



HAL
open science

Conceptually correct and algorithmically efficient pruning and allocation procedures in statistical LUCC modelling

François-Rémi Mazy

► **To cite this version:**

François-Rémi Mazy. Conceptually correct and algorithmically efficient pruning and allocation procedures in statistical LUCC modelling. Applications [stat.AP]. 2019. hal-02302133

HAL Id: hal-02302133

<https://inria.hal.science/hal-02302133>

Submitted on 1 Oct 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ GRENOBLE ALPES
MASTER 2 STATISTIQUES ET SCIENCE DES DONNÉES

RAPPORT DE STAGE

Conceptually correct and algorithmically efficient pruning and allocation procedures in statistical LUCC modelling

du 1^{er} mars au 31 juillet 2019
Équipe STEEP – Inria Grenoble

Élève :
François-Rémi Mazy

Encadrants :
Pierre-Yves Longaretti (Inria)
Rémy Drouilhet (UGA)

August 26, 2019

Contents

Contexte du travail effectué	2
1 Introduction	3
1.1 Generalities	3
1.2 Spatially explicit models architecture	3
1.3 Models limits and work introduction	4
2 Notation and definitions	6
2.1 Pixel characterization parameters	6
2.2 Justification elements of the probabilistic approach in the LUCC modeling	7
3 A case study: the SCoT of Grenoble	9
4 Calibration through explanatory variables	10
4.1 Probabilities in the limit of statistical independence of explanatory variables	10
4.2 Dinamica weights of evidence	12
4.3 Imperfect closure	12
5 Pruning and currently used allocation methods	13
5.1 Position of the problem	13
5.2 Summary of Dinamica pruning and allocation strategies involving patches	13
5.3 Defining conceptually correct pruning	15
6 New land use change allocation algorithm	17
6.1 Calibration and allocation scenario	17
6.2 Generalized acceptance rejection test	17
6.3 Allocation algorithm	18
7 Islands and expansions allocation procedures	21
7.1 Calibration	21
7.2 Island design process	22
7.3 Expansion design process	23
7.4 Patch design implementation within the allocation algorithm	24
8 Conclusion	26

Contexte du travail effectué

J'ai pris contact en avril 2018 avec l'équipe STEEP du centre Inria de Grenoble et nous avons rapidement monté un projet de doctorat portant sur la modélisations du changement d'usage des sols. Nous nous sommes alors donné un an pour trouver un financement de thèse et dans ce laps de temps il a été convenu que je suive un master 2 de statistiques afin de compléter ma formation initiale d'ingénieur dans ce domaine. Le présent document rapporte mon travail effectué dans le cadre du stage de master 2 Statistiques et Sciences des Données (M2 SSD) de l'Université Grenoble Alpes réalisé de mars à juillet 2019. Nous avons finalement obtenu en juin 2019 une bourse de thèse entièrement financée par l'Inria.

Ce stage constitue donc une entrée en matière à ce travail de thèse qui s'effectuera sur les trois prochaines années. Un travail de bibliographie ayant déjà été réalisé en amont par mon directeur de thèse Pierre-Yves Longaretti, j'ai pu me confronter pendant quatre mois directement aux premiers problèmes soulevés par le sujet de thèse. Ce travail m'a ainsi permis de me familiariser avec les modèles de changement d'usage des sols et ainsi confirmer mon attrait pour ce sujet de recherche. Il s'agissait également de prendre de l'avance, si cela est concevable, sur le doctorat et la première publication.

Ce contexte de stage particulier m'a donc invité à travailler de manière à pouvoir ré-exploiter les éléments traités dans la thèse future. Ainsi, l'ensemble de ce rapport est rédigé en anglais. De plus, la majorité des chapitres relèvent de la théorie des modèles de changements d'usage des sols afin d'être au plus proche de la structure d'un article scientifique. L'implémentation de ces méthodes et en particulier du nouvel algorithme d'allocation n'est décrite que dans le chapitre 6.

Ce document constitue ainsi pour moi un point d'étape dans le travail effectué, sans ambition de résultat et qui se trouve être un véritable marche pied avant la thèse qui s'annonce.

1 Introduction

1.1 Generalities

Land use and cover change (LUCC) models are mainly used in environmental sciences. The term land cover refers to the cover kind such as urban or plant as it can be characterized by an air or satellite photo; land use refers to their management.

LUCC models are used on demarcated areas from landscapes of some square kilometers to whole continental surface as Europe. Quantify environmental impact due to public policies among others parameters is one of the main model purpose considering constraints imposed by global changes, especially climate change. These models are generally employed in research studies but can also advice public decision.

In this two contexts, a way to apprehend the interface between environment and society is the concept of ecosystem service by highlighting the whole of services and functions fulfilled by ecosystems for society benefit. But land use and cover changes can not fully determine kind and quality services change. It is then necessary to adjoin to LUCC models for all ecosystem of interest a model in order to evaluate the link between land use and cover change and ecosystem services on a same territory. Services modeling is a complex subject which is not approached in this report.

A wide variety of models are described in the literature: dynamic or static, agent or grid based, global or local *etc.* (Verburg et al., 2006), on diverse levels of spatio-temporal and decisional complexity (Agarwal et al., 2002). We focus here on spatially explicit models which present robustness and development maturity. They produce future state maps of the territory for discrete time steps, typically for intervals of some years; for example, in the project ESNET (Ecosystem Services NETWORKS) which constitutes the case study of this report, the time step is fixed to 5 years between 2010 and 2040.

The studied territory is divided in a large amount of elements – pixels in an arbitrary case or parcels if they correspond to homogeneous real units of the landscape – small enough to be characterized by a single cover use (a typically size is about a few dozen meters). A preliminary step of any LUCC project consists then to define a list of all possible cover uses in coherence with the study purpose.

The maps produced by these models and the information that we derive from them are one of the main supports for scientific analysis of changes on the one hand, and discussion with elected officials and decision-making bodies of the other. Time horizon of these models is limited to a few decades in the future due to increasing imprecisions of temporal projections.

Among the best-known and most widely-developed spatially explicit models are the CLUE model family (Verburg et al., 2002; Verburg and Overmars, 2009), the Dinamica EGO model (Soares-Filho et al., 2002), and the LCM model (Land Change Modeler). (Eastman 2012). These are in fact software environments allowing the user to create a specific model for a given study area and problem, from elementary modules, the most important of which are described below.

1.2 Spatially explicit models architecture

LUCC models analyze statistically cover change based on three main modules (fig. 1).

The first one (probabilities module) determines probabilities of pixel state change between different possible land use/land cover types for one time step. This module produces maps of change probabilities. For example, an agricultural pixel, very close to urban areas has an higher probability as a mountain pixel to become an urban pixel at the next time step. All areas

constraints (especially protected areas for environmental reasons or natural risk management) are introduced in a spatially explicit way at this stage. Change probabilities are calibrated on past evolution from the difference between two land use and cover maps at two different dates. These changes are then statistically characterized thanks to several explanatory variables, whose choice is left to the user. For example, in the case of urban sprawl, we can consider as explanatory variable the distance to the main urban centers, real estate market prices, the local terrain slope, *etc.* Evolution scenarios, which refer to global land use and cover changes, are defined thanks to either a specific model, or an expert, or a planning organization for example. Whenever possible, models are validated with a third land use and cover map and validation methods are described in literature (Pontius 2002, Hagen 2003). However, specific methods can be used for each particular problem.

The second module (demand module) establishes global changes (for example in hectare by years) on the whole territory between the different land use for one time step. These surfaces of changes can be deduced from past evolutions with simple tendencies extrapolation or be based on evolution scenarios of the territory. The more relevant changes relate to changes between different type of agricultural crops, exchanges between forest areas and agricultural areas (especially due to agricultural abandonment), and urbanization of agricultural land.

The last module (allocation module) affects explicitly land use and cover change by time step based on informations provided by the first two modules and on an initial land use map. An algorithm selects pixels whose cover use will change during the time step. A new land use map is produced and the process can be repeated until the whole projection period has been simulated. Even if change probabilities are defined for each pixels, changes are allocated as patches has observed in reality.

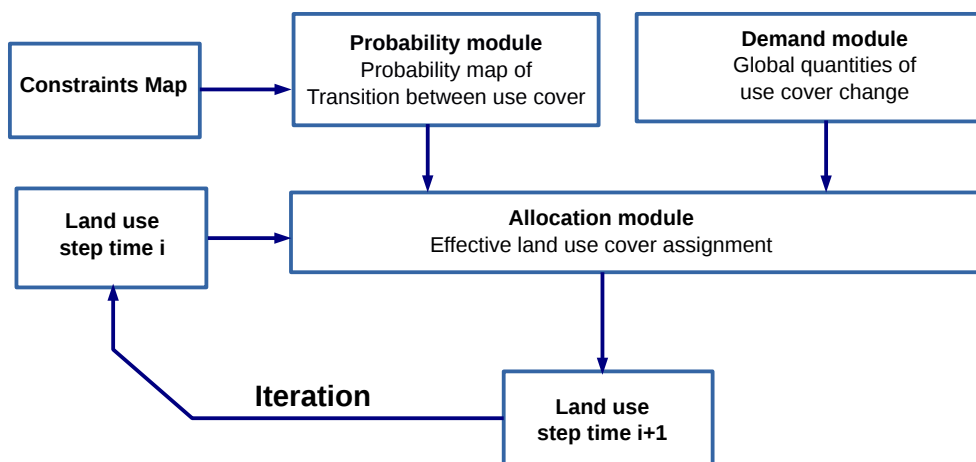


Figure 1: Schematic diagram of LUCC models architecture.

1.3 Models limits and work introduction

It should be noted beforehand that the statistical nature of the model implies that the maps obtained represent a possible state of the territory, but certainly not a prediction; the quantitative information from these models is only a statistical value. In addition, models use various methods to establish statistical correlation between land use and cover change and explanatory variables. Moreover, the final pixel state allocation can be assigned in different ways. These remarks can lead to noteworthy model behavior differences for a same problem (Mas et al., 2011, 2014). This point is essential to consider for any decision support application and insofar as possible, the

modeler should evaluate the robustness of the conclusions for the decision-maker.

