Gen*: a generic toolkit to generate spatially explicit synthetic populations

Kevin Chapuis, Patrick Taillandier, Renaud Misslin, Alexis Drogoul

tGIS2e

Gen*: a Generic Toolkit to Generate Spatially Explicit Synthetic Populations

Chapuis Kevin[a], Taillandier Patrick[b], Misslin Renaud[c] and Drogoul Alexis[d]

[a] *UMI 209 UMMISCO, IRD/UPMC, Bondy, France*; [b] *MIAT, University of Toulouse, INRA, Castanet-Tolosan, France* [d] *ICTLab, USTH, VAST, Hanoi, Vietnam*

(*Received 27 June 2017; final version received 12 February 2018*)

Agent-based models tend to integrate more and more data that can deeply impact their outcomes. Among these data, the ones that deal with agent attributes and localization are particularly important, but are very difficult to collect. In order to tackle this issue, we propose a complete generic toolkit called Gen* dedicated to generating spatially explicit synthetic populations from global (census and GIS) data. This article focuses on the localization methods provided by Gen* that are based on regression, geometrical constraints and spatial distributions. The toolkit is applied for a case-study concerning the generation of the population of Rouen (France) and shows the capabilities of Gen* regarding population spatialization.

Keywords: Synthetic population; Spatialization; Social simulation; Multi-agent model

1. Introduction

Agent-based simulations are now widely used by researchers in the social sciences to study complex systems. Whereas early models tended to be very simple and abstract – the KISS (Keep It Simple, Stupid) approach (Axelrod 1997) – the current trend is now towards the development of richer data driven models – the KIDS (Keep It Descriptive, Stupid) approach (Edmonds and Moss 2005). The motivation behind this is principally the use of such models as predictive tools in a decision-support context.

In such models, behaviors of social agents (e.g., individuals, households or institutions) are strongly determined by their attributes, well as by their location in the artificial worlds they populate. In this context, generating synthetic populations of spatially localized agents that conform to the data available on real populations has become a necessity and a concern for most social modelers.

Some recent studies have already proposed methods to generate these types of populations (Harland *et al.* 2012, Cornelis *et al.* 2013, Swarup and Marathe 2016). However,

*Corresponding author. Email: kevin.chapuis@gmail.com

these methods remain underused because of the difficulty to use them with data that is not well-formatted: for example, the simSALUD (Kosar and Tomintz 2015) web-based generator requires input data to fit a template with fixed position columns that describe attribute IDs, weights, etc. Along the same lines, the SPEW (Gallagher *et al.* 2017) R-based generator requires input data to be in form of IPUMS formatted data tables. Therefore, most attempts to generate localized synthetic populations are limited to empirical cases (Arentze *et al.* 2008, Barthelemy and Toint 2012) and propose methods with stringent requirements as to the type, availability and format of the data. Furthermore, the classical approach considers spatialization of the population in the same way as the other agent attributes: being a woman or a man is the same as living there or in the next county. Indeed, the spatial information usually comes from the same source of data: either censuses or surveys. Yet, this spatial information is generally not precise enough for many simulation cases. For instance, many residential mobility models work at the building scale (Antoni *et al.* 2016), and therefore require a specific building to be assigned to each household and not just a city or a district. It is thus often necessary to specify or extrapolate a localization from spatial data. In this case, the problem is not simply to infer an inner distribution of demographic attributes but to estimate what actually drives the spatial localization of people. As pointed out by Anderson *et al.* (2014), obtaining a reliable population estimation at a given resolution is a significant challenge.

To our knowledge, there has been only one recent attempt to use generic input data for population generation and localization: the SPEW open source R library (Gallagher *et al.* 2017). However, even with this library, a data sample of a portion of the population from IPUMS is needed in order to generate a reliable synthetic population. Concerning the localization of the population, SPEW proposes some simple methods to uniformly locate entities across a region or along roads, but does not allow to take benefits of multiple layers of information.

To deal with this lack of flexibility, we propose a complete toolkit known as Gen*, which allows modelers to generate a realistic synthetic population even with not-classically formatted data and to localize it at the required scale from one or several layers of geographical data. This toolkit is developed as an open-source Java library. An extension of the GAMA platform (Grignard *et al.* 2013) and a Kepler workflow package that make it possible for non computer scientist to generate a spatially explicit population with Gen* through user friendly interfaces are available as well. In what follows, we focus on the the localization features of Gen*.

The article is structured as follows: Section 2 presents the related studies concerning the spatialization of synthetic populations. Section 3 is dedicated to the presentation of our toolkit. More precisely, we give an overview of the global architecture of the toolkit, and we present the spatialization methods. A case-study of the city of Rouen (France) is described in Section 4. Finally, Section 5 proposes a conclusion and perspectives for this work.

2. Context

In many agent-based social simulations the geographical environment has to be taken into account with a granularity that depends on the objectives of the model and the geographical data available. Simple models can rely on conceptual environments, i.e. continuous or discrete space. However, applied and descriptive models tend more and more to rely on GIS data, possibly combining geometrical (shapes and set-relationships among shapes)

and continuous information (location of objects and agents in the underlying continuous space).

When dealing with GIS data, several methods are classically used in order to integrate census data that describe the population, but they usually do not generate populations of agents. Among the approaches available in this case, the census centroid data (Bracken and Martin 1989) is the most widely used. Dasymetric modeling is also a well-recognized approach for the spatial decomposition of census data in order to increase the spatial resolution of the population distribution (Bhaduri *et al.* 2007). A combination of the methods described above is often used in an ad hoc manner. This is, for example, the case in the SVERIGE project (Holm 2002) that aims at modeling the demographic evolution of Sweden by explicitly modeling (including the localization) each inhabitant of the country.

