# Inference for Populations: Uncertainty Propagation via Bayesian Population Synthesis

Christopher Grubb

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Statistics

Leanna L. House, Co-chair
David M. Higdon, Co-chair
Jennifer Van Mullekom
Jyotishka Datta

July 31, 2023
Blacksburg, Virginia

Keywords: Bayesian Statistics, Synthetic Populations
Copyright 2023, Christopher Grubb

# Inference for Populations: Uncertainty Propagation via Bayesian Population Synthesis

Christopher Grubb

(ABSTRACT)

In this dissertation, we develop a new type of prior distribution, specifically for populations themselves, which we denote the Dirichlet Spacing prior. This prior solves a specific problem that arrises when attempting to create synthetic populations from a known subset: the unfortunate reality that assuming independence between population members means that every synthetic population will be essentially the same. This is a problem because any model which only yields one result (several very similar results), when we have very incomplete information, is fundamentally flawed. We motivate our need for this new class of priors using Agent-based Models, though this prior could be used in any situation requiring synthetic populations.

# Inference for Populations: Uncertainty Propagation via Bayesian Population Synthesis

Christopher Grubb

(GENERAL AUDIENCE ABSTRACT)

Typically, statisticians work with parametric distributions governing independent observations. However, sometimes operating under the assumption of independence severely limits us. We motivate the move away from independent sampling via the scope of Agent-based Modeling (ABM), where full populations are needed. The assumption of independence, when applied to synthesizing populations, leads to unwanted results; specifically, all synthetic populations generated from the sample sample data are essentially the same. As statisticians, this is clearly problemated because given only a small subset of the population, we clearly do not know what the population looks like, and thus any model which always gives the same answer is fundamentally flawed. We fix this problem by utilizing a new class of distributions which we call spacing priors, which allow us to create synthetic populations of individuals which are not independent of each other.

# Dedication

*This work is dedicated to the late Dr. G. Robert Himmer Jr., my grandfather, who unfortunately passed away a few short weeks prior to my decision to pursue a doctorate. I sometimes whether that was purely a concidence.*

# Acknowledgments

A number of former teachers and professors have had a hand in guiding me in this direction. In particular, I would like to thank Mr. Kevin Mikula and Mr. David Hively from Red Lion Area School District. Both of them not only did an exemplary job teaching mathematics, but also clearly loved the subject, which rubbed off on me. From my undergraduate studies at Millersville University, I would like to thank Dr. Bruce Ikenaga and Dr. James Fenwick; both of them had a huge impact on where I am today. It is very unlikely that I would have considered taking the GRE if not for Dr. Ikenaga, who believed I could do quite well on the exam. Only after taking the exam, and doing as well as he predicted, did I realize that this option was actually a possiblity for me. Dr. Fenwick had an enormous part to play in changing my trajectory from pure mathematics to statistics, and is also one of the main reasons I ended up at Virginia Tech, which I am eternally thankful for.

I would also like to thank my entire committee, and especially Dr. Jennifer Van Mullekom, for giving me such an incredible opportunity to work at SAIG for as long as I was williong, and recommending me for several special projects which led me to several publications and certainly influenced my career trajectory greatly.

# Contents

# List of Figures

# Preface

While the main focus of this document is of course my dissertation, this document also represents my attempt to transition from a traditional **LaTeX** dissertation towards something more modern. When I first learned about the bookdown(Xie 2022) package for **R**, I was very excited at the prospect of using it to create my dissertation. Thus, I was able to (with only one corpus) create both a pdf document for the official submission as well as a gitbook format for hosting on the web. While bookdown is very well documented, thanks to the wonderful Yuhui Xie (Xie 2015, 2016), the process was not without difficulty. Several factors were to blame for most of this difficulty:

- Getting the pdf output to match the required university format *without* butchering the simplicity and elegance of a gitbook. This was, at times, rather annoying. Outside of the obvious, like needing lists of tables and figures and a section for acknowledgements and dedications for the pdf but not the html, getting the appendices to match the required format for the pdf document and still look pretty in the html was a nightmare. The only solution that I found was to use an excessive number of coded *if-else* statements.
- Hosting the gitbook somewhere that my advisors and I could access for edits, but not be accessible to the general public. **Git** is truly amazing, and the newly introduced git−pages made my academic life much easier than it would have been just a few years ago. However, they decided to restrict private git−pages to enterprise accounts, so even professional accounts cannot use that feature. I ended up having to do the hosting myself; luckily, I have a server to actually do this, otherwise I would have had to pay for hosting elsewhere.
- **R** packages are constantly changing. One of the more frustrating issues I encountered

during this process were things that worked one week, and then caused errors (or, even worse, failed but did not throw an error) the next week. There are of course ways to deal with this, like never updating **R** packages or using a static repository such as **MRAN**, but you inevitably end up finding reason to upgrade your packages, and then something will break.

Despite these issues, I still feel that forcing myself to use bookdown helped me in the long run. There are of course other ways to create an online book, just like there are ways to make **R**-based web apps without using shiny(Chang et al. 2021), but packages such as these give researchers and industry professionals that are familiar with **R** a way to create professional documents and apps while still being relatively easy to use.

# Chapter 1

# Introduction

This effort seeks to develop an initial framework for carrying out formal Bayesian inference of populations, explicitly; e.g., inference on all observational-units jointly in a defined population, not only on population parameters. To do so, we consider an entire population to be an unknown, high-dimensional random variable about which we learn from multiple data sources, such as sampled observations and population data summaries. Using our methods, we will estimate high-dimensional posterior distributions for entire populations.

Notably, inference on a population is different from inference on population parameters. Parametric inference – which has long been the focus of both Bayesian and classical statistics, going back to Laplace in the late 1700's and Fisher in the early 1900's (Stigler 1986; Stephen E. Fienberg 2006) – certainly has advantages and limitless applications. In the recent past, however, solutions to a subset of modern problems have conditioned on simulated (i.e., synthetic) populations explicitly and methods to estimate uncertainty in these solutions are underdeveloped. For example, agent-based models (ABMs) have been developed in the field of epidemiology to assess the occurrence, distribution, and influential factors of disease (Beckman, Baggerly, and McKay 1996; Zhu and Ferreira Jr 2014; Gallagher et al. 2018). These ABMs are typically seeded with a synthetic population of people, households, and/or other observational units within a defined geographical region, e.g. a town or city. Predictions from these population-conditioned ABMs have shown effective, but uncertainty

in these predictions is not fully understood (Eubank et al. 2010). Estimating uncertainty sourced from changes in synthetic populations is a hard, high-dimensional challenge that this dissertation addreses.

This dissertation builds from a common, but flawed approach that relies on iterative proportional fitting (IPF) (Beckman, Baggerly, and McKay 1996). IPF is explained in section 2.2. Previous work has shown that predictions from ABMs clearly vary with alternative population specifications from IPF. But, analyses of this variation do not result in principled characterizations of the uncertainty in the predictions which sources from uncertainty in synthetic populations. Thus, this dissertation starts with Bayesian principles to better characterize the uncertainty in synthetic population estimates - which may then propagate to uncertainty in AGM predictions in principled ways.

## 1.1 The Bayesian Set-up

Consider a realization $\mathbf{y}$ of population $\mathbf{Y}$, and observed data $\mathbf{x}$. We aim to estimate the high-dimensional posterior distribution of $\mathbf{Y}$, from the joint posterior distribution of $\mathbf{y}$ and $\theta$:

$$
\begin{aligned}
f(\theta, \mathbf{Y} = \mathbf{y} | \mathbf{X} = \mathbf{x}) &= \frac{f(\mathbf{X} = \mathbf{x} | \theta, \mathbf{Y} = \mathbf{y}) f(\theta, \mathbf{Y} = \mathbf{y})}{f(\mathbf{X} = \mathbf{x})} \\
&\propto f(\mathbf{X} = \mathbf{x} | \mathbf{Y} = \mathbf{y}, \theta) f(\theta, \mathbf{Y} = \mathbf{y}) \qquad (1.1) \\
&\propto f(\mathbf{X} = \mathbf{x} | \mathbf{Y} = \mathbf{y}, \theta) f(\mathbf{Y} = \mathbf{y} | \theta) f(\theta) \qquad (1.2)
\end{aligned}
$$

Furthermore, we may sometimes integrate out $\theta$ to use the posterior predictive distribution

$$f(\mathbf{Y} = \mathbf{y}|\mathbf{X} = \mathbf{x}) = \int f(\theta, \mathbf{Y} = \mathbf{y}|\mathbf{X} = \mathbf{x})d\theta. \tag{1.3}$$

With this basic, Bayesian reasoning, we will develop a probability model for the population at the finest, individual-level scale. A challenge with this approach relates to choosing reasonable high-dimensional priors for $\mathbf{Y}$. Such priors will depend on context and are developed in Chapter 3. For example, in some contexts, we may enumerate all possible populations and directly apply a prior mass to each possible population. Whereas, in other contexts, enumerating all possible populations is impossible, and a hierarchical prior works better. Similarly, in some contexts we may assume members of populations are independent and in other contexts we may not.

Developing a population probability model at the individual-level offers a natural resolution at which to combine observed information at different levels of aggregation. In Chapter 4, we scale methods from Chapter 3 to include different forms of data $\mathbf{x}$. Meaning, data $\mathbf{x}$ may include observations from population $\mathbf{y}$ and/or additional, aggregated information about $\mathbf{y}$. Then, in Chapter 5, we scale methods from Chapter 3 again to simulate multivariate observations in population $\mathbf{y}$.

## 1.2 A Simple, Motivating Example

To give a more concrete basis for our general approach, consider an unknown population $\mathbf{y}$ of size $N = 30$, where each member is described by either yellow or blue,

$$\mathbf{y} = (y_1, \dots, y_N) \in \{yellow, blue\}^N, \ N = 30.$$

Also, each population member $y_i$ is located sequentially along a one-dimensional space, where $i = 1$ at the leftmost location, and $i = N$ at the rightmost location. Using a stratified by region random sample $\mathbf{x}$, we aim to learn about

$$f(\mathbf{y}|\mathbf{x}) \propto f(\mathbf{x}|\mathbf{y})f(\mathbf{y}).$$

To start, we specify the prior $f(\mathbf{y})$ to be uniform over all $2^N$ possible population configurations. Thus, $\pi(y) = 2^{-N}$. Draws from this prior are shown in Figure 1.1. Note, this particular prior is simple and works fine for the current demonstration. However, it has some drawbacks; population priors are discussed more in Chapter 3.

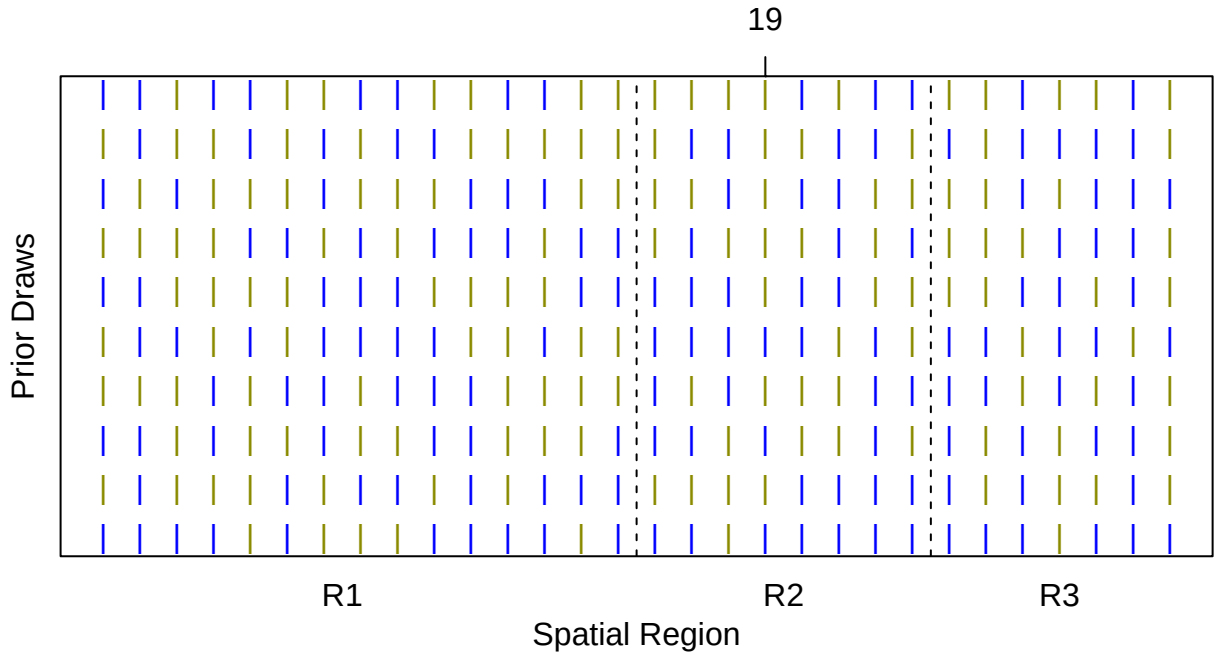


Figure 1.1: Prior draws for a binary (blue or yellow) population; three spatial regions are shown.

Notice also from Figure 1.1 that the population $\mathbf{y}$ divides into three regions with $N_{R1} = 15$, $N_{R2} = 8$, and $N_{R3} = 7$ observations respectively. Let spatial region 1 hold the sub-population $\mathbf{y}_{R1} = \{y_1, \ldots, y_{N_{R1}}\}$, spatial region 2 hold the subpopulation

$\mathbf{y}_{R2} = \{y_{N_{R1}+1}, \ldots, y_{N_{R1}+N_{R2}}\}$, etc. In each sub-population, we have an unknown number of blue members, denoted $K_{R1}$, $K_{R2}$, and $K_{R3}$ respectively.

From these three regions, simple random samples of size $n_{R1} = 9$, $n_{R2} = 0$, and $n_{R3} = 3$ are collected (there is no SRS from region 2). When we consider one SRS at a time and count observed blue observations, we naturally have a hypergeometric model for the data per region. For example, in from region 1, we have $n_{R1} = 9$ that produced $k_{R1} = 6$ blue individuals. The sampling probability mass function for $k_{R1}$ is governed by the hypergeometric so that

$$f(k_{R1}|y_{R1}) = \frac{\binom{K_{R1}}{k_{R1}}\binom{N_{R1}-K_{R1}}{n_{R1}-k_{R1}}}{\binom{N_{R1}}{n_{R1}}} \propto \binom{K_{R1}}{k_{R1}}\binom{N_{R1}-K_{R1}}{n_{R1}-k_{R1}}$$

where $K_{R1}$ is the number of blue individuals in the subpopulation $\mathbf{y}_{R1}$. Similarly for spatial region 3, $R3$, we observe $k_{R3} = 1$ from a SRS of size $n_{R3} = 3$. Thus we have the sampling distribution for $k_{R3}$

$$f(k_{R3}|y_{R3}) \propto \binom{K_{R3}}{k_{R3}}\binom{N_{R3}-K_{R3}}{n_{R3}-k_{R3}}$$

where the subpopulation size is $N_{R3} = 7$ and the number of blue individuals in the subpopulation $\mathbf{y}_{R3}$ is given by $K_{R3}$. Finally, we have information that the individual at spatial location $i = 19$ is blue. Hence we have the final contribution to the likelihood: $I[y_{19} = 1]$ (since we are counting blue individuals as 1).

When we assume the simple random samples are independent, we can combine the population prior with these different sources of information to learn the posterior for the population

$$
\begin{aligned}
f(\mathbf{y}|\mathbf{x}) &\propto f(k_{R1}|y_{R1}) \cdot I[y_{19}=1] \cdot f(k_{R3}|y_{R3}) \cdot f(y) \\
&\propto \binom{K_{R1}}{k_{R1}} \binom{N_{R1}-K_{R1}}{n_{R1}-k_{R1}} \cdot I[y_{19}=1] \cdot \binom{K_{R3}}{k_{R3}} \binom{N_{R3}-K_{R3}}{n_{R3}-k_{R3}} \cdot 2^{-N}
\end{aligned}
$$

Draws from the posterior can be produced using a straightforward Markov chain Monte Carlo scheme. Once population $\mathbf{y}$ is initialized with a random configuration, then a metropolis proposal for each population member $y_i$ is proposed and updated according to the metropolis rule

$$
y_i = \begin{cases} y_i' & \text{with probability } \alpha \\ y_i & \text{with probability } 1-\alpha \end{cases}
$$

where $\alpha = \min\{\pi(y'|\mathrm{x})/\pi(y|\mathrm{x}), 1\}$, and $y'$ and $y$ are identical except for the $i^{\text{th}}$ element. Updates cycle through population members $i = 1, \ldots, N$ individually. After a short sequence of updates for burn-in, the resulting draws $\{\mathbf{y}^1, \ldots, \mathbf{y}^T\}$, where $T$ is a predetermined number of populations to synthesize, can be collected as (dependent) realizations from $f(\mathbf{y}|\mathbf{x})$.

Five draws from the posterior are show in Figure 1.2. Note that these draws incorporate dependencies produced by the SRS results and, trivially, the direct observation of $y_{19} = 1$. The treatment of the full population as the unknown random variable distinguishes this work from individual based descriptions (e.g. Farooq et al. (2013) which produce populations via i.i.d. draws from an estimated joint distribution for an individual. It is also worth noting that even this very simple example provides a means to combine information from spatial aggregates and from the individual.

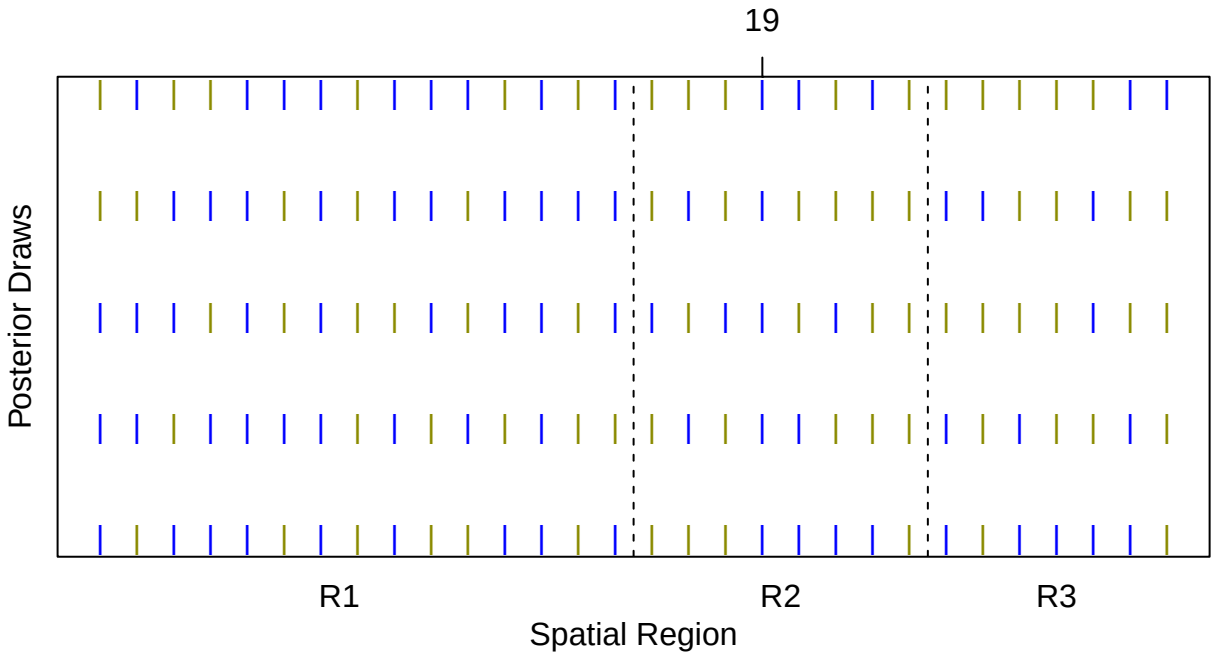


Figure 1.2: Posterior draws for a binary (blue or yellow) population, using both simple random samples and knowledge of $y_{19}$.

## 1.3 Blacksburg Data

Effectively, scaled methods in Chapters 4 and 5 transform theoretical ideas in Chapter 3 to those that are practical and applicable in real-life scenarios. To show this, examples are provided in chapters 3 - 5 using data collected from Blacksburg, Virginia. Thus, in this section, we will outline the data that we have for Blacksburg.

