synthACS: Spatial MicroSimulation Modeling with Synthetic American Community Survey Data

Alex Whitworth

Abstract

synthACS is an R package that provides flexible tools for building synthetic micro-datasets based on American Community Survey (ACS) base tables; methods for data-extensibility; and conducts spatial microsimulation modeling (SMSM) via simulated annealing. To the author's knowledge, it is the first R package to provide broadly applicable tools for SMSM with ACS data as well as the first SMSM that uses unequal probability sampling in the simulated annealing algorithm. In this paper, the author contextualizes these developments within the SMSM literature, provides a hands-on user-guide to **synthACS**, presents a case study of SMSM related to population dymanics, and notes areas for future research.

Keywords: R, SMSM, ACS, spatial microsimulation modeling, simulation, synthetic data, simulated annealing.

1. Introduction

Survey data helps local officials, community leaders, elected representatives, and researchers understand the changes taking place in their communities. Survey data is also used to determine the equitable distribution of public funds and enables the design of public policy. In the United States, the American Community Survey (ACS), which provides detailed information on forty-six topics, is one of the premier sources of survey information on how Americans live. It is therefore very important that the research community and community officials have flexible and easy-to-use tools for accessing, manipulating, and using this data.

When examining survey data, it is important to keep two principles in mind. Firstly, researchers should contextualize data spatially. Spatial characteristics of a population provide a more comprehensive understanding of differences and inequities in patterns, trends, and other impacts of community change. Secondly, community behavior is more realistically modeled by examining individual behavior individuals' interaction than by looking at aggregated data directly. These two principles form the basic premise of spatial microsimulation modeling (SMSM).

Despite the tremendous value of using SMSMs, creating them is a demanding exercise: it is both computationally intensive and requires specialized knowledge of programming simulation models. A general solution enabling researchers to easily create microsimulation models with high quality, easily accessible data–such as the ACS–is currently lacking. *Spatial Microsimulation: A Reference Guide for Users* Tanton and Edwards (2013), which provides an SMSM blueprint, provides a partial solution. However, there is still a shortage of general

software solutions for conducting SMSM.

In this paper, we introduces **synthACS**, a package which takes advantage of the availability of reliable statistics on small area populations from the ACS to enable SMSM at any desired United States geographic level. **synthACS** is thus, to out knowledge, the first R package which provides a general SMSM toolkit for any user-specified United States geography. **synthACS** is also data-extensible. It provides flexible tools for users to add additional data elements, both from the ACS and otherwise, to their micro data thereby enabling more targeted research questions.

The remainder of the paper proceeds as follows: Section 2 explains the methodology more clearly; Section 3 provides a hands-on user guide to the software; Section 4 provides a case study utilizing **synthACS** and discusses areas for future research; and Section 5 concludes.

2. Research background and methodology

Spatial microsimulation fits within a broader literature concerned with spatial analysis. Perhaps the best known subset of this literature concerns Geographical Information Systems (GIS). Existing software for GIS within R are discussed by Bivand, Pebesma, and Gomez-Rubio (2008) and enabled by the associated packages **sp** Pebesma and Bivand (2005) and **maptools** Bivand and Lewin-Koh (2016). **UScensus2010** Almquist (2010) provides an extension to this framework with a suite of R packages enabling easy access to the US Decennial Census. Similarly, the **acs** package Glenn (2016) provides easy access to the American Community Survey. In addition to the above software for accessing, manipulating, and mapping spatial data, **UrbanSim** Waddell *et al.* (2003) provides a modeling system designed to support metropolitan land use and transportation planning.

Methods for spatial microsimulation were first described by Orcutt *et al.* (1961); although it was not until 1984 that one of the first generally accepted SMSM was developed Clarke *et al.* (1984). While the general procedure for conducting SMSM has several steps (see Tanton and Edwards (2013)), the process can be broadly categorized into two stages: collecting, or synthetically generating, micro data; and fitting this micro data to the macro, or aggregate, data constraining tables.[1] The only open source spatial microsimiulation software that we is aware of is **sms** Kavroudakis (2015). **sms** includes tools for conducting SMSM with user-provided micro and macro datasets. **sms** performs optimization between the micro and macro data using either hill-climbing or simulated annealing algorithms.

Unlike **sms**, **synthACS** does not require external micro and macro datasets. Instead, it leverages the high quality data available from the ACS and acessible in R via **acs**. **synthACS** also provides methods for data-extensibility so that users are not limited to ACS data. But **synthACS** is explicitly targeted at users interested in modeling and analyzing United States geographic data. By focusing on a single geographic region and choosing a single primary data source, **synthACS**'s aims to 'do one thing well.' **synthACS** provides a novel method of synthetic micro data creation, including thoughtful tradeoffs around data availability (see Section 2.2.1), that enables end-users to focus on their substantive research questions instead of collecting survey data.

synthACS also has two computational improvements over **sms**. Firstly, **synthACS** makes

[1] From this point forward, we will use the term 'micro data' to indicate the sample of low level data on individuals and the term 'macro data' to indicate the aggregate constraining tables for each small area.

full use of a multicore architecture compared to **sms**, which only uses half of the available computing cores. Secondly, **synthACS** provides a more flexible simulated annealing algorithm than **sms**, which is detailed in Section 2.2.3. In the remainder of this section, we review the methological background of SMSM and describes the design of **synthACS** in detail.

2.1. Spatial microsimulation literature

The process of spatial microsimulation can be broadly divided into two stages, one concerning the collection of micro data and the other concerning fitting the micro data to the macro data. Researchers have approached this problem from multiple angles in the literature, including: conditional probabilities (Birkin and Clarke (1988); Clarke and Holm (1987)), deterministic reweighting (Harding *et al.* (2003); Ballas *et al.* (2005b)), combinatorial optimization Clarke *et al.* (1997), and simulated annealing Williamson, Birkin, and Rees (1998).

The conditional probabilities method was first introduced by Birkin and Clarke (1988). The method creates a synthetic micro dataset from the probabilities implied by the macro data. Attributes–such as gender, age, educational status, and race–are iteratively assigned for each individual using a stochastic process. The decision criteria governing the stochastic process for each individual are based on the macro data summary information. The method is considered conditional because the decision criteria for each attribute after the first is conditioned on previously assigned attributes. For instance, if we first assign an individual's 'gender', the 'age' attribute would be assigned based on the conditional distribution of ages within the 'Male' and 'Female' populations respectively.

Deterministic reweighting was first introduced by Ballas *et al.* (2005b). Unlike conditional probability models, which build a synthetic micro population, deterministic reweighting requires a sample micro population of individual records. The procedure then proportionally reweights each individual record in the sample based on the macro data. For example, if the researcher has a 1% sample of individuals, a crude reweighting scheme would be to simply duplicate each record one hundred times. As with the conditional probabilities method, attributes are reweighted iteratively, for instance, proceeding from 'gender' to 'age' as discussed above.

Another approach the researcher may take is to view the sample micro population as a subsample of a larger candidate population from which he or she draws samples for each small area (eg. census tract). Characterized in this manner, the problem can be thought of as finding the best combination of individual records that fit the macro data. This characterization is thus solved via algorithms for combinatorial optimization. Williamson *et al.* (1998) look at several strategies for combinatorial optimization: hill climbing, simulated annealing, and genetic algorithms. Taking consideration of computing cost, they find that their modified simulated annealing algorithm (SA2), which takes on some aspects of hill climbing, "emerges as the single best solution."

Each of the above methods has advantages and disadvantages as shown in Table 1 and further discussed in Harland *et al.* (2012). For both conditional probability and deterministic reweighting methods, the order in which attributes are introduced is important. Best results are achieved by starting with low entropy attributes, such as gender and age, and then proceeding to higher entropy attributes such as education level, income, and race. An additional concern with deterministic reweighting and current combinatorial optimization methods, since each requires a sample micro population, is the representativeness of the sample micro pop-

	Conditional probabilities	Deterministic reweighting	Simulated annealing
Sensitive to constraint order	Yes	Yes	No
Limited number of constraints	Yes	Yes	No
Requires a sample population	No	Yes	Yes
Can avoid local minima	No	No	Yes
Stochastic	Yes	No	Yes

Table 1: Summary comparison of algorithms for synthetic population construction; from Harland *et al.* (2012).

ulation to each small area under study. Error is minimized by only using sample populations which are relatively homogenous to the small area in question. This may be problematic when studying a large number of heterogenous small areas. As Clarke and Holm (1987) note, "in the worst but perhaps most typical case, no appropriate micro-data will be available." In this case, the only solution is to either conduct a large, and therefore expensive, survey or to accept poor quality micro data fits.

2.2. Design of synthACS

Synthetic micro data

As with other SMSM methods, the methodology in **synthACS** can be broadly divided into two stages: data generation at the individual level and optimization between the micro and macro data. But **synthACS** creates synthetic micro data based on the ACS instead of conducting an individual level survey. Therefore, one key contribution of **synthACS** is providing a fitting via simulated annealing without requiring a micro dataset.

To do so, **synthACS** borrows ideas from both conditional probability and reweighting methods in synthetic data generation. Specifically, we use the macro data to create synthetic individuals[2] and assigns each a probability of being selected during the fitting procedure via unequal probability sampling Rao *et al.* (1962). As with the conditional probabilities method, data attributes are added iteratively and are conditioned on prior attributes. However, fitting is not conducted at this stage; instead, we only reweight individual sampling probabilities when each new attribute is added. **synthACS** uses attributes which are widely available in the ACS base tables; and, attribute conditioning is influenced by data availability in the ACS base tables. These attributes are described in Table 2 below.