Also, spatially explicit models are frequently cited and used in the literature but their description is incorrect. In particular, the theoretical background underlying such models is deficient, especially in comparison with the level of sophistication reached by the various modeling environments mentioned above. Some modeling principles are used intuitively and pragmatically without adequate justification; some of these are plainly wrong once formulated in an appropriate probability theory language. Indeed, no theoretical tools are available for neither software users nor developers of other modeling environments even though those objects are essential to evaluate these models in a critical fashion. Important differences in performance can be noticed on specific problems but without any explication about the origin of quantitative discrepancies – even qualitative on occasion. These deficiencies impede systematic improvements of modeling performances.

In this work which constitutes the first step in a more extensive LUCC theoretical analysis, we focused on a general approach. Introducing the notations used in this report, an explicit theoretical framework is introduced to define probabilities of change of land use/land cover. This framework is strictly correct only in the limiting case of statistically independent explanatory variables; this hypothesis is never satisfied in practice, but statistical correlations are minimized in practice by an appropriate choice of these variables. In any case, it provides a useful framework to analyze the origin of the problems of existing allocation procedures. I focused on what apparently constitutes the most important limitation in LUCC modeling, i.e., the allocation procedure itself. A new conceptually correct allocation method is devised and presented. It is then applied in a actual problem. To this effect, I have developed a Python module called Demeter.

2 Notation and definitions

2.1 Pixel characterization parameters

We consider a study area for which we have raster data at a same scale for two or three different dates. They are used for the calibration and the validation procedure of the land use and cover change model.

Several quantities characterize a given pixel:

1. Each pixel is identified by its index j . $\mathbb{J} = \llbracket 1, j^m \rrbracket$ is the set of all indexes.
2. Pixel coordinates are given by the couple (x, y) . The coordinates of the pixel j are therefore (x^j, y^j) . There is of course a bijection between the set \mathbb{J} and the set of used coordinate couples.
3. The symbol v refers to land use and cover types. A natural number is affected to each type and therefore $v \in \llbracket 0, v^m \rrbracket$ where 0 refers to a lack of data and only $\llbracket 1, v^m \rrbracket$ are meaningful. We use also the term state to indicate a land use and cover type. The model calibration is based on land use and cover at two different dates. v_i and v_f refers respectively to a pixel state at the initial moment and the final one. For simplicity, these subscripts are also intuitively used in the allocation phase: v_i refers to the pixel state at the current step time and v_f is the allocation outcome state. Subsets of \mathbb{J} can be defined regarding the land use state. For example, $\mathbb{J}_{v_i} = \{j \in \mathbb{J}, v^j = v_i\}$. In the following, the sequential aspect of land use states is implicit, *i.e.* the subscript v_i, v_f refers to an element which have transited from v_i to v_f during the studied phase. For example, \mathbb{J}_{v_i, v_f} is the set of pixel which have transited from v_i to v_f .
4. Z denotes the set of explanatory variables which are considered as relevant explanations of states and each element has an index: $Z = \{Z_1, \dots, Z_{k^m}\}$. The explanatory variables are defined for a specific initial state v_i . Thus, we consider that all elements relative to Z take this initial state as prerequisite and we omit to mention it in the notation. The value of the explanatory variable Z_k for the pixel j is written z_k^j . It can be both part of \mathbb{R} (for a continuous variable) or \mathbb{N} (for a discrete or qualitative variable). Continuous variables are discretized in order to handle all variables in the same way while taking into account the limited volume of some continuous variables samples. The discretization bins are written α_k^q with $q \in \llbracket 1, q^m \rrbracket$ (q^m can be different for each Z_k – the subscript k is omitted) such as:

$$\begin{aligned} \alpha_k^1 &= \min_{j \in \mathbb{J}_{v_i}}(z_k^j) \\ \alpha_k^{q^m} &= \max_{j \in \mathbb{J}_{v_i}}(z_k^j) \end{aligned} \tag{1}$$

The other elements of α_k^q can be freely determined based on the selected discretization procedure used (chapter ?? *Frem: faudra regarder ça à un moment...*).

For each pixel j and each explanatory variable Z_k , the discrete value of \hat{z}_k^j is therefore¹:

¹ $z_k^j < \alpha_k^1$ and $z_k^j > \alpha_k^{q^m}$ cases are only conceivable in the allocation phase where \mathbb{J}_{v_i} is not the same as the set used for the discretization setting.

$$\hat{z}_k^j = \begin{cases} 0 & \text{if } z_k^j < \alpha_k^1 \\ q \mid \alpha_k^q \leq z_k^j < \alpha_k^{q+1} & \text{if } z_k^j < \alpha_k^{q^m} \\ q^m - 1 & \text{if } z_k^j = \alpha_k^{q^m} \\ q^m & \text{if } z_k^j > \alpha_k^{q^m} \end{cases} \quad (2)$$

Some notations are follows:

- \hat{z}^j is the n-tuple of all discretized explanatory values for the pixel j , *i.e.* $\{\hat{z}_1^j, \hat{z}_2^j, \dots, \hat{z}_{k^m}^j\}$.
- \hat{z} is any n-tuple of all discretized explanatory values, including cases which are not represented in the studied case.
- $\hat{\mathbb{Z}}$ is the set of all possible \hat{z} .
- $\hat{\mathbb{Z}}^*$ is a subset of \mathbb{Z} which includes only n-tuples \hat{z} which are really represented in the case, *i.e.* $\hat{\mathbb{Z}}^* = \{\hat{z}^j, j \in \mathbb{J}_{v_i}\}$.

The main parameters for pixel characterization are summarized in table 1.

symbol	set	definition
j	$\mathbb{J} = \llbracket 1, j^m \rrbracket$	pixel unique index
(x, y)	\mathbb{R}^2	pixel coordinates
v	$\mathbb{V} = \llbracket 0, v^m \rrbracket$	land use state; $v = 0$ refers to a lack of data and only $v = 1 \dots v^m$ are meaningful
v^j	\mathbb{V}	land use state of the pixel j
$v_i \rightarrow v_f$	$\mathbb{V} \rightarrow \mathbb{V}$	land use transition from an initial to a final state
Z_k	\mathbb{Z}	explanatory variable k
z_k^j	\mathbb{R}, \mathbb{N}	value of the explanatory variable Z_k for the pixel j
α_k^v	$\mathbb{R}^{q_k^m+1}, \mathbb{N}^{q_k^m+1}$	bins returned by the Z_k discretization
q_k	$\hat{\mathbb{Z}}_k = \llbracket 0, q_k^m \rrbracket$	bin index as defined in (1)
\hat{z}_k^j	$\hat{\mathbb{Z}}_k$	discretized value of the explanatory variable Z_k as defined in (2), for the pixel j
\hat{z}	$\hat{\mathbb{Z}}$	any tuple of discretized values of all explanatory variables (may not occur in the calibration phase)
\hat{z}^j	$\hat{\mathbb{Z}}^*$	tuple of discretized values of all explanatory variables for the pixel j

Table 1: Notations and definitions

\mathbb{J} can be divided in subsets according to conditions specified with appropriate subscripts. For example, $\mathbb{J}_v = \{j \in \mathbb{J} \mid v^j = v\}$. The cardinal function is written $\#$ and consequently, $\#\mathbb{J}$ is the cardinal number of \mathbb{J} . For example, $\#\mathbb{J}_{v_i, v_f, \hat{z}}$ is equal to the number of pixels such as their initial state is v_i , their final state is v_f and their discrete explanatory values are equal to \hat{z} .

2.2 Justification elements of the probabilistic approach in the LUCC modeling

Two distributions of probability are mainly required in spatially explicit LUCC models:

- $P(v_f | v_i, \hat{z})$ describes the distribution of final states knowing the initial state and the explanatory variables.

- $P(\hat{z}|v_i, v_f)$ describes distribution of explanatory variables knowing initial and final states.

By drawing pixels, one get a simple but not exclusive way to introduce the probabilities presented above as random variables. As a draw defines simultaneously a value for all variables, on can define in this context the joint probability: $P(V_i, V_f, \hat{Z})$ which defines the probability for a given pixel to transit from v_i to v_f during the time step. Also, for each step of the simulation, the final state is unknown and as for any problem with complex causes, one prefers handle the result of the evolution as a random process product.

It is essential to understand the difference between the two probabilities introduced above. The first one is applied to any given pixel during the simulation. It gives the probability for that pixel with the initial state v_i and knowing its explanatory variables values to change to v_f . The second one corresponds to pixels repartitions function of explanatory variables knowing initial and final states. It cannot be the probability for a given pixel to present a specified \hat{z} , because each pixel has already a fixed \hat{z} which does not change during the time step. During the calibration stage, one can realize a fair sampling within the subset \mathbb{J}_{v_i, v_f} and get on average a distribution in compliance with $P(\hat{z}|v_i, v_f)$. Thus, one use for the calibration the Bayes formula in the frequency sense. However, during a simulation, it is *a priori* a probability in the Bayesian sens which defines the distribution function of \hat{z} , knowing v_i and v_f . To simulate a trend evolution based on the calibration, the distribution is used as it is. Alternately, on can edit this probability during the simulation according to a specified scenario. This Bayesian aspect is used to relate the two probabilities in section 4.1. Also, only $P(v_f|v_i, \hat{z})$ makes sense to generate a map of probability during a simulation.

3 A case study: the SCoT of Grenoble

The ESNET project aims to analyse future land use trajectories and their effects on biodiversity and ecosystem services for the Grenoble urban area. As part of this project, a case study dedicated to LUCC had been defined and I use it as case study of this work.