The majority of our data for Blacksburg comes from the US Census Bureau. In addition to performing the census, the US Census Bureau conducts a survey called the American Community Survey (2019b). A number of products are created with this survey data. For our purposes, we will use the Public Use Microdata Sample (PUMS) (2019c) and 5-Year Estimates (2019a). The PUMS provides a small stratified random sample of the population living within areas known as Public Use Microdata Areas (PUMAs). PUMAs contain, at minimum, one-hundred thousand individuals. The PUMA containing Blacksburg contains

approximately 184,000 individuals (as of 2019), and contains the towns of Pulaski and Radford, in addition to Blacksburg; a high-resolution map detailing the PUMA can be found online (US Census Bureau 2010). The 5-Year Estimates provide tabulated data at various levels of granularity, including the census block group level. The level of granularity at which a variable is reported upon depends upon the identifiability of each variable individually. The published tables report the information you would get from a bar plot (estimated membership within each level of a categorical variable), as well as a margin of error. For some pairs of variables, cross-tabulated data is available. As the name implies, these tabulated data reflect estimates using 5 years of survey data (US Census Bureau 2019a).

Additionally, we will be using parcel and tax records from the town of Blacksburg. This information is made publicly available online (Virginia Tech Library Maps & GIS Division 2019). Whenever possible, we use data from 2019; parcel records for 2019 were readily available at the time this project started, however tax records had not been posted yet and had to be obtained my emailing one of the site maintainers.

### 1.3.1 American Community Survey (PUMS)

Though the PUMS (2019c) data contains many variables, our examples primarily focus on two: household income and property value. Due to the extremely skewed nature of incomes and property values, often a transformation is applied; in our case, we use the square root transformation on both variables. The relationship between these two variables at the PUMA level is shown in Figure 1.3. Keep in mind that the PUMA is much larger than the town of Blacksburg proper. However, it is the closest thing we have to a simple random sample from Blacksburg.

From the ACS 5-Year Estimates (2019a) we get marginal information about census tracts

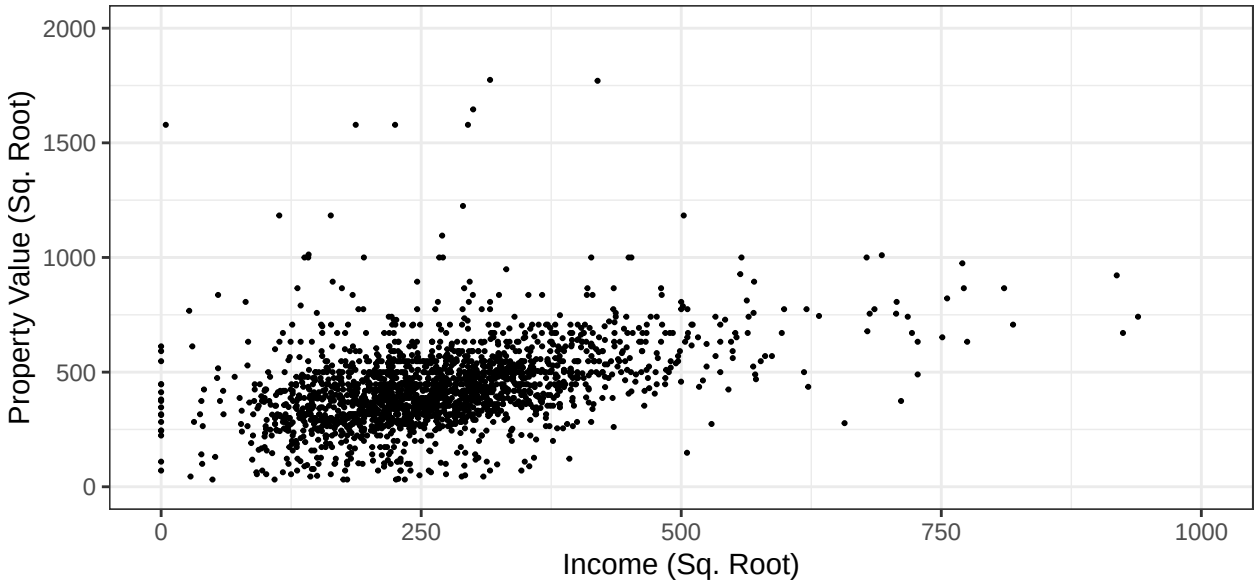


Figure 1.3: Scatterplot showing relationship between the square root of household income (y-axis) and the square root of property value (x-axis).

that exist inside of the town of Blacksburg, as well as the town of Blacksburg. For example, product S1901 gives marginal information on household income, additionally broken down by families and non-families if desired. If we use block groups, ACS will only provide an estimate of median income and not marginal income, for identifiability reasons; this comes from product B19013 (2019a). Figure 1.4 shows an example of an S1901 table for one census tract inside of Blacksburg (in this case, the entire census tract is not within the town of Blacksburg proper).

For property value, the ACS 5-Year Estimates (2019a) also provides marginal information, however the data is limited to owner-occupied housing units. Product DP04 is one source of this information, which can also be broken down to the census tract and block group level. Figure 1.5 shows an example of a DP04 table for the same census tract as above, with other variables we are not concerned with removed.

| | Census Tract 203, Montgomery County, Virginia | | | |
|---|---|---|---|---|
| | Households | | Families | |
| Label | Estimate | Margin of Error | Estimate | Margin of Error |
| Total | 2,653 | ±212 | 1,270 | ±153 |
| Less than $10,000 | 19.6% | ±6.1 | 1.9% | ±2.3 |
| $10,000 to $14,999 | 8.4% | ±5.5 | 6.5% | ±8.7 |
| $15,000 to $24,999 | 9.5% | ±5.2 | 7.1% | ±8.3 |
| $25,000 to $34,999 | 7.6% | ±4.6 | 7.4% | ±6.0 |
| $35,000 to $49,999 | 6.5% | ±3.7 | 3.8% | ±3.5 |
| $50,000 to $74,999 | 10.3% | ±4.9 | 9.1% | ±5.6 |
| $75,000 to $99,999 | 4.9% | ±2.3 | 5.5% | ±3.7 |
| $100,000 to $149,999 | 15.5% | ±4.5 | 26.7% | ±8.0 |
| $150,000 to $199,999 | 9.6% | ±3.9 | 16.5% | ±8.0 |
| $200,000 or more | 8.1% | ±3.6 | 15.6% | ±6.5 |
| Median income (dollars) | 46,378 | ±13,200 | 109,795 | ±11,840 |
| Mean income (dollars) | 75,346 | ±9,211 | 118,608 | ±18,585 |

Figure 1.4: Example table for S1901: Income in the past 12 months (in 2019 inflation-adjusted dollars).

| | Census Tract 203, Montgomery County, Virginia | | | |
|---|---|---|---|---|
| Label | Estimate | Margin of Error | Percent | Percent Margin of Error |
| VALUE | | | | |
| Owner-occupied units | 1,278 | ±153 | 1,278 | (X) |
| Less than $50,000 | 85 | ±104 | 6.7% | ±7.7 |
| $50,000 to $99,999 | 36 | ±39 | 2.8% | ±3.1 |
| $100,000 to $149,999 | 13 | ±21 | 1.0% | ±1.6 |
| $150,000 to $199,999 | 16 | ±26 | 1.3% | ±2.0 |
| $200,000 to $299,999 | 431 | ±120 | 33.7% | ±8.5 |
| $300,000 to $499,999 | 610 | ±141 | 47.7% | ±10.1 |
| $500,000 to $999,999 | 87 | ±56 | 6.8% | ±4.4 |
| $1,000,000 or more | 0 | ±17 | 0.0% | ±2.7 |
| Median (dollars) | 317,900 | ±31,370 | (X) | (X) |

Figure 1.5: Example table for DP04: Selected housing characteristics.

### 1.3.2 Parcel and Tax Records

Records from Montgomery County provide three variable pieces of information:

- Geographical information on parcels
- Zoning and classification status
- Property value

Just like the PUMS data, this data includes records outside of Blacksburg. Figure 1.6 shows the parcels after limiting to only records within Blacksburg.

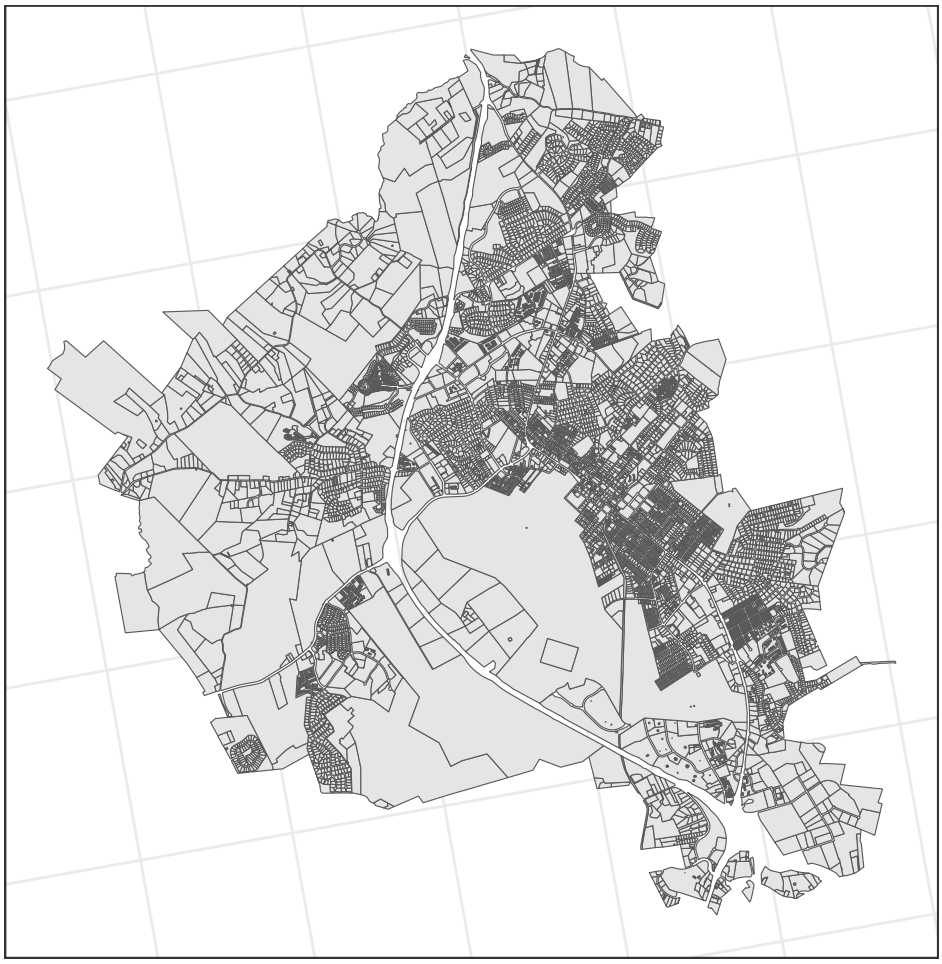

Figure 1.6: All Blacksburg parcels, from Montgomery County's public records.

Since Blacksburg is large enough that we cannot show the parcels well on paper, we will focus (graphically) on a zoomed-in view near the center of town. Additionally, we can overlay the census block groups onto our view of Blacksburg, and show median income obtained from the ACS 5-Year Estimates (2019a). Figure 1.7 shows this information. Henceforth, plots will only show the square area, so that maps are easier to read; also, we are omitting the actual parcels themselves in this figure.

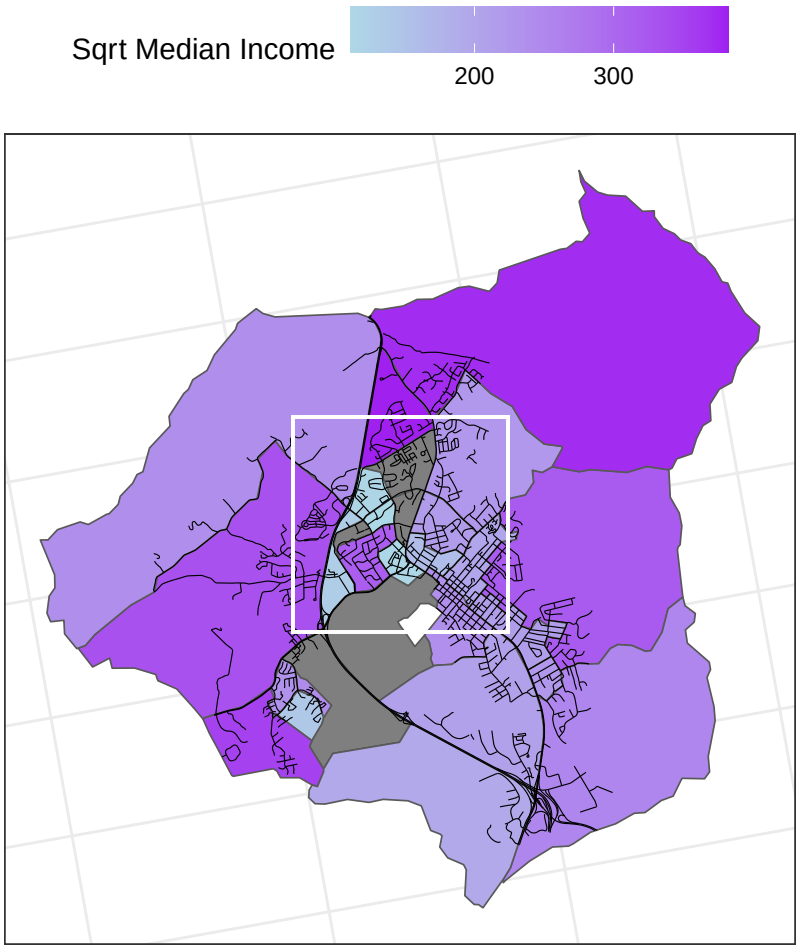


Figure 1.7: Census block groups within Blacksburg, colored by square root of median income.

In addition to geographic information on the parcels, we also get zoning status. This allows is to limit our scope to residential properties via tax zoning, and we can further limit to single-family housing if we so desire via a classification variable within the tax records. Figure 1.8 shows the area within the zoomed-in view that is classified as residential or planned residential. Of note, several areas zoned as planned residential were already occupied residences in 2019, so there is certainly some delay in when properties are rezoned correctly. We will use parcels that are classified as residential or planned residential. Several tax zoning codes are being merged here; most areas classified as "other" are university or commercial properties.

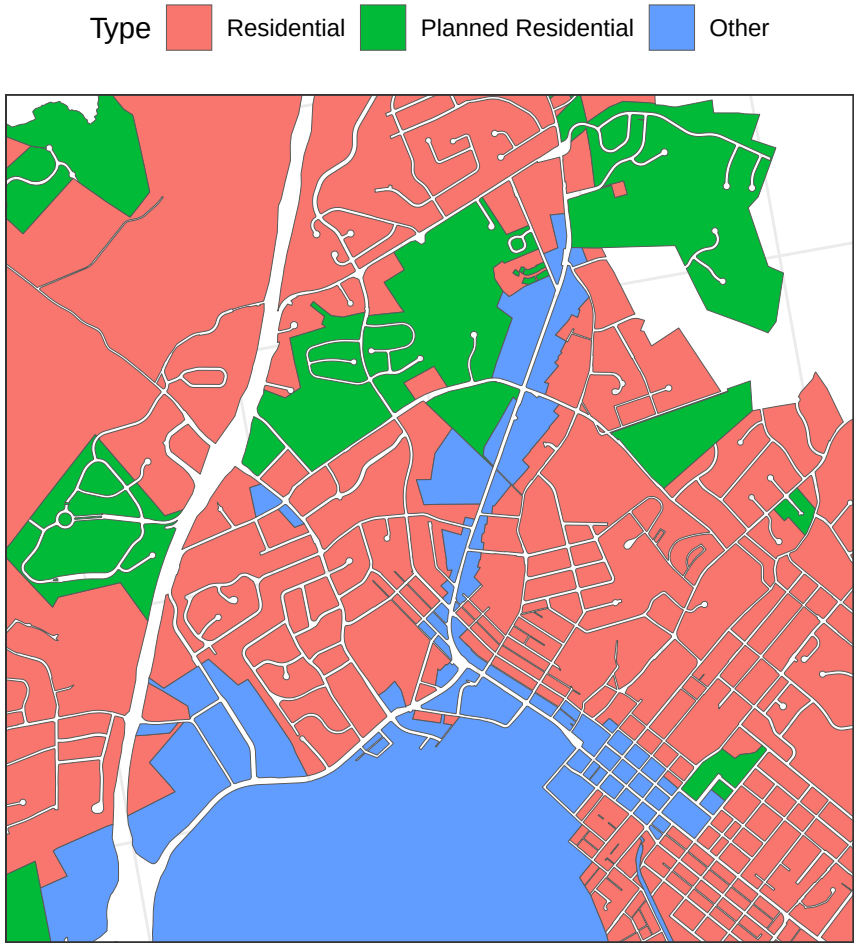


Figure 1.8: Zoning status of properties. Zoning statuses are merged so that the boundaries are not obscuring the plot.

After filtering out all properties that are not zoned as residential or planned residential, we can further look at the classification of properties as single or multi-family residential. Figure 1.9 shows this data. One thing to note here is that some apartments show up as several single-family parcels inside of a larger parcel classified as "other", but some of the larger apartment buildings are entirely classified as multi-family. This is just one complication to the data set we are working with.

The final important piece of information that we get from the parcel and tax records is the property value of every parcel of interest. Figure 1.10 shows this data.

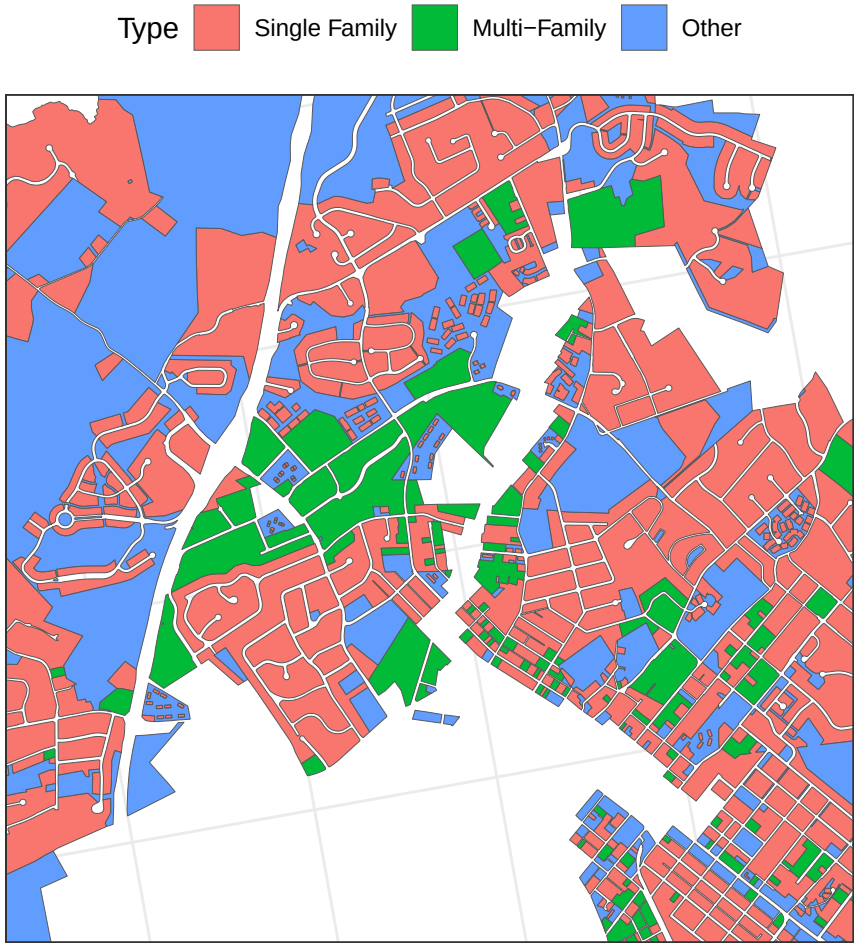


Figure 1.9: Classification of parcels as single or multi-family households, only showing parcels zoned as residential or planned residential.

Throughout the rest of this thesis, we will apply population synthesis methods to the Blacksburg data we have introduced here. As we progress through this document, we will refine our prior specification in Chapter 3, include multiple data sources in Chapter 4, and finally model multivariate populations in 5.

