An introduction to spatial microsimulation with R

Lovelace, Robin
r.lovelace@leeds.ac.uk

April 7, 2014

1 Introduction

2 Building a spatial microsimulation model in R

Computer hardware has long influenced, and even *determined* the types of analysis that can be conducted at the individual level. Hardware limitations are far less of a constraint than they used to be, elevating the importance of software. As Clarke and Holm (1987) made clear more than 20 years ago, the choice of software also has a major impact on the model's flexibility, efficiency, reproducibility and ease of coding. It was noted that "little attention is paid to the choice of programming language used" (Clarke and Holm, 1987, p. 153), an observation that appears to be as true now as it was then. For this research, a conscious decision was made early on to use R, and this has had an impact on the model construction, features, analysis and even design philosophy. It is at this stage, therefore, that R as a platform for undertaking spatial microsimulation is discussed in some detail. The theory is discussed in section 2.2

2.1 Why R?

The majority of the quantitative analysis conducted for this thesis, and the entirety of the spatial microsimulation model used, was written in R. This was a deliberate choice made at the outset rather than an arbitrary decision based on predecessors. This section briefly explains the importance of choosing appropriate computer software in academic research in general, with respect to reproducibility, a cornerstone of science. The choice of R in particular is then described. R was chosen for its virtues, which are summarised well in **?**:

- "a public-domain implementation of the widely-regarded S statistical language; R/S is the de facto standard among professional statisticians
- comparable, and often superior, in power to commercial products in most senses

- available for Windows, Macs, Linux
- in addition to enabling statistical operations, it's a general programming language, so that you can automate your analyses and create new functions
- object-oriented and functional programming structure
- your data sets are saved between sessions, so you don't have to reload each time
- open-software nature means its easy to get help from the user community"

? also provides five examples of the type of people who would be interested in programming in R, rather than using it as a quick and easy tool for graphing and numerical analysis. Of particular relevance to this thesis is the second of Matloff's categories of people for whom R is recommended: "Academic researchers developing statistical methodology that is either new or combines existing methods into an integrated procedure that needs to be codified for usage by the general research community" (**?**, p. xiii).

The quote also suggests some of the potential advantages of writing multi-use scripts in R rather than a collection of unrelated functions: by its very nature modelling is an iterative exercise, so it is important to be able to invoke specific chunks of code (e.g. using the `source()` command) that are modular. While this capability is not unique to R, the range of statistical functions that can be performed within a unified environment is. The rapidly growing use of R for spatial data analysis was another factor that makes it well-suited to spatial-microsimulation and other types of geographic modelling (e.g. **?**). R overcomes the need to switch between several different programs (e.g. one for analysis, one for graphing, one for mapping), increasing simplicity and (eventually) productivity.

Despite all these advantages, R has a number of weaknesses itemised below along with techniques and projects which mitigate them:

- R loads everything into RAM. This can be problematic when querying large datasets, of which only one part needs to be accessed at a time.[1] There are numerous tools that overcome this constraint by querying databases (stored on the hard-disk) from within R, including RMySQL (**?**) and Rattle (**?**). **?** queried a PostGIS database from within R to estimate the route taken by school commuters, for the estimation of associated CO_2 emissions.

[1] This is especially common with geographical analysis, which often focus on a small area of a large map at a time (**?**).

- R can be slow, for example running for loops and when used as a general programming language which is not Rs main purpose. R being an interpreted language there are times when the performance advantages of a compiled language such as C/C++ are needed. To this end the RCPP package was developed, which provides "Seamless R and C++ integration" (**?**). Packages are also available to integrate R with Java (rJava), Python (rpy2) and text markup languages such as Markdown and LaTeX(knitr). Also, the base installation of R provides an inbuilt C compiler for doing the 'heavy lifting' tasks such as kernel density estimation (**?**). These links to other languages could be useful for porting pre-existing algorithms for spatial microsimulation into R (e.g. **?**Ballas et al., 2007).

- R's base graphics are unattractive and unintuitive. This problem has been tackled most comprehensively in a PhD thesis by Hadley Wickham (**?**). The aim was to implement the 'grammar of graphics' (**?**), a comprehensive and coherent approach to data visualisation, into an existing open-source statistical programming language. The result is ggplot2, which has a very active user and developer community (**?**). ggplot2 has been used throughout this thesis for plotting with help from key references (**??**).

- R's visualisations are not dynamic. This problem has been partly overcome in the realm of GIS with two QGIS plugins: ManageR and Home range. For dynamic web applications, the R package Shiny provides similar interactive functionality as Google's Fusion tables project. There is also a nascent interface between R and Processing (rprocessing), an abstraction of Java ideal for dynamic visualisations of geographic data (e.g. **?**).

2.2 IPF theory: a worked example

In most modelling texts there is a strong precedence of theory over application: the latter usually flows from the former. The location of this section after a description of the programming language R is therefore a little unconventional but there is a logic to this order. Having demonstrated the power and flexibility of the programming language in which the model is written, the next stage is to analyse the task to which it is to be set. More importantly for reproducible research, this theory section is illustrated with a simple worked example that culminates in a question to the reader, to test his or her understanding.

IPF is a simple statistical procedure, "in which cell counts in a contingency table containing the sample observations are scaled to be consistent with various externally given population marginals" (McFadden et al., 2006). In other words, and in the context of *spatial* microsimulation, IPF produces maximum likelihood estimates for the frequency with which people appear in different areas. The method is also known as 'matrix raking' or the RAS algorithm, (Birkin and Clarke, 1988; **?**; Simpson and Tranmer, 2005; Kalantari et al., 2008; Jiroušek and Peučil, 1995) and has been described as one particular instance of a more general procedure of 'entropy maximisation' (Johnston and Pattie, 1993; Blien and Graef, 1998). The mathematical properties of IPF have been described in several papers (Bishop et al., 1975; Fienberg, 1970; Birkin and Clarke, 1988). Illustrative examples of the procedure can be found in Saito (1992), Wong (1992) and Norman (1999). Wong (1992) investigated the reliability of IPF and evaluated the importance of different factors influencing the its performance. Similar methodologies have since been employed by Mitchell et al. (2000), Williamson et al. (2002) and Ballas et al. (2005; **?**) to investigate a wide range of phenomena.