Two main steps can be used to address the localization problem: the first one referred to as *Areal interpolation*, consists in translating data from aggregated areas to more precise ones. The second step referred to as *Explicit localization*, consists in providing individuals with a precise location, that could be a coordinate (x,y) or a geographical object (e.g. a building). In the next sub-sections, we review the available methods that can be used to perform these two steps, and we briefly outline the typical data that are used by modelers to carry out the localization process.

2.0.1. Areal Interpolation Methods (AIM) and the dasymetric method

Areal Interpolation Methods aim at transferring spatial data from one set of spatial units to another (Eicher and Brewer 2001). Redistribution of population data from an administrative unit to a smaller unit (usually a pixel of a grid matching a remote sensing image) is a particular type of areal interpolation known as "dasymetric mapping". This type of method uses ancillary data to refine the population data originally given at a coarser level (Li and Weng 2010). Not only can these methods provide users with a solution to produce more precise population data, but they also make it possible to reduce the errors resulting from the modifiable areal unit problem (MAUP) (Su *et al.* 2010).

The redistribution of populations from census units to a finer scale using dasymetric mapping is a two-steps process. The first step involves the fitting of a statistical model like a linear regression model (Li and Weng 2005, Reibel and Agrawal 2007) or a random forest model (Stevens *et al.* 2015). The second step of the dasymetric process consists in applying the statistical relationships resulting from the model fitting at the finer scale. This final step involves some extra normalization processes because of the possible floating point or negative results from the statistical model.

The use of dasymetric mapping methods requires at least two spatial datasets: one containing the aggregated population data and one (or more) containing ancillary data used by the algorithm to disaggregate population data. Original coarse-level population data are usually vector-based census data representing a set of administrative units (Yang *et al.* 2013, Briggs *et al.* 2007). Ancillary data are usually raster-based datasets resulting from the pre-processing of remote sensing images Wu *et al.* (2005). Most studies use pre-classified land cover data (Reibel and Agrawal 2007, Su *et al.* 2010). Various kinds of data can be used, including unprocessed spectral bands (Li and Weng 2005), land surface temperature data (Li and Weng 2010), elevation data (Su *et al.* 2010), night light emission images (Yang *et al.* 2013), vegetation indexes (Li and Weng 2005), etc. Some studies have used additional vector data as ancillary data such as road networks (Su *et al.* 2010).

2.0.2. Explicit localization methods (ELM)

AIM can help to go deeper into population spatial allocation but is not in any way a method to provide a location for individuals. It targets the population as a whole or identifiable subgroups, whereas agent-based simulation requires each individual agent to be localized. To go even further, ELM propose to localize entities across space, based on the most precise population density data which can be built from the AIM process or derived from raw observation data. If no information at all is available, the challenge is to locate agents in the world by following some specific rules defined by the modelers.

There is no methodology to explicitly determine the localization of agents in a synthetic population: one must rely on *rule of thumb* algorithms. In some cases, the localization of agents is just randomly chosen (Wheaton *et al.* 2009). In other cases, roads (Gallagher *et al.* 2017) or buildings (Zhu and Ferreira 2014, Fosset *et al.* 2016) could be used with different spatial allocation techniques.

If most of existing works rely on a ad hoc approach to carry out this localization process, some of them propose a generic process that can be adapted to different case studies and data. Amongst them, SPEW proposes three different localization methods based on the same general approach: use a given probability distribution over a set of geographical objects (regions, roads...) to sample agents according to this distribution. The difference between these three methods comes from the used probability distribution and the type of geographical objects:

- uniformly across a polygon (region)
- uniformly along a polyline (road)
- according to a given spatial distribution of agent locations (weighted polygons or lines).

In addition, as it is sometimes necessary to link agents with spatial objects with which the agents may interact (schools, workplaces...), SPEW proposes a simple method based on a gravity model to assign agents to geographical objects.

The methods proposed by SPEW are adapted when few data are available (just a shapefile of regions or roads), but cannot take benefit of many possible sources of data that can improve the localization process such as density map, expert knowledge or satellite imagery. In the next section we present our generic proposal to localize a synthetic population using the Gen* toolkit.

3. The Gen* toolkit

3.1. *General architecture*

The Gen* library consists of four modules, each one responsible for a portion of the population synthesis: `gospl` is responsible for the population generation, `spll` make it possible to localize the synthetic entities of the generated population and, finally, `spin` allows us to connect entities to setup a population network. While modules are independent, they all rely on the `core` module which defines the higher-order abstraction for populations, entities, their attributes and associated values. Hence, a synthetic population is a collection of entities made up of identical attributes but each entity has its own vector of values. Thanks to a common reference to this higher order population definition, the API architecture provides a way to localize or connect, in any order, entities of any synthetic population that comply with the interfaces define in `core`. In the next subsection, we detail the content of the module that is responsible for population localization.

3.2. *Spll: Synthetic population localization library*

3.2.1. General scope

`Spll` make it possible to spatialize a synthetic population defined by `Gospl` (generated with the algorithms provided by `Gospl` or directly loaded from a csv/xlsx file). The configuration of the population localization is defined through a configuration file that harmonizes the input data. These data should at least contain a geographic file that specifies the spatial objects on which `Spll` will locate individuals. It can also contain other data such as matching data to provide a link between individuals of the population and geographic objects, for example the number of people per administrative area. The user can provide other data that can help the localization process, like land use raster or road vector file. In addition to the localization of individual, `Spll` make it possible to link each agent to a set of geographical objects (workplace, schools...).