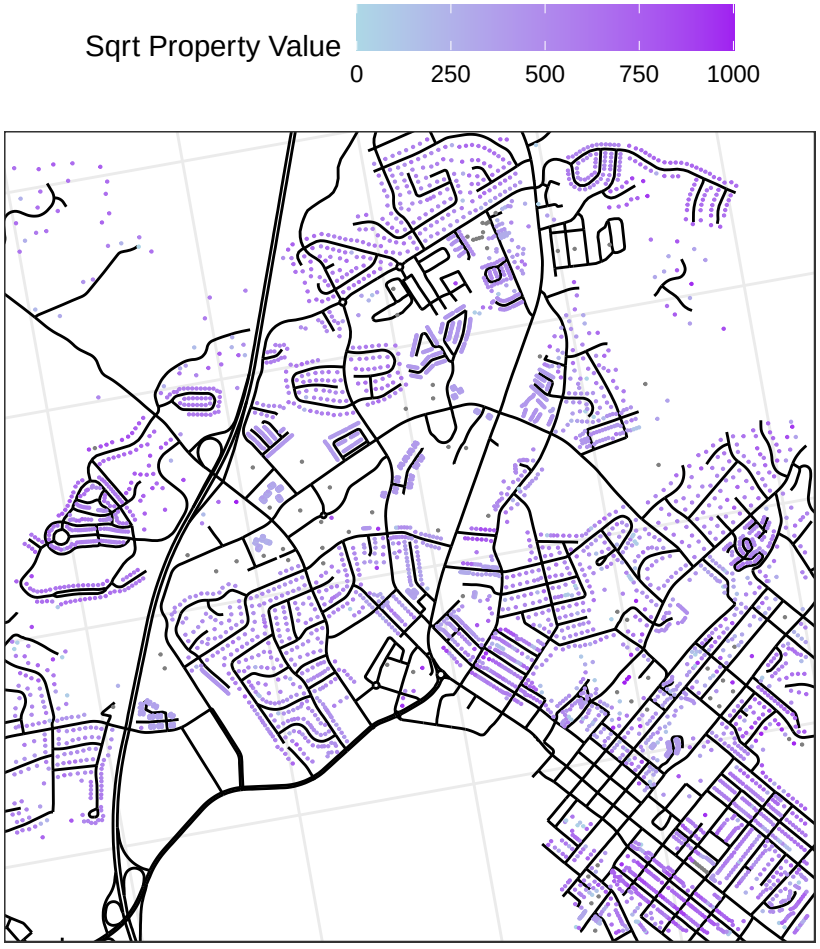


Figure 1.10: Square root of property value, shown for single and multi-family households zoned as residential or planned residential. Values over 1,000,000 are excluded.

# Chapter 2

# Literature Review

As this dissertation is concerned with the construction of synthetic populations, this chapter will review several existing methods that create synthetic populations. Additionally, we will provide a more in depth look at Iterative Proportional Fitting (IPF), by far the most commonly utilized method for synthesizing populations.

## 2.1 Existing Methods

Several reviews of existing methods for creating synthetic populations have been published, including Müller and Axhausen (2010), Müller and Axhausen (2012), Hermes and Poulsen (2012), and Yaméogo et al. (2021). One of the main difficulties when discussing population synthesis methods is that the various disciplines making use of synthetic populations do not necessarily use the same terminology. We will borrow terminology from the transportation modeling world; in particular, we will split methods into three categories: synthetic reconstruction (SR), combinatorial optimization (CO), and statistical learning (SL), per Yaméogo et al. (2021) and others. Figure 2.1 shows the state-of-the-art methods from each category.

### 2.1.1 Synthetic Reconstruction

Methods in the synthetic reconstruction (SR) class work by calculating weights that are assigned to each row or observation within a sample, that represent the number of times a

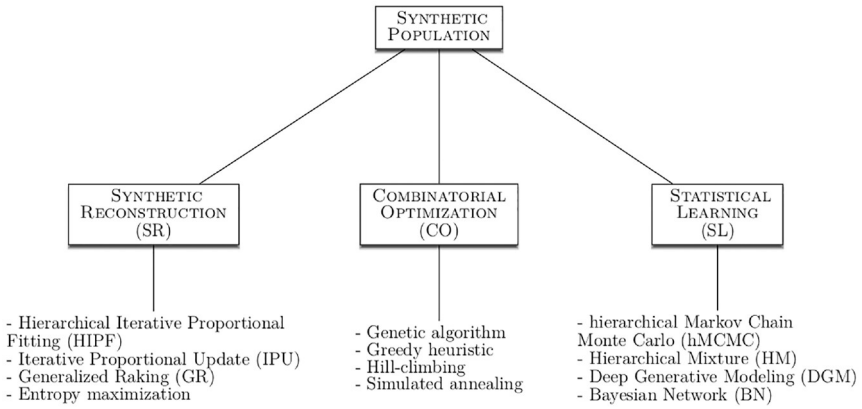


Figure 2.1: Chart of existing methods, from Yaméogo et al. (2021)

specific observation should be replicated within the population. The novel approach within this class of methods, which most of the others are based off of, is Iterative Proportional Fitting (IPF), first applied to population synthesis by Beckman, Baggerly, and McKay (1996). IPF is a very old method, which has been re-invented several times for various reasons (Deming and Stephan 1940; Stephen E. Fienberg 1970). At it's core, IPF is a procedure that finds the closest approximation $\mathbf{Y}$ of an existing matrix, $\mathbf{X}$, by matching the margins of a third matrix, $\mathbf{Z}$. When applied to population synthesis, IPF requires both a sample $x$ as well as a source of marginal information $z$ to create a synthetic population $y$. IPF is a very simple algorithm, and the resulting population has a few desirable qualities. For one, the population is guaranteed to match the margins provided, and the correlation structure of the sample will be preserved (Rich and Mulalic 2012). Additionally, variables that are *not* used to match margins will be reasonably estimated (Beckman, Baggerly, and McKay 1996).

However, there are a few negative qualities as well. Firstly, the original IPF algorithm works only for a single level, e.g., individuals or households, but *not* both. Additionally, IPF (and in fact all SR algorithms) are deterministic, meaning that the resulting population will always be the same if the algorithm is performed multiple times. Lastly, IPF exhibits the "zero-cell-value problem"; as a consequence of IPF duplicating and not creating new sample

records, combinations of levels will not be present within the population if they are not already observed in a sample.

Example 2.1 shows a basic 2-dimensional contingency table illustrating how the IPF algorithm works. In this example, $x$ is the starting sample matrix, $z$ is the target margins, and $y$ is the resulting matrix after the algorithm converges. In 2 dimensions, the algorithm iterates between rows and columns and multiplies the cell counts to match target margins. The algorithm iterates between dimensions until a stopping criteria has been met (e.g., the maximum difference between 2 iterations is less than a small $\epsilon$). Extending this algorithm to $k > 2$ dimensions is very simple, but difficult to show visually; the process repeats but iterates between $k$ dimensions instead of 2. In the general $k > 2$ case, guaranteeing convergence is complicated (Yaméogo et al. 2021), but when $k = 2$, convergence is guaranteed as long as the margins are positive and the contingency table cannot be permuted into a block-diagonal matrix. When using real data, these conditions will typically be satisfied. Problems can occur when a specific variable is very unbalanced (i.e., a specific value is very unlikely) in the population. If certain conditions are satisfied, IPF will converge to the MLE, and that MLE will be unique. When the conditions are not met, it will converge to the *extended MLE*, but this may occur arbitrarily slowly (Haberman 1974).

**Example 2.1.**

| | 1 | 2 | *Total* | *Target* |
|---|---|---|---|---|
| 1 | 50 | 100 | 150 | 250 |
| 2 | 75 | 25 | 100 | 200 |
| *Total* | 125 | 125 | 250 | |
| *Target* | 200 | 250 | | 450 |

→

| | 1 | 2 | *Total* |
|---|---|---|---|
| 1 | 83.33 | 166.67 | 250 |
| 2 | 150 | 50 | 200 |
| *Total* | 233.33 | 216.67 | |

→

| | 1 | 2 | *Total* |
|---|---|---|---|
| 1 | 71.43 | 192.31 | 263.74 |
| 2 | 128.57 | 57.69 | 186.26 |
| *Total* | 200 | 250 | |

→

| | 1 | 2 | *Total* |
|---|---|---|---|
| 1 | 67.71 | 182.29 | 250 |
| 2 | 138.05 | 61.95 | 200 |
| *Total* | 205.76 | 244.24 | |

→

| | 1 | 2 | *Total* |
|---|---|---|---|
| 1 | 65.81 | 186.59 | 252.40 |
| 2 | 134.19 | 63.41 | 297.60 |
| *Total* | 200 | 250 | |

→ ··· →

| | 1 | 2 | *Total* |
|---|---|---|---|
| 1 | 64.66 | 185.34 | 250 |
| 2 | 135.34 | 64.66 | 200 |
| *Total* | 200 | 250 | |

Since the introduction of IPF as a method for population synthesis, a number of modifications have been developed to overcome the single level nature of IPF; however, most fail to guarantee consistency between both levels. There are a few methods based on IPF that are worth mentioning: Iterative Proportional Updating (IPU), Entropy Maximization, and Hierarchical Iterative Proportional Fitting (HIPF). Various authors had proposed methods that dealt with multiple levels sequentially, i.e., performing IPF at the individual level, and then again at the household level. Within the same month, two groups presented the first IPF-based algorithms to guarantee consistency between multiple levels. IPU was presented by Ye et al. (2009), who developed a fractional expansion factor at the household level such

that both household-level and individual-level controls (margins) would be guaranteed satisfied. An Entropy Maximization solution to the same problem was presented by Bar-Gera et al. (2009) at the same time. Two years later, Müller and Axhausen (2011) presented HIPF, and proved that HIPF introduces the least amount of new information at the household level. One important property to note is that all three afore-mentioned algorithms yield the exact same results if applied to a single level problem (i.e., data is limited to the individual level) (Müller and Axhausen 2011).

Finally, Generalized Raking (GR) encompasses all methods that minimize a distance between the sample weights and population weights in the same manner as IPF. Deville, Särndal, and Sautory (1993) consider the traditional raking method (Deming and Stephan 1940) a special case using a particular distance function. Expanding the traditional algorithm to allow a choice of distance functions means that GR can yield the same results as the algorithms mentioned above. For example, using cross-entropy distance should yield the same results as Entropy Maximization. However, there are several other distance functions one could use that do not equate to other algorithms, so GR can be an alternative to the afore-mentioned algorithms (hIPF, IPU) if a specific distance function is desired.

### 2.1.2 Combinatorial Optimization

In Example 2.1, note that the solution resulted in fractional individuals (e.g., 64.66 individuals); this will be the case with all SR algorithms. If you desire an algorithm such that individuals and households are always duplicated a whole number of times, CO algorithms are one option. This represents one of two major differences between SR and CO algorithms. The other difference is that CO algorithms are stochastic (Yaméogo et al. 2021). Not counting these differences, one advantage over SR algorithms is that they require less data to operate (Templ et al. 2017). However, one major disadvantage is that they are computa-

tionally expensive, being unable to guarantee finding the optimal solution as population size and dimensionality grow (D.-H. Lee and Fu 2011).

Templ et al. (2017) outlines how CO algorithms work; first, they separate the synthetic population into small (usually geographic) areas for which tabulated data is available (e.g., ZCTAs, census tracts, census block groups, etc.). Then, using cross-tabulated data for households within these areas, different combinations of households are chosen for each area that best fit the constraints. Then, at the individual level, the algorithm is initialized with a random filling of each household with individuals from microdata. Sequentially, each individual is then swapped with another individual from the microdata, and the goodness of fit is recalculated. If the fit improved, the swap is kept; if the fit does not improve, the swap is undone and another swap is considered. This process repeats until some stopping criteria has been met. Harland et al. (2012) suggests using goodness-of-fit defined by the relative sum of squared $Z$-scores, but there are other metrics one could use.

### 2.1.3 Statistical Learning

Statistical learning methods are simulation-based approaches that attempt to sample directly from a full joint distribution (Yaméogo et al. 2021); since this full joint distribution is not available, and the data needed to estimate it fully is never available, some pieces must be estimated. The general framework is to use given data that can be used to approximate marginal and/or conditional marginal distributions and hopefully a random sample to create a synthetic population with an empirical distribution as close to the true joint distribution as possible.

The various SL algorithms go about sampling from this approximated joint distribution in different ways. Perhaps the most simple method, proposed by Farooq et al. (2013),

uses Gibbs sampling within an MCMC framework from estimated conditional distributions. Other approaches include Saadi et al. (2016), who used a Hidden Markov Model or Casati et al. (2015), who used an hierarchical MCMC framework to build off of Farooq et al. (2013). More complicated approaches to sampling from this joint density include Sun and Erath (2015) and Zhang et al. (2019), who used Bayesian Networks, and Borysov, Rich, and Pereira (2019), who used a Deep Generative Model, essentially a neural network with many layers that is often used to approximate a complicated joint distribution.

One drawback of SL methods when compared to SR and CO methods is that they are not guaranteed to match marginal distributions if provided. One could easily fix this by feeding synthetic populations created by an SL method into an SR method such as IPF. There are a number of potential advantages to SL methods as well; in general, they work well with smaller samples than SR and CO methods require, and they do not exhibit the "zero-cell value problem" that plagues SR methods (Yaméogo et al. 2021).

## 2.2 Iterative Proportional Fitting

Within this section, we will provide an example application of Iterative Proportional Fitting applied to Blacksburg data. For this example, we will synthesize two variables: household income and property value. Two data sources for these variables are:

- American Community Survey's 5-Year Estimates (2019a)
- American Community Survey's PUMS (2019c)

The first data source provides marginal information for Blacksburg. Specifically, we will look at the *S506/S507: financial characteristics for housing units with/without a mortgage* to extract the housing prices. Additionally we will get marginal incomes from *S1901: in-*

*come in the past 12 months* (using families). For the sample, we will use the PUMS data; unfortunately our chosen town of Blacksburg, VA is much smaller than a single PUMA, so our sample will actually be taken from a larger area.

First, we have to extract the categories from the PUMS; since they are not categorized, we bin the values into the categories defined within the 5-year estimates. We also need to calculate the target margins; there is not perfect agreement between the two variables (the total is different), so we need to slightly modify one or the other. From there, we can look at the contingency table of the resulting categories from the PUMS.

| | | Income | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | <10k | 10k-15k | 15k-25k | 25k-35k | 35k-50k | 50k-75k | 75k-100k | 100k-150k | 150k-200k | >200k | Total | Target |
| House Price | <50k | 16 | 6 | 29 | 22 | 19 | 27 | 13 | 9 | 1 | 1 | 143 | 153 |
| | 50k-100k | 14 | 25 | 33 | 51 | 52 | 60 | 35 | 15 | 5 | 2 | 292 | 52 |
| | 100k-300k | 33 | 20 | 86 | 87 | 174 | 297 | 215 | 213 | 48 | 35 | 1208 | 2017 |
| | 300k-500k | 5 | 3 | 12 | 10 | 21 | 53 | 49 | 71 | 52 | 46 | 322 | 1705 |
| | 500k-750k | 3 | 0 | 4 | 3 | 5 | 10 | 9 | 14 | 10 | 29 | 87 | 496 |
| | 750k-1000k | 0 | 0 | 1 | 2 | 0 | 1 | 2 | 1 | 2 | 7 | 16 | 66 |
| | >1000k | 1 | 1 | 3 | 1 | 2 | 4 | 3 | 1 | 2 | 7 | 25 | 0 |
| | Total | 72 | 55 | 168 | 176 | 273 | 452 | 326 | 324 | 120 | 127 | 2093 | |
| | Target | 261 | 103 | 193 | 198 | 180 | 566 | 629 | 885 | 674 | 800 | | 4489 |

Figure 2.2: Table showing the discretized two-way table of household income and housing proce from ACS PUMS, with the desired margins taken from ACS 5-Year Estimates.

There are numerous implementations of IPF available; here we use the R package *mipfp*, which provides an easy-to-use implementation of IPF (Barthelemy and Suesse 2018). From Figure 2.3, you can see that the algorithm worked despite some of the target margins being less than the observed margins, even for a target margin of zero.

| | | Income | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | <10k | 10k-15k | 15k-25k | 25k-35k | 35k-50k | 50k-75k | 75k-100k | 100k-150k | 150k-200k | >200k | Total |
| House Price | <50k | 40.84 | 11.35 | 23.54 | 19.22 | 8.35 | 20.03 | 13.74 | 11.62 | 2.09 | 2.26 | 153 |
| | 50k-100k | 6.34 | 8.39 | 4.76 | 7.91 | 4.06 | 7.90 | 6.57 | 3.44 | 1.85 | 0.79 | 52 |
| | 100k-300k | 136.46 | 61.27 | 113.10 | 123.12 | 123.84 | 356.86 | 368.09 | 445.57 | 162.47 | 126.21 | 2017 |
| | 300k-500k | 49.46 | 21.99 | 37.75 | 33.85 | 35.75 | 152.34 | 200.69 | 355.30 | 421.06 | 396.80 | 1705 |
| | 500k-750k | 27.89 | 0 | 11.83 | 9.55 | 8.00 | 27.02 | 34.64 | 65.85 | 76.10 | 235.12 | 496 |
| | 750k-1000k | 0 | 0 | 2.02 | 4.36 | 0 | 1.85 | 5.27 | 3.22 | 10.42 | 38.86 | 66 |
| | >1000k | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Total | 261 | 103 | 193 | 198 | 180 | 566 | 629 | 885 | 674 | 800 | 4489 |

Figure 2.3: Table showing the results of IPF performed on the discretized table from above. Note that all target margins from the above table have been satisfied.

Throughout this thesis, we can look back to the results of IPF whenever we wish to compare our methodology against an existing method. This example above represents only the minimum amount of data one could use. One of the main goals of this thesis is to develop methodology that allows us to incorporate multiple data sources, including data that could not be included in IPF without serious modifications.

# Chapter 3

# A Bayesian Formulation for Population Synthesis

In Chapter 1, we highlighted the Bayesian foundations for modeling populations. From these models, populations $\mathbf{y}$ are simulated. Ideally, these simulated populations meet the following criteria:

1. All data samples $\mathbf{x}$ are contained within each synthetic population $\mathbf{y}$.
2. Inference on population parameters from sets of synthetic populations $\{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_m\}$ should be comparable to inference made with traditional Bayesian methods.

The first property is relatively straight-forward, but can be violated easily. For instance, in the example from section 1.2, some simulated populations contain fewer yellow members than the observed sample. This happened because focus was only on the parameter of interest $K$, the number of blue members of the population, and not on $K$ and $N - K$ jointly (the number of blue and yellow members of the population). That is posterior draw had as many or more blue members than the sample (in each designated region), but the same was not true for yellow.

Another desired property of methods for population synthesis is that, regardless of *how* we learn about population parameters $\theta$, inference about $\theta$ should remain comparable, if not

the same. That is, mathematically, we know

$$f(\theta|\mathbf{x}) = \int f(\theta, \mathbf{y}|\mathbf{x}) d\mathbf{y}.$$

However, in practice, model assumptions or computational approximations might disrupt the equality. This means that, in practice, what is learned about $\theta$ given synthetic populations $\mathbf{y}$ might deviate from what is learned of $\theta$ directly from $\mathbf{x}$.

Starting with specifications for population priors, $f(\mathbf{y})$, decisions made for modeling populations consider criteria 1) and 2). This section highlights options for population priors in a simplified context, where all $N$ elements of a population $\mathbf{y}$ are univariate and the data $\mathbf{x}$ is a simple random sample (SRS) of size $n$ from the population:

Population $\mathbf{y} = \{y_1, y_2, \dots, y_N\}$, where $y_i$ is $p \times 1$, $p = 1$

Data $\mathbf{x} = \{x_1, x_2, \dots, x_n\}$, where $\mathbf{x}$ is sampled from $\mathbf{y}$

## 3.1 Equal Mass Priors

When little is known about a population, it makes sense to consider a uniform prior on all possible populations. However, even when intended to be non-informative, uniform priors on populations can be quite informative, particularly about population features; e.g., population mean and variance. This section demonstrates the interplay between non-informativeness and informativeness of priors for populations with binary members.

### 3.1.1 Example: Binary population members

With finite populations whose members are defined by one, independent, discrete attribute, uniform priors over all possible populations are easy to specify; the space of all possible populations can be enumerated easily and assigned equal probability. For example, consider binary members of a population with size $N$. There are $2^N$ populations, and any realization of these populations will occur with probability, $1/2^N$,

$$\pi(y) = 2^{-N}.$$

However, this obvious, uniform prior over populations is quite informative of population summaries, such as the total number of ones $K = \sum_{i=1}^{N} y_i$ in the population. The summary $K$ may take on any value from 0 to $N$ and the prior on $K$, $\pi(K)$ that is induced from specifying a uniform prior on $y$ will not place equal probability on $K = \{0, 1, 2, ..., N\}$. In fact, $\pi(K)$ will place more mass near $K = N/2$ than in the extremes, e.g., near 0 or $N$.