Note that the order in which these attributes are added implies an indirect conditioning between attributes. For example, while income is only explicitly conditioned on poverty and nativity statuses, nativity status is itself conditioned on age, and poverty is conditioned on both gender and employment. Income is therefore also implicitly conditioned on age, gender, and employment status. The same logic can be applied to, for instance, geographic mobility. Since simulated annealing is not conducted at this stage in **synthACS**, the order of attribute creation does not impact constraints; however, it does impact the implicit conditioning of data attributes. Therefore, the order in which data attributes are added does matter; and, attribute order was carefully chosen.

[2] **synthACS** only uses individual data, not households.

Data attribute	Conditioning attribute(s)	ACS base table(s) used
1. Age and Gender	Jointly Distributed	B01001
2. Marital Status	Age and Gender	B12002
3. Educational Attainment	Age and Gender	B15001
4. Employment Status	Age and Gender	B23001
5. Nativity Status	Age	B06001
6. Poverty Status	Gender and Employment Status	B17005
7. Geographic Mobility	Educational Attainment	B07009
8. Income	Poverty Status and Nativity Status	B06010
9. Race	Gender	B01001B-I, B02001

Table 2: Attribute order and conditioning for synthetic micro data creation in **synthACS**.

In general, the detail of the ACS base tables is tremendously helpful. But ACS base tables do not always define their populations and data attributes in exactly the same manner. A lack of perfect match between micro and macro datasets has long bedeviled SMSM researchers and **synthACS** is no different. **synthACS** makes the following assumptions:

- Marital status: Table B12002 reports on the population age 15 and over. All individuals under age 15 are assumed to have never been married.
- Educational attainment: Table B15001 reports on the population age 18 and over. Individuals under age 15 are assumed to have 'less than high school' attainment and individuals aged 15-17 are assumed to have 'some high school' attainment.
- Employment status: Table B23001 reports on the population age 16 and over. Individuals under age 15 are assumed to not be in the labor force. Individuals aged 15-17 are assumed to be represented by reported employment status of those aged 16-19. Individuals aged 18-24 are assumed to be represented by reported employment status of those aged 20-24.
- Poverty status: Table B17005 reports on the civilian population 16 years and over for whom poverty status is determined. Poverty status is assumed to be independent of age conditional on employment status. Since the poverty rate for those not in the labor force is roughly twice that of those in the labor force (21.8% vs 9.4%) US Census Bureau (2015), this likely slighly overststates childhood poverty.
- Geographic mobility: Table B07009 reports on the population 25 years. Geographic mobility is assumed to be independent of age conditional on educational attainment. Additionally, Table B07009 does not have the same education categories as B15001. 'Some high school' in B15001 is mapped to 'less than high school' in B07009, while 'associate degree' in B15001 is mapped to 'some college' in B07009.
- Income: Table B06010 reports on the population age 15 and over. Income is assumed to be independent of age conditional on poverty status and nativity status.

Of all these assumptions, we find those related to individual income most troubling. In addition to the implicit conditioning described above, the ACS reports top-coded individual

incomes where the maximum top coded value is "$75,000 and up". Researchers for whom income is a key variable in SMSM may find this unsatisfactory and may wish to add a different measure of individual income. A generic method for doing so will be discussed in Section 3.2.1. Another method is provided by Crimi and Eddy (2014).

Simulated annealing

Once the synthetic micro data is created, researchers can proceed to problem of selecting the set of individual records which best fit the macro data. The loss function chosen to optimize is the total absolute error (TAE) defined

$$TAE = \sum_{j}^{D} \sum_{i}^{n_j} |S_{ij} - E_{ij}| \tag{2.2.1}$$

where S_{ij} and E_{ij} are the observed (ie. simulated) and expected (ie. census) counts for attribute $j; j = 1, \ldots, D$ taking on category $i; i = 1, \ldots, n_j$.

TAE has the advantage of simplicity of calculation and interpretation, but we also note two disadvantages. Firstly, TAE double counts misclassification, which is easily illustrated using the case of one attribute with two categories. Given ten observations which are all expected to be in category one but are misclassified as category two, the TAE is $|10-0|+|0-10| = 20$. Secondly, we cannot calculate the gradient of TAE for any attributes. The method chosen for optimization in **synthACS** is therefore simulated annealing (Metropolis *et al.* (1953); Szu and Hartley (1987); Ingber (1989)). As a heuristic method, simulated annealing is designed to produce a good solution, although there is no guarantee that the solution is optimal. We describe the typical simulated annealing algorithm below.

Suppose that the solution space S is the finite set of all possible solutions and the loss function f is a real-valued function defined on the members of S. The goal is to find a solution $i \in S$ that minimizes f over S. Simulated annealing starts with a random solution $i_0 \in S$ and computes the loss of this solution $f(i_0)$. For each iteration, generate a candidate solution $j \in S$, compute $f(j)$ and compute the change in loss $\gamma = f(j) - f(i)$. If the candidate solution is better than the current solution ($\gamma < 0$), always accept the candidate solution. Otherwise, accept the candidate solution with acceptance probability proportional to γ and controlled by a control parameter T, which is typically termed the temperature.

Simulated annealing therefore always accepts downhill moves but also sometimes accepts uphill moves. Small uphill moves are more likely to be accepted than large ones; and, when T is high, most moves will be accepted, while when T is low, most uphill moves will be rejected. The value of T therefore controls the tradeoff between descent and avoiding local minima, while decay in T, commonly termed the cooling schedule, controls the rate at which simulated annealing favors descent relative to the possibility of being trapped in a local minimum.

In **synthACS**, the simulated annealing algorithm works to optimize a loss function across D attributes, yielding a D-dimensional parameter space. Note that each parameter may have a different number of categories and that TAE sensitivity may vary across parameters. For simplicity, there is no consideration for sensitivities between parameters that might require different cooling schedules. Instead, the cooling schedule chosen in **synthACS** is a simplified

version of that proposed by Ingber (1989):

$$T_k = T_0 \exp(\frac{-1}{20}\frac{k}{D}) \tag{2.2.2}$$

where k is the iteration and T_0 is the initial temperature.

Uphill moves are accepted with probability proportional to the relative loss between iterations k and $k-1$ conditional on the temperature at iteration k

$$\mathsf{P}(\text{accept}|T_k) \propto \frac{f(i_k)}{f(i_{k-1})} \tag{2.2.3}$$

.

Options in simulated annealing

It is important to carefully consider tradeoffs between accuracy of fit and computation time of the simulated annealing algorithm. In **synthACS**, we considered three standard options. We firstly examined the initial acceptance probability and the number of allowable iterations by examining final TAE for several datasets across a test-grid of initial acceptance probabilities and maximum allowable iterations. We have chosen defaults of 10,000 iterations and a 40% initial acceptance probability based on these results.

Secondly, we considered how to move from one candidate state $i_t \in S$ to the future candidate states $i_{t+1} \in S$. It is typically advisable to initially take large jumps in the solution space and in later iterations to mostly move to nearby candidate solutions. This is controlled by both the cooling schedule and by how candidate solutions are generated. In **synthACS**, these choices are guided by understanding that candidate solutions generally improve over time as SA mostly moves downhill. For a thorough treatment of this, please see Ingber (2000).

synthACS initially explores the candidate space by choosing new candidate solutions that replace 10% of the current solution's observations for the first 100 iterations. It then generates new candidate solutions by replacing `max(500, n * .005)` observations in each iteration. In addition, a large jump in the candidate space, `n * 0.2` observations, is proposed every 500 iterations to increase the probability of avoiding local minima. These choices were guided by experimental results, as with the choice of initial acceptance probability and maximum iterations.

Thirdly, **synthACS** allows for early stopping. To be clear, early stopping is defined here as taking fewer iterations than the maximum allowable iterations because the loss criteria has fallen below a given threshhold. Cross-validation on a test data set, typical of many early stopping regularization techniques, is not used. The goal of this early stopping choice is to reduce computation time, not to reduce generalization error. By default, the rule for early stopping used for a small area with n individuals and D constraining attributes is to stop if TAE falls below $\frac{n}{2000}D$, eg. a misclassification rate of 1 individual per thousand for each constraining attribute.

3. User guide

synthACS function	Data attributes available
`pull_bachelors`	Degrees awarded by major
`pull_edu`	Educational enrollment and attainment
`pull_geo_mobility`	Cross sectional data on geographic mobility
`pull_household`	Housing stock, occupancy, and rental rates
`pull_inc_earnings`	Income, hours worked, measures of income distribution
`pull_mar_status`	Cross sectional data on marital status
`pull_population`	Cross sectional data on population
`pull_pov_inc`	Income and poverty metrics
`pull_transit_work`	Method and length of transit and type of work

Table 3: Summary of functions in **synthACS** accessing ACS base tables.

We now turn to the use of **synthACS**, which is available as a package for the statistical software R (R Core Team (2016)) and can be run in any environment that R can. R is freely available from `https://www.r-project.org/`. To install **synthACS** from a CRAN (Comprehensive R Archive Network) mirror, simply type the following command in the R command prompt,

```
R> install.packages("synthACS").
```

If you wish to use the most current development version, you can install it from GitHub. This is most easily done using the **devtools** package Wickham and Chang (2016), via

```
R> devtools::install_github("alexWhitworth/synthACS").
```

To keep your copy of **synthACS** up to date, you should use the R command `update.packages()`.

Before using **synthACS**, users must setup access to ACS data via the Census API. Data is accessed via the **acs** package Glenn (2016), on which **synthACS** depends. Specifically, users need to request a Census Developer Key at `http://api.census.gov/data/key_signup.html` and install it via `acs::api.key.install`. This installation is saved in the system and thus only required once.

```
R> acs::api.key.install(key = "...<redacted>...")
```

3.1. The macroACS class

To begin using **synthACS** researchers first define the geography they wish to work with, via `acs::geo.make`. After defining a geography, users should pull data on their geography. The first feature to note in **synthACS** is an extension of **acs** that provides functions for accessing specific macro data contained in the ACS base tables. In essence, these functions provide easy access to a curated set of base tables which we thinks may be of interest to researchers. In the case that researchers are interested in alternative tables, they can still use the `acs::acs.fetch` function to access them. Table 3 enumerates these functions and detailed descriptions for each function can be found via the R help documenation.