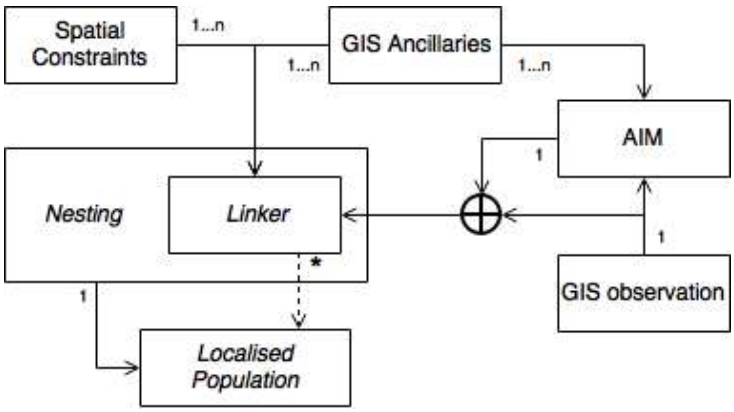

Figure 1.: The `Spll` process to localize a synthetic population. It starts at the right with one input data (GIS observation) or optional estimated data (AIM) file to feed the main step: the *nesting* process. It grants individual entities a unique location based on a spatial distribution function and spatial constraints encapsulated in a dedicated *Linker*. An unlimited number of *Linkers* can be setup to optionally bound synthetic individual with spatial objects (dashed arrow)

Figure 1 describes the conceptual process underlying `Spll`. As illustrated in the diagram, localized populations is achieved through the nesting procedure (center left). This consists of linking each individual with a *nest*, i.e. a spatial object in which the individual will be located, and by defining a location in this nest. We chose to locate individuals in parameterized spatial objects in order for each user to define the most suitable location according to the model needs: the nests could be buildings, cells of a raster file or even just x and y coordinates. In order to help the nesting process, `Spll` can optionally generate a contingency map or a density map from different sources of data, either through raw input data – referred to here as GIS observation at the bottom right – or AIM based generated data at the center right – using for example linear regression. The main nesting phase is take in charge by a linker which encapsulate a spatial distribution and spatial constraints. Those last are constraints based on GIS data: for example, a maximum distance from which to localize individual along roads or within a given distance from a point of interest (POI). Spatial distribution is a function that assigns weights to each nest candidates, and can be setup with GIS data (GIS based observation) and rules to compute weights, e.g. uniformly or based on the area of spatial object. The linkers can

also be used to bind individual with spatial objects, for example chose a workplace or a school for an individual agent. Finally, `Spll` can export the spatialized population in a GIS file.

3.2.2. Spatializing a synthetic population with Spll

The general localization approach is designed in order to be as adaptable as possible to all case-studies. Thus, the only mandatory step is the specification of a geographical data file (vector or raster) to locate the agent inside – i.e. GIS observation. Each individual of the population will be located in a spatial object defined by this input file. For example, individuals can be assigned to buildings taken from a building shapefile. If no spatial object are defined, hence individual will be assigned to the world itself and will have a precise x and y location in it. `Spll` provides two ways to improve this input observation data: individuals matched with areal spaces and estimated disaggregated spatial distribution. In the next two sections, we show how to build those refined observation. Then, we illustrate how the nesting process works by introducing the spatial distribution function and spatial constraints. Finally, we describe the ability of `spll` to link individual with any spatial object using *linkers*.

3.2.2.1. Matching population and space. This optional step can help to distribute each individuals into custom spatial areas. It has two implications: to have a file that describes some spatial areas as input as well as a means to connect individuals with these areas. This allocation should yield a population count per spatial object or a key ID code for the spatial objects to match with individuals. Most of the time, this space will be administrative areas with either a density or an ID. However, it is not limited to this case, and can be a raster density file or any other GIS format file with a population count or pairwise ID. In this latter case, `Spll` is able to automatically match individuals and space given that both ID key attributes have been bound in the configuration file. As an advantage, this procedure allocates actual synthetic individuals, whereas density allocation only provides a population count. They both end up defining a number of people per spatial object of a space file and this record can then help for the nesting phase or for the interpolation process.

3.2.2.2. Areal interpolation. Based on raw observations provided by the previous matching step, `Spll` make it possible to further specify areal scale density or count using any interpolation technique such as regression. Like in the previous matching phase this interpolation step is optional. The core idea of areal interpolation is to estimate a more precise spatial distribution using two inputs: first, an observation to be explained, which will be the densities or the contingencies per area, and second, a set of explanatory data, which will be spatial objects of finer granularity linked to a numerical or categorical feature. Therefore, `Spll` provides several estimation techniques to explain observations based on provided spatial variables: the result of the algorithm takes the form of a formula and can then be applied to compute disaggregated areal population density. The current version of Gen* uses linear regression which is a standard estimation technique in dasymetric mapping (see section 2.0.1). It does well when coupled with pretreated land cover or land use data (Li and Weng 2010). On the contrary, when used together with raw satellite imagery, it performs poorly because numerical band values usually do not interact linearly with the population allocation (Stevens *et al.* 2015). To overcome this problem, we plan to implement other techniques such as the generalized non-linear

regression and other AI-based estimation technique (see section 5).