The employment area of Grenoble covers 4 450 km². In 2012 according to INSEE, there were 800 000 inhabitants and 500 000 jobs. The boundaries of this territory have been defined regarding the economic influence of the agglomeration of Grenoble and the diversity of landscapes which outline the region. The economic aspect, boundaries are based on EPCI² areas. Selected EPCIs belong to the SCoT³ of Grenoble and some surrounding EPCIs. On the landscape aspect, the region of Grenoble presents a large diversity of natural sites. Planar valleys of Grenoble and the Grésivaudan encourage urban extension. The three mountain ranges (the Vercors, the Chartreuse and Belledune) structure the territory and provide natural landscape with numerous protection areas (regional nature parks, nature reserve).

Thus, the employment area gathers 311 municipalities in a radius of about fifty kilometers organized in ten EPCIs: the agglomeration of Grenoble, the south of Grenoble, the Grésivaudan and around Voiron (these areas form the “Y” of Grenoble), the Chartreuse and the Vercors (which are the mountain range areas apart from the SCoT of Grenoble), the Trièves, the Matheysine, the south of the Grésivaudan, the Bièvre Valloire (which are mainly agricultural plains).

As part of the ESNET project, existing spatial data about this territory have been merged with photo interpretation and IGN maps in order to provide a data base of land use maps of 1998, 2003 and 2009 with a precision of 15 meters (the pixel width). The topology of this data base is composed by 34 states ranked in four precision levels (fig. 2). It distinguishes artificial areas (with within it residential zones, zones of industrial and commercial activity, roads, railway networks. . .). It describes also several agricultural types (with within it monocultures, pastures, market gardening. . .). Forest and semi-natural⁴ environments are characterized at the population level for wooded areas.

In this work, I focused on transitions from agriculture to urban residential area and zone of industrial and commercial activity. The land use maps used are then only constituted by 8 states as described in table 2.









id	state	color	
0	null data	black	
1	aquatic	blue	
2	residential	red	
3	agricultural	orange	
4	forest	green	
5	semi-natural	brown	
6	road	yellow	
7	industrial and commercial activity	purple	

Table 2: Selected land use states

²EPCI (*Établissement Public de Coopération Intercommunale*) are administrative structures gathering several municipalities.

³The SCoT (*Schéma de Cohérence Territoriale*) is a french urban plan determining on the scale of several municipalities in order to put in coherence all local policies including accommodations, transports, commercial areas, environment and landscape.

⁴Semi-natural areas present unfavorable lands for agriculture or are inaccessible (especially in mountain range).

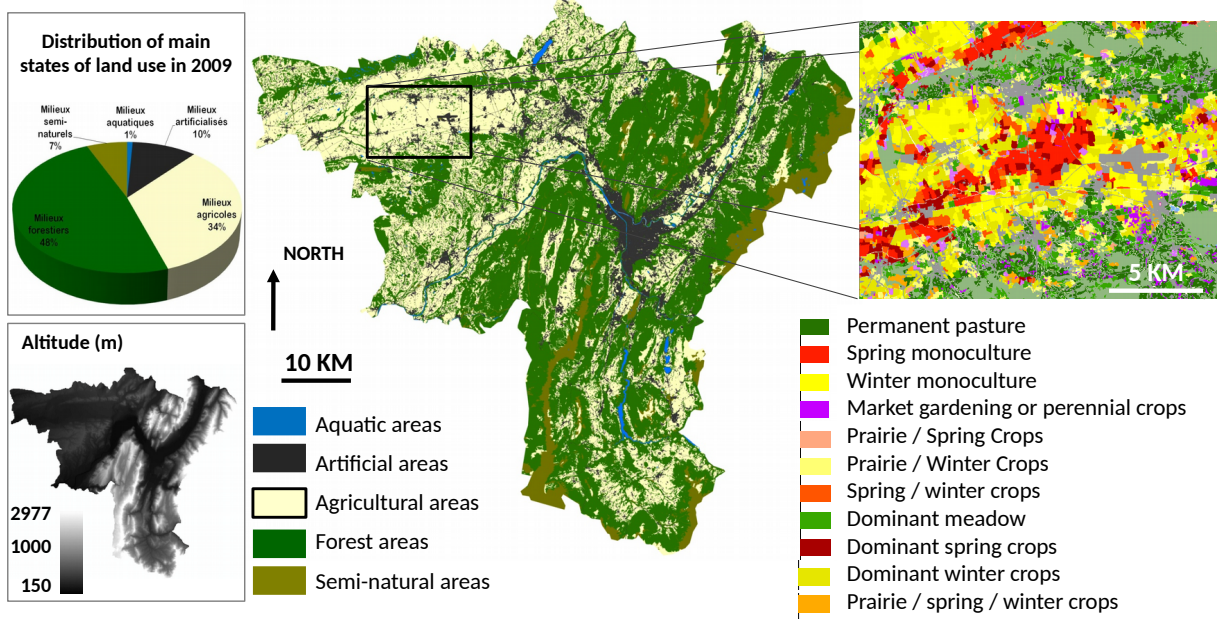


Figure 2: Land use states within the employment area of Grenoble in 2009.

4 Calibration through explanatory variables

4.1 Probabilities in the limit of statistical independence of explanatory variables

Based on land use and cover maps at two distinct dates, the calibration phase aims at determining transition probabilities for each pixels according to their initial state and explanatory variable values.

This probability can be written:

$$P(v_f | v_i, \hat{z}) = \frac{P(v_f, v_i, \hat{z})}{P(v_i, \hat{z})}, \quad (3)$$

where both factors are based on the calibration stage. Alternately, one can rewrite this probability using Bayes formula:

$$P(v_f | \hat{z}, v_i) = P(v_f | v_i) \times \frac{P(\hat{z} | v_i, v_f)}{P(\hat{z} | v_i)}. \quad (4)$$

This second formula is a useful problem formulation which isolates an element with no spatial dependence, $P(v_f | v_i)$. This probability will then be considered as a scenario parameter during the allocation phase.

L'indépendance statistique est une hypothèse; j'ai donné rapidement plus haut dans un des commentaires une justification de l'usage de cette hypothèse. On fait cette hypothèse parce que ça permet de formuler les choses analytiquement de façon assez simple, et donc d'identifier les problèmes.

To apply this Bayesian statistics approach and for simplicity, explanatory variables are considered as independent random variables. This hypothesis is never satisfied but explanatory variables \mathbb{Z} are chosen in order to approach it. It is often sufficient considering the precision level aimed for LUCC modeling. It is possible to conceive more sophisticated statistic methods for calibration with dependent variables but keeping this hypothesis is very useful to pinpoint the various problems underlying incorrect (implicit or explicit) assumptions in existing modeling environments. With this assumption:

$$\tilde{P}(\hat{z}|v_i, v_f) = \prod_k P_k(\hat{z}_k|v_i, v_f) \quad (5)$$

$$\tilde{P}(\hat{z}|v_i) = \prod_k P_k(\hat{z}_k|v_i) \quad (6)$$

Where P_k only considers the explanatory variable Z_k and the symbol \tilde{P} indicates that the independent random variables hypothesis is used.

In practice, this hypothesis makes the transition probability easier to calibrate. Indeed, one needs a smaller amount of calibration data to calibrate a one variable probability at a given precision level, compared to multiple variable probabilities. In fact, the number of calibration pixels is usually limited due to the relatively small number of pixels undergoing a transition of a given type in the calibration data. Indeed, without equations (5 - 6), calibrating would mean to evaluate every distinct combinations of \hat{z} . In the present case, it just implies to consider all bins of each explanatory variable separately. Also, this formulation allows to evaluate probabilities for some (v_i, \hat{z}) combinations not occurring in the calibration data. Such combinations are not necessarily unrealistic; they are only not represented due to the inherent sampling noise of the calibration data.

Thereby:

$$\tilde{P}(v_f|\hat{z}, v_i) = P(v_f|v_i) \times \prod_k \frac{P_k(\hat{z}_k|v_i, v_f)}{P_k(\hat{z}_k|v_i)} \quad (7)$$

Basically, probabilities are estimated by pixels counting:

$$P(v_f|v_i) = \frac{\#\mathbb{J}_{v_i, v_f}}{\#\mathbb{J}_{v_i}} \quad (8)$$

$$P_k(\hat{z}_k|v_i, v_f) = \frac{\#\mathbb{J}_{v_i, v_f, \hat{z}_k}}{\#\mathbb{J}_{v_i, v_f}} \quad (9)$$

$$P_k(\hat{z}_k|v_i, v_f) = \frac{\#\mathbb{J}_{v_i, \hat{z}_k}}{\#\mathbb{J}_{v_i}} \quad (10)$$

$\#\mathbb{J}_x$ are random variables. As we got only one “statistical experience” which corresponds to the observed reality, the noise question on probabilities estimation arises. Dinamica proposes a computing method by weights of evidence described in section 4.2. Note that the calibration phase could be here achieved and come down to know for all k $P_k(\hat{z}_k|v_i, v_f)$ and $P_k(\hat{z}_k|v_i)$. $P(v_f|v_i)$ is determined by the scenario. Of course, the obtained probabilities verify the closure relations:

$$\sum_{v_f \in \mathbb{V}} P(v_f|v_i) = 1 \quad (11)$$

$$\sum_{\hat{z} \in \hat{\mathbb{Z}}} P(\hat{z}|v_i) = \sum_{\hat{z} \in \hat{\mathbb{Z}}} \prod_k P_k(\hat{z}_k|v_i) = 1 \quad (12)$$

$$\forall v_f \in \mathbb{V}, \sum_{\hat{z} \in \hat{\mathbb{Z}}} P(\hat{z}|v_i, v_f) = \sum_{\hat{z} \in \hat{\mathbb{Z}}} \prod_k P_k(\hat{z}_k|v_i, v_f) = 1 \quad (13)$$

$$\forall \hat{z} \in \hat{\mathbb{Z}} \sum_{v_f \in \mathbb{V}} P(v_f|v_i, \hat{z}) = 1 \quad (14)$$

4.2 Dinamica weights of evidence

The weights of evidence is a useful and generic method to characterize transition probability when explanatory variables are independent. However, the approximation used in Dinamica to compute weights of evidence is particular and it proves to be constrained by a necessary condition. Dinamica makes this choice in order to reduce statistic noise due to the restricted sampling. One presents here only practical formulas.