To illustrate how IPF works in practice, a simplified example is described below. This is a modified version of a simpler demonstration from **?**.[2] Table 1 describes a hypothetical microdataset comprising 5 individuals, who are defined by two constraint variables, age and sex. Each has two categories. Table 2 contains aggregated data for a hypothetical area, as it would be downloaded from census dissemination portal Casweb. Table 3 illustrates this table in a different form, which shows our ignorance of interaction between age and sex.

Table 1: A hypothetical input microdata set (the original weights set to one). The bold value is used subsequently for illustrative purposes.

Individual	Sex	Age-group	Weight
1	Male	Over-50	1
2	Male	Over-50	1
3	Male	Under-50	**1**
4	Female	Over-50	1
5	Female	Under-50	1

Table 4 presents the hypothetical microdata in aggregated form, that

[2] In **?** the interaction between the age and sex constraints are assumed to be known. (Their equivalent of table 3 contains data for every cell, not question marks.) This results in IPF converging instantly. However, in Census data, such cross-tabulation is often absent, and IPF must converge over multiple constraints and iterations. This latter scenario is assumed in the worked example below. Other worked examples of the principles are provided in Johnston (1985, Appendix 3) (for entropy maximisation), Norman (1999) and Simpson and Tranmer (2005) (using the proprietary statistical software SPSS).

Table 2: Hypothetical small area constraints data (s).

Constraint ⇒	i		j	
Category ⇒	i_1	i_2	j_1	j_2
Area ⇓	Under-50	Over-50	Male	Female
1	8	4	6	6

Table 3: Small area constraints expressed as marginal totals, and the cell values to be estimated.

Marginal totals		j		
	Age/sex	Male	Female	T
i	Under-50	**?**	?	8
	Over-50	?	?	4
	T	6	6	12

can be compared directly to Table 3.

Table 4: The aggregated results of the weighted microdata set ($m(1)$). Note, these values depend on the weights allocated in Table 1 and therefore change after each iteration

Marginal totals		j		
	Age/sex	Male	Female	T
i	Under-50	**1**	1	2
	Over-50	2	1	3
	T	3	2	5

Using these data it is possible to readjust the weights of the hypothetical individuals, so that their sum would add up to the totals given in Table 3 (12). In particular, the weights can be readjusted by multiplying them by the marginal totals, originally taken from Table 2 and then divided by the respective marginal total in 4. Because the total for each small-area constraint is 12, this must be done one constraint at a time. This can be expressed, for a given area and a given constraint (i or age in this case), as follows:

$$w(n+1)_{ij} = \frac{w(n)_{ij} \times sT_i}{mT(n)_i} \tag{1}$$

where $w(n+1)_{ij}$ is the new weight for individuals with characteristics i (age, in this case), and j (sex), $w(n)_{ij}$ is the original weight for individuals with these characteristics, sT_i is element marginal total of the small area constraint, s (Table 2) and $mT(n)_i$ is the marginal total of category j of the aggregated results of the weighted microdata, m (Table 4). n represents the iteration number. Although the marginal totals of s are known, its cell values are unknown. Thus, IPF estimates the interaction (or cross-tabulation) between constraint variables. (Follow the emboldened values in the tables to see how the new weight of individual 3 is calculated for the sex constraint.) Table 5 illustrates the weights that result. Notice that the sum of the weights is equal to the total population, from the constraint variables.

Table 5: Reweighting the hypothetical microdataset in order to fit Table 2.

Individual	Sex	age-group	Weight	New weight, w(2)
1	Male	Over-50	1	$1 \times 4/3 = \frac{4}{3}$
2	Male	Over-50	1	$1 \times 4/3 = \frac{4}{3}$
3	Male	Under-50	1	$\mathbf{1} \times \mathbf{8/2} = 4$
4	Female	Over-50	1	$1 \times 4/3 = \frac{4}{3}$
5	Female	Under-50	1	$1 \times 8/2 = 4$

After the individual level data have been re-aggregated (table 6), the next stage is to repeat eq. (1) for the age constraint to generate a third set of weights, by replacing the i in sT_i and $mT(n)_i$ with j and incrementing the value of n:

$$w(3)_{ij} = \frac{w(2)_{ij} \times sT_j}{mT(2)_j} \tag{2}$$

To test your understanding of IPF, apply eq. (2) to the information above and that presented in table 6. This should result in the following vector of new weights, for individuals 1 to 5:

$$w(3) = (\frac{6}{5}, \frac{6}{5}, \frac{18}{5}, \frac{3}{2}, \frac{9}{2}) \tag{3}$$

As before, the sum of the weights is equal to the population of the area (12). Notice also that after each iteration the fit between the marginal totals of m and s improves. The total absolute error (TAE, see **??** below) from $m(1)$ to $m(2)$ improves from 14 to 6 in table 4 and table 6 above. TAE for $m(3)$ (not shown, but calculated by aggregating $w(3)$) improves even more, to 1.3. This number would eventually converge to 0 through subsequent iterations, as there are no empty cells in the input microdataset; a defining feature of IPF.

The above process, when applied to more categories (e.g. socio-economic class) and repeated iteratively until a satisfactory convergence occurs, results in a series of weighted microdatasets, one for each of the small areas being simulated. This allows for the estimation of variables whose values are not known at the local level (e.g. income) (**?**). An issue with the results of IPF (absent from combinatorial optimisation methods), however, is that it

Table 6: The aggregated results of the weighted microdata set after constraining for age ($m(2)$).

