatial analyses.

5. Lesson 4: Calculating accumulated cost surface and least-cost pathway

What will you learn?

- How to calculate a friction surface, cost surface and least-cost pathway
- Functors:
  - *Calc Cost map*
  - *Calc Pathway map*

This exercise requires the calculation of a friction surface, representing the relative cost of crossing a unit cell depending on the land use. We can express this surface both in terms of distance – no differential cost exits among types of land uses; hence the least-cost pathway will be the shortest route, i.e. the Euclidian distance – time, financial cost or some type of effort. Thus, this value is calculated in relation to some unit (time, transportation cost, etc).

In this exercise, we explore the use of the functors *Calc Cost Map* and *Calc Pathway Map.* To find an optimum solution for the accumulated cost surface, the functor *Calc Cost Map* uses a heuristic algorithm that recursively brushes a map until the best cost surface is obtained. As the number of passes increase so does an approximation of an optimum solution. Use "0" for an optimum solution, but in general, only two passes are sufficient to obtain a surface very close to the optimum solution.

Question:

We want to define the least-cost pathway for a railway that will connect an existing railroad to a town located in the region. From the engineers' point of view, it must be the least expensive, i.e. the shortest, but some land use cannot be converted to open space for the rails, thus representing barriers, and others have a very high cost to cross. Our task is to determine the least-pathway between the town and the existing railroad.

The dataset employed in this exercise comprises:

1. A land use and cover map of Northern Mato Grosso region, Brazilian Amazon (fig. 7). (landuse.tif)
2. A slope map. (slope.tif)
3. A map with the town to be connected. (town1.tif)
4. A map with the current railroad (railroad.tif)

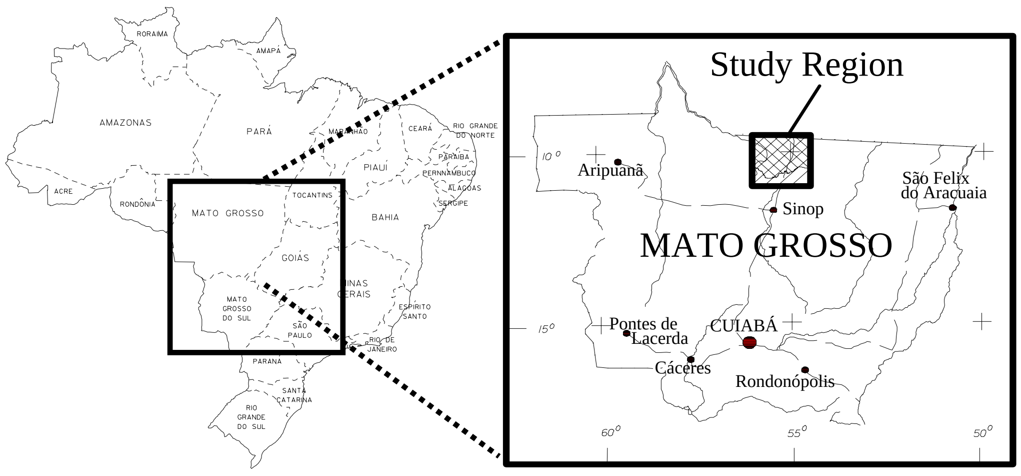


Fig. 7 – Northern Mato Grosso and its location with respect to Brazil.

Open Dinamica EGO and load the aforementioned maps from **lesson4\originals** on the Map Viewer using the color table "mt". Open the slope map using "Pseudocolor" as the current color palette and in the Histogram click on Limits to Actual and Histogram Equalize. As a first step, you need to reclassify the land use map to depict the cost of crossing each one of its land uses. Also you need to reclassify the slope map and then combine it with the land use map. For the land use map use the following table:

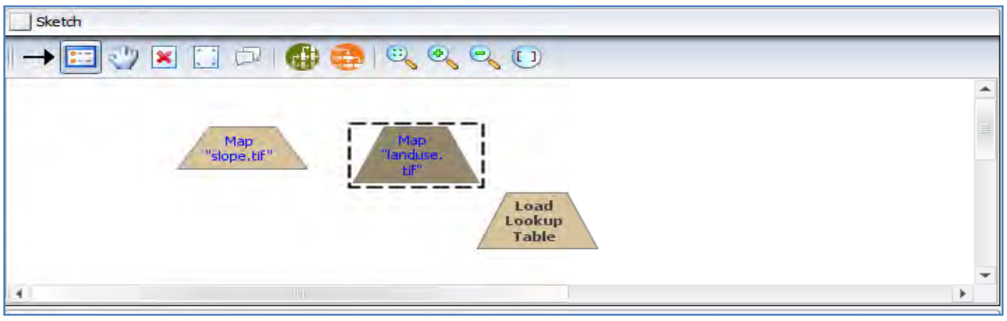
Land use	**Attribute**	**friction**	**Explanation**
Flooded plains	0	10000	Virtually a barrier
Small Rivers	1	50	There is a need to build bridges
Pasture	2	1	The basic cost
Regrowth	3	10	It is necessary to clear cut the bushes and small trees
Forest Remnant	4	500	It is necessary to obtain a license to suppress the native forest, which has an intrinsic conservation value
Urban	5	10000	Virtually a barrier
Roads	6	30	It is necessary to build overpasses or install signs to halt the traffic

Friction increases as a function of slope ranges as follows:

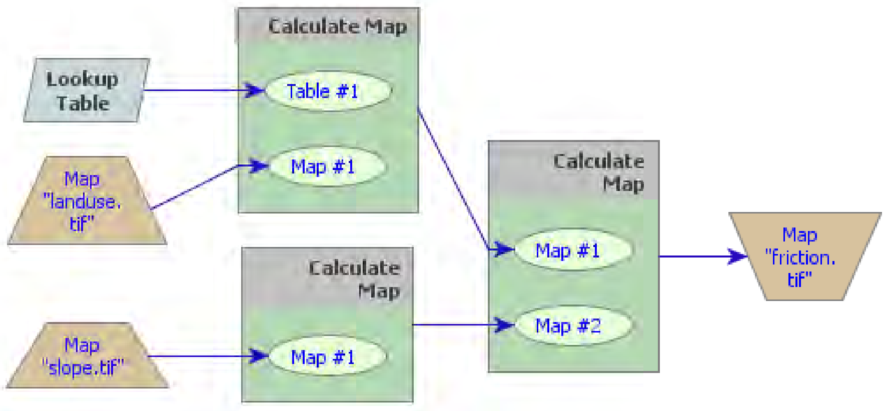
Slope (degrees)	**Friction**
0-1	1
2-5	1.3
5-10	1.5
10-15	1.9
15-20	2.5
> 20	5

Let's begin the model by loading the maps "landuse.tif" and "slope.tif" using the functor *Load Map*. Then, let's incorporate the two previous tables. Add a *Lookup Table* from Table tab.

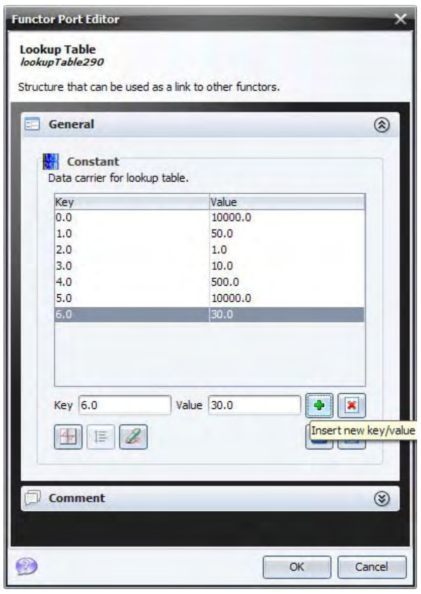
You should have something like this.



Now place three *Calculate Map* and four *Number Map* functors, one within each *Calculate Map and two Number Map* functors within the third, and one *Number Table* within one of the two first *Calculate Map* functors and *Save Map*. Open *Number Map*, assign a unique number (1 and 2) to each one and assign "1" to *Number Table*. Finally connect *Map* "landuse.tif" to *Number Map 1* and *Map "*slope.tif*"* to *Number Map 2* of the two first *Calculate Map.* Then, connect the two first to the third *Calculate Map and it to Save Map*. Open *Save Map* and enter "friction.tif". First change the folder to an upper level, change the file format to "geotiff". This is what you get.



Now, you need to enter the land use table in the functor *Lookup Table*. Open it with Edit Functor and start adding keys and values (fill in the fields and press the plus button).Open the *Calculate Map* that contains the *Number Table* and write the following formula: **t1[i1]**



Functor Port Editor

Lookup Table
lookupTable290

Structure that can be used as a link to other functors.

General

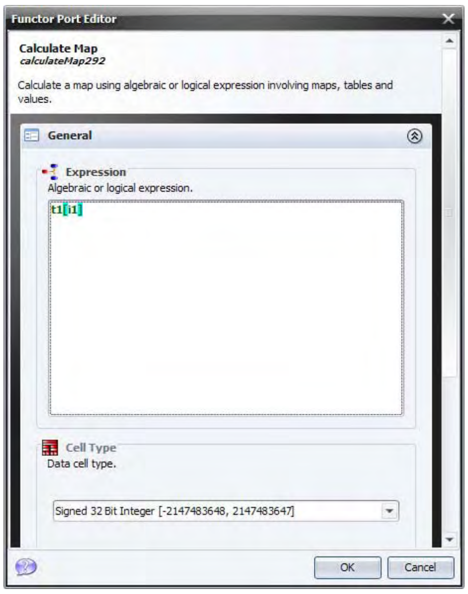
Constant
Data carrier for lookup table.

Key	Value
0.0	10000.0
1.0	50.0
2.0	1.0
3.0	10.0
4.0	500.0
5.0	10000.0
6.0	30.0

Key 6.0 Value 30.0

Insert new key/value

Comment



Functor Port Editor

Calculate Map
calculateMap292

Calculate a map using algebraic or logical expression involving maps, tables and values.

General

Expression
Algebraic or logical expression.

```
t1[i1]
```

Cell Type
Data cell type.

Signed 32 Bit Integer [-2147483648, 2147483647]

This formula will get the value from the map and use it as a key to access the table, therefore reclassifying the map according to the cost values.

Enter the following equation into the second *Calculate Map:*

if i1 < 1 then 1 else if i1 < 5 then 1.3 else if i1 < 10 then 1.5 else if i1 < 15 then 1.9 else if i1 < 20 then 2.5 else 5

This corresponds to the slope friction table.

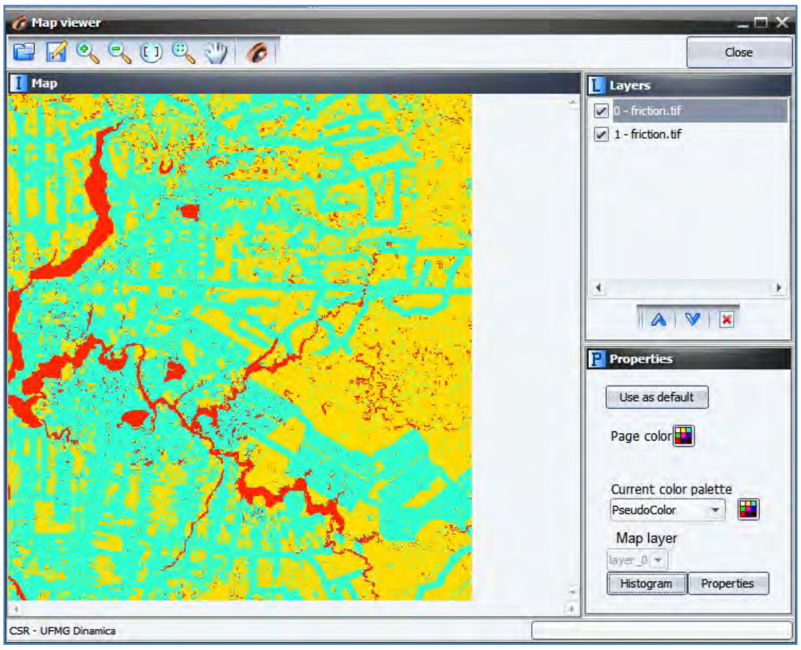
In the third *Calculate Map* enter:

i1*i2

Save the model as "my_friction", verify its integrity and if it is O.K., run it.

Open on the Map Viewer "friction.tif*",* using "Pseudocolor" as **Current Color Palette** and in the Histogram click on **Limits to Actual** and **Histogram Equalize**.

What do you see? Observe the red color representing the areas with high costs to cross. **TIP:** You may feel free to use other software to view the maps. Geotiff is automatically opened in ARC GIS, ER MAPPER or ERDAS packages.



Let's move on to the second part of this exercise. Load "town1.tif" from \originals using *Load Map* and "railroad.tif" using *Load Categorical Map*. Remember that this functor categorizes a map.

Drag *Calc Cost Map* and *Calc Pathway Map* from the Map Algebra tab and *Save Map*.

Here are some notes on this algorithm:

The algorithm that calculates the cost map is a general type of "Pushbroom". However its spatial performance approximates the so called "Pushgrow" algorithm, especially when using two or more passes.

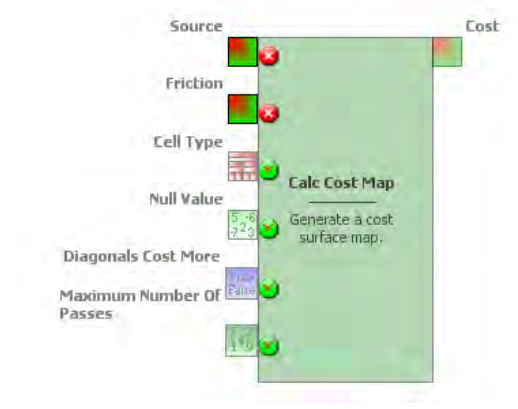
By default, the cells dimensions (width and height) are not considered in the calculation of cost. To take this into consideration, turn on in Advanced options **Friction is relative**.

Penalization for diagonal movements is effective only when friction values are high or the cost map is represented using real numbers.

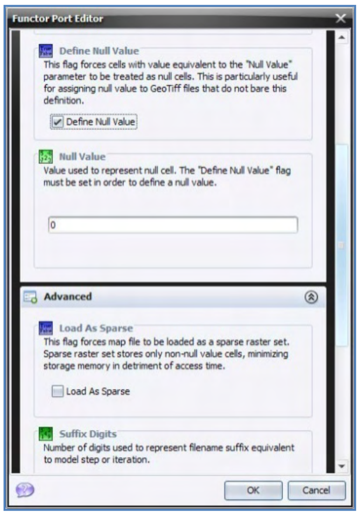
Friction map with cells represented in real numbers requires cost map with cells represented in real numbers or an error will be reported.

Each pass used to calculate the cost map corresponds to four map passes originating from opposite directions.

Unreachable places on the friction map are excluded on the cost map and thus represented as null value cells. Cost are not accumulated across null value cells, thus regions surrounded by null value cells will not have their cell costs calculated, unless there is a source feature inside this region.



The **Source** port will receive the Map "railroad.tif" and the friction map output by the third *Calculate Map*. Turn on **Diagonals Cost More**. This will penalize the movement across diagonal cells. Set **Maximum Number of Passes** to "2". Leave all other options untouched.


Functor Port Editor

Define Null Value
This flag forces cells with value equivalent to the "Null Value" parameter to be treated as null cells. This is particularly useful for assigning null value to GeoTiff files that do not bare this definition.

[✓ Define Null Value]

Null Value
Value used to represent null cell. The "Define Null Value" flag must be set in order to define a null value.

0

Advanced

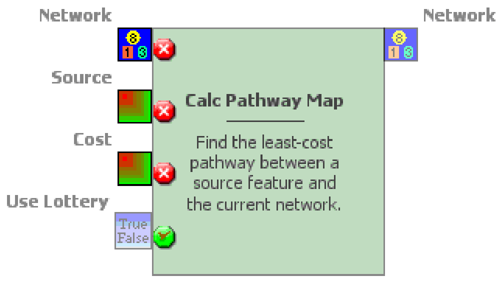
Load As Sparse
This flag forces map file to be loaded as a sparse raster set. Sparse raster set stores only non-null value cells, minimizing storage memory in detriment of access time.

☐ Load As Sparse

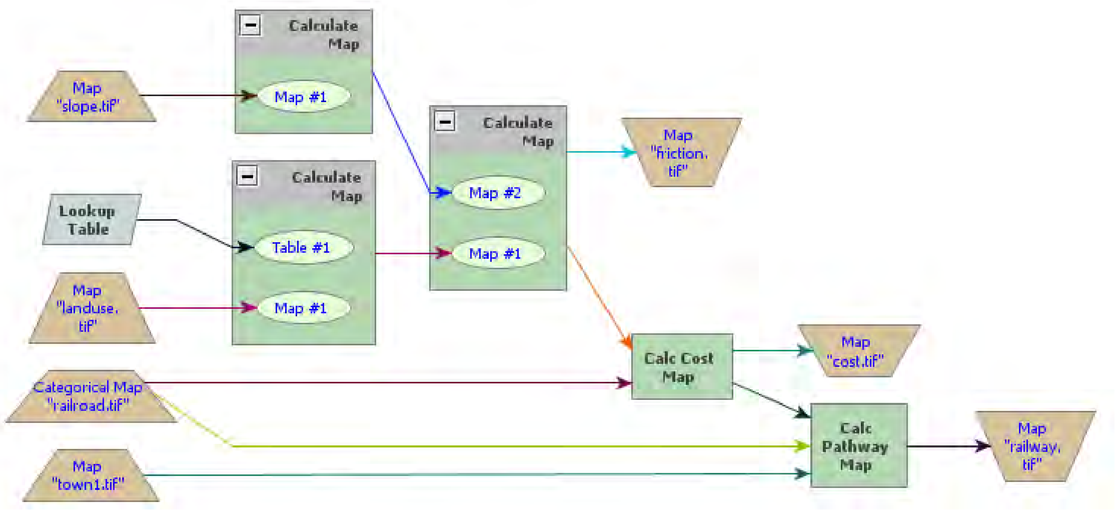
Suffix Digits
Number of digits used to represent filename suffix equivalent to model step or iteration.

Open *Calc Pathway Map* now with the Edit Functor Ports. Link *Map* "town1.tif" to the Source port (**TIP**: Source, in this case, also represents the destiny since the cost map was built from the existing railroad. Thus, this algorithm will search for the least-cost pathway from the source to the existing feature, i.e. the railroad), link the map output from Calc Cost Map to the port **Cost** and Map railroad.tif to **Network** (because it represents a linear feature network) and the output **Network** port to *Save Map*. Activate the option **Use Lottery** (this is an artifact that permits the model to solve the path when two or more local minima are found).

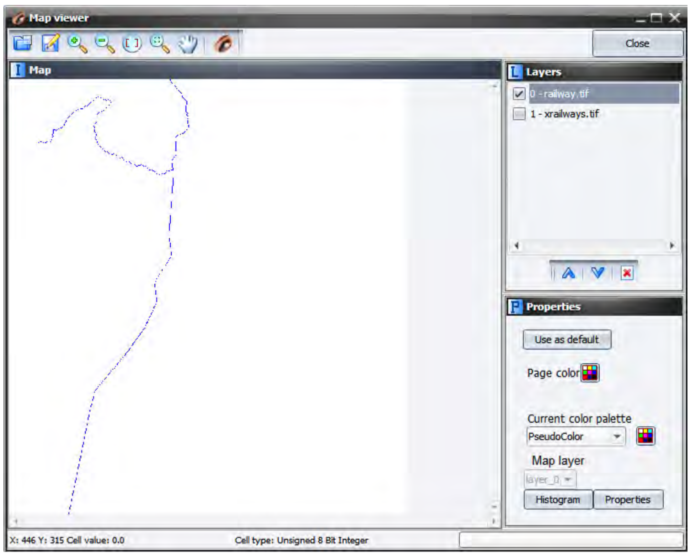
Calc Pathway Map ignores cells with values equal or lesser than 0 or null cells. In turn *Calc Cost Map* needs a network map with null cells representing non-features. Go to *Categorical Map* and open it with the Edit Functor. Press the flag **Define Null Value** and make sure **Null Value** is set to "0".


Network
Source
Cost
Use Lottery
True
False
Calc Pathway Map
Find the least-cost pathway between a source feature and the current network.
Network

Click on *Save Map* with the Edit Functor, change the folder to an upper level, change the file format to "geotiff" and set **Suffix to Digits** to "0", finally enter "railway.tif". The final model will look as follows:

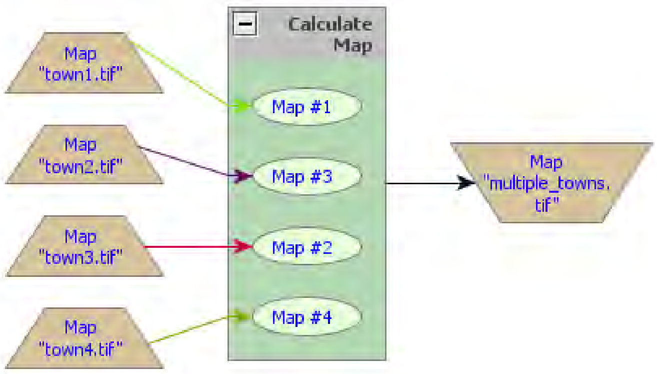


Save the model to a new file "my_pathway.xml", verify it and if it is O.K., run it. This is going to take only a little while. Dinamica EGO has superior performance in relation to most commercial GIS packages; you may want to try this model on other software just for performance comparison. Open on the Map viewer "railway.tif", using "PseudoColor" as **Current Color Palette**. What do you see?



You may try to maximize the solution for the *Calc Cost Map* algorithm by setting the **Maximum Number of Passes** to "0". Compare the time spent by this run and its resulting path with that of previous model? Did it make a big difference?

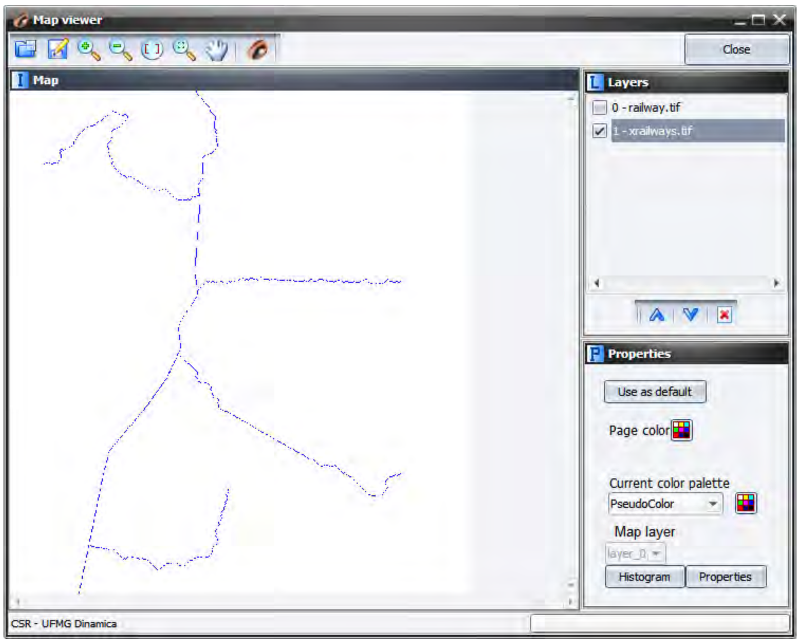
This type of model can also be modified to develop simultaneously multiple paths. Open the model "join_towns.xml" in **lesson 4** folder.



This model shows how you can use *Calculate Map* to merge information from several maps into a single one. The product will be a map depicting the center cells for four towns. **TIP:** use always a sole cell to represent a location to be reached by *Calc Pathway Map*.

Now replace the input in *Map "*town1.tif" with the file "multiple_towns.tif" and change the file in Map "railway.tif" to "xrailways.tif".

Did you get something like this?



If you go to Examples\run_lucc_northern_mato_grosso\run_roads_with_comments and open the model "mato_grosso_road.xml", you will see how this set of algorithms can be adapted and combined to build a Road Constructor Module, a submodel that simulates the expansion of the road network in an Amazonian frontier region. This model is an example of the ability of Dinamica EGO platform for the ingenious design of spatial models.

6. Lesson 5: Multiple Criteria Evaluation in Dinamica EGO

What will you learn?

- MCE for urban and regional planning.
- Functors:
  - *Calculate Distance to Feature Map*
  - *Group*

Multiple Criteria Evaluation (MCE) is a method often used for environmental impact assessments or urban and regional planning. In this exercise you will apply MCE to identify areas that are favorable to develop a new town in Northern Mato Grosso – an Amazonian frontier region –, mitigating at the same time possible environmental impacts from urban settlement. Again, we confront two points of views, one from developers and another from conservationists. A way to solve this question is to list all criteria that favor the location of a future urban site and the ones that constrain or impede it.

In this problem, we have two types of criteria, the ones that constrain our analysis only to specific areas, thus they are Boolean, either 1 or 0, and the others that assign a degree of suitability for a site depending on its biophysical and infrastructure attributes.

Our criteria are:

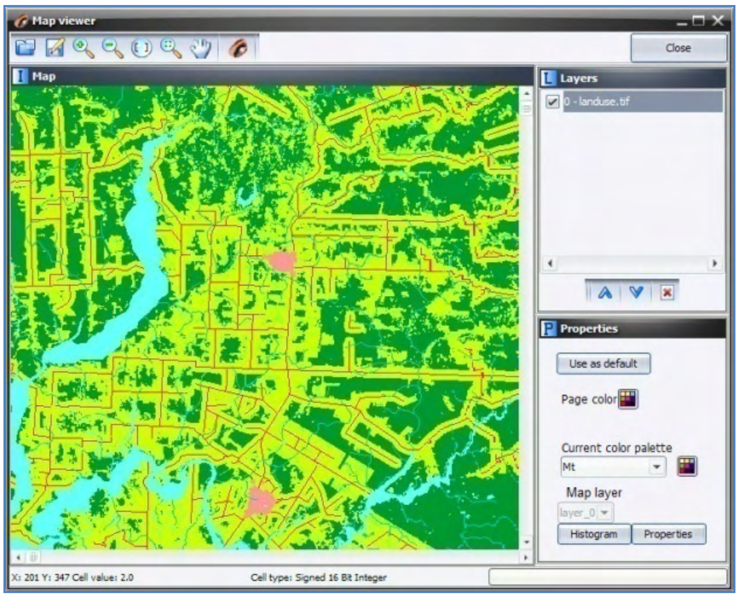
1) Distance to main roads < 15 kilometers.
2) Driving distance to neighboring towns < 30 minutes
3) At least 10 km away from existing towns.
4) At least 1000 meters away from flooded plains (Malaria is endemic in the region)
5) In addition, we need to consider that not all land is available. We don't want to encourage more deforestation in the region, thus we need to use only land that is deforested or abandoned. Also existing urban areas, rivers and flooded plains must be excluded.
6) At least areas larger than 1000 hectares.
7) Average slope < 0.5 degrees.

6.1 First step: Finding unconstrained areas

The land use map has the following classes:

Land use	**id**
Flooded plains	0
Small Rivers	1
Pasture	2
Regrowth	3
Forest Remnant	4
Urban	5
Roads	6

Load "\lesson4\originals\landuse.tif"it on the Map Viewer and change the **Current color palette** to "mt".

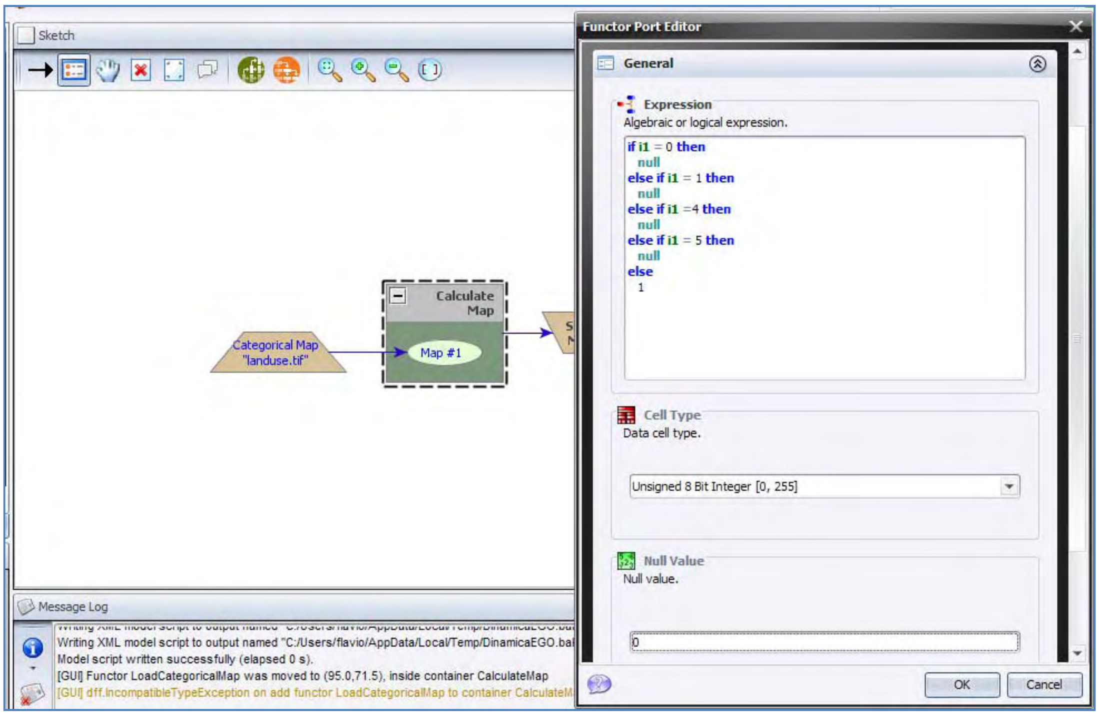


Let's start developing the model. Place *Load Categorical Map* on the sketch and open the file "\lesson4\original\ landuse.tif". Now place *Calculate Map* and *Number Map* within it. Enter "1" in **Map Number** in *Number Map*. Now place *Save Map* and enter "unconstrained_areas.tif", make sure you have changed the folder to **lesson5**. Change file format to **Geotiff** and set **Suffix Digits** to "0". (**TIP**: you can increase model performance by avoiding saving intermediate map files). In this step, you will save the map just to check the result from this operation. Do not forget to connect all functors.

Now open the *Calculate Map* with the Edit Functor and write the following equation:

if i1 = 0 then null else if i1 = 1 then null else if i1 =4 then null else if i1 = 5 then null else 1

The null value will mask out rivers, flooded plains, urban areas, and roads in subsequent map operations. Do not forget to set **Data Cell Type** to "Unsigned 8 Bit Integer" and **Null Value** to "0



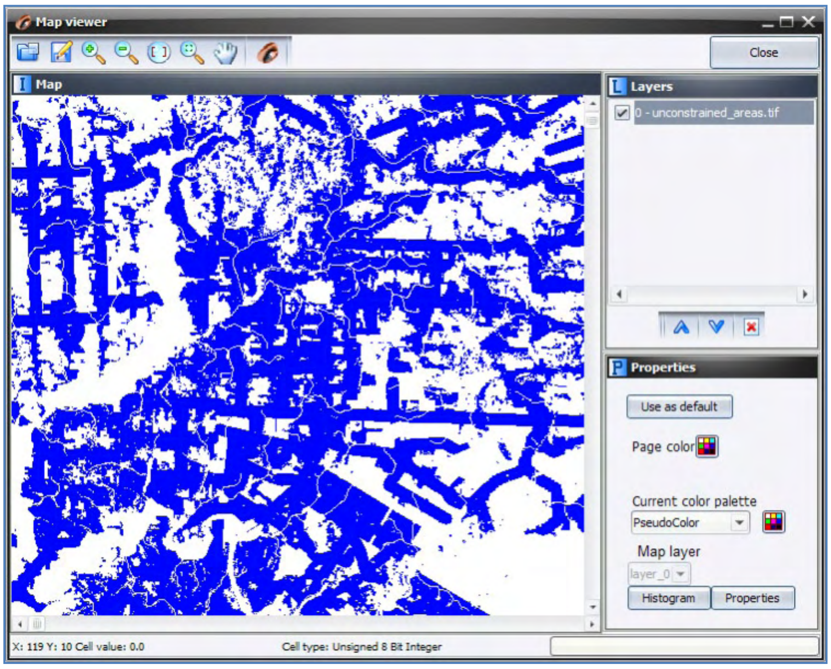
```
if i1 = 0 then
  null
else if i1 = 1 then
  null
else if i1 =4 then
  null
else if i1 = 5 then
  null
else
  1
```

Unsigned 8 Bit Integer [0, 255]

0

```
Writing XML model script to output named "C:/Users/flavio/AppData/Local/Temp/DinamicaEGO.ba
Model script written successfully (elapsed 0 s).
[GUI] Functor LoadCategoricalMap was moved to (95.0,71.5), inside container CalculateMap
[GUI] dff.IncompatibleTypeException on add functor LoadCategoricalMap to container CalculateM
```

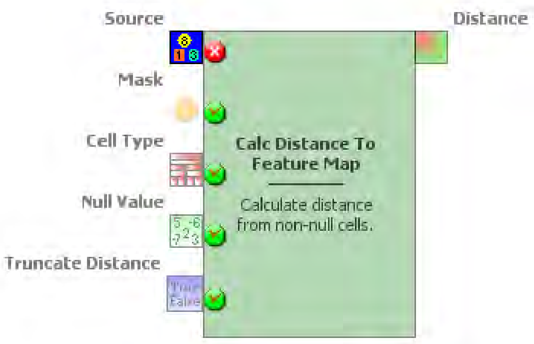
Save the model as "My_MCE_part1.xml", run it and examine the resulting map on the Map Viewer.



6.2 Second step: Building buffers to define unsuitable areas

In this step you need to build buffers to solve distance criteria to towns and flooded areas.

Load one *Calculate Categorical Map* together with its *Number Map* (**TIP:** The only difference between *Calculate Map* and *Calculate Categorical Map* is that the latter outputs a categorical map). You will need *Calculate Categorical Map* in order to use the functor *Calc Distance to Feature Map*; you can access it from the Map Algebra tab too. Although not necessary, you can save the map by placing *Save Map*. Save it as "distance_to_towns.tif". Now connect *Categorical Map "*landuse.tif*"* to *Number Map*, after setting the **Map Number** to "1". Open *Calc Distance to Feature Map* with the Edit Functor Ports.


Source
Mask
Cell Type
Null Value
Truncate Distance
Distance
Calc Distance To Feature Map
Calculate distance from non-null cells.

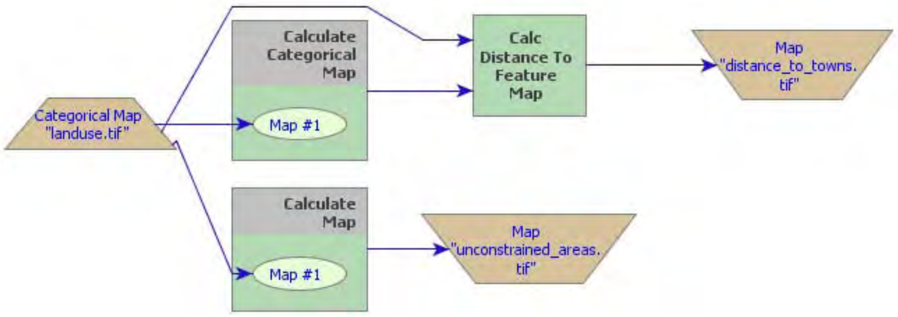
Source is a map containing the features to which distances will be calculated. A feature is represented by a non-null value. **Mask** is a map used to mask the distance calculation on its null cells. Let's connect the output from *Calculate Categorical Map to* **Source** *and Categorical*

Map "landuse.tif" to **Mask**. Now open the *Calculate Categorical Map* and write the following equation:

if i1 = 5 then1 else null

Remember that 5 is the number identifier for urban. You do not need to change either **Data cell type** or **Null Value** in both functors, although to save memory you could set "Signed 8 Bit Integer" and **Null Value** to "0" in *Calculate Categorical Map.* You may also turn on the option **Truncate Distance** in *Calc Distance to Feature Map*. This will avoid exceeding the maximum numerical representation selected for **Data cell type.**

Now connect the port **Distance** in *Calc Distance to Feature Map* to *Save Map.* The model should be ready to run. Always save the model before putting it to run. Save it as "my_MCE_part1&2.xml". Examine the result with the Map Viewer.

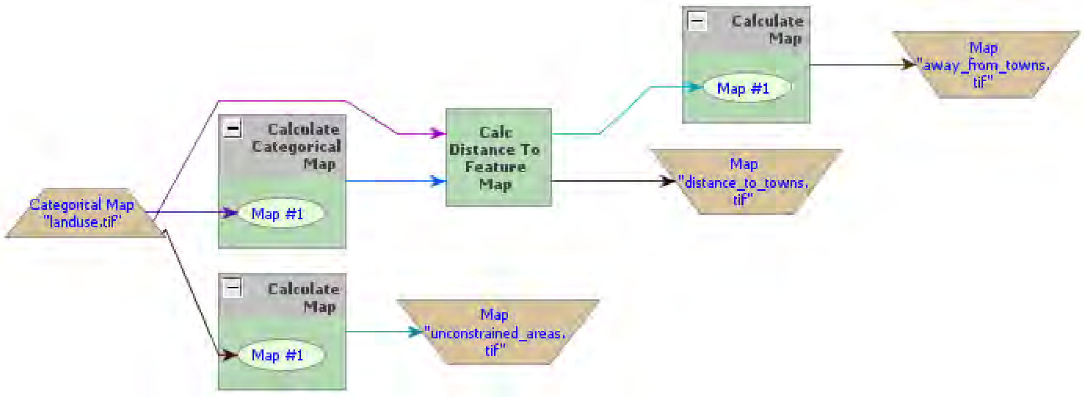

Calculate Categorical Map
Map #1
Calc Distance To Feature Map
Map "distance_to_towns.tif"
Categorical Map "landuse.tif"
Calculate Map
Map #1
Map "unconstrained_areas.tif"

As a last procedure, add one *Calculate Map* and its *Number Map,* connect the output of *Calc Distance to Feature Map* to its *Number Map* and output the result with *Save Map "*away_from_towns.tif*".* Notice that at this stage, we are omitting some steps, which should be well-known by now. Write in the equation box of the *Calculate Map*.

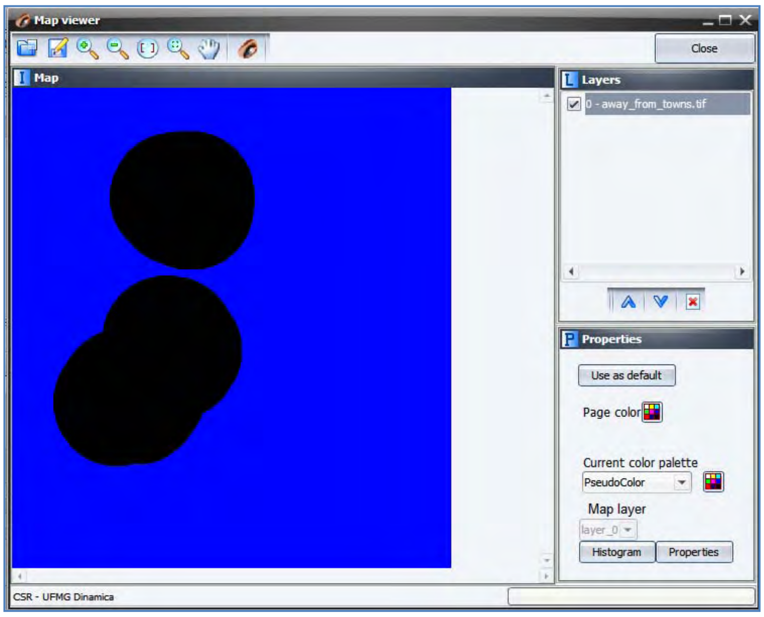
if i1 < 10000 then null else 1

TIP: The distance map is always produced in meters; the constraint criterion is always set to null.

You will have the following model.


Calculate Map
Map #1
Map "away_from_towns.tif"
Calculate Categorical Map
Map #1
Calc Distance To Feature Map
Map "distance_to_towns.tif"
Categorical Map "landuse.tif"
Calculate Map
Map #1
Map "unconstrained_areas.tif"

Save and run it. Open the map *"away_from_towns.tif".* Is this what you got?



How would you solve the problem having the following criteria?

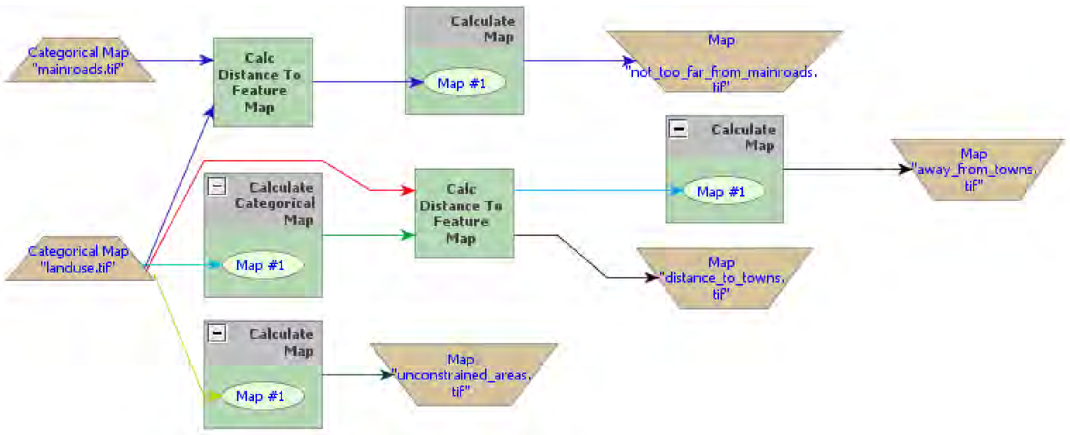
- Distance to major roads < 15 kilometers.
- At least 1000 meters away from flooded plains.

You will need to load another *Load Categorical Map* and enter the file mainroads.tif from folder lesson5. You will need to turn on the option **Define Null Value** and set the **Null Value** to "0". You can connect this functor directly to another *Calc Distance to Feature Map* that you will need to place on the sketch. Use *Categorical Map "* landuse.tif" again as the **Mask**.

Use the following equations in the respective Calculate Map (you will need to place one for each criterion): **if i1 > 15000 then null else 1**

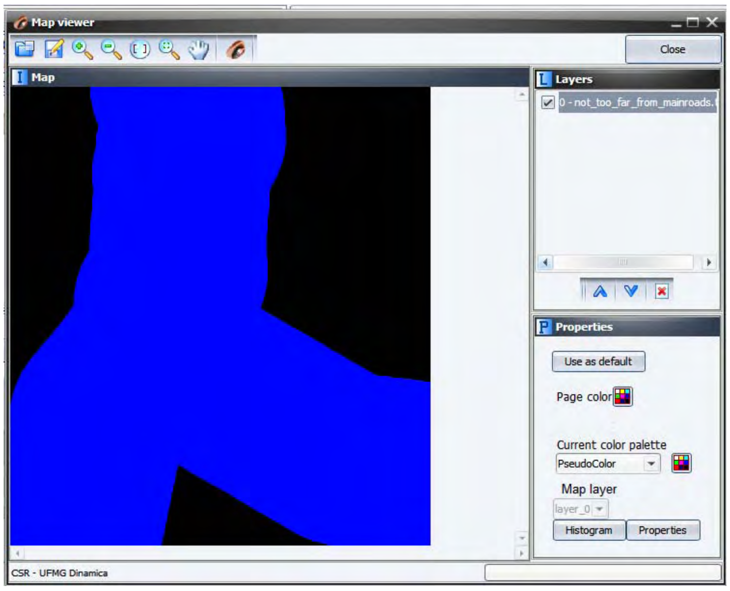
Now connect its output to *Save Map* "not_too_far_from_mainroads.tif"

You will have the following model.