These functions share a common syntax. To illustrate, we define the geography of interest as all counties in California and pulls rolling five-year data on bachelors degrees awarded with end year 2014. Here, `endyear` is used to specify the end year of the data collection period, `span` is used to indicate the length in years of data collection, and `geography` is used to specify the geography of interest.

```
R> library("synthACS")
R> ca_geo <- geo.make(state = "CA", county = "*")
R> ca_bachelors <- pull_bachelors(endyear = 2014, span = 5,
     geography = ca_geo)
```

All of the functions noted in Table 3 return an S3 class object of class `macroACS`, which has a very similar structure to the S4 class `acs-class` from the **acs** package. The elements `endyear`, `span`, and `geography` are the same as the corresponding elements from `'acs-class'` objects, while the `estimates` and `standard_error` elements are `lists` of `data.frames` instead of objects of class `matrix` as in `'acs-class'` objects. There is an additional element as well, `geo_title`, which gives summary information on the geography set. The `macroACS` class also provides several methods, which are discussed below.

```
R> methods(class = "macroACS")
```

```
[1] fetch_data  gen_attr_vectors  get_dataset_names  get_endyear
[5] get_geography  get_span  split
```

`split` and `gen_attr_vectors` will be described in Section 3.2 and Section 3.2.1 respectively. The other methods are described here:

- The methods `get_span`, `get_endyear`, and `get_geography` return information on the data collection span, data collection endyear, and the geography of the `macroACS` object.
- `get_dataset_names` is used to return the names of the datasets which can be supplied to `fetch_data`.
- `fetch_data` returns macro data from either the `estimates` or `standard_error` elements of the `macroACS` object for a subset of the object's geographies.

An example of these methods is provided. Assume users want to know the median age by sex for Los Angeles and San Diego Counties. Or, alternatively, they may be intersted in standard errors for median age by sex for all counties. To do so, users first use the `pull_population` function to pull the appropriate ACS data and then `fetch_data` on the resulting `macroACS` object.

```
R> ca_county_pop <- pull_population(2014, 5, geography = ca_geo)
R> get_span(ca_county_pop)
```

```
[1] 5
```

```
R> get_endyear(ca_county_pop)

[1] 2014

R> get_geography(ca_county_pop)

[[1]]
"geo" object: [1] "All counties, California"

R> get_dataset_names(ca_county_pop)

[1] "sex_by_age"  "median_age_by_sex"  "pop_by_race"
[4] "place_birth_by_lang_at_home" "place_birth_by_mar_status"
[6] "place_birth_by_edu_attain" "place_birth_by_income"
[8] "median_income_by_place_birth" "place_birth_by_pov_status"
```

`fetch_data` takes four arguments. The first, `acs`, supplies the `macroACS` class object. The second, `geography`, takes a character vector and uses regular expression pattern matching to return the requested subset of geographies. The special character `"*"` may be used to indicate the full set of geographies. The third argument, `dataset`, indicates that either the ACS data estimates or the estimates' associated standard errors should be returned. And the final argument, `choice`, specifies the dataset of interest.

```
R> fetch_data(ca_county_pop, geography = c("Los Ang", "San Diego"),
     dataset = "estimate", choice= "median_age_by_sex")
R> fetch_data(ca_county_pop, geography = "*", dataset= "st.err",
     choice = "median_age_by_sex")

                              total male female
Los Angeles County, California  35.3 34.1   36.5
San Diego County, California    34.9 33.5   36.4
## standard errors
                                      total       male     female
Alameda County, California         0.06079027 0.06079027 0.06079027
Alpine County, California          3.64741641 2.06686930 2.55319149
Amador County, California          0.24316109 0.24316109 0.36474164
... output truncated ...
Ventura County, California         0.12158055 0.06079027 0.12158055
Yolo County, California            0.12158055 0.12158055 0.18237082
Yuba County, California            0.12158055 0.24316109 0.18237082
```

3.2. Generating synthetic micro data

In addition to the functions listed in Table 3, there is one special function for pulling data. `pull_synth_data` is used to pull the default data used for synthetic data generation, as outlined in Table 2, and marks our entry into SMSM.

```
R> ca_dat_SMSM <- pull_synth_data(2014, 5, ca_geo)
```

`pull_synth_data` returns data from the macro constraining tables that will be used to both derive the sample of synthetic micro data (Section 2.2.1) and, to constrain the simulated annealing algorithm (Section 2.2.2).

Deriving the sample synthetic micro data is a memory intensive process. For the California county data illustrated here, the 20th-percentile county's synthetic sample contains 2.9 million rows and takes roughly 140 mb of memory. It is therefore best to use **synthACS** on a high performance machine. If such a machine is not available, the `macroACS` class offers the `split` method. `split` allows users to split the task of SMSM into sets of tasks which your machine's memory can handle. Recombining the output will be discussed in Section 3.4. `split` has straightforward syntax. The below command splits `ca_dat_SMSM`, which contains fifty-eight counties into ten subsets, eight containing six counties and two containing five counties.

```
R> split_ca_dat <- split(ca_dat_SMSM, n_splits= 10)
```

We now return to the main topic of conducting the SMSM. The next step in this process is to derive the sample synthetic micro datasets. This is accomplished via `derive_synth_datasets`, which operates in parallel by default, and returns an object with the S3 class `synthACS`.

```
R> library("parallel")
R> ca_synthetic <- derive_synth_datasets(ca_dat_SMSM, leave_cores = 0)
```

The argument `leave_cores` allows users to optionally leave one or more of the cores on their multicore machine available for other tasks.

Flexible tools for new data attributes

synthACS provides ten default attributes for each individual as described in Table 2. These attributes should be sufficient for most research questions; and, if so, users can proceed immediately to fitting the data to the macro constraining tables via simulated annealing (see Section 3.3). But many users' research questions will not be properly served by the defaults. **synthACS** provides flexible tools for both adding new data attributes to the default elements and for removing unneeded attributes. For example, users might wish to use an alternative income metric such as data from the Current Population Survey (CPS), found at http://www.census.gov/programs-surveys/cps/data-detail.html. Or they may not be interested in the `race`, `pov_status`, or other default attributes.

Marginalizing attributes is straightforward and accomplished by `marginalize_attr`. Users simply provide the object with which they wish to work, the attribute list to marginalize, and whether they wish to keep the specified attributes or to remove them.

```
R> ca_marg <- marginalize_attr(ca_synthetic, varlist = c("race",
     "ind_income", "nativity", "geog_mobility"), marginalize_out = TRUE)
```

Here, we remove the attributes `race`, `ind_income`, `nativity`, and `geog_mobility` from the synthetic datasets, where `marginalize_out` indicates that we wish to remove the listed attributes instead of keep them. By default, `marginalize_out` is `FALSE`.

Adding attributes is more difficult. To add attributes, users must first get the data on the new attribute(s) they wish to add and also specify the conditional model with with the new attribute should be added. There are four different conditional models which may be considered for each new attribute. The models have increasing complexity, moving from attribute independence to the general case of complete joint dependence.

1. Independence: Each of the attributes is independent of the others. That is, the new attribute is unconditional on prior attributes.
2. Pairwise conditional independence: Attributes are related to only one other attribute and independent of all others.
3. Conditional independence: Attributes are depedent on some subset of other attributes and independent of the rest.
4. General case: In the most general case, all attributes are jointly interrelated.

All four conditional models are supported by **synthACS**, via `synthetic_new_attribute` for a single small area and `all_geog_synthetic_new_attribute` for a set of small areas.

To illustrate, we utilize mode of transit to work data from the ACS assuming a conditional independence model. Mode of transit to work is dependent on both employment status and age and independent of all other attributes.

```
R> ca_transit <- pull_transit_work(2014, 5, ca_geo)
R> get_dataset_names(ca_transit)

[1] "time_to_work_by_sex"  "mode_transit_by_age"
[3] "median_earnings_by_mode_transit"  "median_age_by_mode_transit"
[5] "mode_transit_by_occ"  "place_of_work_metroSA"
[7] "place_of_work_microSA"

R> mode_levels <- c("no_work", "drove_alone", "carpool", "transit",
     "walk", "other", "work_at_home")
R> transit_work <- gen_attr_vectors(ca_transit,
     choice = "mode_transit_by_age")
```

`mode_levels`, specified above, represents the possible values of the new attribute. The ACS records that individuals either do not work, or get to work via driving along, carpooling, taking public transit, walking, other methods, or work at home. `transit_work`, also specified above, is ACS data on the counts of individuals in each county where counts are tabulated by age and method of transport to work. An example of the raw data is provided below.[3] `transit_work` will be used to reweight the existing synthetic records when deriving the new attribute.

```
R> round(transit_work[[1]], 0)
```

[3] Calculated percentages have been removed from the displayed output to simplify presentation and ease comprehension.

```
cnt_all drove_alone_all drove_alone_16_19 drove_alone_20_24
 701525     451989              7094             33272
drove_alone_25_44 drove_alone_45_54  drove_alone_55_59  drove_alone_60_64
 207478              110652               43689              30728
drove_alone_65up carpool_all  carpool_16_19  carpool_20_24  carpool_25_44
 19076              71593        2068           5885           37891
carpool_45_54  carpool_55_59  carpool_60_64  carpool_65up  transit_all
 15011            5270           3481           1987          89723
transit_16_19 transit_20_24 transit_25_44 transit_45_54  transit_55_59
 1679           8469          49523         16419          6926
transit_60_64 transit_65up  walk_all  walk_16_19  walk_20_24 walk_25_44
 4338           2369          25802     2046        5301       10552
walk_45_54  walk_55_59  walk_60_64  walk_65up  other_all  other_16_19
 3781        1697        1296        1129        23420      448
other_20_24  other_25_44 other_45_54  other_55_59  other_60_64  other_65up
  2365         12405       4362         1789         1227         824
work_home_all  work_home_16_19  work_home_20_24  work_home_25_44
  38998          605               1644             15725
work_home_45_54  work_home_55_59   work_home_60_64  work_home_65up
  9571             4618              3461             3374
```

Users must manipulate their raw input data, in this case `transit_work`, to fit the input requirements of **synthACS**. Often, this will also require one or more simplifying assumptions due to inexact data availability (eg. see Section 2.2.1).