Marginal totals		i		
	Age/sex	Male	Female	T
j	Under-50	4	4	8
	Over-50	$\frac{8}{3}$	$\frac{4}{3}$	4
	T	$6\frac{2}{3}$	$5\frac{1}{3}$	12

results in non-integer weights: fractions of individuals appear in simulated areas. As described in the introduction, this is not ideal for certain applications. Integer weights allow the results of spatial microsimulation to be further processed using dynamic microsimulation and agent based modelling techniques (Pritchard and Miller, 2012).

Spatial microsimulation can also provide insight into the likely distribution of individual level variables about which only geographically aggregated statistics have been made available. An issue with the results of IPF (absent from combinatorial optimisation methods), however, is that it results in non-integer weights: fractions of individuals appear in simulated areas.

2.3 Implementing IPF in R

The above example is best undertaken by hand, probably with a pen and paper to gain an understanding of IPF, before the process is automated for larger datasets. This section explains how the IPF algorithm described above was implemented in R, using a slightly more complex example. (Lovelace and Ballas, 2013).[3]

Loading in the data

In the full model the input datasets are stored as .csv files, one for each constraint and one for the input microdata, and read in with the command `read.csv`. For the purposes of understanding how the model works, the dataset is read line by line, following the example above. The following code creates example datasets, based on the same hypothetical survey of 5 individuals described above, and 5 small areas. The spatial microsimulation model will select individuals based on age and sex and mode of transport (mode of transport is also used on the larger online example described in footnote 3). For consistency with the (larger) model used for the paper, the individual level data will be referred to as USd (Understanding Society dataset) and the geographic data as all.msim (for all constraint variables). The code to read-in the individual level data are presented in code sample

[3]This tutorial is available from Rpubs, a site dedicated to publishing R analyses that are reproducible. It uses the RMarkdown mark-up language, which enables R code to be run and presented within documents. See http://rpubs.com/RobinLovelace/5089 .

1. When called, the data are then displayed as a table (see listing 2). The

Listing 1: Manual input of individual level data in R

```
# Read in the data in long form (normaly read.table() used)
c.names <- c("id", "age", "sex")
USd <- c(        1, 59, "m",
                 2, 54, "m",
                 3, 35, "m",
                 4, 73, "f",
                 5, 49, "f")
USd <- matrix(USd, nrow = 5, byrow = T) # Long data into matrix
USd <- data.frame(USd) # Convert this into a dataframe
names(USd) <- c.names # Add correct column names
USd$age <- as.numeric(levels(USd$age)[USd$age]) # Age is a numeric
```

Listing 2: Output of the USd data frame

```
USd # Show the data frame in R
##   id age sex
## 1  1  59   m
## 2  2  54   m
## 3  3  35   m
## 4  4  73   f
## 5  5  49   f
```

same procedure applies to the geographical data (listing 3).

IPF relies on the assumption that all constraint variables will contain the same number of people. This is logical (how can there be more people classified by age than by sex?) but can cause problems for constraint variables that use only a subset of the total population, such as those who responded to questions on travel to work. To overcome this problem, it is possible to normalise the constraint variables, setting the total for each to the one that has the most reliable total population. This worked example simply checks whether or not they are (listing 4).

Reweighting the survey dataset

Iterative proportional fitting determines the weight allocated to each individual for each zone to best match the geographically aggregated data. A weight matrix is therefore created, with rows corresponding to individuals and columns to zones, as described in section 2.2. In R, this, and the creation of the aggregated results matrix, is done with code presented in listing 5).[4]

[4] In subsequent versions of the model, single, multi-dimensional weight and aggregated result matrices are used, to reduce the length of the scripts.

Listing 3: Geographic data input

```
category.labels <- c("16-49", "50+" # Age constraint
             ,"m", "f" # Sex constraint
             # more constraints could go here
             )
all.msim <- c(  8, 4,    6, 6,
# Original aggregate data
                2, 8,    4, 6,   # Elderly
                7, 4,    3, 8,   # Female dominated
                5, 4,    7, 2,   # Male dominated
                7, 3,    6, 4    # Young
                )
all.msim <- matrix(all.msim, nrow = 5, byrow = T)
all.msim <- data.frame(all.msim) # Convert to dataframe
names(all.msim) <- category.labels # Add correct column names
```

Listing 4: R code to check the constrain populations match

```
# Check totals for each constraint match
rowSums(all.msim[,1:2]) # Age constraint
## [1] 12 10 11  9 10
rowSums(all.msim[,3:4]) # Sex constraint
## [1] 12 10 11  9 10

rowSums(all.msim[,1:2]) == rowSums(all.msim[,3:4])
## [1] TRUE TRUE TRUE TRUE TRUE
```

It is important to note that in real survey data, the variables are not always neatly categorised into the same bins as the levels of the aggregate data. Age, for example can be classified in many different ways. Also, a wide form is useful for subsequent steps. Therefore, it is necessary to convert the 'thin' survey dataset into a wider form, by converting a single column such as age or sex into multiple columns corresponding to the number of categories. Sometimes the cut-off points of the categories can be decided (as with age), or categories can be merged (when many different NA options are available, for example). The code that performs this important process for our example dataset is presented in listing 6.