Categorical Map "mainroads.tif"
Calc Distance To Feature Map
Calculate Map
Map #1
Map "not_too_far_from_mainroads.tif"
Calculate Map
Map #1
Map "away_from_towns.tif"
Calculate Categorical Map
Map #1
Calc Distance To Feature Map
Categorical Map "landuse.tif"
Map "distance_to_towns.tif"
Calculate Map
Map #1
Map "unconstrained_areas.tif"

Check its integrity, save as my_MCE_part1&2.1.xml and run it.

Did you get a “not_too_far_from_mainroads.tif” map like this?



For the flooded plains criterion, you will need to add another *Calculate Categorical Map* to pinpoint the flooded areas before passing the resulting map to *Calc Distance to Feature Map*.

Connect *Categorical Map “*landuse.tif” to it and write the following equation in the *Calculate Categorical Map* equation box.

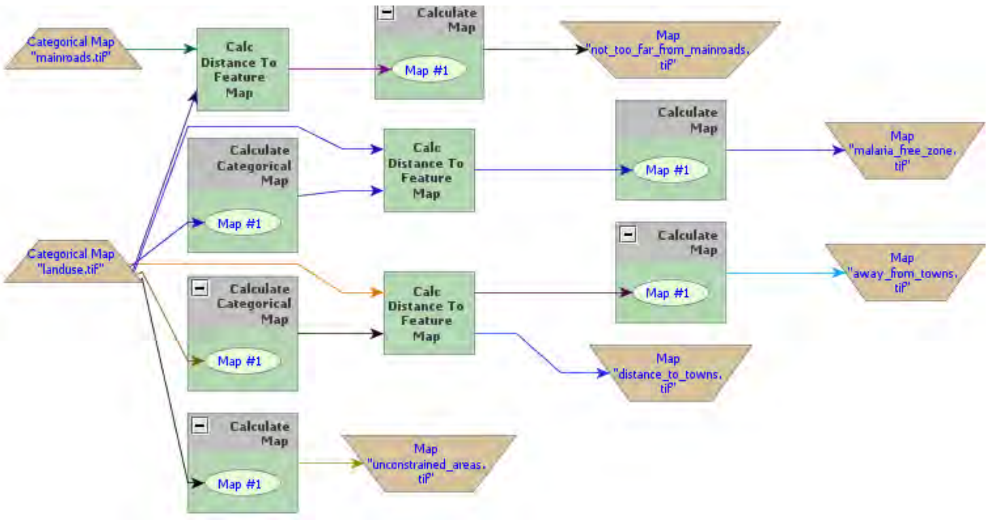
if i1 = 0 then 1 else null

Now connect this output to *Calc Distance to Feature Map* as the **Source** port*,* again use *Categorical Map “*landuse.tif” as the **Mask** port**,** then connect its output to another Calculate Map and write the following equation:

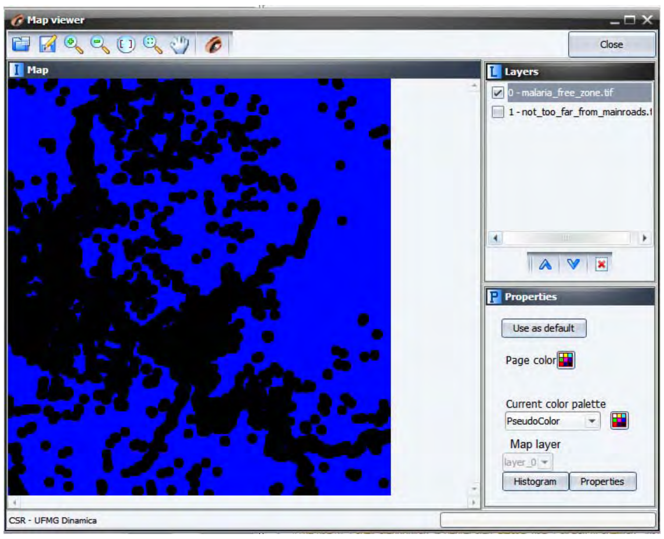
if i1 < 1000 then null else 1

Save the output as “malaria_free_zone.tif”

The model for the three criteria combined should look like this:


Categorical Map "mainroads.tif"
Calc Distance To Feature Map
Calculate Map
Map #1
Map "not_too_far_from_mainroads.tif"
Calculate Categorical Map
Map #1
Calc Distance To Feature Map
Calculate Map
Map #1
Map "malaria_free_zone.tif"
Categorical Map "landuse.tif"
Calculate Categorical Map
Map #1
Calc Distance To Feature Map
Calculate Map
Map #1
Map "away_from_towns.tif"
Map "distance_to_towns.tif"
Calculate Map
Map #1
Map "unconstrained_areas.tif"

Check the model, save it as “my_MCE_part1&2complete.xml”, run and analyze the map “malaria_free_zone.tif”. Does it look like this?



6.3 Third step: Calculating driving time to neighboring towns

Driving traveling time to neighboring towns must not exceed 30 minutes. This fourth criterion requires the use of *Calc Cost Map* as well as a transportation friction surface.

To solve this criterion the model will establish two speeds of traveling: one for the road network and another for all types of land uses. Traveling on road has an average speed of 60km/h. Thus the corresponding speed to cross a unit cell of 100 meters of resolution will be 0.1 minutes. In turn, traveling through the outback has an average speed of 20 km/h, which is equivalent to 0.3 cell/minute.

Explanation

x (hour) -> 0.1 km

1 h -> 60 km

This is equal to = 0.1 km/60 km x 60 minutes = 0.1 minute

Let’s then reclassify *Map landuse.tif* to produce the friction surface. In order to this you need to place another *Calculate Map* and write into its equation box the following equation:

if i1 = 6 then 0.1 else 0.3

Now connect its output to a *Calculate Cost Map* functor (to its **Friction** port). You will need another *Calculate Map* to pinpoint the neighboring towns from the land-use map using the equation below:

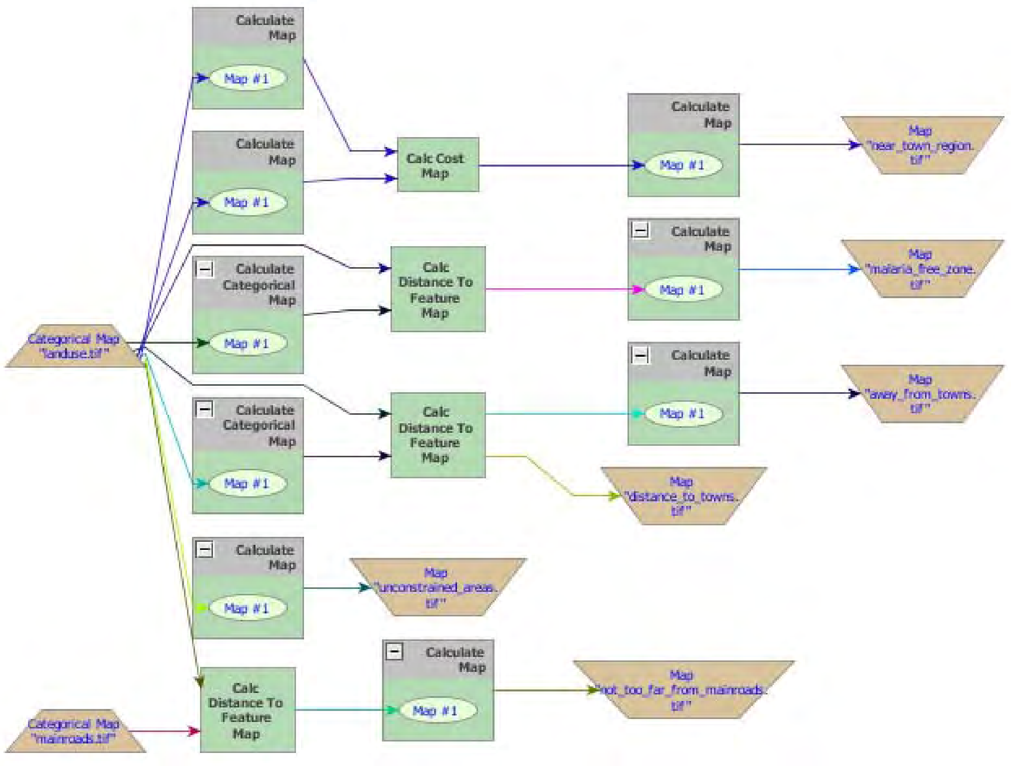
if i1 = 5 then 1 else null

Important: Do not forget to change **Cell Type** both in *Calculate Map* that generates the friction map and also in *Calc Cost Map* to “IEEE 754 Bit Real”. Remember that the cell cost is expressed in fractional number.

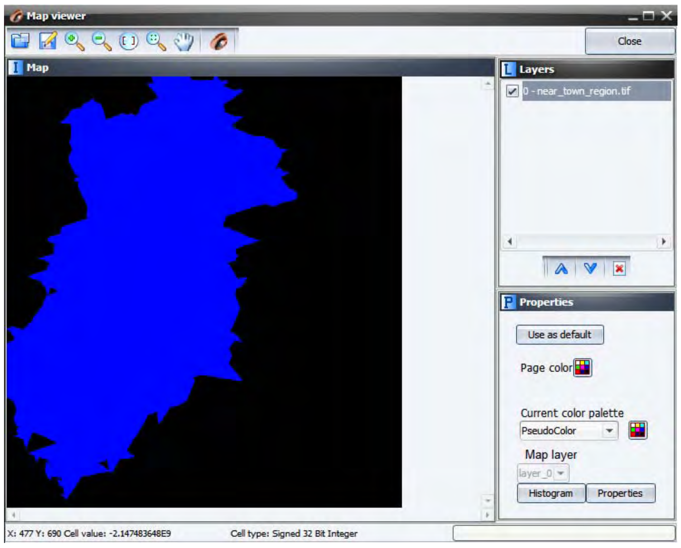
Now pass the output of the *Calc Cost Map* to another *Calculate Map* to produce a Boolean map according to the maximum time of traveling, as follows:

if i1 < 30 then 1 else null

Finally save the output map as “near_town_region.tif”. Note that you have added to the model three additional *Calc Map* functors, three *Number Map* functors, one *Calc Cost Map* and another *Save Map*. The model should look like the following figure, in which the functors added in this step appear in the upper part.


Calculate Map
Map #1
Calc Cost Map
Calculate Categorical Map
Calc Distance To Feature Map
Categorical Map "landuse.tif"
Categorical Map "mainroads.tif"
Map "near_town_region.tif"
Map "malaria_free_zone.tif"
Map "away_from_towns.tif"
Map "distance_to_towns.tif"
Map "unconstrained_areas.tif"
Map "not_too_far_from_mainroads.tif"

Check model, integrity, save it as my_MCE_part1&2&3.xml, and run it. Examine map “near_town_region.tif”. Is this what you got?



6.4 Fourth step: Combining the Boolean criteria

So far you have solved the following criteria.

1) Distance to main roads < 15 kilometers.
2) Driving distance to neighboring towns < 30 minutes
3) At least 10 km away from existing towns.

4) At least 1000 meters away from flooded plains.
5) In addition, we need to consider that not all land is available. We don't want to encourage more deforestation in the region, thus using only land that is deforested or abandoned. Also existing urban areas, rivers and flooded plains must be excluded.

The two remaining criteria (areas equal or greater than 1000 hectares, and average slope < 0.5 degrees) must be solved using a zonal approach as provided by the Region functors. Before moving on to this step, you need to combine all criteria already solved into a single map.

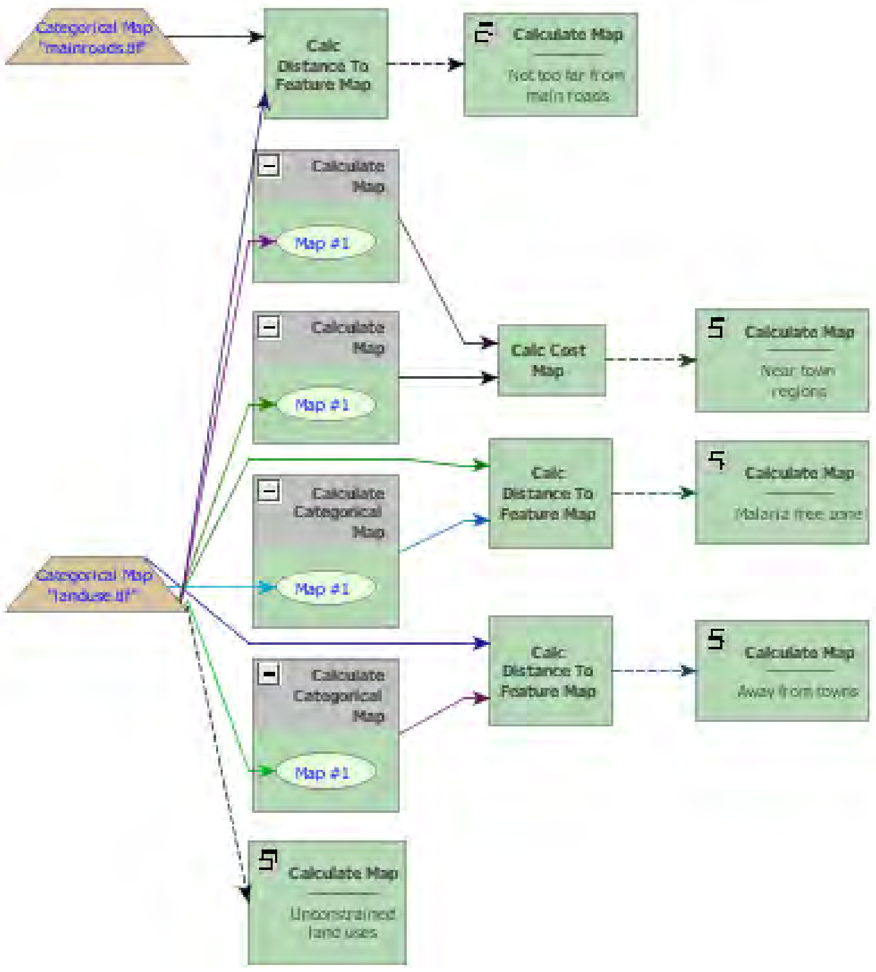
As you have noticed this model has reached a certain level of complexity, and it beginning to become difficult to grasp its structure through the graphical interface. Dinamica EGO presents some facilities to reduce the visual complexity of models.

Firstly, you can delete the *Save Map* functors, since you have already checked their results. Second, you can add comments to some functors highlighting their outcomes. Let's do it.

Grab the Add Comment to Functor tool from the sketch toolbar. Describe the outcome in each functor preceding the *Save Map* functors, so you can trace each data flow back to its initial node (that is Map "landuse.tif"). In order to be able to see the comment you need to close the container *Calculate Map* first. Do it by clicking on its top left icon.

Write in respective functors: Near town regions, Malaria free zone, Away from towns, Unconstrained land uses, Not too far from main roads.

At this stage, you can delete the *Save Map* functors, since you have already checked their results. After doing so, organize the layout from left to right. **TIP:** resize the window sketch to get a closer view at the model. This is what you should get.


Categorical Map "mainroads.tif"
Calc Distance To Feature Map
Calculate Map
Not too far from main roads
Calculate Map
Map #1
Calculate Map
Map #1
Calc Cost Map
Calculate Map
Near town regions
Calculate Categorical Map
Map #1
Calc Distance To Feature Map
Calculate Map
Malaria free zone
Categorical Map "landuse.tif"
Calc Distance To Feature Map
Calculate Map
Away from towns
Calculate Categorical Map
Map #1
Calculate Map
Unconstrained land uses

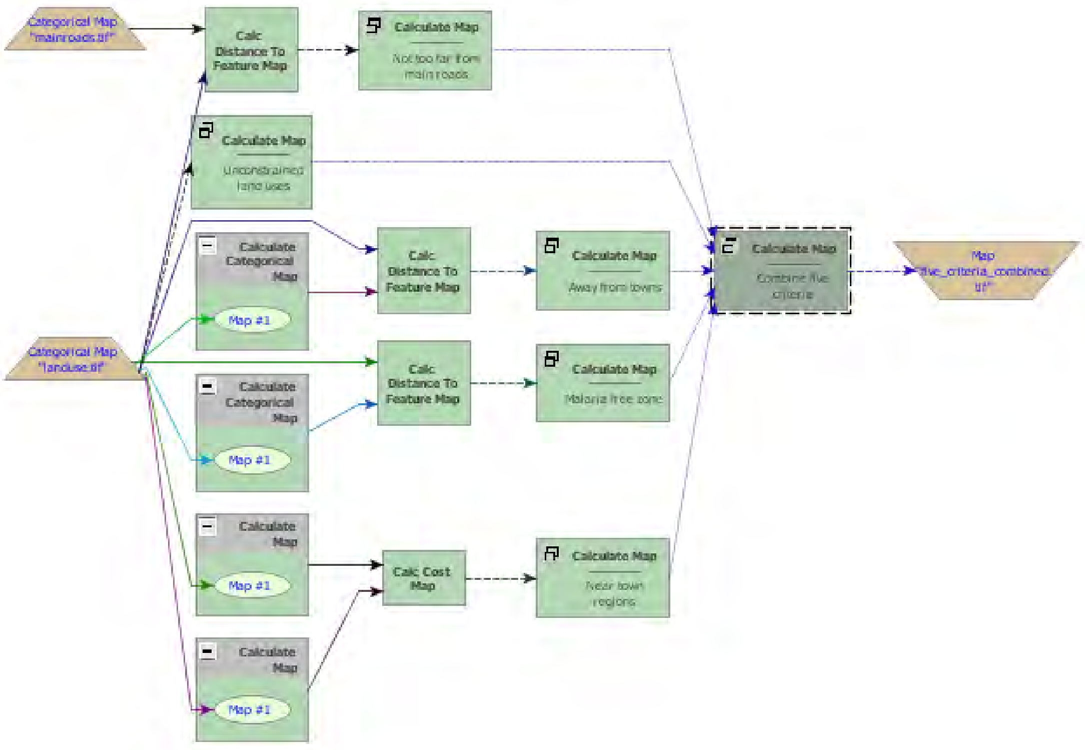
The five titled *Calculate Map* functors produce the five criteria solved so far. Now you just need to combine them with *Calculate Map*.

Now place five *Number Map* functors within it. Open it and write the following equation.

i1*i2*i3*i4*i5

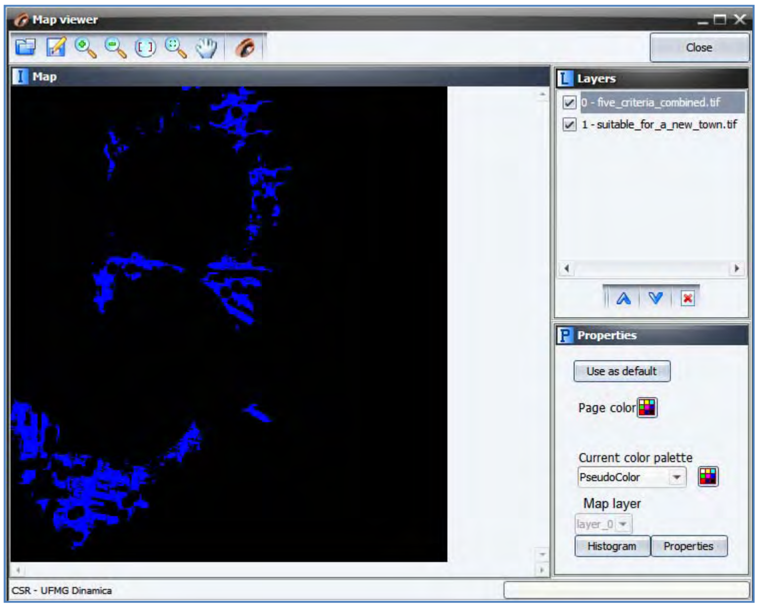
TIP: the Calculate Map does not process null cells.

Change **Cell Type** to "UnSigned 8 Bit Integer" **Null Value** to "0"; write the comment "Combine five criteria" and add *Save Map* to view the result, naming the file as "five_criteria_combined.tif". Finally organize the layout left to right.

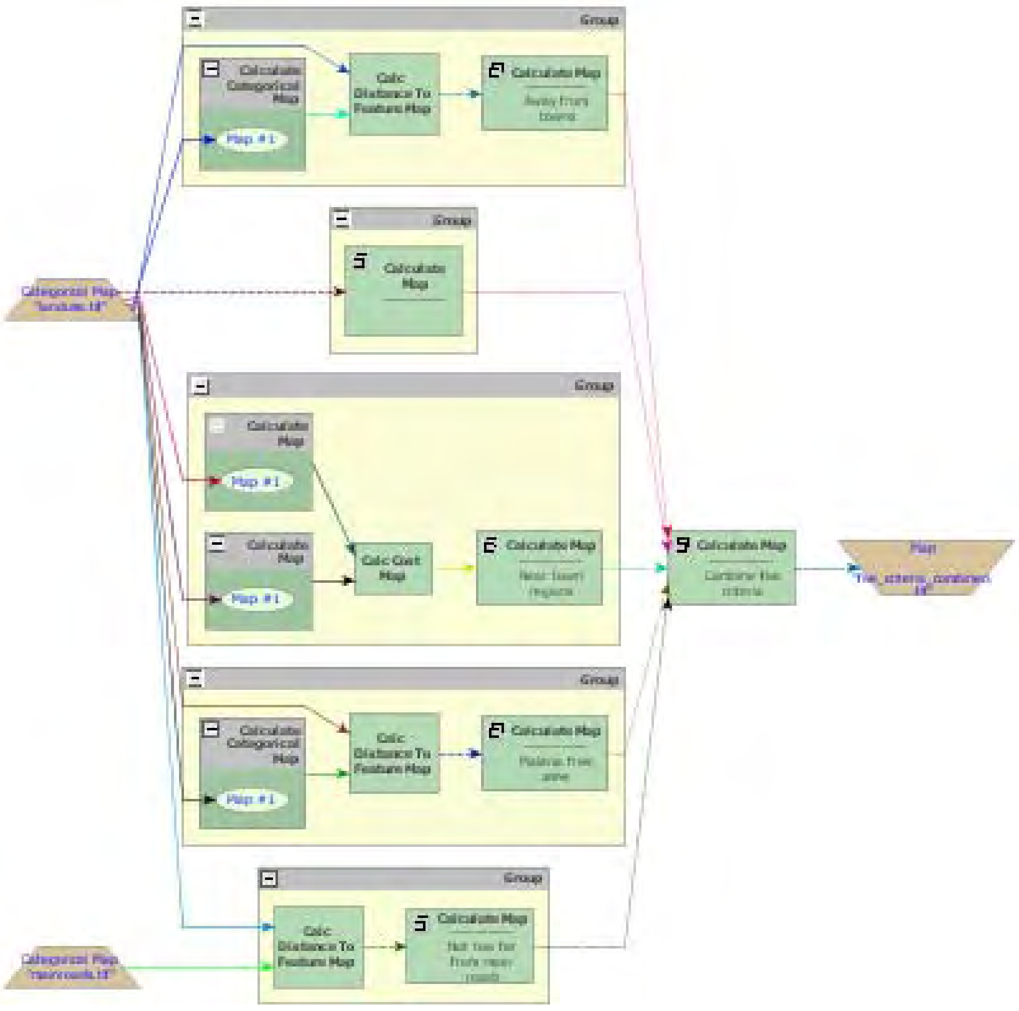


Categorical Map "mainroads.tif"
Calc Distance To Feature Map
Calculate Map
Not too far from main roads
Calculate Map
Unconstrained land uses
Calculate Categorical Map
Map #1
Calc Distance To Feature Map
Calculate Map
Away from towns
Calculate Map
Combine five criteria
Map "five_criteria_combined.tif"
Categorical Map "landuse.tif"
Calculate Categorical Map
Map #1
Calc Distance To Feature Map
Calculate Map
Malaria free zone
Calculate Map
Map #1
Calc Cost Map
Calculate Map
Near town regions
Calculate Map
Map #1

Close the right-most container to show the comments, save the model as my_MCE_part1&2&3&4.xml, check and run it. Is this what you get?



To further simplifier the view of the model, another resource available in Dinamica EGO is the *Group* functor. Drag five *Group* functors from the Control tab and place them on the sketch. Now select each functor of a data flow chain that ends up in *Calculate Map* “Combine five criteria” (Press **Crt + right button** to keep on continually selecting functors), leave the *Categorical Map* functors out, now drag them into a *Group*. Repeat this procedure for the five criteria.


Group
Calculate Categorical Map
Map #1
Calc Distance To Feature Map
Calculate Map
Away from towns
Group
Calculate Map
Categorical Map "landuse.tif"
Group
Calculate Map
Map #1
Calculate Map
Map #1
Calc Cost Map
Calculate Map
Near town regions
Calculate Map
Combine five criteria
Map "five_criteria_combined.tif"
Group
Calculate Categorical Map
Map #1
Calc Distance To Feature Map
Calculate Map
Malaria free zone
Group
Calc Distance To Feature Map
Calculate Map
Not too far from main roads
Categorical Map "mainroads.tif"

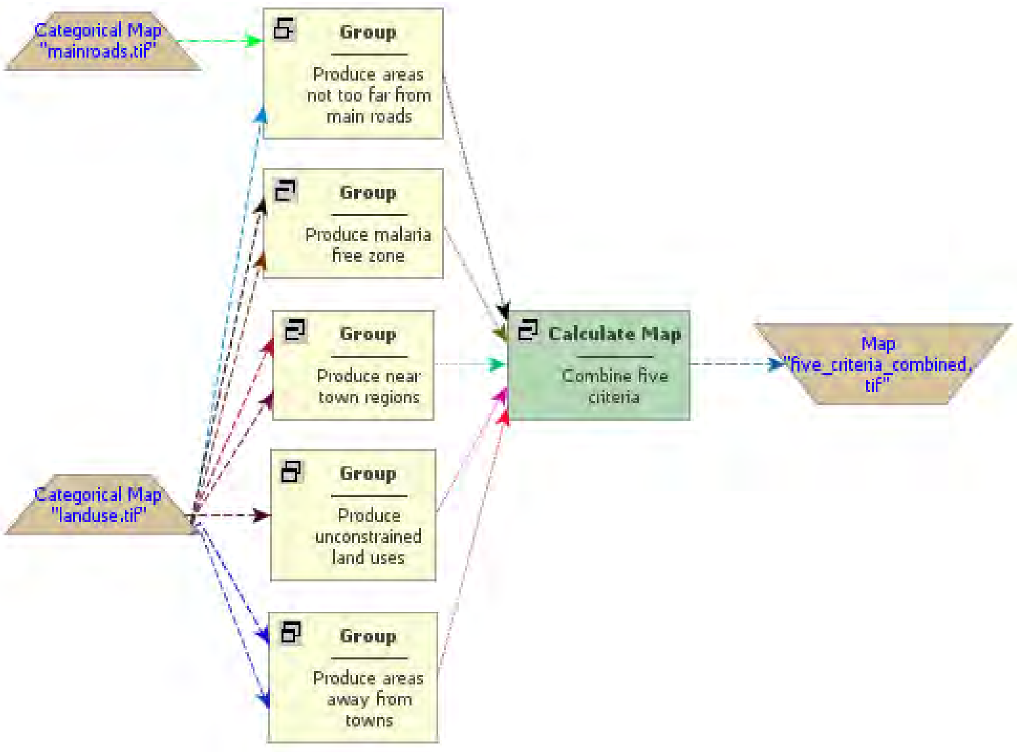
Close each *Group* and write the following comments in their respective *Group*. You may need to zoom in.

Produce near town regions
Produce malaria free zone
Produce areas away from towns
Produce unconstrained land uses
Produce areas not too far from main roads

See model on the next page. Didn’t the model become much easier to visualize? Of course, you will need to open the *Group* functor to understand how the data are produced. You can do this one by one. Note that the function of the *Group* functor is to help organize the model diagram.

TIP: *Group* is also used to ensure a proper order of execution, given that it always executes its enveloped functor before other subsequent functors.

Save this model as my_MCE_part1&2&3&4group.xml and let's move forward to the next step.

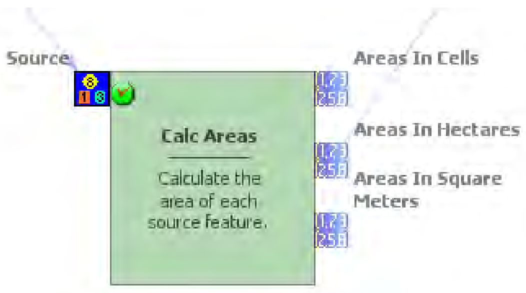

Categorical Map "mainroads.tif"
Group
Produce areas not too far from main roads
Group
Produce malaria free zone
Group
Produce near town regions
Categorical Map "landuse.tif"
Group
Produce unconstrained land uses
Group
Produce areas away from towns
Calculate Map
Combine five criteria
Map "five_criteria_combined.tif"

6.5 Fifth step:

In this step, you will identify the suitable areas equal to or greater than 1000 hectares. First, delete Map "five_criteria_combined.tif". Remember that the less the software accesses the disk, the faster the model runs. Now, drag from Map Algebra tab the functor *Calc Patch Label Map*. This functor labels patches of cells using sequential numbers. A patch consists of a continuous group of neighboring cells of the same category (see also lesson 10: Landscape metrics in Dinamica EGO). You need this in order to solve the area criterion. Connect the output from "Combine Five Criteria" to it. Some of its parameters are:

Initial Patch Label: Set it to "1". **Important:** Set **Null Value** to "0"and **Cell Type** to **Signed 32 Bit Integer**.

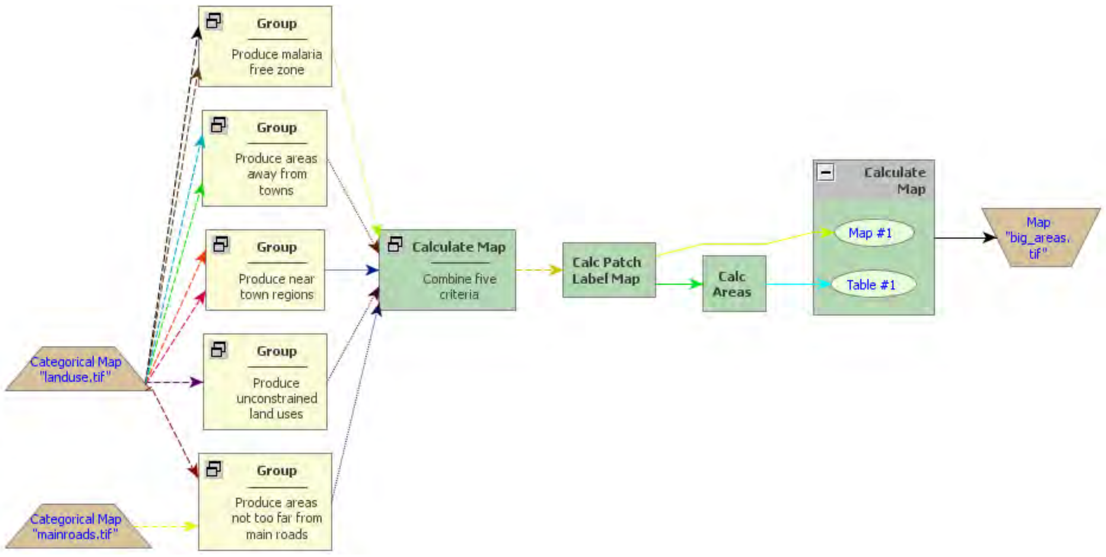
This will be sufficient to store all patch label number identifiers. Leave the other parameters untouched. Go to help for further details. Now connect its output to *Calc Areas*, available in the Map Algebra tab. As output this functor provides 3 options: We need area in hectares, so use the port **Areas in Hectares** for subsequent connection.



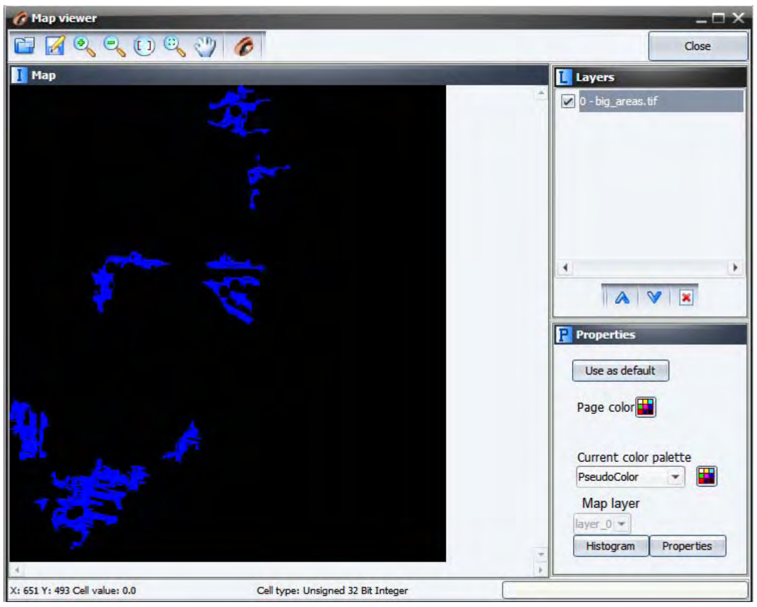
Now, place on the sketch one *Calculate Map* and one *Number Map* and one *Number Table* within it. Connect *Calc Areas* to *Table 1* (this implies that output of *Calc Areas* is a lookup table), and **Map** output from **Calc Patch Label Map** to *Map #1*. Now open *Calculate Map* and write:

if (t1[i1] < 1000) then null else 1

Using this formula, the functor will analyze the size of each patch and then eliminate the patches smaller than 1000 hectares. Now place one *Save Map* and enter "big_areas.tif".

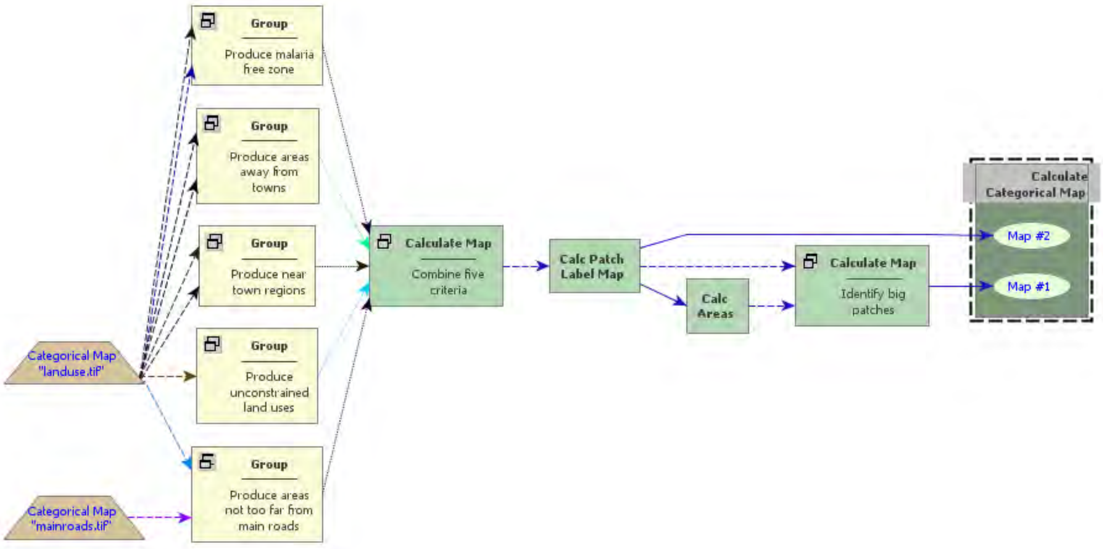

Group
Produce malaria free zone
Group
Produce areas away from towns
Group
Produce near town regions
Group
Produce unconstrained land uses
Group
Produce areas not too far from main roads
Categorical Map "landuse.tif"
Categorical Map "mainroads.tif"
Calculate Map
Combine five criteria
Calc Patch Label Map
Calc Areas
Calculate Map
Map #1
Table #1
Map "big_areas.tif"

Check the model, save it as "my_MCE_part1&2&3&4&5.xml, run it and examine its output. Is this what you got?



Great, what a haul! But before moving on to the next step, there is still a last thing to do. In order to keep the patch labels, you need to multiply the output from the last *Calculate Map*, to which you have added the comment "Identify big patches", with the output from *Calc Patch Label*. Therefore, you will keep only the labels for the big patches.

Place one *Calculate Categorical Map* and two *Number Map* functors within it. Set their *Number Map* functors to “1” and “2” respectively, and remove the *Save Map* functor. Now connect functors as follows:


Group
Produce malaria free zone
Group
Produce areas away from towns
Group
Produce near town regions
Group
Produce unconstrained land uses
Group
Produce areas not too far from main roads
Categorical Map "landuse.tif"
Categorical Map "mainroads.tif"
Calculate Map
Combine five criteria
Calc Patch Label Map
Calc Areas
Calculate Map
Identify big patches
Calculate Categorical Map
Map #2
Map #1

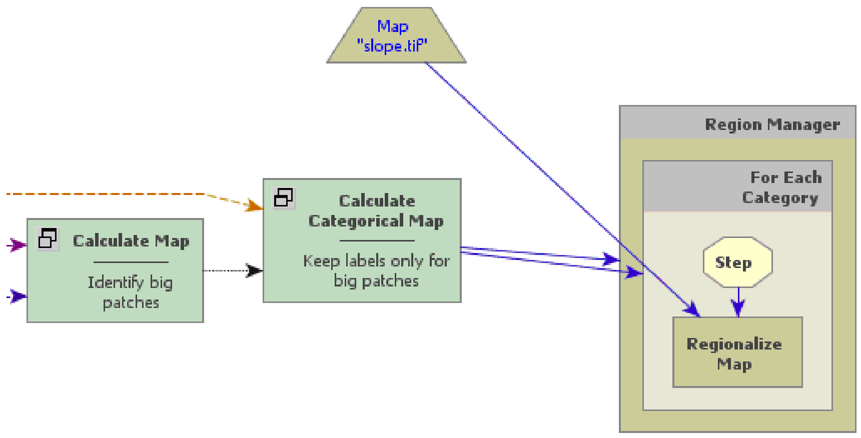
TIP: You need *Calculate Categorical Map* instead of *Calculate Map*, because patch labels will be used ahead to regionalize the slope map, set **Null Value** to “0”and **Cell Type** to **Signed 32 Bit Integer**.

Write: **i1*i2**, close the functor and insert the comment “Keep labels only for big patches”. Now you are ready to move on to the last step.

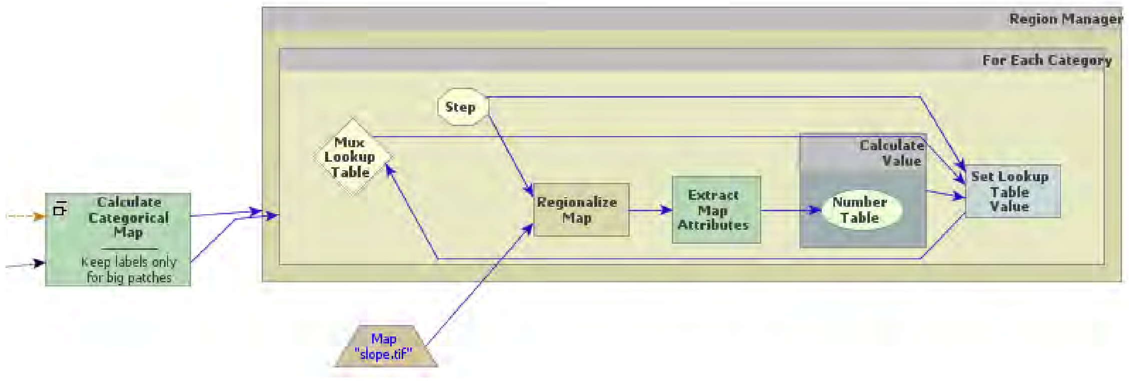
6.6 Sixth step:

The final criterion establishes that the average slope on the selected patches must be less than 0.5 degrees. Place *Load Map* and load it with “\lesson4\originals\slope.tif”. Now revise Lesson3, as you will to need to reapply the concept of region.

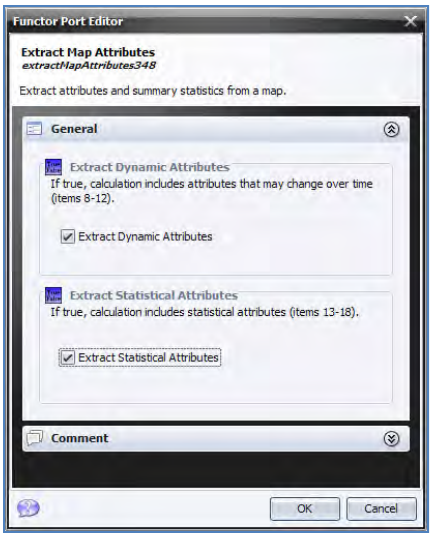
Drag *Region Manager*, *For Each Category*, *Regionalize Map*, and *Step* and insert the two latter functors within the previous and *For Each Category* within the *Region Manager* and connect them as follows.



The map to be regionalized is the “slope.tif”. The map output from the last *Calculate Categorical Map* will control both *Region Manager* and *For Each Category*. Remember that this functor enables the model to iterate according to the map categories passed to it. Now place one *Mux Lookup Table*, one *Extract Map Attributes*, one *Calculate Value*, one *Number Table* and one *Set Lookup Table*, all within *For Each Category* and connect them as follows.



Write equation: **t1[13]** in the *Calculate Value*. Remember that 13 is the key for the average of all non null cells in the attribute table. This submodel is very similar to the one introduced in Lesson 3. Open *Extract Map Attributes* and set the option:


Functor Port Editor

Extract Map Attributes
extractMapAttributes348

Extract attributes and summary statistics from a map.

General

Extract Dynamic Attributes
If true, calculation includes attributes that may change over time (items 8-12).

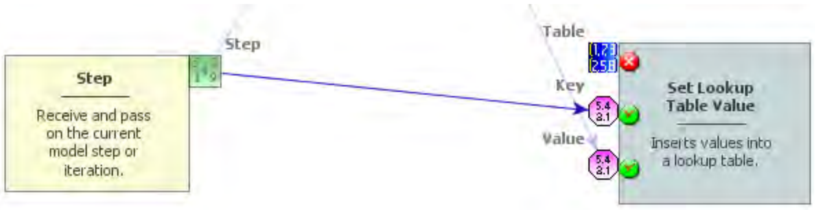
☑ Extract Dynamic Attributes

Extract Statistical Attributes
If true, calculation includes statistical attributes (items 13-18).