Readers should immediately notice two things when looking at `transit_work`: (1) the ages specified do not exactly match the ages in the synthetic micro data; and (2) employment status is not explicitly stated. For the first issue, users have to make intelligent choices to map differences in age groupings; for the second, note that transit to work implicitly assumes that the individual is employed.[4] For the second issue, the data must be augmented to account for individuals not employed; and, for the first, we assume the following:

1. Ages 15-17 can be best represented by the age 16-19 transit to work data.
2. Ages 18-24 can be best represented by the age 20-24 transit to work data.
3. A loss of data granularity for the transit attribute is unavoidable. Ages 25-85+ in the existing synthetic data, which are in 5-year age buckets, will be mapped to their associated bucket in the transit to work data: 24-44, 45-54, 60-64, 65+.

We now manipulate `transit_work`, via anonymous function passed to `lapply`, so that it is an acceptable input to `all_geog_synthetic_new_attribute`.

```
R> transit_work <- lapply(transit_work, function(x, lev) {
  unemp <- data.frame(emp_status = "unemployed", age = NA, pct = 1,
    level = "no_work")
```

[4]This can also be seen in the counts. The transit vector counts 701,525 individuals; but there are 1,559,308 individuals in the county population.

```
  NI_LF <- data.frame(emp_status = "not_in_labor_force", age = NA,
    pct = 1, level = "no_work")

  x <- x[!grepl(paste(c("_all", "pct_"), collapse= "|"), names(x))]
  emp_status <- rep("employed", length = length(x))
  age <- substr(names(x), nchar(names(x)) - 4, nchar(names(x)))
  age <- ifelse(substr(age, 1, 1) == "_", substr(age, 2,
    nchar(age)), age)
  age <- ifelse(age == "16_19", "15_17",
    ifelse(age == "20_24", "18_24", age))

  levels <- substr(names(x), 1, nchar(names(x)) - 5)
  levels <- ifelse(substr(levels, nchar(levels), nchar(levels)) == "_",
    substr(levels, 1, nchar(levels) - 1), levels)

  employed <- data.frame(emp_status = emp_status, age = age, pct = x,
    level = levels)
  rownames(employed) <- NULL;
  employed <- split(employed, employed$age)
  employed <- list(employed[['15_17']], employed[['18_24']],
    replicate(n = 4, employed[['25_44']], simplify = FALSE),
    replicate(n = 2, employed[['45_54']], simplify = FALSE),
      employed[['55_59']], employed[['60_64']],
    replicate(n = 5, employed[['65up']], simplify = FALSE))

  employed[[3]][[1]]$age <- "25_29"
  employed[[3]][[2]]$age <- "30_34"
  employed[[3]][[3]]$age <- "35_39"
  employed[[3]][[4]]$age <- "40_44"
  employed[[4]][[1]]$age <- "45_49"
  employed[[4]][[2]]$age <- "50_54"
  employed[[7]][[1]]$age <- "65_69"
  employed[[7]][[2]]$age <- "70_74"
  employed[[7]][[3]]$age <- "75_79"
  employed[[7]][[4]]$age <- "80_84"
  employed[[7]][[5]]$age <- "85up"
  employed[[3]] <- do.call("rbind", employed[[3]])
  employed[[4]] <- do.call("rbind", employed[[4]])
  employed[[7]] <- do.call("rbind", employed[[7]])

  return(rbindlist(list(unemp, NI_LF, rbindlist(employed))))
}, lev = mode_levels)
```

The resulting output is a symbol table, where the key in the symbol table's key-value pair is the set of conditioning variables and their values; and the value is the new attribute's conditional counts and the associated new attribute value. To satisfy the general case of joint attribute dependence, **synthACS** allows for any number of conditioning variables; however,

data is rarely available for more than a handful. Also note that conditional counts can be specified as counts or as percentages. Both cases are used here: percentages for `emp_status` in `c("unemployed", "not_in_labor_force")`, while `emp_status == "employed"` uses counts.

With the raw data properly cleaned, we now proceed to adding the attribute, which is accomplished via `all_geog_synthetic_new_attribute`. Here `attr_name` is a string specifying the name of the new attribute, `conditional_vars` is a character vector specifying the variables we wish to condition the new attribute upon, and `st_list` is the list of symbol tables created above.

```
R> ca_and_transit <- all_geog_synthetic_new_attribute(ca_marg,
     attr_name = "transit_work",
     conditional_vars = c("emp_status", "age"),
     st_list = transit_work)
```

We now return to the unmodified synthetic microdata to continue our discussion of SMSM.

3.3. SMSM via simulated annealing

There are three steps to conducting SMSM on the now generated synthetic micro data in **synthACS**. Firstly, the researcher must decide which data attributes to optimize against; a choice which should be driven by the research question. In certain cases, for instance, having an accurate representation of age, gender, and maritial status may be critically important; in others, gender may be unimportant but income and educational status may be critical. After determining which attributes to optimize against, the researcher then builds constraints from the macro tables and maps them between the micro and macro data. Finally, the researcher conducts simulated annealing. These steps are illustrated below.

We optimize against `age`, `gender`, `mar_status`, `race`, and `edu_attain` for our example. We now build constraints and map them to the data. For the ten default data attributes provided in `pull_synth_data`, **synthACS** provides user friendly methods for building constraints and for for mapping constraints between the micro and macro dataset.

```
R> methods(class = "synthACS")
```

```
 [1] all_geog_constraint_age              all_geog_constraint_edu
 [3] all_geog_constraint_employment       all_geog_constraint_gender
 [5] all_geog_constraint_geog_mob         all_geog_constraint_income
 [7] all_geog_constraint_marital_status   all_geog_constraint_nativity
 [9] all_geog_constraint_poverty          all_geog_constraint_race
[11] marginalize_attr
```

Each of the constraint methods has the same syntax. The first argument takes an object of class `synthACS` and the second indicates if we wish to build the constraints from the sample synthetic data or from the macro constraining tables. There is very little difference in result; but the differences that do exist are the results of the assumptions outlined in Section 2.2.1 describing differences in the populations on which the ACS base tables are defined.

```
R> a <- all_geog_constraint_age(ca_synthetic, method = "macro.table")
R> g <- all_geog_constraint_gender(ca_synthetic, method = "macro.table")
R> m <- all_geog_constraint_marital_status(ca_synthetic,
    method = "macro.table")
R> r <- all_geog_constraint_race(ca_synthetic, method = "synthetic")
R> e <- all_geog_constraint_edu(ca_synthetic, method = "synthetic")
```

These constraints are then mapped to the data via `all_geogs_add_constraint`.

```
R> cll <- all_geogs_add_constraint(attr_name = "age",
     attr_total_list = a, macro_micro = ca_synthetic)
R> cll <- all_geogs_add_constraint(attr_name = "gender",
     attr_total_list = g,
     macro_micro = ca_synthetic, constraint_list_list = cll)
R> cll <- all_geogs_add_constraint(attr_name = "marital_status",
     attr_total_list= m,
     macro_micro = ca_synthetic, constraint_list_list= cll)
R> cll <- all_geogs_add_constraint(attr_name = "race",
     attr_total_list = r,
     macro_micro = ca_synthetic, constraint_list_list = cll)
R> cll <- all_geogs_add_constraint(attr_name = "edu_attain",
     attr_total_list = e,
     macro_micro = ca_synthetic, constraint_list_list = cll)
```

Here, `attr_name` provides the attribute name in the micro data, `attr_total_list` provides the previously generated constraints, `macro_micro` provides the dataset on which we will conduct SMSM, and `constraint_list_list` is used to indicate that we are updating an existing constraint set.

We are finally ready for simulated annealing, performed via `all_geog_optimize_microdata`.

```
R> opt_ca <- all_geog_optimize_microdata(ca_synthetic, seed = 6550L,
     constraint_list_list = cll, p_accept = 0.4, max_iter = 10000L,
     verbose = TRUE)

Beginning parallel optimization...
... Optimization complete
```

The most important arguments are to specify the dataset and the `constraint_list_list`. Several defaults are also illustrated above. Firstly, we choose a specific `seed` for reproducibility and a maximum number of allowable iterations (`max_iter`). An inital probability of acceptance (`p_accept`) for uphill moves is also specified. This acceptance probability will be controlled throughout the annealing process through the cooling schedule Equation 2.2.2. Finally, the `verbose` option is specified for intermediate outputs.