A major drawback is that the output of areal interpolation techniques is usually in the form of a floating and possibly negative number. The estimated value should therefore be normalized to fit the population count, i.e. a positive or null integer. Some computation procedures have been proposed in the literature to overcome these two issues (Yuan *et al.* 1997). First, we chose to upscale values to a given threshold, with a default value of 0 for the population. We then uniformly reduced other values to return to the original overall sum as a compensation process. Second, we used a heuristic algorithm to round off floating values in order to obtain a positive integer. The process iterates over all spatial objects reducing (truncating) or increasing (rounding off) associated floating values, while maintaining the absolute difference between the overall sum and the expected overall population count as close to zero as possible.

At the end, the areal interpolation gives an estimated areal count or density for each explanatory spatial object. Depending on observation provided by the matching step, each cell can refer to any individual or a specific set of individual bound to a more aggregated spatial area. In both cases, the resulting dasymetric map will be used as a spatial constraint in the linker used by the *nesting* process.

3.2.2.3. Nesting process. `Spll` provides a set of algorithms – i.e. nesting process – to bind spatial objects – i.e. nests – and individuals of the population. The process is driven by a linker that encapsulate optional constraints and a distribution function. It also must be initialize using any input data file that describes the *nests*. When an individual has been placed into his/her nest, a "location" inside the assigned spatial object is then computed. In the following paragraphs we detail how `Spll` link individuals to spatial objects and provides individuals with a particular location in this nest.

As a minimal requirement, the *nesting* process needs an input file that describes the spatial object in which to locate an individual. If this file is the only constraint given to the algorithm, then each individual of the population will randomly choose a nest among all the possible ones according to the distribution function. In order to obtain a more realistic spatialization of the individuals, the user can define spatial constraints to filter the possible nests. `Spll` integrates three basic types of spatial constraints: geometric, contingency and density constraints. The first one is used to bind the spatial objects – the geometries associated with these objects – and the individuals; while the two others rely on matching a predefined number or density of entity per nest. If the constraints do not make it possible to find an appropriate nest, they can be relaxed. More precisely, for each constraint, a relaxation process can be defined as well as a maximal relaxation function that defines whether the constraint can be relaxed yet or not.

Geometric constraint specifies that individuals have to choose a nest inside a given geometry. Typically, this constraint make it possible to link geographic data with one of the individual's attributes which is similar to the previous optional match and binding step. For instance, if we have information about the district where each individual is living (through a district attribute defined by `Gospl` for example), this constraint can be used to restrict the list of possible nests to those located inside the given district. If there are none inside the reference geometry, the relaxation will allows the constraint to also include the nests that are close to the reference geometry. The notion of closeness depends on a distance variable that will increase as the constraint is more and more relaxed until reaching a given user defined threshold. A geometric constraint could also be used if we want to specify that an individuals

location should be close to a roads or near a POI like in the SPEW toolkit (Gallagher *et al.* 2017).

Contingency constraint specifies the maximum number of individuals for one or more geometries. Typically, this constraint comes as a demographic count per geographic area. For instance, we can compute a contingency grid from diverse spatial layers that will specify the expected number of individuals per cell. This constraint will filter the possible nests to only keep the ones for which the max contingency constraint is not yet reached. If all nests are full, the relaxation will increase the capacity of the nest whereas the maximal increase is given by a user defined threshold.

Density constraint specifies the maximum density of individuals inside one or more geometries. This constraint works like the contingency one but with density rather than contingency. The relaxation process relies on identical mechanisms and with the same relaxation threshold variable.

Depending on the application, it can be more interesting to first try to relax the geometry constraint before the contingency one and vice-versa. To specify this notion of priority of relaxation, the user can define a priority for each constraint. An ordering process is also performed when searching for a proper nest in which to locate individual, for which each constraint will successively filter the list of possible spatial nests. When several nests satisfy all the constraint, Gen* uses a distribution function to draw one to bind individual with. We provide user with several predefined distribution functions:

Uniform distribution (default distribution function) specifies that all the nests has the same probability to be chosen. (Gallagher *et al.* 2017).

Area distribution specifies that the probability of a nest to be chosen is a linear function of its area.

Capacity distribution specifies that the probability of a nest to be chosen is a linear function of its capacity. The capacity of the nest is updated each time it has been drawn to be bound with an individual.

Note that if the list of predefined constraints or distribution functions does not cover the user needs, he/she can define its own new types of constraint by implementing a Java interface, and distribution function by overloading a Java method. For instance, if the user wants to locate the agents in buildings according to the income of the agents and the standard of buildings, he/she will just have to implement a new distribution function that defines the probability for an individual to chose a building according to its standard and the individual's income.

When each agent has been associated with a spatial object, the next procedure is to assign a precise x/y coordinate location inside it. `Spll` proposes two basic algorithms to do so: centroid and random. First, the *centroid* mechanism retrieves the centroid of the object. For polygons, it is computed as the weighted sum of the centroids of a decomposition of the area into triangles. For lines, the centroid computes the average of the midpoints of all line segments weighted by the segment length. Finally, for points, the centroid is the average coordinate for all points. Second, the *random* mechanism retrieves a random point in the object. For a polygon, the algorithms consists in selecting random points in the object envelope until one of the point overlaps the object. For a line, a

random value is retrieved between 0 and the perimeter of the line. The point corresponds to the one at this length from the first point following the line. Finally, in the case of points, one of them is chosen at random. At the end of the localization process, each individual from the synthetic population should have a *nest* – i.e. a spatial object – and a precise x/y coordinates inside it. They can be accessed using attribute accessors in the spatialized population model or exported as a GIS formated file.