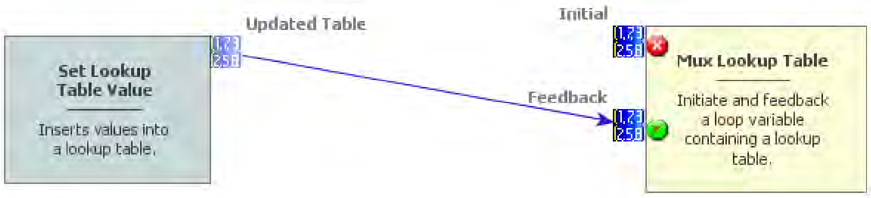
☑ Extract Statistical Attributes

Comment

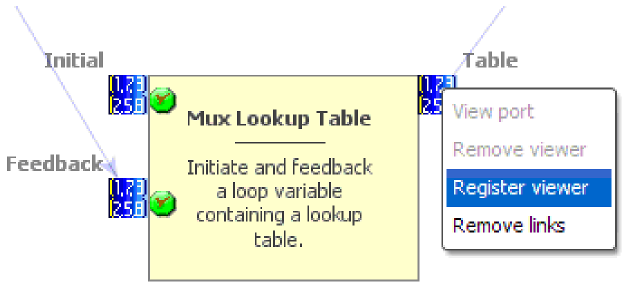
Remember to link *Step* to **Key** on the *Set Lookup Table*


Step
Receive and pass on the current model step or iteration.
Step
Table
Key
Value
Set Lookup Table Value
Inserts values into a lookup table.

As well as **Updated Table** to **Feedback**

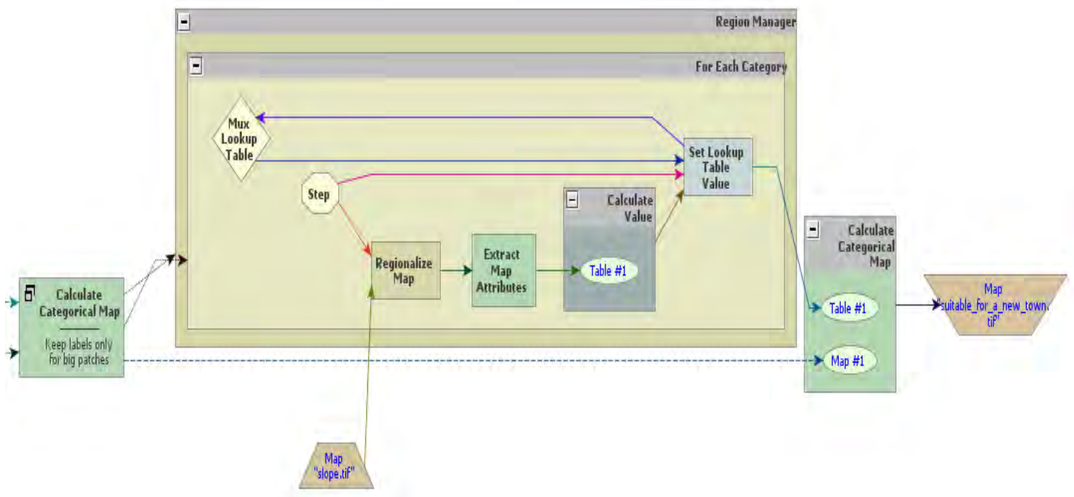

Set Lookup Table Value
Inserts values into a lookup table.
Updated Table
Initial
Feedback
Mux Lookup Table
Initiate and feedback a loop variable containing a lookup table.

Open *Mux Lookup Table* with the Edit Functor Ports and click on **Initial,** enter a blank table. Just enter once “0” and “0”, as **Key** and **Value**. Register the **Table** output port to be viewed. You might want to see the results.

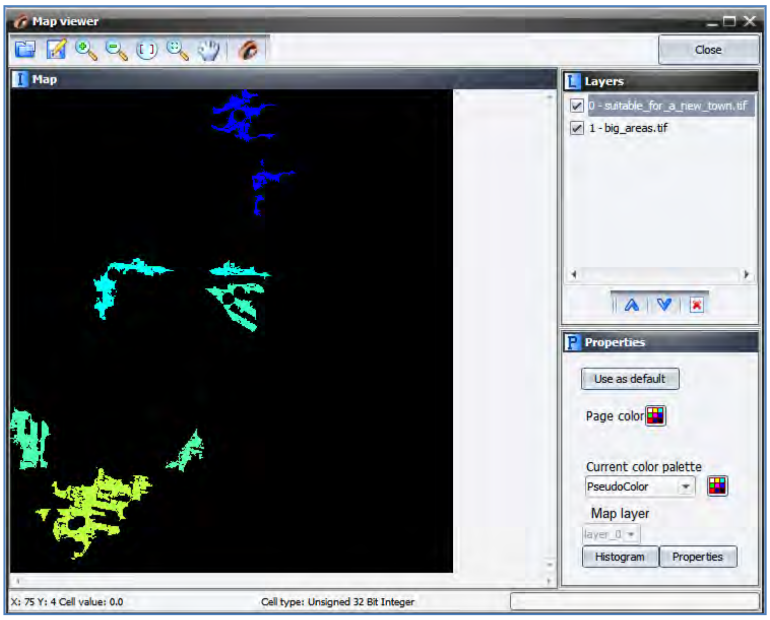


Finally place *Calculate Categorical Map* after the *Region Manager* and place *Map Number Map* and *Number Table* within it. Number them and write the following equation: **if t1[i1] > 0.5 then null else i1**

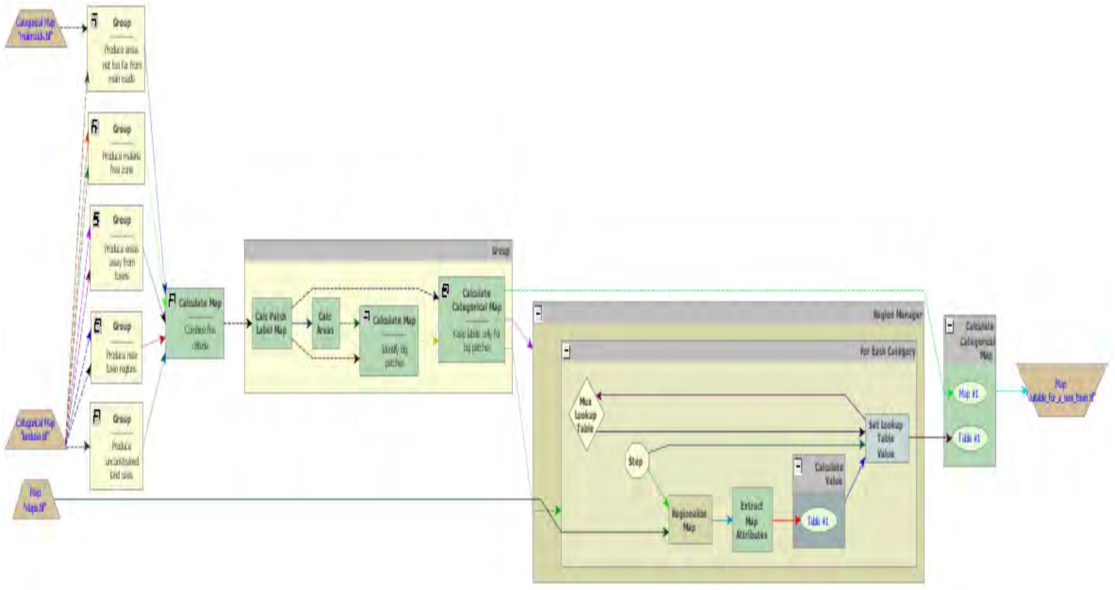
Save the result as “suitable_for_a_new_town.tif”.



Check the model, save it as “my_MCE_part1&2&3&4&5&6.xml” and run it. Is this what you got?

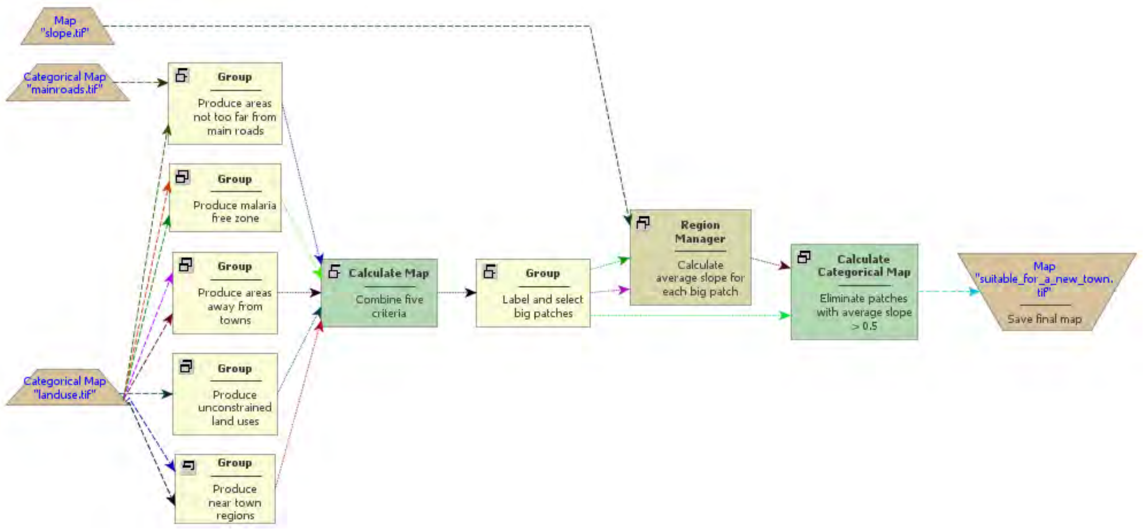


Finally you just need to organize the model in a more legible layout. Place one *Group* and drag *Calc Patch Label Map*, *Calc Areas*, *Calculate Map* "Identify big areas", and *Calculate Map* "Keep labels only for big patches" into it.


Categorical Map "mainroads.tif"
Group
Produce areas not too far from main roads
Group
Produce malaria free zone
Group
Produce areas away from towns
Group
Produce near town regions
Group
Produce unconstrained land uses
Categorical Map "landuse.tif"
Map "slope.tif"
Calculate Map
Combine five criteria
Group
Calc Patch Label Map
Calc Areas
Calculate Map
Identify big patches
Calculate Categorical Map
Keep labels only for big patches
Region Manager
For Each Category
Mux Lookup Table
Step
Set Lookup Table Value
Regionalize Map
Extract Map Attributes
Calculate Value
Table #1
Calculate Categorical Map
Map #1
Table #1
Map "suitable_for_a_new_town.tif"

Close it and add the comment "Label and select big patches", close the *Region Manager* and add "Calculate average slope for each big patch", close the last *Calculate Map* functor and add "Eliminate patches with average slope > 0.5" and lastly, add the comment "Save final map" to *Map "*suitable_for_a_new_town.tif*".*

Save the model as "my_MCE_part1&2&3&4&5&6_complete.xml". This is the final model.


Map "slope.tif"
Categorical Map "mainroads.tif"
Group
Produce areas not too far from main roads
Group
Produce malaria free zone
Group
Produce areas away from towns
Group
Produce unconstrained land uses
Group
Produce near town regions
Categorical Map "landuse.tif"
Calculate Map
Combine five criteria
Group
Label and select big patches
Region Manager
Calculate average slope for each big patch
Calculate Categorical Map
Eliminate patches with average slope > 0.5
Map "suitable_for_a_new_town.tif"
Save final map

HOME WORK

Examining the map "suitable_for_a_new_town.tif", you will note a large patch on its southwest portion, which would be the best target for a new town, considering the areal extent as an additional criterion. Would you be able to add a submodel to the current model to pinpoint this patch?

TIP: You will need to recalculate the areas for the remaining patches and pass this table to *For Each*, which allows to browse through a table. Place two *Mux Value* inside the *For Each*, one will handle the maximum area value and the other will store its key.

7. Lesson 6: Building a land-use and land-cover change simulation model

What will you learn?

- How to simulate land-use change
- How to calculate transition matrix
- How to calibrate a land-use change model
- Using Weights of Evidence
- Map Correlation
- Model Validation
- Functors:
    - *Determine Weights Of Evidence Ranges*
    - *Determine Weights Of Evidence Coefficients*
    - *Calc WOfE Probability Map*
    - *Calc Change matrix*
    - *Patcher*
    - *Expander*

The development of space-time models, in which the state or attribute of a certain geographical location changes over a time span as a response of a set of drivers, is an utmost requirement for environmental modeling and thus opens an avenue of possibilities for the representation of dynamic phenomena.

In this context, this exercise explores the use of Dinamica EGO as a simulation platform for land-use and cover change (LUCC) models. The goal is to calibrate, run and validate a LUCC model, in this case a simulation model of deforestation. You will need to go through 10 steps in order to complete the model, as depicted in fig. 8. To facilitate this process, each one of these steps will be represented as a separate model. Although, all steps could be joined into a single model, for reason of simplicity we will keep them as separate models.

The input dataset represents a region of Rondonia state, around the town of Ariquenes, in the Brazilian Amazon (fig. 9). Open the maps “23267_1997.ers” and “23267_2000”.ers, located in “\Examples\setup_run_and_validate_a_lucc_model\originals” using the **Color Palette**, “Amazon”. These maps correspond to a Landsat image (232/67) classified by PRODES (INPE, 2008) – Brazilian program for monitoring deforestation - for the years 1997 and 2000.

In this LUCC model, Dinamica EGO will use the 1997 map as the initial landscape and the 2000 map as the final, considering the landscape as a bi-dimensional array of land use types.

The landscape maps have the following classes; the null is represented by 0:

Key	Land cover classes
1	Deforested
2	Forested
3	Non-forest

TIP: Keep in mind the numbers that identify the map classes, since Dinamica EGO does not explicitly handle class names.

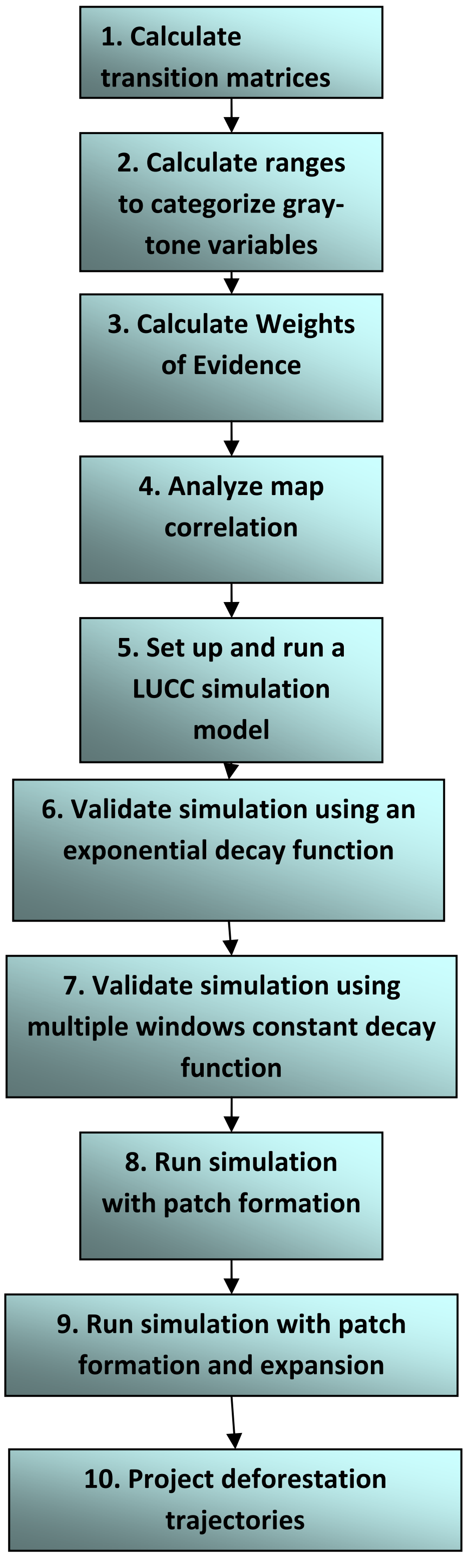

1. Calculate transition matrices
2. Calculate ranges to categorize gray-tone variables
3. Calculate Weights of Evidence
4. Analyze map correlation
5. Set up and run a LUCC simulation model
6. Validate simulation using an exponential decay function
7. Validate simulation using multiple windows constant decay function
8. Run simulation with patch formation
9. Run simulation with patch formation and expansion
10. Project deforestation trajectories

Fig. 8 – Ten steps of the land cover change simulation model developed in lesson 6.

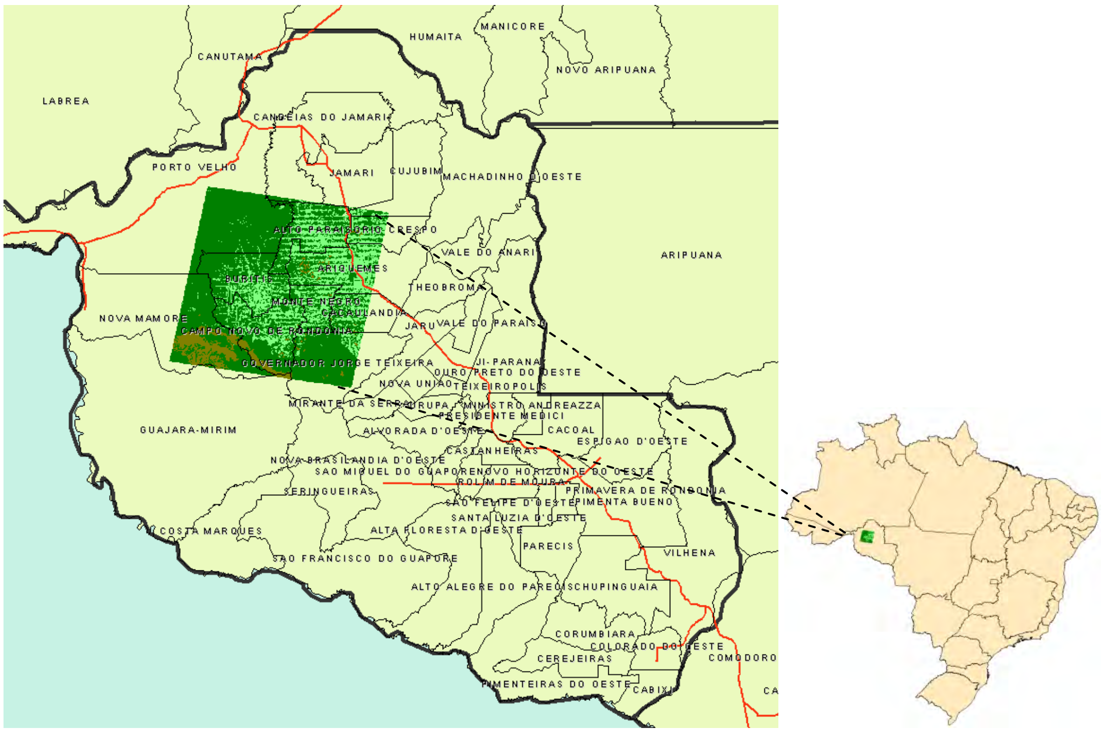


Fig. 9 - The study area with respect to Rondonia state and Brazil (dark green is forest, light green deforested land, and brown non-forest vegetation).

7.1 First step: Calculating transition matrices

First, you need to calculate the historical transition matrices. The transition matrix describes a system that changes over discrete time increments, in which the value of any variable in a given time period is the sum of fixed percentages of values of all variables in the previous time step. The sum of fractions along the column of the transition matrix is equal to one (eq. 1). The diagonal line of the transition matrix does not need to be specified since Dinamica EGO does not model the percentage of unchangeable cells, nor do the transitions equal to zero. The transition rate can be passed to the LUCC model as a fixed parameter or be updated from model feedback.

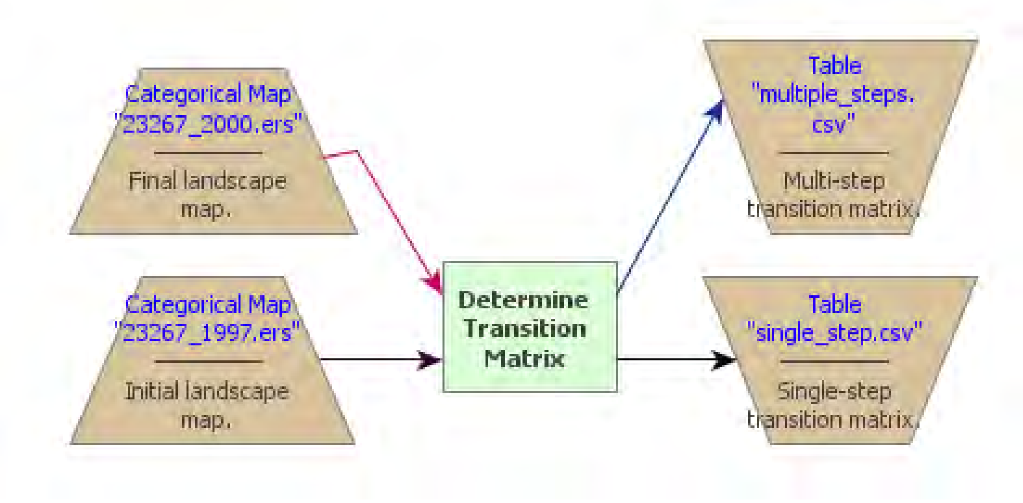
$$\begin{bmatrix} 1 \\ 2 \\ . \\ j \end{bmatrix}_{t=v} = \begin{bmatrix} P_{11} & P_{12} & P_{1.} & P_{1j} \\ P_{21} & P_{22} & P_{2.} & P_{2j} \\ P_{31.} & P_{32} & P_{33} & P_{3j} \\ P_{j1} & P_{j2} & P_{j.} & P_{jj} \end{bmatrix}^{v} * \begin{bmatrix} 1 \\ 2 \\ . \\ j \end{bmatrix}_{t=0} \quad (1)$$

The single-step matrix corresponds to a time period represented as a single time step, in turn the multiple-step matrix corresponds to a time step unit (year, month, day, etc) specified by dividing the time period by a number of time steps. For Dinamica EGO, time step can comprise any span of time, since time unit is only an external reference. A multiple-step transition matrix can only be derived from an Ergodic matrix, i.e. a matrix that has real number Eigen values and vectors.

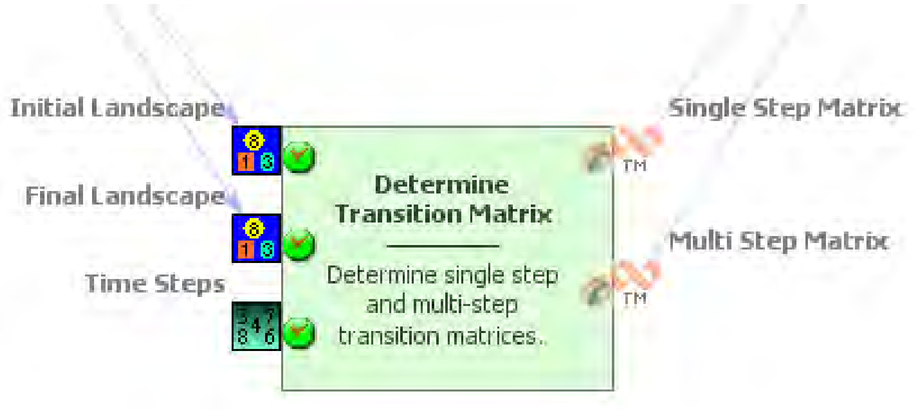
The transition rates set the net quantity of changes, that is, the percentage of land that will change to another state (land use and cover attribute), and thus they are known as net rates, being adimensional. In turn, gross rates are specified as an area unit, such as hectares or km^2 per unit of time. In the case that there is not a solution for the multiple-step transition matrix, you still can run the model in several time steps, as defined above, calculating a fixed gross rate per time step (e.g. year) by dividing the accumulated change over the period by the number of steps over which the period is composed (this might not apply to complex transition model). Dinamica EGO converts gross rates into net rate, dividing the extent of change by the fraction of each land use and cover class prior to change, before passing it to the transition functors: *Patcher* and *Expander*.

Open the model "determine_transition_matrix.xml" located in "\setup_run_and_validate_a_ lucc_model \1_transition_matrix_calculation"

This model calculates the single-step and multiple step matrices.

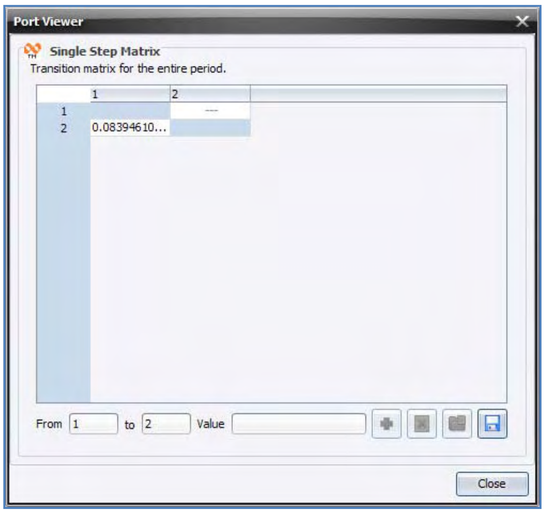

Categorical Map "23267_2000.ers"
Final landscape map.
Categorical Map "23267_1997.ers"
Initial landscape map.
Determine Transition Matrix
Table "multiple_steps.csv"
Multi-step transition matrix.
Table "single_step.csv"
Single-step transition matrix.

Open the functor *Determine Transition Matrix* with the Edit Functor Ports. **TIP:** This functor, together with others applied to calibrate the model, is found in Calibration tab.


Initial Landscape
Final Landscape
Time Steps
Determine Transition Matrix
Determine single step and multi-step transition matrices.
Single Step Matrix
Multi Step Matrix
TM
TM

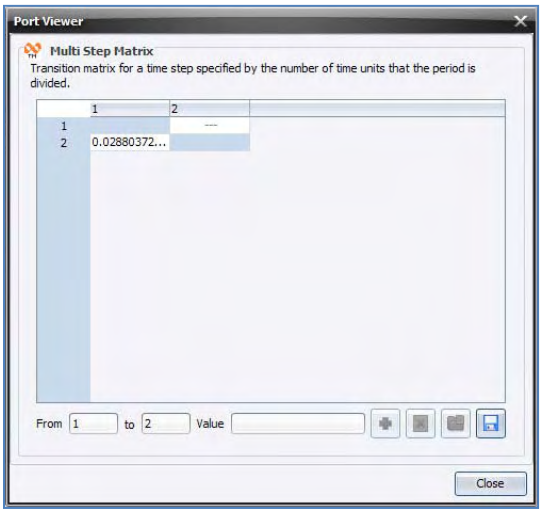
Note that *Categorical Map "2367_1997.ers"* connects to the **Initial Landscape** port and the "2367_2000.ers" to the **Final Landscape**. Check the number of time steps. In this case "3", therefore, you want to determine the multi-step matrix per annual steps (2000 – 1997 = 3 years).

Set **Register viewer** on in the output ports and run the model. Open the resulting matrices by clicking with the right button on the output ports.   Is this what you got?


Port Viewer
Single Step Matrix
Transition matrix for the entire period.

	1	2
1		---
2	0.08394610...	

From 1 to 2 Value


Port Viewer
Multi Step Matrix
Transition matrix for a time step specified by the number of time units that the period is divided.

	1	2
1		---
2	0.02880372...	

From 1 to 2 Value

You can also see the model results in the window message. You just need to browse it back. **TIP:** you can copy the results from the log window and paste them as follows:

```
Single Step Transition Matrix:

 From \ To |          1          2          3
---------------------------------------------
         1 |       XXXX         --         --
         2 |  0.0839461       XXXX         --
         3 |         --         --       XXXX

Multi Step Transition Matrix:

 From \ To |          1          2          3
---------------------------------------------
         1 |       XXXX         --         --
         2 |  0.0288037       XXXX         --
         3 |         --         --       XXXX
```

The only transition occurring is from forest (2) to deforested (1). The rates indicate that a percent of the forest is changing to deforested per unit time step, which is 3 years for the first matrix and 1 year for the latter. Thus, within this time-period deforestation is occurring at a net rate of 2.8 % per year, which means that the remaining forest is shrinking 2.8% per year. Bear in mind that like interest rates, transition rates are superimposed again and again over the stock variable, which in this case is represented by the extent of remaining forest.

The net transition matrix is passed to the simulation model and Dinamica EGO browses the landscape (land use and cover) map to count the number of cells to calculate the gross rate in terms of quantity of cells to be changed. In order to set a constant gross rate you will need to pass a variable net rate to the model, which is also possible due to the ability of Dinamica EGO to incorporate feedback into the simulation. Let's move on to the next step.

7.2 Second step: Calculating ranges to categorize gray-tone variables

The Weights of Evidence method (Goodacre *et al*. 1993; Bonham-Carter, 1994) is applied in Dinamica EGO to produce a transition probability map (fig. 10), which depicts the most favourable areas for a change (Soares-Filho *et al*. 2002, 2004).

Weights of Evidence consists of a Bayesian method, in which the effect of a spatial variable on a transition is calculated independently of a combined solution. The Weights of Evidence represent each variable's influence on the spatial probability of a transition i$\Rightarrow$j and are calculated as follows.

$$O\{D|B\} = \frac{P\{D|B\}}{P\{\overline{D}|B\}} \qquad (2)$$

$$\log\{D|B\} = \log\{D\} + W^{+} \qquad (3)$$

Where W^{+} is the Weight of Evidence of occurring event *D*, given a spatial pattern *B*. The post-probability of a transition i$\Rightarrow$j, given a set of spatial data (*B, C, D,... N*), is expressed as follows:

$$\mathrm{P}\{i \Rightarrow j|B \cap C \cap D ... \cap N\} = \frac{e^{\sum W_N^{+}}}{1 + e^{\sum W_N^{+}}} \qquad (4)$$

Where *B, C, D,* and *N* are the values of *k* spatial variables that are measured at location *x,y* and represented by its weights W^{+}_{N}

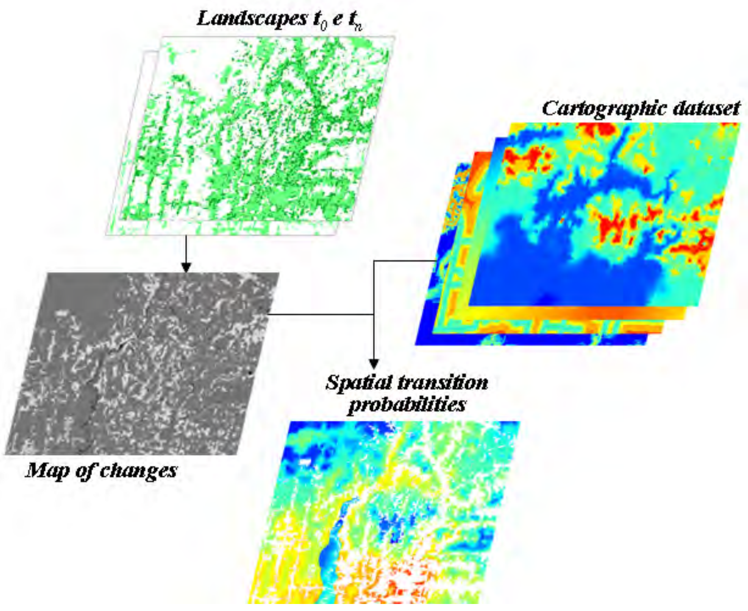


Fig. 10. Calculation and application of Weights of Evidence to produce a transition probability map.

Since Weights of Evidence only applies to categorical data, it is necessary to categorize continuous gray-tone maps (quantitative data, such as distance maps, altitude, and slope). A key issue to any categorization process concerns the preservation of the data structure. The present method adapted from Agterberg & BonhamCarter (1990), calculates ranges according to the data structure by first establishing a minimum delta – specified as the increment in the graphical interface – (*Dx*) for a continuous gray-tone variable x that is used to build n incremental buffers (*Nx*) comprising intervals from $x_{minimum}$ to $x_{minimum}$ + *nDx*. Each *n* defines a threshold that divides the map into two classes: (*Nx*) and ($\overline{Nx}$). *An* is the number of cells for a buffer (*Nx*) multiple of *n* and *dn* is the number of occurrences for the modeled event (*D*) within this buffer. The quantities An and dn are obtained for an ordered sequence of buffers *N(xminimum + nDx)*. Subsequently, values of W^+ for each buffer are calculated using equations 2 to 4. A sequence of quantities *An* is plotted against $An * \exp(W^+)$. Thereafter breaking points for this graph are determined by applying a line-generalizing algorithm (Intergraph, 1991) that contains three parameters: 1) minimum distance interval along *x*, *mindx*, 2) maximum distance interval along *x*, *maxdx*, and 3) tolerance angle *ft*. For *dx* (a distance between two points along *x*) between *mindx* and *maxdx*, a new breaking point is placed whenever *dx >= maxdx* (an angle between *v* and *v'*- vectors linking the current to the last point and the last point to its antecedent, respectively) exceeds the tolerance angle *ft*. Thus, the number of ranges decreases as a function of *ft*. The ranges are finally defined by linking the breaking points with straight lines. Note that *An* is practically error-free whereas *dn* is subject to a considerable amount of uncertainty because it is regarded as the realization of a random variable. Since small *An* can generate noisy values for W^+, Goodacre *et al.* (1993) suggest that, instead of calculating it employing equations 2 and 4, one should estimate W^+ for each defined range through the following expression:

$$W^+ = \ln\left(\frac{y_{n=k} - y_{n=k-1}}{A_{n=k} - A_{n=k-1}}\right) \quad (5)$$

where $y_n = An * \exp(W^+)$ and *k* represents the breakpoints defined for the *n* increments of *Dx* (Fig.11).

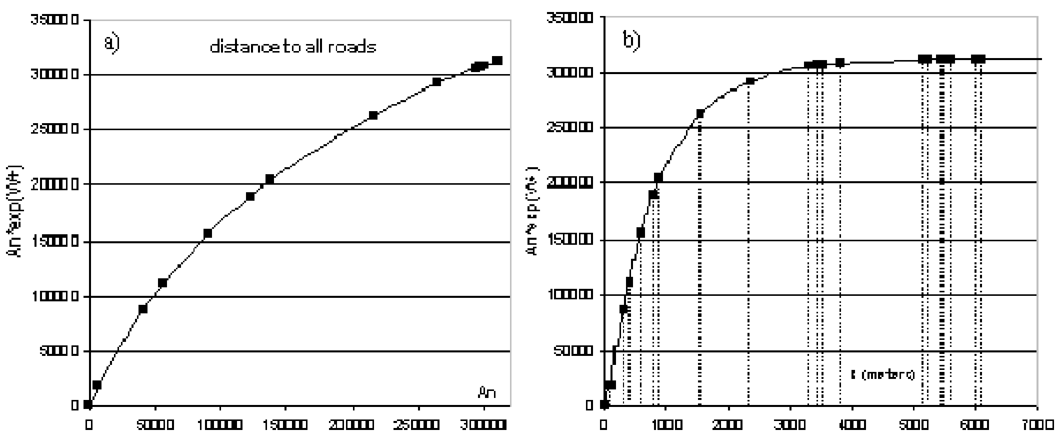
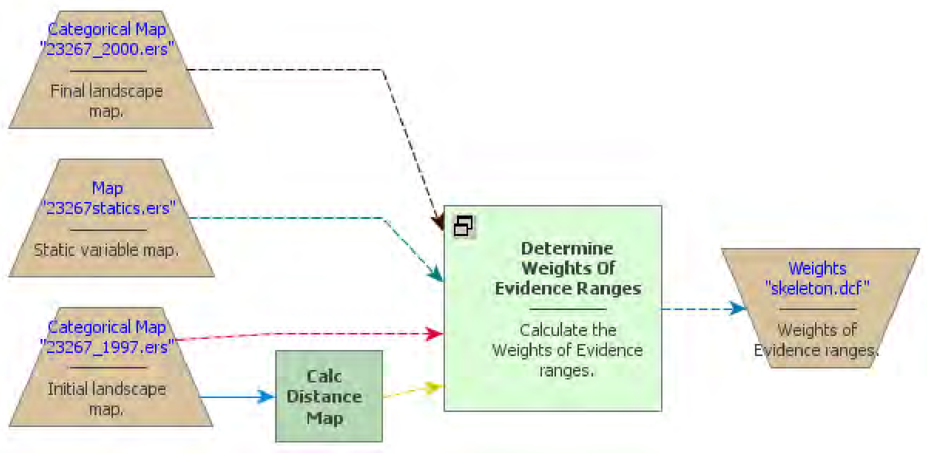


Fig. 11 a) Plot of *An* against for the variable "distance to all roads".

The best-fitting curve can be approximated by a series of straight-line segments using a line-generalizing algorithm as explained in the text. This approach is used to define the breaking points for this curve and subsequently category intervals for a continuous variable (b).

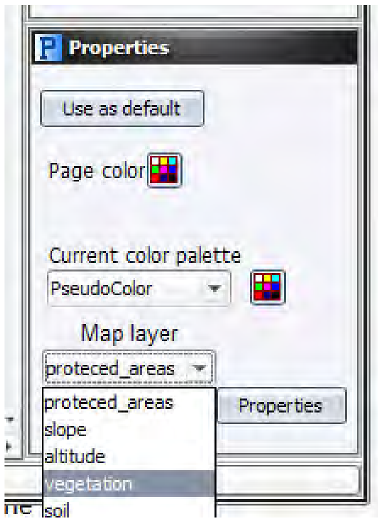
Now, open the model "determine_weights_of_evidence_ranges.xml" located in "\setup_run_and_validate_a_lucc_model\2_weights_of_evidence_ranges_calculation".

This model calculates ranges in order to categorize continuous gray-tone variables for deriving the Weights of Evidence. It selects the number of intervals and their buffer sizes aiming to better preserve the data structure. See the help for a further description of this method. As a result, its output is used as input for the calculation of Weights of Evidence coefficients.

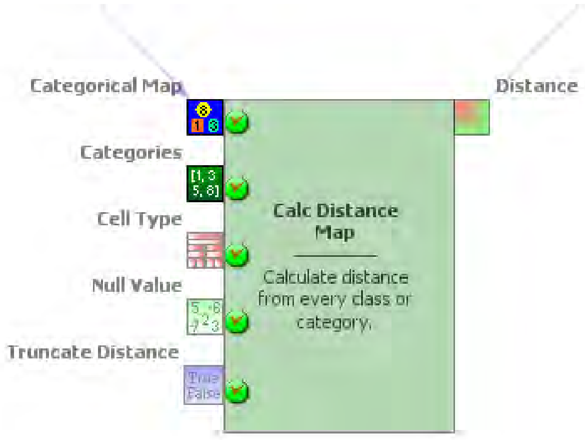

Categorical Map "23267_2000.ers"
Final landscape map.
Map "23267statics.ers"
Static variable map.
Categorical Map "23267_1997.ers"
Initial landscape map.
Calc Distance Map
Determine Weights Of Evidence Ranges
Calculate the Weights of Evidence ranges.
Weights "skeleton.dcf"
Weights of Evidence ranges.

In addition to the initial and final landscape maps, this model receives a raster cube composed of a series of static maps, e.g. vegetation, soil, altitude (they are named so because they do not change during model iteration. A raster cube encompasses a set of co-registered map layers.

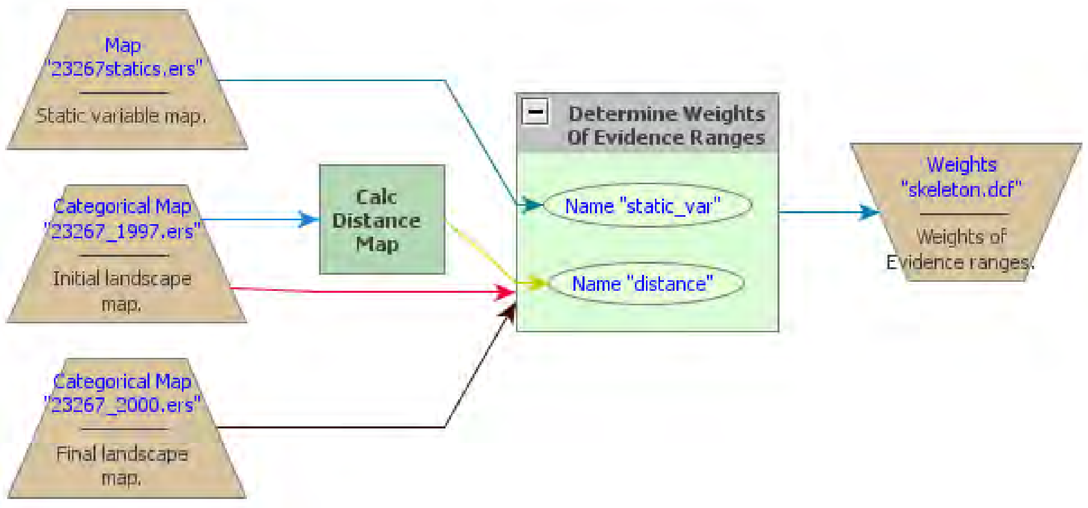
Open the file "23267static.ers" from "\setup_run_and_validate_a_lucc_model\originals" on the Map Viewer. An option to select the layer will appear on the bottom left of the Map Viewer. Change the layer to examine the other maps. **TIP:** you can build a raster cube assembling a set of co-registered raster maps through the functor Create Cube Map and extract a layer from the cube using the functor *Extract Map Layer*. Cube raster data are only supported in ER format.



Furthermore, Dinamica EGO can incorporate dynamic layers into the simulation, which are so-called because they are updated during model iteration. For this model you will include the variable "distance to previously deforested areas" as a dynamic map. For this purpose, the model employs the functor *Calc to Distance Map***.** Open it with the Edit Functor Ports.