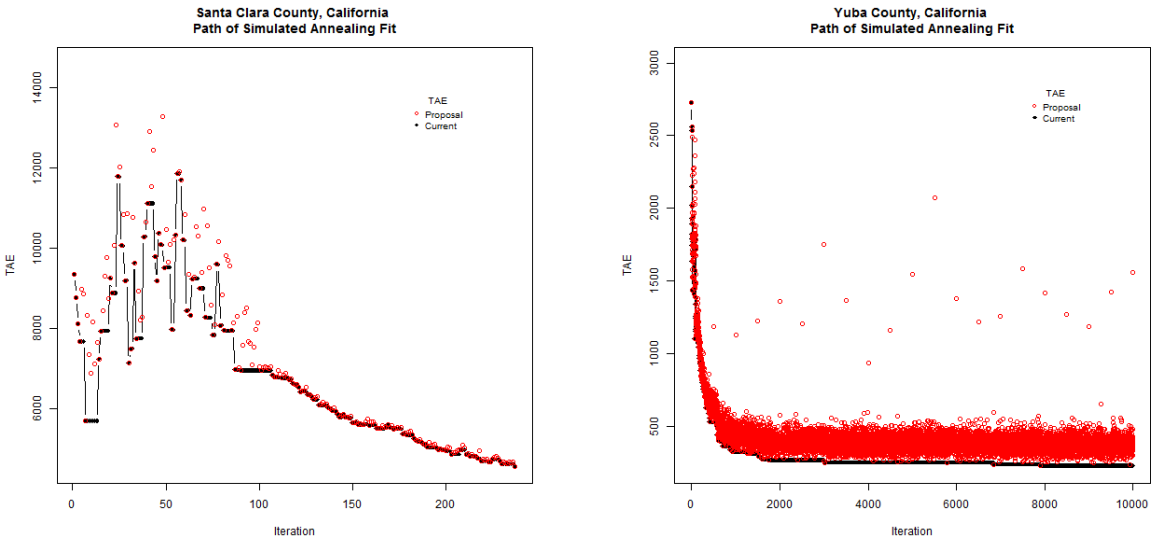
3.4. Examining the SMSM output

`all_geog_optimize_microdata` returns an '`smsm_set`' class object. The optimized micro populations for each small area and their associated final TAE are in the named `best_fit` and `tae` elements of an '`smsm_set`' object respectively. They can be accessed individually via `get_best_fit` and `get_final_tae`. In a similar manner to `get_data` in Section 3.1, these methods work by specifying a character string that identifies a geography by name.

```
R> LA_fit <- get_best_fit(opt_ca, geography = "Los Ang")
R> LA_tae <- get_final_tae(opt_ca, geography = "Los Ang")
```

It is also possible to visually examine the path of TAE during the simulated annealing algorithm for each geography. This is done via `plot_TAEpath`, which again uses string matching on geography names.

```
R> plot_TAEpath(opt_ca, geography = "Santa Cl")
R> plot_TAEpath(opt_ca, geography = "Yuba")
```



(a) Santa Clara County, CA **(b)** Yuba County, CA

Figure 1: Plots of TAE paths from simulated annealing.

`plot_TAEpath` uses solid black circles to represent the current best solution and empty red circles to represent candidate solutions. Here we see that the simulated annealing procedure for Santa Clara County, CA converged in a relatively quick 237 iterations. We also see several uphill moves during the first 90 iterations of simulated annealing followed by a consistently downward trajectory in TAE after that. For Yuba County, we see a rapid descent in TAE over the first few hundred iterations followed by relative statis in TAE being interupted by the occassional better candidate solution.

Users may also be interested in assessing the fit of all small area geographies simultaneously. The `summary` method is available for this purpose. In this case we see that 65.5% of California counties reached an acceptable TAE before the maximum allowable iterations and that the median percentage TAE was 0.000496.

```
R> summary(opt_ca)

Call:
 all_geog_optimize_microdata(macro_micro = ca_syn, prob_name = "p",
    constraint_list_list = cll, p_accept = 0.4, max_iter = 10000L,
    seed = 6550L)

 Seed:  6550
 n-Constraints:  5

 Maximum Iterations:  10000
 %-Early Stop:  0.6552

 Mean %-TAE:  0.001063
 Median %-TAE:  0.000496
 Max %-TAE:  0.008319
 %-TAE quantiles:
       0%      25%      50%      75%      90%      95%
0.000425 0.000487 0.000496 0.000804 0.002329 0.002911
```

Lastly, recall that in Section 3.2 we noted that synthetic data creation in **synthACS** is very memory intensive. We proposed the `split` method to work with objects of manageable size. To recombine objects which were `split`, users should use `combine_smsm`. This function inputs a list of `'smsm_set'` objects and outputs a single combined `'smsm_set'` object.

4. Case study

We now present a case study related to population dynamics to illustrate how **synthACS** and the microsimulation modeling framework can be used in practice. We look at both fertility and mortality in Los Angeles County, California and examine the spatial distribution of both demographic events. Examining fertility and mortality within a spatial framework enables policy makers and community health officials to assess whether existing health facilities are sufficient to support the community, whether a geographic region provides equal access to health facilities, and to examine the future need for healthcare facilities based on likely population changes over time.

Fertility and mortality also form the basis of population projection, which distills to the basic demographic identity that the population at the next time point is equal to the population at the current time point plus births and immigrants and minus deaths and emigrants as in Equation 4.

$$P_{t+1} = P_t + B_t - D_t + M_t \tag{4.0.1}$$

Here, P is the population, B is the aggregate births, D is the aggregate deaths, and M is net migration. The provided case study can thus also be considered a population projection method holding net migration equal to zero.

The standard framework for population projection involves the so-called cohort component method. The basic idea of the cohort component method is that populations change because individuals experience demographic events (eg. births, deaths, and migration) and that these events can be generalized by age, gender, and event type. For each group, or cohort, the impact on population for a given event is determined by the size of the cohort and the incidence rate among that cohort.

In the macro deterministic approach, the change in population is calculated by multiplying the incidence rate for each cohort and event by the cohort size and then summing across both event type and cohort. In the SMSM framework, the deterministic approach is replaced by a random experiment for each individual using the Monte Carlo method. While SMSMs do exist for population projection Van Imhoff and Post (1998), most population projections are currently done deterministically using the cohort component method (eg. Department of Economic and Social Affairs: Population Division (2013); Colby and Ortman (2014); Raftery *et al.* (2012)).

4.1. Fertility and mortality in Los Angeles County

The data examined in this case study is of Los Angeles County, California at the census tract level. We use five-year data with end year 2014. As historic data, the data does not account for population changes that have occured between the time of data collection and present day. For instance, the California State Demographic Research Unit estimates that population has grown from 9,889,520 in 2012, the median year of data collection, to 10,136,559 in 2015 (an annual growth rate of 0.8%).[5] The total population from the five-year ACS data used in this simulation is 9,997,939.

Data on fertility and mortality rates comes from the National Vital Statistics System (Hamilton *et al.* (2015), Xu, Murphy, and Kochanek (2015)). For fertility, data on female birth rate by age, race and hispanic origin, and marital status are used. Additonally, we include an adjustment for multiple births (eg. twins and triplets) and birth rates are adjusted for the difference between the state of California and the US as a whole. Mortality data used includes death rate by gender, age, and race and hispanic origin. These datasets have been made available in the **synthACS** package.

To prepare the data for population projection, we conduct SMSM for all census tracts in Los Angeles County, California as outlined in Section 3. The following attributes were used to constrain the simulated annealing algorithm: `age`, `race`, `gender`, `mar_status`, and `edu_attain`. The resulting misclassification rate across the 2,319 census tracts in Los Angeles County was 2.19 individuals per 1,000 at the 25th percentile, 2.51 individuals per 1,000 at the 50th percentile, and 3.40 individuals per 1,000 at the 90th percentile.

One thousand simulations were run for both fertility and mortality in each census tract. Summary data for both demographic events are provided in Table 4. The fertility data is broken out by major racial groups and the mortality data is broken out by major age brackets. We note that the choice of racial and age groups are arbitrary, reporting can be done by any cross sectional attribute the researcher desires. It is clear from examining Table 4 that most census tracts will have either no deaths or one death in many age groups. This leads to assymetric distributions about the median for these age groups.

[5] Table po04 courtesy of Los Angeles Almanac (`http://www.laalmanac.com/population/po04.htm`) Accessed: 7-12-2016

Fertility	10th percentile	50th percentile	90th percentile
All	114,493	134,579	155,743
Asian	10,970	16,508	23,161
Black or African American	6,635	10,344	15,318
Hispanic or Latino	56,038	69,207	83,461
White (non-Hispanic)	24,194	32,311	41,638
Mortality	10th percentile	50th percentile	90th percentile
All	46,033	59,560	74,257
0-1	0	49	2,314
1-4	0	0	74
5-9	0	0	2
10-14	0	0	7
15-17	0	0	425
18-24	9	79	2,389
25-29	0	48	2,176
30-34	0	56	2,275
35-39	0	81	2,484
40-44	0	400	3,160
45-59	9	1,202	4,349
50-54	133	2,264	5,853
55-59	409	3,066	7,107
60-64	728	3,808	8,130
65-69	930	4,114	8,531
70-74	1,310	4,600	9,227
75-79	2,109	5,815	10,760
80-84	3,490	7,670	13,054
85+	12,146	18,693	21,190

Table 4: Summary of expected fertility and mortality in Los Angeles County, California. Sums of the cross-sections will not sum to the totals as quantiles are calculated independently. Some racial groups are omitted.

Simulation accuracy can also be assessed. Los Angeles County, California had 131,697 births in 2012 and 128,523 in 2013 vs the median birth simulation of 134,579. This discrepancy can be almost completely explained by the difference in fertility rate between Los Angeles County and California, which is not controlled for in the simulation. Los Angeles County had a fertility rate of 60.6 and 59.3 in 2012 and 2013 respectively California Department of Public Health (2013) versus 62.4 for California as a whole in 2014–the fertility rate used in the simulations. Similarly, there were 57,988 deaths in Los Angeles County in 2011 and 58,498 in 2012 Office of Health Assessment and Epidemiology (2015) versus the median simulation of 59,560. Differences in the mortality data may be due to differences in the 2014 end-year population from the census or from differences in death rates in Los Angeles County relative to the United States.