Another important step shown in section 2.2 was that of converting the 'long' survey dataset into a form that can be compared directly with the aggregated constraint variables. Listing 7 shows how this is done in R, and the code needed to view the results. (Notice that the first row of all.msim is the same as those displayed in table 2)

Listing 5: Creating arrays of weights in R

```
weights0 <- array(dim=c(nrow(USd),nrow(all.msim)))
weights1 <- array(dim=c(nrow(USd),nrow(all.msim)))
weights2 <- array(dim=c(nrow(USd),nrow(all.msim)))

weights0[,] <- 1 # sets initial weights to 1

USd.agg <- array(dim=c(nrow(all.msim),ncol(all.msim)))
USd.agg1 <- array(dim=c(nrow(all.msim),ncol(all.msim)))
USd.agg2 <- array(dim=c(nrow(all.msim),ncol(all.msim)))
colnames(USd.agg1) <- category.labels
```

Listing 6: R code to convert the survey dataset into binary form

```
USd.cat <- array(rep(0), dim=c(nrow(USd),
                          length(category.labels !=0)))

USd.cat[which(USd$age < 50),1] <- 1 # Age, "< 50"
USd.cat[which(USd$age >= 50),2] <- 1 # "50+"
USd.cat[which(USd$sex =="m"),3] <- 1 # Sex constraint: "m"
USd.cat[which(USd$sex =="f"),4] <- 1 #"f"
sum(USd.cat) # Should be 10
```

With the data loaded and processed into comparable formats, one is in a position to start comparing how well our individual level survey dataset fits with the aggregate constraints (see listing 7). Note that for USd.agg, the results are the same for every zone, as each individual has a weight of 1 for every zone. Note also the very poor fit between the variables at the aggregate level, as illustrated by poor correlation between the constraint and microdata variables (r = 0.05), and a plot of the fit presented in fig. 1. The next stage is to apply the first constraint, to adjust the weights of each individual so they match the age constraints (listing 8 — note that the top row USd.agg1 is the same as table 6). After this operation, the fit between the constraint variables and the aggregated microdata are far better (r = 0.67), but there is still a large degree of error (fig. 2).

unnamed-chunk-5

Figure 1: Scatter plot of the fit between census and survey data. This plot can be re-created using the plot command in listing 7.

We will perform the same checks after each constraint to ensure our

Listing 7: R code to aggregate the survey dataset

```
 for (i in 1:nrow(all.msim)){ # Loop creating aggregate values
  USd.agg[i,]    <- colSums(USd.cat * weights0[,i])
}

# Test results
USd.agg

##       [,1] [,2] [,3] [,4]
## [1,]     2    3    3    2
## [2,]     2    3    3    2
## [3,]     2    3    3    2
## [4,]     2    3    3    2
## [5,]     2    3    3    2

all.msim

##   16-49 50+ m f
## 1     8   4 6 6
## 2     2   8 4 6
## 3     7   4 3 8
## 4     5   4 7 2
## 5     7   3 6 4

plot(as.vector(as.matrix(all.msim)),
  as.vector(as.matrix(USd.agg)), xlab = "Constraints",
    ylab = "Model␣output")
abline(a = 0, b = 1)
```

model is improving. To see how the weights change for each individual for each area, one simply types `weights1`, for constraint 1 (listing 9). Note that the first column of weights 1 is the same as table 2.

Listing 9: The new weight matrix. Previously all weights were set to one.

```
##        [,1]  [,2]  [,3]  [,4] [,5]
## [1,] 1.333 2.667 1.333 1.333  1.0
## [2,] 1.333 2.667 1.333 1.333  1.0
## [3,] 4.000 1.000 3.500 2.500  3.5
## [4,] 1.333 2.667 1.333 1.333  1.0
## [5,] 4.000 1.000 3.500 2.500  3.5
```

To further improve the fit, one next constrains by the second aggregate

Listing 8: Reweighting of first constraint and testing of results

```
for (j in 1:nrow(all.msim)) {
 weights1[which(USd$age < 50),j] <- all.msim[j,1]/USd.agg[j,1]
 weights1[which(USd$age >= 50),j] <- all.msim[j,2]/USd.agg[j,2]
}
# Aggregate the results for each zone
for (i in 1:nrow(all.msim)) {
 USd.agg1[i,] <- colSums(USd.cat * weights0[,i] * weights1[,i])
}
# Test results
USd.agg1
##      16-49 50+     m     f
## [1,]     8   4 6.667 5.333
## [2,]     2   8 6.333 3.667
## [3,]     7   4 6.167 4.833
## [4,]     5   4 5.167 3.833
## [5,]     7   3 5.500 4.500

plot(as.vector(as.matrix(all.msim)),
 as.vector(as.matrix(USd.agg1)), xlab = "Constraints",
 ylab = "Model output")
abline(a = 0, b = 1)
```

unnamed-chunk-6

Figure 2: Scatter plot showing the fit after constraining by age.

constraint: sex (listing 10). To check that our implementation in R produces the same results as the hand-calculated example, the resulting weights where queried. As shown by `weights3[,1]`, these are the same as those calculated for $w(3)$ above.

The model fit improves greatly after constraining for sex (r = 0.992). However, to ensure perfect fit more iterations are needed. Iterating just once more, as done on the online version of this section[5] results in a fit that is virtually perfect (fig. 3). More iterations are needed for larger datasets with more constraints to converge.

The worked code example in this section is replicable. If all the code snippets are entered, in order, the results should be the same on any computer running R. There is great scope for taking the analysis further: some further tests and plots are presented on the on-line versions of this section.

[5] See rpubs.com/RobinLovelace/6193

Listing 10: Code to constrain the weights by sex

```
for (j in 1:nrow(all.msim)) {
    weights2[which(USd$sex == "m"),j] <-
                   all.msim[j,3]/USd.agg1[j,3]
    weights2[which(USd$sex == "f"),j] <-
                   all.msim[j,4]/USd.agg1[j,4]
}

weights3 <- weights0 * weights1 * weights2
for (i in 1:nrow(all.msim)) {
  USd.agg2[i,] <- colSums(USd.cat * weights3[,i])
}

weights3[,1]

## [1] 1.2 1.2 3.6 1.5 4.5
```

unnamed-chunk-8 unnamed-chunk-11

Figure 3: Improvement of model fit after constraining by sex (left) and after two complete iterations (right).