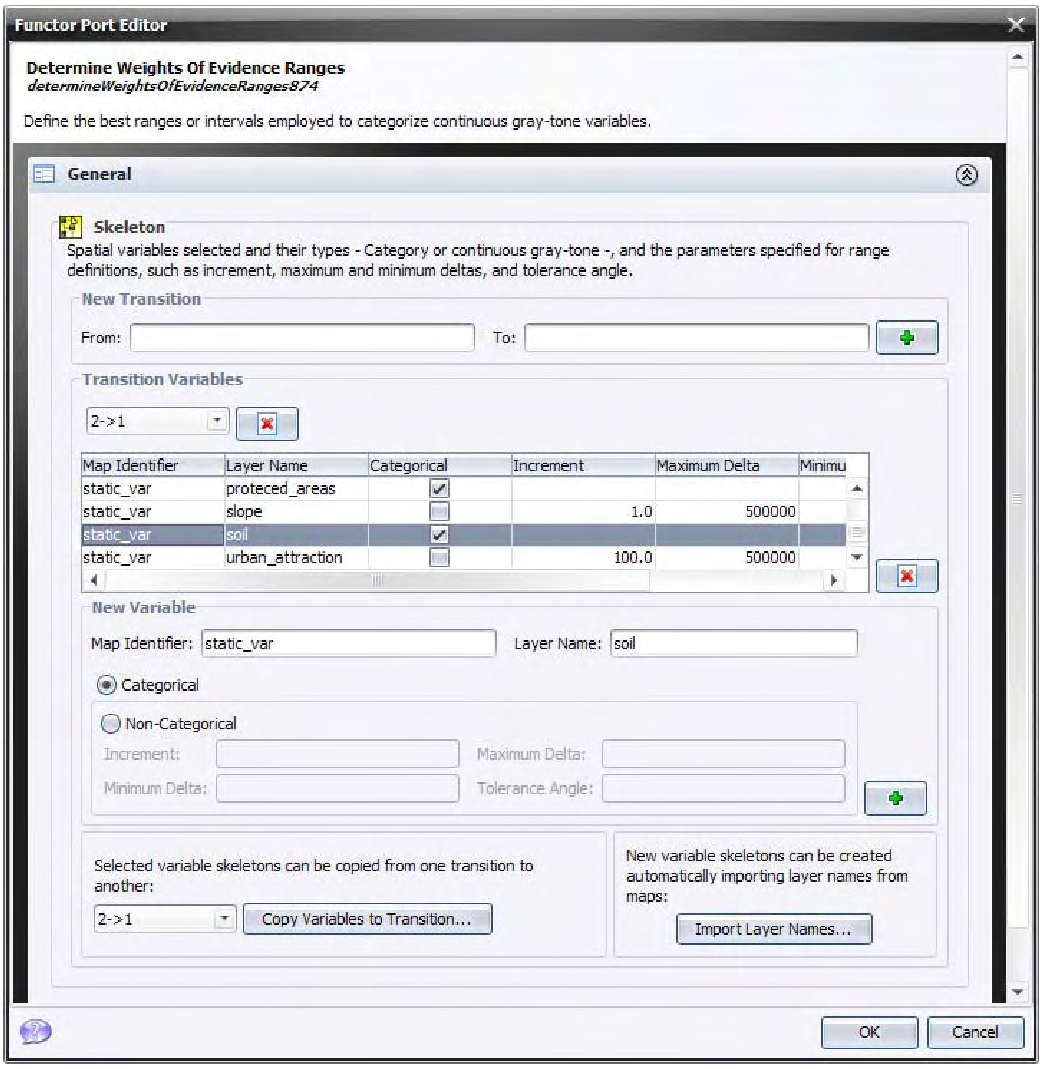
This functor receives as input a categorical map, in this case the landscape map. Click now on the **Categories** port. This functor generates a map of frontage distance (nearest distance) from the cells of each map class of category defined by the user. In this case class "1" represents deforested land. Thus, the model takes into account the proximity of the previously deforested land on the probability of new deforestation. Now open the container Determine *Weights Of Evidence Ranges* clicking on the icon on its top left side.



Instead of *Number Map*, now you find *Name Map* within this container. This functor is applied to containers that need a map name or alias to identify the maps passed to them. *Name Map* is found in the Map Algebra Supplementary tab. This can be any name, but you must be consistent, therefore using the same names when setting the container internal parameters, as shown below. Examples of containers that need *Name Map* are *Determine Weights Of EvidenceRanges*, *Determine Weights Of Evidence Coefficients*, and *Calc WOfE Probability Map*.

There are two *Name Map* functors within this container, one for the *Map "*23267statitcs.ers" and another for the distance map output from the *Calc Distance Map*.

Now open *Determine Weights Of EvidenceRanges* with the Edit Functor. Resize the window as follows:



Functor Port Editor

Determine Weights Of Evidence Ranges
determineWeightsOfEvidenceRanges874

Define the best ranges or intervals employed to categorize continuous gray-tone variables.

General

Skeleton
Spatial variables selected and their types - Category or continuous gray-tone -, and the parameters specified for range definitions, such as increment, maximum and minimum deltas, and tolerance angle.

New Transition

From: To:

Transition Variables

2->1

Map Identifier	Layer Name	Categorical	Increment	Maximum Delta	Minimu
static_var	proceed_areas	✓			
static_var	slope		1.0	500000	
static_var	soil	✓			
static_var	urban_attraction		100.0	500000	

New Variable

Map Identifier: static_var Layer Name: soil

(●) Categorical

( ) Non-Categorical

Increment: Maximum Delta:

Minimum Delta: Tolerance Angle:

Selected variable skeletons can be copied from one transition to another:

2->1 Copy Variables to Transition...

New variable skeletons can be created automatically importing layer names from maps:

Import Layer Names...

OK Cancel

Note that *Name Map* distance has layer named "distance_to_1" and static_var has a series of layers, each one representing a cartographic variable.

TIP: Of the three supported file formats only ER Mapper format supports names for layers. This is a BIL (Band Interleaved) format with a separate ASCII header file.

The raster cube contains the layers altitude, d_all_roads (d means distance), d_major_rivers, d_paved_roads, d_settlement, d_trans_rivers, protected_areas, slope, urban_attraction (a potential interaction map), and vegetation.

TIP: Layer names must be the same as the ones specified in the header file. Dinamica EGO converts Geotiff into ER Mapper format, you just need to connect *Load Map* to *Save Map*.

Protected areas, vegetation and soil are already categorical data, thus mark them as such. The others will need to be categorized. The parameters for the categorization process are the increment – the map unit minimum buffer increment, e.g. in meters or degrees (**TIP:** in case of the map of distance, this will be equivalent to the cell resolution), the minimum and maximum deltas representing intervals on the Y axis of the graphs in fig. 11, and the tolerance angle, which measures the angle of deviation from a straight line. Some default values are suggested. The output of this functor will be a Weights of Evidence skeleton file, showing the categorization ranges, but with all weights set to zero. Run the model and open the output file with a text editor. **TIP:** if the model has more than one transition, you can copy and paste the range parameters on the other transitions' windows. Dinamica EGO enables defining different

```
:static_var/soil    5:9    9:10    10:11    11:13    13:14
2,1    0    0    0    0    0
:static_var/urban_attraction    0:100    100:200    200:1900
2,1    0    0    0
:static_var/vegetation    4:5    5:15    15:17    17:18
2,1    0    0    0    0
```

ranges for each transition.

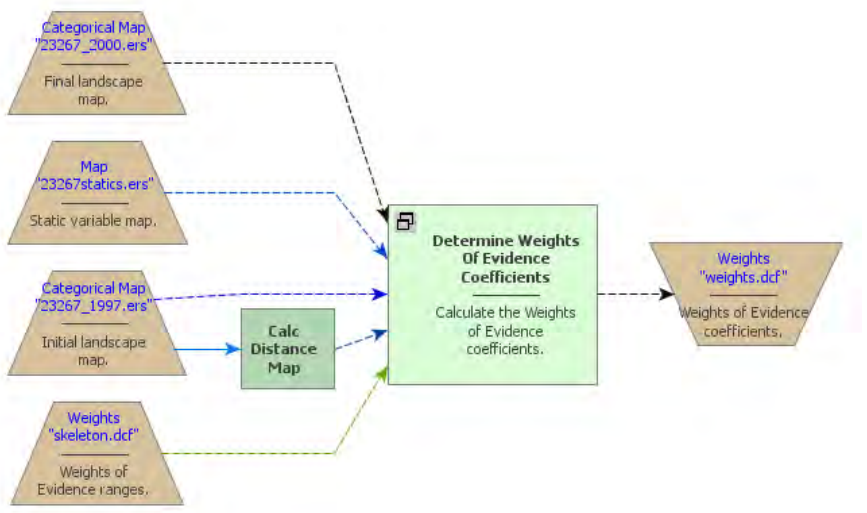
Example of Skeleton structure for Weights of Evidence

The first line contains the ranges and the second the transition and their respectively Weights of Evidence coefficients, which are still set to zero.

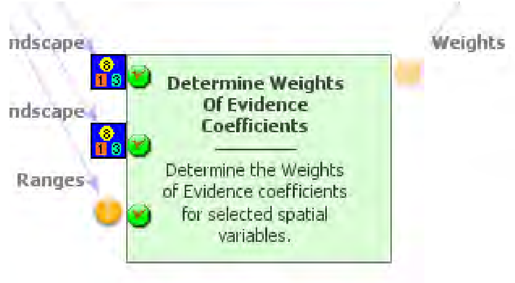
Although the mathematical fundamentals of this module may be a little hard to grasp at the first sight, it provides an easy means to handle models with multi-states and transitions. Let's move on to the Weights of Evidence calculation.

7.3 Third step: Calculating Weights of Evidence coefficients

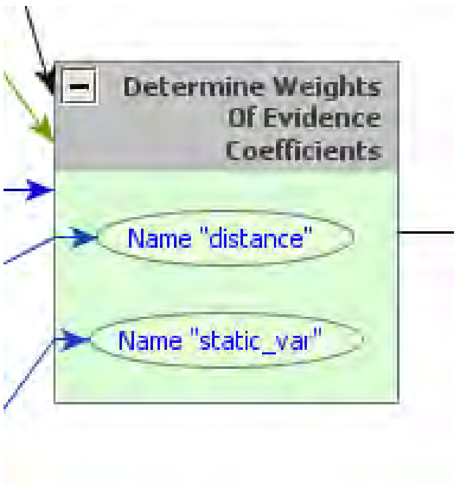
Open the model "determine_weights_of_evidence_coefficients.xml" located in setup_run_and_validate_a_lucc_model\3_weights_of_evidence_coefficient_calculation.


Categorical Map "23267_2000.ers"
Final landscape map.
Map "23267statics.ers"
Static variable map.
Categorical Map "23267_1997.ers"
Initial landscape map.
Weights "skeleton.dcf"
Weights of Evidence ranges.
Calc Distance Map
Determine Weights Of Evidence Coefficients
Calculate the Weights of Evidence coefficients.
Weights "weights.dcf"
Weights of Evidence coefficients.

The same dataset of the previous step is applied again, plus the WEOFE skeleton, which is loaded through the *Load Weights* functor located in the input/output tab. There is no parameter to set in the *Determine Weights Of Evidence Coefficients* functor, you only need to set the proper links as follows.

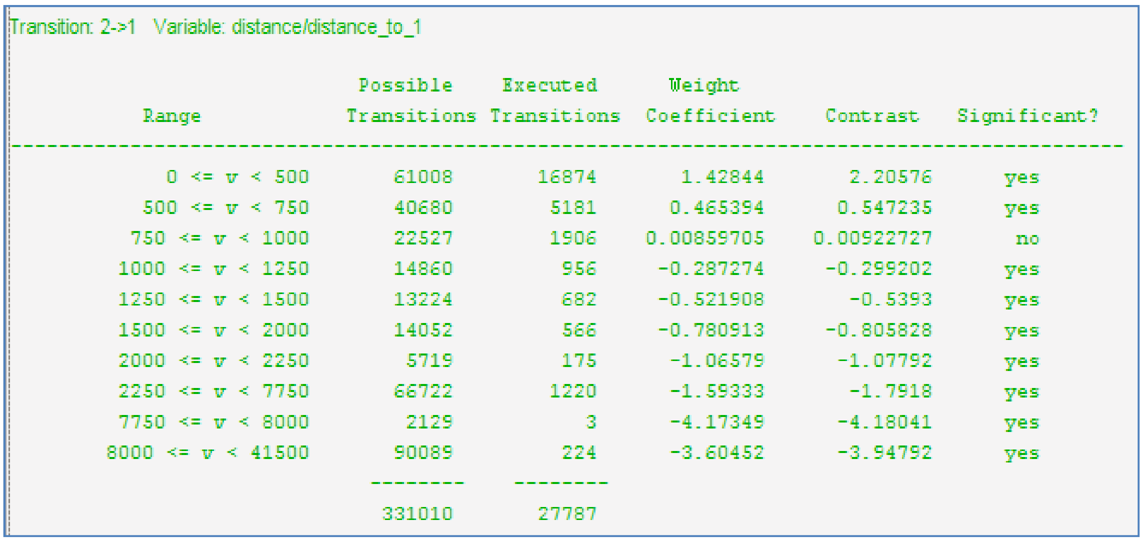

ndscape
ndscape
Ranges
Determine Weights Of Evidence Coefficients
Determine the Weights of Evidence coefficients for selected spatial variables.
Weights

As well as the input maps:



TIP: Use always the same names in *Name Map*.

Maximize the log window and run the model, let's analyze the results for variable "distance_to_1".



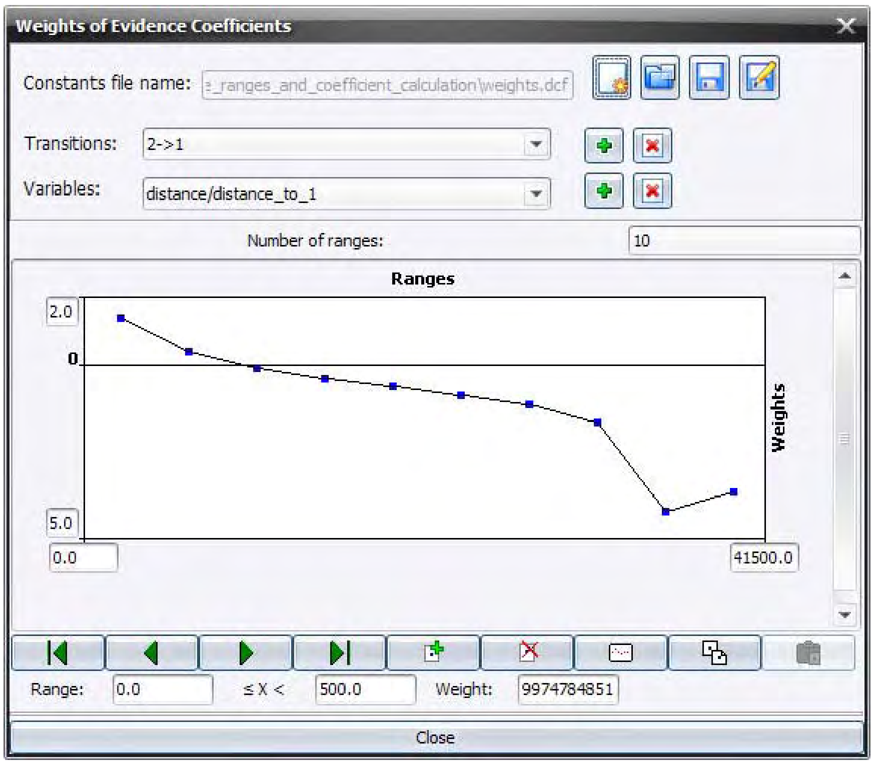
```
Transition: 2->1   Variable: distance/distance_to_1

                                Possible     Executed      Weight
         Range                 Transitions  Transitions   Coefficient    Contrast   Significant?
--------------------------------------------------------------------------------------------
            0  <= v <  500        61008        16874       1.42844       2.20576       yes
          500  <= v <  750        40680         5181      0.465394      0.547235       yes
          750  <= v < 1000        22527         1906    0.00859705    0.00922727        no
         1000  <= v < 1250        14860          956     -0.287274     -0.299202       yes
         1250  <= v < 1500        13224          682     -0.521908       -0.5393       yes
         1500  <= v < 2000        14052          566     -0.780913     -0.805828       yes
         2000  <= v < 2250         5719          175      -1.06579      -1.07792       yes
         2250  <= v < 7750        66722         1220      -1.59333       -1.7918       yes
         7750  <= v < 8000         2129            3      -4.17349      -4.18041       yes
         8000  <= v < 41500       90089          224      -3.60452      -3.94792       yes
                                --------     --------
                                 331010        27787
```

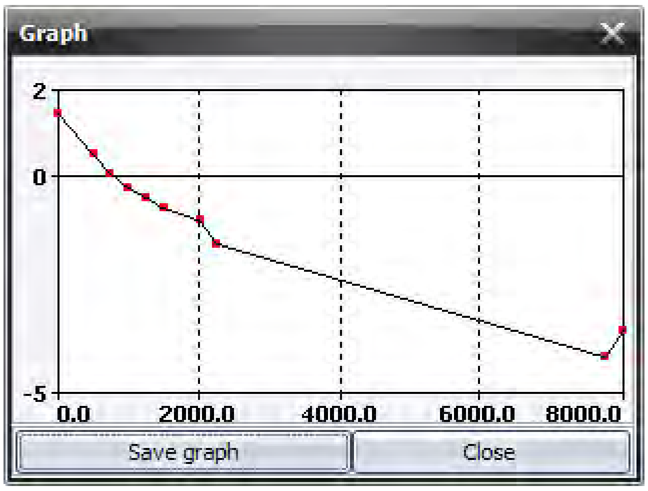
The first column shows the ranges, the second the buffer size in cells, the third the number of transitions occurring within each buffer, the fourth the obtained coefficients, the fifth the Contrast measure and the last the result for the statistical significance test. Go to help for further details. **TIP:** Dinamica EGO Help brings a detailed description of more complex algorithms, such the ones employed in calibration and validation functors.

Notice that the first ranges show a positive association, favoring deforestation, especially the first, in contrast, the final ranges show negative values, thus repelling deforestation. The middle range shows values close to zero, meaning that these distance ranges do not exert an effect on deforestation.

The Contrast measures the association/repelling effect. Near zero, there is no effect at all, whereas the larger and more positive it becomes, the greater is the attraction; on the other hand, the larger and more negative the value, the greater is the repelling effect.

Now open the graphical Weights of Evidence editor within the *Save Weights* functor (eye icon).


Weights of Evidence Coefficients
Constants file name: ≥_ranges_and_coefficient_calculation\weights.dcf
Transitions: 2->1
Variables: distance/distance_to_1
Number of ranges: 10
Ranges
2.0
0
5.0
0.0
41500.0
Weights
Range: 0.0 ≤ X < 500.0 Weight: 9974784851
Close



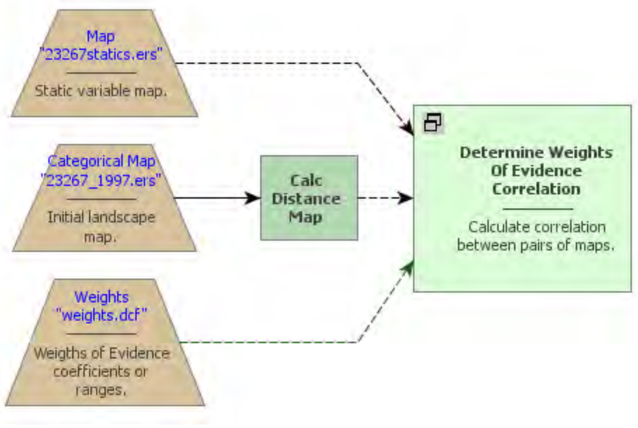
You can graphically edit the Weights of Evidence coefficients and also get a view of a continuous Weights of Evidence function, clicking on the bird view button. Notice how deforestation likelihood varies as a function of distance to previously deforested areas.

7.4 Fourth step: Analyzing map correlation

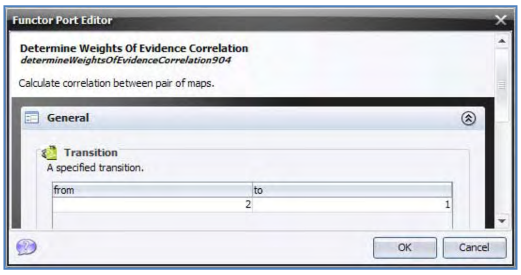
The only assumption for the Weights of Evidence method is that the input maps have to be spatially independent. A set of measures can be applied to assess this assumption, such as the Cramer test and the Joint-Uncertainty Information (Bonham-Carter, 1994). As a result, correlated variables must be disregarded or combined into a third that will replace the correlated pair in the model.

Open model "weights_of_evidence_correlation.xml" in setup_run_and_validate_a_lucc_ model\4_weights_of_evidence_correlation.

The model in 3_and_2_weights_of_evidence_ranges_and_coefficient_calculation corresponds to steps 2 and 3 joined together.



This model performs pairwise tests for categorical maps in order to test the independence assumption. Methods employed are the *Chi^2*, *Crammers,* the *Contingency*, the *Entropy* and the *Uncertainty Joint Information* (Bonham-Carter, 1994). In addition to the links to be connected, the only parameter to be set in the *Determine Weights of Evidence Correlation* is the transition as follows:

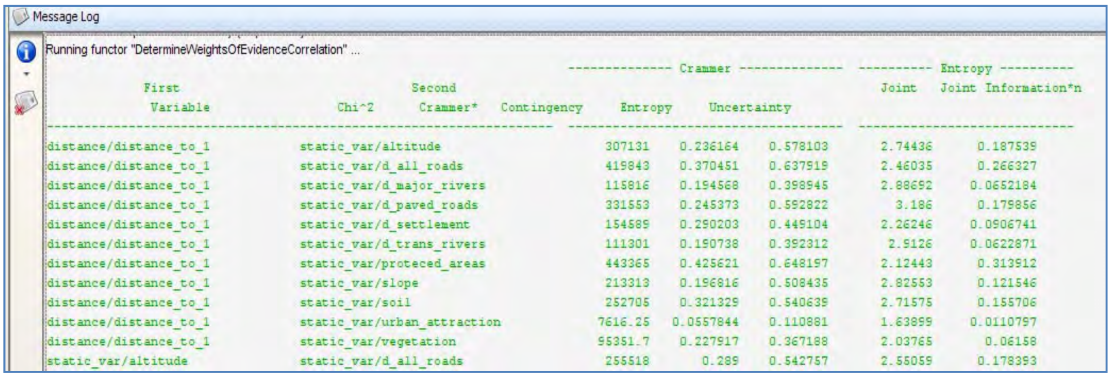


Determine Weights Of Evidence Correlation
determineWeightsOfEvidenceCorrelation904
Calculate correlation between pair of maps.

Transition
A specified transition.

from	to
2	1

Before running the model, maximize the log window. This is a part of the log reported:



```
Running functor "DetermineWeightsOfEvidenceCorrelation" ...
                                                                              -------------- Crammer -------------- ---------- Entropy ----------
                First                                Second                                                           Joint   Joint Information*n
               Variable                   Chi^2     Crammer*   Contingency     Entropy     Uncertainty
---------------------------------------------------------------------------- ------------------------------------- -----------------------------
distance/distance_to_1           static_var/altitude               307131    0.236164    0.578103       2.74436       0.187539
distance/distance_to_1           static_var/d_all_roads            419843    0.370451    0.637919       2.46035       0.266327
distance/distance_to_1           static_var/d_major_rivers         115816    0.194568    0.398945       2.88692      0.0652184
distance/distance_to_1           static_var/d_paved_roads          331553    0.245373    0.592822         3.186       0.179856
distance/distance_to_1           static_var/d_settlement           154589    0.290203    0.449104       2.26246      0.0906741
distance/distance_to_1           static_var/d_trans_rivers         111301    0.190738    0.392312        2.5126      0.0622871
distance/distance_to_1           static_var/proteced_areas         443365    0.425621    0.648197       2.12443       0.313912
distance/distance_to_1           static_var/slope                  213313    0.196816    0.508435       2.82553       0.121546
distance/distance_to_1           static_var/soil                   252705    0.321329    0.540639       2.71575       0.155706
distance/distance_to_1           static_var/urban_attraction      7616.25   0.0557844    0.110881       1.63899      0.0110797
distance/distance_to_1           static_var/vegetation            95351.7    0.227917    0.367188       2.03765        0.06158
static_var/altitude              static_var/d_all_roads            255518       0.289    0.542757       2.55059       0.178393
```

TIP: it would be nice to copy the log, place it in a text editor and then export it to a spreadsheet. In the spreadsheet, you could set up a cross table showing all pairs of variables.

Let's check for correlated pairs of maps. The following pair draws our attention:

	Chi^2	Crammer*	Contingency	Entropy	Uncertainty Joint I.
d_major_rivers -> d_trans_rivers	2.80782e+06	0.958014	0.906132	1.62982	0.885453

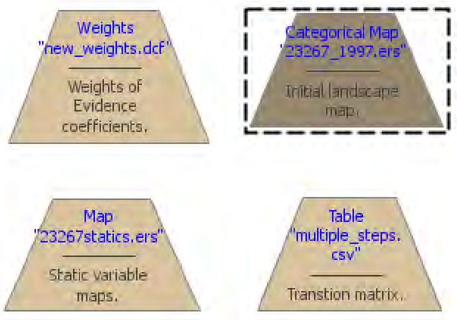
Although there is no agreement on what threshold should be used to exclude a variable, all tests highlight a high correlation for this pair of variables. Hence you must exclude one of these. Let's remove d_major_rivers. Delete the variable from the Weights of Evidence file using its graphical editor. Open it on the *Load Weights* using the eye icon.

Now save the WEOFE coefficients as "new_weights.dcf" in the folder \setup_run_and_validate_a_lucc_model\4_weights_of_evidence_correlation

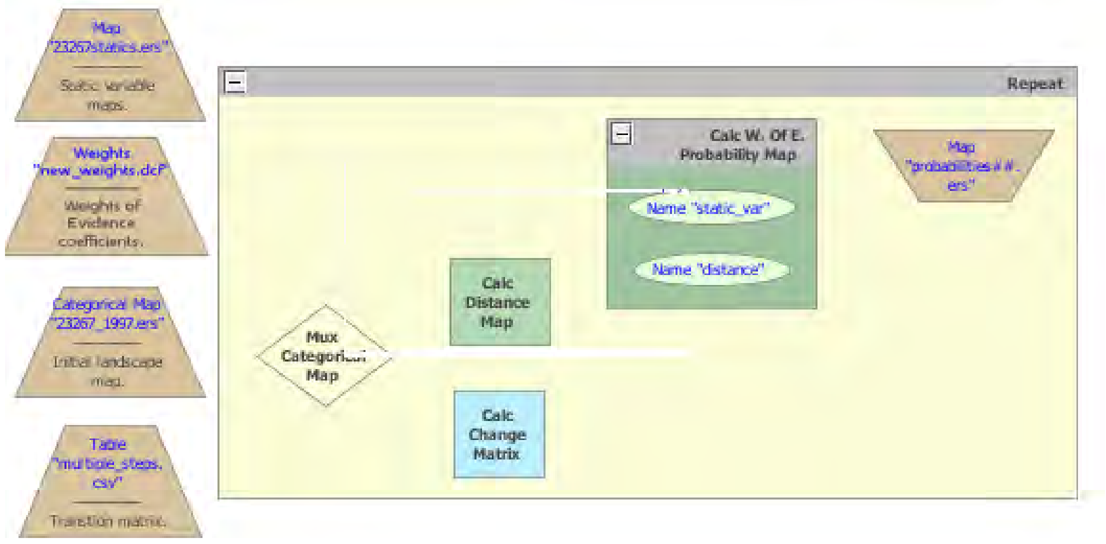
Great, you have come through the calibration process, now you can start setting up the simulation model. Let's move forward.

7.5 Fifth step: Setting up and running a LUCC simulation model

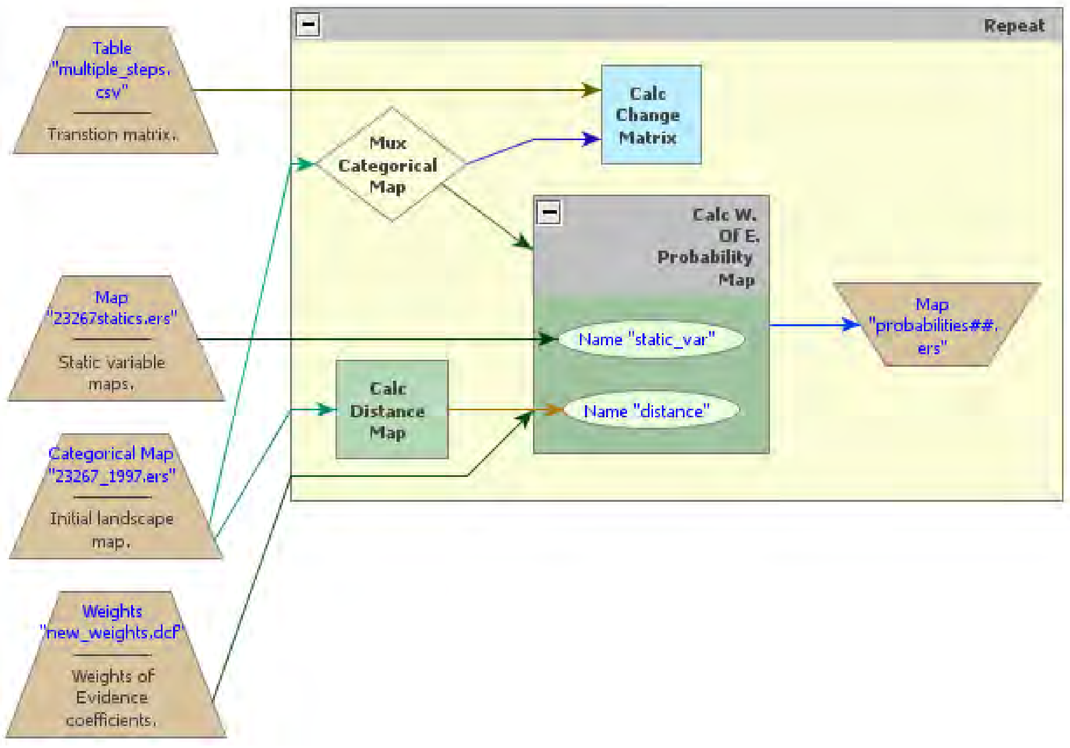
Let’s start setting up the deforestation simulation model by loading the input data. You will need *Load Categorical Map* to load the initial landscape: “original/23267_1997.ers”, *Load Map* for “originals/23267statics.ers”, *Load Weights* for “new_weights.dcf”, and *Load Lookup Table* for the multi-step transition matrix: “multiple_steps.csv” because you will run the model in annual time-steps. Add the following comments to each functor:



Now, drag a *Repeat*, and place within it one *Calc Distance Map*, one *Mux Categorical Map* from Control tab, one *Calc Weights of Evidence Probability Map* and one *Save Map*. Also place two *Name Map* functors within *Calc Weights of Evidence Probability Map*. Rename them with the same alias as the second and third steps. Open the *Save Map* and write “probabilities.ers” in the folder \5_run_lucc. Leave the option **Suffix** = “2”.

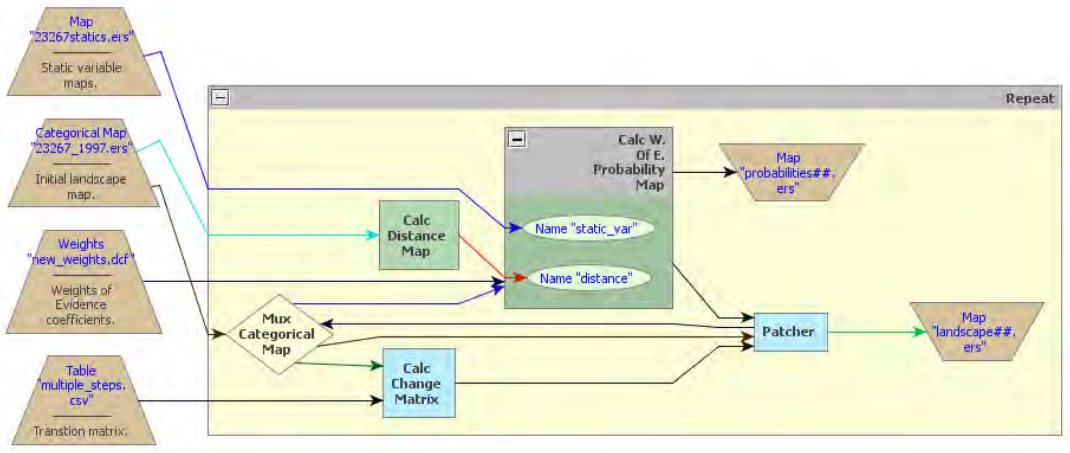


Let’s connect the functors: first *Categorical Map “*23267_1997.ers” to the **initial** port of the *Mux Categorical Map*, the output from this to *Calc Distance Map, Calc Change Matrix* and *Calc W. Of E. Probability Map*. Connect *Table “*multiple_steps.csv*”* to *Calc Change Matrix*, *Map “*23267statics.ers*”* to *Name “*static_var*”*, the output from *Calc Distance Map* to *Name “*distance*”*, *Weights “*new_weigths.dcf*”* to *Calc W. Of E. Probability Map* and its output to *Map “*probabities##.ers*”* (Note that the suffix ## will receive the step from the model iteration).



Now place two additional functors, *Patcher* from the Simulation tab and another *Save Map*. Enter "Landscape.ers" and leave "2" as **Suffix**. Connect the output from *Mux Categorical Map* to the *Patcher* **Landscape** port, the output from *Calc W. Of E. Probability Map* to the *Patcher* **Probabilities** port, and the output from *Calc Changes* to the **Changes** port of the *Patcher*.

To close the loop you will need to connect the output port **Changed Landscape** of *Patcher* to the port **Feedback** of *Mux Categorical Map* and to *Map "*Landscape##.ers" to save the maps from model execution. The model should now look like this:



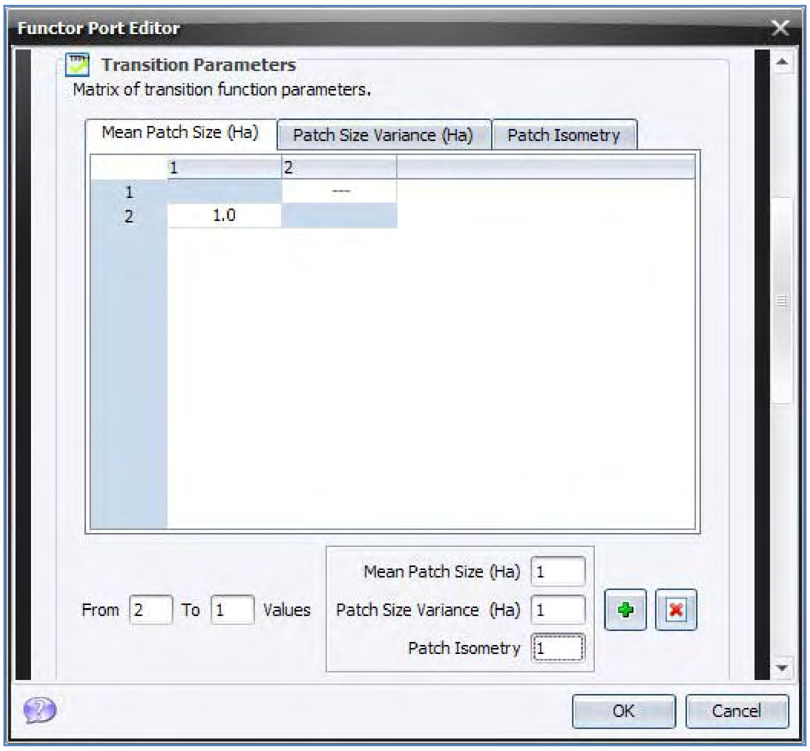
The functor *Mux Categorical Map* enables dynamic update of the input landscape map. It receives the *Categorical Map "*23267_1997.ers*"* in its **Initial** port in the beginning of the simulation and thereafter the map output from *Patcher* via the **Feedback** port.

The functor *Calc W. OF. E. Probability Map* calculates a transition probability map for each specified transition by summing the Weights of Evidence, using equation 4.

In turn, the *Calc Change Matrix* receives the transition matrix, composed of net rates, and uses it to calculate crude rates in terms of quantity of cells to be changed by multiplying the transition rates by the number of cells available for a specific change.

Dinamica EGO uses as a local Cellular Automata rule a transition engine composed of two complementary transition functions, the *Expander* and the *Patcher,* especially designed to reproduce the spatial patterns of change (both are found in the Simulation tab). The first process is dedicated only to the expansion or contraction of previous patches of a certain class, while the second process is designed to generate or form new patches through a seeding mechanism. The *Patcher* searches for cells around a chosen location for a joint transition. The process is started by selecting the core cell of the new patch and then selecting a specific number of cells around the core cell, according to their *Pij* transition probabilities.

By varying their input parameters, these functions enable the formation of a variety of sizes and shapes of patches of change. The *Patch* **Isometry** varies from 0 to 2. The patches assume a more isometric form as this number increases. The sizes of change patches are set according to a lognormal probability distribution. Therefore, it is necessary to specify the parameters of this distribution represented by the mean and variance of the patch sizes to be formed. Open this functor with the Edit Functor and enter its parameters as follows (leaving the others untouched):


Functor Port Editor

Transition Parameters

Matrix of transition function parameters.

Mean Patch Size (Ha) | Patch Size Variance (Ha) | Patch Isometry

	1	2
1		---
2	1.0	

From 2 To 1 Values

Mean Patch Size (Ha) 1

Patch Size Variance (Ha) 1

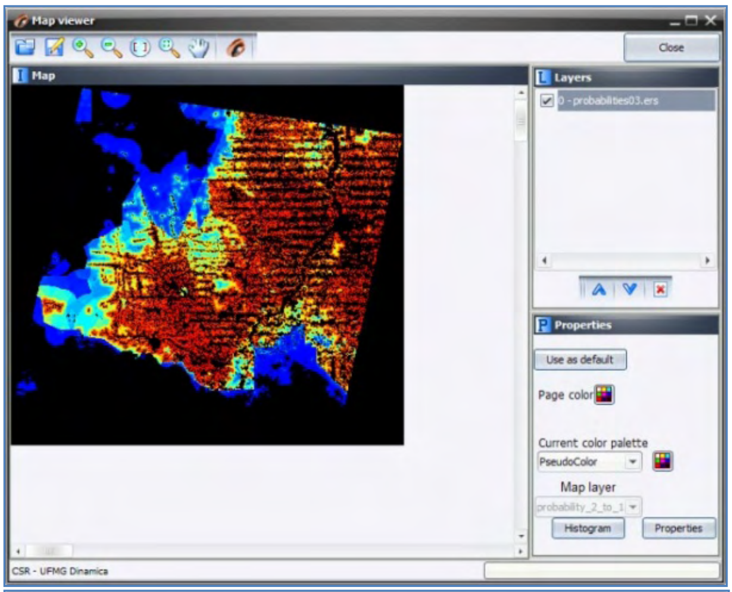
Patch Isometry 1

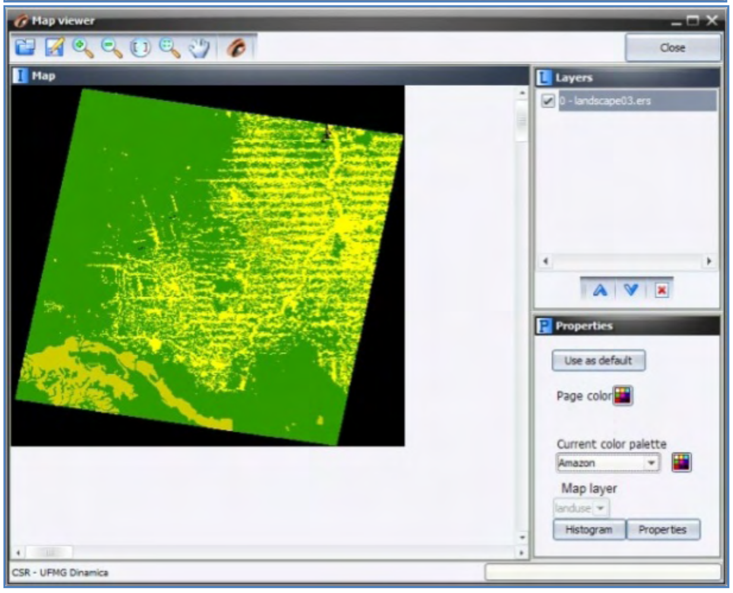
OK Cancel

Since the input map resolution is approximately 250 meters, setting the **Mean Patch** and **Patch Size Variance** to “1” won’t allow the formation of patches.

Now let’s set the other functors’ parameters. Open *Calc Distance Map* and enter “1”. Remember that you want “distance_to_1” (deforested areas). Open *Calc W. Of E. Probability Map* and enter transition “2 to 1”, which represents deforestation. Finally open *Repeat* and set the **Number of Iterations** to “3”, as you will run the model in annual time steps.

Finally verify the model and run it. Check the log reported and open the maps “probabilities3.ers” using “PseudoColor” and “landscape3.ers” using “Amazon” **Color Palette** on the Map Viewer. **TIP:** When *Save Map* is placed within *Repeat*, it saves one output map per time step, if you want only the final simulated landscape, drag it out of *Repeat*).





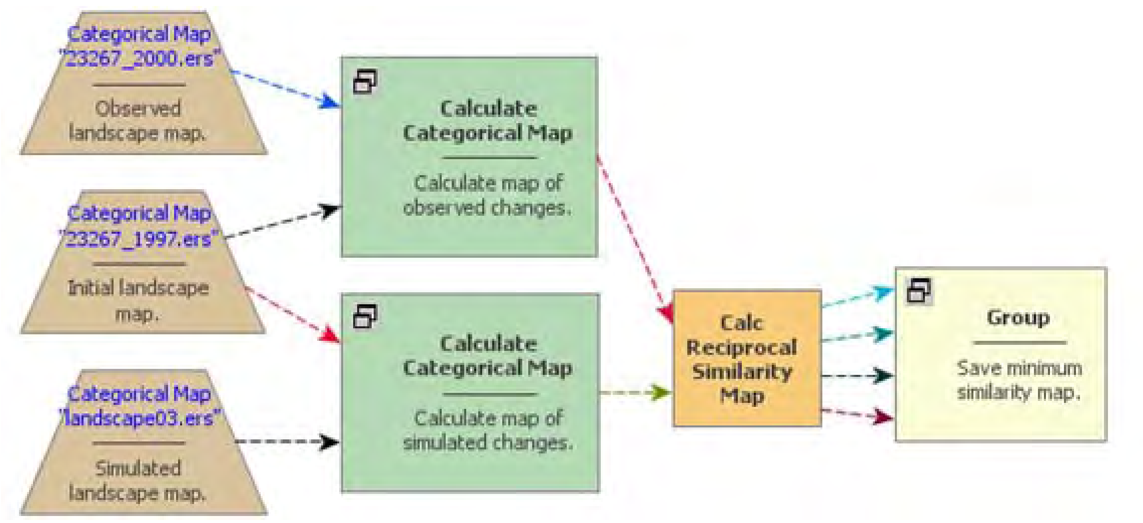
Observe the high probability areas for deforestation and compare the simulated map “landscape3.ers” with the final landscape “23267_2000.ers”. Do they resemble each other? In order to perform a quantitative comparison, let’s move on to the next step.

7.6 Sixth step: Validating simulation using an exponential decay function

Spatial models require a comparison within a neighborhood context, because even maps that do not match exactly cell-by-cell could still present similar spatial patterns and likewise spatial agreement within a certain cell vicinity. To address this issue several vicinity-based comparison methods have been developed. For example, Costanza (1989) introduced the multiple resolution fitting procedure that compares a map fit within increasing window sizes. Pontius (2002) presented a method similar to Costanza (1989) that differentiates errors due to location and quantity. Power et al. (2001) provided a comparison method based on hierarchical fuzzy pattern matching. In turn, Hagen (2003) developed new metrics, including the Kfuzzy, considered to be equivalent to the Kappa statistic, and the fuzzy similarity which takes into account the fuzziness of location and category within a cell neighborhood.