Dinamica defines the weights of evidence W_k^+ in the following way:

$$\forall \hat{z}_k, \quad W^+(\hat{z}_k|v_i, v_f) = \ln \left(\frac{P_k(\hat{z}_k|v_i, v_f)}{P_k(\hat{z}_k|v_i)} \right) \quad (15)$$

Thus:

$$\tilde{P}(v_f|v_i, \hat{z}) = P(v_f|v_i) \prod_k \exp(W^+(\hat{z}_k|v_i, v_f)) \quad (16)$$

Dinamica only keeps track of the weights of evidence at the end of its calibration procedure.

4.3 Imperfect closure

$\hat{\mathbb{Z}}^*$ is a subset of all possible \hat{z} which includes only \hat{z} which are really represented in the calibration case. Then, we can write the inequation for all v_f regarding the closure relations:

$$\sum_{\hat{z} \in \hat{\mathbb{Z}}^*} \tilde{P}(\hat{z}|v_i, v_f) \leq \sum_{\hat{z} \in \hat{\mathbb{Z}}} \tilde{P}(\hat{z}|v_i, v_f) = 1. \quad (17)$$

In practice, it means that $\sum_{\hat{z} \in \hat{\mathbb{Z}}^*} \tilde{P}(\hat{z}|v_i, v_f)$ is lower than 1 because $\hat{\mathbb{Z}}^*$ is a subset of $\hat{\mathbb{Z}}$, *i.e.* all \hat{z} are not considered within the calibration phase. This also implies that for a given \hat{z} , $\sum_{v_f \in \mathbb{V}} P(v_f|v_i, \hat{z})$ is also lower than 1.

Missing (v_i, \hat{z}) combinations in the calibration data are not the only reason of imperfect closure. The lack of occurring cases is not the only origin of the closure relation rejection. Indeed, we have assumed the explanatory variables independence hypothesis. Yet, an error on this assumption implies a wrong evaluation of $\tilde{P}(\hat{z}|v_i, v_f)$ and leads also to an error on the closure relation.

5 Pruning and currently used allocation methods

We present in this section the method used by Dinamica to perform the allocation phase. This description is required to propose then a new allocation method in section 6.

5.1 Position of the problem

In principle, and without making any specific assumption, the allocation process can be viewed as a two-step process at any time step p in a simulation. First one randomly selects a pixel in \mathbb{J} . This random selection results in a known v_p^j, \hat{z}^j for the selected pixel. Second, one allocates to this pixel a final state, v_{p+1}^j .

This being said, without further relying on some form of a priori knowledge, this procedure, although theoretically correct, is extremely inefficient; inefficient here means that most random draws of pixels are performed uselessly as most pixels will not change state, so that most of the computational time will be spent in checking that no transition occurs. All software suffer from this problem; in order to minimize it, a preselection stage, called pruning, is introduced and two different modes of pruning are used in Dinamica and LCM.

We can now specify the various stages of the allocation procedure that should actually be implemented for efficiency purposes:

1. Pruning (preselection) of an ensemble of appropriate pixels for transitions from v_p^j to other states v_{p+1}^j .
2. Random selection of a pixel in this pruned ensemble.
3. Allocation of a final state to this pixel.

This being said, it is unclear which procedure to choose in the pruning and allocation steps, the only clear step being the random selection of a pixel once pruning is done. The present section is devoted to clarifying this issue. First, the procedures used in LCM and Dinamica are recalled. Then, the conceptually correct procedure is constructed. This will show that neither Dinamica nor LCM are correct in their choices.

5.2 Summary of Dinamica pruning and allocation strategies involving patches

Dinamica ranks pixels by decreasing value of $\sum_{v_f \neq v_i} P(v_f|v_i, \hat{z})$ and keeps c times the number of pixels needed for the transition. c is specified by the user ; the default is $c = 10$. By this way, it assures that selected pixels are the most likely to transit from v_i to another state $v_f \neq v_i$. The probability value $P(v_f|v_i, \hat{z})$ is directly determined by the weights of evidence calibration and a scenario which provides $P(v_f|v_i)$ (equation 16). Then, Dinamica randomly draws a pixel inside the pruning ensemble and test the various transitions with $P(v_f|v_i, \hat{z})$. If the test is accepted, *i.e.* a transition is performed to $v_f \neq v_i$, the pixel is allocated to v_f and is removed from the pruned ensemble (figure 3). This test is looped on a huge number of drawn pixels until all the required volumes of allocation have been performed. The chosen pruning method is questionable. Indeed, $P(v_f|v_i, \hat{z})$ does not perform a closure relation on \hat{z} , *i.e.* $\sum_{\hat{z}} P(v_f|v_i, \hat{z}) \neq 1$. For Dinamica it is just a convenient method to select pixels which are likely to transit without making too much mistake. We introduce in the following section a pruning process based on $P(\hat{z}|v_i, v_f)$. Note that as described in section 4.2, Dinamica only keeps track of the weights of evidence at the end of its calibration procedure. Thereby, $P(\hat{z}|v_i, v_f)$ is inaccessible to Dinamica in the state when the weights of evidence method is applied.

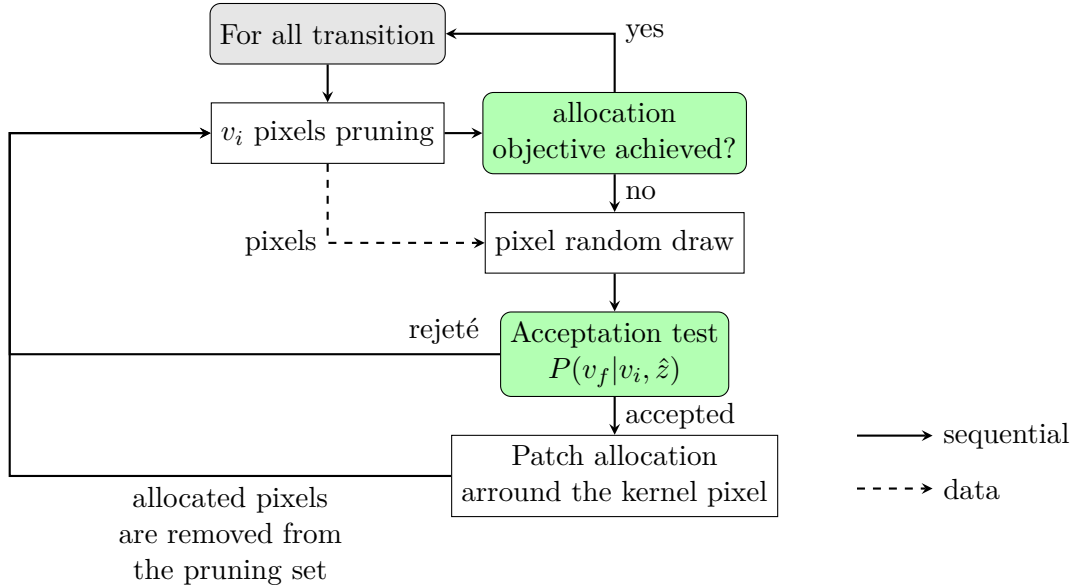


Figure 3: Schematic Dinamica allocation procedure.

The tension between the basic units of map characterization – pixels – and the basic units of LUCC change – patches according ecological observations, *i.e.* connected (often simply connected in the mathematical sense) ensembles of pixels – is ubiquitous in LUCC modeling, but does not seem to have been discussed or resolved in the literature.

All LUCC models only employ distribution functions for a single pixel. For example, an explanatory variable such as the distance to road describes the probability for a drawn pixel to be at a given distance from the road. Yet, a patch is defined as a manifestation of a correlation between adjacent pixels. It implies that the probability that a pixel changes state is much greater if a nearby pixel also changes state. State changes are not independent between nearby pixels and probabilities for several pixels are not simply the product of single pixel probabilities.

In these conditions, the single pixel function (the usual probability in models) does not contain the whole information required to model cover change and it would make more sense to develop n-pixels probabilities. Distribution functions implying several objects are problematic because it is in practice very difficult to obtain a reliable theory when the correlations between objects are strong and the number of objects involved is large, which is precisely the case with patches. This complex approach is then not possible here and it is simpler to directly compute probability distribution of patches.

Even though one can imagine ecological processes that would occur at singular points more or less randomly distributed in space, most if not all LUCC models tackle LUCC changes that are known to occur in patches, *i.e.*, across many connected pixels at once, and usually for quite a few such patches per unit time step. As single pixel probabilities are easy to manage, the community has developed these methods without considering formal relations between single pixel probabilities and probability distribution of patches. Pixels drawn by Dinamica by single pixel probabilities are therefore considered as kernel pixels and a patch builder called to allocate empirically pixels around the kernel pixel according to patch parameters defined by the user. It should be noted that this is the most sophisticated approach used in the literature and in practice to date.

There is definitely a relation between kernel pixel probabilities and single pixel probabilities.

Under certain conditions which are probably often satisfied in practice, the both probability distributions are virtually identical. This aspect is not addressed at all in the literature but Pierre-Yves Longaretti has studied it and has identified conditions of validity testable on data. His work is not yet published and it will be reviewed and enriched during the course of the PhD.

5.3 Defining conceptually correct pruning

Because the explanatory variables Z are initially chosen in terms of their relevance for the transitions to be allocated, one expects that the higher $P(\hat{z}|v_i, v_f)$ for any pixel, the higher the contribution of such pixels to the overall transition. This intuitive argument suggests that this is the correct quantity to be used for pruning, a point that we justify here in a more formal way.