The simplest case is contained in Rpubs document 6193 and a more complex case (with three constraints) can be found in Rpubs document 5089. The preliminary checks done on this code are important to ensure the model is understood at all times and is working correctly. More systematic methods for model checking are the topic of the following section.

3 Model checking and validation

The R scripts that implement the methods described in section 2 and **??** contain over 1000 lines of code. This means that making mistakes while writing the code was almost inevitable, from time to time.[6] The large size of the output files (approximately 250 Mb for 10 iterations of the spatial

[6] A couple of examples serve to illustrate this point: during the construction of vulnerability metrics based on the individual level output from the spatial microsimulation model, the estimated expenditure on commuting was divided by equivalised household income (a proxy of disposable income). One issue was that trip cost estimates are per year while the income estimates are supplied per month in the USd. It took several more alterations and runs of the model before the cause of the high proportion of income spent on commuting (sometimes over 100%) was realised. Another example is simple typing errors while writing the code. The results are presented in fig. 5, and are described below.

microsimulation model for Yorkshire and the Humber) means that it would be easy to miss fundamental errors. Hence the need for a systematic strategy of checking the output. Beyond checking the model's internal validity, it is necessary to test its external validity. This process, validation, is inherently limited by lack of real spatial microdata. Validation is a crucial step to take before the results are presented, discussed and used as the basis of policy guidance. To make an analogy with corporate food safety standards, it is important be open about and highlight times when things do go wrong, in order to achieve high standards (**?**). Transparency is needed in modelling for similar reasons (**?**). This section is therefore an overview of the methods used to find fault in the model, rather than assuming that everything is working perfectly as the rest of the thesis does. It is divided into two halves: first the process of comparing the model results with knowledge of how it *should* perform *a-priori* (model checking). Second, the internally consistent model results are compared with external empirical data (validation). Validation is also discussed in the context of a single case study in **??**.

3.1 Model checking

A proven method of checking that data analysis and processing is working is wide ranging and continual visual exploration of its output (**?**). This strategy has been employed throughout the modelling process, both to gain a better understanding of the behaviour of the underlying R code, and to search for unexpected results. These were often precursors to error identification.

An example of this, that illustrates the utility of ad-hock checks, is the continual plotting of model inputs and outputs to ensure that they make sense. The R commands `summary()` and `plot()` are ideal for this purpose. The former provides basic descriptive statistics; the latter produces a graphical display of the object. Both are *polymorphic*, meaning that command adapts depending on the type of object it has been asked to process (**?**). Thus, to check that the number of people in each age and sex category in the input and output dataset made sense overall, the following command was issued, resulting in the plot illustrated in fig. 4:

```
plot(cut(USd$age, breaks=(seq(0,100,20))), USd$sex)
```

age-sex-plot

Figure 4: Diagnostic plot to check the sanity of age and sex inputs. (Square brackets indicate that the endpoint is not included in the set — see International Organization for Standardization (ISO) 80000-2:2009, formerly ISO 31-11 on "mathematical signs and symbols for use in physical sciences and technology").

These common-sense methods of data checking may seem overly simplistic to warrant mention. Yet such basic sanity tests are the 'bread-and-butter' of quantitative analysis. They ensure that the data are properly understood (?). Had the input data represented in fig. 4 contained an equal proportion of people under 20 as over 20, for example, one would know that the input data for commuters was faulty. This approach, whereby major problems are revealed early on in frequent tests, is preferable to waiting until the results of the full spatial microsimulation are analysed. Hours were saved, and understanding of the input datasets was improved.[7]

The basic tenet of spatial microsimulation is that simulated and actual data should match at the aggregate level (Ballas et al., 2007). This knowledge led to the continual plotting of census vs simulated results in the early stages of the model construction, and the development of more sophisticated plots (see **??**). Still, the humble scatter plot was used frequently for preliminary analysis. To provide an example, after the model was run for Yorkshire and the Humber region for 20 iterations, I was confident the results were correct: the results had been tested for Sheffield, and everything *seemed* to be working as expected.

Knowledge of how model-census fit should look started alarm bells ringing when an imperfect plot was discovered: fig. 5 (A) was cause for concern, not only for the low correlation between the two variables (which was still greater than 0.8), but because the direction of the error: the model had *always* overestimated the number of people travelling short distances to work in past runs. This seemed suspicious, and the relationship was plotted for earlier constraints to identify where the problem was variables were plotted. fig. 5 (B) was the result of this, after constraining by distance. Something had clearly gone wrong because no people who work from home had been registered in the aggregate output. These issues led to a re-examination of the code contained within the file cats.r. It was found that a faulty placement of an equals sign (such that values "greater than or equal" to 0 were accepted as 0 - 2 km travel to work). The problem was solved, and the model correlation improved as a result (fig. 5 (C)).

The two examples described above provided insight into how the model was performing by its own standards. The more challenging stage is to validate the model against factors external to it.

3.2 Model validation

Beyond 'typos' or simple conceptual errors in model code, more fundamental questions should be asked of spatial microsimulation models. The validity of the assumptions on which they are built, and the confidence one should

[7] The use of the same command to check model output was crucial to the identification of important errors, including a small mistake in the code which led to large errors in the synthetic microdata output for the distance constraint variables.

errors3-1

Figure 5: Three diagnostic plots used to identify a code error in the spatial microsimulation model (for the distance category 'travels 0–2 km to work'). The x-axis is census data, the y-axis is the simulated result. A) First plot analysed (for iteration 20); B) second plot, which illustrated the source of the problem, in the distance constraint; C) satisfactory diagnostic plot, after the problem had been resolved.

have in the results are important. This is especially true of models designed to inform policies which have the potential to influence quality of life. Yet evaluation and 'validation' are problematic for any models that attempt to explain extensive, complex systems such as cities or ecosystems. The urban modelling approach, of which spatial microsimulation of commuters is a subset, has been grappling with this problem since its infancy. Lacking a crystal ball, time-machine or settlements on which controlled experiments can be performed, the difficulty of model evaluation can seem intractable: "only through time can a model be verified in any conventional sense of the word", by comparing the range of projected futures with the reality of future change in hindsight (**?**, p. 15).