To see this, let $N = 4$. Each possible population $y$ lives in the set $\{0, 1\}^4$, and there are $2^4$ possible populations. These populations are shown in Figure 3.1. Notice, that the probability of selecting a population (in this example) for which $K = 2$ is six times higher than the probability of selecting a population for which $K = 0$.

Similarly, let $N = 10$. Even though 10 is larger than 4 (the previous example), we can still enumerate every possible population. There are $2^{10} = 1024$ populations. A histogram of $K$ from each population is shown in Figure 3.2. Now notice that $P[K = 5]$ is 252 times higher than $P[K = 0]$ or $P[K = 10]$.

Choosing a prior that is highly informative of $K$ works theoretically and criteria 1 and 2 (from Section 3) can be met with this prior. However, some analysts might prefer to model

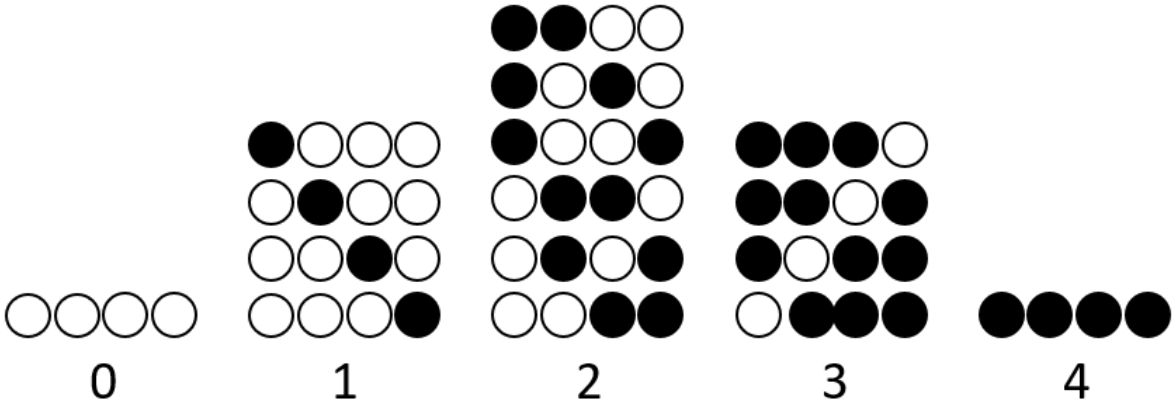


Figure 3.1: All possible populations of size $N = 4$, sorted by $K$, the number of 1s within the population. Individuals with $y_i = 1$ are given by the black circles; individuals with $y_i = 0$ are given by hollow circles.

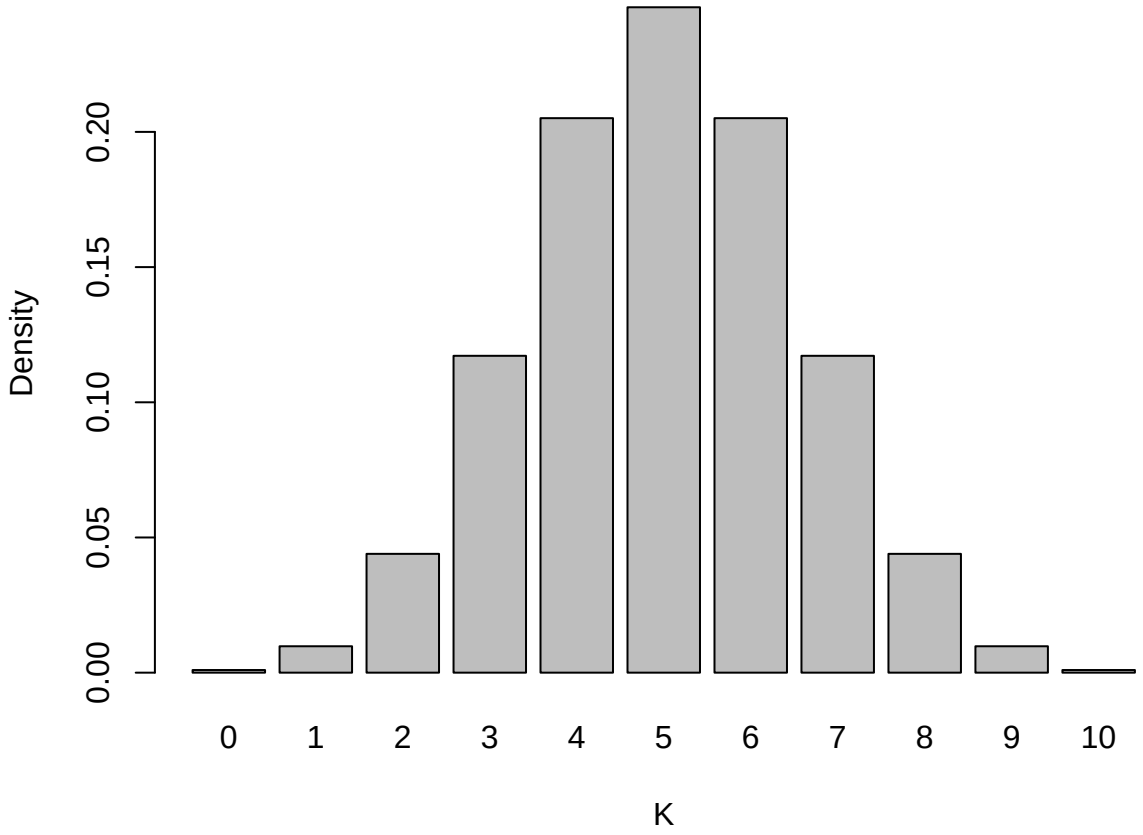


Figure 3.2: Bar plot of K across all possible binary populations of size 10

their expert judgment (or lack thereof) with a less informative prior on $K$. Such a prior requires modeling $\mathbf{y}$ and $K$ hierarchically. We explore this prior in Section 3.2.

### 3.1.2 Example using Blacksburg Data

Since our primary variable of interest in the Blacksburg data is income, one of the simplest processes we could follow to synthesize this variable is to consider the discretized form of the variable and synthesize populations using a flat prior over every possible population. For this purpose, we discretize the variable of income (from the ACS PUMA data) using the ranges from the ACS 5-Year Estimates, seen in Figure 1.4. After doing so, we can apply an MCMC algorithm, where we iterate through changing one member of the population at a time, and accept or reject based on the posterior (which in this case is a simple multivariate hypergeometric likelihood).

Figure 3.3 shows the results of this process. Notice that the posterior shows shrinkage towards equal representation in each discrete income category. This happens because

$$\underset{\mathbf{K}}{\operatorname{argmax}}\binom{N}{\mathbf{K}_1, ..., \mathbf{K}_c} = \left(\frac{N}{c}, ..., \frac{N}{c}\right).$$

In Figure 3.2, we saw that in the binary case, equally weighting all populations resulted in favoring $K = N/2$. When considering a variable that has more than two categories, this translates to favoring the $c$-vector $\mathbf{K} = N/c$, where $c$ is the number of categories.

## 3.2 Hierarchical Priors

In Section 3.1, we saw how using equal mass priors led to poor results. To produce an uninformative prior for $K$ and populations $\mathbf{y}$, we need to specify a population prior that gives

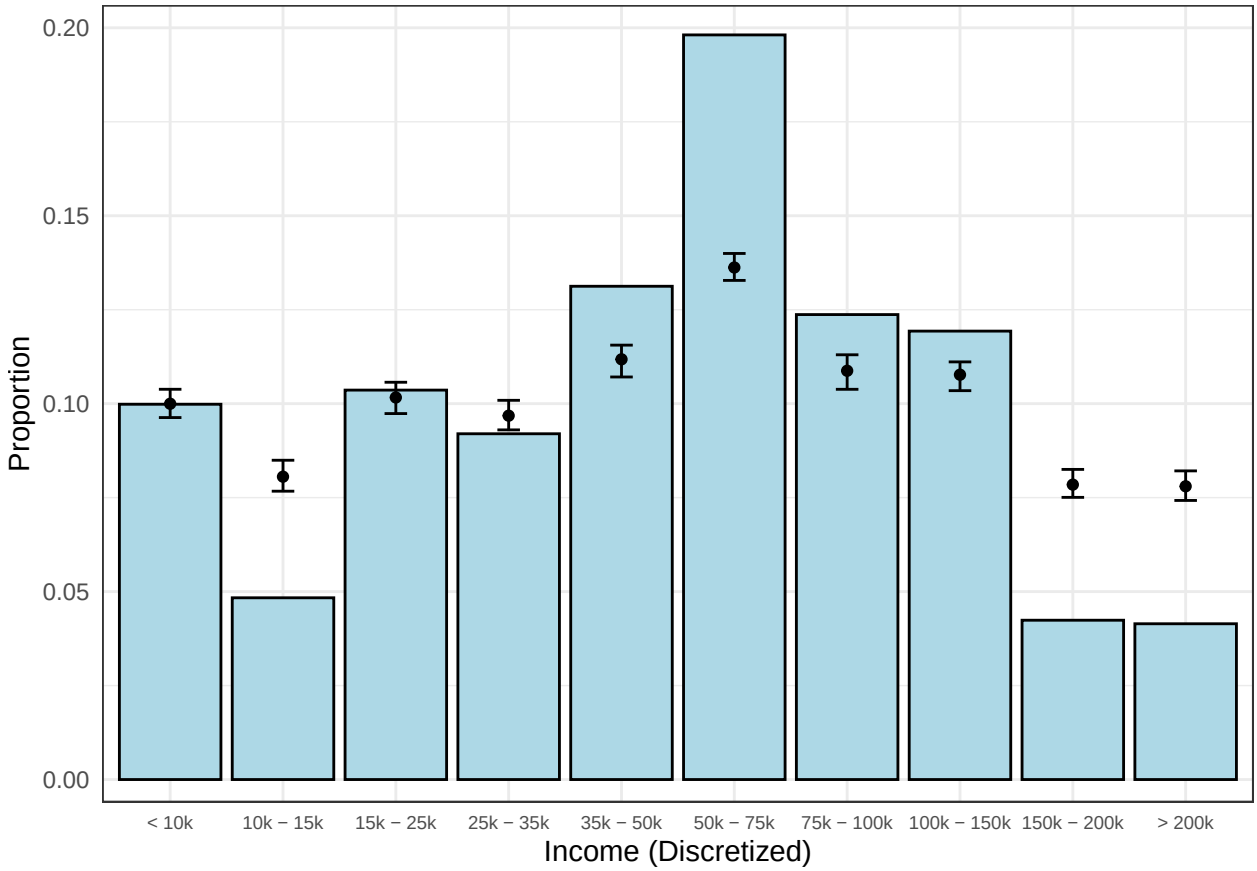


Figure 3.3: Income sample data from ACS PUMS (blue bars) with distribution of Incomes generated via synthetic populations with an equal mass prior; $5^{\text{th}}$ and $95^{\text{th}}$ quantiles represented via error bars, with mean represented with a dot.

different probabilities for each population $\mathbf{y}$ based on the value of $K$. To do so, we could apply a "correction factor" to uniform prior population probabilities, or consider modeling population $\mathbf{y}$ and $K$ hierarchically.

A correction factor for the binary example scales the prior $\pi(\mathbf{y}) = 2^{-N}$ such that $\pi(\mathbf{y}) \propto 2^{-N} \times \binom{N}{K}^{-1}$. A similar scaling with proper priors results from specifying a uniform prior on $K$, and then a prior for $\mathbf{y}|K$ that is uniform over all populations for which $\sum_i^N y_i = K$.

$$\begin{aligned}\pi(\mathbf{y}, K) &= \pi(K) \times \pi(\mathbf{y}|K) \\ &= \frac{1}{N+1} I_{\{K \in \{0,\ldots,N\}\}} \times \binom{N}{K}^{-1} I\left[\sum_{i=1}^{N} y_i = K\right]. \end{aligned} \tag{3.1}$$

This specific hierarchical prior applies when considering a independent discrete attribute $y_i$ for each population member $i$. We generalize the prior for population $\mathbf{y}$ with parameter $\theta$, a prior $f(\theta)$, and a population produced by $N$ i.i.d. draws from $f(\cdot|\theta)$:

$$\pi(\mathbf{y}, \theta) = \pi(\theta) \times \prod_{i=1}^{N} f(y_i|\theta). \tag{3.2}$$

For this hierarchical population prior, if we observe an i.i.d. sample $\mathbf{x}$ of size $n$ from the population $\mathbf{y}$, the sample is determined by the $n$-vector of population indices $\mathbf{S} : \mathbf{x} = \{y_{\mathbf{S}_1}, y_{\mathbf{S}_2}, \ldots, y_{\mathbf{S}_n}\}$. Note that under i.i.d. (and simple random) sampling, all sets of $n$ indices $\mathbf{S}$ are equally likely. The remaining population members are indexed by the complementary set $\mathbf{S}^c : \mathbf{y}_c = \{y_{\mathbf{S}_1^c}, y_{\mathbf{S}_2^c}, \ldots, y_{\mathbf{S}_{N-n}^c}\}$ and $\mathbf{y} = \mathbf{x} \cup \mathbf{y}_c$. The posterior for $\theta$ is then wholly determined by the observed sample $\mathbf{x}$ because

$$\begin{aligned}\pi(\mathbf{y}, \theta|\mathbf{x}) &\propto \pi(\theta) \times \prod_{i=1}^{N} f(y_i|\theta) f(\mathbf{x}|\mathbf{y}) \\ &\propto \pi(\theta) \times \prod_{i \in S} f(y_i|\theta) \cdot \prod_{j \in S^c} f(y_j|\theta);\end{aligned}$$

the remaining population vector $\mathbf{y}_c$ can be integrated out giving

$$\pi(\theta|\mathbf{x}) \quad \propto \quad \pi(\theta) \times \prod_{i \in \mathbf{S}} f(y_i|\theta).$$

This is equivalent to standard Bayesian estimation for the parameter $\theta$ given the i.i.d. sample of $\mathbf{x}$ from $\mathbf{y}$. The posterior distribution for the remaining population $y_c$ is produced by sampling from the posterior predictive distribution

$$f(\mathbf{y}_c|\mathbf{x}) = \int \pi(\theta|\mathbf{x}) \times \prod_{j \in S^c} f(y_j|\theta) d\theta.$$

While using a flat prior for $f(\theta)$ is certainly possible, it may be more desirable to use standard reference priors instead. In the next two sections, we will briefly explore reference priors when population members are categorical.

Thus far, in this chapter we have focused completely on independent sampling schemes, meaning they were sampled with replacement. In the real world, samples are often taken without replacement. Fortunately, not much changes; looking at Equation (3.2), we would instead have

$$\pi(\mathbf{y}, \theta) = \pi(\theta) \times f(\mathbf{y}|\theta),$$

where we are unable to decompose $f(\mathbf{y}|\theta)$ further. This would occur if we know our sample was taken without replacement.

If a population is sufficiently large, we may choose to *pretend* that the sample was taken with replacement despite knowing it was not. This is simply because likelihoods for sampling with replacement (Binomial, Multinomial) are computationally easier to work with. For very

large sample and population sizes, the differences between likelihoods for sampling with and without replacement can be ignored. The table below summarizes which likelihood we will use, depending on the number of categories and whether we think our sampling scheme included replacement.

| | Independence/Replacement? | |
|---|---|---|
| **No. Categories** | **Yes** | **No** |
| 2 | Binomial | Hypergeometric |
| > 2 | Multinomial | MV Hypergeometric |

Figure 3.4: Table showing the sampling distribution of **y** under different scenarios.

The bulk of the remainder of this section is spent detailing each of these four choices, including common prior choices and analyzing any differences on inference when comparing population synthesis to traditional Bayesian parametric inference.

### 3.2.1 Independent Sampling Schemes

In this section, we explore how to create synthetic populations in the i.i.d. discrete case, where we think our sample was taken with replacement. For the binary and categorical cases, we will discuss the likelihood, conjugate and reference priors, and provide pseudocode for creating synthetic populations. One thing to note is that in this case, we will not impose the constraint that $\mathbf{x} \in \mathbf{y}$. For example if $\mathbf{x}$ is binary and $\sum_i x_i = s$, then we do not need to impose $\sum_j y_j \geq s$, because some population members could have been sampled more than once.

#### 3.2.1.1 Binary Populations

Things are very simple if our population is defined by a single binary variable and our sample was taken with replacement. This leads us to use a Binom$(n, \rho)$ distribution for $f(\mathbf{X} = \mathbf{x}|\rho)$. Here we will use $\rho$ to represent to the population's proportion of successes; we could just as well use $K$, the number of successes in the population. Thus,

$$\begin{aligned} f(\mathbf{Y} = \mathbf{y}, \rho|\mathbf{X} = \mathbf{x}) &= \frac{f(\mathbf{X} = \mathbf{x}|\mathbf{Y} = \mathbf{y}, \rho) f(\mathbf{Y} = \mathbf{y}, p)}{f(\mathbf{X} = \mathbf{x})} \\ &\propto f(\mathbf{X} = \mathbf{x}|\rho) f(\mathbf{Y} = \mathbf{y}|\rho) \pi(\rho). \end{aligned}$$

We already know the first piece of this puzzle, $f(\mathbf{X} = \mathbf{x}|\rho)$, will be a Binom$(x|n, \rho)$ distribution. The second piece, $f(\mathbf{Y} = \mathbf{y}|\rho)$, we have not spoken about. Here we will use a flat distribution, $f(\mathbf{y}|\rho) = \binom{N}{N\rho}^{-1}$, because we have no reason to favor certain configurations of $\mathbf{y}$ over others. By this, we mean that $\mathbf{y} = \{1, 0, 1, 0\}$ and $\mathbf{y} = \{1, 1, 0, 0\}$ should be equally likely, when we condition on $N\rho = 2$. There are certainly situations where this would not be the case, but we do not explore them here. Now we must discuss the final piece, $\pi(\rho)$.

If we were not performing population synthesis, the most popular choice would be to let $\pi(\rho) = \text{Beta}(\rho|a, b)$, since a Beta distribution is conjugate for the Binomial likelihood. This would form the canonical posterior Beta$(k + a, n - k + b)$, where $k = \sum_i x_i$. Popular choices for $a, b$ would be $a = b = 1$ (the continuous uniform distribution), $a = b = 0.5$ (the Jeffreys and reference prior). Any choice where $a = b$ and $a, b \leq 1$ is often termed "uninformative", though no such prior technically exists.

Since we want inference on our synthetic populations to be analogous to traditional Bayesian

inference, we should choose to use the same prior $\pi(\rho)$ that we would use if we were not performing population synthesis. While we could sample directly using full conditionals, we will instead resort to MCMC sampling, for which we provide the following pseudocode. The reason for this choice is that sampling from full conditionals will become impossible as the situation becomes more complex by adding multiple data sources (Chapter 4), or considering multivariate populations (Chapter 5).

**INPUT**: $\mathbf{x}$, $N$, $B$, $a$, $b$

**OUTPUT**: $B$ realizations of $\mathbf{y}$

Initialize $\mathbf{y}$ (any valid population will work)

**FOR** $i$ **IN** $1, \ldots, B$

  **FOR** $j$ **IN** $1, \ldots, N$

    Propose $\mathbf{y}^*$ and $\rho^*$, where $y_j$ is changed

    Calculate $\alpha = \min\Big(1, \frac{f(\mathbf{y}^*, \rho^* | \mathbf{x}, a, b)}{f(\mathbf{y}, \rho | \mathbf{x}, a, b)}\Big)$

    Set $\mathbf{y} = \mathbf{y}^*$ with probability $\alpha$

  **SAVE** $\mathbf{y}$

**RETURN** $B$ realizations of $\mathbf{y}$

Here,

$$f(\mathbf{y}, \rho | \mathbf{x}, a, b) \propto \text{Binom}\Big(\sum_{i=1}^{n} x_i \Big| n, \rho\Big) \times \frac{1}{\binom{N}{N\rho}} \times \text{Beta}(\rho | a, b),$$

where $\rho = \frac{1}{N}\sum_{j=1}^{N} y_j$. Code for this algorithm can be found in Appendix 7.1.1.

To illustrate, we synthesize populations $\mathbf{y}$ of size $N = 100$ from a sample $\mathbf{x}$ of size $n = 20$ where $\sum_i^n x_i = 6$, using the hierarchical prior described above with $a = b = 0.5$. We calculate the distribution of $\rho = \frac{1}{N}\sum_{i=1}^{N} y_i$ for 50000 synthetic populations, and graphically

show the empirical distribution of $\rho$ across these populations. Figure 3.5 shows these results, and compares the empirical distributions against the canonical posterior for $\rho$ that would result from using a Beta$(0.5, 0.5)$ prior in a traditional Bayesian analysis. As discussed above, in that case $\pi(\rho|\mathbf{x}, a, b)$ would be Beta$(6.5, 14.5)$.