Recall firsts the point made above that for the transition $v_i \rightarrow v_f$, the first pruning selection criterion is obvious: one keeps only pixels belonging to \mathbb{J}_{v_i} as v_i must be a conditionally known quantity in the process.

To proceed further, note that

$$\#\mathbb{J}_{v_i, v_f} = P(v_i|v_f)\#\mathbb{J}_{v_i} = \sum_{\hat{z}} P(\hat{z}, v_f|v_i)\#\mathbb{J}_{v_i} \quad (18)$$

is the total number of pixels changing state from v_i to v_f . Therefore, the higher $P(\hat{z}, v_f|v_i)$, the most likely the corresponding pixels ($\mathbb{J}_{v_i, v_f, \hat{z}}$), whose number amounts to $\#\mathbb{J}_{v_i, v_f, \hat{z}} = P(\hat{z}, v_f|v_i)\#\mathbb{J}_{v_i}$, are to contribute to \mathbb{J}_{v_i, v_f} . Conversely, one cannot rely on $P(v_f|\hat{z}, v_i)$ to identify the most likely pixels to undergo the required transition, as there is no closure relation on \hat{z} on this probability, *i.e.* $\sum_{\hat{z}} P(v_{p+1}|\hat{z}, v_p) \neq 1$; this simple remark disqualifies the choice made in Dinamica. In fact, the probability distribution $P(\hat{z}, v_f|v_i)$ is clearly the only one having $P(v_f|v_i)$ as marginal probability at given starting state, so that the point made in this paragraph is unavoidable.

Note that one can equivalently use $P(\hat{z}|v_i, v_f) = P(\hat{z}, v_f|v_i)/P(v_f|v_i)$ to define a pruning ensemble for the $v_i \rightarrow v_f$ transition, because $P(v_f|v_i)$ is pixel independent; algebraically, this follows because one can also write $\#\mathbb{J}_{v_f, v_i, \hat{z}} = P(\hat{z}|v_f, v_i)\#\mathbb{J}_{v_i, v_f}$, so that the higher $P(\hat{z}|v_f, v_i)$, the most likely the corresponding pixels ($\mathbb{J}_{v_i, v_f, \hat{z}}$) will contribute to \mathbb{J}_{v_i, v_f} .

An extreme case of this argument occurs when a vast majority of pixels j have zero contribution to the sum, *i.e.*, $P(\hat{z}, v_{p+1}|v_p) = 0$. Removing these pixels from \mathbb{J}_{v_p} does not change anything in the final allocation, but considerably reduces the required number of random pixel selection. By extension, removing the pixels with small probability $P(\hat{z}|v_i, v_f)$ should have little effect on the result.

At first sight, it might be surprising that the pruning criterion involves a probability distribution $P(\hat{z}|v_i, v_f)$ that differs from the allocation procedure one $P(v_f|\hat{z}, v_i)$. On second thought, this is quite normal: the pruning criterion occurs prior to randomly selecting a pixel, *i.e.* before \hat{z} is known, while the allocation procedure occurs after random selection, *i.e.* after \hat{z} is known. Pruning and allocation are therefore two essentially different processes, and there is no reason they should be based on the same conditional probability.

Let us now turn to the actual elaboration of a pruning procedure. Consider first the case with a single transition $v_i \rightarrow v_f$. According to the preceding argument, it is logical to order the pixels in decreasing values of $P(\hat{z}|v_i, v_f)$.

For any given $\tau \in [0, 1]$, one can define a subset of $\hat{\mathbb{Z}}$:

$$\hat{\mathbb{Z}}_\tau = \{\hat{z} \in \hat{\mathbb{Z}} \mid \sum_{\hat{z} \in \hat{\mathbb{Z}}} P(\hat{z}|v_i, v_f) \geq \tau\} \quad (19)$$

where \hat{z} in the sum are taken in decreasing values of $P(\hat{z}|v_i, v_f)$. Then, pixels are selected as follows:

$$\mathbb{J}_{v_i, v_f, \tau} = \{j \in \mathbb{J}_{v_i} \mid \hat{z}^j \in \hat{\mathbb{Z}}_\tau\} \quad (20)$$

In plain words, one keeps the fraction τ of the highest probabilities contributing to the cumulative transition probability. Note that if $\tau = 1$, no pruning is performed.

In case of multiple transitions, selected pixels for each transition $v_i \rightarrow v_f$ are gathered by union (note that τ may well be different for each transition):

$$\mathbb{J}_{v_i, \tau} = \bigcup_{v_f} \mathbb{J}_{v_i, v_f, \tau} \quad (21)$$

Later on, for the specific case study that we use to illustrate the formal points made in this report, we make use of a specific procedure that allows us to forego the use of pruning. Nevertheless, the present discussion of pruning serves two purposes:

1. Doubtless, some problems involve too many pixels and too many variables, making pruning necessary for computational efficiency purposes. The present discussion, establishing the correct pruning procedure, is therefore unavoidable in such instances.
2. The clarifications exposed here allow us to analyze in a quantitative way the biases involved both in this pruning procedure and the incorrect ones used in existing modeling environments. This was not possible during the course of this internship by lack of time, but this question will be addressed in the course of the PhD that will pursue the present work where it is left off, among other objectives.

6 New land use change allocation algorithm

Dinamica allocation principle is based on a pruning selection. Indeed, this software spends a large part of the computation time to pick and reject pixels. It seems that bad development choices are behind this wasting operation. In order to conceive a new allocation algorithm which could be fast and theoretically correct, we developed a new allocation procedure called Demeter⁵ written in Python 3. This allocation procedure will eventually turn into a new full LUCC modeling environment during the course of the PhD that will be devoted to the problem. This language choice, different from Dinamica which is written in C++, has several advantages. First, Python is an interpreted, high-level, general-purpose programming language which has a high code readability and provides a large amount of efficient libraries. Three libraries are especially employed in Demeter: Numpy for large matrix manipulations, Scipy for digital image processing (*i.e.* map processing) and Pandas for database manipulation. These libraries enable the design of a high efficient LUCC model. Second, the desktop graphic information system QGIS⁶ provides a very simple python plug-in implementation. That way, it is straightforward to design a GUI layer over our python model to call it and visualize directly the results on QGIS.

This new allocation algorithm is based on a very efficient generalized acceptance rejection test which is able to evaluate transition of a large number of pixels for all possible transitions as described in section 6.2.

6.1 Calibration and allocation scenario

The calibration phase provided for all pixel j probabilities for transition from v_i to any other v_f according to the Bayes formula (eq. 4 reminded below).

$$P(v_f|\hat{z}, v_i) = P(v_f|v_i) \times \frac{P(\hat{z}|v_i, v_f)}{P(\hat{z}|v_i)} \quad (4 \text{ revisited})$$

$P(\hat{z}|v_i, v_f)$ and $P(\hat{z}|v_i)$ are both coming from the calibration. Using the explanatory variable independence hypothesis allows us to deal with combinations of explanatory variable values that are not present in the calibration data. $P(v_f|v_i)$ is considered as a scenario parameter and is defined by the user. This probability is directly related to the targeted transitioned surface $\#\mathbb{J}_{v_i, v_f} = \#\mathbb{J}_{v_i} P(v_f|v_i)$.

6.2 Generalized acceptance rejection test

An essential step of the allocation process concerns the transition test from v_i to v_f . In order to insure a statistically correct draw, it is necessary to test all possible transitions at the same time. Von Neumann (1951) presents a method to test a simple acceptance function. A multiple acceptance rejection test is proposed by Sunter (1977) without replacement in the sample. We define therefore a simpler method of generalized acceptance rejection test (GART) which is used to determine a potential land use change for a given pixel. The simplification is that the draw probabilities are not updated to reflect the fact that the overall sample is decreasing. One justifies by the fact that the sample is very large in front of the number of pixels drawn.

⁵In ancient Greek mythology, Demeter is the goddess of the harvest and agriculture, presiding over grains and the fertility of the earth.

⁶QGIS is a free and open-source cross-platform desktop geographic information system (GIS) application that supports viewing, editing, and analysis of geospatial data.

For a pixel j , knowing v_i^j and \hat{z}^j , the calibration allows to determine $P(v_f|v_i^j, \hat{z}^j)$ for all v_f (eq. 4). For this purpose, we define $\eta_p, \forall p \in \llbracket 0, v^m \rrbracket$ as the cumulative sum of $P(v_f|v_i^j, \hat{z}^j)$:

$$\begin{aligned} \eta_0 &= 0 \\ \forall p \in \llbracket 1, v^m \rrbracket, \quad \eta_p &= \sum_{v_f=1}^p P(v_f|v_i, \hat{z}) \end{aligned} \quad (22)$$

Then, a simple random float is sufficient to test simultaneously all possible transitions for j . For x , a random float in the half-open interval $[0, 1)$, it exists η_p such as $\eta_{p-1} \leq x < \eta_p$. The test returns in that case p .

We define φ as the random function of the generalized acceptance rejection test which takes the pixel j and returns a state $p \in \mathbb{V}$ as a random variable. If $p \neq v_i^j$, the test is said as *accepted* and as *rejected* otherwise.

In fact, we consider output states as sections of the unit interval with length equal to the various transitions probability; the total length is then equal to one. This test presents several advantages: all possible final states are tested at the same time, the order of the sections has no importance and it is numerically very efficient to apply such a test to a large set of pixels. An example of python implementation of such test similar to the one used in Demeter is presented in the listing 1 and illustrates the cited advantages. In this piece of code, `J_vi` is a pandas DataFrame which informs all pixels transitions probabilities. Its columns are pixels indexes j and some transitions probabilities whose columns names are listed in `P_names`. As usual, `vi` is the studied initial state and `vf` is here a list of possible transited states in the same order as transitions probabilities columns.