Why do urban models pose such a problem? Previously unknown knock-on impacts cannot be ruled out due to the vast number of links between system elements.[8] Rigorous real-world testing is usually impossible due to the scale of the system and ethics involved with intervening in peoples' lives for the sake of research. Controlled experiments cannot be performed on real settlements in the same way that experiments can be performed in the physical sciences and, even if two similar settlements could be found on which to apply different interventions, there is no guarantee that all other factors will be held constant throughout the duration of the experiment.

Additional evaluation problems apply to spatial microsimulation models in particular for a number of reasons, including:

- The aggregate values of categorical 'small area' constraint variables are already known from the Census, so should be accurate. Checking the distribution of continuous variables such as age and distance travelled to work against these crude categories is problematic.[9]

[8] It is, of course, impossible to know how every resident of an area interacts with every other, let alone predict the future impacts of this interaction, even in the era of ubiquitous digital communications.

[9] For example, if 50% of commuters in a particular area travel 2–5 km to work according to the Census, does that mean that there is a normal distribution of trip distances with the mean focussed on 3.5? Or is it more likely that there is a single large employer located somewhere between 2 and 5 km from the bulk of houses in the area, which accounts for the majority of these jobs and leads to a skewed distribution of home-work distances. In

- Target variables are not generally known as geographic aggregates. Therefore checking their validity for small areas is difficult: new surveys may be needed.

- Spatial microsimulation results in long lists of individuals for each zone. With thousands of individuals in each zone and hundreds of zones, the datasets can become large and unwieldy.

Regarding the target variables, inaccuracies can be expected because they are determined entirely by their relationships with constraint variables. Also it can be expected these relationships will not remain constant for all places: perhaps in one area the number of female drivers is positively correlated to distance travelled to work, yet there may be a different strength of correlation, or the variables may be unrelated in another.

As mentioned above, validation of target variables is especially problematic due to lack of data. To overcome this problem, two techniques were employed. First, the interaction between constrained variables and unconstrained variables was tested using data from the Census. Second, an additional dataset from the UK's National On-line Manpower Information System (Nomis) was harnessed to investigate the correlation between unconstrained 'interaction' variables — those composed of two or more constraint variables such as 'female driver'.

The first approach tested the model's ability to simulate income. Although income data are lacking for small areas, Neighbourhood Statistics provides estimates of net and gross household incomes at the MSOA level. For the purposes of this study, equivalised net income was used. The fit between the Neighbourhood Statistics and simulated values are displayed in fig. 6.

Income-check-morevars

Figure 6: Scatter plot illustrating the correlation between mean income simulated from the model and official estimates at the MSOA leve.

The results show the microsimulation model could be used to predict income (modelled income), accounting for almost 80% of the variation in the Neighbourhood Statistics data using an ordinary least squares (OLS) regression model. This is impressive, given that the aim of the model is not to simulate income but energy costs of work travel, based on mode, distance, age/sex and class. Of these socio-economic class is the only constraint variable traditionally thought to be closely associated with income. The main problem with the income estimates generated through spatial microsimulation is the small range of estimates simulated: the standard deviation was

every event, spatial microsimulation will ignore such subtleties and smooth out extreme skewness by approximating the national distance trends within each distance bin.

£1,194 and £3,596 for the simulated and National Statistics data respectively. (Note the differences in the x and y axis scales in fig. 6.) This underestimation of variance can be explained because social class, distance and modes of transport are not sufficient to determine the true variability in household incomes. Constraining by car ownership and tenure variables would be likely to improve the fit.

The purpose of this fitting exercise is not so much to provide accurate income estimates at the local level but to evaluate the performance of the spatial microsimulation model. In terms of income, a variable that is unconstrained in the model yet available from the survey data, the spatial microsimulation model has worked well. The results suggest that the values of unconstrained variables will not simply repeat the national average for every small area, but will vary based on how their variation at the national level is related to the constraint variables. In this case, the assumption that the relationships between the target variable (income) and constraint variables at the local level (in Yorkshire and the Humber) are similar to the relationships between these variables at the national level, receives support. How well does the model simulate other target variables such as environmental habits, domestic energy use and levels of deprivation? These are interesting questions that merit further attention based on available data.

The second approach relies on Nomis, which provides cross-tabulations of census variables, for example transport mode by class. The downside is that the data are randomised, as stated at the bottom of each of their small-area census tables: "Figures have been randomly adjusted to avoid the release of confidential data" (this phrase appears in many of Nomis's tables. One example can be found here: http://www.nomisweb.co.uk/livelinks/4652.xls).

In order to harness Nomis data to test the accuracy of the microsimulation model for calculating, it was first necessary to establish how accurate Nomis data are. How much have Nomis data been randomised, and in what way? This question is relatively easy to answer because of the census variables shared between those published by Nomis and by Casweb at the MSOA level. Scatter plots suggest Nomis data are faithful to the original census results:

nomis-vs-cens-car nomis-vs-cens-bike

Figure 7: Scatter graphs illustrating the fit between Nomis and Casweb versions of the same census variables. The correlation (Pearson's r) is 0.9998 and 0.9969, for the number of car drivers and number of cyclists in each MSOA respectively.

From fig. 7 it is interesting to note that the correlation decreases for cyclists. This, it was inferred, could represent an increase in the signal-to-noise ratio for variables with small values to a fixed randomising factor.