In addition to examining simulation accuracy, different quantiles of the simulations can be contextualized spatially. This is illustrated in the below figures.[6]

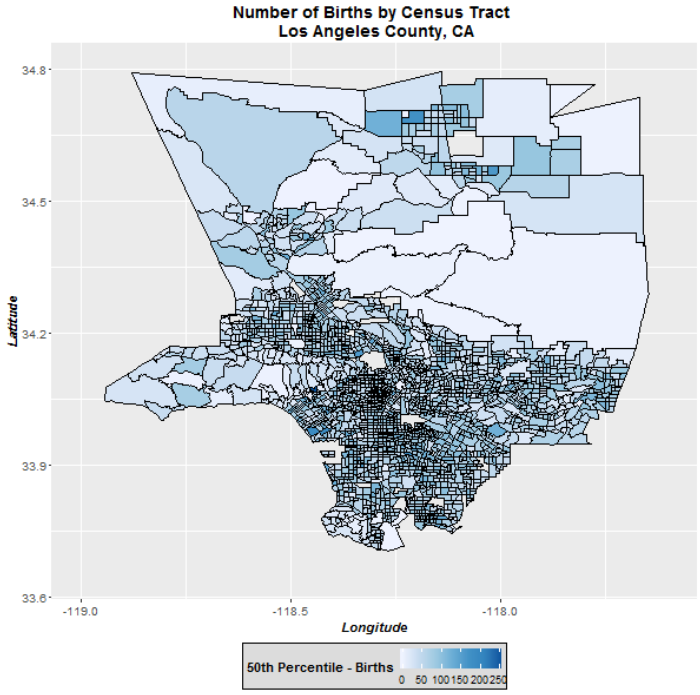


Figure 2: The 50th percentile simulation of births in Los Angeles County, CA.

[6]Catalina Island has beed excluded from the below figures.

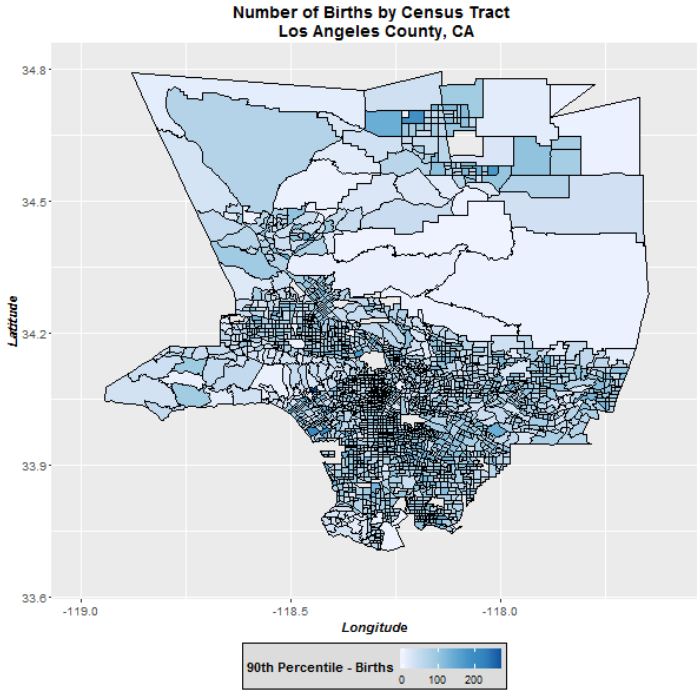


Figure 3: The 90th percentile simulation of births in Los Angeles County, CA.

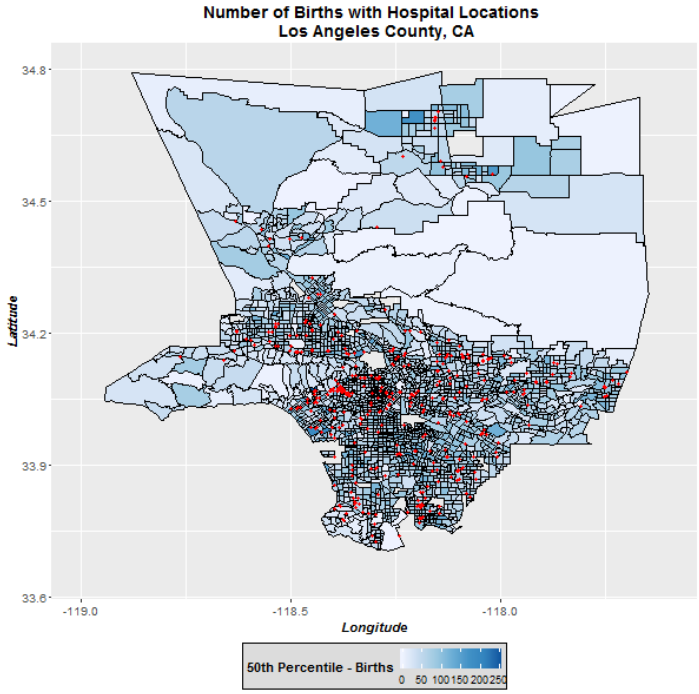


Figure 4: The 50th percentile simulation of births in Los Angeles County, CA with overlay of hospital locations (NAICS: 622110).

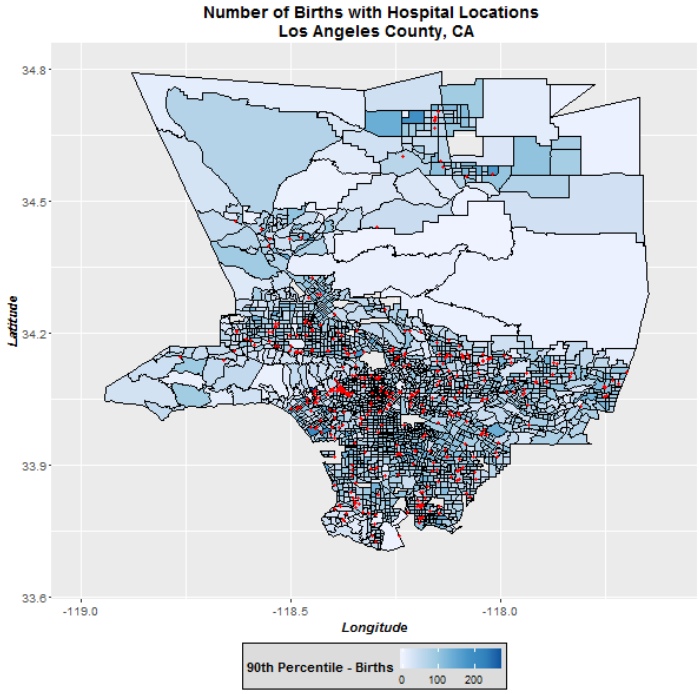


Figure 5: The 90th percentile simulation of births in Los Angeles County, CA with overlay of hospital locations (NAICS: 622110).

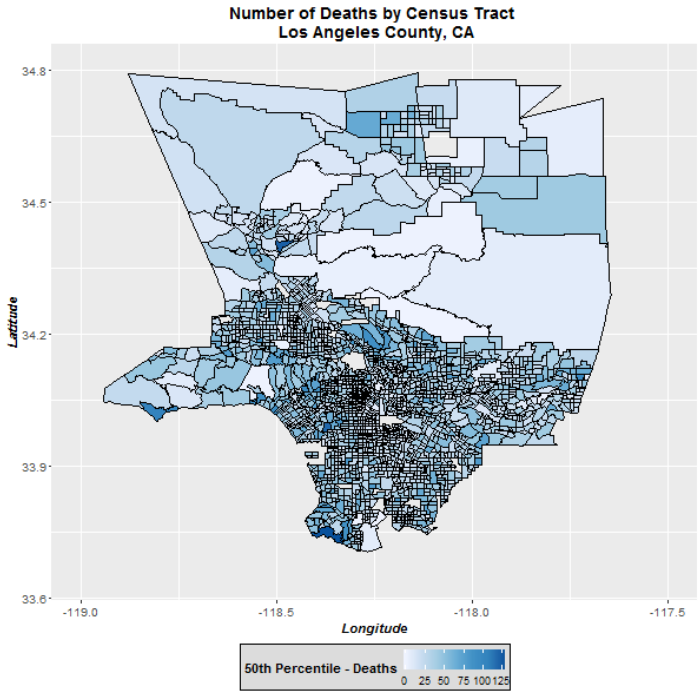


Figure 6: The 50th percentile simulation of deaths in Los Angeles County, CA.

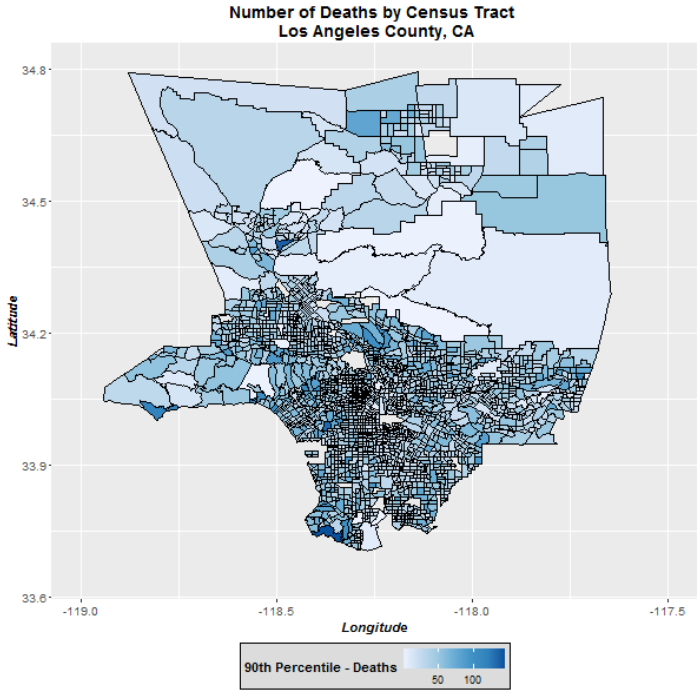


Figure 7: The 90th percentile simulation of deaths in Los Angeles County, CA.