Listing 1: Generalized Acceptation Rejection Test applied in Python using the Pandas library

```

1 J_vi[ 'vf_kernel_candidate' ] = vi
2 J_vi[ 'x' ] = np.random.random(J_vi.j.size)
3
4 for i in range(len(vf)-1, -1, -1):
5     J_vi.loc [ J_vi.x < J_vi[ P_names[ i ] ], 'vf_kernel_candidate' ] = vf[ i ]

```

6.3 Allocation algorithm

The allocation algorithm deployed by Dinamica is time expansive due to pixel draws even if it is reduced by pruning. We present here a new allocation procedure which requires no pruning and reduces the computing time significantly.

The global idea is to reduce the necessity of loops on given pixels which are expensive in term of running time. For this purpose, the procedure begins by applying the generalized acceptance rejection test on all concerned pixels as presented in section 6.2, then determines a patch surface for all accepted pixels, selects just enough of kernel pixels to complete the targeted surface and finally allocates these patches with a loop.

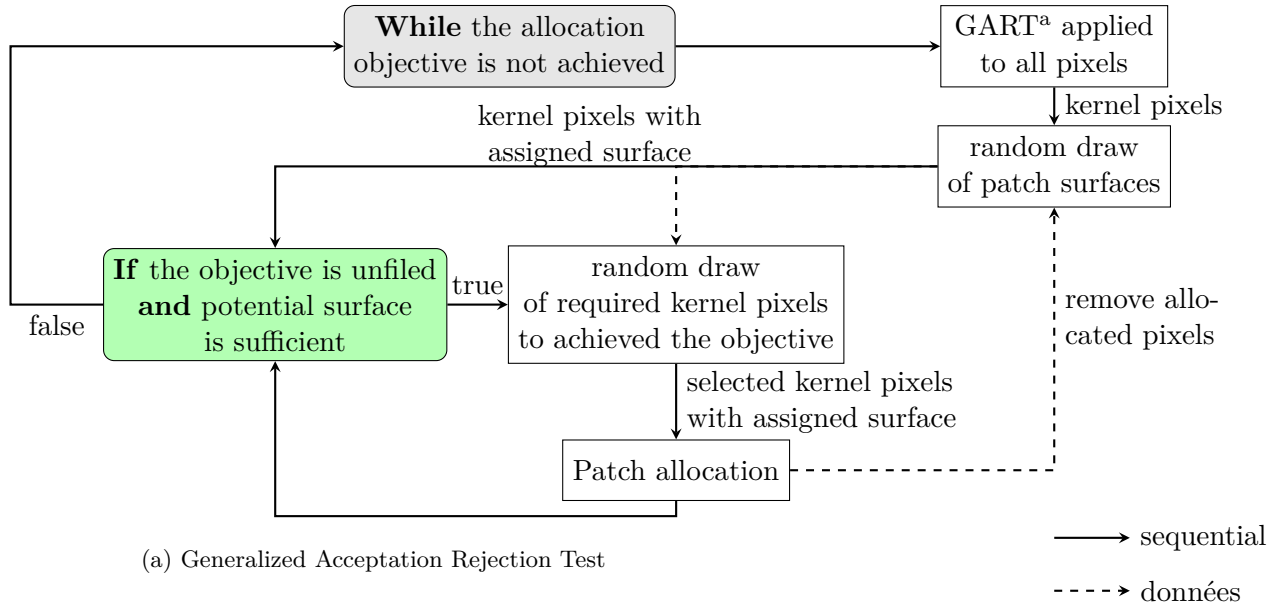


Figure 4: Schematic new allocation procedure for each transition

The figure 4 presents a schematic algorithm of the allocation procedure and the flowchart 5 presents the allocation algorithm in a technical way and is explained in the following (numbers in parenthesis refers to numbers on the figure 5). For each v_i (the order has no consequence), we apply a generalized acceptance rejection test φ to all pixels of \mathbb{J}_{v_i} (4). Then, for all possible transitions $v_f \neq v_i$, accepted pixels, i.e. with $\varphi(j) = v_f$ are selected and constitute sets written \mathbb{J}'_{v_i, v_f} which are sampled. Those pixels are considered as candidates to become transited patch kernels. For each, a random function draws patch surface according to the patch surface distribution for the considered transition⁷ (5). The obtained sets are commonly large regarding the number of patch kernels required. Thus, for each v_f , we select just enough of kernel pixels to fill the targeted surface and selected pixels are removed from \mathbb{J}'_{v_i, v_f} . We obtain in this way for each transition a set of candidates and their sampled union is \mathbb{J}_{v_i} (7). Finally, each $j \in \mathbb{J}_{v_i}$, which are not yet transited, are processed by the patch design algorithm with the previously determined surface S_j (8). This method could lead by construction to a lower transited surface than the target one because of patch overlapping. Then, if the target is not reached down to ϵ for any v_f , and if for uncompleted v_f , kernel candidates \mathbb{J}'_{v_i, v_f} provide enough potential surface, a new \mathbb{J}_{v_i, v_f} is constituted from remaining pixels within \mathbb{J}'_{v_i, v_f} (6). Otherwise, a new draw should be done on all pixels which are not yet transited (3). When the total transited surface is reached down to ϵ for all v_f , a new initial state v_i is studied (2). The algorithm finally returns the allocated map.

In order to prevent some pathological cases, we have also implemented two additional parameters which limit the number of while loops for (2) and (5).

This algorithm has been implemented in Demeter and the first results are extremely convincing. For the same given real case of study, with the same discretization and calibration, Dinamica has taken 15 minutes of computation time whereas Demeter has taken only 5 seconds, that being a computation time relation of 100. More precise tests will be performed in the future.

⁷The patch surface distribution can be empirically defined or erected during the calibration phase

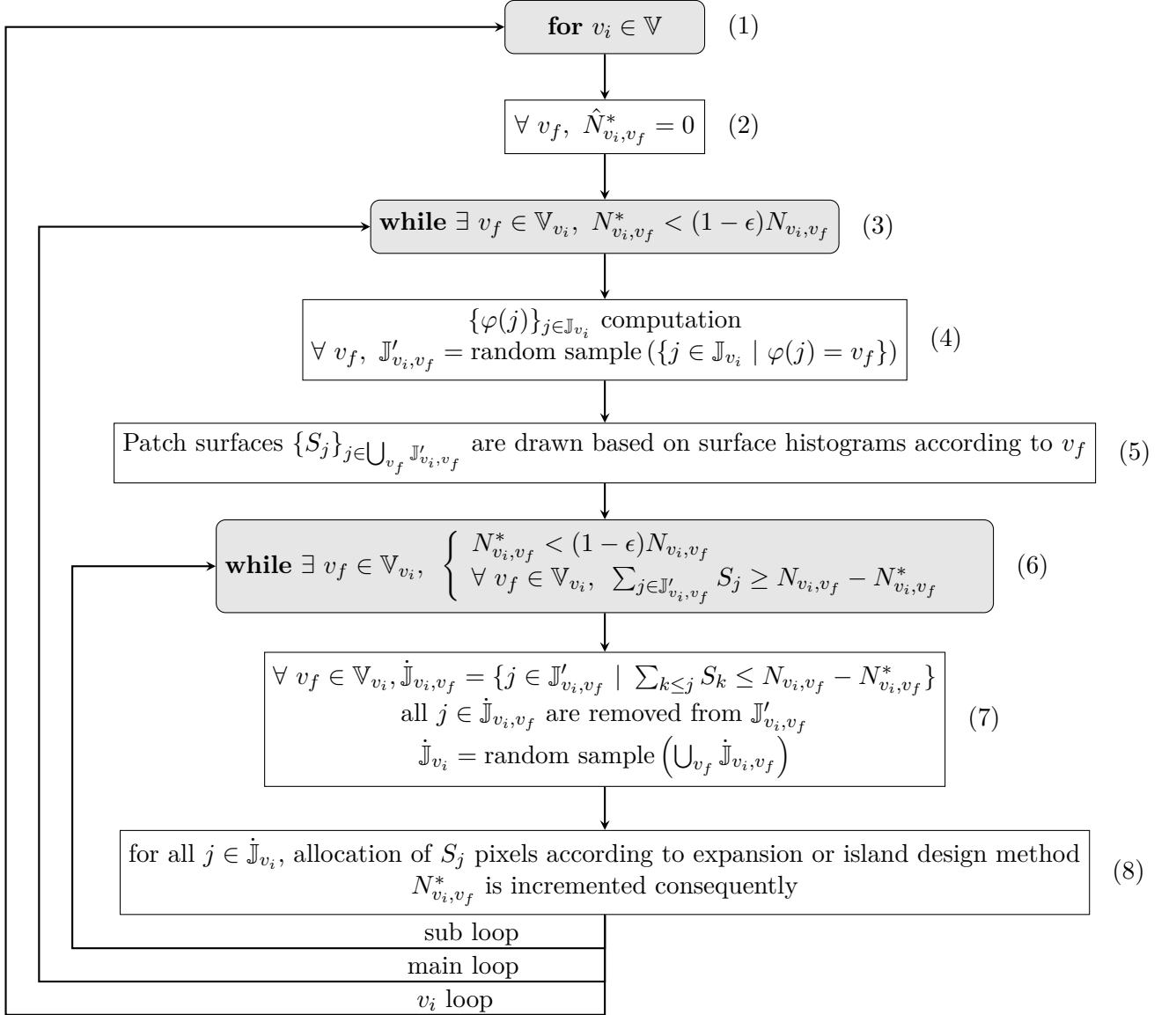


Figure 5: New allocation algorithm

7 Islands and expansions allocation procedures

The actual allocation procedure has no vocation to be a prediction. It should only present a plausible final situation which corresponds to the requested scenario. Two patches types can be distinguished: expansions and islands. The first one refers to transitions which occur on the boundary between v_i and v_f . The second one is about transitions which come out in v_i without any v_f adjacency.