To test this, the errors were plotted for variables with large (car drivers) and small (cyclists) totals. The results indicate that the noise added by randomisation is equal for each variable, regardless of the cell count (fig. 8).

car-ers bike-ers

Figure 8: Errors (Casweb values – Nomis values) associated with car driver (right) and bicycle commuter (left) census variables.

The errors seem to be similar, with a range of approximately 70 and a mean of zero. This observation is confirmed by descriptive statistics for each set of errors (standard deviation = 11.01, 9.47; mean = 0.15, 0.23) for car driver and cyclist variables respectively. We can therefore conclude that the error added by randomisation is constant for each variable and this was confirmed by plotting the errors for additional census variables. Q-Q plots — which compare the quantile values of one distribution against another, in this case those of the errors against those of the normal distribution — suggest that the distribution of error is approximately normal.

These exploratory methods provide confidence in the Nomis data, but only for relatively large cell counts (the signal-noise ratio approaches 1:1 as the cell count approaches 20): therefore evaluations based on Nomis data are better suited to cross tabulated categories that have high cell counts, for example car drivers. In our microsimulation model, both gender and mode of transport are constrained, but not simultaneously, so the fit between the Nomis cross-tabulation and the cross-tabulation resulting from our model provides some indication of accuracy. The results are presented in fig. 9. Interestingly, the accuracy of this 'partially constrained' simulated target variable appears to be worse than that of the completely unconstrained income variable (compare fig. 9 and fig. 6). In both cases, the correlation is reasonably strong and positive (0.47 and 0.80 respectively). However, as with the income estimates, the *distribution* of estimates arising from the model is less dispersed than actual data: the standard deviation for the former (0.30) is substantially less than for the latter (0.44). This illustrates the tendency of spatial microsimulation models to underestimate the extent of spatial variation.

no-vs-sim-mcar

Figure 9: Scatter plot of the proportion of male drivers in each MSOA area in Yorkshire and the Humber according to simulated and Nomis data.

3.3 Additional validation methods

The methods described above illustrate the techniques used to prevent model errors and ensure that the results were compatible with external data sources. But they only scratch the surface of what is possible in terms of model validation. This section will not go into detail. Its purpose is to draw attention to additional methods that could be conducted as lines of future research and discuss the merits of each. Specifically, the following additional validation methods could (given sufficient resources) be implemented:

- Primary data collection of target variables at the individual level in specific areas to validate the spatial microdata locally.
- Comparing of the spatial microdata over entire region with a survey data that specifies home region of resident.
- Aggregating local model outputs to coarser geographical levels at which cross-tabulated data are available.
- Comparison of mode and distance data with external correlates of personal travel (e.g. MOT data on distance travelled and bus usage data).

Other than the sanity check of age-sex ratios presented in fig. 4, the evaluation methods considered above operate at the level of geographically aggregated counts. However, the unique feature of spatial microsimulation is its simulation of individuals. Evaluation techniques should therefore operate at the individual level as well. Because simulation, almost by definition, estimates something that is not otherwise known, it is hard to find reliable individual level data against which the estimates can be evaluated. For this reason individual level surveys could be conducted in a specific area where spatial microdata have been generated. To take one example, a randomised sample of households could be taken in a single ward. Respondents would be asked the mode of travel to work, distance and frequency of trip and other variables. This would allow the model to be evaluated not only in terms of the correlations that it outputs between different categories, but also for the evaluation of the assumptions on which the energy calculations are based.

One of the main advantages of spatial microsimulation over just using aggregated data is that it provides insight into the *distribution* of continuous variables within each zone, rather than just counts of categories which are often rather coarse. T-tests and Analysis of Variance (ANOVA) tests could then be used to check if the mean and variance of the simulated and survey data are statistically likely to be from the same population. However, the raw results of IPF are not conducive to such tests at the individual level because they do not contain whole individuals. Integerisation of the weight matrices is needed.

4 Integerisation

5 Extensions to the basic model

The results show that TRS integerisation outperforms the other methods of integerisation tested in this section. At the aggregate level, accuracy improves in the following order: simple rounding, inclusion threshold, counter-weight, proportional probabilities and, most accurately, TRS. This order of preference remains unchanged, regardless of which (from a selection of 5) measure of goodness-of-fit is used. These results concur with a finding derived from theory — that "deterministic rounding of the counts is not a satisfactory integerization" (Pritchard and Miller, 2012, p. 689). Proportional probability and TRS methods clearly provide more accurate alternatives.

An additional advantage of the probabilistic TRS and proportional probability methods is that correct population sizes are guaranteed.[10] In terms of speed of calculation, TRS also performs well. TRS takes marginally more time than simple rounding and proportional probability methods, but is three times quicker than the threshold and counter-weight approaches. In practice, it seems that integerisation processing time is small relative to running IPF over several iterations. Another major benefit of these non-deterministic methods is that probability distributions of results can be generated, if the algorithms are run multiple times using unrelated pseudo-random numbers. Probabilistic methods could therefore enable the uncertainty introduced through integerisation to be investigated quantitatively (Beckman et al., 1996; Little and Rubin, 1987) and subsequently illustrated using error bars.

Overall the results indicate that TRS is superior to the deterministic methods on many levels and introduces less error than the proportional probabilities approach. We cannot claim that TRS is 'the best' integerisation strategy available though: there may be other solutions to the problem and different sets of test weights may generate different results.[11] The issue will still present a challenge for future researchers considering the use of IPF to generate sample populations composed of whole individuals: whether to use deterministic or probabilistic methods is still an open question (some may favour deterministic methods that avoid psuedo-random numbers, to

[10] Although the counter-weight method produced the correct population sizes in our tests, it cannot be guaranteed to do so in all cases, because of its reliance on simple rounding: if more weights are rounded up than down, the population will be too high. However, it can be expected to yield the correct population in cases where the populations of the areas under investigation are substantially larger than the number of individuals in the survey dataset.