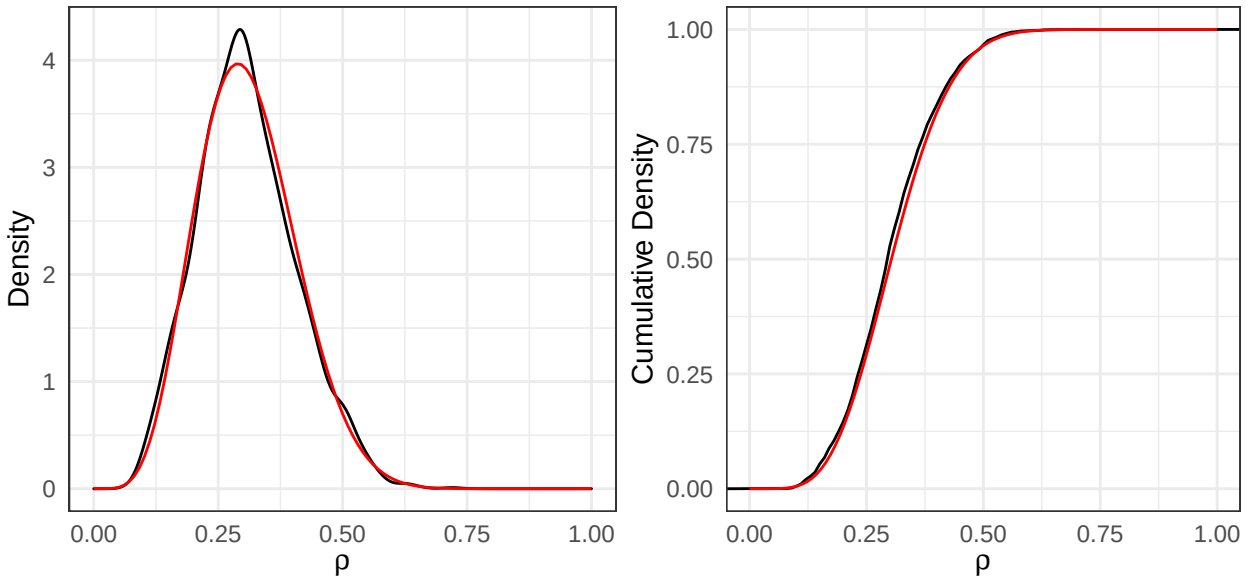


Figure 3.5: Empirical pdf (left) and cdf (right) for K with theoretical posterior from traditional Bayesian analysis using Beta(0.5, 0.5) prior shown in red.

We see that the empirical distribution of $\rho$ from the synthetic populations closely matches the canonical posterior. Theoretically, these distributions should match *exactly*, though with a finite number of synthetic populations we expect some deviation.

#### 3.2.1.2 Categorical Populations

If our variable of interest is not binary but instead takes on $c \geq 3$ categories, we can no longer use the Binomial distribution. Instead, we will use the Multinomial distribution for $f(\mathbf{X} = \mathbf{x}|\vec{\rho})$, where $\vec{\rho}$ is a $c-$vector instead of a scalar.

Some other pieces must change as well. For one, $f(\mathbf{Y} = \mathbf{y}|\vec{\rho})$ must slightly change; we will use the multinomial choose function $f(\mathbf{y}|\vec{\rho}) = \binom{N}{N\rho_1,\ldots,N\rho_c}^{-1}$, instead of the binomial version

(we could of course use other distributions for $f(\mathbf{y}|\vec{\rho})$). Likewise, our prior for $\vec{\rho}$ cannot be Beta$(a, b)$. The multivariate extension of the Beta distribution, the Dirichlet, is the natural choice. Thus, we will use a Dirichlet$(\vec{\alpha})$ for our prior. Values of $\vec{\alpha}$ that are common mirror values of $a, b$ from the Beta$(a, b)$ prior above. For this section, we will use $\vec{\alpha} = \{0.5\}^c$; this constant $c$-vector functions as both the Jeffreys and reference prior.

If we were not performing population synthesis, this choice of prior would result in the canonical Dirichlet$(\alpha_0 + k_1, \ldots, \alpha_0 + k_c)$ posterior for $\vec{\rho}$, where $k$ represents the number of sample members within each category.

To make an MCMC sampler for this posterior on $\mathbf{y}$, not much has to change compared to the binomial sampler above. Below, we provide basic pseudocode to sample from this posterior.

**INPUT**: $\mathbf{x}$, $N$, $B$, $\vec{\alpha}$

**OUTPUT**: $B$ realizations of $\mathbf{y}$

Initialize $\mathbf{y}$ (any valid population will work)

**FOR** $i$ **IN** $1, \ldots, B$

  **FOR** $j$ **IN** $1, \ldots, N$

    Propose $\mathbf{y}^*$ and $\vec{\rho}^*$, where $y_j$ is changed via a symmetric proposal, e.g. sample from all *other* categories with equal probability

    Calculate $a = \min\left(1, \frac{f(\mathbf{y}^*, \vec{\rho}^*|\mathbf{x}, \vec{\alpha})}{f(\mathbf{y}, \vec{\rho}|\mathbf{x}, \vec{\alpha})}\right)$

    Set $\mathbf{y} = \mathbf{y}^*$ with probability $a$

  **SAVE** $\mathbf{y}$

**RETURN** $B$ realizations of $\mathbf{y}$

Here,

$$f(\mathbf{y}, \vec{\rho}|\mathbf{x}, \vec{\alpha}) \propto \text{Multi}\Big(\sum_{i=1}^{n} 1_{\{x_i=1\}}, \dots, \sum_{i=1}^{n} 1_{\{x_i=c\}})|n, \rho\Big) \times \binom{N}{N\rho_1, \dots, N\rho_c}^{-1} \times \text{Dir}(\vec{\rho}|\vec{\alpha}),$$

where $\rho = \frac{1}{N}\Big\{\sum_{j=1}^{N} 1_{\{y_j=1\}}, \dots, \sum_{j=1}^{N} 1_{\{y_j=c\}}\Big\}$.

Depending how you code multiple categories, working with them can be quite difficult inside **R**. The easiest categorical encoding to work with is $\mathbf{C} = \{1, 2, \dots, c\}$. This simplifies things significantly because it allows us to use tabulate, which speeds up the process considerably. The code provided for this section (found in Appendix 7.1.2) may fail if you pass it a sample with either of the following properties:

- A category denoted as 0 (e.g. `x = c(1, 0, 1, 0, 1, 2, 1, 2)`)
- Either *missing* or *non-consecutive* categories (e.g. `x = c(1, 3, 1, 1, 3, 1, 3, 3)`) – it will assume that there is a category 2 that was not observed in the sample

To illustrate, we synthesize populations $\mathbf{y}$ of size $N = 100$ from a sample $\mathbf{x}$ of size $n = 20$. Let $\mathbf{C} = \{1, 2, 3, 4\}$, and the tabulated sample categories be $\mathbf{k} = \{3, 5, 2, 10\}$. We calculate the empirical distribution of $\vec{\rho} = \frac{1}{N}\{\sum_i^N 1_{\{y_i=1\}}, \dots, \sum_i^N 1_{\{y_i=4\}}\}$ for 20000 synthetic populations, and graphically show the distribution of $\vec{\rho}$ across these populations. Figure 3.6 shows these results, and compares the empirical distributions against the canonical posterior for $\vec{\rho}$ that would result from using a Dirichlet($\vec{0.5}$) prior.

### 3.2.2 Dependent Sampling Schemes

The main difference between this section and section above is that here we will assume that samples were taken without replacement. This means that if a sample $\mathbf{x}$ has certain properties, then our population $\mathbf{y}$ should contain, as a subset, population members identical

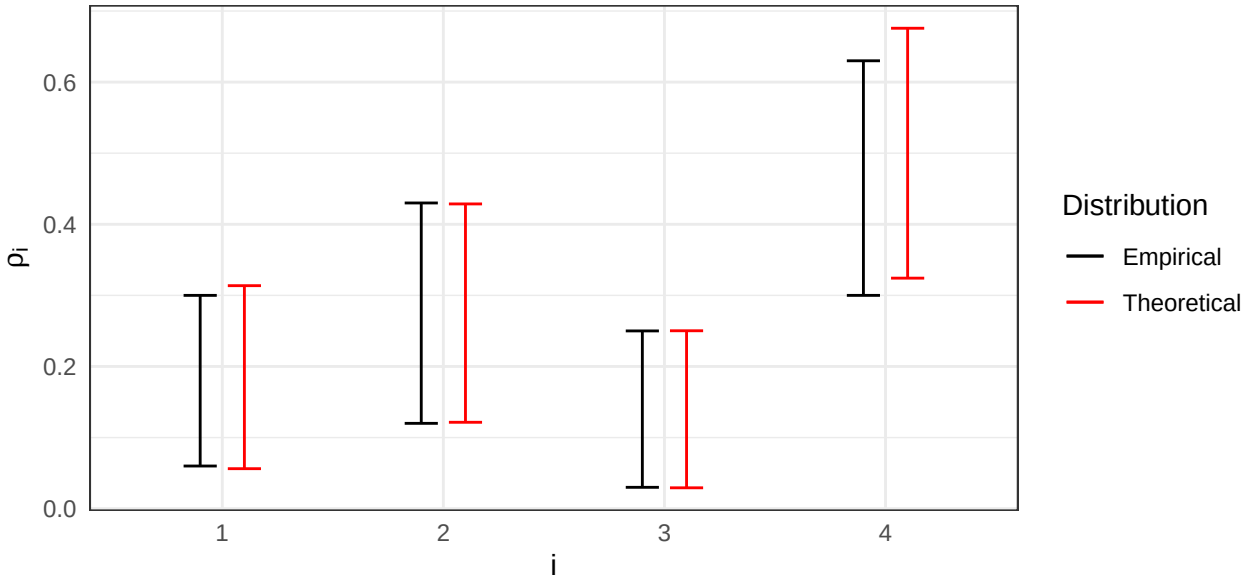


Figure 3.6: Empirical distribution (black) and theoretical distribution (red) for elements of $\vec{\rho}$, using Dirichlet$(0.5, \ldots, 0.5)$ prior. Bars represent 90% intervals.

to $\mathbf{x}$. Above we were using Binomial and Multinomial sample distributions for $f(\mathbf{x}|\mathbf{y}, \theta)$, but now we will have to switch to distributions better suited for samples taken without replacement.

#### 3.2.2.1 Binary Populations

If our sample was taken without replacement, and population members are represented by a single binary variable, then the correct sampling distribution governing $\mathbf{x}$ is Hypergeometric. A parameterization of the Hypergeometric is as follows. Let $f(\mathbf{x}, k|\mathbf{y}, K)$ denote the Hypergeometric probability mass function, where:

- $\mathbf{y}$ is the population (i.e., the vector of $\{0, 1\}$ with length $N$)
- $K$ is the number of successes in the population $\mathbf{y}$: $\sum_{i=1}^{N} 1_{\{y_i=1\}}$
- $\mathbf{x}$ is the sample, with length $n$
- $k$ is the number of successes in the sample $\mathbf{x}$: $\sum_{i=1}^{n} 1_{\{x_i=1\}}$

Now, we apply the above theory.

$$f(\mathbf{Y} = \mathbf{y}, K|\mathbf{X} = \mathbf{x}, k) = \frac{f(\mathbf{X} = \mathbf{x}, k|\mathbf{Y} = \mathbf{y}, K)f(\mathbf{Y} = \mathbf{y}, K)}{f(\mathbf{X} = \mathbf{x}, k)}$$
$$\propto f(k|K)f(\mathbf{Y} = \mathbf{y}|K)\pi(K)$$

For $f(\mathbf{Y} = \mathbf{y}|K)$, we will reuse the same logic we used when using the Binomial distribution. There is no reason for us to favor certain configurations of $\mathbf{y}$, so we let the distribution be flat when we condition on $K$. Thus, $f(\mathbf{Y} = \mathbf{y}|K) = \binom{N}{K}^{-1}$.

We also need to choose $\pi(K)$, the prior on the number of successes in the population. If you wanted, you could simply look up the conjugate prior for the Hypergeometric distribution, and find that it is Beta-binomial. Another way to find this prior is by modeling $K$ and $\rho$ (the same parameter from the Binomial distribution) hierarchically. If we are working with $\rho$ from the Binomial distribution, we would likely want to use some version of the conjugate prior, Beta(a, b).

From Jeffreys (1946, 1961), the induced prior on $K$ can be found by constructing a hierarchical prior of the form

$$\text{Binom}(K|\rho)\text{Beta}(\rho|a, b)$$

and then marginalizing to find $\pi(K)$. Thus,

$$\begin{aligned}\pi(K) &= \int_0^1 \binom{N}{K}\rho^K(1-\rho)^{N-K}\frac{1}{\beta(a,b)}\rho^{a-1}(1-\rho)^{b-1}d\rho \\ &= \binom{N}{K}\frac{\beta(K+a, N-K+b)}{\beta(a,b)} \\ &= \binom{N}{K}\frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\frac{\Gamma(K+a)\Gamma(N-K+b)}{\Gamma(N+a+b)}\end{aligned}$$

This is a Beta-binomial$(K|N,a,b)$ distribution. For certain values of $a$ and $b$, this will simplify much further, but not in general. Lets visualize this prior for $N = 100$.

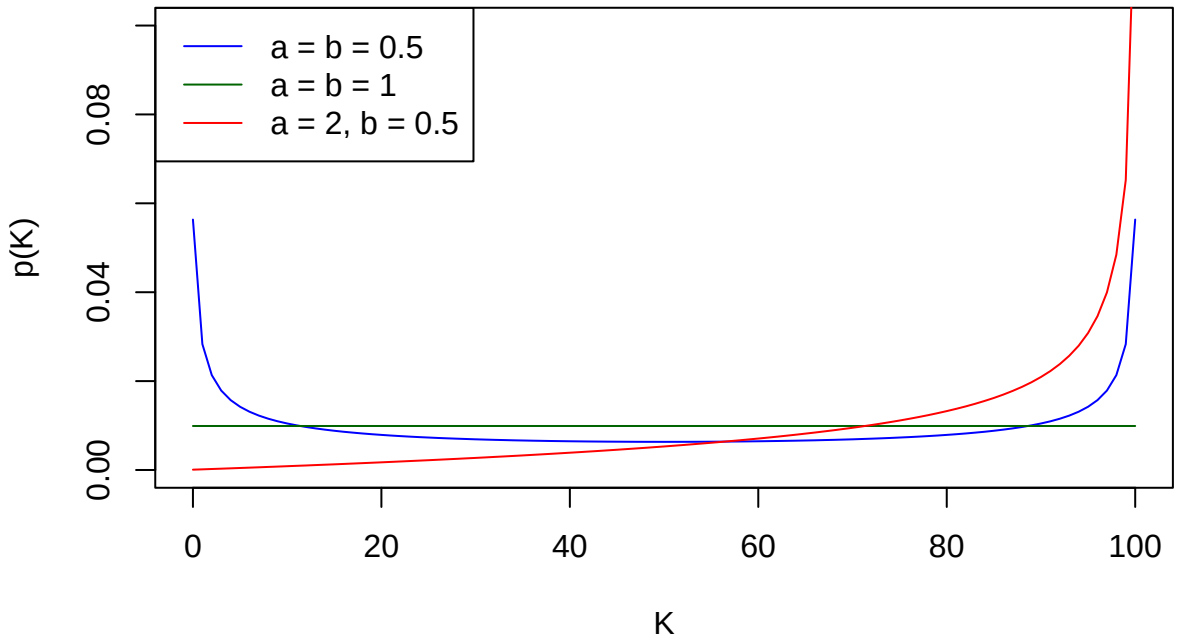


Figure 3.7: Visualization of Beta-binomial prior on $K$ with $N = 100$, for various values of $a, b$.

It is not surprising that the shape of these curves resemble the underlying Beta$(a,b)$ priors. Whatever choice of $a$ and $b$ we make, creating the updated prior from here is fairly straightforward. For our purposes, we will consider $\pi(K)$ to be the induced prior when $a = b = 0.5$

We once again provide the following pseudocode (full code can be seen in Appendix 7.1.3)to generate synthetic populations using this posterior.

**INPUT**: $\mathbf{x}$, $N$, $B$, $a$, $b$

**OUTPUT**: $B$ realizations of $\mathbf{y}$

Initialize $\mathbf{y}$ (any valid population will work)

**FOR** $i$ **IN** $1, \ldots, B$

  **FOR** $j$ **IN** $1, \ldots, N$

    Propose $\mathbf{y}^*$ and $K^*$, where $y_j$ is changed

    Calculate $\alpha = \min\left(1, \frac{f(\mathbf{y}^*, K^* | \mathbf{x}, a, b)}{f(\mathbf{y}, K | \mathbf{x}, a, b)}\right)$

    Set $\mathbf{y} = \mathbf{y}^*$ with probability $\alpha$

  **SAVE** $\mathbf{y}$

**RETURN** $B$ realizations of $\mathbf{y}$

Here,

$$f(\mathbf{y}, K | \mathbf{x}, a, b) \propto \text{Hyper}(\mathbf{x}, k | \mathbf{y}, K) \times \binom{N}{K}^{-1} \times \text{Beta-binomial}(K | N, a, b).$$

To illustrate, we will use the exact same setup and sample as we did within section 3.2.1.1; we synthesize populations $\mathbf{y}$ of size $N = 100$ from a sample $\mathbf{x}$ of size $n = 20$ where $\sum_i^n x_i = 6$. We calculate the distribution of $K = \sum_i^N y_i$ for 50,000 populations, and graphically show the empirical distribution of $K$ across these populations. Figure 3.8 shows these results, and compares the empirical distributions against the "canonical" posterior for $K$ that would result from an induced prior on $K$ using a Beta$(0.5, 0.5)$ prior, as described above. The resulting posterior on $K - k$ out of $N - n$ population members is Beta-Binomial with shape

parameters $a = 6 + 0.5$ and $b = 14 + 0.5$.

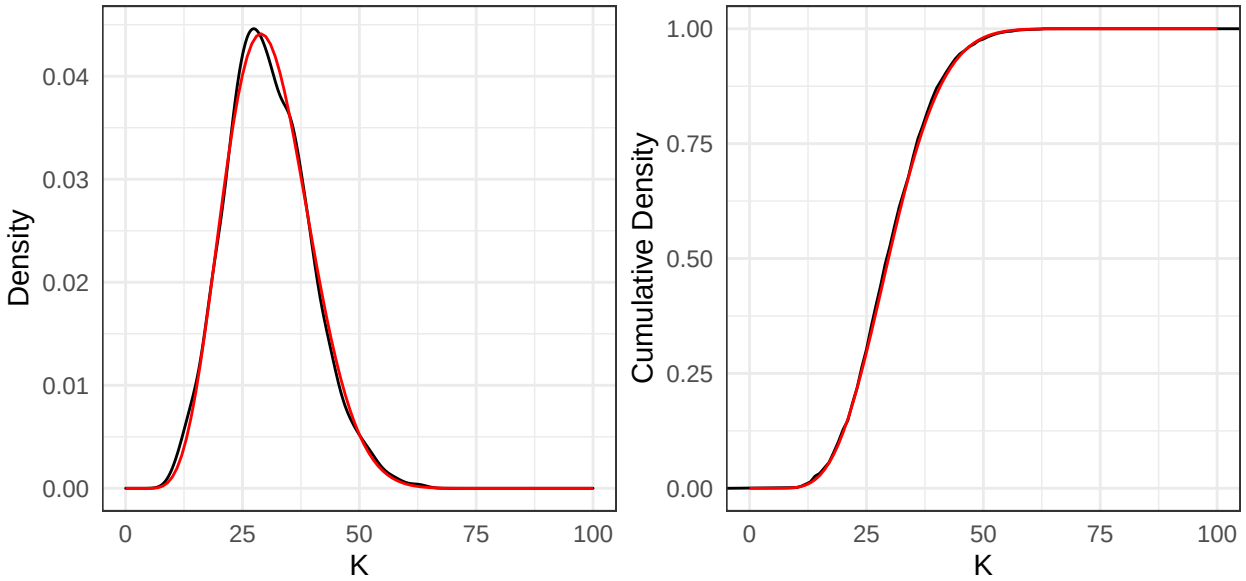


Figure 3.8: Empirical pdf (left) and cdf (right) for K with theoretical posterior from traditional Bayesian analysis using Beta-Binomial(0.5, 0.5) prior shown in red.

Here we see that the empirical distribution on $K$ matches the canonical posterior that we would arrive at by using traditional Bayesian analysis with a popular conjugate prior.