In principle, there is no reason to make this distinction – it occurs inherently due to the distribution of patch sizes and statistic choices of kernel pixel positions. In practice, distinguish expansion and island patches is useful for two reasons: certain transitions present actually different characteristics in both cases and it could also be numerically easier to use specific algorithm for both cases.

7.1 Calibration

In order to model patches, we consider only the surface parameter for both island and expansion types. It is thus necessary to evaluate patches surface during the calibration phase in order to obtain a probability distribution which can be then adapted to the desired scenario.

The process to determine if a calibration patch is an island or an expansion can be time and computing expensive. The idea is to select \mathbb{J}_{v_i, v_f} and \mathbb{J}_{v_f} and to determine distinct groups of pixels for both by using the `scipy` function `ndimage.measurements.label`. A group of pixel is simply a set of neighboring pixels. An index is assigned to each groups. For a group of \mathbb{J}_{v_i, v_f} , the group of \mathbb{J}_{v_f} which presents the same pixels is identified. Finally, both groups are compared by size. If their size are equal, then it is an island patch. Otherwise, it is an expansion patch. Patches histograms are finally computed by `numpy` which provides efficient bins parameters.

The figure 6 presents the expansion surface histogram for the transition crop to urban in the SCoT case. It is then possible to directly apply this probability density in the allocation phase.

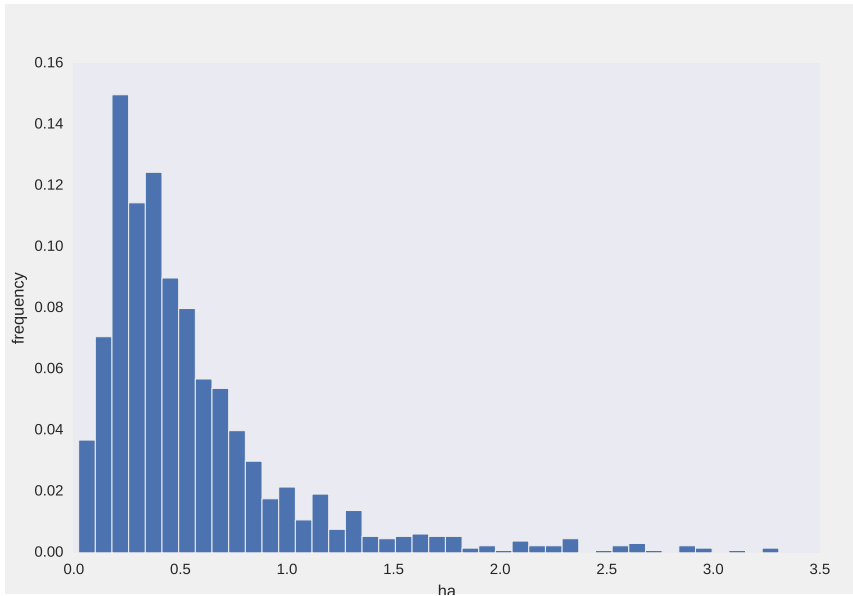


Figure 6: Expansions surface in hectare for the transition crop to urban in the SCoT case

7.2 Island design process

In order to design a patch, all methods begin from a first pixel called kernel pixel. Then, several possibilities can be considered to create an island patch associated to this kernel pixel:

- Dinamica uses a recursive algorithm based on an anisotropic parameter.
- It is possible to sample the required shape from a bank of shapes, itself constructed from the calibration data. Then, a geometric similarity is randomly applied (size and orientation) to correspond to the desired surface.
- It is also possible to use a simple elliptic function or any other shape with the desired surface. This option however will tend to produce figures that are too regular compared to reality.

Only a very simple growth algorithm has been implemented, spiraling outwards from the kernel pixel. This is not intended to be realistic in any way, but as a first and quick step to produce actual patches and make some simple statistical tests and checks in the limited time available in the present internship.

In practice, the process starts from a pixel j and recursively selects pixels along a clockwise spiral. If the selected pixel is in state v_i , it is transited to state v_f and the next pixel is tested, otherwise the algorithm goes directly to the next pixel. The spiral ends when the desired island surface is reached. An actual example coming from a real case is represented in figure 7. On this map, a relatively large island is constructed in a crop zone. The rectangular shape is an artefact of the over-simplistic algorithm just described, but satisfies two constraints: the mean position and the selected patch surface.

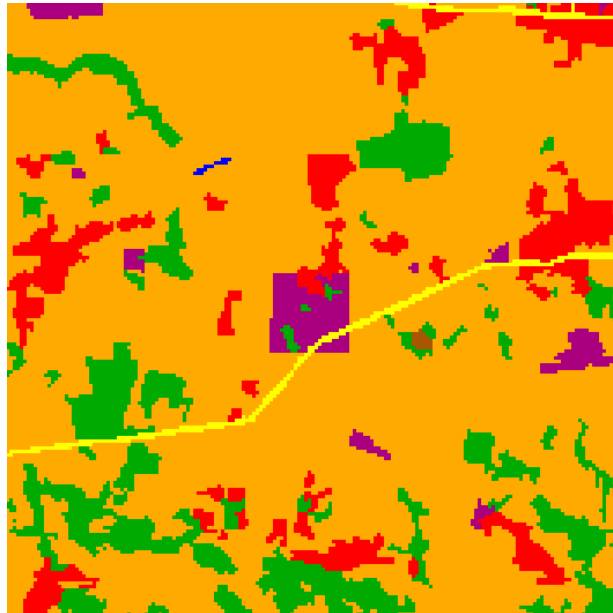


Figure 7: Island designed by the spiral algorithm – focus on a small part of the study area – State colors are defined in table 2.

7.3 Expansion design process

The expansion process implies to evaluate the transition probability on the border of already present v_f pixels. Soares-Filho et al. (2002) propose a method which favors pixels with higher $P(v_f|v_i, \hat{z})$ and higher number of v_f neighbors. I propose here a method based on matrix operations in order to reduce computing time cost. In order to select border pixels, we perform the following convolution:

$$M_{v_f}^* = M_{v_f} * \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad (23)$$

where M_{v_f} is a zero matrix of dimension equal to the map whose values of coordinates of pixels belonging to \mathbb{J}_{v_f} are 1. $M_{v_f}^*$ is then a matrix which indicates for each pixel the number of v_f neighbors. It is then easy to select pixels from \mathbb{J}_{v_i} with at least one neighbor in the desired post-transition state. We can then evaluate transition probability for these border pixels.

The expansion process idea is based on the evaluation of the transition probabilities around already transited pixels. However, a simple algorithm which takes the highest probability around pixels of the desired state can lead to create biased expansion patches, again because correlations between pixels are ignored in such a procedure. A simple way to create more satisfying patches is to draw the new pixel in the list of neighbors with probabilities weights scaled to one. It is presented in the algorithm 1 which does not verify if a pixel is already in the pixel neighbors list. This fact gives more weight to pixels with several neighbors already transited.

Algorithm 1: New allocation procedure

Data: v_i, v_f , the kernel pixel j and the desired surface S

Result: J_{v_i, v_f}

$S^* = 0$

add all neighbors of j which are v_i in \mathbb{J}_{j^*}

while $S^* < S$ and $\#\mathbb{J}_{j^*} > 0$ **do**

P is the set of ordered transition probabilities of \mathbb{J}_{j^*}

 a pixel to transite is drawn in \mathbb{J}_{j^*} with P as weights

S^* is incremented by one

 all neighbors of the new transited pixel, which are v_i , are appended to \mathbb{J}_{j^*}

Figure 8 represents an example of the expansion process applied to a real case. In this figure, urban areas are in red whereas their allocated expansions are in cyan and industrial areas are in purple whereas their allocated expansions are in light purple. At first sight, the algorithm provides quite satisfying expansion shapes. However, “holes” are apparent; they are constituted of “orphan” v_i pixels that were not selected for transition. These are unrealistic, and must be removed. The `scipy` library provides again a useful function `ndimage.morphology.binary_fill_holes`.

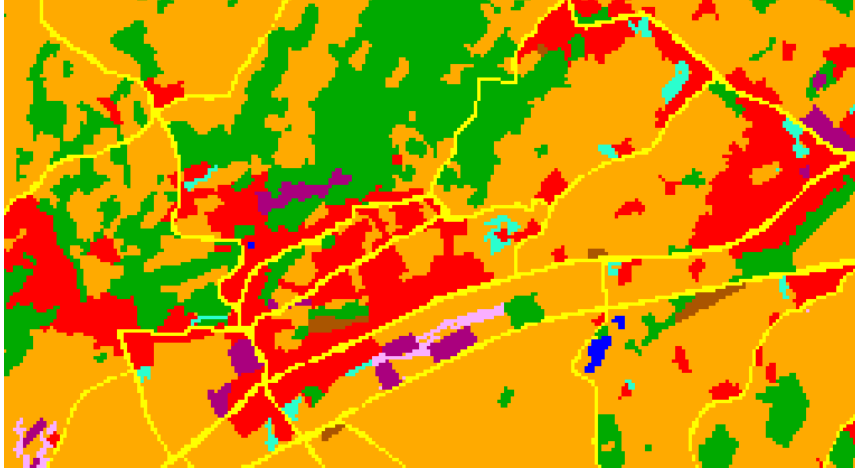


Figure 8: Expansions designed by the algorithm 1 – focus on a small part of the study area – State colors are defined in table 2 – $\mathbb{J}_{3 \rightarrow 2}$ is in cyan and $\mathbb{J}_{3 \rightarrow 7}$ is in light purple.

7.4 Patch design implementation within the allocation algorithm

Taking islands and expansions design algorithms in consideration requires to modify the allocation process presented in section 6.3. The flowchart in figure 9 presents the patch design implementation within the new allocation algorithm. Several changes have been made and are detailed in the following where numbers in parenthesis refer directly to the ones in the figure.