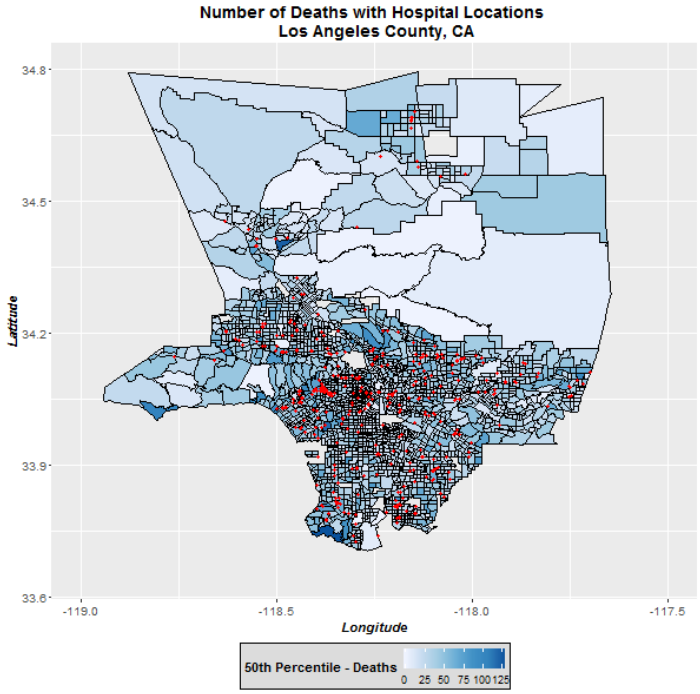


Figure 8: The 50th percentile simulation of deaths in Los Angeles County, CA with overlay of hospital locations (NAICS: 622110).

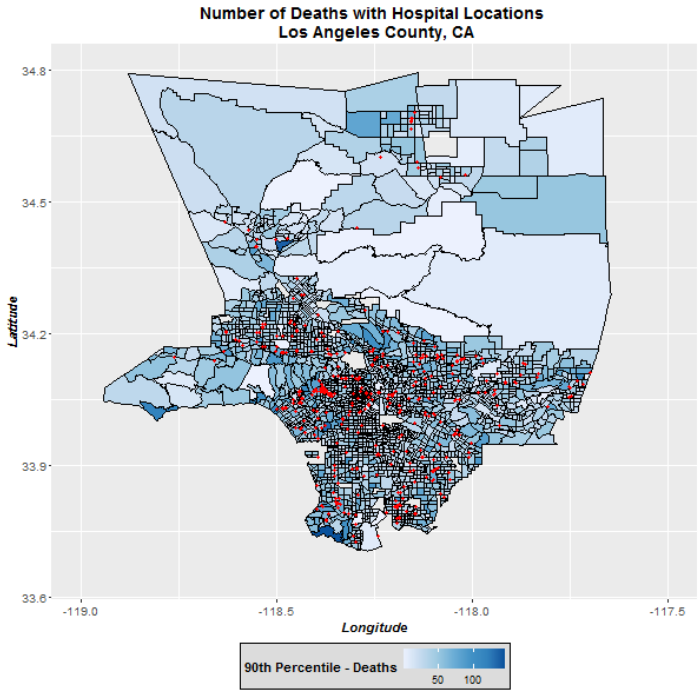


Figure 9: The 90th percentile simulation of deaths in Los Angeles County, CA with overlay of hospital locations (NAICS: 622110).

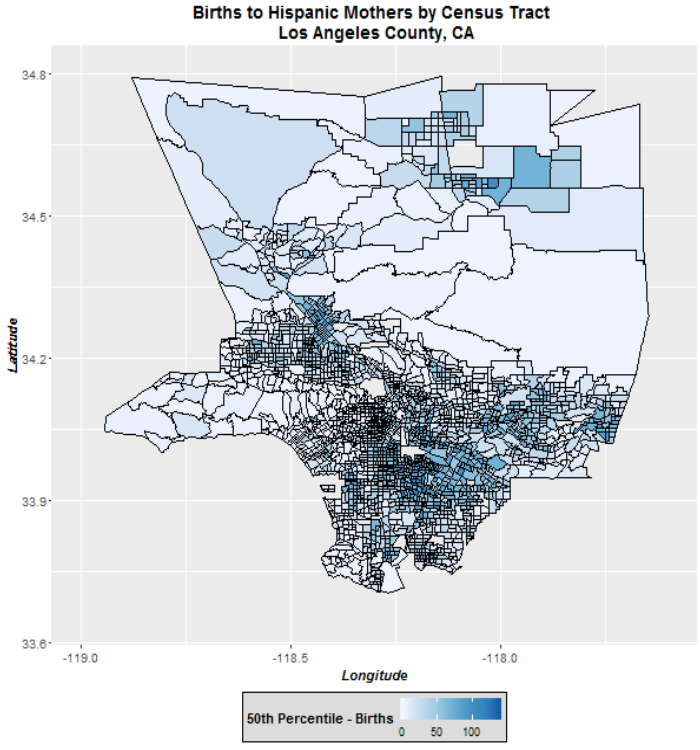


Figure 10: The 50th percentile simulation of births to hispanic mothers in Los Angeles County, CA.

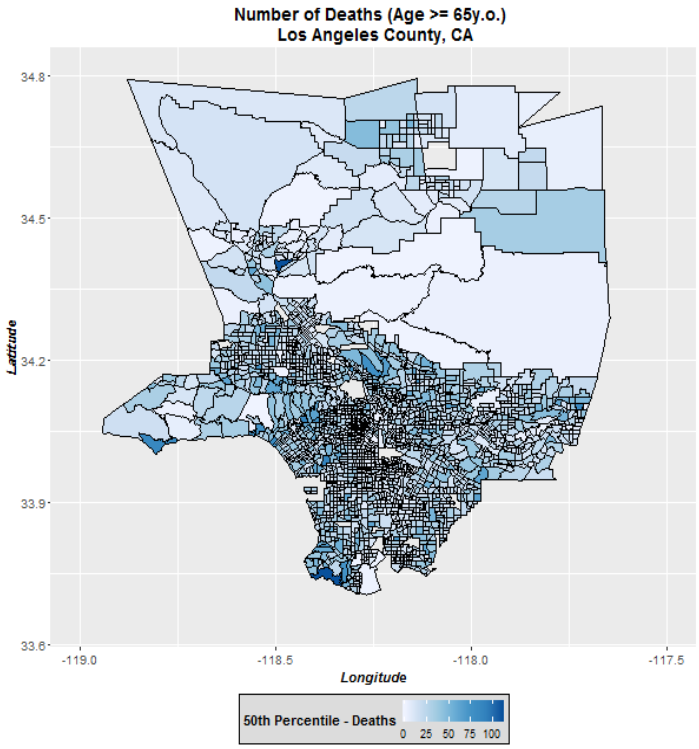


Figure 11: The 50th percentile simulation of deaths of individuals aged 65 or older in Los Angeles County, CA.

The above figures show that both fertility and mortality in Los Angeles County, California is concentrated in the Southern half of the county. This is not surprising since the Northern half of Los Angeles County is predominantly composed of Angeles National Forest, Los Padres National Forest, and the Western edge of the Mojave Desert. However, this region also includes the cities of Lancaster and Palmdale, which have a combined population of roughly 320,000 individuals US Census Bureau (2015), as well as several smaller towns.

Looking at Figures 8 and 9 in particular, it appears that the Northern region of Los Angeles County may be underserved with health facilities. Specifically, we observe an elevated number of deaths relative to the number of hospitals, where hospitals are businesses with NAICS code 622110. In addition, when looking at Figure 10, we observe that the area of East Los Angeles County has a high rate of births to hispanic mothers, corresponding to the large hispanic population there. However, as can be seen in Figure 4, there appears to be a relative dearth of hospitals in this area.

While policy suggestions are beyond the scope of this paper, we note that, based on this limited case study, health officials in Los Angeles County may wish to examine the equality of access to health facilities in the Northern half of Los Angeles County as well as in East Los Angeles County. We acknowledge that more substantial study would be needed to validate these observations.

4.2. Discussion of SMSM for population projection

One critique of the macro-based deterministic approach to population projection is that it does not provide any estimate of uncertainty. The standard approach is to use a total fertility rate that is one-half child lower for a "Low" projection and one-half child higher for a "High" projection. These alternative scenarios have no probabilistic interpretation. Raftery *et al.* (2012) and Sevcikova, Li, Kantorova, Gerland, and Raftery (2015) describe an alternative method for communicating uncertainty in population projection by probabilistically projecting the period total fertility rates (TFR) and life expectancies for all countries using Bayesian hierarchical models. Azose, Sevcikova, and Raftery (2016) improve these methods to also take into account uncertainty of migration patterns. Their methods enable probabilistic projections with cohorts defined by age and sex.

In contrast to the standard macro-based approach, SMSMs for population projection provide a natural estimate of uncertainty because they are based on Monte Carlo simulation. Analysts can run multiple simulations and provide uncertainty estimates via quantiles of the bootstrapped simulations. Of course, an obvious drawback of SMSMs is that running a simulation for every individual requires substantially more computation and memory.