#### 3.2.2.2 Categorical Populations

If our population members are defined by a categorical variable with $c \geq 3$ categories, we can no longer use the Hypergeometric distribution as our likelihood for $\mathbf{x}$. Instead, we must use the Multivariate Hypergeometric distribution, which extends the Hypergeometric distribution to multiple categories in the same way that the Multinomial is seen as an extension of the Binomial in the same fashion as the Hypergeometric. We define the MVHyper$(\mathbf{x}, \mathbf{k}|\mathbf{y}, \mathbf{K})$ as follows:

- $\mathbf{y}$ is the population (i.e., the vector of $\{1, 2, \ldots, c\}$ with length $N$)
- $\mathbf{K}$ is the $c$-vector denoting categorical membership in $\mathbf{y}$: $\left\{ \sum_{i=1}^{N} 1_{\{y_i=1\}}, \ldots, \sum_{i=1}^{N} 1_{\{y_i=c\}} \right\}$
- $\mathbf{x}$ is the sample, with length $n$

- $\mathbf{k}$ is the $c$-vector denoting categorical membership in $\mathbf{x}$: $\left\{ \sum_{i=1}^{n} 1_{\{x_i=1\}}, \dots, \sum_{i=1}^{n} 1_{\{x_i=c\}} \right\}$

Now, we once again apply the theory above.

$$\begin{aligned} f(\mathbf{Y}=\mathbf{y}, \mathbf{K} | \mathbf{X}=\mathbf{x}, \mathbf{k}) &= \frac{f(\mathbf{X}=\mathbf{x}, \mathbf{k} | \mathbf{Y}=\mathbf{y}, \mathbf{K}) f(\mathbf{Y}=\mathbf{y}, \mathbf{K})}{f(\mathbf{X}=\mathbf{x}, \mathbf{k})} \\ &\propto f(\mathbf{k}|\mathbf{K}) f(\mathbf{Y}=\mathbf{y}|\mathbf{K}) \pi(\mathbf{K}) \end{aligned}$$

For $f(\mathbf{Y}=\mathbf{y}|\mathbf{K})$, we will once again use a flat distribution. Thus, $f(\mathbf{Y}=\mathbf{y}|\mathbf{K}) = \binom{N}{K_1, \dots, K_c}^{-1}$.

Next, we need to find a prior $\pi(\mathbf{K})$. We will follow the same steps that we followed for the Hypergeometric case. We can no longer use a Beta prior to induce our prior on $\mathbf{K}$, since $\mathbf{K}$ is multivariate. Instead, we must use a Dirichlet($\vec{\alpha}$) prior on $\vec{\rho}$ and induce a prior on $\mathbf{K}$. Suppose there are $c$ categories; following the same steps as above, we have:

$$\begin{aligned} \pi(\mathbf{K}) &= \int_0^1 \binom{N}{\mathbf{K}_1, \dots, \mathbf{K}_c} \prod_{i=1}^{c} \rho_i^{\mathbf{K}_i} \frac{1}{\beta(\vec{\alpha})} \prod_{i=1}^{c} \rho_i^{\alpha_i - 1} d\rho \\ &= \binom{N}{\mathbf{K}_1, \dots, \mathbf{K}_c} \frac{\beta(\mathbf{K} + \vec{\alpha})}{\beta(\vec{\alpha})} \\ &= \binom{N}{\mathbf{K}_1, \dots, \mathbf{K}_c} \frac{\Gamma(\alpha_0) \prod_{i=1}^{c} \Gamma(K_i + \alpha_i)}{\Gamma(N + \alpha_0) \prod_{i=1}^{c} \Gamma(\alpha_i)}, \end{aligned}$$

where $\alpha_0 = \sum_{i=1}^{c} \alpha_i$. Those very familiar with the Dirichlet-multinomial distribution may recognize this prior as such. Normally the multinomial coefficient is decomposed using Gamma functions, but we refrain from doing so here.

We once again provide pseudocode (full code can be found in Appendix 7.1.4) to generate synthetic populations using this posterior.

**INPUT**: $\mathbf{x}$, $N$, $B$, $\vec{\alpha}$

**OUTPUT**: $B$ realizations of $\mathbf{y}$

Initialize $\mathbf{y}$ (any valid population will work)

**FOR** $i$ **IN** $1, \dots, B$

  **FOR** $j$ **IN** $1, \dots, N$

    Propose $\mathbf{y}^*$ and $\mathbf{K}^*$, where $y_j$ is changed

    Calculate $a = \min\left(1, \frac{f(\mathbf{y}^*, \mathbf{K}^* | \mathbf{x}, \vec{\alpha})}{f(\mathbf{y}, \mathbf{K} | \mathbf{x}, \vec{\alpha})}\right)$

    Set $\mathbf{y} = \mathbf{y}^*$ with probability $a$

  **SAVE** $\mathbf{y}$

**RETURN** $B$ realizations of $\mathbf{y}$

Here,

$$f(\mathbf{y}, \mathbf{K} | \mathbf{x}, \vec{\alpha}) \propto \text{MVHyper}(\mathbf{x}, \mathbf{k} | \mathbf{y}, \mathbf{K}) \times \binom{N}{K_1, \dots, K_c}^{-1} \times \text{Dirichlet-multinomial}(\mathbf{K} | N, \vec{\alpha}).$$

To illustrate, we will use the exact same setup and sample as we did within section 3.2.1.2; we synthesize populations $\mathbf{y}$ of size $N = 100$ from a sample $\mathbf{x}$ of size $n = 20$. Let $C = \{1, 2, 3, 4\}$, and the tabulated sample categories be $\mathbf{k} = \{3, 5, 2, 10\}$. We calculate the distribution of $\mathbf{K} = \{\sum_i^N 1_{\{y_i=1\}}, \dots, \sum_i^N 1_{\{y_i=4\}}\}$ for 20,000 populations, and graphically show the empirical distribution of $\mathbf{K}$ across these populations. Figure 3.9 shows these results, and compares the empirical distributions against the "canonical" posterior for $\mathbf{K}$ that would result from an induced prior on $\mathbf{K}$ using a Dirichlet$(0.5, \dots, 0.5)$ prior, as described above. We compare

against traditional Bayesian inference on $\mathbf{K}$, using a conjugate Dirichlet-multinomial prior with $\vec{\alpha} = \{0.5\}^c$. The resulting posterior on $\mathbf{K}-\mathbf{k}$ out of $N-n$ draws is Dirichlet-multinomial with parameter $\vec{\alpha} = \{3.5, 5.5, 2.5, 10.5\}$.

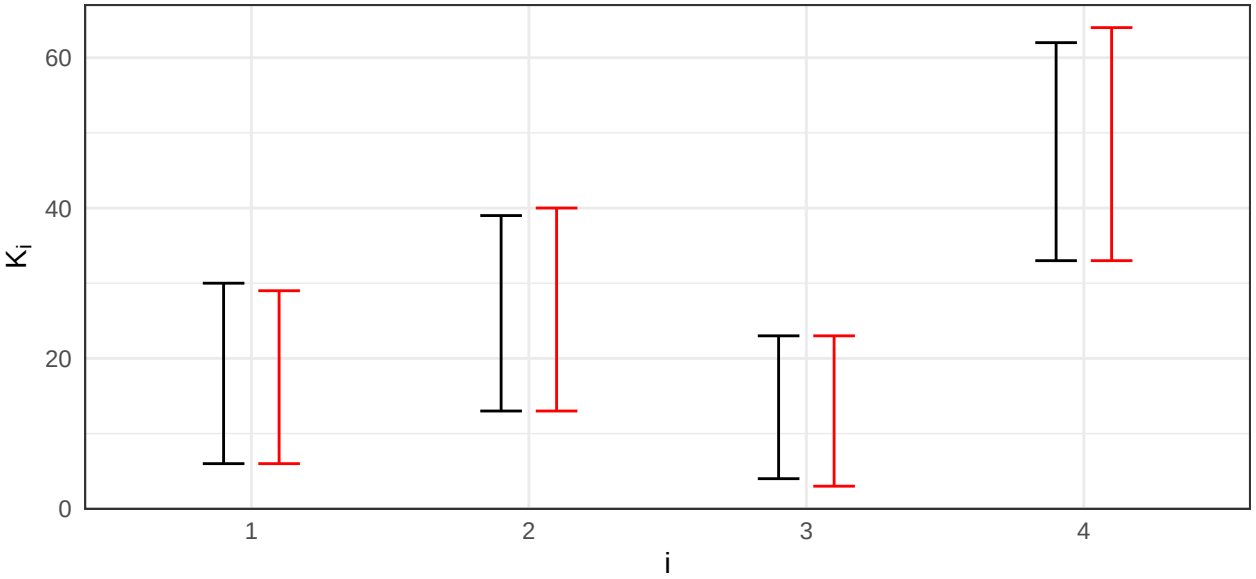


Figure 3.9: Empirical distribution (black) and theoretical distribution (red) for elements of K, using Dirichlet-Multinomial(0.5,..., 0.5) prior on K.

Here we see that the empirical distribution on $\mathbf{K}$ matches the canonical posterior that we would arrive at by using traditional Bayesian analysis with a popular conjugate prior. To conclude this section, we now apply what we have learned to a real-world example by returning to the Blacksburg data.

### 3.2.3 Example using Blacksburg Data

In section 3.1.2, we saw that using a prior that puts equal mass on each possible population resulted in undesirable – though perhaps expected – results, in which the posterior was shrunk towards uniformity across $\mathbf{K}$. In this example, instead of using an equal mass prior

on each population, we use a hierarchical prior of the form $f(y|\mathbf{K})f(\mathbf{K})$; specifically, we use

$$f(y|\mathbf{K})f(\mathbf{K}) = \binom{N}{\mathbf{K}_1, ..., \mathbf{K}_c} \frac{\Gamma(\alpha_0)\prod_{i=1}^{c}\Gamma(K_i+\alpha_i)}{\Gamma(N+\alpha_0)\prod_{i=1}^{c}\Gamma(\alpha_i)},$$

where $\vec{\alpha} = \{0.5\}^{10}$. This $\vec{\alpha}$ parameter is from a Dirichlet distribution, though the Dirichlet kernel has disappeared through integration (see section 3.2.2.2). This choice of $\vec{\alpha}$ yields the reference prior for the Multivariate Hypergeometric. From Figure 3.10, we see that the resulting populations do not appear shrunk towards a uniform $\mathbf{K}$; instead, the mean proportion of the population within each category is almost perfectly centered at the observed sample proportions.

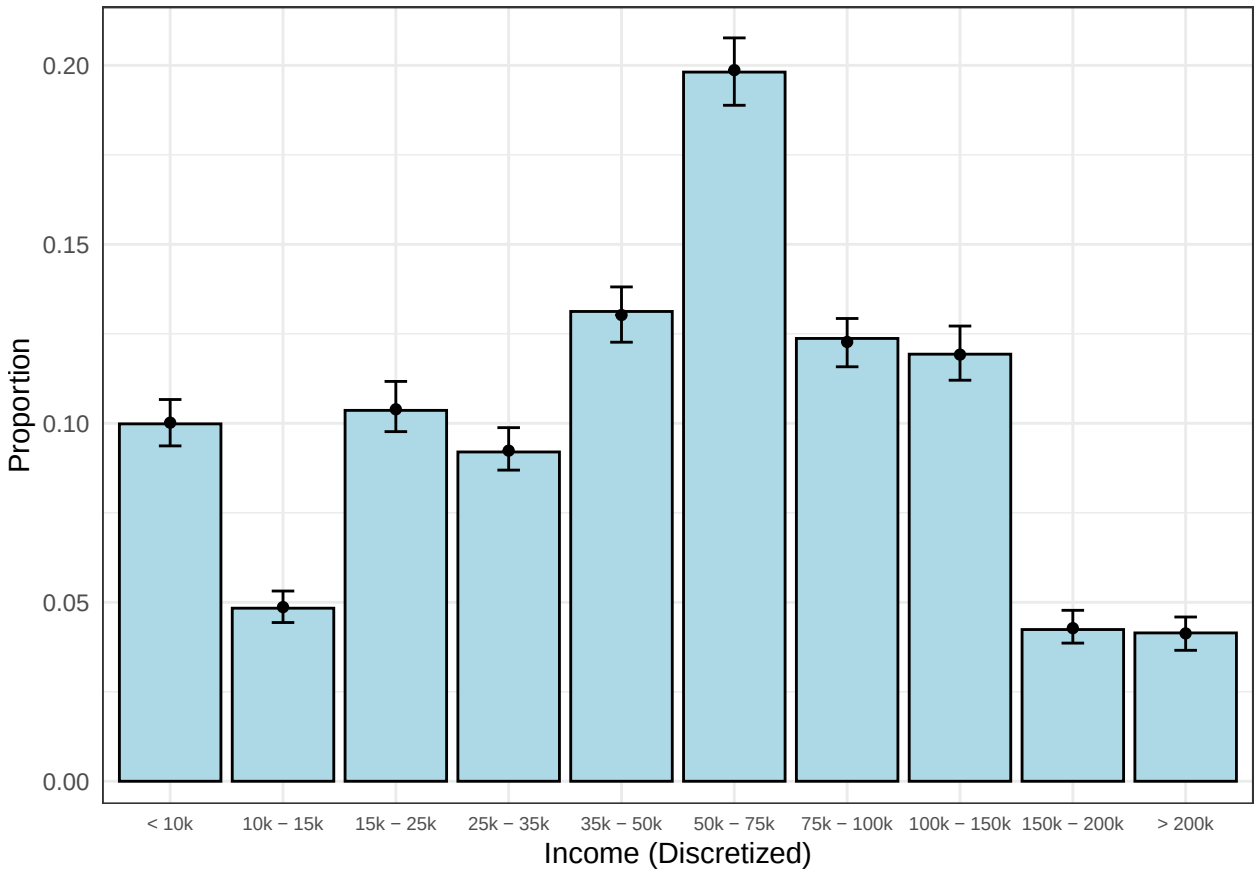


Figure 3.10: Income sample data from ACS PUMS (blue bars) with distribution of Incomes generated via synthetic populations with an equal mass prior; $5^{\text{th}}$ and $95^{\text{th}}$ quantiles represented via error bars, with mean represented with a dot.

## 3.3 Dirichlet Spacing Prior

Hierarchical priors, such as those found in Section 3.2, for i.i.d. discrete population members, have both mathematical and practical advantages in that they are easy to interpret and the prior distribution for $\mathbf{y}$ is stated explicitly:

$$\mathbf{y} \sim \int_\theta \prod f(y_i|\theta)\pi(\theta)d\theta.$$

However, in some cases, we need flexibility in specifying a population prior. For example, when considering income (a common continuous feature of populations), we often observe very different income distributions across different populations. To show this, Figure 3.11 includes income histograms from three populations, i.e. PUMAs ($N = 2057, 2571, 4061$ respectively), with clearly *very* different distributions of income.

While we could model the three populations in Figure 3.11 completely separately, for example using Gamma distributions with different parameter values, we instead seek a prior for $\mathbf{y}$ that will allow potentially extreme deviations from a base distribution that we observe in real communities.

Specifically, the approach we take is to modify the priors $G$ (e.g., those seen in Section 3.2) through the use of a Dirichlet distribution on the spacings $p$ between observations, where

$$\begin{aligned} f(\mathbf{p}) &= \text{Dir}(\alpha, N+1) \\ \mathbf{u} &= \text{cumsum}(p_1, p_2, \dots, p_N) \\ F(\mathbf{y}|\mathbf{p}) &= G^{-1}(\mathbf{u}). \end{aligned}$$

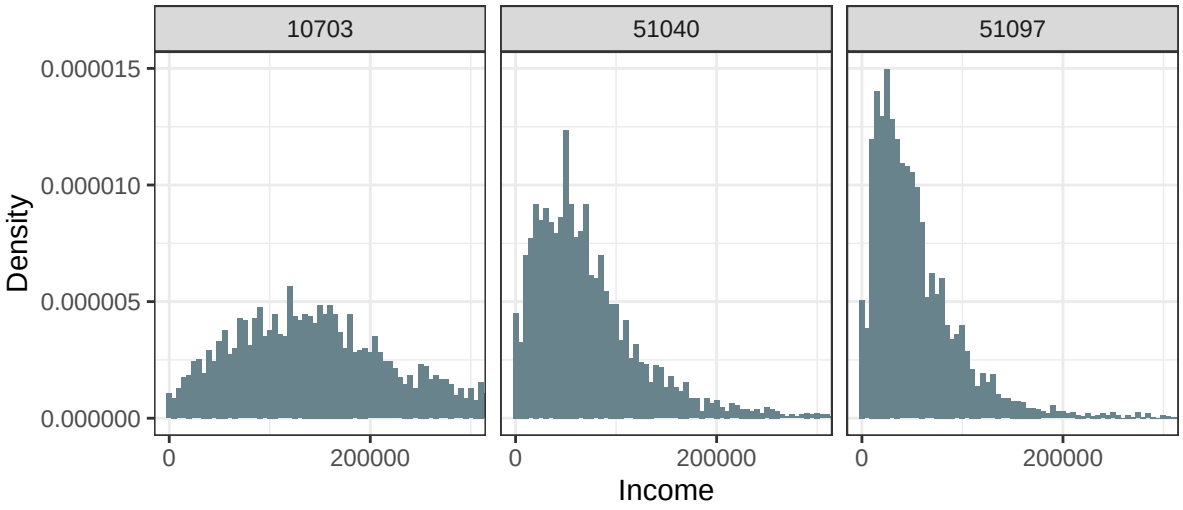


Figure 3.11: Histogram of incomes from three different PUMAs (Blacksburg's PUMA is in the center).

Notably, the concentration parameter $\alpha$ of a Dirichlet distribution serve to control to what degree the distribution of populations can deviate from the base distribution $G$. By applying standard transformation of variables we obtain the resulting density for this process,

$$\pi(\mathbf{y}|G_\theta, \alpha) \propto \prod_{i=1}^{N+1} [G_\theta(y_{(i)}) - G_\theta(y_{(i-1)})]^{\alpha-1} \times \prod_{i=1}^{N} g_\theta(y_i) \times \pi(\theta), \tag{3.3}$$

where $y_{(0)} = G^{-1}(0)$, $y_{(N+1)} = G^{-1}(1)$, $\theta$ are parameters of base distribution $G$, and $\pi(\theta)$ is a hyperprior for $\theta$ (if necessary). Henceforth, we refer to this distribution as the Dirichlet Spacing prior and introduce notation Dirichlet-Sp$(G, \alpha)$ to refer to this distribution.

Samples from a Dirichlet-Sp$(G, \alpha)$ are depicted in Figure 3.12. This figure shows a single synthetic population of size $N = 10$ created with three different values of $\alpha$. Specifically, graph (a) shows i.i.d. population members with $\alpha = 1$, graph (b) shows what happens when $\alpha > 1$, and graph (c) shows the result of $\alpha < 1$.