First, expansions are allocated and secondly island patches are designed (1). It is an arbitrary choice motivated by tendency of expansion patches to catch up with island ones if operations would be performed in the opposite order. For all v_f , \mathbb{J}_{v_i, v_f}^c refers to the selected pixels within \mathbb{J}_{v_i} regarding such patch type (2). More precisely, in the expansion case, only pixels with v_f neighbors are selected and in the island case, only pixels far enough from v_f areas are selected – the *far enough* definition has to be defined according to the applied island design algorithm. One defines the island – expansion ratio η according to surface values (3). Thus, the target N_{v_i, v_f} is adapted to the patch type:

$$\hat{N}_{v_i, v_f} \begin{cases} \eta N_{v_i, v_f} & \text{in the island case} \\ (1 - \eta) N_{v_i, v_f} & \text{in the expansion case} \end{cases} \quad (24)$$

and consequently \hat{N}_{v_i, v_f}^* is the target counter according to the current patch type. the acceptance rejection test has only one positive issue in the expansion case which corresponds to the v_f neighbour (4). Indeed, in the expansion case, the pixel close to v_f can only transit to v_f . Particular cases where a pixel is near two different v_f are solved by expansion design construction: the two outputs are uniformly considered and the ensemble sampling determines a priority order between transitions. patches surface histogram are different for expansion and island cases (5). Finally, patch design is of course performed according to the expansion or island case (6).

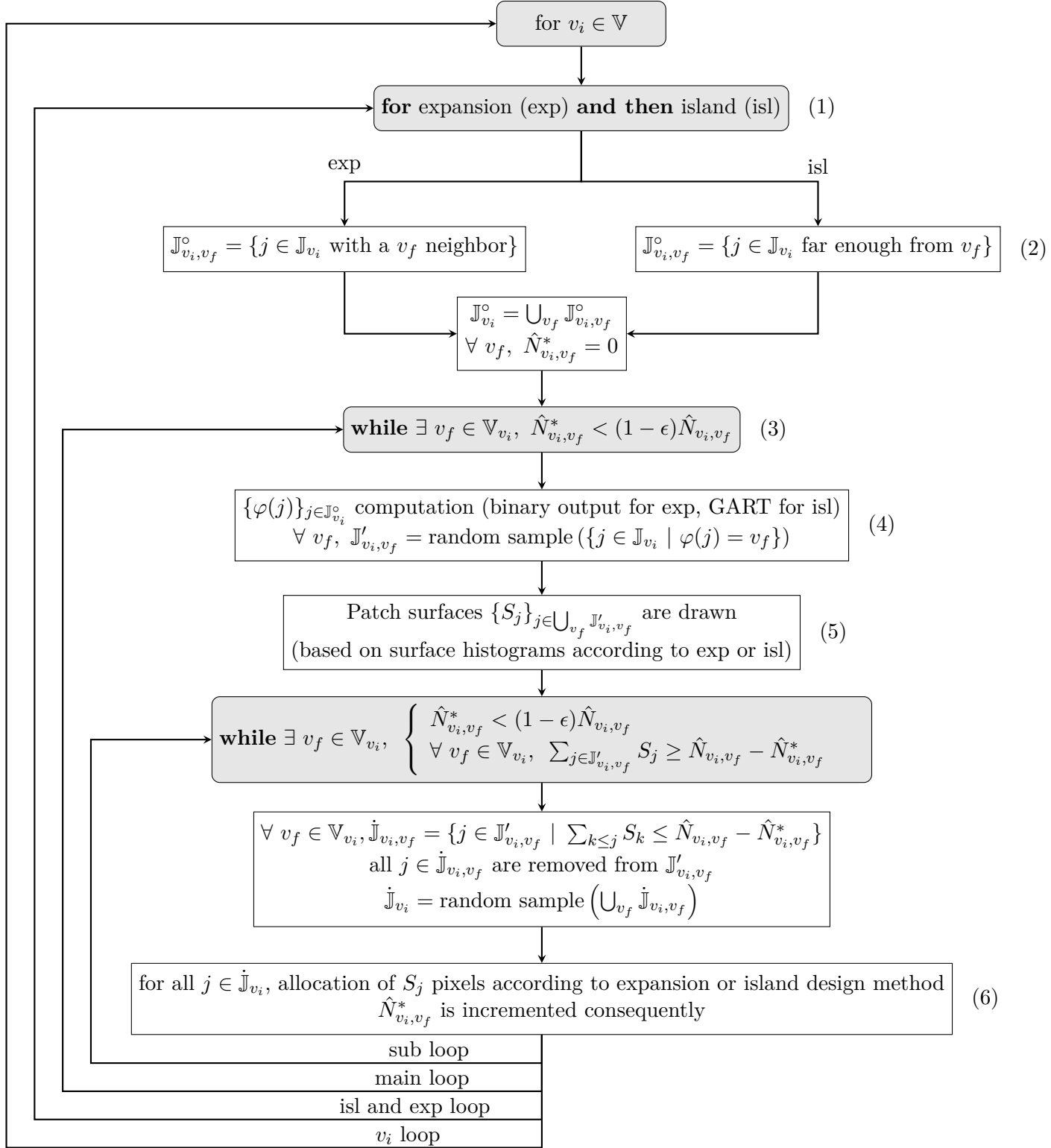


Figure 9: Demeter allocation process implementing expansion and island patch formation.

8 Conclusion

This preliminary work of the PhD has introduced several aspects of LUCC models which require more detailed studies. It takes part in the progression toward a general theoretical framework. Some bias implied by the supposed independence of explanatory variables has been described and should be the main topic of a future work by defining a threshold condition of independence. Likewise, the patch point of view for LUCC has been presented and conditions to use single pixel probabilities to draw kernel pixels deserve a specific attention in the future. I focused on the allocation procedure by analysing Dinamica choices, especially about the pruning method which is proving to use a wrong statistical value ($P(v_f|v_i, \hat{z})$ instead of $P(\hat{z}|v_i, v_f)$). A correct pruning method is then defined. I introduce a new allocation method which apply simultaneously the acceptance rejection test to all pixels. It is then convenient to draw kernel pixels within the accepted population. This procedure avoid the pitfall of Dinamica which draws a large amount of pixels which are then rejected. This new algorithm already shows off very promising results with a saving of time of calculation of factor 100. More precise tests will be performed in the future. Finally, numerical considerations have been advanced about the practical allocation operation for both patch types that are island and expansion.

Future work are already envisaged. First on the allocation procedure, allocation patch design, performance comparison with other softwares and pruning error evaluation have to be studied. Also the theory of patch modeling have to be explored as well as markovian aspect of LUCC models.

References

- Chetan Agarwal, Glen M. Green, J. Morgan Grove, Tom P. Evans, and Charles M. Schweik. A review and assessment of land-use change models: Dynamics of space, time, and human choice. Technical Report NE-GTR-297, U.S. Department of Agriculture, Forest Service, Northeastern Research Station, Newtown Square, PA, 2002. 1.1
- J. Ronald Eastman. *IDRISI Selva Manual, Guide to GIS and Image Processing*. Clark University, 2012. 1.1
- Alex Hagen. Fuzzy set approach to assessing similarity of categorical maps. *International Journal of Geographical Information Science*, 17(3):235–249, April 2003. ISSN 1365-8816, 1362-3087. doi: 10.1080/13658810210157822. 1.2
- Jean-François Mas, Melanie Kolb, Thomas Houet, Martin Paegelow, and Maria Camacho Olmedo. Éclairer le choix des outils de simulation des changements des modes d’occupation et d’usages des sols. Une approche comparative. page 27, 2011. 1.3
- Jean-François Mas, Melanie Kolb, Martin Paegelow, María Teresa Camacho Olmedo, and Thomas Houet. Inductive pattern-based land use/cover change models: A comparison of four software packages. *Environmental Modelling & Software*, 51:94–111, January 2014. ISSN 13648152. doi: 10.1016/j.envsoft.2013.09.010. 1.3
- R Gil Pontius. Statistical Methods to Partition Effects of Quantity and Location During Comparison of Categorical Maps at Multiple Resolutions. *PHOTOGRAMMETRIC ENGINEERING*, page 9, 2002. 1.2
- Britaldo Silveira Soares-Filho, Gustavo Coutinho Cerqueira, and Cássio Lopes Pennachin. Dinamica—a stochastic cellular automata model designed to simulate the landscape dynamics in an Amazonian colonization frontier. *Ecological Modelling*, 154(3):217–235, September 2002. ISSN 03043800. doi: 10.1016/S0304-3800(02)00059-5. 1.1, 7.3
- A. B. Sunter. List Sequential Sampling with Equal or Unequal Probabilities without Replacement. *Applied Statistics*, 26(3):261, 1977. ISSN 00359254. doi: 10.2307/2346966. 6.2
- Peter H. Verburg and Koen P. Overmars. Combining top-down and bottom-up dynamics in land use modeling: Exploring the future of abandoned farmlands in Europe with the Dyna-CLUE model. *Landscape Ecology*, 24(9):1167–1181, November 2009. ISSN 0921-2973, 1572-9761. doi: 10.1007/s10980-009-9355-7. 1.1
- Peter H. Verburg, Welmoed Soepboer, A. Veldkamp, Ramil Limpiada, Victoria Espaldon, and Sharifah S.A. Mastura. Modeling the Spatial Dynamics of Regional Land Use: The CLUE-S Model. *Environmental Management*, 30(3):391–405, September 2002. ISSN 0364-152X, 1432-1009. doi: 10.1007/s00267-002-2630-x. 1.1
- Peter H. Verburg, Kasper Kok, Robert Gilmore Pontius, and A. Veldkamp. Modeling Land-Use and Land-Cover Change. In Eric F. Lambin and Helmut Geist, editors, *Land-Use and Land-Cover Change*, pages 117–135. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-32201-6 978-3-540-32202-3. doi: 10.1007/3-540-32202-7_5. 1.1
- John Von Neumann. Various techniques used in connection with random digits. *National Bureau of Standards*, 1951. 6.2