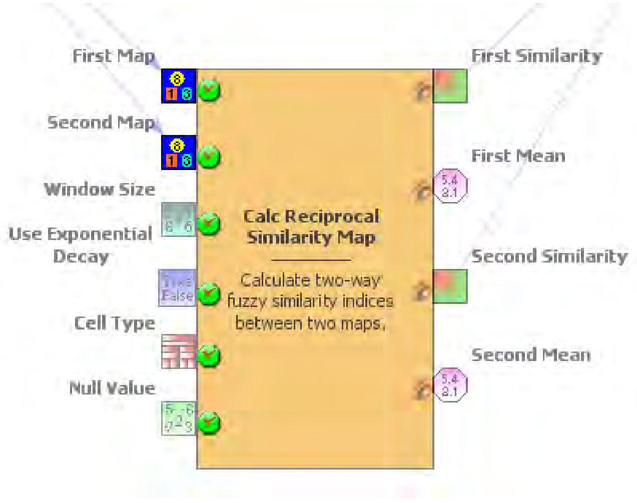
The method we apply here is a modification of the latter and named in Dinamica EGO as *Calc Reciprocal Similarity*. This method employs an exponential decay function with distance to weight the cell state distribution around a central cell (See scheme in fig.12 and then go to Help for further details on this method).

Open the model "determine-similarity-of-differences.xml" in \6_validate_using_exponential_decay_function folder.



Categorical Map "23267_2000.ers"
Observed landscape map.
Categorical Map "23267_1997.ers"
Initial landscape map.
Categorical Map "landscape03.ers"
Simulated landscape map.
Calculate Categorical Map
Calculate map of observed changes.
Calculate Categorical Map
Calculate map of simulated changes.
Calc Reciprocal Similarity Map
Group
Save minimum similarity map.

As inputs, the model receives the initial and final landscapes, and the final simulated landscape. Notice that there are two *Calculate Categorical Map* functors before the *Calc Reciprocal Similarity Map*. Because simulated maps inherit the spatial patterns of the initial landscape map, to remove this inheritance, this model evaluates the spatial fit between maps of changes.



Open the *Calculate Categorical Map*, you will find the following equation:

if (i1 = i2) then null else i2

Therefore, the resulting map only depicts the cells that have changed. *Calc Reciprocal Similarity Map* calculates a two-way similarity, from the first map to the second and from the second to the first. It is advisable to always choose the smaller similarity value since random maps tend to produce an artificially high fit when compared univocally, because they spread the changes all over the map. This test employs an exponential decay function truncated outside of a window size 11x11.

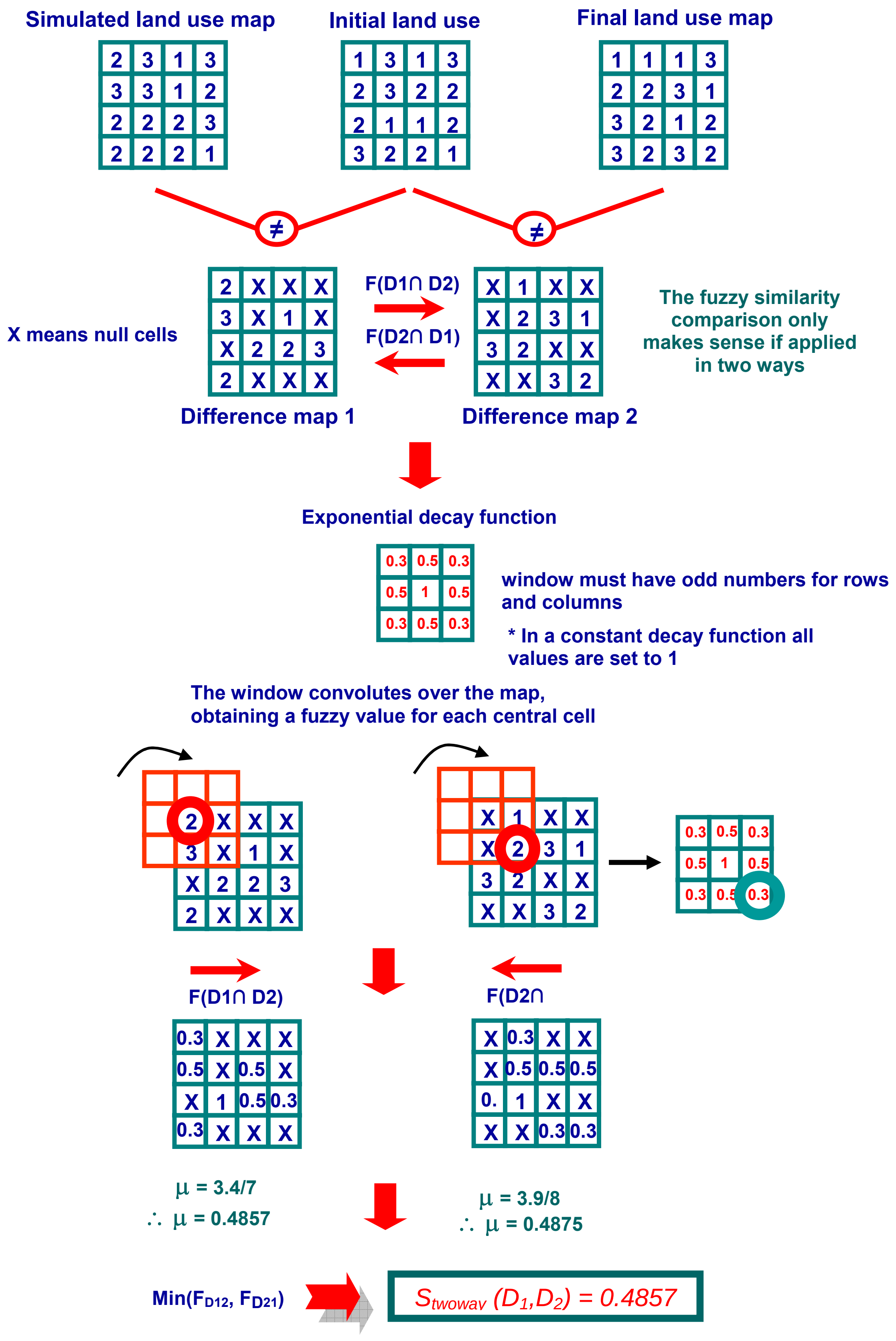
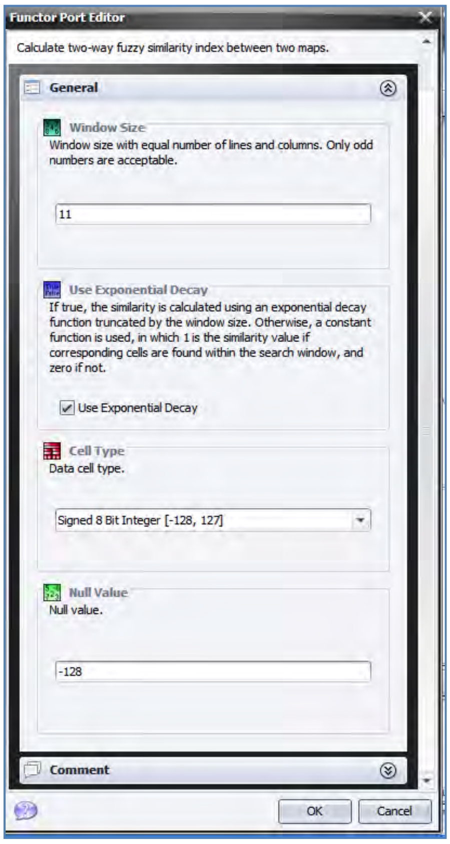

Simulated land use map
Initial land use
Final land use map
≠
≠
X means null cells
F(D1∩ D2)
F(D2∩ D1)
The fuzzy similarity comparison only makes sense if applied in two ways
Difference map 1
Difference map 2
Exponential decay function
window must have odd numbers for rows and columns
* In a constant decay function all values are set to 1
The window convolutes over the map, obtaining a fuzzy value for each central cell
F(D1∩ D2)
F(D2∩
μ = 3.4/7
∴ μ = 0.4857
μ = 3.9/8
∴ μ = 0.4875
Min(FD12, FD21)
Stwowav (D1,D2) = 0.4857

Fig. 12 - Fuzzy comparison method using map of differences and an exponential decay function. The same process applies to constant decay function, in which all window weights are set to 1.


Functor Port Editor

Calculate two-way fuzzy similarity index between two maps.

General

Window Size
Window size with equal number of lines and columns. Only odd numbers are acceptable.

11

Use Exponential Decay
If true, the similarity is calculated using an exponential decay function truncated by the window size. Otherwise, a constant function is used, in which 1 is the similarity value if corresponding cells are found within the search window, and zero if not.

[✓] Use Exponential Decay

Cell Type
Data cell type.

Signed 8 Bit Integer [-128, 127]

Null Value
Null value.

-128

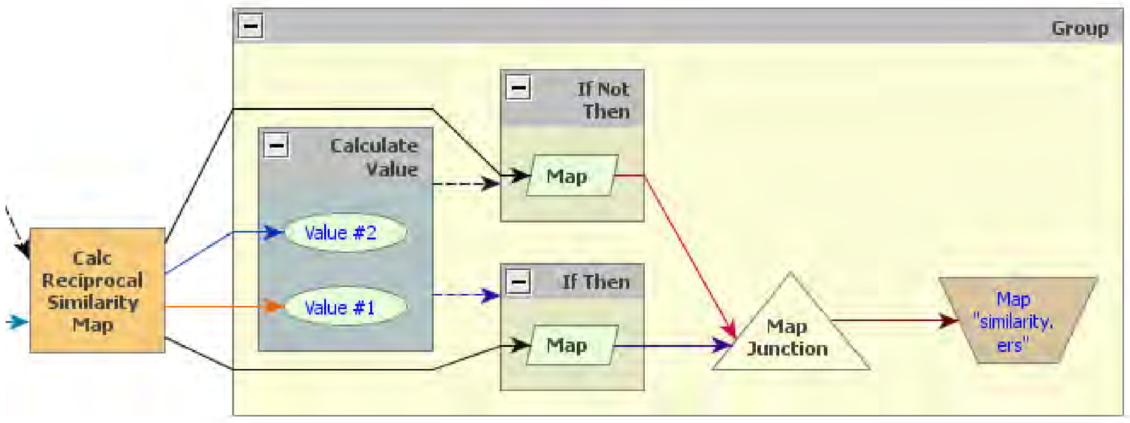
Comment

Open the *Calc Reciprocal Similarity Map* with the Port editor:

It receives as input two maps, the first and the second and outputs two similarity maps plus two values of similarity, the **First Mean** and the **Second Mean**. Now open it with Edit Functor.

The two parameters to set are the **Window Size** and the **Use Exponential Decay**.

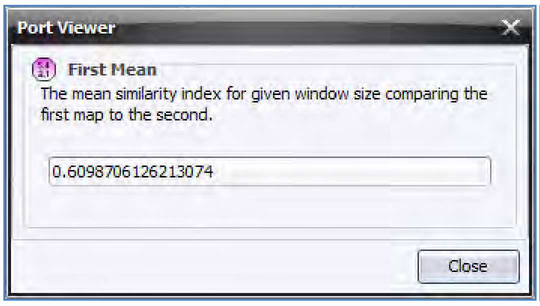
Open the *Group* “Save minimum similarity map”.



This set of functors allows the choice of the minimum similarity map to be saved. In order to do this, we have introduced three new functors. *If Then, If Not Then* and *Join Map* – available in the Control tab. The first two are containers that receive a Boolean flag as input (0 negates the condition and any number other than 0 asserts it). Before them *Calculate Value* examines the two values output from *Calc Reciprocal Similarity Map*, the **First Mean** and the **Second Mean**, passing 0 or 1 depending on which one is greater. *If Then* envelops a *Calculate Map* that receives the **First Similarity** Map and *If Not Then* one that receives the **Second Similarity** Map. Depending on the Boolean result of the *Calculate Value*, one of the two containers will pass on to *Map Junction* its result, thus always permitting the map with the minimum overall similarity value to be saved.

TIP: These three new functors allow the design of model that contains bifurcation to two or more execution pipelines. See fig. 1 in the beginning of this guidebook. This a fantastic feature for the design of complex models.

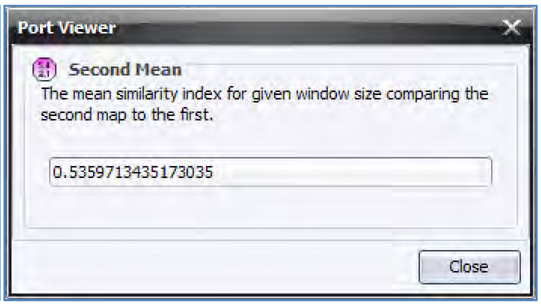
Examine now the **First Mean** and **Second Mean** ports clicking on them with the right button.


Port Viewer

First Mean

The mean similarity index for given window size comparing the first map to the second.

0.6098706126213074

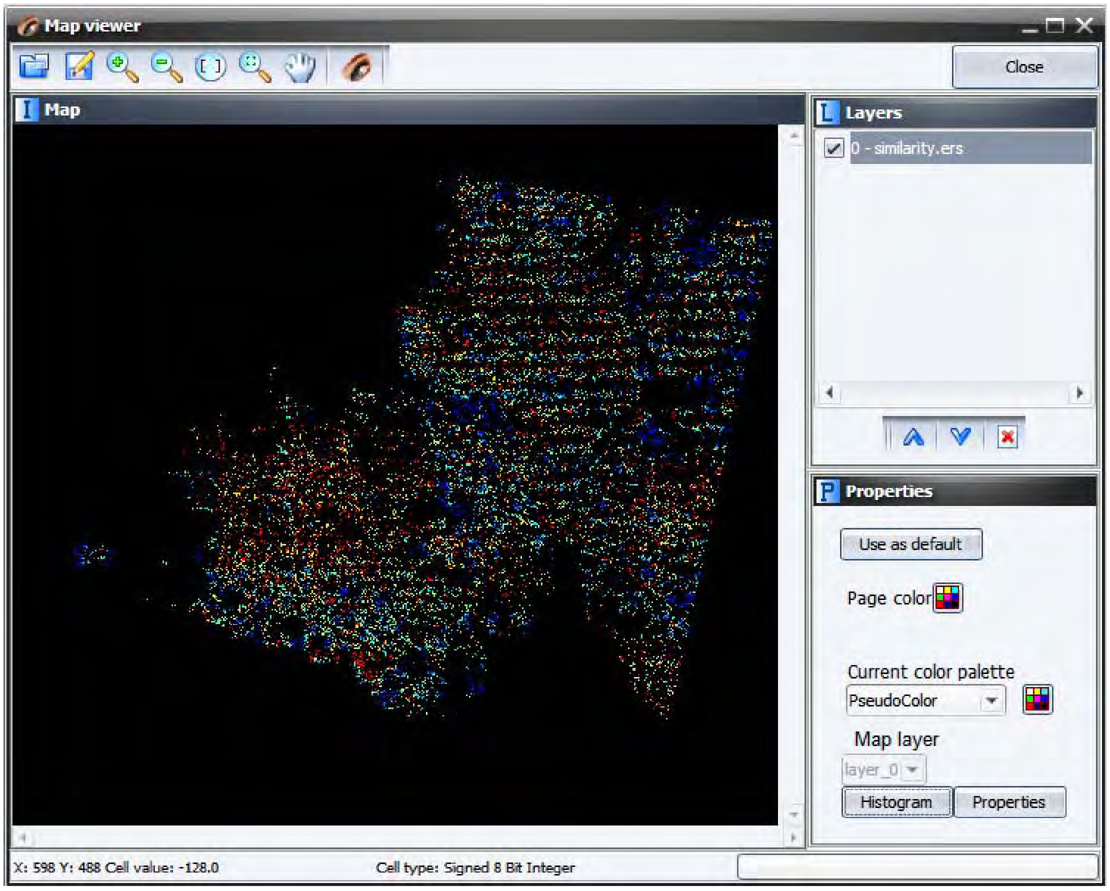

Port Viewer

Second Mean

The mean similarity index for given window size comparing the second map to the first.

0.5359713435173035

The minimum similarity map corresponds to the similarity obtained by comparing the simulated changes against the actual ones. Visualize now the resulting similarity map by opening it on the Map Viewer. Use **PseudoColor**, **Limits to Actual**, and **Histogram Equalize.**



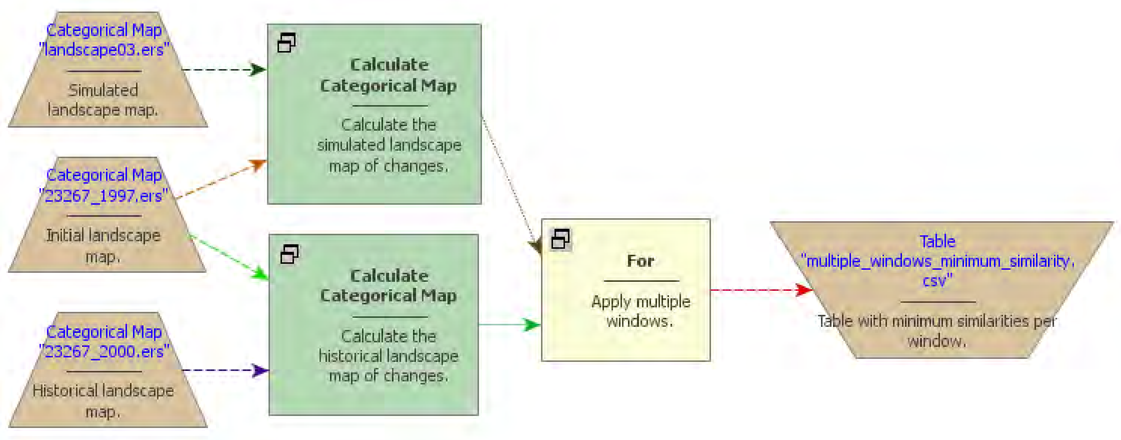
X: 598 Y: 488 Cell value: -128.0 Cell type: Signed 8 Bit Integer

The red and yellow areas show high to moderate spatial fit, whereas blue indicates poor fit.

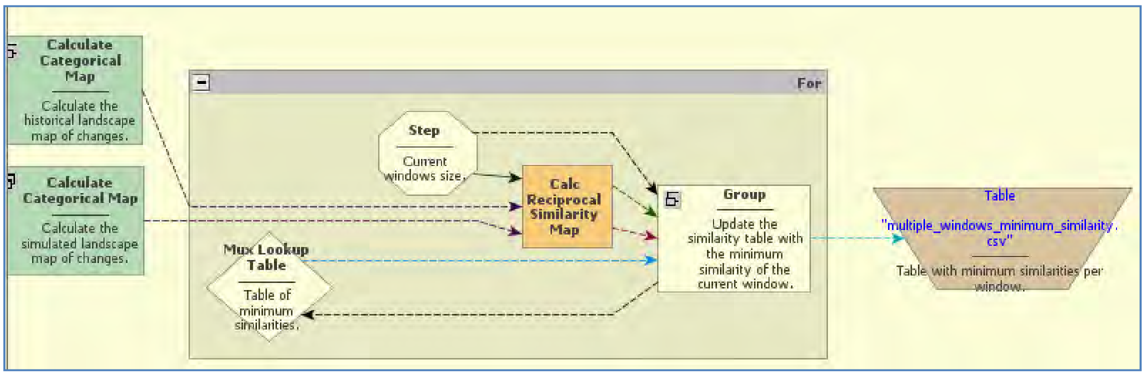
Another way to measure the spatial fitness between two maps is by means of multiple window similarity analysis. This method employs a constant decay function within a variable window size. If the same number of cells of change is found within the window, the fit will be 1 no matter their locations. This represents a convenient way to assess model fitness through decreasing spatial resolution. Models that do not match well at high resolution may have an appropriate fit at a lower resolution. Let's develop a multiple resolution fitness comparison in the next step.

7.7 Seventh step: Validating simulation using multiple windows and constant decay function

Open the model "determine-muti-window-similarity-of-differences.xml" in folder \7_validate_using_multiple_windows_constant_decay_function.

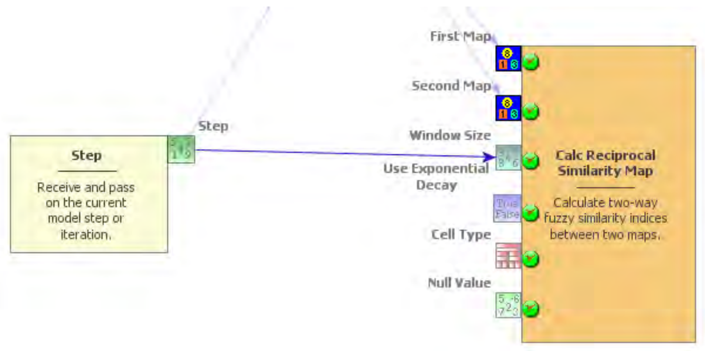

Categorical Map "landscape03.ers"
Simulated landscape map.
Categorical Map "23267_1997.ers"
Initial landscape map.
Categorical Map "23267_2000.ers"
Historical landscape map.
Calculate Categorical Map
Calculate the simulated landscape map of changes.
Calculate Categorical Map
Calculate the historical landscape map of changes.
For
Apply multiple windows.
Table "multiple_windows_minimum_similarity.csv"
Table with minimum similarities per window.

The first part of this model is similar to the previous one. Two *Calculate Map* functors are employed to derive the maps of changes. Now open the *For* functor. Let's examine its contents in detail.

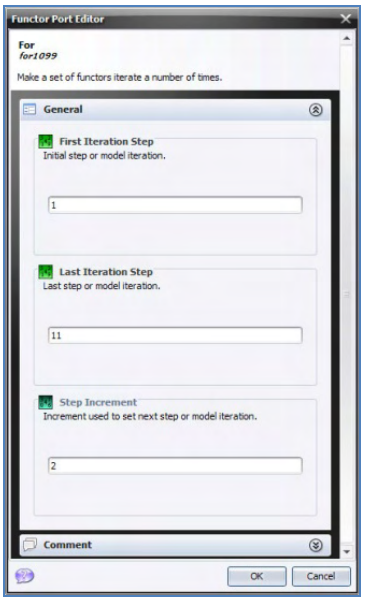

Calculate Categorical Map
Calculate the historical landscape map of changes.
Calculate Categorical Map
Calculate the simulated landscape map of changes.
For
Step
Current windows size.
Calc Reciprocal Similarity Map
Group
Update the similarity table with the minimum similarity of the current window.
Mux Lookup Table
Table of minimum similarities.
Table "multiple_windows_minimum_similarity.csv"
Table with minimum similarities per window.

At the center is located a *Calc Reciprocal Similarity Map*. Open it with the Edit Functor. Note that the option **Use Exponential Decay** is off, what means that a constant decay function is being used.

Click on the link between this functor and *Step* with the Edit Functors Port.


First Map
Second Map
Window Size
Use Exponential Decay
Cell Type
Null Value
Step
Step
Receive and pass on the current model step or iteration.
Calc Reciprocal Similarity Map
Calculate two-way fuzzy similarity indices between two maps.

The *Step* coming from the enveloping *For* is controlling the **Window Size**. *For* is a particular case of *Repeat*, in which initial and final steps as well as the step increment can be defined as follows:



Functor Port Editor

For
for1099

Make a set of functors iterate a number of times.

General

First Iteration Step
Initial step or model iteration.

1

Last Iteration Step
Last step or model iteration.

11

Step Increment
Increment used to set next step or model iteration.

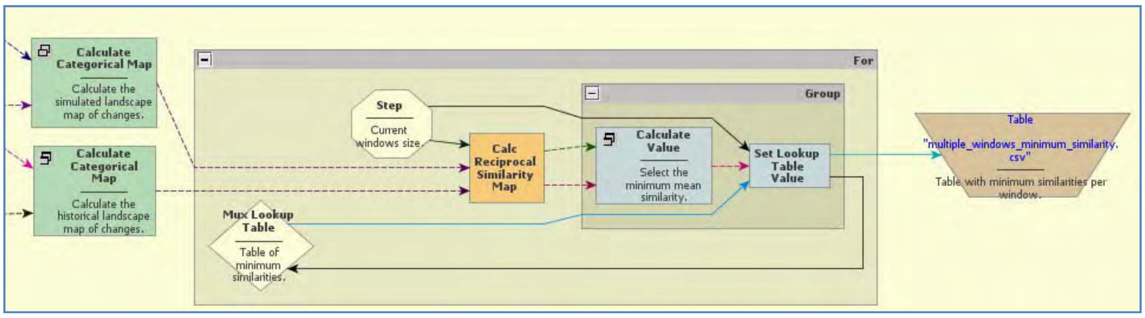
2

Comment

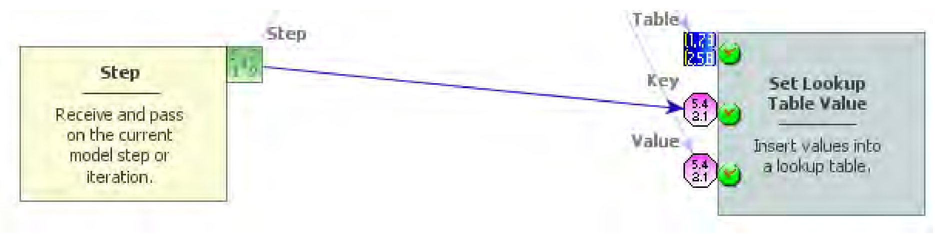
OK Cancel

In this *For* container, step goes from 1 to 11 by an increment of two. This is necessary because **Window Size** must be odd numbers. As a result **Window Size** will vary from 1x1, to 3x3, 5x5, 7x7, 9x9, and 11x11.

In the same way as in lesson 2, *Mux Lookup table* is employed to update the table containing the minimum similarity means of window sizes. Open *Group* containing the functors that update these tables.

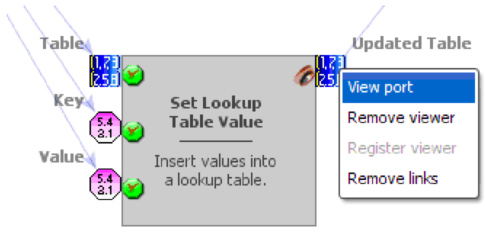

Calculate Categorical Map
Calculate the simulated landscape map of changes.
Calculate Categorical Map
Calculate the historical landscape map of changes.
For
Step
Current windows size.
Calc Reciprocal Similarity Map
Group
Calculate Value
Select the minimum mean similarity.
Set Lookup Table Value
Mux Lookup Table
Table of minimum similarities.
Table
"multiple_windows_minimum_similarity.csv"
Table with minimum similarities per window.

Calculate Value selects the minimum fitness value (formula applied is min(v1, v2)) and passes it to *Set Lookup Table*, which also receives as input the current step as the table key.

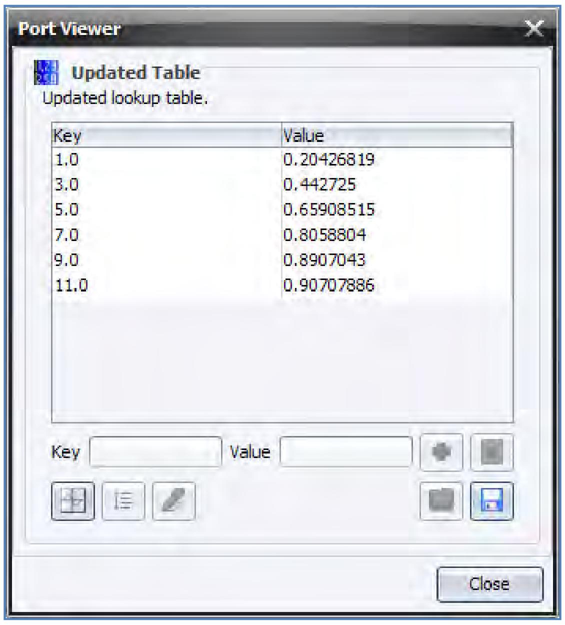

Step
Step
Receive and pass on the current model step or iteration.
Table
Key
Value
Set Lookup Table Value
Insert values into a lookup table.

As the *For* iterates, the *Calc Reciprocal Map* calculates similarity values for a window size and passes them to *Calculate Value*, which selects the minimum and passes it on to *Set Lookup Table*, which updates the value for the table key corresponding to the model step and feedbacks the updated table to *Mux Lookup Table*. When *For* is finished, the table is passed to *Save Table*.

Register viewer on the **Updated Table** port of *Set Lookup Table Value*, run the model and analyze the resulting table by clicking on **Updated Table** port with the right button.



This is what you get.



Updated Table
Updated lookup table.

Key	Value
1.0	0.20426819
3.0	0.442725
5.0	0.65908515
7.0	0.8058804
9.0	0.8907043
11.0	0.90707886

The fitness goes from 21% at 1 by 1 cell to 90% at 11 by 11 cell resolution. Note that because the simulation receives as input a fixed transition matrix setting the quantity of changes, we only need to assess the model fitness with respect to the location of changes. Taking into account that cell resolution is 250 meters and the window search radius is half of the resolution, you can draw a graph depicting model fitness per spatial resolution (fig. 13). **TIP**: Open table with the chart editor to produce the following graph.

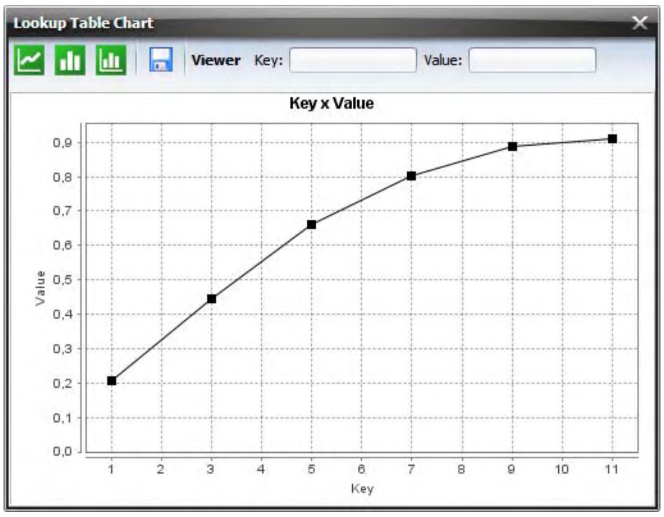


Fig. 13 – Model fitness

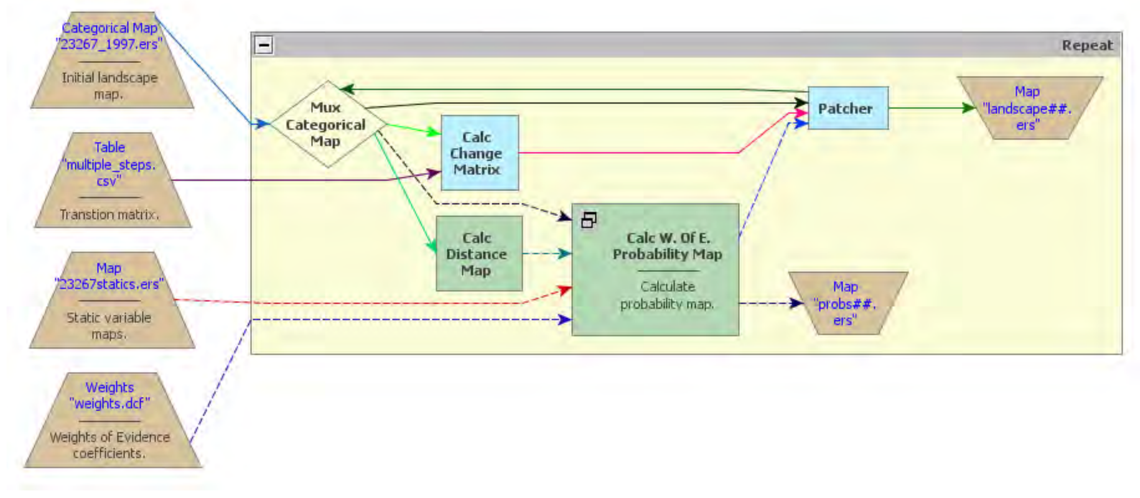
From the graph in fig. 13, we can assert the simulation reached a similarity fitness value over 50% at a spatial resolution of ≈800 meters. **TIP:** This method applies to multiple transitions too.

Now that the validation phase is completed, you can start playing with the simulation model parameters to explore possible outcomes. Let's do it in the next step.

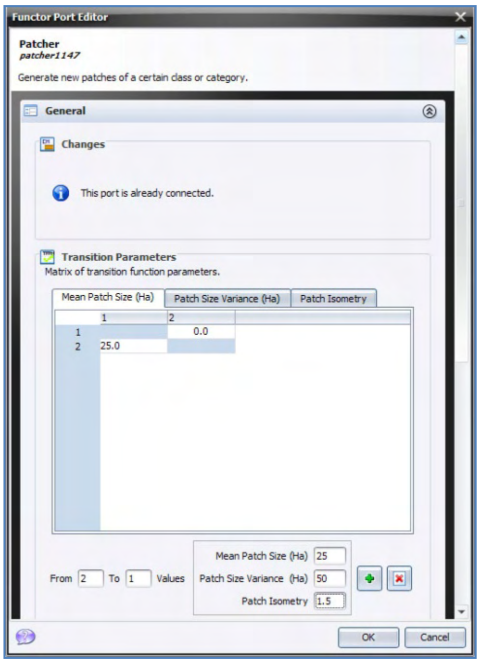
7.8 Eighth step: Running simulation with patch formation

Before going through this step, open and read the file "simulating_the_spatial_patterns _of_change.pdf" in Examples\patterns_of_change. This paper presents and discusses the results of a series of simulations using simplified synthetic maps and varying the parameters of the transition functions. The simulation results are evaluated using selected landscape structure metrics, such as fractal dimension, patch cohesion index, and nearest neighbor distance. These simulation examples are used to show how these functions can be calibrated, as well as its potential to replicate the evolving spatial patterns of a variety of dynamic phenomena (Soares-Filho *et al.*, 2003). You can also examine these models available in Examples\patterns_of_change.

This step aims at analyzing the parameters of the *Patcher* transition function on the structure of the simulated landscape. Open the model "simulate_deforestation_from_1997_2000_ with_patch_formation"from \Examples\setup_run_and_validate_a_lucc_model\8_run_lucc_ with_patch_formation. This is the same model as the one in the fifth step.



Now open *Patcher* with the Edit Functor tool.


Functor Port Editor

Patcher
patcher1147

Generate new patches of a certain class or category.

General

Changes

This port is already connected.

Transition Parameters
Matrix of transition function parameters.

Mean Patch Size (Ha) | Patch Size Variance (Ha) | Patch Isometry

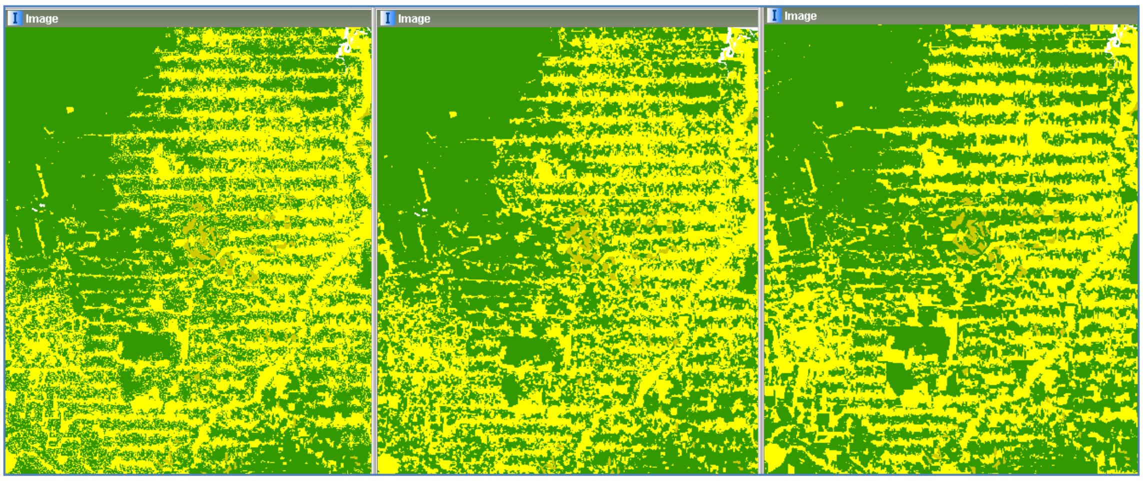
	1	2
1		0.0
2	25.0	

From 2 To 1 Values

Mean Patch Size (Ha) 25
Patch Size Variance (Ha) 50
Patch Isometry 1.5

Mean Patch Size is set to 25 ha, **Patch Size Variance** to 50 ha, and **Patch isometry** to 1.5. Since cell size is equal to 6.25 hectares (250 x 250 meters), the patches to be formed will have in average 4 cells and a variance of 8 cells.

Run the model; open both landscape3.ers files from this model and from the fifth step as well as 23267_2000.ers for visual comparison.



5th Step This Step 23367_2000

Note that the simulated landscape from this step shows a landscape structure closer to that of the final historical landscape. Let's now add the *Expander* Functor to the simulation model.

7.9 Ninth step: Running simulation with patch formation and expansion

The *Expander* functor is dedicated only to the expansion or contraction of previous patches of a certain class. Thus in the *Expander*, a new *Pij* spatial transition probability depends on the amount of cells type j around a cell type I, as depicted in fig. 14.

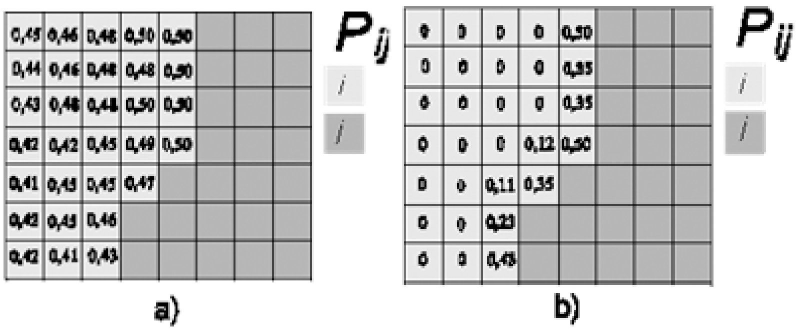
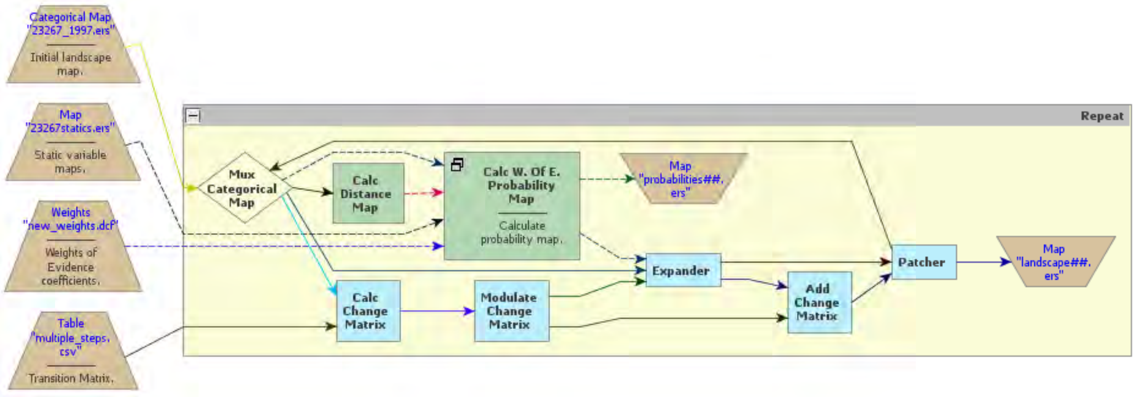
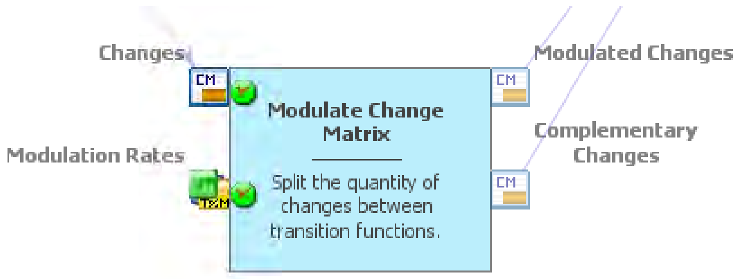


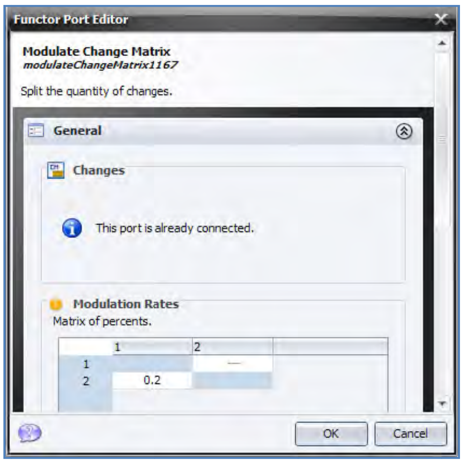
Fig. 14 – transition probabilities transformed to simulate expansion process.

Now, open the model "simulate_deforestation_from_1997_2000_with_patch_formation_and_expansion.xml" in the folder \Examples\setup_run_and_validate_a_lucc_model\9_run_lucc_with_patch_formation_and_expansion.



This model differs from the previous by three new functors: *Modulate Change Matrix*, *Expander*, and *Add Change Matrix*. *Modulate Change Matrix* splits the number of cells to be changed per transition into two matrices: **Modulated Changes** and **Complementary Changes**. The first goes to *Expander* and the second to *Add Change Matrix*. In this case, 20% of the changes 2 to 1 go to *Expander*.




Functor Port Editor

Modulate Change Matrix
modulateChangeMatrix1167
Split the quantity of changes.

General

Changes

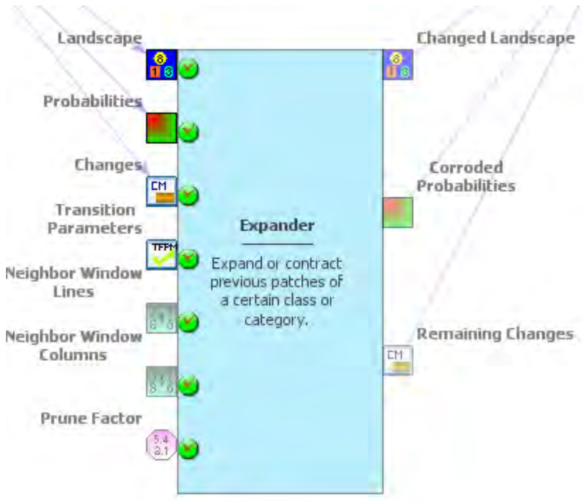
This port is already connected.

Modulation Rates
Matrix of percents.