Similar to how a Gaussian Process prior is updated when there is observed data, we must update our prior to reflect the observed $\mathbf{x}$. We do so by including $\mathbf{x} \in \mathbf{y}$, which can be

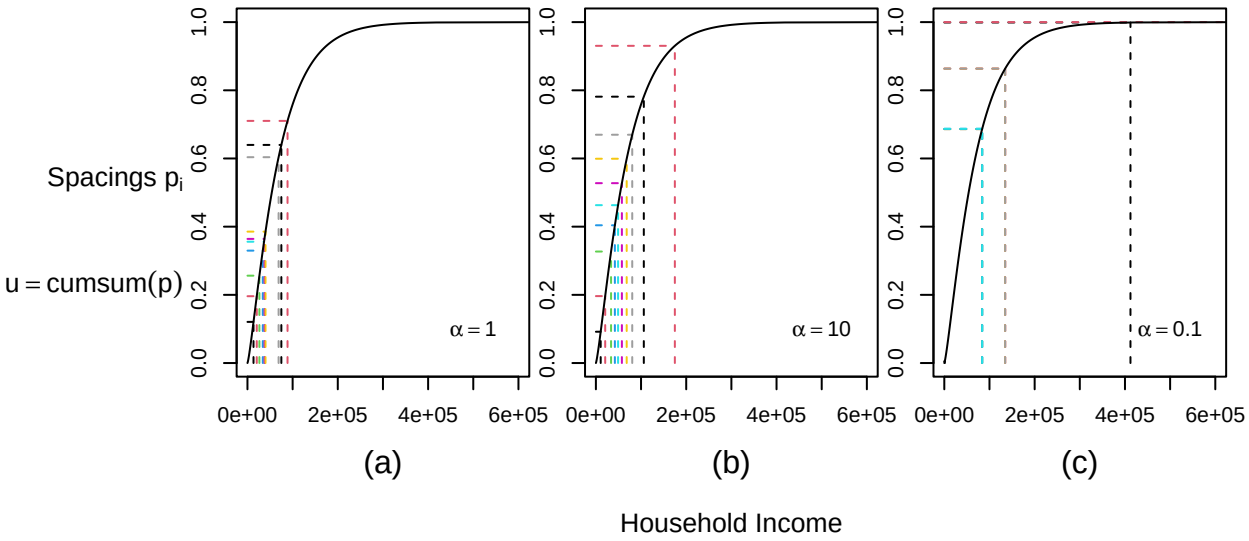


Figure 3.12: Adapting the inverse-cdf construction to produce a population: (a) standard construction of i.i.d. population members $y$ using $F^{-1}(u) = y$, (b) a more regularized population is produced from spacings $p \sim \text{Dir}(\mathbf{1}_{N+1} \times 10)$, and (c) a more clustered population is produced from spacings $p \sim \text{Dir}(\mathbf{1}_{N+1} \times \frac{1}{10})$.

accomplished via MCMC sampling. For simple examples where $\theta$ and $\alpha$ are known, this can be performed using a basic MCMC algorithm, which we outline below:

**INPUT**: $\alpha$ - concentration parameter

$G$ - base distribution

$N$ - population size

$T$ - number of iterations

**OUTPUT**: $T$ realizations of $\mathbf{y}$

Initialize $\mathbf{y}$

**FOR** $i$ **IN** $1, \ldots, T$

**FOR** $j$ **IN** $n+1, \ldots, N$

Propose $y_j^*$ from $p(y_j^*|y_j)$

Calculate $a = \min\left(1, \frac{f(\mathbf{y}^*|G,\alpha)}{f(\mathbf{y}|G,\alpha)} \times \frac{p(y_j|y_j^*)}{p(y_j^*|y_j)}\right)$

Set $y_j = y_j^*$ with probability $a$

**SAVE y**

**RETURN** $T$ realizations of $\mathbf{y}$

Computationally, it is advantageous to work in $(0, 1)$ space and then convert using base distribution $G$ at the very end. For small populations, this MCMC will work even with a simple proposal such as a Uniform distribution. For large populations (especially those with a small $\alpha$ value), this sampler will run into computational issues. An alternative is explored in Section 3.4.

The Dirichlet Spacing prior described above is explored more with examples in this section. Extending these ideas to populations for which multiple sources of information exist is explored in Chapter 4. Examples with multivariate populations are presented in Chapter 5, though a general approach for multivariate attributes is beyond the scope of this thesis.

### 3.3.1 Estimating $\alpha$

We have introduced a new parameter $\alpha$, but not discussed how to estimate it, or otherwise model it. One way to represent $\alpha$ is to consider the quantity $\alpha = \frac{N_{\text{eff}+1}}{N+1}$. We choose the denominator $N + 1$ because there are $N + 1$ spacings in a population of size $N$. This transformation yields $\alpha = 1$ when $N_{\text{eff}} = N$. We refer to $N_{\text{eff}}$ as the *effective population size*. Our prior for $\mathbf{y}$ (3.3) differs from i.i.d. populations by creating *clusters* of observations via the Dirichlet distribution. If we tabulate the number of *semi-unique* population members (for example, count all population members *not* within $\epsilon$ of another population member, for small $\epsilon$), we find that increasing $N_{\text{eff}}$ increases this quantity, while decreasing $N_{\text{eff}}$ decreases this quantity.

Another of measuring the number of *semi-unique* population members is to use a clustering algorithm. Since we know every clustering algorithm would give different results, it would be naive to expect a single algorithm to choose $k$ clusters equivalent to our $N_{\text{eff}}$. However, we should at the very least see that increasing $N_{\text{eff}}$ increases the number of estimated clusters, $k$. To illustrate, we create several populations of size $N = 1000$ with $N_{\text{eff}} = 50$ and $N_{\text{eff}} = 100$, using a continuous uniform as base distribution $G$. Figure 3.13 shows the results of a K-Means algorithm, where we see that our elbow plots indicate that $k$ is higher when $N_{\text{eff}} = 100$ compared to when $N_{\text{eff}} = 50$ (though the difference is less than we would have hoped for).

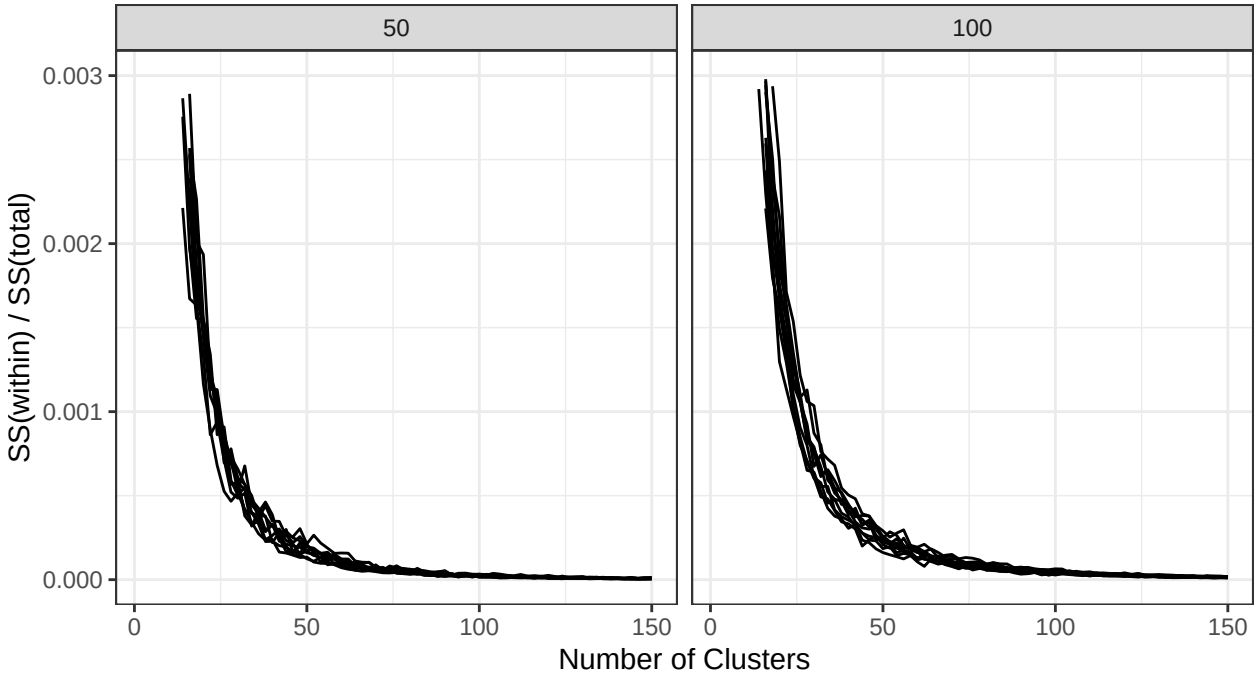


Figure 3.13: K-Means clustering results for populations created with $N_{\text{eff}} = 50$ (left) and $N_{\text{eff}} = 100$ (right).

Fortunately, small changes in $N_{\text{eff}}$, relative to the size of $N$, have very little impact on the resulting synthetic populations. Despite this, it is abundantly clear that we cannot use a clustering algorithm to estimate $N_{\text{eff}}$. We must instead consider other options. One potential solution is to estimate the parameter empirically. Another option is to place a hyperprior on $N_{\text{eff}}$ (or the resulting $\alpha = \frac{N_{\text{eff}}+1}{N+1}$). Here we will explore both options.

**Empirical Estimation**

The first option is to develop one (or several) empirical estimators for $N_{\text{eff}}$. While certainly not the best estimators under any criteria, we present two such estimators. To do so, we let $\alpha = f(z)$, where $z$ is a dummy variable only relevant for these estimators, and then optimize the Dirichlet likelihood.

1. $\widehat{N_{\text{eff}}} = \arg max\ z\ \text{Dir}\left(\mathbf{p}|\frac{z(1-((z-1)/z))^n)+1}{n+1}\right)$
2. $\widehat{N_{\text{eff}}} = \arg max\ z\ \text{Dir}\left(\mathbf{p}|\frac{\sum_{i=1}^{z}(1-((z-i)/z)^n)+1}{n+1}\right)$

In both cases, $\mathbf{p}$ are the spacings of the (ordered) observed sample $\mathbf{x}$, and $n$ is the sample size. Both of these estimators are based on the fact that a Maximum Likelihood Estimation (MLE) estimator for a Dirichlet distribution's $\alpha$ parameter can be found quite easily when $\alpha$ is a constant vector. The problem is that we cannot directly do this because we lack the full population, instead only having access to a random sample. Thus, both of our estimators are based on essentially pretending that our sample is the population, and modifying the resulting MLE estimator. However, we modify the solution using $f(z)$, which accounts for the fact that the full population is not being observed.

Figure 3.14 shows the behavior of these estimators for several synthetic populations created with known $N_{\text{eff}}$ values. The performance of each estimator varies with the values of $n$, $N$, and the true $N_{\text{eff}}$.

Here we see that both estimators tend to underestimate $N_{\text{eff}}$. This opens up potential opportunities to improve these estimators, but we choose to instead focus on our second option: using a hyperprior.

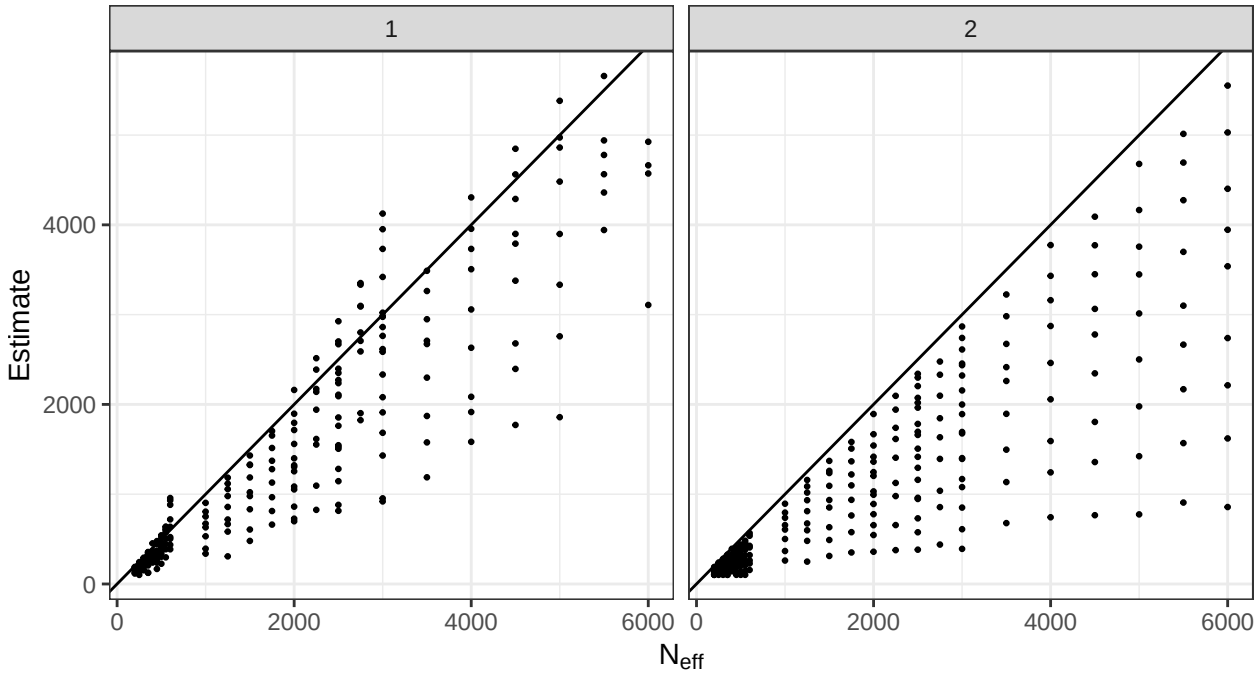


Figure 3.14: Comparison of performance for two estimators.

**Hyperparamterization**

Instead of estimating $N_{\text{eff}}$, another option is to simply put a hyperprior distribution on it (or $\alpha$) and learn $N_{\text{eff}}$ or $\alpha$ aposteriori. It is relatively simple to modify the Dirichlet Spacing prior to include a hyperprior on $\alpha$. Remember that going between $\alpha$ and $N_{\text{eff}}$ is a simple linear transformation (since $N$ is fixed), so we can easily solve for $N_{\text{eff}}$ if so desired.

Including a hyperprior on $\alpha$ in the full Dirichlet-Sp$(G, \alpha)$ process is rather straightforward. We simply modify Equation (3.3) by appending $\pi(\alpha)$. This yields

$$f(\mathbf{y}, \alpha | G_\theta) \propto \prod_{i=1}^{N+1} [G_\theta(y_{(i)}) - G_\theta(y_{(i-1)})]^{\alpha-1} \times \prod_{i=1}^{N} g_\theta(y_i) \times \pi(\theta) \times \pi(\alpha),$$

and for our purposes we will ignore $\pi(\theta)$, and assume base distribution parameters are known.

To evaluate the performance, we conduct a simulation study using a Unif$(0, 1)$ base distri-

bution, with a fixed population size $N = 500$ and sample size $n = 200$. We first create a *true* population with a chosen $\alpha \in (0.2, 0.9)$, then generate a sample from this population. Using this random sample, we use MCMC to sample from the joint posterior for $\mathbf{y}, \alpha$. We then construct an equal-tailed credible interval for $\alpha$, and check to see whether it contains the true value. This entire process is replicated 1000 times.

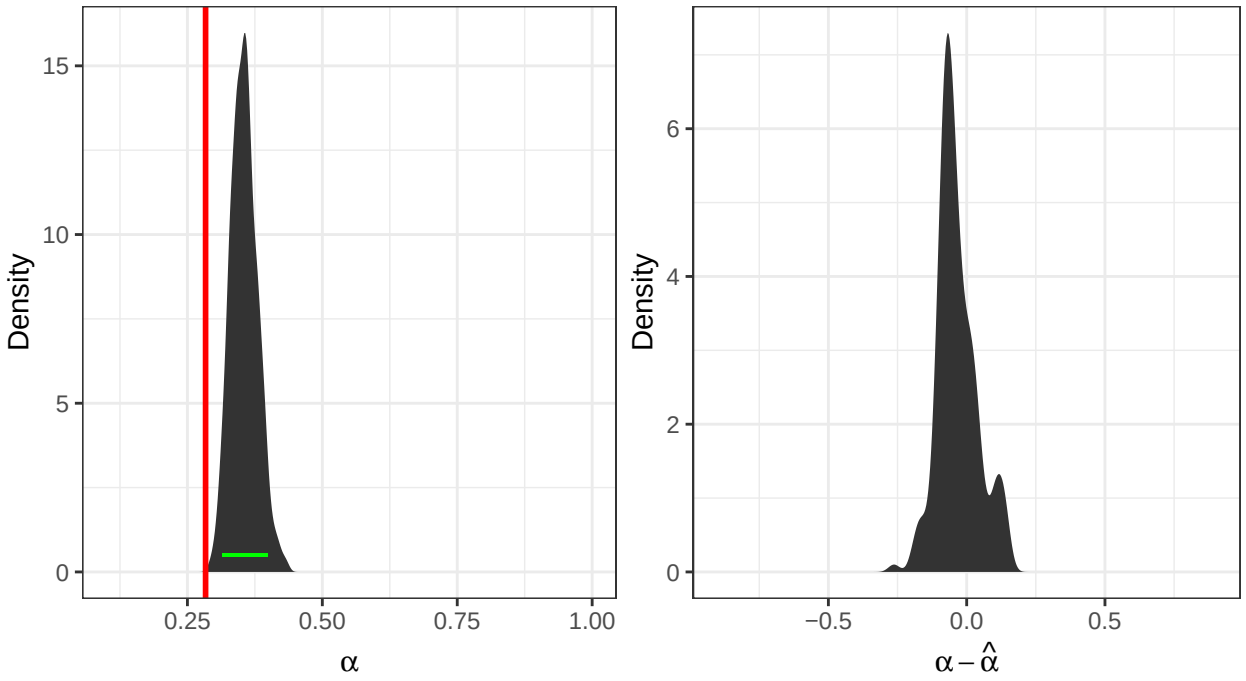


Figure 3.15: An example posterior for $\alpha$ (left), showing equal-tailed confidence interval (green) and the true $\alpha$ (red); the density of $\alpha - \hat{\alpha}$ (right).

Figure 3.15 shows the results of this simulation study. Here we see the effectiveness of estimating $\alpha = \frac{N_{\text{eff}}+1}{N+1}$ via posterior medians. Our overall coverage rate (from 200 simulations) for 90% equal-tailed credible intervals was 88.5%.

### 3.3.2 Quantifying Model Fit

Since we are in essence sampling from the posterior predictive distribution $f(\mathbf{y}|\mathbf{x})$ under a certain model, one way of quantifying model fit is to use Posterior Predictive Model Checking

(PPMC) (Rubin 1984; Gelman et al. 1995; Gelman, Meng, and Stern 1996). PPMC is a fairly straight-forward; to compare realizations $\mathbf{y}^{rep}$ from a posterior predictive distribution against observed data $\mathbf{y}$, authors propose sampling from the posterior predictive distribution and then calculating a summary statistic $T(\mathbf{y}^{rep})$ and $T(\mathbf{y})$. In general, the distribution of $T(\mathbf{y}^{rep})$ should be near $T(\mathbf{y})$. Deviations away from $T(\mathbf{y})$ are indication that the model being used is flawed.

However, a single summary statistic $T(\cdot)$ is generally not enough in multivariate settings like ours. Instead, we need to use at least a vector summaries of the data, or the data itself. Crespi and Boscardin (2009) propose using a measure of dissimilarity such as distance functions to achieve the same goal. In particular, they calculate distances $d(\mathbf{y}^{obs}, \mathbf{y}^{rep,m})$ for $m$ replicated data sets, and $d(\mathbf{y}^{rep,r}, \mathbf{y}^{rep,s}),\ r \neq s$ among the replicated data sets. In general, if the model fits well, we should expect these sets of distances to have comparable distributions; otherwise, the distributions will be significantly different from each other. One option to test this hypothesis is to use a non-parametric hypothesis test such as the Mann-Whitney U Test (Gibbons et al. 1976).

For example, suppose we observe a sample $\mathbf{y}^{obs}$ of size $n = 20$ from a Normal distribution with unknown $\mu$ but known $\sigma = 1$. If we put a prior on $\mu$, we can construct a posterior for $\mu$ and posterior predictive distribution for $\mathbf{y}$. If we create synthetic populations of size $N = 100$ using the posterior predictive distribution, we can simplify this process to:

1. Sample $\mu_i$ from $\pi(\mu|\mathbf{y}^{obs})$, the posterior for $\mu$
2. Sample $N - n$ observations from Normal$(\mu_i, 1)$
3. Create $\mathbf{y}^{rep,i}$ by concatenating $\mathbf{y}^{obs}$ and our sampled values

If we repeat this process for $i = 1, \ldots, m$ synthetic populations, we can construct histograms of $d(\mathbf{y}^{obs}, \mathbf{y}^{rep,m})$ and $d(\mathbf{y}^{rep,r}, \mathbf{y}^{rep,s}), r \neq s$. Figure 3.16 shows the resulting histograms

that we want to compare.