[11] Despite these caveats, the order of accuracy identified in this section is expected to hold in most cases. Supplementary Information (Section 4.4), shows the same order of accuracy (except the threshold method and counter-weight methods, which swap places) resulting from the integerisation of a different weight matrix.

ensure reproducibility regardless of the software used), and the question of whether combinatorial optimisation algorithms perform better has not been addressed.

Our results provide insight into the advantages and disadvantages of five integerisation methods and guidance to researchers wishing to use IPF to generate integer weights: use TRS unless determinism is needed or until superior alternatives (e.g. real small area microdata) become available. Based on the code and example datasets provided in the Supplementary Information, other are encouraged to use, build on and improve TRS integerisation.

A broader issue raised by this research, that requires further investigation before answers emerge, is 'how do the integerised results of IPF compare with combinatorial optimisation approaches to spatial microsimulation?' Studies have compared non-integer results of IPF with alternative approaches (Smith et al., 2009; **?**; **?**; Harland et al., 2012). However, these have so far failed to compare like with like: the integer results of combinatorial approaches are more useful (applicable to more types of analysis) than the non-integer results of IPF. TRS thus offers a way of 'levelling the playing field' whilst minimising the error introduced to the results of deterministic reweighting through integerisation.

In conclusion, the integerisation methods presented in this section make integer results accessible to those with a working knowledge of IPF. TRS outperforms previously published methods of integerisation. As such, the technique offers an attractive alternative to combinatorial optimisation approaches for applications that require whole individuals to be simulated based on aggregate data.

References

Ballas, D., Clarke, G., Dorling, D., Eyre, H., Thomas, B., Rossiter, D., 2005. SimBritain: a spatial microsimulation approach to population dynamics. Population, Space and Place 11, 13–34.

Ballas, D., Clarke, G., Dorling, D., Rossiter, D., 2007. Using SimBritain to model the geographical impact of national government policies. Geographical Analysis 39, 44–77.

Beckman, R., Baggerly, K., McKay, M., 1996. Creating synthetic baseline populations. Transportation Research Part A: Policy and Practice 30, 415–429.

Birkin, M., Clarke, M., 1988. SYNTHESIS – a synthetic spatial information system for urban and regional analysis: methods and examples. Environment and Planning A 20, 1645–1671.

Bishop, Y., Fienberg, S., Holland, P., 1975. Discrete Multivariate Analysis: theory and practice. MIT Press, Cambridge, Massachusettes.

Blien, U., Graef, F., 1998. Entropy optimizing methods for the estimation of tables, in: Classification, Data Analysis, and Data Highways. Springer, pp. 3–15.

Clarke, M., Holm, E., 1987. Microsimulation methods in spatial analysis and planning. Geografiska Annaler. Series B. Human Geography 69, 145–164.

Fienberg, S., 1970. An iterative procedure for estimation in contingency tables. The Annals of Mathematical Statistics 41, 907–917.

Harland, K., Heppenstall, A., Smith, D., Birkin, M., 2012. Creating Realistic Synthetic Populations at Varying Spatial Scales: A Comparative Critique of Population Synthesis Techniques. Journal of Artificial Societies and Social Simulation 15, 1.

Jiroušek, R., Peučil, S., 1995. On the effective implementation of the iterative proportional fitting procedure. Computational Statistics & Data Analysis 19, 177–189.

Johnston, R.J., 1985. The geography of English politics: the 1983 general election. Croom Helm London.

Johnston, R.J., Pattie, C.J., 1993. Entropy-Maximizing and the Iterative Proportional Fitting Procedure. The Professional Geographer 45, 317–322.

Kalantari, B., Lari, I., Ricca, F., Simeone, B., 2008. On the complexity of general matrix scaling and entropy minimization via the RAS algorithm. Mathematical Programming 112, 371–401.

Little, R., Rubin, D., 1987. Statistical analysis with missing data. Wiley Series in Probability and Statistics. 1st edition.

Lovelace, R., Ballas, D., 2013. Truncate, replicate, sample: A method for creating integer weights for spatial microsimulation. Computers, Environment and Urban Systems 41, 1–11.

McFadden, D., Heiss, F., Jun, B.h., Winter, J., 2006. On testing for independence in weighted contingency tables. Medium for Econometric Applications 14, 11–18.

Mitchell, R., Shaw, M., Dorling, D., 2000. Inequalities in Life and Death: What If Britain Were More Equal? Policy Press, London.

Norman, P., 1999. Putting Iterative Proportional Fitting (IPF) on the Researchers Desk. Technical Report October. School of Geography, University of Leeds.

Pritchard, D.R., Miller, E.J., 2012. Advances in population synthesis: fitting many attributes per agent and fitting to household and person margins simultaneously. Transportation 39, 685–704.

Saito, S., 1992. A multistep iterative proportional fitting procedure to estimate cohortwise interregional migration tables where only inconsistent marginals are known. Environment and Planning A 24, 1531–1547.

Simpson, L., Tranmer, M., 2005. Combining sample and census data in small area estimates: iterative proportional fitting with standard software. The Professional Geographer 57, 222–234.

Smith, D.M., Clarke, G.P., Harland, K., 2009. Improving the synthetic data generation process in spatial microsimulation models. Environment and Planning A 41, 1251–1268.

Williamson, P., Mitchell, G., McDonald, A.T., 2002. Domestic Water Demand Forecasting: A Static Microsimulation Approach. Water and Environment Journal 16, 243–248.

Wong, D.W.S., 1992. The Reliability of Using the Iterative Proportional Fitting Procedure. The Professional Geographer 44, 340–348.

Index