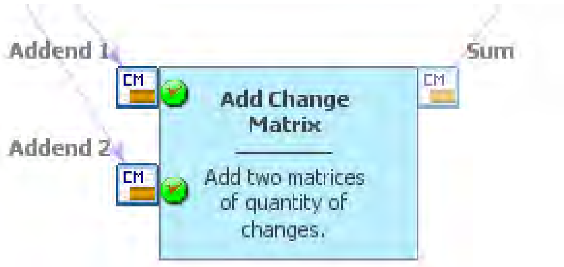
	1	2
1		—
2	0.2	

OK Cancel

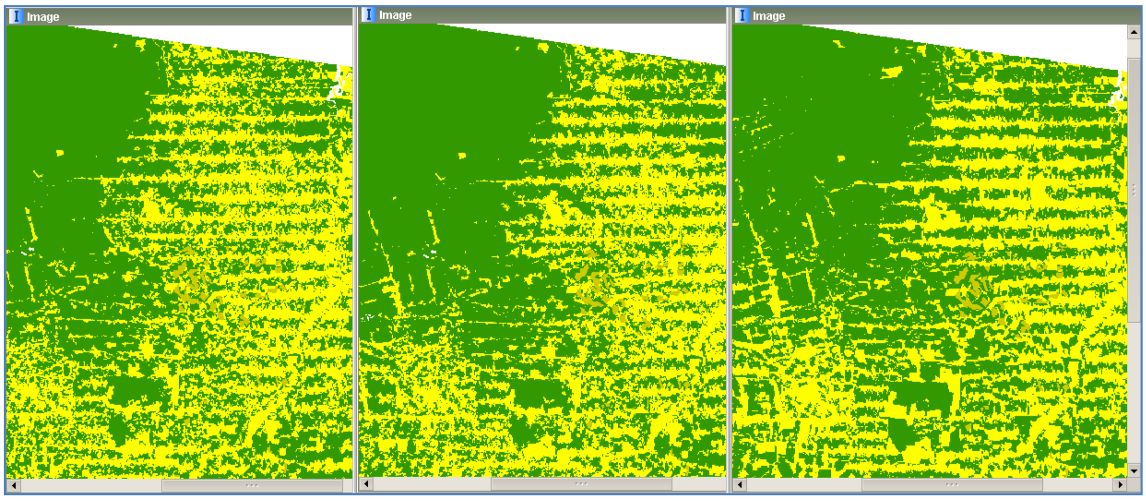
Also, the probability map coming from the *Calc W.OF E. Probability Map* goes first to *Expander*, usually placed before *Patcher*, because one cannot ensure that the *Expander* will execute the total quantity of changes passed to it. Still you need to connect its output ports **Changed Landscape** and **Corroded Probabilities** to *Patcher,* and **Remaining Changes** to *Add Change Matrix.*



Thus, *Patcher* will receive the landscape map after *Expander* has modified it and the probability map corroded (set to 0) where transitions took place. In case *Expander* does not succeed in making all specified changes, a matrix of remaining quantity of changes for each transition will be passed to *Add Change Matrix*, which will combine the matrices coming from *Modulate Change Matrix* and the port **Remaining Changes** of *Expander*. Its other parameters are set like those of *Patcher*.



Now run the model, access the log for the report, and compare its output with the previous ones.



8th Step 9th Step 23367_2000

Did this model approximate more to the structure of the final historical landscape?

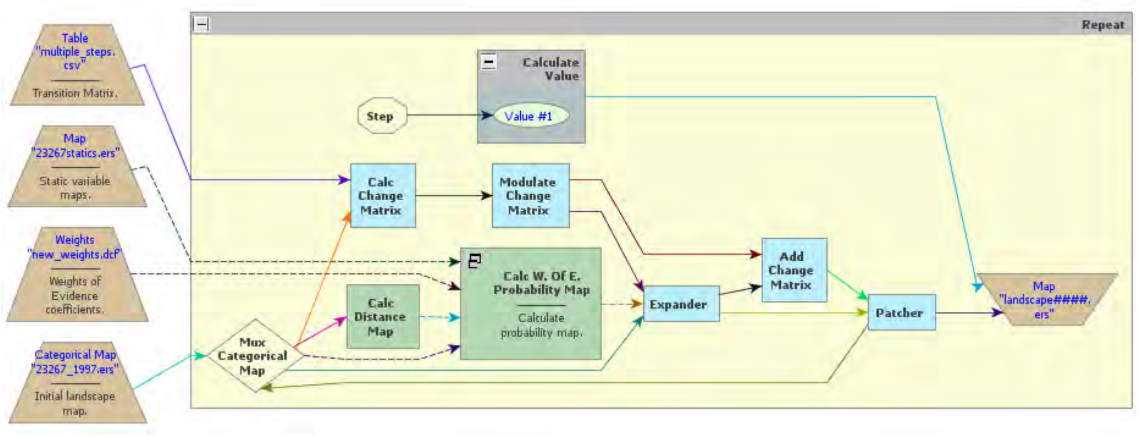
Try to change the parameters of *Modulate Change Matrix*, *Expander* and *Patcher* to see what you get. **TIP:** change only one parameter at a time. Now that model is calibrated both in regard to the location of changes and the landscape structure as well as validated, you can apply it for projection purpose. Let's move on to the last step.

7.10 Tenth step: Projecting deforestation trajectories

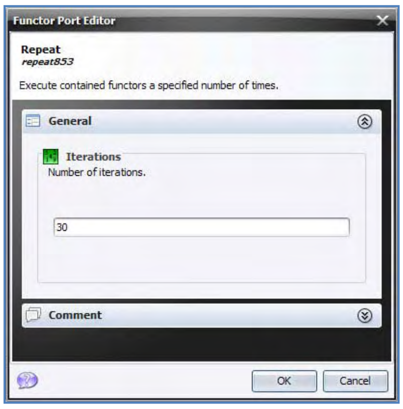
Simulation models can be envisaged as a heuristic device useful for assessing, from short to long terms, the outcomes from a variety of scenarios, translated as different socioeconomic, political, and environmental frameworks. A special class among them, spatially explicit models simulate the dynamics of an environmental system, reproducing the way its spatial patterns evolve, to project the probable ecological and socioeconomic consequences from the system dynamics.

Thus, as the current example, we can apply the simulation model to assess the impacts of future trajectories of deforestation under various socioeconomic and public policies scenarios, on the emission of greenhouse gases (Soares-Filho *et al.*, 2006), regional climate change (Schneider *et al.*, 2006; Sampaio *et al*, 2007), fluvial regime (Costa *et al*, 2003, Coe *et al*, 2009), habitat loss and fragmentation (Soares-Filho *et al.*, 2006, Texeira *et al.*, 2009) as well as the loss of the forest environmental services and economic goods (Fearnside, 1997).

Open the model simulate_deforestation_from_1997_2000_30years_ahead.xml available in \Examples\setup_run_and_validate_a_lucc_model\10_run_deforestation_trajectories and keep opening the *Group* functors.



The only differences between this model and the prior one consist of the initial landscape, which is now the 2000 map, the number of iterations set to "30", and three additional functors *Calculate Value*, *Number Value*, and *Step* employed to provide the suffix for the name of the simulated landscape in years starting in 2001. Also, to save space, the probability map won't be saved.


Functor Port Editor

Repeat
repeat853

Execute contained functors a specified number of times.

General

Iterations
Number of iterations.

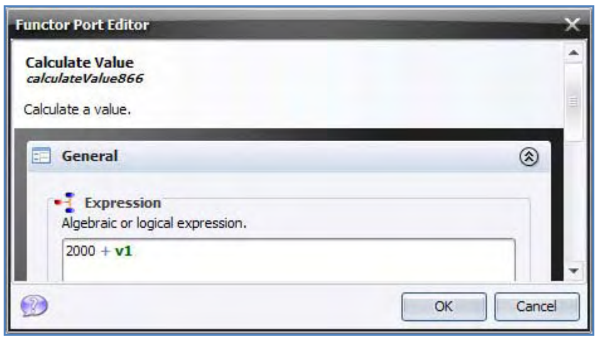
30

Comment

TIP: you need to move *Map "*Landscape####.ers*"* out of *Repeat* to break the **Step** link.

As this model uses fixed transition rates, we can consider that it projects the historical trend into the future, thus named as the historical trend scenario.

Check the differences between this model and the prior one by opening the functors and their connections and then run the model and open the landscape2030.ers. **TIP**: You might want to save only the final landscape map. To do this, you just need to drag *Save Map* out of *Repeat*.

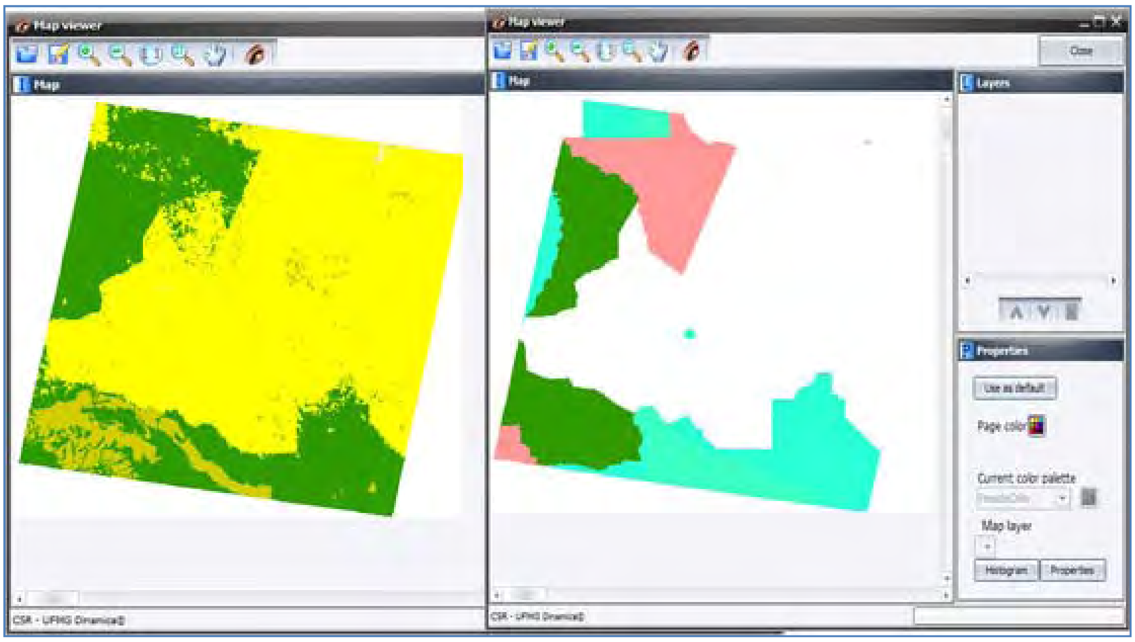

Functor Port Editor

Calculate Value
calculateValue866
Calculate a value.

General

Expression
Algebraic or logical expression.
2000 + **v1**

This is what you get:



Note that the forest virtually disappears outside of protected areas, which begin to be encroached. This is what will happen if the recent historical deforestation trend persists into the future.

Animated maps are a very powerful tool to make the general public aware of the possible outcomes of system dynamics, such as the Amazon deforestation. **TIP:** Save the output maps per time step as Geotiff. Thereafter you can import then into a GIS or image processing software to build animations.

You may want to improve this model by incorporating dynamic rates, other dynamic variables, submodels, such as the road constructor, and feedbacks from the landscape attributes to the transition rate calculation. All of these are possible in Dinamica EGO. An example of road constructor module is found in the model mato_grosso_road.xml in \Examples\run_lucc_northern_mato_grosso\run_roads_with_comments. Also, Dinamica EGO allows the incorporation of economic, social and political scenarios into a model that integrates the effect of these underlying causes (Geist, & Lambin, 2001) on the trajectory of deforestation (usually these variables are input to the model using tables with keys assigned to a geographic unit, such as county, state or country). Finally, Dinamica EGO also enables the coupling of external models developed in VENSIM, a system thinking software (www.vensim.com). For example, VENSIM could be used to model the effect of a complex scenario on the transition rates. Part of these advanced feature will be demonstrated in the advanced lessons and in an example of the application of Dinamica EGO in modeling scenarios for a REDD (Reduced Emissions from Deforestation and Forest Degradation) case study. But before moving to these lessons, let' learn some features that are key for developing complex models in Dinamica EGO.

8. Dinamica EGO script language and console launcher

What will you learn?

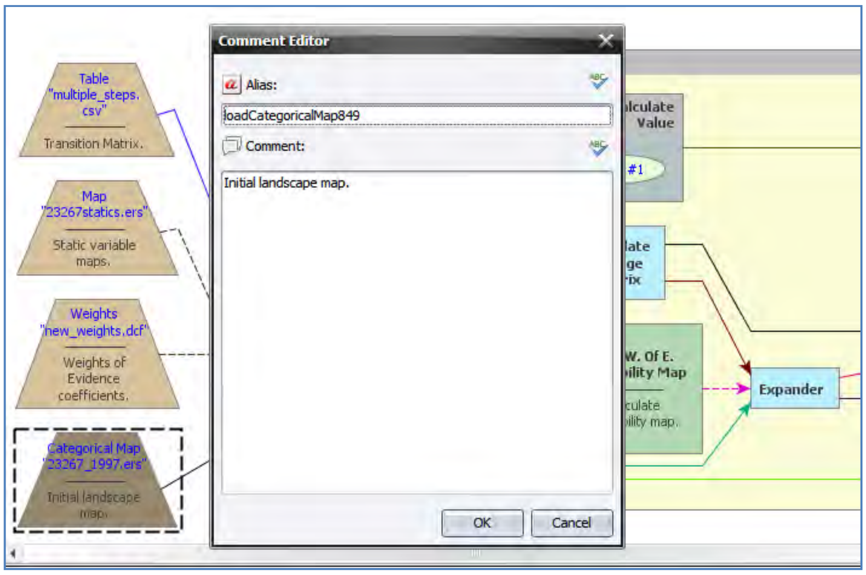
- Naming variables with alias
- Dinamica EGO programming language
- Dinamica EGO Console launcher

As a model becomes more and more complex, you may find useful to save it in EGO format to keep developing the model using EGO programming language on a text editor (e.g. NotePad++ or Context). For example, the Amazon logging model (Merry *et al.*, 2009) developed in Dinamica EGO involves more than one thousand script lines. Both script formats (XML, EGO) are 100% compatible, so users can take back and forth a model from the graphical interface to the text editor without losing any information. Advanced modelers will greatly benefit from this well structured and tractable programming language. **TIP:** EGO format is also a good way to merge two models, since you can cut and paste parts of scripts.

Open again the model "simulate_deforestation_from_1997_2000_30years_ahead.xml". To make the model more intelligible on the text editor, you first need to name some key variables and then add comments to parts of the model. Let's do this. Select the Add Comment to Functor tool and click on the *Categorical Map "*23267_1997.ers".

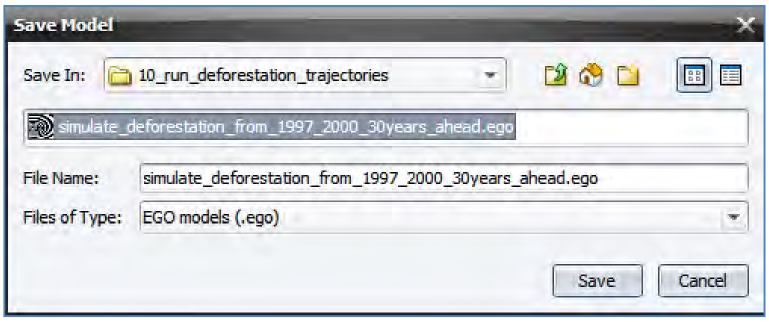
Write in the alias field "initial_landscape", click on *Calc Distance Map* and write the alias "distance_to_deforested".

Do the same to the functors: *Map* "23267statics.ers", *Calc W. Of. E. Probability Map, Calculate Value, Expander, Patcher, and Mux Categorical Map*, writing the respective alias: "statics", "probability", "year", "landscape_with_patches_expanded", "landscape_with_new_patches", and "feed_back_landscape". Now save the model in EGO format.



Alias: loadCategoricalMap849

Comment: Initial landscape map.


Save Model
Save In:
10_run_deforestation_trajectories
simulate_deforestation_from_1997_2000_30years_ahead.ego
File Name:
simulate_deforestation_from_1997_2000_30years_ahead.ego
Files of Type:
EGO models (.ego)
Save
Cancel

Open in a text editor (preferentially one that highlights the script in C++ format, e.g. Notepad++, available on http://notepad-plus.sourceforge.net) the model "simulate_deforestation_from_1997_2000_30years_ahead.ego". The EGO script is as follows:

```
/**
 dff.date = Fri Sep 04 12:57:13 2009
 dff.version = 1.3.4.20090827-beta
 metadata.author = Dinamica Team
 metadata.description = This script corresponds to a deforestation model. Only one transition is modeled: 2 to 1, time-period comprises 2000 to 2030, divided in annual time steps. The land cover classes are non-forest (3), forest (2), and deforested (1). The model is set to form patches and to expand previous patches of deforested land.
 metadata.organization = CSR / UFMG
 metadata.showproperties = yes
 metadata.title = Simulate Deforestation
*/
Script {{
  // Initial landscape map.
  initial_landscape := LoadCategoricalMap {
    filename = "../originals/23267_1997.ers",
    loadAsSparse = .no,
    defineNullValue = .no,
    nullValue = 0,
    suffixDigits = 0,
    step = .none
  };

  // Static variable maps.
  statics := LoadMap {
    filename = "../originals/23267statics.ers",
    loadAsSparse = .no,
    defineNullValue = .no,
    nullValue = 0,
    suffixDigits = 0,
    step = .none
  };
```

This is the header

Here the model starts

This is the load categorical map

A functor is elapsed by { }

Alias replaces EGO internal names

e.g. initial_landscape

Comments after //

```
// Weights of Evidence coefficients.

loadWeights851 := LoadWeights "../4_weights_of_evidence_correlation/new_weights.dcf" 0 .none;


// Transition Matrix.

loadLookupTable852 := LoadLookupTable "../1_transition_matrix_calculation/multiple_steps.csv" 0 .none;


// Simulation model.

@collapsed = no

Repeat 30 {{

   step = step;


   feed_back_landscape := MuxCategoricalMap initial_landscape landscape_with_new_patches;


   modulatedChanges complementaryChanges := ModulateChangeMatrix (CalcChangeMatrix feed_back_landscape
loadLookupTable852) [

      2->1 0.2

   ];


   distance_to_deforested := CalcDistanceMap {

      categoricalMap = feed_back_landscape,

      categories = [ 1 ],

      cellType = .int32,

      nullValue = -2147483648,

      truncateDistance = .yes

   };


   // Calculate probability map.

   @collapsed = yes

   probability := CalcWOfEProbabilityMap {

      landscape = feed_back_landscape,

      weights = loadWeights851,

      transitions = [ 2->1 ],

      cellType = .uint8,

      nullValue = 0

   } {{

      NameMap statics "static_var";


      NameMap distance_to_deforested "distance";

   }};


   @alias = landscape_with_patches_expanded

   changedLandscape corrodedProbabilities remainingChanges := Expander {
```

Container Repeat begins

```
        landscape = feed_back_landscape,
        probabilities = probability,
        changes = modulatedChanges,
        transitionParameters = [
            2->1 20 50 1.5
        ],
        neighborWindowLines = 3,
        neighborWindowColumns = 3,
        pruneFactor = 10
    };

    step861 := Step step;

    landscape_with_new_patches _ _ := Patcher {
        landscape = changedLandscape,
        probabilities = corrodedProbabilities,
        changes = (AddChangeMatrix remainingChanges complementaryChanges),
        transitionParameters = [
            2->1 25 50 1.5
        ],
        neighborWindowLines = 3,
        neighborWindowColumns = 3,
        pruneFactor = 10
    };

    year := CalculateValue [
        2000 + v1
    ] .no 0 {{
        NumberValue step861 1;
    }};

    SaveMap {
        map = landscape_with_new_patches,
        filename = "landscape.ers",
        suffixDigits = 4,
        step = year,
        useCompression = .yes
    };
  }};
}};
```

Expander receives feed_back_lansdcape

Patcher receives changed_lansdcape

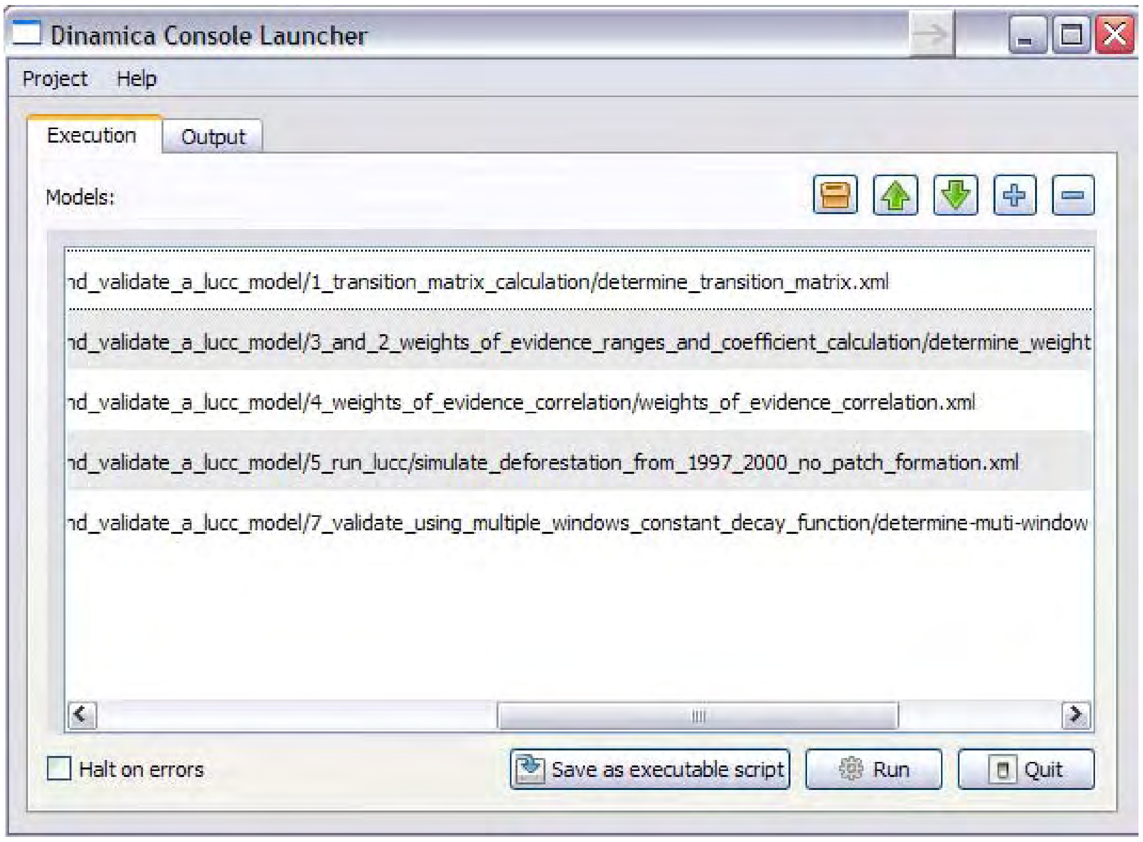
Functor SaveMap

Close Container Repeat

Close Model

Now that you have learned a bit about programming with EGO script language, another resource available in Dinamica EGO is the Console Launcher. Any model saved in Dinamica EGO, either XML or EGO, can be run from the command prompt. Running the model from the command prompt increases model performance, as the model becomes free of the burden of the graphical interface. In addition, from the command prompt, Dinamica EGO takes advantage of multiprocessor architecture, either splitting execution pipelines on different processors or running several functors' internal algorithms using parallel processing. Therefore, execution becomes much faster. Depending on the amount of data loaded in the memory, it is also possible to disable disk swapping. **TIP:** Windows 32 bits only handles up to 3 gigabytes of memory for a single process.

To facilitate running a model from the command prompt, we have developed a Console Launcher tool. Its use is strongly advised for complex models and others with large amount of data. Also, Console Launcher allows for queuing models and running them sequentially Call the Console Launcher from the tools subfolder in the Start/Programs/Dinamica EGO shortcut. It will open the following interface.



```
nd_validate_a_lucc_model/1_transition_matrix_calculation/determine_transition_matrix.xml
nd_validate_a_lucc_model/3_and_2_weights_of_evidence_ranges_and_coefficient_calculation/determine_weight
nd_validate_a_lucc_model/4_weights_of_evidence_correlation/weights_of_evidence_correlation.xml
nd_validate_a_lucc_model/5_run_lucc/simulate_deforestation_from_1997_2000_no_patch_formation.xml
nd_validate_a_lucc_model/7_validate_using_multiple_windows_constant_decay_function/determine-muti-window
```

Some parameters of Console Launcher are**: Model**: name of the model, **Scheduler**: how the model functors are going be sequenced, **Verifier**: Type of script verification, **Log Level:** Maximum level of log report, **Use Predefined seed:** Check it to use a predefined seed to generate random numbers, **Processors:** number of processors that model will use. Use 0 to automatically detect the number available. **Run Model:** Check it to run model completely. You can save the Console Launcher configuration into a batch file too. This is useful in case you want to couple Dinamica EGO with other software. **TIP:** another way to loosely couple other computer programs with Dinamica EGO is by means of functor *Run External Process*, available in the Control Tab. By using this functor, you will be able to make a call to an external program from within Dinamica, pass to this program intermediate results and feed its output back again into the Dinamica EGO model.

Congratulations, you have come so far. But still, there are numerous possibilities to explore in designing a model with Dinamica EGO. To those who pursue more advanced modeling skills we introduce the next chapter with examples using advanced resources.

9. Advanced resources

What will you learn?

- Varying parameters in a simulation
- Using subregions in a simulation model
- Using the concept of sojourn time
- Using deterministic transitions
- Using local saturation
- Functors:
    - *Select Transition Matrix*
    - *Calc Change Matrix*
    - *Modulate Change Matrix*
    - *Add Change Matrix*
    - *Select Weights*
    - *Extract Map Layer*
    - *Create Cube Map*

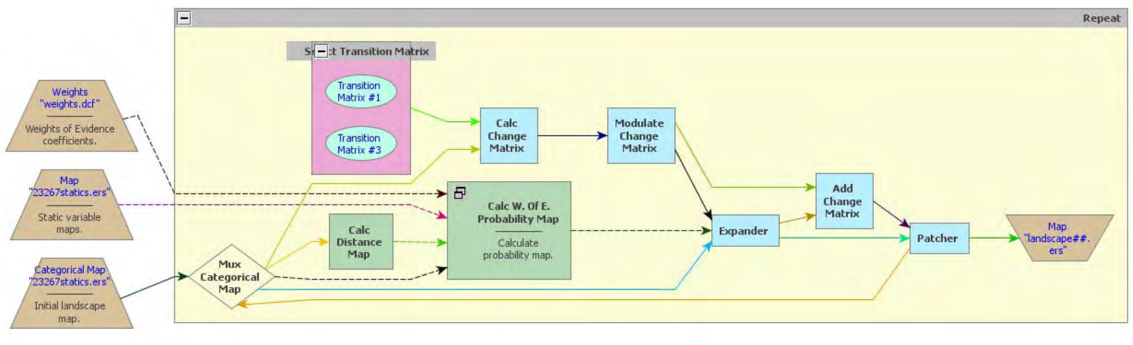
9.1 Varying parameters in a simulation

Nothing should remain constant in a dynamic model. In this respect, Dinamica EGO enables to vary model parameters either in different simulation phases or at each time step. In the Stack tab, a set of Select Functors can be used to change the values of input parameters according to the model step.

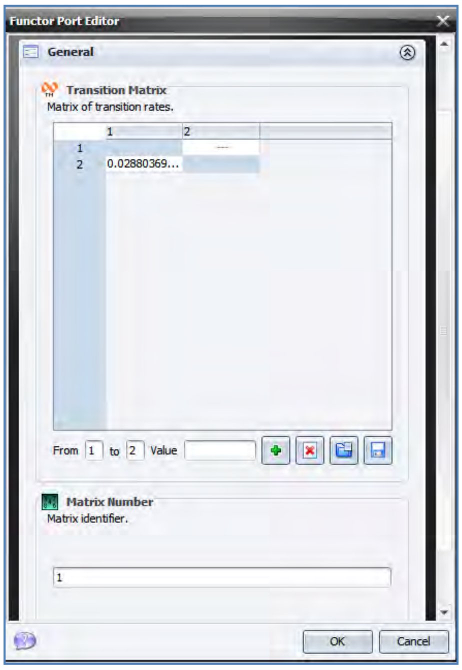
Using different transition matrices

For examples, instead of using only a constant transition matrix, *Select Transition Matrix* enables the use of different transition matrices in a simulation.

To start, load the model "simulate_deforestation_using_multiple_transition_matrix.xml" from \ advanced\multiple_parameters\simulate_deforestation_using_multiple_transition_matrix



Using more than one transition matrix in a model is quite simple. In this version of a deforestation model, *Select Transition Matrix* was added to allow the choice of a transition matrix according to the model step. Each transition matrix is entered in a *Number Transition Matrix,* click on this functor with the Edit Functor.


Functor Port Editor

General

Transition Matrix
Matrix of transition rates.

	1	2
1		----
2	0.02880369...	

From 1 to 2 Value

Matrix Number
Matrix identifier.

1

OK Cancel

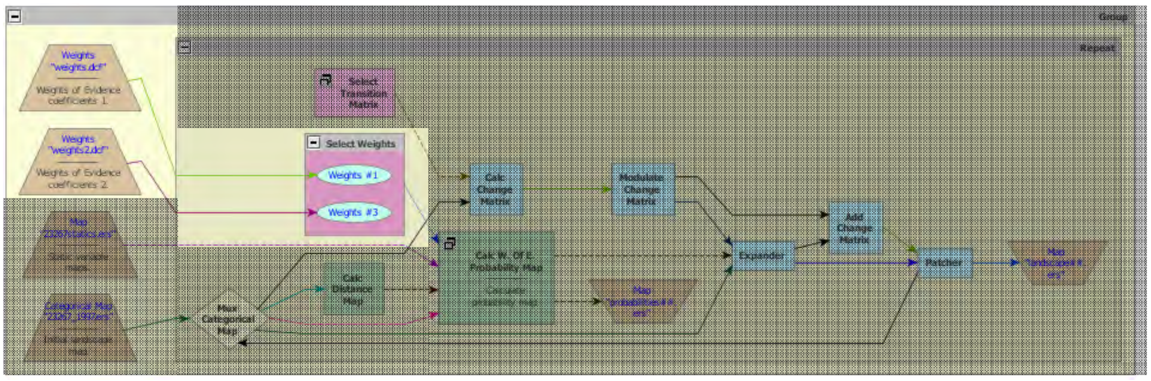
Matrix Number in this functor indentifies the model step in which this transition matrix begins to operate. The model will use this matrix until it finds a *Number Transition Matrix* with **Matrix Number** that corresponds to the current model step, while from this point onwards *Select Transition Matrix* switches to the new transition matrix. **TIP:** you can use as many transition matrices as time steps. You just need to add more *Number Transition Matrix* functors and number them according to the model time intervals.

Using different Weights of Evidence coefficients.

The same procedure applies to the selection of other model parameters, such as the Weight of Evidence coefficients.

Load the model: “simulate_deforestation_using_multiple_transition_matrix_and_weights.xml” from \Examples\advanced\multiple_parameters\simulate_deforestation_using_multiple_transition_matrix_and_weights

A *Select Weights* functor is added to the model to enable the selection of more than one Weights of Evidence coefficient files. A *Number Weight* is inserted within the container *Select Weights* for each Weights of Evidence coefficient file. Another *Load Weights* is added to the model in the outside part of the main loop (*Repeat*). This functor loads the coefficients from a file and is connected to one of the *Number Weights* inside the *Select Weights*. Open each one of the *Number Weights* and edit the **Weight Number** field to define the step when each coefficient file begins to take part in the model.



9.2 Using subregions in a simulation model

Load the model: “simulate_deforestation_using_sub_regions.xml” from Examples\advanced\sub_regions\simulate_deforestation_using_sub_regions

The Subregion set of functors are used to split a map into parts to process each subregion’s dataset separately, and then combine the results again (fig. 15). By making use of the subregion approach, you can define a sequence of operations that will be applied only to certain subregions or establish different parameters and coefficients for each subregion, modeling as a result the regional context that influences a particular phenomenon.

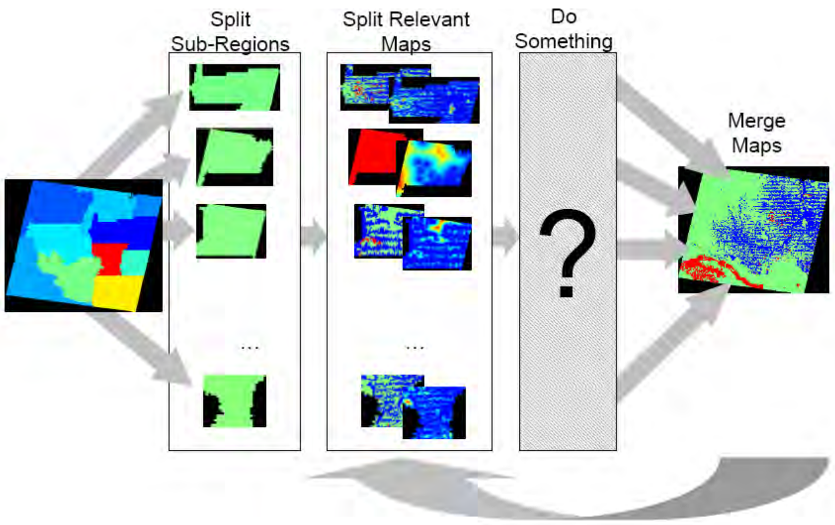
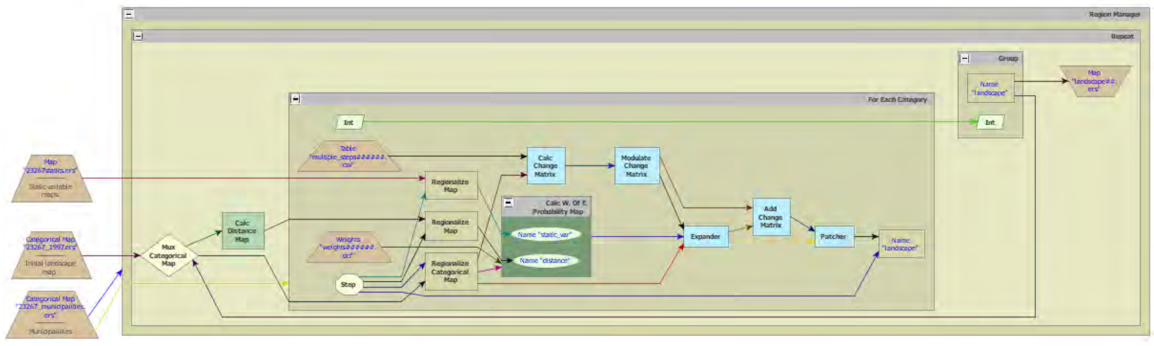


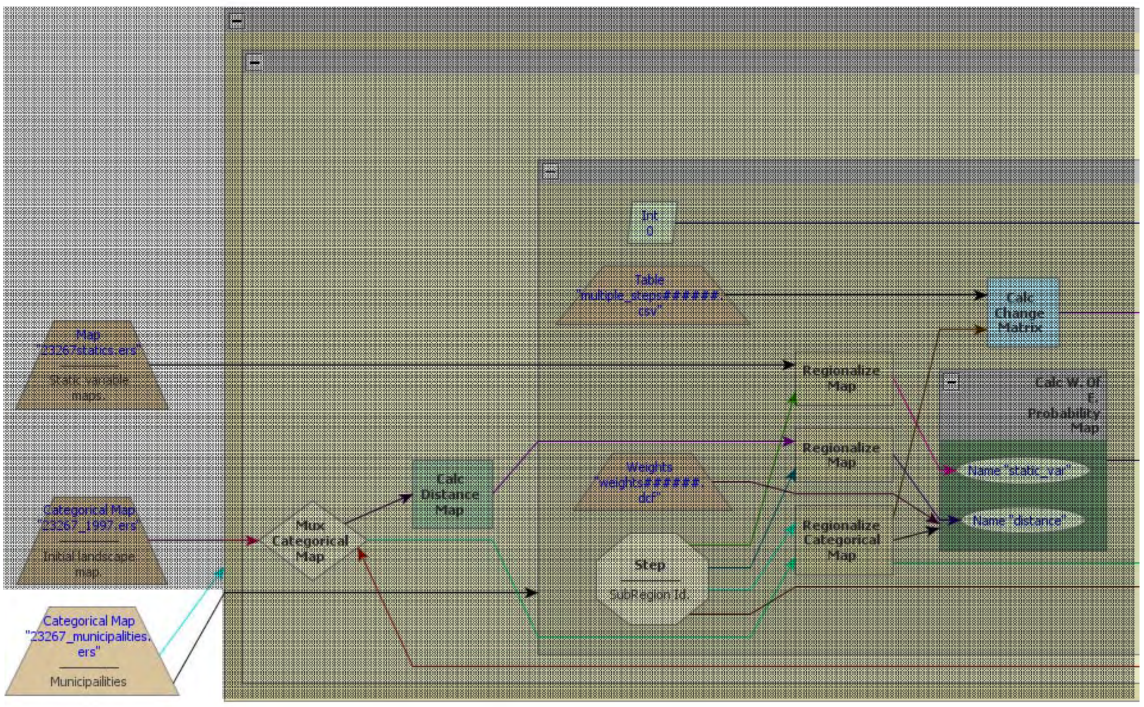
Fig. 15 -Subregion scheme enables the application of different submodels to parts of a map.

In this model, we introduce subregions in a simulation to enable the use of different transition matrices and Weight of Evidence coefficients for each subregion of a map.

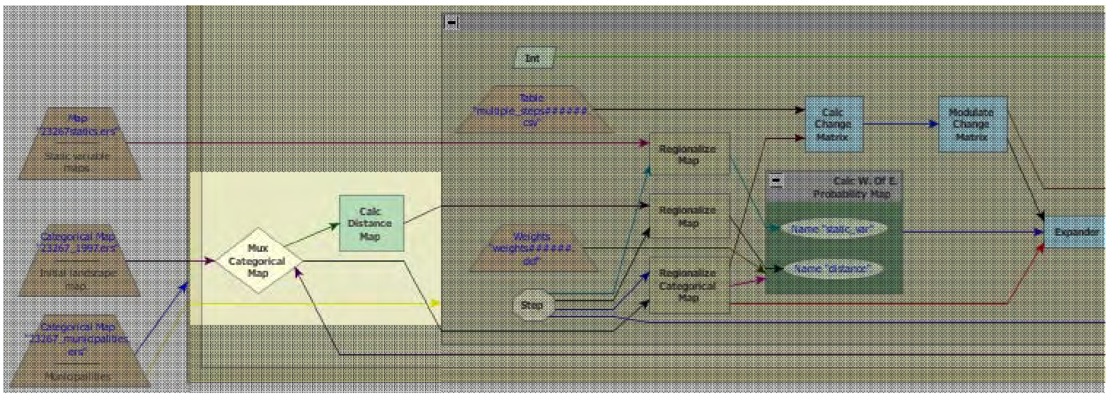


In order to this, you need to slightly modify the simulation model of lesson 7. First, a specific calibration for each subregion should be performed. In the example above, transition matrices and a set of weights of evidence coefficients were calculated for each one of the map subregions. See models for calibrating different subregions in “Examples\advanced\sub_regions\calibration”.

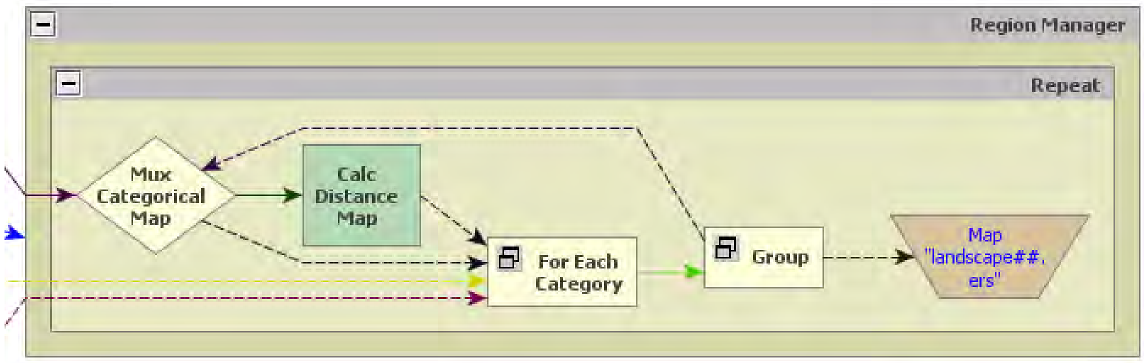
In the simulation model, one *Load Categorical Map* must be added. It will be responsible for loading the map with the subregions **TIP:** this must be a categorical map (in this example, a map of municipal limits). The area of each municipality is identified by its 6 digit code. This code will identify the subregions and control the regionalization process.



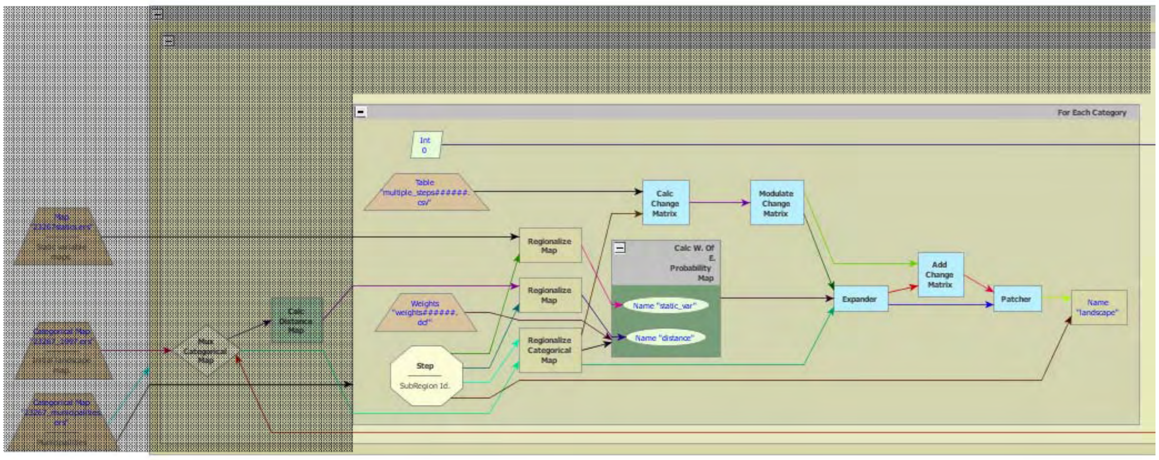
In this model, deforestation is simulated on an annual basis for each one of the municipalities. After that, the resulting maps are combined into a new land-use map that will be split again in the beginning of the next step. To ensure spatial continuity across subregions, some operations, such as the calculation of distance map to deforested cells using *Calc Distance Map,* are performed for the entire map.



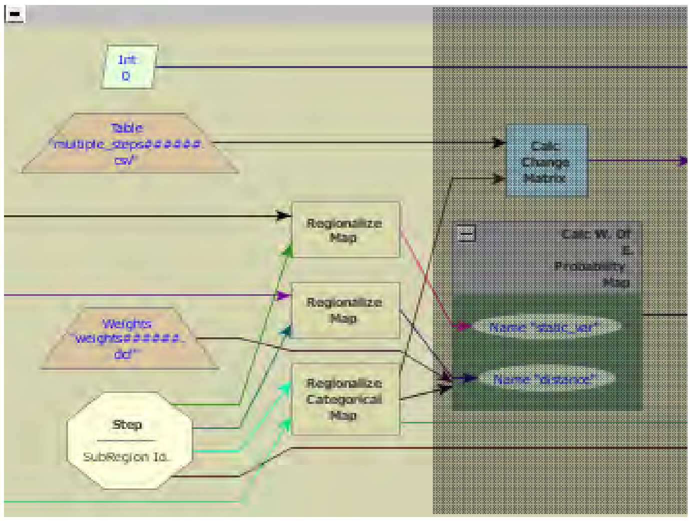
A *Region Manager* is added outside of the main loop (*Repeat)*. This container controls the creation and merging of subregions.