3.2.2.4. Geographical object linking. Similarly to SPEW, Gen* makes it possible to link a set of geographical objects to each individual. This linking process is very close to the localization process (in fact, the nesting step is a spatial case of linking): the principle is to choose for each linking function (e.g. workplace, school), a geographical object among a set of possible ones to link individual with. To do that, `Spll` uses a *linker*: that is a combination of spatial constraints and a spatial distribution function. However, as it is often necessary to link individuals to close geographical objects, `Spll` offers the possibility to define a distribution according to the distance between the agent location and the geographical objects. The toolbox provide several predefined distribution function to be readily used:

Distance-based distribution specifies that the probability of a geographical object to be chosen by an individual is a negative linear function of the distance between them.

Gravity model specifies that the probability of a geographical object to be chosen by an individual depends on the mass of the geographical object divided by the distance between them. By default, the distance corresponds to the euclidean distance between the individual and the geographical distance. Two basic functions are proposed for the mass: the sum of distances between the geographical object and all individuals, and the number of individuals within a given buffer around the geographical object. This second function with a buffer of 0.0 corresponds to the one used by SPEW.

All previously detailed spatial distribution functions and constraints can be re-used to drive the linking process. For example, the user can link individual to places based on spatial object area or capacity, while defining a limited geographical zone as a constraints to filter possible spatial objects to bind individual with.

3.2.2.5. Discussion. `Spll` proposes a flexible tool to localize a population based on the definition of spatial constraints and distribution functions. To compare with SPEW, `Spll` can do everything SPEW does, but allows to go further. Indeed, the possibility to define spatial constraints allows to integrate new knowledge in the spatialization process. In addition, `Spll` allows to use Areal interpolation techniques to use information coming from different sources to improve the spatialization. At last, the genericity of the proposed spatialization approach (independent of the type of geographical object concerned) allows to easily extend it through the definition of new spatial constraints and distribution functions. It makes it possible to adapt the process to available input data – the only mandatory piece of data is the space itself – and to adjust the output to specific needs of modelers – spatial precision as well as custom distribution function and spatial constraints.

4. Case study: the city of Rouen, France

The Gen* library comes with documented examples, that are all freely available on the Web (see closure section that provides useful links). In this section we present one of this example in detail: the population localization of the city of Rouen in France.

4.1. *Context of the work*

Rouen, which is the capital of the Normandy region in France, is a city of 110,755 inhabitants built on both sides of the Seine River. Many chemical industries are located near its city center which makes a potential accident particularly problematic. In order to be able to evaluate the impact of an accident and the possible evacuation scenario that could be implemented, a model was produced (Czura *et al.* 2015). In this model, the properties and the localization of the inhabitants are particularly important. In the following sections, we will review several options that have been proposed for the model developed in (Czura *et al.* 2015).

4.2. *Population spatialization*

In this example, we consider that we have generated a population with Gen* from the data available from the French National Institute of Statistics (INSEE) that cover a wide number of demographic and social characteristics. The location is given as an aggregated count of people per IRIS code, the main and most precise statistical areal unit in France. We consider three scenarios corresponding to different types of data available:

- localization inside the IRIS geometries (polygons)
- localization along the roads (polylines)
- localization inside buildings, with information about the IRIS geometries, satellite image and spatial constraints.

4.3. *Scenario 1: localization inside the IRIS geometries*

As a preliminary step, we defined a configuration file that specifies how to spatialize the population. In particular, we specify the geographical data that will be used: the vector files for IRIS geometries. The localization process started using the most precise spatial demographic data: the population of each IRIS in the city, depicted in Figure 2.
Based on this information we can extrapolate a precise location for people inside these areas. To do so, Gen* matches people and IRIS. The matching process is based on the key attribute of home location which is part of the generation process: each individual has an IRIS ID that allows him/her to be located in the city. Once this match is made, we know the list of generated entities living inside each IRIS area. Once the list is defined, we let Gen* give a precise location to each individual of the population inside the corresponding IRIS geometry. In this simple example, we use the random location assignment algorithm to do so. Figure 3 shows an example of result obtained.
The overall localization process took less than 3 sec. to be accomplished. This include matching, assigning and finding a location within IRIS for each of the 110 688 individuals of the city, which is totally acceptable.

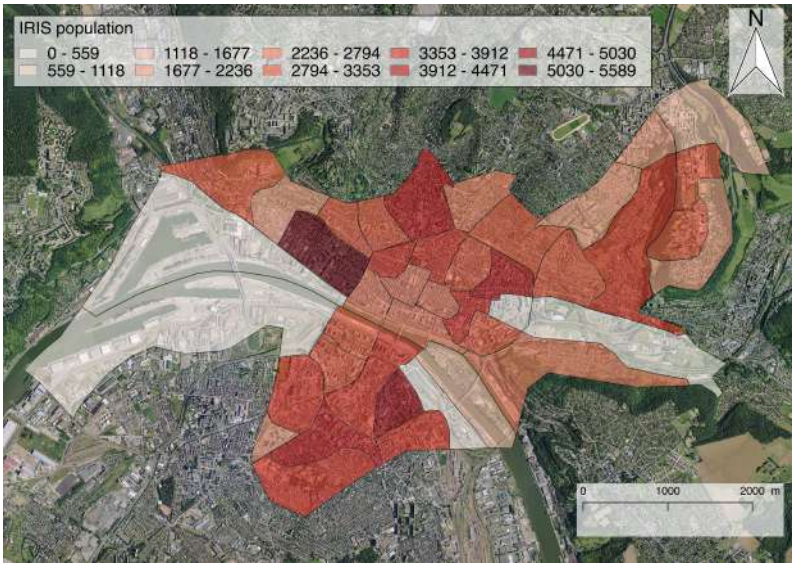

Figure 2.: Geometries of IRIS in the city of Rouen

Figure 3.: Scenario 1: precise x and y coordinate population location for the city of Rouen