Another critique of the standard cohort-component model is that cohorts are defined soley by age and gender. This leaves out a richness of attributes, such as education levels and race, which may be of interest to researchers. Another potential benefit of using SMSM for population projection is therefore the possibility to model the these extra individual attributes. We expect that population projection methods with estimates that "carry along" these additional individual attributes would be very useful to researchers. This is especially true in the case where researchers wish to examine the behavior of future populations where individuals express differential actions and preferences based on these attributes. For example, differential preferences in housing markets due to education level, income, nativity status, and race. Further research should be directed towards estimating the change in levels of these attributes

among population subgroups within an SMSM framework.

5. Conclusion

In this paper, we introduced **synthACS**, an R package which provides a general toolkit for conducting SMSM with any user-specified US geography. **synthACS** conducts SMSM by first creating synthetic micro data and then using simulated annealing with unequal probability sampling for optimization. The choice to use synthetic data is guided by the availability of high quality ACS data, while simulated annealing is chosen for accurate optimization.

synthACS provides intelligent default attributes for synthetic individuals. But, recognizing that researchers will have diverse data needs, **synthACS** is also data extensible. It provides tools for both adding new synthetic attributes and marginalizing undesired default attributes. **synthACS** also provides wrappers to the **acs** library enabling users to quickly access a curated selection of ACS data without requiing detailed knowledge of the ACS data structure.

This paper also provided a motivating example for **synthACS**, which examined the use of SMSMs for population projection and to assess the geographic equality of healthcare access. Specifically, we showed that SMSM can simulate births and deaths in Los Angeles County, California with a very high degree of accuracy. We also notes that SMSMs, unlike the standard deterministic cohort-component method, naturally provide a measure of uncertainty in demographic events. We suggest that further research on the use of SMSM for population projection should explore Monte Carlo methods for estimating and projecting individual attributes beyond just age and gender. These more comprehensive population projection methods would be very useful to researchers aiming to estimate future outcomes of individuals who exhibit differential preferences based on common attributes such as education level, income, and race.

References

Almquist ZW (2010). "US Census Spatial and Demographic Data in R: The **UScensus2000** Suite of Packages." *Journal of Statistical Software*, **37**(6), 1–31. doi:10.18637/jss.v037.i06.

Azose JJ, Sevcikova H, Raftery AE (2016). "Probabilistic Population Projections with Migration Uncertainty." volume 113, pp. 6460–6465.

Ballas D, Clarke GP, Wiemers E (2005a). "Building a Dynamic Spatial Microsimulation Model for Ireland." *Population, Space and Place*, **11**, 157–172. doi:10.1002/psp.359.

Ballas D, *et al.* (2005b). "**SimBritain**: A Spatial Microsimulation Approach to Population Dynamics." *Population, Space and Place*, **11**, 13–34. doi:10.1002/psp.351.

Birkin M, Clarke M (1988). "**SYNTHESIS** – A Synthetic Spatial Information System for Urban and Regional Analysis: Methods and Examples." *Environment and Planning A*, **20**(12), 1645–1671. doi:10.1068/a201645.

Bivand R, Lewin-Koh N (2016). ***maptools**: Tools for Reading and Handling Spatial Objects*. R package version 0.8-40, URL https://CRAN.R-project.org/package=maptools.

Bivand R, Pebesma E, Gomez-Rubio V (2008). *Applied Spatial Data Analysis with R*. Springer-Verlag.

California Department of Public Health (2013). "Birth Statistics Master Files." As cited on www.kidsdata.org, a program of the Lucile Packard Foundation for Children's Health. Accessed: July 27th, 2016, URL http://www.kidsdata.org/topic/610/fertility-rate/.

Clarke GP, *et al.* (1997). "Estimating Small Area Demand for Water: A New Methodology." *Water and Environment Journal*, **11**, 186–192. doi:10.1111/j.1747-6593.1997.tb00114.x.

Clarke M, Holm E (1987). "Microsimulation Methods in Spatial Analysis and Planning." *Geografiska Annaler Series B*, **69**, 145–169. doi:10.2307/490448.

Clarke M, *et al.* (1984). "A Strategic Planning Simulation Model of a District Health Service System: The In-Patient Component and Results." In W van Elmeren, R Engelbrecht, CD Flagle (eds.), *Systems Science in Health Care*. Springer-Verlag.

Colby SL, Ortman JM (2014). "Projections of the Size and Composition of the U.S. Population: 2014 to 2060." *Technical Report P25-1143*, U.S. Census Bureau. Current Population Reports.

Crimi N, Eddy W (2014). "Top-Coding and Public Use Microdata Samples from the U.S. Census Bureau." *Journal of Privacy and Confidentiality*, **6**, 21–58.

Department of Economic and Social Affairs: Population Division (2013). "World Population Prospects: The 2012 Revision, Highlights and Advance Tables." *Technical Report ESA/P/WP.228*, United Nations.

Eddelbuettel D, Francois R (2011). "**Rcpp**: Seamless R and C++ Integration." *Journal of Statistical Software*, **40**, 1–18. doi:10.18637/jss.v040.i08.

Glenn EH (2016). ***acs**: Download, Manipulate, and Present American Community Survey and Decennial Data from the US Census*. R package version 2.0, URL https://CRAN.R-project.org/package=acs.

Hamilton BE, *et al.* (2015). "Births: Final Data for 2014." *Technical Report 12*, National Vital Statistics Reports.

Harding A, *et al.* (2003). "Assessing Poverty and Inequality at a Detailed Regional Level: New Advances in Spatial Microsimulation." In *Inequality, Poverty and Human Well-Being Conference*. Helsinki, Finland.

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**, 1–15. doi:10.18564/jasss.1909.

Huang Z, Williamson P (2001). "A Comparison of Synthetic Reconstruction and Combinatorial Optimisation Approaches to the Creation of Small-Area Microdata." *(Working Paper 2001/2). Liverpool: Population Microdata Unit, Department of Geography, University of Liverpool.*

Ingber L (1989). "Very Fast Simulated Re-Annealing." *Mathematical and Computer Modelling*, **12**(8), 967–973. doi:10.1016/0895-7177(89)90202-1.

Ingber L (2000). "Adaptive Simulated Annealing (ASA): Lessons Learned." *CoRR*, **cs.MS/0001018**. URL http://arxiv.org/abs/cs.MS/0001018.

Kavroudakis D (2015). "**sms**: An R Package for the Construction of Microdata for Geographical Analysis." *Journal of Statistical Software*, **68**(2), 1–23. doi:10.18637/jss.v068.i02.

Metropolis N, *et al.* (1953). "Equation of State Calculations by Fast Computing Machines." *The Journal of Chemical Physics*, **21**(6), 1087–1092. doi:10.1063/1.1699114.

Nakaya T, Fotheringham AS, Hanoaka K, Clarke GP, Balals D, Yano K (2007). "Combining Microsimulation and Spatial Interaction Models for Retail Location Analysis." *Journal of Geographical Systems*, **4**, 345–369. doi:10.1007/s10109-007-0052-2.

Office of Health Assessment and Epidemiology (2015). "Mortality in Los Angeles County 2012: Leading Causes of Death and Premature Death with Trends for 2003-2012." *Technical report*, Los Angeles County Department of Public Health.

Orcutt G, *et al.* (1961). *Microanalysis of Socioeconomic Systems: A Simulation Study*. Harper.

Pebesma E, Bivand R (2005). *Classes and Methods for Spatial Data in R*. URL http://cran.r-project.org/doc/Rnews/.

Raftery AE, *et al.* (2012). "Bayesian Probabilistic Population Projections for all Countries." volume 109, pp. 13915–13921.

Rao J, *et al.* (1962). "On a Simple Procedure of Unequal Probability Sampling Without Replacement." *Journal of the Royal Statistical Society B (Methodological)*, pp. 482–491.

R Core Team (2016). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Sevcikova H, Li N, Kantorova V, Gerland P, Raftery AE (2015). "Age-Specific Mortality and Fertility Rates for Probabilistic Population Projections." *ArXiv e-prints*. 1503.05215.

Szu H, Hartley R (1987). "Fast Simulated Annealing." *Physics Letters A*, **122**(3), 157–162. doi:10.1016/0375-9601(87)90796-1.

Tanton R, Edwards KL (2013). *Spatial Microsimulation: A Reference Guide for Users*. Springer-Verlag. doi:10.1007/978-94-007-4623-7.

Townend P, *et al.* (2009). "**MoSeS**: Modelling and Simulation for e-Social Science." *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, **367**, 2781–2792. doi:10.1098/rsta.2009.0041.

US Census Bureau (2009). *A Compass for Understanding and Using American Community Survey Data: What Researchers Need to Know*. Washington, DC.

US Census Bureau (2015). "US Census Bureau Quick Facts." Accessed: 2016-09-12, URL https://www.census.gov/quickfacts/.

Van Imhoff E, Post W (1998). "Microsimulation Methods for Population Projection." *Population: An English Selection*, pp. 97–138.

Waddell P, *et al.* (2003). "Microsimulation of Urban Development and Location Choices: Design and Implementation of **UrbanSim**." *Networks and Spatial Economics*, **3**, 43–67. doi:10.1023/A:1022049000877.

Wickham H, Chang W (2016). ***devtools**: Tools to Make Developing R Packages Easier*. R package version 1.10.0, URL https://CRAN.R-project.org/package=devtools.

Williamson P, Birkin M, Rees P (1998). "The Estimation of Population Microdata by Using Data from Small Area Statistics and Samples of Anonymised Records." *Environment and Planning A*, **30**, 785–816. doi:10.1068/a300785.

Xu JQ, Murphy S, Kochanek KD (2015). "Deaths: Final Data for 2013." *Technical Report 2*, National Vital Statistics Reports.

Affiliation:

Alex Whitworth
Seattle, WA 98109
E-mail: whitworth.alex@gmail.com