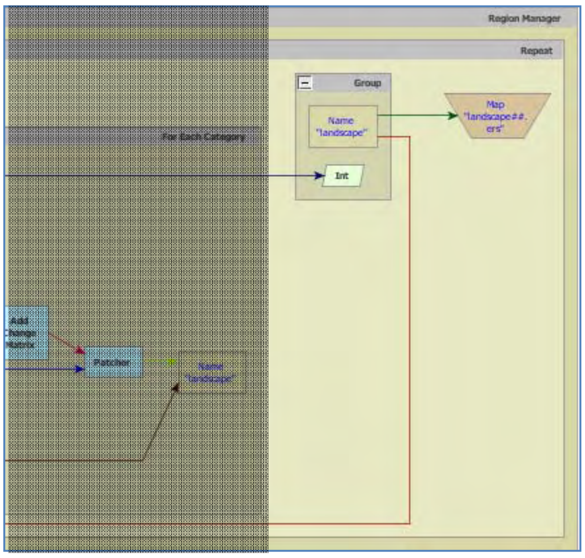
The container *For Each Category* also controls the subregion creation process. For each map category, *For Each Category* will repeat the sequence of functors within it. Make sure the Categorical Map “23267_municipalities.ers” is connected to *For Each Category* as well as to *Region Manager.* In this case, *Step* placed within the innermost container receives and passes on the subregions’ codes. Also any sequence of functors that affect each region should be put within this container.



The functors *Regionalize Map* and *Regionalize Categorical Map* spilt the map into regional maps. Make sure *Step* is connected to them. All maps that are combined after these *functors* need to have the same dimensions. So make sure that they all are regionalized through these functors, as the example of distance maps and static variables. Note also, that you can set different transition matrices or Weights of Evidence files for the subregions by assigning the subregion’s code (in this case 6 digits) as a suffix to these files and placing *Load Table* and *Load Weights* inside *For Each Category* Container. **TIP:** you need also to set suffix digits to 6.


Int 0
Table "multiple_steps######.csv"
Calc Change Matrix
Regionalize Map
Regionalize Map
Regionalize Categorical Map
Calc W. Of E. Probability Map
Name "static_var"
Name "distance"
Weights "weights######.dcf"
Step
SubRegion Id.

At the end of an iteration of *For Each Category,* a functor *Regional Categorical Map* stores the regional maps. Assign a **Global Map Name** to these maps. After *For Each Category* runs, the regional maps can be merged into an updated landscape map. The functor *Merge Regional Categorical Map* is in charge of merging the regional maps. Its **Global Map Name** must be the same as *Regional Categorical Map*. *Merge Regional Categorical Map* must be placed within *Region Manager* and, in order to ensure a proper sequence of processing, you need to establish a dependence effect between this functor and the *Regional Categorical Map.* A way to do this is by placing *Merge Regional Categorical Map* within a *Group* and linking this *Group* to *For Each Category* using a functor *Int* as a link. *Int* simply passes an integer constant from *For Each Category to Group*, establishing as a result a time-dependence between the executions of both.

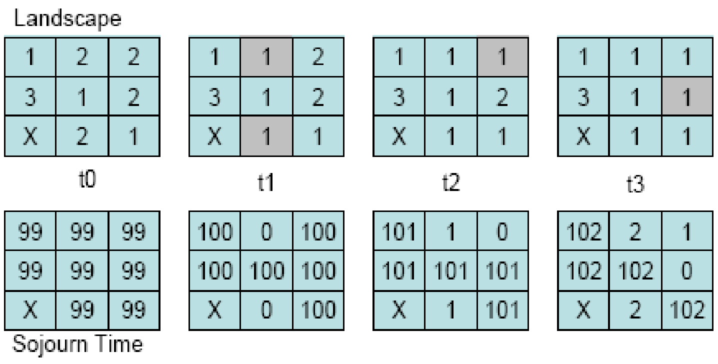


After the regional maps are merged, the model iterates and a new landscape map is fed back into *Mux Categorical Map* closing the loop. Therefore, the saved landscape map represents a mosaic of changed regional landscapes.

9.3 Using sojourn time

Load the model "simulate_deforestation_and_abandonment_using_sojourn_time.xml" from \Examples. The use of sojourn time implies that a specific transition or event will not occur unless a certain time has passed since a specific event has occurred.

The concept of sojourn time can be implemented using a map that keeps track of the time since a particular event has occurred (for example, a transition from state *i* to *j*). For each cell, a clock will count the time past after a particular event has occurred. In the end of iteration, this map is updated, if there is no new event, the model simply adds one to the cell's sojourn time, and if an event occurs, the clock is set to zero (fig. 16).



Landscape:

t0			t1			t2			t3		
1	2	2	1	1	2	1	1	1	1	1	1
3	1	2	3	1	2	3	1	2	3	1	1
X	2	1	X	1	1	X	1	1	X	1	1

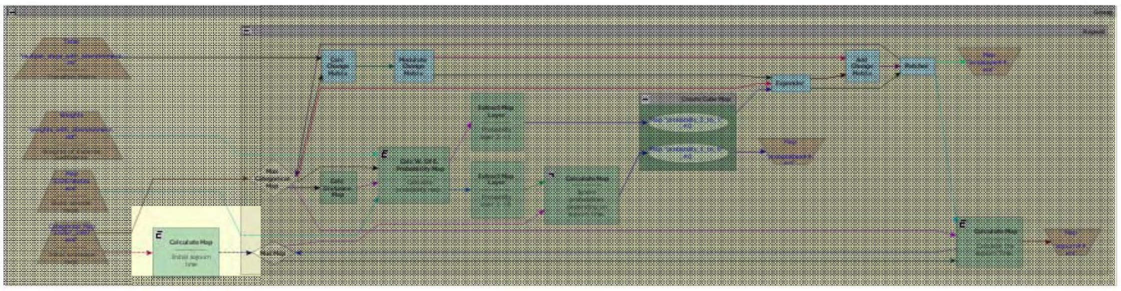
Sojourn Time:

t0			t1			t2			t3		
99	99	99	100	0	100	101	1	0	102	2	1
99	99	99	100	100	100	101	101	101	102	102	0
X	99	99	X	0	100	X	1	101	X	2	102

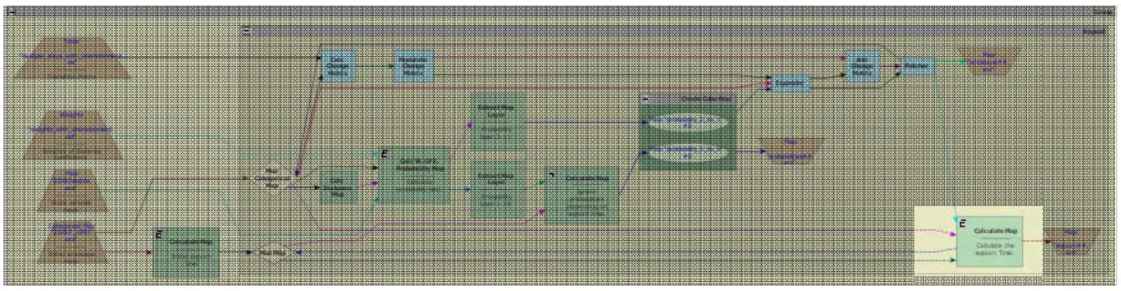
Fig. 16 – In this case, sojourn time map keeps tracks of the time past after a cell has changed to a new state. All cells begin with 99 time units.

The sojourn time can be used as a rule to constrain a transition from a state to the other only after a certain sojourn in a particular state. In this example, the sojourn time is used to influence the probabilities from deforested land (value 1) to abandon – regrowth (value 8). Land is only abandoned after 4 years in use: i.e. after deforestation has taken place (transition 2 to 1).

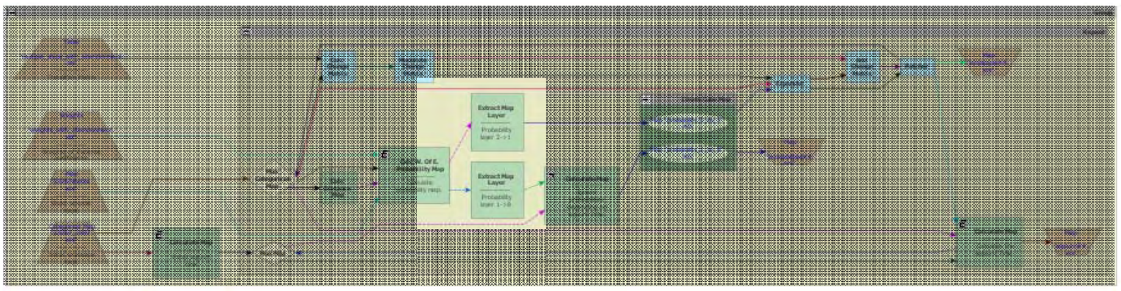
The sojourn map is initialized with high values (e.g. 9999). By doing so, it's possible to differentiate cells where a certain transition has never occurred from cells that have changed during the simulation. This might be useful for further analyses.



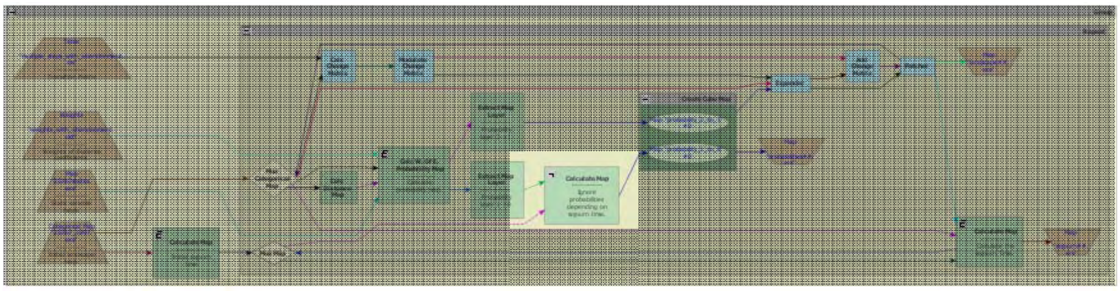
At the end of iteration, a new sojourn map is calculated by comparing the previous land-use map with the current one. Cells whose states have changed, the corresponding sojourn times are set to zero; otherwise they are incremented by one time unit.



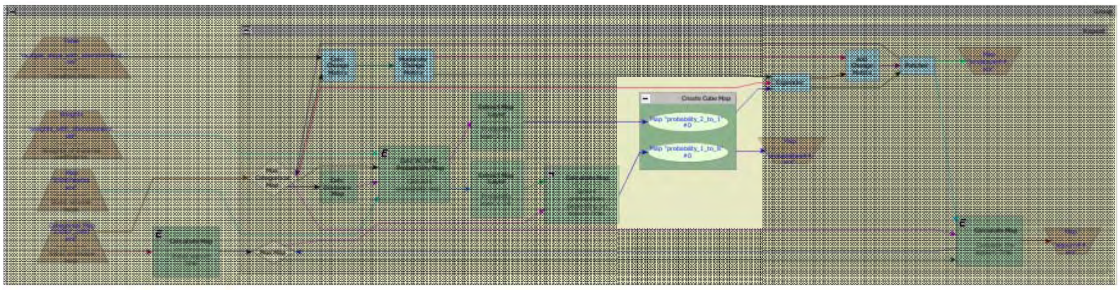
The sojourn time concept is applied to constrain a change. An easy way to do this is by modifying the corresponding probability map. As these maps are stacked in a cube raster set, you need first to extract the layers from the cube raster set, so the *Calculate Map* will be able to manipulate them separately.



After calculating the probability map using Calc W. of E. Probability Map, the layer of the transition from deforested to abandon (transition 1 to 8) undergoes a new calculation. This is a comparison of the sojourn time. If a deforested cell is older than 3 years, its probability is kept unchanged. If it is not, the probability to change to abandon is set to zero, impeding, as result, this transition to take place. **TIP:** The transition functors (*Patcher and Expander*) do not change cells with transition probabilities equal to zero.



Then, the probability layers (the original layer with transition 2 to 1 and the new layer with transition 1 to 8) are put together again as a new cube raster dataset, which is passed on to the transition functions, *Expander* and *Patcher*.



9.4 Using sojourn time and deterministic transitions

Load the model "simulate_def_aband_and_deterministic_transition.xml" from \Examples\ sim_defor_and_aband_using_sojourn_time_and_deterministic_transition

In this example, transition from abandon (land abandoned undergoes regeneration) to forest is deterministically determined based only on the sojourn time. So after a period in regrowth, the abandoned parcel has acquired a forest structure similar to a mature forest, thereby considered as forest. The minimum sojourn time for this transition to occur is 21 years (fig. 17).

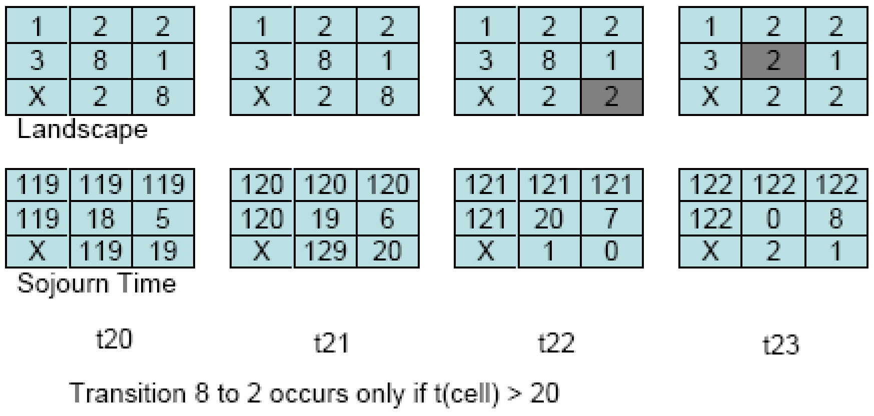
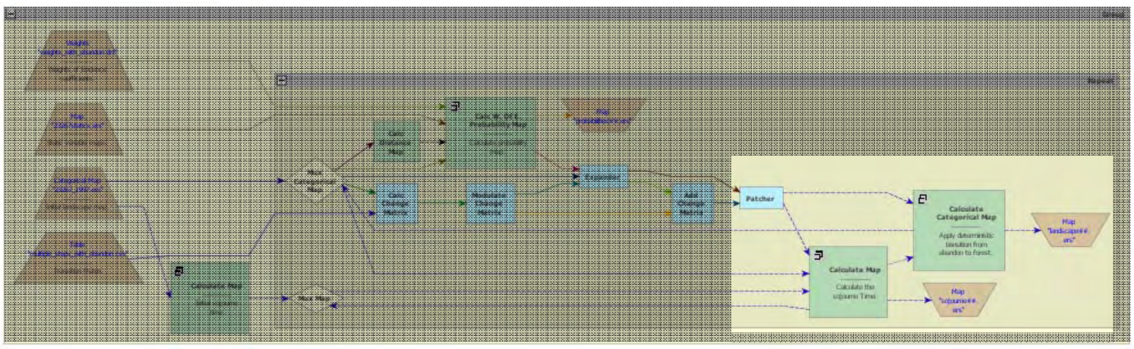


Fig. 17 –Transition from abandon to forest are deterministically determined solely based on sojourn time.

An additional transition from deforested land to abandon is added to the previous model. *Calculate Categorical Map* placed at the end of the model is employed to perform this transition. Its rule is the following:

if i1 = 8 and i2 > 20 then 2 else i1.

i1 is the landscape map and *i2* is the sojourn map (forest is 2, abandon is 8). Note that for simplicity, it's not possible to clear areas in abandon (transition 8 to 1). This could be added to the model as a new transition. Texeira *et al*. (2009) provide an example of modeling the landscape dynamics of the Atlantic forest, which includes multistate and transition from forest to various land-use.


Patcher
Calculate Categorical Map
Apply deterministic transition from abandon to forest.
Map "landscape##.ers"
Calculate Map
Calculate the sojourn Time
Map "sojourn##.ers"

9.5 Using local saturation

Load the model "simulate_deforestation_using_local_saturation.xml" from \Examples\advanced\local_saturation

Local saturation prevents a change from occurring within a specific region, in which a class area is greater than an established threshold (fig. 18). This feature is useful to simulate diffusion process as well as to establish a minimum forest remaining area (as established by the Brazilian forest code for private properties). Local saturation can be implemented by reducing the probability of a transition using an asymptotic function as follows:

$$P2 = P1 * (Li - Oc) / (Li + Oc), \text{ case } Li >= Oc$$

$$P2 = 0, \text{ case } Li < Oc$$

where *P2* is the new probability, *P1* is the original probability, *Li* is the maximum number of cells of a particular class that is allowed to occur within a certain map neighborhood and *Oc* is the number of cells of that class within that neighborhood.

In this example, deforestation must stop in a local region (in this case a window size of 59.25 hectares or 9 cells - 3x3 window) when 50% of this region is deforested.

For each model step, the amount of deforested cells is calculated in every 3x3 window of the landscape map. In this case, the threshold consists of five cells as follows:

$$P2 = P1 * (5 - Oc) / (5 + Oc), \text{ case } Li >= Oc$$

$$P2 = 0, \text{ case } 5 < Oc$$

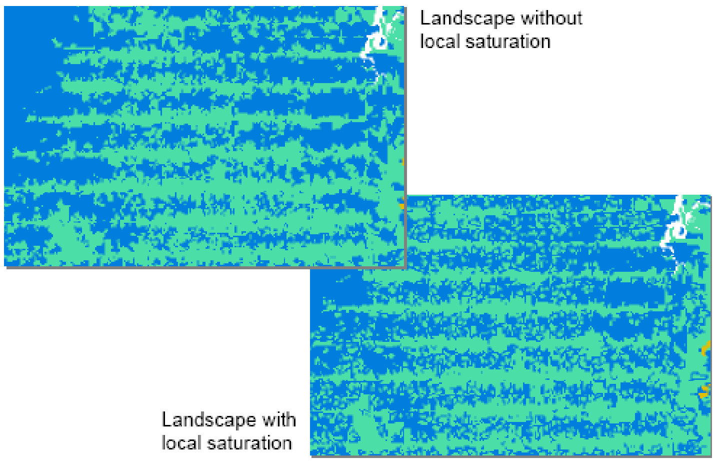

Landscape without local saturation
Landscape with local saturation

Fig. 18 –Two landscapes without and with local saturation effect.

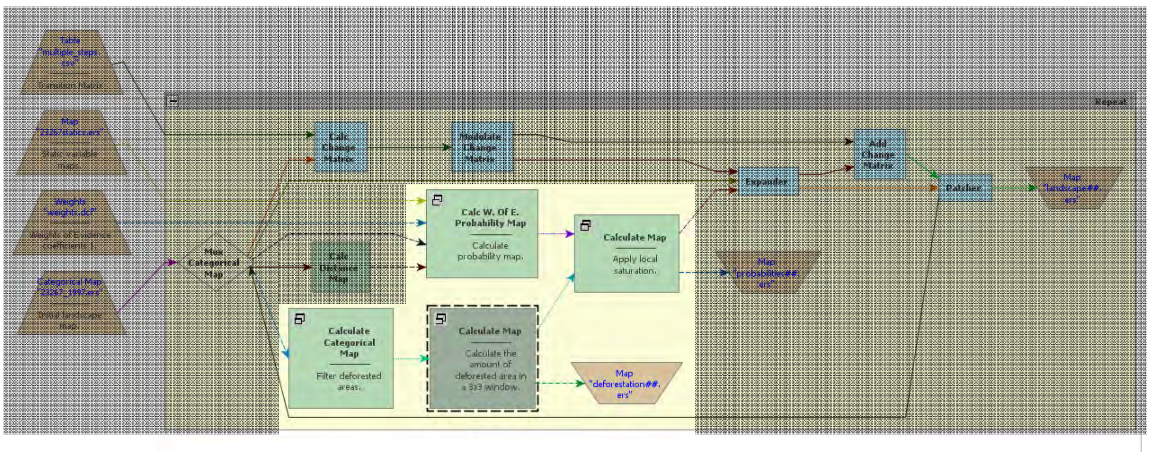
Three *Calculate Map* functors are added to a deforestation simulation model in order to incorporate the saturation effect. Open them to see their equations. The first one assigns “1” to deforested cells and “0” to forest. The second counts the amount of cells deforested within a window size 3x3. It uses a neighborhood operator as follows:

nbCount(i1, 3, 3)

where *nbCount* is the neighborhood counting operator, *i1* is *map # 1*, and 3,3 is the window size in cells. You could easily increase the neighborhood size, changing these values. Then, a third *Calculate Map* is added to apply the local saturation rule on the probability map as follows:

if v1 - i1 >= 0 then i2 * (v1 - i1) / (v1 + i1) else 0

Where *v1* is the saturation value in number of cells (in this case equal to 5).



As a result, deforestation will cease in local regions as they reach a saturation threshold and move to new areas with lower probability of deforestation, simulating therefore the diffusion of deforestation.

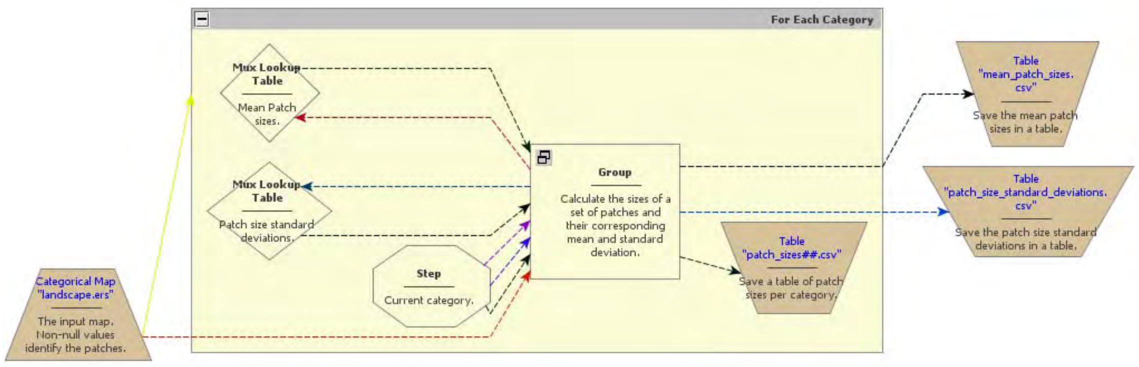
10. Landscape metrics in Dinamica EGO

What will you learn?

- Calculate landscape metrics
- Mean patch size
- Mean patch edge length
- Fractal dimension
- Functors:
    - *Log policy*

Landscape metrics (McGarigal and Marks, 1995) can be useful tools to assess the quality of habitats when extensive biodiversity inventories or ecological data are not available or are difficult to obtain, as landscape metrics are strongly related to biodiversity indicators (Metzger, 2006). For example, landscape metrics can be applied to identify the best landscape configuration for forest species conservation ─ independently of the perceptions of individual species─, which is hypothetically a landscape with: i) a high forest cover; ii) a small number of forest fragments; iii) a high largest forest patch index that can support stable populations and be a source for small patches; iv) a large mean forest patch area; and v) a high mean forest proximity index (Texeira et al., 2009). Therefore, the application of these metrics consists of a powerful tool to describe the consequences of land-use and land-cover dynamics on the conservation of biodiversity. In this context, we introduce here a series of models designed in Dinamica EGO to calculate landscape metrics. Instead of providing a black box solution, all these metrics are developed using Dinamica EGO modeling language, thus they serve as templates to derive a wide variety of metrics.

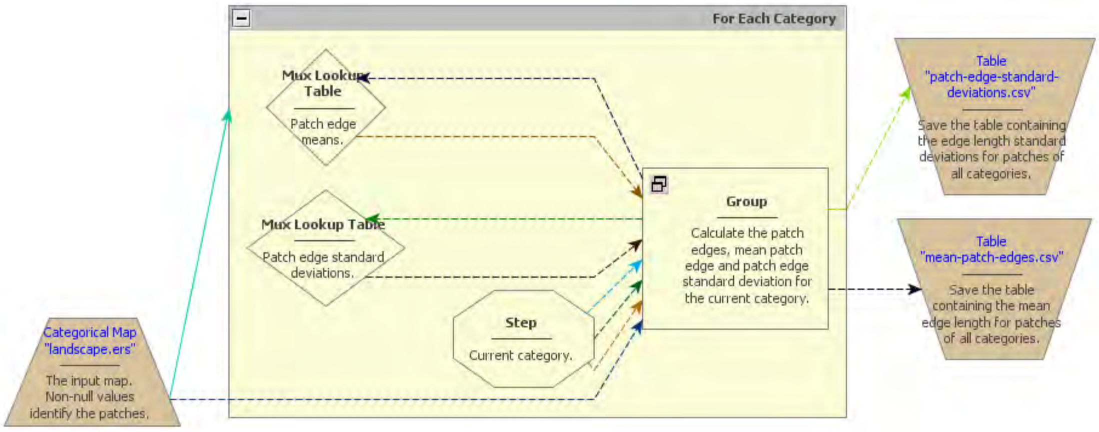
Load the model “calc_mean_patch_sizes_and_standard_deviations.ego” from Examples\landscape_metrics\calc_mean_patch_sizes_and_standard_deviations


For Each Category
Mux Lookup Table
Mean Patch sizes.
Mux Lookup Table
Patch size standard deviations.
Step
Current category.
Group
Calculate the sizes of a set of patches and their corresponding mean and standard deviation.
Categorical Map "landscape.ers"
The input map. Non-null values identify the patches.
Table "patch_sizes##.csv"
Save a table of patch sizes per category.
Table "mean_patch_sizes.csv"
Save the mean patch sizes in a table.
Table "patch_size_standard_deviations.csv"
Save the patch size standard deviations in a table.

This model calculates the size of each patch of a landscape class and for each class the mean patch size and patch size standard deviation. A table is output for each metric. Examine this model by opening the *Group* functor and then run it. The functor *Log Policy* present within *Group* is used in order to run the model in silent mode, thus increasing its processing speed.

Load now the model “calc_mean_patch_edges_and_standard_deviations.ego” from Examples\landscape_metrics\ calc_mean_patch_edges_and_standard_deviations.

This model calculates the edge length of each patch of a landscape class and for each class the mean patch edge length and patch edge length standard deviation. Examine this model by opening the *Group* functor and then run it.


For Each Category
Mux Lookup Table
Patch edge means.
Mux Lookup Table
Patch edge standard deviations.
Step
Current category.
Group
Calculate the patch edges, mean patch edge and patch edge standard deviation for the current category.
Categorical Map "landscape.ers"
The input map. Non-null values identify the patches.
Table "patch-edge-standard-deviations.csv"
Save the table containing the edge length standard deviations for patches of all categories.
Table "mean-patch-edges.csv"
Save the table containing the mean edge length for patches of all categories.

These two set of metrics are input for more complex metrics, such as fractal dimension and largest patch index. For example, fractal dimension for a landscape class can be estimated using the perimeter-area relationship, so If sufficient data are available, the slope of the line obtained by regressing log(P), patch edge lengths, on log(A), patch areas, is equal to 2/D (Burrough, 1986). Check other landscape metrics models, such as mean patch distance, largest patch index, in Examples\landscape_metrics.

11. REDD case study

What will you learn?

- How to project deforestation rates based on socioeconomic variables
- Convert gross rates into net rates
- How to develop a carbon bookkeeping model
- Functors:
    - *Calc Neighborhood*
    - *Calc Spatial Lag*

Tropical deforestation is the second source of anthropogenic GHG (Greenhouse Gases). Seven to twenty eight percent of the worldwide CO_2 emissions come from tropical deforestation, what is equivalent to 0.5 to 2.4 billion of tons of carbon emitted per year (Houghton *et al.*, 2005). A proposal to compensate countries for reducing emissions from deforestation and forest degradation (REDD) was initially presented by IPAM and other institutions at the ninth Climate COP (Conference of the Parties) in Milan, 2003 (Santilli *et al*. 2005; Moutinho & Schwartzman 2005). This proposal recommended that developing countries that were able to reduce deforestation below a historical baseline over a period of time would be eligible to receive financial compensation from the international community through a market of carbon credits. Later, this proposal was officially endorsed by Papua-New Guinea, Costa Rica, and other tropical countries during the eleventh COP in Montreal, 2005 (Silva-Chavez and Petsonk 2006; Schlamadinger *et al*. 2007; Skutsch *et al* 2007; Sedjo and Sohngen 2007). In December 2007, the thirteenth Climate COP held in Bali, Indonesia, laid out a road map to a post-Kyoto climate protocol, beginning 2013, which stressed the need to pursue mechanisms to provide incentives for developing countries to reduce carbon emissions from deforestation. This occasion triggered a worldwide discussion on how to establish evaluation methods, rules and economic means for REDD programs.

One of the main matters of REDD consists of measuring the contribution of a project (country, state or region level) to reducing carbon emissions from deforestation and forest degradation in order to credit this effort. This topic is one of the most contentious issues of REDD because there are several ways to weigh this reduction effort depending on the region's deforestation history, stocks of forest carbon, and potential for future deforestation (e.g. Cattaneo, 2008). For example, the concept of reducing emissions below a historical baseline applies well to countries with a historically high rate of deforestation, thus with a large margin to reduce its emissions from deforestation, such as the case of Brazil with a baseline of 19.500 km^2year^{-1}. On the other hand, countries with large expanses of tropical forest and current low deforestation rates (e.g. Peru) would not be eligible to be compensated, since there is no additionality in its effort to curb deforestation (fig. 19). However, deforestation in Peru is likely to increase after the completion of the paving of Interoceanic highway, which links Brazil to the Pacific Ocean, as well other infrastructure projects underway in the region (fig. 20). These projects have already triggered a large inflow of people to the Madre de Dios department in Peru.

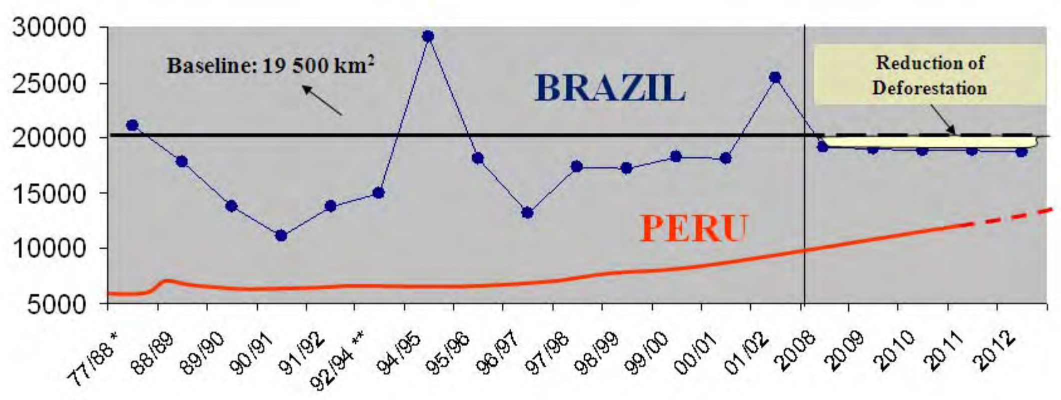


Fig. 19 –Different deforestation trajectories in Brazil and Peru and their implications for REDD proposals.

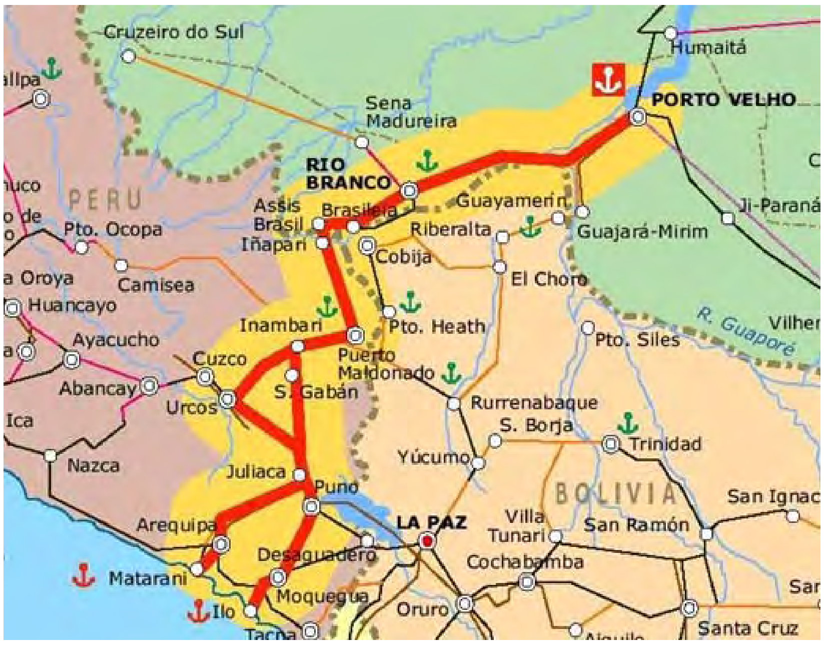


Fig. 20 –Infrastructure projects in the MAP region. MAP stands for Madre de Dios (Peru), Acre (Brazil) and Pando (Bolivia).

To perform an ex ante evaluation of the potential for future deforestation under a Business-as-Usual (BAU) scenario, pioneer REDD projects, such as the JUMA conservation PPD – Project Design Document (FAS, 2008) adopted the use of simulation models (Soares-Filho *et al*., 2006). In this case, a set of assumptions is established to differentiate the BAU from the governance scenario, such as road paving, agricultural expansion, population movements, expansion and consolidation of protected areas and the effectiveness of public policies in curbing deforestation. The model simulates deforestation trajectories under the modeled scenarios and calculates their respective carbon emissions. So, instead of a historical baseline, the potential for reduction is compared by subtracting the accumulated amount of emissions under the governance scenario from the one under the BAU scenario by a specific future year (in general 2050).

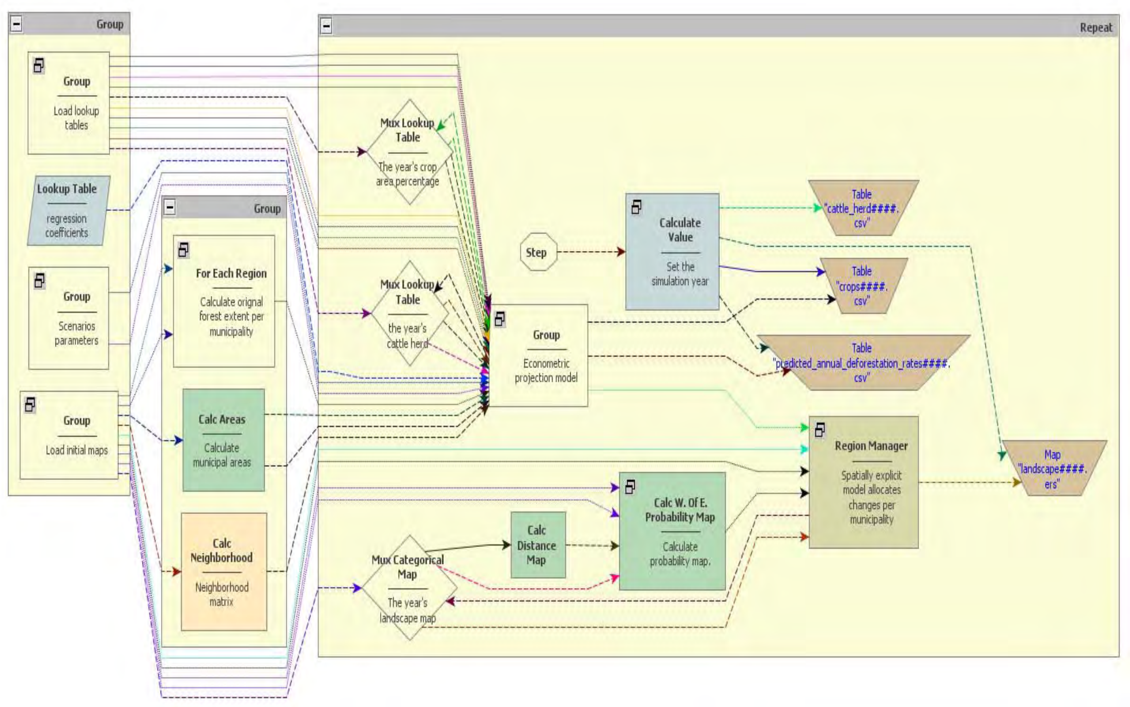
Although simulation models play an important role in modeling the effects of alternative land-use policy scenarios on the landscape dynamics, this methodology should be applied with caution to REDD projects. The need to estimate potential future emissions as a way to measure the contribution of a public policy or conservation initiative to REDD, such as the creation of a protected area, is rapidly disseminating the use of simulation models as a tool for REDD projects. Several commercial and non-commercial packages are available for developing spatial simulation models. However, there is no ready solution for a specific REDD project (despite that some vendors say so). In addition to simulating the effects of spatial determinants on the location of deforestation (see lesson 7), there is a need to model the local, regional, and even international drivers of deforestation. This is far more difficult and relies heavily on the availability of temporal socioeconomic data at several scales as well as wall-to-wall deforestation time series. Those models must be built from the ground (i.e. using bottom-up approaches rather than top-down models), incorporating our knowledge of the proximate and underlying causes of deforestation, and must pass through validation not only in terms of their spatial prediction, but also regarding the power to predict the recent deforestation trajectory based on changes in the socioeconomic and political context. Even so, simulation models are no crystal ball, modeled future trajectories must be regarded as a likely possibility only; beyond that it is all speculation.

In this context, the model presented here aims to demonstrate the possibilities of Dinamica EGO in representing land-change phenomena, no matter their complexity. Therefore, instead of a ready solution for a REDD project, which probably will not work anyway, the Dinamica EGO modeling platform provides the means to materialize in a computer realm our knowledge about complex dynamic phenomena, such as the way the local, regional and international socioeconomic and political contexts interact, producing as a result deforestation.

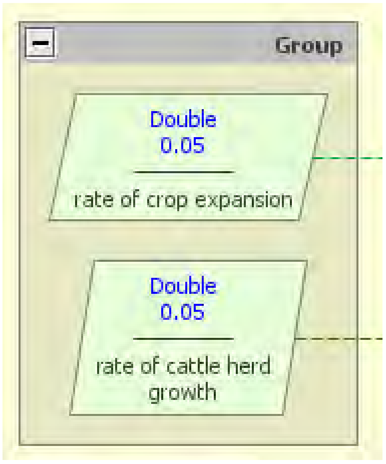
11.1. Developing an econometric projection model that predicts deforestation rates based on changes in the socioeconomic context of municipalities

In this example, an econometric model is coupled to a spatially-explicit simulation model of deforestation. The econometric projection model predicts deforestation rates based on changes in the socioeconomic context of municipalities (Soares-Filho *et. al*, 2008, Soares-Filho *et. al*, 2010). A spatial lag regression is applied to compute the influence of five variables on the deforestation trajectory: Crop area expansion, cattle herd growth, percent of protected areas, proximity to paved roads, and migration rates. A spatial neighborhood matrix allows the model to incorporate the influence of the socioeconomic context of neighboring municipalities in the prediction of deforestation rates within a certain municipality.

Load the model "simulate_deforestation_under_socioeconomic_scenarios.xml" from \ Examples\REDD_case_study. This model is composed of three main parts: the input data, pre-calculation, and the simulation model itself.



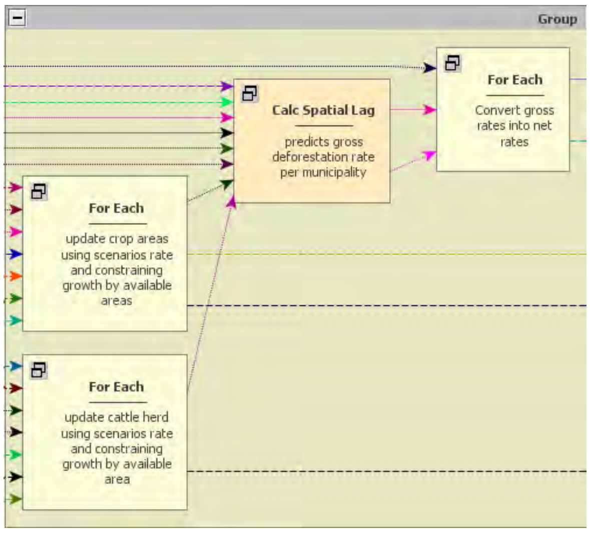
In this simplified version of Soares-Filho *et al*. (2008), the user can modify the scenario by changing the annual rates of crop expansion and cattle herd growth, which are input to the model. Other variables could be also changed by editing the input lookup tables.


Group

Double
0.05
rate of crop expansion

Double
0.05
rate of cattle herd growth

The pre-calculation group calculates the original forest extent per municipality, the municipality area, and the neighborhood matrix (*Calc neighborhood*) that defines which municipalities are neighbors. These are going to be inputs for the projection model. Open the group named "Econometric projection model".

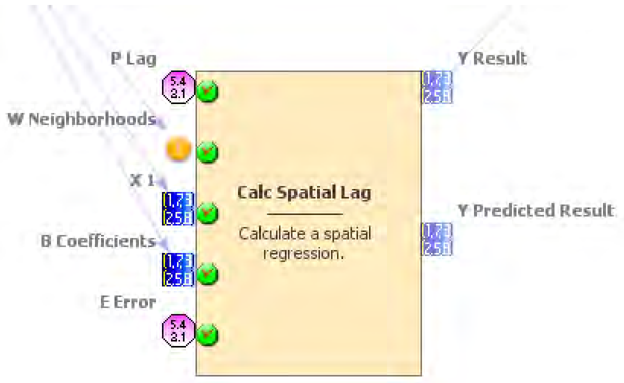

Group
Calc Spatial Lag
predicts gross deforestation rate per municipality
For Each
Convert gross rates into net rates
For Each
update crop areas using scenarios rate and constraining growth by available areas
For Each
update cattle herd using scenarios rate and constraining growth by available area

This *Group* contains three *For Each* and one *Calc Spatial Lag*. The two first *For Each* update the cattle herd and crop area lookup tables and calculate their annual rates of change, which are input to the spatial lag regression. **TIP:** *For Each* browses the elements of a table allowing its manipulation. In addition to the lookup tables of the five independent variables, *Calc Spatial Lag* receives as input the lag coefficient, the neighborhood matrix, an initial *x1* dependent variable table, the regression coefficients, and a random error term. This functor represents a spatial lag regression equation as follows (Anselin, 2002):

$y = \rho Wy + X\beta + \varepsilon$

Where ρ is the autoregressive coefficient, *W* is a first order neighborhood matrix, *y* the dependent variable, *X* the matrix of observations for the independent variables, β the vector of regression coefficients and ε a random error term. In this equation the term ρW is calculated in an iterative way using moving averages of the *y* responses from the neighboring municipalities. In this case, instead of a classical linear model, a spatial lag regression was adopted since the regression model failed the autocorrelation tests (Anselin, 2002).