4.4. *Scenario 2: localization along the roads*

Like for the previous scenario, as a preliminary step, we defined a configuration file that specifies that we want to use a road shapefile for the spatialization. However, as in this scenario individuals have to be located along the roads and not directly on them, we use as a first step a built-in function of Gen* that defines proxy spatial object to locate individual in: it allows to compute for each geographical object of a shapefile a new geometry that corresponds to the area that is between a minimal and a maximal distance to the objects. For this example, the individuals are located at distance between 2.0 and 10.0 meters of the roads. Once this geometry have been defined for all the roads, we just applied the normal localization process of Gen* with the spatial distribution

function based on area. This allow us to favor the selection of long roads rather than short one for the nest selection.

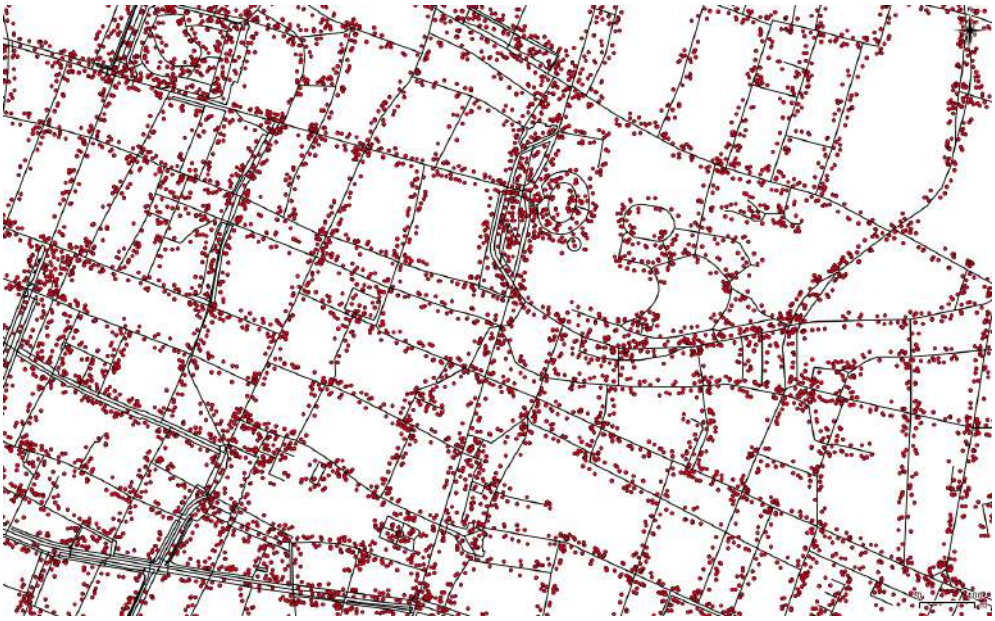

Figure 4.: Scenario 2: precise x and y coordinate population location for a small part of the city of Rouen

Figure 4 shows a snapshot of the result we obtain using road based nesting for the city of Rouen. The overall computation time was 99 sec., with 87 sec. dedicated to the computation of the area around the 4700 roads.

4.5. *Scenario 3: localization inside buildings with spatial interpolation*

In this last scenario, we specified in the configuration file that three sources of geographical data will be used: land use pre-processed satellite imagery, vector files for IRIS and building geometries. The first one was performed using raw satellite imagery from Landsat 8, with a raster cell of $30m^2$. We used a well known clustering algorithm known as "isodata" (Memarsadeghi *et al.* 2007) to compute height different categories of land use: very high, high and low density buildings, high and low green space, water, road and bare land space.

As for the first scenario, the localization process started using the IRIS shapefile to determine the list of generated entities living inside each IRIS area. Then, this spatial distribution is used as the objective of an interpolation process and the land use category of the raster file described above as explanatory variables. Spatial interpolation terminates with a function that allows Gen* to determine a desegregated population distribution within each IRIS. More precisely, the algorithm results in a number of people for each $30m^2$ raster cell used as an explanatory variable. While aggregated, these computed contingencies fit the known population count at the IRIS level. Map 5 gives an overview of the resulting contingencies for the valid cells (excluding no-data and null values for water), with lighter cells standing for contingency equal to 1 individual (low