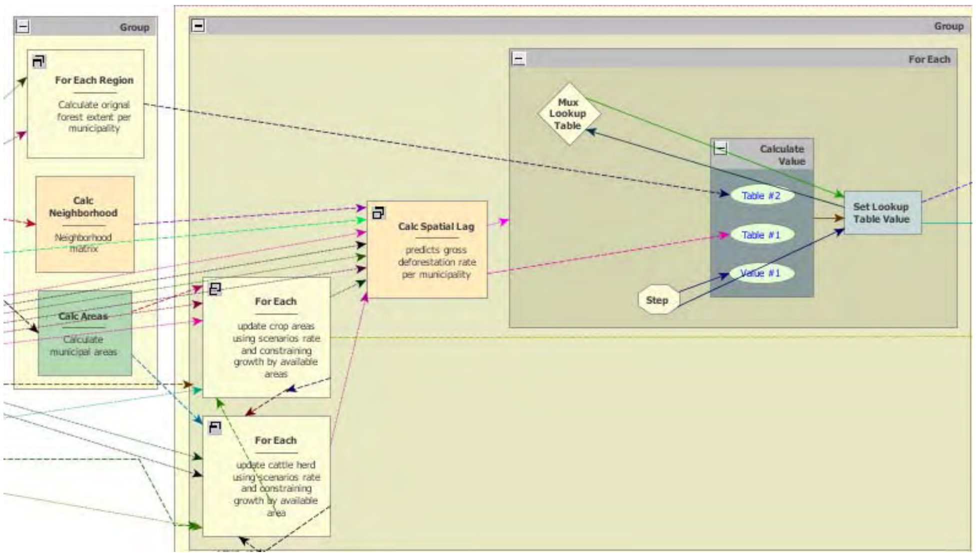
Dinamica EGO does not provide a method to develop a spatial lag regression, but only to solve the equation, which was developed using *Geoda* (*www.geoda.uiuc.edu*).



Finally, the third *For Each* converts gross deforestation rates output from *Calc Spatial Lag into* net deforestation rates *using the following formula:*

if t1[v1] / t2[v1] > 1 then 1 else if t1[v1] / t2[v1] < 0 then 0 else t1[v1] / t2[v1]

where *t1[v1]* is the gross rate and *t2[v1]* the original forest extent of a municipality.

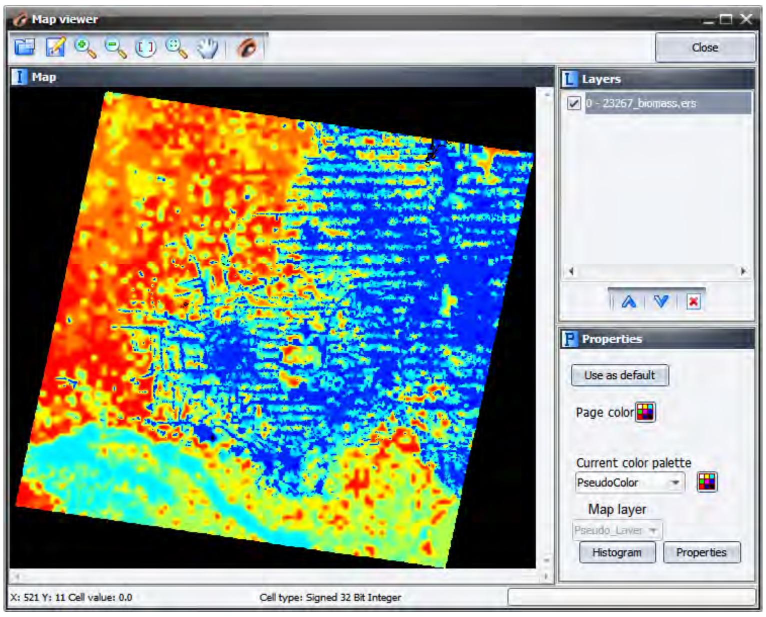

Group
For Each Region
Calculate original forest extent per municipality
Calc Neighborhood
Neighborhood matrix
Calc Areas
Calculate municipal areas
Group
For Each
Mux Lookup Table
Calculate Value
Table #2
Table #1
Value #1
Step
Set Lookup Table Value
Calc Spatial Lag
predicts gross deforestation rate per municipality
For Each
update crop areas using scenarios rate and constraining growth by available areas
For Each
update cattle herd using scenarios rate and constraining growth by available area

The econometric model passes on the predicted deforestation rates to a spatially explicit model that allocates deforestation in each municipality using the subregion approach of lesson 9.2.

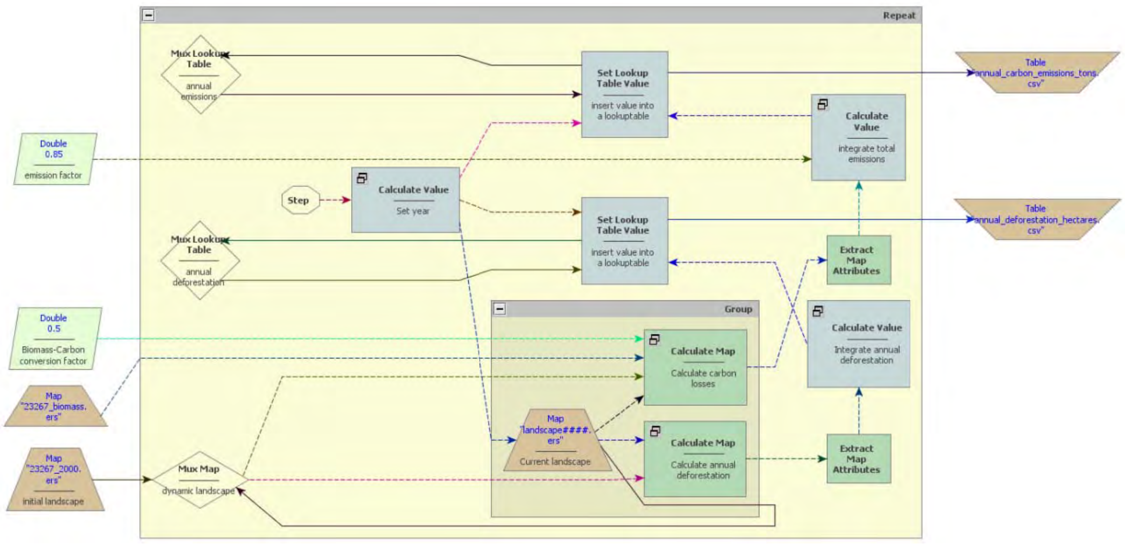
11.2. Developing a carbon bookkeeping model

Load the model "carbon_bookeeping_model.xml" from \ Examples\REDD_case_study

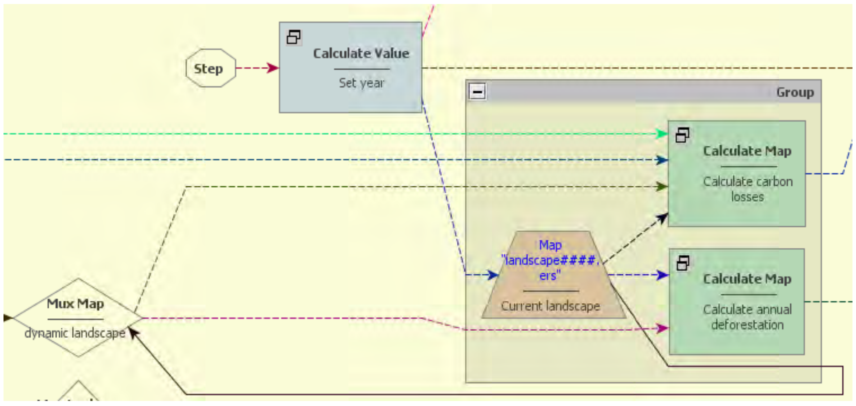
This model calculates annual carbon emissions by identifying annual deforestation and then overlaying these areas on a map of forest carbon biomass – figure below (Saatchi *et al*., 2007), and assuming that carbon content is 50% of wood biomass (Houghton *et al*., 2001) and that 85% of the carbon contained in trees is released to the atmosphere with deforestation (Houghton *et al*., 2000).



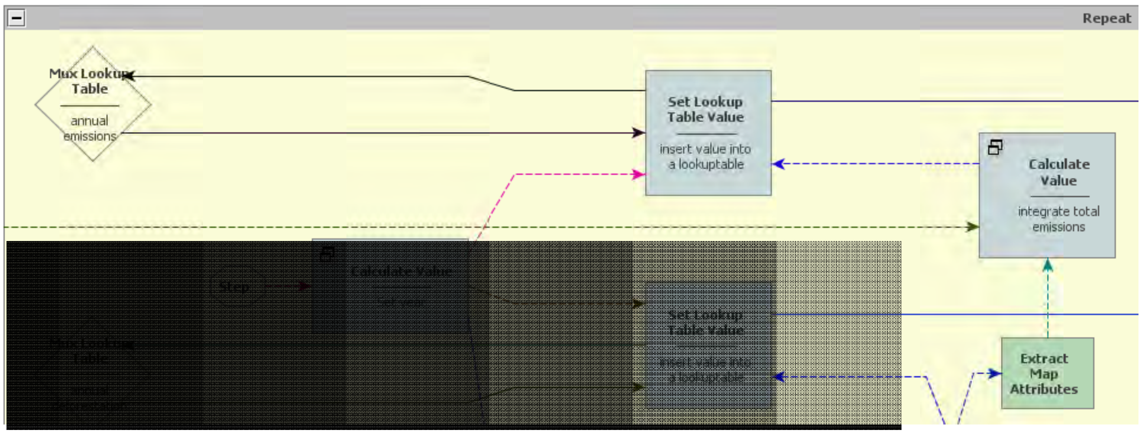
In order to calculate annual deforestation, the model compares in each time step the current land-use map with the previous one. Dinamica EGO allows the loading of multiple maps using *Load Map* within *Repeat* and passing *Step* as a pointer to the name file that has the model step number as its suffix. Note that in this case the suffix has 6 digits to bear the simulation year (2002-2020).



A *Load Map* is placed within a *Group* to ensure a proper order of execution. The previous land-use map is kept in *Mux Map*, so both *Calculate Map* functors in this container receive the previous land-use map as **i1** and the current as **i2**.



After annual deforestation cells are indentified, the model picks up the corresponding biomass stocks in the biomass map and convert them into carbon and then into emissions. *Extract Map Attribute* is applied to calculate the total amount of cells and *Calculate Value* integrates those figures on an annual basis. Its output is passed to *Set Lookup Table* that updates a table with annual carbon emissions (Fig. 21).



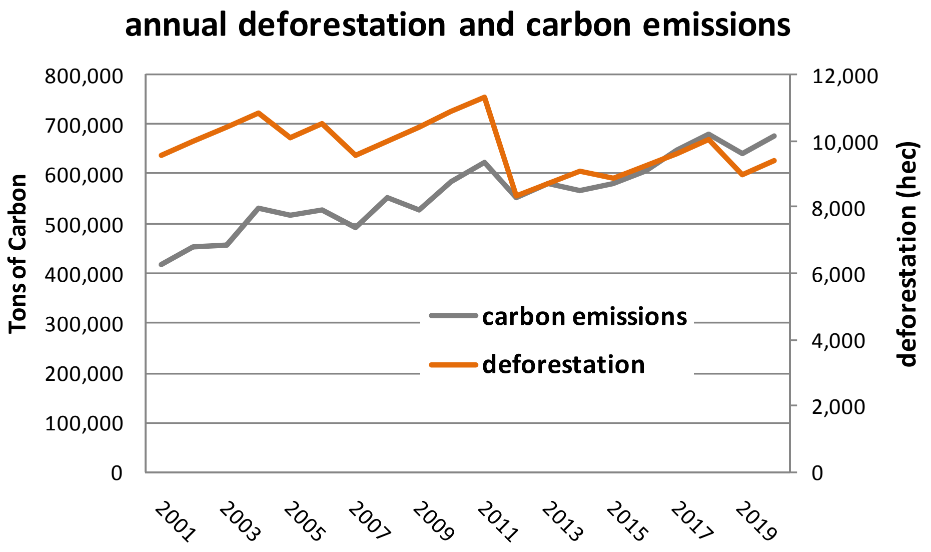


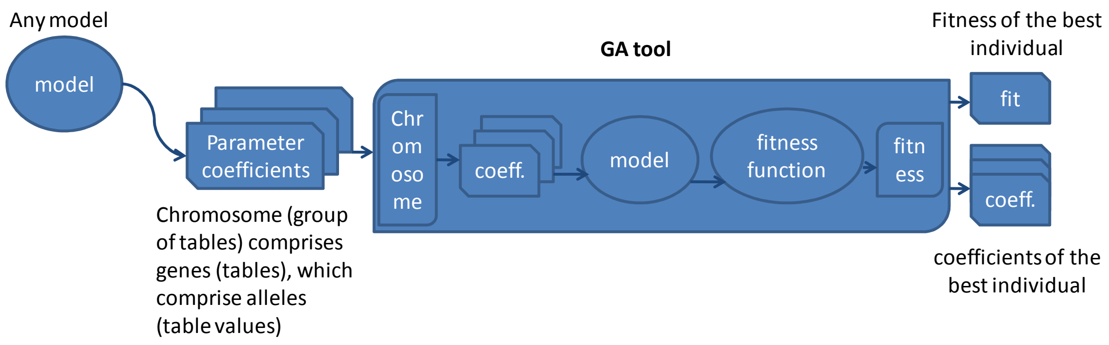
Fig. 21 –Annual deforestation and emissions calculated for the model region.

12. Heuristic calibration of models by using Genetic Algorithm

What will you learn?

- How to calibrate a land change model by using *Genetic Algorithm*
- How to set up a gene from model parameters
- How to get model parameter coefficients from a gene
- How to pass the model fitness to the GA tool
- Functors:
  - *Create Lookup Table Group*
  - *Extract Lookup Table from Lookup Table Group*
  - *Get Current Individual*
  - *Set Fitness*

Within the class of hard predictors (Eastman et al., 2005), the *Genetic Algorithm* (*GA*) *tool* provides a powerful means of calibrating environmental models. By mimicking the principle of biological evolution (Koza, 1992), *GA tool* uses massive computing and heuristics to seek for a global optimum solution for a set of model parameters. Dinamica EGO´s *GA tool* consists of a container, which requires the placement of a sequence of functors within it and, in particular, of two associated functors: *Get Current Individual* and *Set Fitness*. Fig. 21 depicts a model calibration scheme using *GA tool*. First, one needs to get the coefficients from the model parameters to be calibrated and assemble these coefficients in tables. Thus each model parameter will represent an allele in a table that corresponds to a gene. In turn, these tables are assembled by using *Create Lookup Table Group* in a group of tables to form a chromosome. This group of tables is an input to *GA tool*. *GA tool* spawns a population based on the genotype passed in a group table. Inside *GA tool*, *Get Current Individual* is placed to get the genes from the individuals of a generation. Other functors, such as *Extract Lookup Table from Lookup Table Group*, are sequenced to catch the parameter coefficients and pass them to the model, which is executed once per individual. An evaluation function is coupled to the output of the model and its result is passed to *Set Fitness*, which returns the fitness value to *GA tool* for the selection process. The internal sequence of functors will iterate a number of times equal to the number of individuals multiplied by generations, as specified in *GA tool´s* input ports. When *GA tool* terminates, it will output the fitness of the overall best individual as well as the group of tables that comprises its genes.



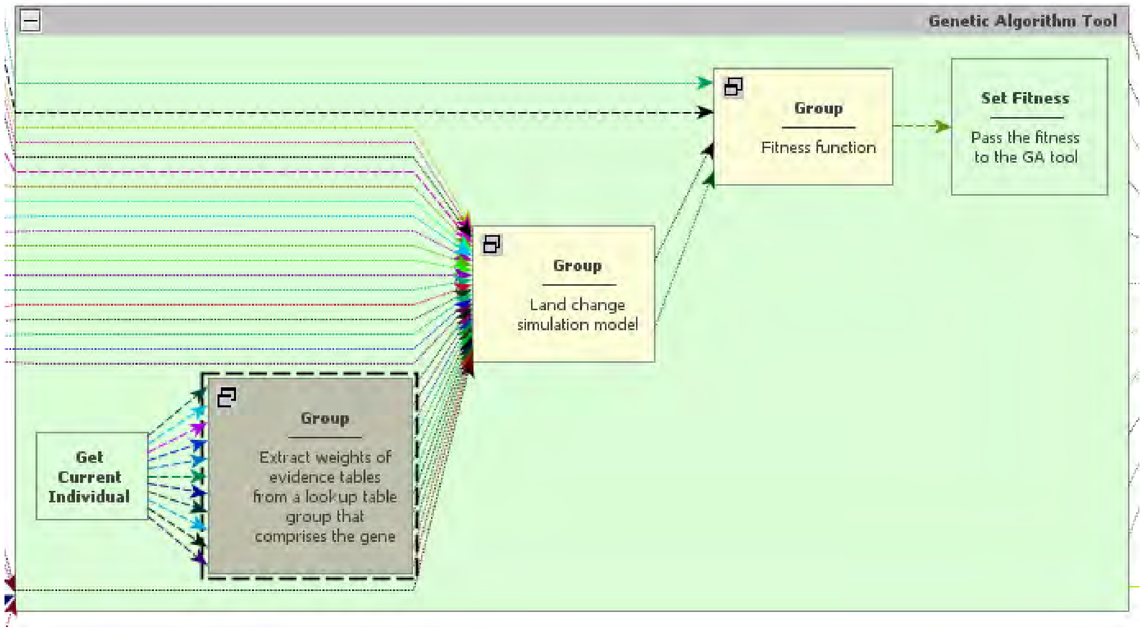
Any model
model
GA tool
Fitness of the best individual
fit
Parameter coefficients
Chr om oso me
coeff.
model
fitness function
fitn ess
coeff.
Chromosome (group of tables) comprises genes (tables), which comprise alleles (table values)
coefficients of the best individual

Fig. 21 –Diagram of *GA tool* with inputs, internal model and fitness function, and outputs.

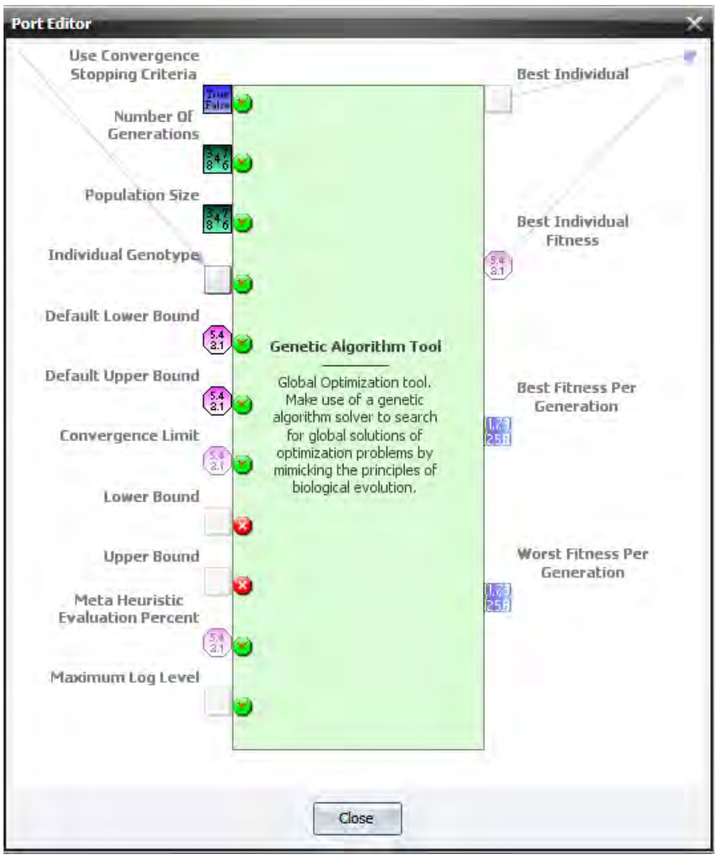
In the example provided, we make available an application of *GA tool* to calibrating a model of deforestation. The aim of this exercise is to demonstrate the potential of *GA tool* for calibrating any type of model ─ as long as there is sufficient computer power to run a model together with its evaluation function numerous times ─ as well as the overspecialization problem that may arise when using such hard predictors, as artificial neural network and GA itself.

First, load the model "Cal_Reciprocal_fitness1x1.ego" from "\ Examples\ Genetic_Algorithm\WEofE". This model calculates the reciprocal fitness of a deforestation model that was calibrated using the Weights of Evidence method – a soft predictor. Run the model and access its output "Reciprocal_fitness1x1.csv" using either a spreadsheet or enabling table viewer on the output port of *Set Key 1*. The concept of this validation measure is provided in 7.6 and 7.7 of lesson 7. The fitness obtained for this model is 0.2060. Open the Weights of Evidence tables in "originals\tables" and their corresponding maps in "originals\maps" under the folder "Genetic_Algorithm". In this model, *Calculate Map* replaces *Calc W. E. Probability Map*. Open it to see the equation that integrates the weights of evidence to produce the transition probability map. The weights of evidence coefficients are input as separate tables, so they can form a group of table and thereby the gene.

Now, open "GAReciprocal_fitness1x1.ego" from "Genetic_Algorithm\GAknn\Reciprocal_fitness1x1". Compare the structure of this model with the diagram from Fig. 21. Open *GA tool*. This container envelops three *Groups* and two functors. *Get Current Individual* obtains the gene of an individual pertaining to a generation and passes it to a sequence of functors that extract the lookup tables that compose the gene. A land change simulation model receives those tables as input and its execution results are passed to a fitness function that assesses its spatial performance. In turn, this function returns the fitness measure that is caught and passed to *GA tool* by *Set Fitness*.


Genetic Algorithm Tool
Group
Fitness function
Set Fitness
Pass the fitness to the GA tool
Group
Land change simulation model
Group
Extract weights of evidence tables from a lookup table group that comprises the gene
Get Current Individual

Click on *GA tool* with the Edit Functor Ports. As input, *GA tool* receives **Number of Generations** (30) and **Population Size** (100), i.e. number of individuals per generation, and **Use Convergence Stopping Criteria**. This last parameter forces *GA tool* to terminate if GA evolution becomes asymptotical, as defined by the **Convergence Limit** (0.99) that must be achieved within the span of generations established by the **Number of Generations**. **Meta Heuristic Evaluation Percent** enables the model to calculate the fitness for a percentage of individuals by use of a meta-heuristic estimation method known as KNN (K-Nearest Neighbor), thus saving computer time. Additional inputs for this model are: **Default Lower Bound** and **Default Upper Bound**, respectively set at -5 and 5. These parameters set a range within which the allele values may vary.



Global Optimization tool. Make use of a genetic algorithm solver to search for global solutions of optimization problems by mimicking the principles of biological evolution.

Run this model. It may take a while depending on the capabilities of the computer system used. Remember that, as a hard predictor, *GA tool* uses massive computing, which in principle would demand a high-performance computer system. Nevertheless, Dinamica EGO 1.6 is designed to take advantage of computer power, such as dual or more processors, extended virtual memory, and pre-compilations of algebraic and logical equations, making it feasible to run the current **GA tool** model even on a laptop computer. Follow the log report and open the file "Reciprocal_fitness_of the_overall_best_individual.csv". The overall best individual attained a fitness of 0.3439, and hence an increase of 66% with respect to the Weights of Evidence method. Now, let´s validate this model.

Note: Model fitness can be increased by lowering **Prune Factor** in *Patcher*. This diminishes model stochasticity, but often results in less realism.

Open "validation_fitness1x1.ego" from "Genetic_Algorithm\GAknn\Reciprocal_fitness1x1" and run it. What is the fitness now? Surprised? What happened?

The answer is simple. *GA tool* attained a very high fitness in the calibration process through overspecialization. That is, it adapted so well to this specific situation given by the map of changes from 1997 to 2000, that the probability map obtained with *GA tool* results became useless when applied to model validation, which is performed by using deforestation from 2000 to 2003. Thus, although *GA tool* can achieve high scores in the calibration process, it will become overspecialized if the formation of genes in the reproduction process (mostly through mutation and crossover) is not constrained within a certain range of variation from the characteristics of the primeval individual.

Dinamica EGO´s *GA tool* provides a means to overcome overspecialization by enabling the user to specify an envelope of maximum variation to the new genes that will be formed during the reproduction process. So, instead of default values, this envelop is delimited by two group tables, consisting, respectively, of the upper and lower gene bounds.

Open model "GAReciprocal_fitness1x1.ego" from "Genetic_Algorithm\GALimitedRanges120\Recipocal_fitness1x1" and examine *Group LowerBoundEnvelope.* Open it. Each *Calculate Lookup Table* establishes a lower bound for a weights of evidence table by using the following equation:

t1[line] - t1[line] * 1.2

Likewise, the *Group UpperBoundEnvelope* will set an upper bound as follows:

t1[line] + t1[line] * 1.2

The use of these tables as gene bounds will constrain the new genes to an envelope of ±1.2 times the values of the original weights of evidence coefficients, thus providing a trend around which the global optimum solution must be found. Although this constraint will result in lower calibration scores (Run "validation_fitness1x1.ego" located in GALimitedRanges120\Reciprocal_fitness1x1"), it will tame the *GA tool* engine, allowing it to improve the Weights of Evidence result for application to a general prediction process (Figs. 22, 23).

In conclusion, hard predictors like *GA tool* must incorporate prior knowledge in order to overcome overspecialization. When this is taken into consideration, *GA tool* can really push the envelope of model optimization. Use your expertise to develop other *GA tool* calibration processes following the scheme of Fig. 21 and examples provided.

GA tool is a contribution from the master´s degree dissertation of Flávio Oliveira at the graduate program on Modeling Environmental Systems of Universidade Federal de Minas Gerais (www.csr.ufmg.br/modelagem).

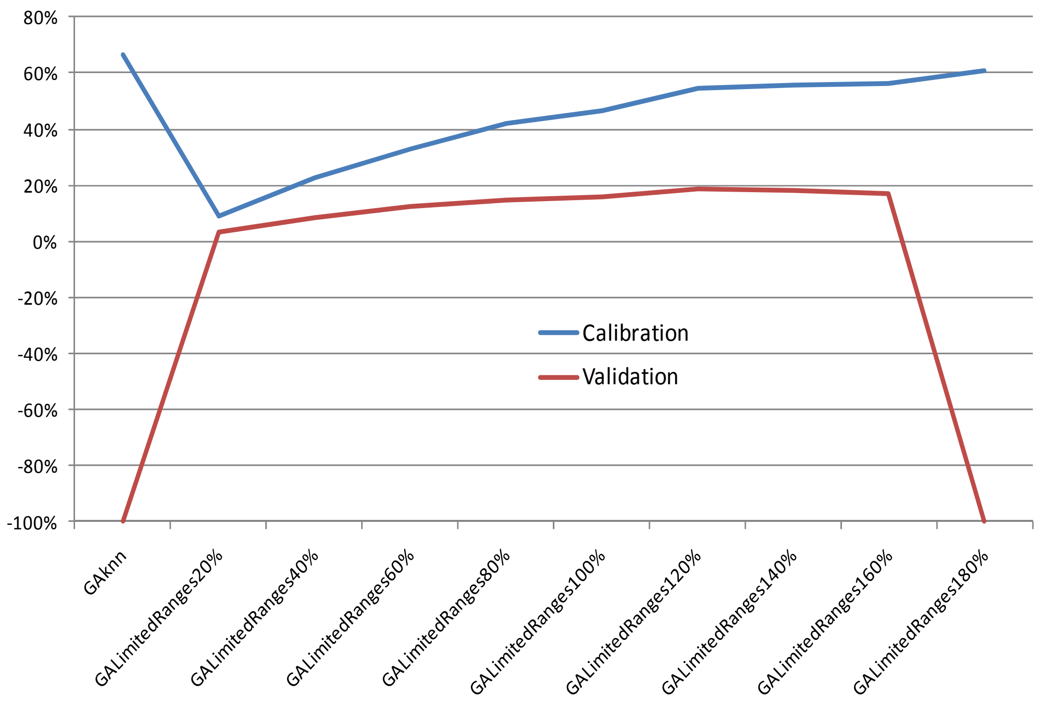


Fig. 22 - *GA tool* performance with respect to Weights of Evidence method (baseline). GA Limited ranges runs use an envelope of variation around the characteristics of the primeval individual.

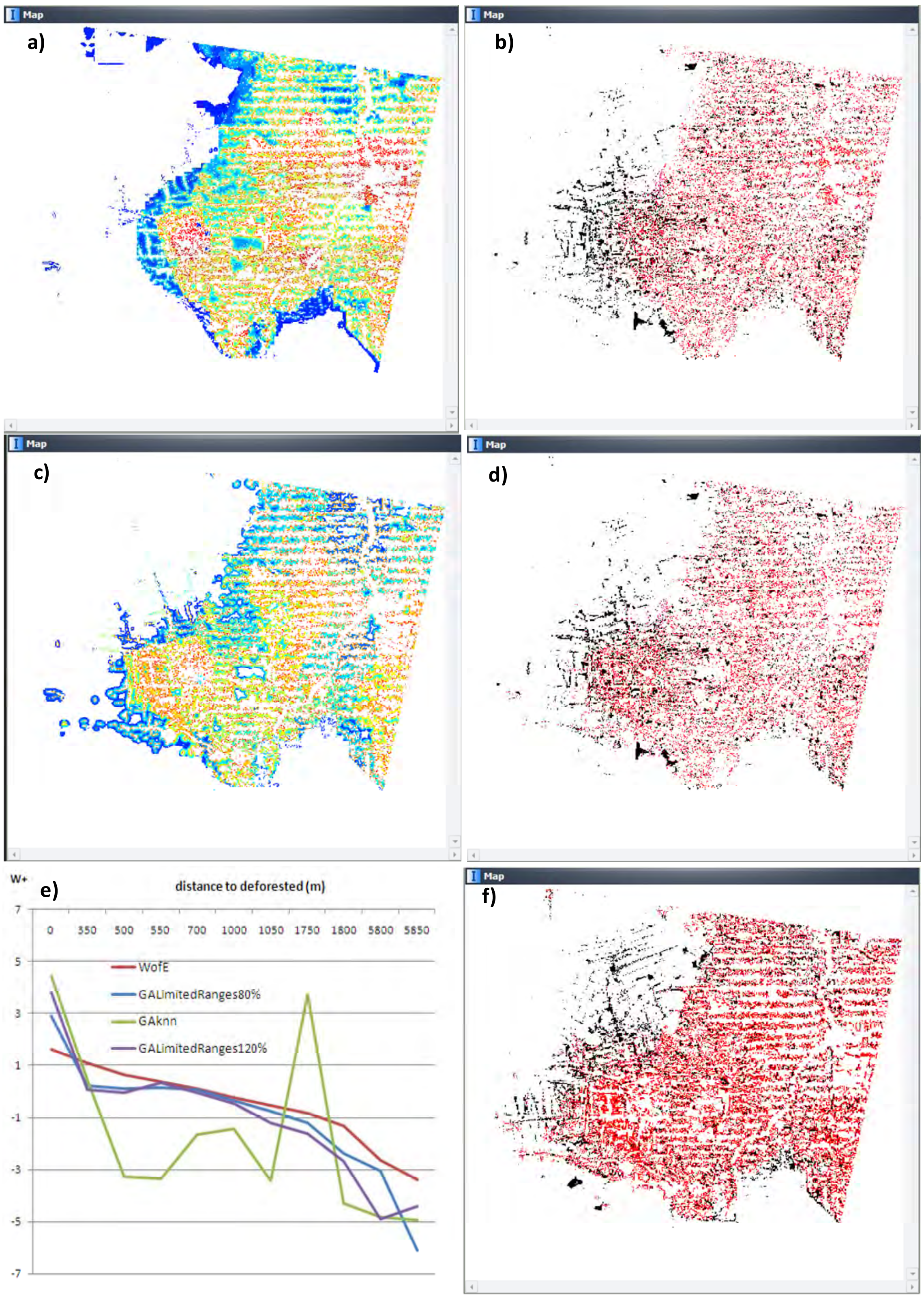


Fig. 23- a) Transition probability map from Weights of Evidence (blue: low probability, red: high probability), b) Simulated changes (red) by using map from a) over historical changes from 1997 to 2000 (black). c) Transition probability map using weights output from Genetic Algorithm tool limited within a 120% envelope. d) Simulated changes (red) using map from b) over historical changes from 1997 to 2000 (black). e) Weights of evidence of deforestation as a function of distance to previously deforested areas (WofE- Weights of Evidence method, GAknn - GA sole method, others: GA with values limited, respectively, within 80% and 120% envelope of original weights of evidence values, f) Simulated changes (red) using map from c) over historical changes from 2000 to 2003 (black).

Final remarks

Congratulations! You have completed this guidebook. What a haul! Other examples and texts on Dinamica EGO and its applications are available in the Examples folder. We hope that you have acquired the rationale for model designing with Dinamica EGO and can envision by now its enormous possibilities for environmental modeling. This is only the tip of the iceberg, now it is up to you to explore this wide avenue for the creative and ingenious design of environmental models.

Dinamica EGO is a non-commercial purpose freeware (see copyright below). Several research programs carried out by CSR/UFMG have supported its development. We are very grateful to all sponsors that directly or indirectly contributed to its state-of-art. See list sponsors at www.csr.ufmg.br/dinamica.

We are committed to supporting students and researchers interested in its use. For further inquiries, feel free to contact us - dinamica@csr.ufmg.br. CSR lab at UFMG has a program for visiting scholars.

13. References

Agterberg, F.P. and Bonham-Carter, G.F. Deriving weights of evidence from geoscience contour maps for the prediction of discrete events. *XXII Int. Symposium AP-COM*, 381-395. (1990).

Anselin, L. *Spatial Externalities, Spatial Multipliers and Spatial Econometrics*. (University of Illinois, Urbana-Champaign, 2002).

Bonham-Carter, G. *Geographic information systems for geoscientists: modeling with GIS*. (Pergamon, New York, 1994) 398 pp.

Burrough, P. A. *Principles of Geographical Information Systems for Land Resources Assessment*. (Clarendon Press, Oxford, 1986).

Cattaneo, A. *How to Distribute REDD Funds Across Countries? A Stock-Flow Mechanism*. Submission to the United Nations Framework Convention on Climate Change regarding AWG-LCA (FCCC/AWGLCA/2008/L.7) (2008).

Costa, M. H., Botta, A. and Cardille, J. A. Effects of large-scale changes in land cover on the discharge of the Tocantins river, Southeastern Amazonia. *Journal of Hydrology* **283**, 206-217 (2003).

Costanza, R. Model goodness of fit: a multiple resolution procedure. *Ecological Modelling*, **47**, 199-215 (1989).

Eastman, J. R., Van Fossen, M., Solorzano L. Transition potential modeling for land cover change in: Maguire, D., Batty, M, Goodchild, M., editors. GIS, spatial analysis, and modeling, ESRI Press ,Redlands, Calif. (2005).

FAS (Fundação Amazônia Sustentável). *The Juma Sustainable Development Reserve Project: Reducing Greenhouse Gas Emissions from Deforestation in the State of Amazonas, Brazil. For validation at Climate Community and Biodiversity alliance* (CCBA). [online] <http://www.climate-standards.org/pdf/release_juma_english_v_1_0_3.pdf> (2008).

Fearnside, P. M. Environmental services as a strategy for sustainable development in rural Amazonia. *Ecological Economics* **20** 53-70 (1997).

Geist, H. J. and Lambin, E. F. *What Drives Tropical Deforestation? A Meta-Analysis of Proximate and Underlying Causes of Deforestation Based on Subnational Case Study Evidence*. Belgium, LUCC International Project Office, LUCC Repo6rt Series, 4. 136pp. [online] <http://www.geo.ucl.ac.be/LUCC/lucc.htm> (2001).

Goodacre C. M., Bonham-Carter G. F., Agterberg, F. P. and Wright D. F. A statistical analysis of spatial association of seismicity with drainage patterns and magnetic anomalies in western Quebec. *Tectonophysics* **217**, 205-305 (1993).

Hagen, A. Fuzzy Set Approach to Assessing Similarity of Categorical Maps. *International Journal of Geographical Information Science*, **17**, 235-249 (2003).

Houghton, R.A. et al. Annual fluxes of carbon from deforestation and regrowth in the Brazilian Amazon. *Nature* **403**, 301-304 (2000).

Houghton, R. A., Lawrence, K. T., Hackler, J. and Brown, L. S. The spatial distribution of forest biomass in the Brazilian Amazon: a comparison of estimates. *Global Change Biology* **7**, 731-746 (2001).

Houghton, R.A. in *Tropical Deforestation and Climate Change*. (eds Moutinho and S. Schwartzman) (IPAM and ED, Belém, 2005).

Hirsch, A. I., Little, W. S., Houghton, R. A., Scott, N. A. and White, J. D. The net carbon flux due to deforestation and forest re-growth in the Brazilian Amazon: analysis using a process-based model. *Global Change Biology* **10,** 908-924 (2004).

Koza JR. Genetic Programming: on the programming of computer by means of natural selection - Complex Adaptive Systems. MIT press, 840 pp (1992).

INPE (Instituto Nacional de Pesquisas Espaciais). *Monitoramento da Floresta Amazônica Brasileira por Satélite - Projeto PRODES*. [online] <http://www.obt.inpe.br/prodes> (2007).

Intergraph Corporation. *Intergraph Microstation PC, Version 4 User's Guide.* Bentley Systems, Inc. and Intergraph Corporation, 333 pp (1991).

McGarigal, K. and Marks, B.J. FRAGSTATS: Spatial pattern analysis program for quantifying landscape structure. PNW-GTR-351. U.S. Department of Agriculture, Forest Service, Pacific Northwest Research Station, Portland. (1995).

Metzger, J.P. How to deal with non-obvious rules for biodiversity conservation in fragmented landscapes? *The Brazilian Journal of Nature Conservation* **4**, 125-139 (2006).

Moutinho, P. and S. Schwartzman. *Tropical Deforestation and Climate Change*. (IPAM and ED, Belém, 2005).

Merry, F., Soares-Filho, B.S., Nepstad, D., Amacher, G. and Rodrigues, H. Balancing conservation and economic sustainability: the future of the Amazon timber industry. *Environmental Management* (2009). doi: 10.1007/s00267-009-9337-1

Pontius, R.G.Jr. Statistical methods to partition effects of quantity and location during comparison of categorical maps at multiple resolutions. *Photogrammetric Engineering and Remote Sensing* **68**, 1041-1049 (2002).

Power, C., Simms, A. and White, R. Hierarchical fuzzy pattern matching for the regional comparison of Land Use Maps. *International Journal of Geographical Information Science* **15**, 77-100 (2001).

Saatchi, S. S., Houghton, R. A., Dos Santos Alvala, R. C., Soares, Z. J. V. and Yu, Y. Distribution of aboveground live biomass in the Amazon basin. *Global Change Biology* **13**, 816–837 (2007).

Schlamadinger, B., Johns, T., Ciccarese, L., Braun, M., Sato, A., Senyaz, A., Stephens, P., Takahashi, M. and Zhan, X. Options for including land use in a climate agreement post-2012: improving the Kyoto Protocol approach. *Environmental Science and Policy* **10**, 295-305 (2007).

Sampaio, G., Nobre, C., Costa, M. H., Satyamurty, P., Soares-Filho, B. S., Cardoso, M. Regional climate change over eastern Amazonia caused by pasture and soybean cropland expansion. *Geophysical Research Letters* **34**, 1-7 (2007). doi: 10.102

Santilli, M., Moutinho, P., Schwartzman, S., Nepstad, D. C., Curran, L., and Nobre, C. Tropical deforestation and the Kyoto Protocol: an editorial essay. *Climatic Change* **71**, 267-276 (2005).

Schneider, E. K., Fan, M; Kirtman, B. P. and Dirmeyer, P. Potential effects of Amazon deforestation on tropical climate. *Cola Technical Report* **226**, 1-41 (2006).

Sedjo, R.A., B. Sohngen. *Carbon credits for avoided deforestation*. (Resources for the Future, Washington, D.C, 2007).

Silva-Chavez, G. and Petsonk, A. Rainforest credits. *Carbon Finance* **6**, 18 (2006).

Skutsch, M., Bird, N., Trines, E., Dutschke, M., Frumhoff, P., de Jong, B. H. J., van Laake, P., Masera, O., and Murdiyarso, D. Clearing the way for reducing emissions from tropical deforestation. *Environmental Science and Policy* **10**, 322-334 (2007).

Soares-Filho, B. S., Pennachin, C. L., Cerqueira, G. DINAMICA – a stochastic cellular automata model designed to simulate the landscape dynamics in an Amazonian colonization frontier. *Ecological Modelling* **154**, 217-235 (2002).

Soares-Filho, B. S., Corradi, L., Cerqueira, Araújo, W. Simulating the spatial patterns of change through the use of the dinamica model. In *Simpósio Brasileiro de Sensoriamento Remoto*, **11**, 2003, BH, INPE, 721-728 (2003).

Soares-Filho, B. S., Alencar, A., Nepstad, D., Cerqueira, G., Vera-Diaz, M., Rivero, S., Solórzano, L. and Voll, E. Simulating the response of land-cover changes to road paving and governance along a major Amazon highway: the Santarém-Cuiabá corridor. *Global Change Biology* **10**, 745-764 (2004).

Soares-Filho, B. S., Nepstad, D, Curran, L.,Voll, E., Cerqueira, G., Garcia, R. A., Ramos, C. A., Mcdonald, A, Lefebvre, P. and Schlesinger, P. Modeling conservation in the Amazon basin. *Nature* **440**, 520-523 (2006).

Soares-Filho, B.S., Garcia, R. A., Rodrigues, H., Moro, S. and Nepstad, D. Nexos entre as Dimensões Socioeconômicas e o Desmatamento: A Caminho de um Modelo Integrado. In *Amazônia. Natureza e Sociedade em Transformação*. (eds Batistella, M., Alves, D. and Moran, E.) (Edusp, São Paulo, 2008).

Soares-Filho, B.S., Moutinho, P, Nepstad, D, Anderson, A., Rodrigues, H. Garcia, R. Dietzch, L., Merry F., Bowmna, M., Hissa, L., Silvestrini, R., Maretti, C. Role of Brazilian Amazon protected areas in climate change mitigation. Proc Natl. Acad. Sci. 2010. www.pnas.org/cgi/doi/10.1073/pnas.0913048107.

Teixeira, A. M., Soares-Filho, B.S., Freitas, S. and Metzger, J.P.W. Modeling landscape dynamics in the Atlantic rainforest domain: Implications for conservation. *Forest Ecology and Management* **257**, 1219–1230 (2009).

14. List of functors

Dinamica EGO has been applied to numerous environmental studies, including the modeling of deforestation in the Amazon from local to basin-wide scales, land-use and cover change in the Atlantic Forest and in the dry tropical forest of Mexico, urban dynamics, logging in the Amazon, forest fire risk and spreading, analysis of opportunity costs of forgone forestry and agricultural activities for forest conservation, the role of protected areas in reducing carbon emissions in the Brazilian Amazon, the co-benefits of REDD in Xingu headwaters, and a proposal to end deforestation in the Brazilian Amazon. The aim of this guidebook is to introduce the user to the vast possibilities of Dinamica EGO for the creative design of models that can truly represent the complexity of geographic phenomena.

ISBN 978-85-910119-0-2

9 788591 011902