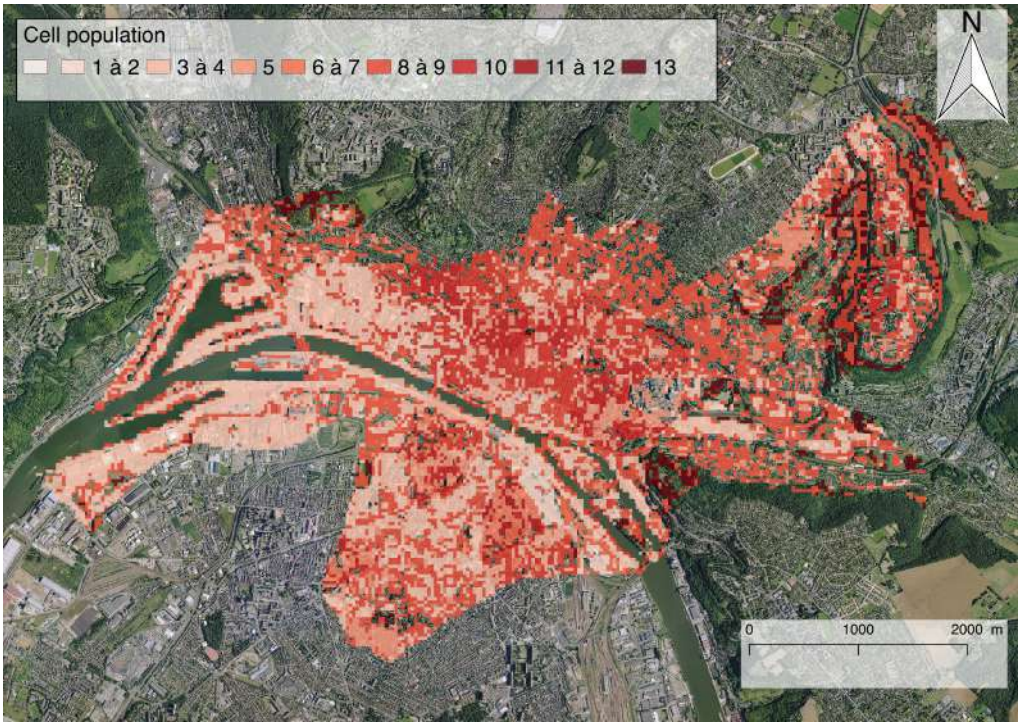

Figure 5.: Resulting densities of the spatial interpolation process

density) and dark red cells for contingency equals to 13 individuals (high density). The entities are randomly assigned to cells within IRIS with computed density constraint.

Figure 6.: Scenario 3: precise x and y coordinate population location for a small part of the city of Rouen

The next step consists into giving a precise location to each individual of the population: this is done using information on buildings. As we can see in Figure 6, thanks to

the input data about buildings, no individual has been placed into water, on bridges or on roads. The density constraint in $30m^2$ cells is released when no building is present within the defined spatial entity. In this case, the closest building is chosen to locate the individual. The location process also takes a limited number of people within the buildings: one cell can receive more than it is supposed to hold and, on the same level, one building can exceed the number of individuals expected to be within it if needed. These two constraints are released one after the other, to allow the location algorithm to always find a solution with alternate concessions on density requirements. To sum up, the localization process is composed of three steps: the matching between entities and IRIS, the spatial interpolation to compute density within raster cells and, finally, the localization of each entity within the buildings. The entire process takes less than 25 sec. to be accomplished.

4.6. *Geographical object linking*

As introduced in the previous step, Gen* allows to link geographical objects to individuals. The nesting process is in itself an example of linking, with a specific status assigned to the selected spatial object. The use of linkers make it possible to bind any individual of the synthetic population with an unlimited number of spatial object. As a major difference with the nesting process, the created link between individual and spatial object will not determine the individual location. However, it can drive dynamic localization process during simulation, like to move to linked school or workplace from current location. For this example, we assign a school to each children of the population (here, individuals with an age lower than 15). We used the gravity model for the linking process. Mass of spatial object candidates (i.e. the schools) is defined as the sum of distances between the geographical object and all individuals.

Figure 7.: Linking between children and schools (similar color): the circle represents the children and the buildings with red border the schools

Figure 7 shows the results obtained: we clearly see in this map the impact of the

distance for the school assignment. The linking process favors to link children with closest schools while also taking into account the number of children within the school. The linking process took less than 0.6 sec. for the 10 501 children.

5. Conclusion

This article presents the localization component of the Gen* toolkit. It allows to localize a population, i.e. gives to each entity of the population a spatial nest and a location, through the definition of spatial constraints and a distribution function. In addition, Gen* provides tools to increase spatial resolution of the population distribution through estimation technique, as well as methods to link geographical several spatial objects to individuals of the synthetic population. As shown by the case-study, Gen* make it possible to build a credible spatialized population with many options and within a respectable computation time.

In order to validate the genericity of Gen* and its adaptability to all contexts, we plan to carry out other case-studies in the near future: the generation of the population of Bangkok, of Marrakesh and of several cities in France and Vietnam. These case-studies will allow us to evaluate the capacity of Gen* to deal with different population scales (from hundreds of thousands to several millions) and different data sources (both for census data and GIS data).

In terms of development, we plan to add many new features to Gen* and in particular to Spll. A first one concerns the addition of new regression techniques like random forest, which seems very promising for complex localization problem and with raw input satellite imagery (Stevens *et al.* 2015). We plan as well to add new types of spatial constraints and distribution functions to improve the flexibility of Spll. In addition, we want to improve the performance of Gen* in terms of computation time by allowing to take benefit of parallel computation. Finally, we plan to continue to work on facilitating the use of Gen* by non-computer scientists by improving the GAMA plugin and the Kepler version of Gen*.

To conclude, Gen* is still an active project. Its goal is not only to develop a library to be used by modelers, but to create a community concerned with the synthetic population generation as well.

Acknowledgment

This work is part of the GEN* ("Generation of Synthetic populations") research project funded by the French National Research Agency.

Website references

The Gen* library suite (API, templates and GAMA plugin) could be find on Github [1] while the GAMA platform has its own website.[2]

[1] https://github.com/ANRGenstar
[2] http://gama-platform.org

References

Anderson, W., *et al.*, 2014. Methods for Estimating Population Density in Data-Limited Areas: Evaluating Regression and Tree-Based Models in Peru. *PLoS ONE*, 9 (7).

Antoni, J.P., Lunardi, N., and Vuidel, G., 2016. Simuler les mobilits individuelles - Les enjeux de linformation gographique. *Revue Internationale de Gomatique*, 26 (2), 237–262.

Arentze, T., Timmermans, H., and Hofman, F., 2008. Creating synthetic household populations: problems and approach. *Transportation Research Record: Journal of the Transportation Research Board*.

Axelrod, R.M., 1997. *The complexity of cooperation: Agent-based models of competition and collaboration*. Princeton University Press.

Barthelemy, J. and Toint, P.L., 2012. Synthetic Population Generation Without a Sample. *Transportation Science*, 47 (2), 266–279.

Bhaduri, B., *et al.*, 2007. LandScan USA: a high-resolution geospatial and temporal modeling approach for population distribution and dynamics. *GeoJournal*, 69 (1-2), 103–117.

Bracken, I. and Martin, D., 1989. The generation of spatial population distributions from census centroid data. *Environment and Planning A*, 21 (4), 537–543.

Briggs, D.J., *et al.*, 2007. Dasymetric modelling of small-area population distribution using land cover and light emissions data. *Remote sensing of Environment*, 108 (4), 451–466.

Cornelis, E., *et al.*, 2013. An original synthetic population tool applied to Belgian case: VirtualBelgium. .

Czura, G., *et al.*, 2015. MOSAIIC: City-Level Agent-Based Traffic Simulation Adapted to Emergency Situations. *In*: *Proceedings of the International Conference on Social Modeling and Simulation, plus Econophysics Colloquium 2014*, 265–274.

Edmonds, B. and Moss, S., 2005. From KISS to KIDS in Multi–Agent and Multi–Agent Based Simulation. *Lecture Notes in Computer Science*, 3415, 130–144.

Eicher, C.L. and Brewer, C.A., 2001. Dasymetric mapping and areal interpolation: Implementation and evaluation. *Cartography and Geographic Information Science*, 28 (2), 125–138.

Fosset, P., *et al.*, 2016. Exploring intra-urban accessibility and impacts of pollution policies with an agent-based simulation platform: GaMiroD. *Systems*, 4 (1), 5.

Gallagher, S., *et al.*, 2017. SPEW: Synthetic Populations and Ecosystems of the World. *arXiv:1701.02383 [physics, q-bio, stat]* ArXiv: 1701.02383.

Grignard, A., *et al.*, 2013. *In*: *GAMA 1.6: Advancing the Art of Complex Agent-Based Modeling and Simulation.*, 117–131 Berlin, Heidelberg: Springer Berlin Heidelberg.

Harland, K., *et al.*, 2012. Creating realistic synthetic populations at varying spatial scales: a comparative critique of population synthesis techniques. *Journal of Artifical Societies and Social Simulation*, 15 (1), 1–15.

Holm, E., 2002. *The SVERIGE spatial microsimulation model: content, validation, and example applications*. Department of Social and Economic Geography, Univ.

Kosar, B. and Tomintz, M., 2015. simSALUD: A Web-based Spatial Microsimulation to Model the Health Status for Small Areas Using the Example of Smokers in Austria. *In*: Austrian Academy of Sciences Press, 207–216.

Li, G. and Weng, Q., 2005. Using Landsat ETM+ imagery to measure population density in Indianapolis, Indiana, USA. *Photogrammetric Engineering & Remote Sensing*, 71 (8), 947–958.

Li, G. and Weng, Q., 2010. Fine-scale population estimation: How Landsat ETM+ imagery can improve population distribution mapping. *Canadian Journal of Remote Sensing*, 36 (3), 155–165.

Memarsadeghi, N., *et al.*, 2007. A fast implementation of the ISODATA clustering algorithm. *International Journal of Computational Geometry & Applications*, 17 (01), 71–103.

Reibel, M. and Agrawal, A., 2007. Areal interpolation of population counts using preclassified land cover data. *Population Research and Policy Review*, 26 (5-6), 619–633.

Stevens, F.R., *et al.*, 2015. Disaggregating census data for population mapping using random forests with remotely-sensed and ancillary data. *PloS one*, 10 (2), e0107042.

Su, M.D., *et al.*, 2010. Multi-layer multi-class dasymetric mapping to estimate population distribution. *Science of the Total Environment*, 408 (20), 4807–4816.

Swarup, S. and Marathe, M.V., 2016. Generating Synthetic Populations for Social Modeling: Tutorial at the Autonomous Agents and Multi-Agents Systems (AAMAS) Conference. *In*: May., Singapore.

Wheaton, W., *et al.*, 2009. *Synthesized population databases: A US geospatial database for agent-based models.* Technical report, RTI Press, Research Triangle Park, NC.

Wu, S.s., Qiu, X., and Wang, L., 2005. Population estimation methods in GIS and remote sensing: a review. *GIScience & Remote Sensing*, 42 (1), 80–96.

Yang, X., Yue, W., and Gao, D., 2013. Spatial improvement of human population distribution based on multi-sensor remote-sensing data: an input for exposure assessment. *International journal of remote sensing*, 34 (15), 5569–5583.

Yuan, Y., Smith, R.M., and Limp, W.F., 1997. Remodeling census population with spatial information from Landsat TM imagery. *Computers, Environment and Urban Systems*, 21 (3-4), 245–258.

Zhu, Y. and Ferreira, J., 2014. Synthetic Population Generation at Disaggregated Spatial Scales for Land Use and Transportation Microsimulation. *Transportation Research Record: Journal of the Transportation Research Board*, 2429, 168–177.