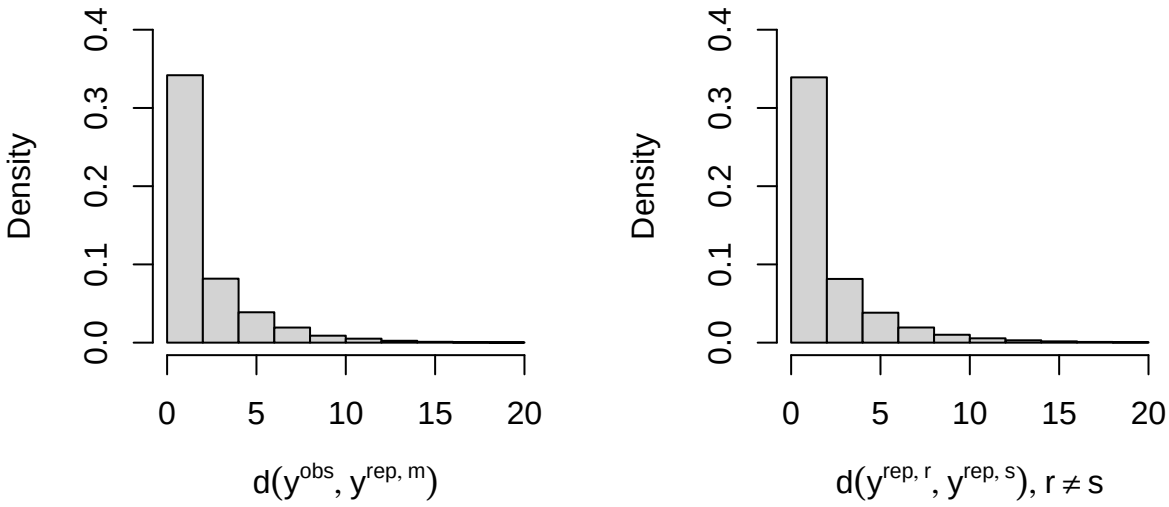


Figure 3.16: Histograms showing distribution of distances between $\mathbf{y}^{obs}$ and $\mathbf{y}^{rep,m}$

Visually, these histograms are clearly similar. The next step is to conduct a Mann-Whitney U Test on these two vectors of distances. For this purpose, we will use the wilcox.test() function in R. In this case, we would use

wilcox.test(d_obs, d_rep, alternative = captionpos"captionposgreatercaptionpos")

which returns a p-value extremely close to 1, signaling that there is not enough evidence to suggest $d(\mathbf{y}^{obs}, \mathbf{y}^{rep,m})$ is greater than $d(\mathbf{y}^{rep,r}, \mathbf{y}^{rep,s}), r \neq s$. One negative consequence of using a traditional p-value hypothesis test is that for very large population sizes, we will reject our null hypothesis that the stochasticity between these sets of distances is equal, even when the sets are very similar to each other. We will ignore this issue for now.

To show the average behavior of this method, we conduct a simulation study. To begin, we can look at the behavior of this method using the same example as above, but comparing three different models: the correct model ($\sigma = 1$), an incorrect model ($\sigma = 1, \mu = 1$), and an even more incorrect model ($\sigma = 1, \mu = 2$).

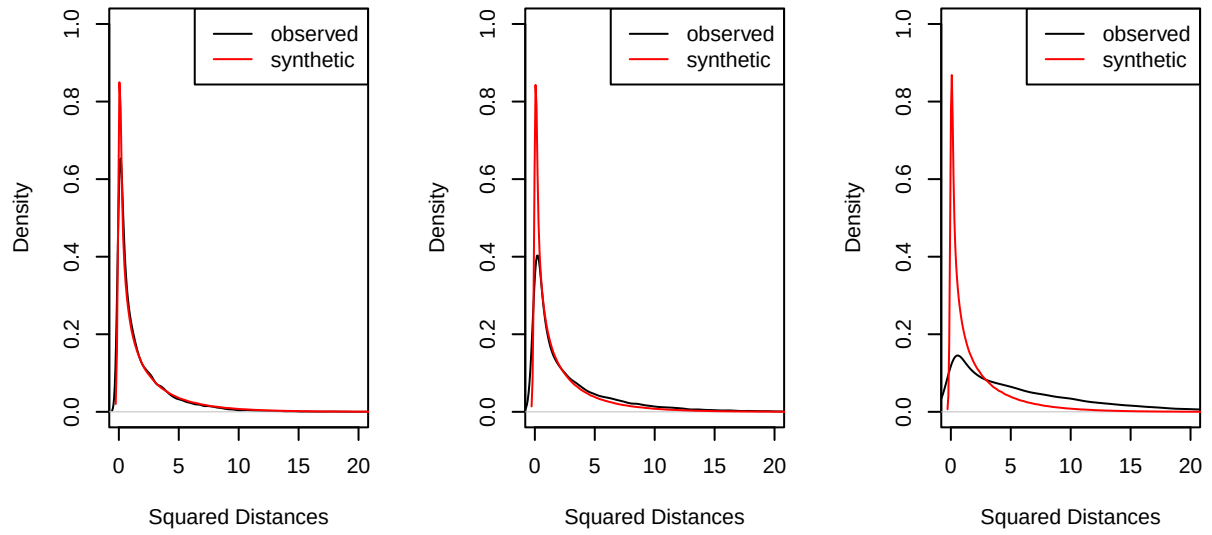


Figure 3.17: Comparison of distances between $\mathbf{y}^{obs}$ and $\mathbf{y}^{rep,m}$ for 3 models: true model (left), incorrect model (middle), and an even worse model (right).

From Figure 3.17, we can see that the correct model yields distances that are much more similar than the two incorrect model specifications. To show the average behavior over multiple repetitions, we perform this entire process 20 times with different observed samples, and compare the distribution of the p-values.

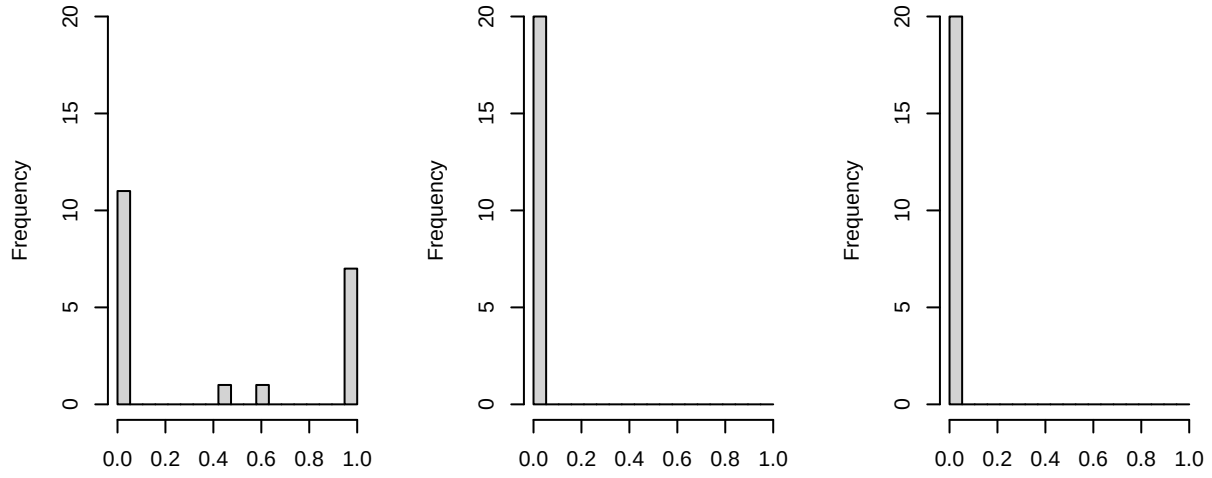


Figure 3.18: Comparison of p-values for testing whether the distribution of distances indicates poor model fit for: correct model (left), incorrect model (middle), and an even worse model (right)

Figure 3.18 shows the p-values from 20 iterations of this simulation. While it is clear that

hypothesis testing correctly rejects the incorrect models, we see that the correct model is often being rejected as well. As mentioned before, this is caused by using a traditional p-value; even when sets of distances are *extremely* close to each other, a large enough N can essentially guarantee that the hypothesis will be rejected anyway. To illustrate, consider Figure 3.19, which shows the density of these distances for one of the simulations where the correct model was rejected, and one where an incorrect model was rejected. Note that the densities are virtually identical for the correct model, but the hypothesis was still rejected.

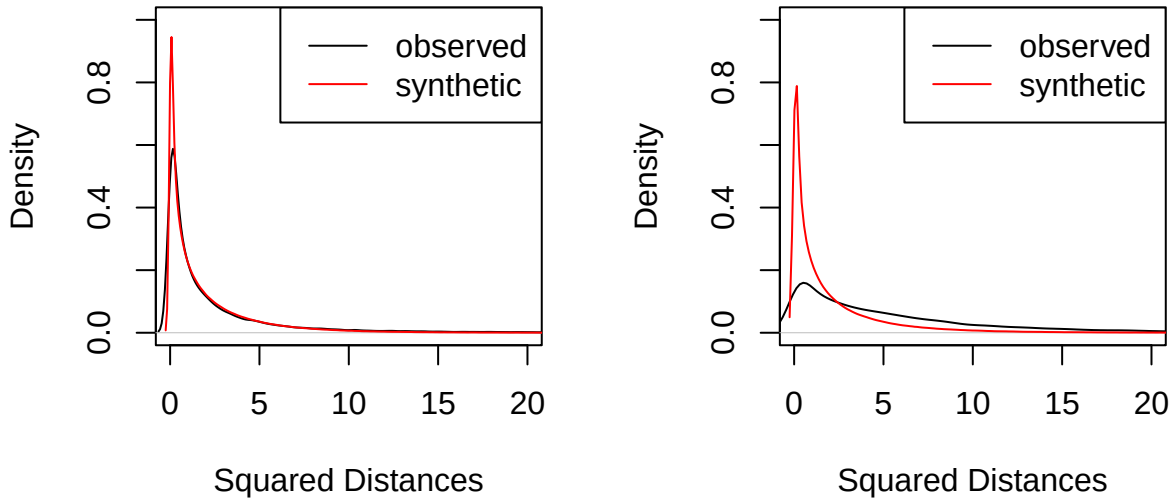


Figure 3.19: Density comparisons for distances between observed and synthetic $\mathbf{y}$ for a false positive (left) and a true positive (right).

Considering this, our best option may be a visual inspection of the density or histogram of distances, instead of an actual p-value from a test such as the Mann-Whitney U Test.

### 3.3.3 Examples

In this section, we will discuss two examples using the Dirichlet spacing prior:

1. A very basic example using a uniform base distribution with $n = 2, N = 10$
2. A slightly more complicated example using a normal base distribution with $n = 20, N = 100$

**Example 1: Uniform base distribution**

Consider the following example; we observe a sample $x = \{0.3, 0.7\}$ from a Unif$(0, 1)$ population of size $N = 10$. With such a simple example, we can use a basic uniform proposal in our MCMC to sample from the posterior of the population given a pre-specified value of $N_{\text{eff}}$.

For example, if we consider $N_{\text{eff}} = \{1, 10, 100\}$, we can visualize the effect $N_{\text{eff}}$ has on the resulting synthetic populations.

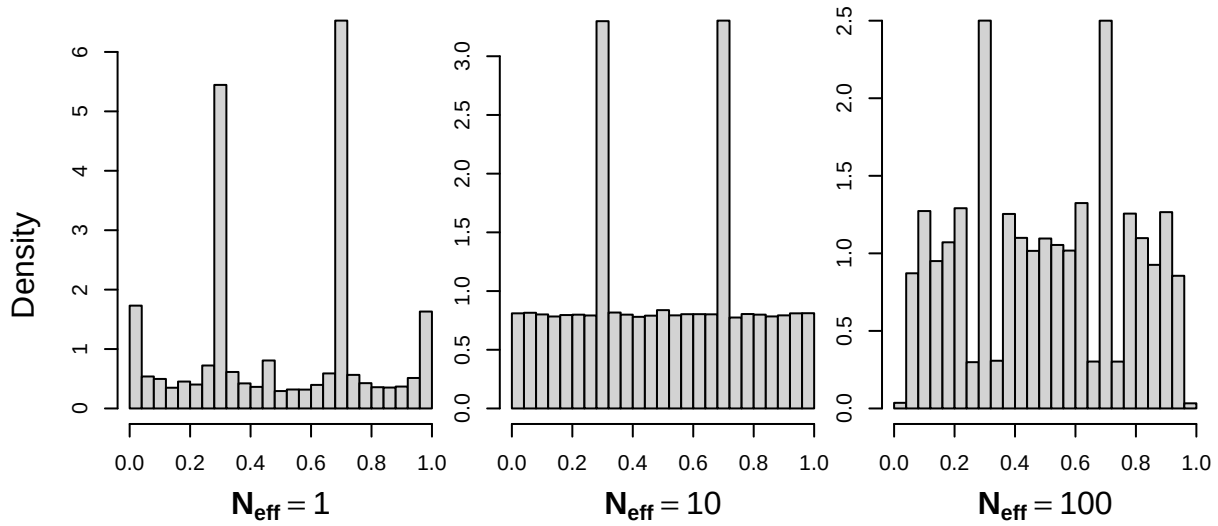


Figure 3.20: Histograms showing average behavior of $\mathbf{y}$ with $N = 10$ for $N_{eff} = 1$ (left), 10 (center), and 100 (right).

Figure 3.20 shows histograms of 10000 populations of size $N = 10$ synthesized using three different values of $N_{\text{eff}}$. There are important things to note for each histogram. When $N_{\text{eff}}$ is small (significantly less than $N$), clusters tend to form on the boundaries of the space (around 0 and 1), in addition to the observed values of $x = \{0.3, 0.7\}$. When $N_{\text{eff}} = N$, the resulting histogram looks uniform except for the observed $x = \{0.3, 0.7\}$; in fact, if those observations are omitted from the histogram, you end up with a histogram that looks nearly flat. This reinforces the fact that $N_{\text{eff}} = N$ results in i.i.d. sampling; even when there are

observed values $x$, the resulting populations are formed via i.i.d. sampling, meaning they just look like samples from the base distribution (in this case, uniform). When $N_{\text{eff}}$ is large (significantly greater than $N$), the resulting synthetic populations should be evenly spaced along the space from 0 to 1. However, since we have observed values at $x = \{0.3, 0.7\}$, it is not possible for the populations to be truly evenly spaced. This high value of $N_{\text{eff}}$ would prefer to put population members at $y = \{1/11, 2/11, ..., 10/11\}$ to achieve the most evenly spaced populations. However, since $x = \{0.3, 0.7\}$ do not coincide with these values, the other population members are forced into nearly symmetric patterns (symmetric because $x = \{0.3, 0.7\}$ is symmetric inside of $[0, 1]$).

In addition to looking at histograms of many synthetic populations, we can investigate the number of synthetic population members in each of the bins defined by $[0, 0.3), (0.3, 0.7), (0.7, 1]$. Figures 3.21, 3.22, and 3.23 show this for the three values of $N_{\text{eff}}$ considered. Here we can see that the variance of the membership in these gaps increases as $N_{\text{eff}}$ decreases.

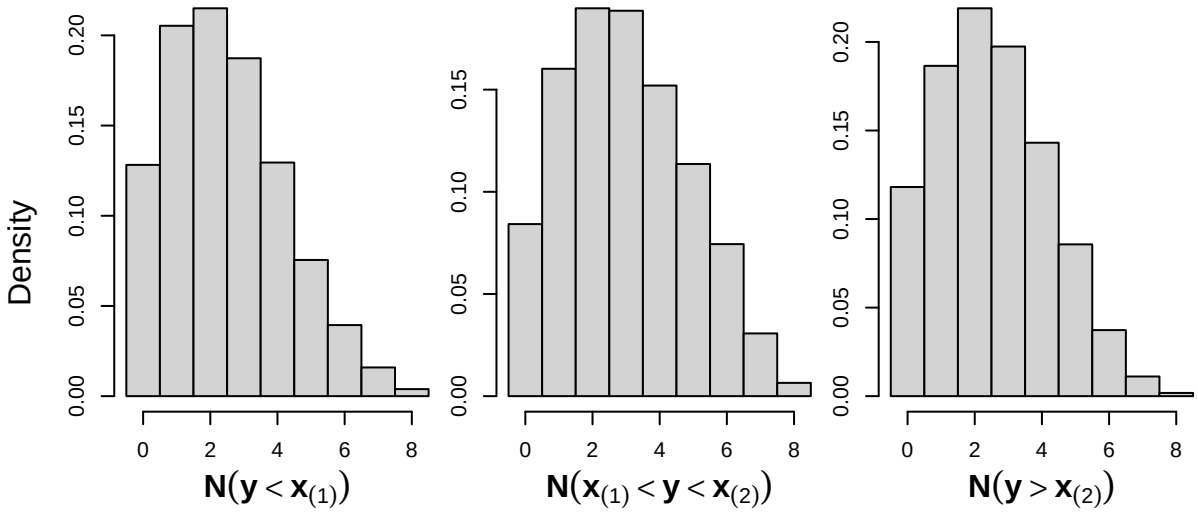


Figure 3.21: Histograms showing membership inside each gap defined by $\mathbf{x}$ for $N_{eff} = 1$.

Finally, we can look at the resulting mean of the populations. Figure 3.24 shows the resulting

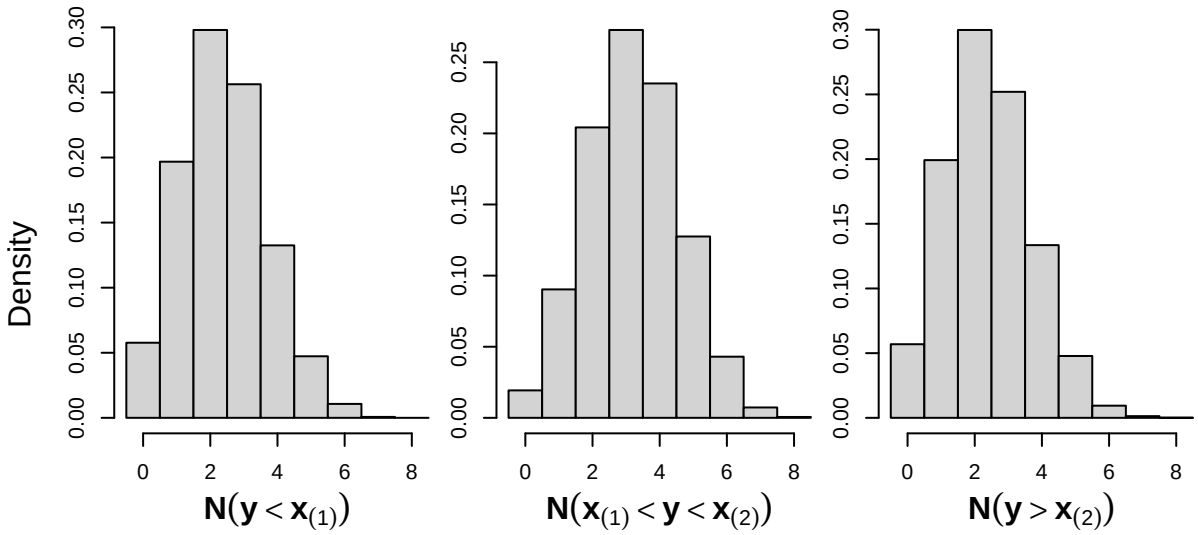


Figure 3.22: Histograms showing membership inside each gap defined by $\mathbf{x}$ for $N_{eff} = N = 10$.

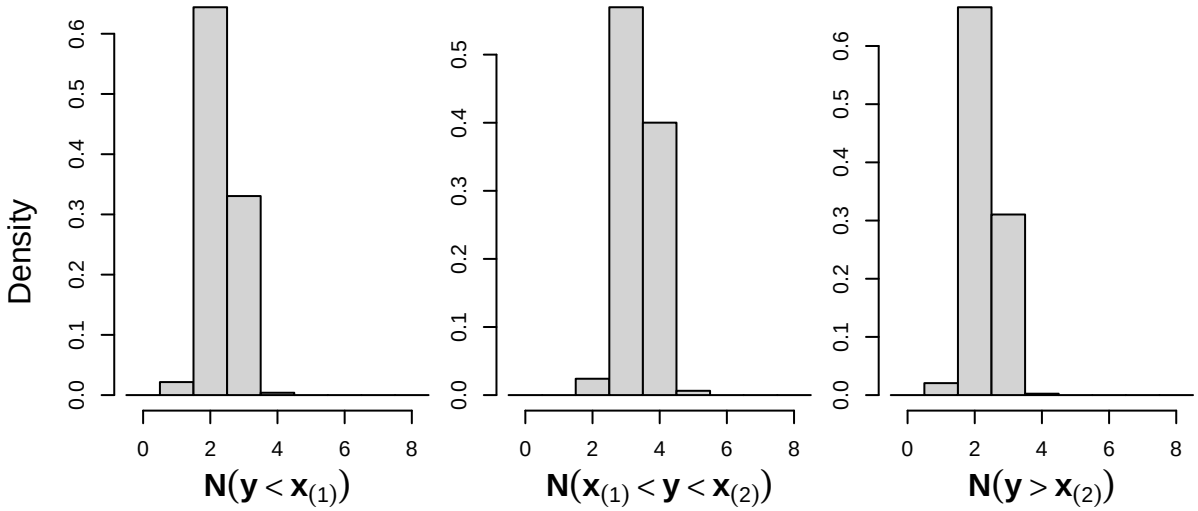


Figure 3.23: Histograms showing membership inside each gap defined by $\mathbf{x}$ for $N_{eff} = 100$.

population means for 10000 synthetic populations with each of the three values of $N_{\text{eff}}$. Just like before, we can see effect of increasing or decreasing this parameter. The resulting mean of the population means will not change; however, the variance of the population mean greatly depends on $N_{\text{eff}}$.

With this simple example out of the way, we can move onto a slightly more complicated

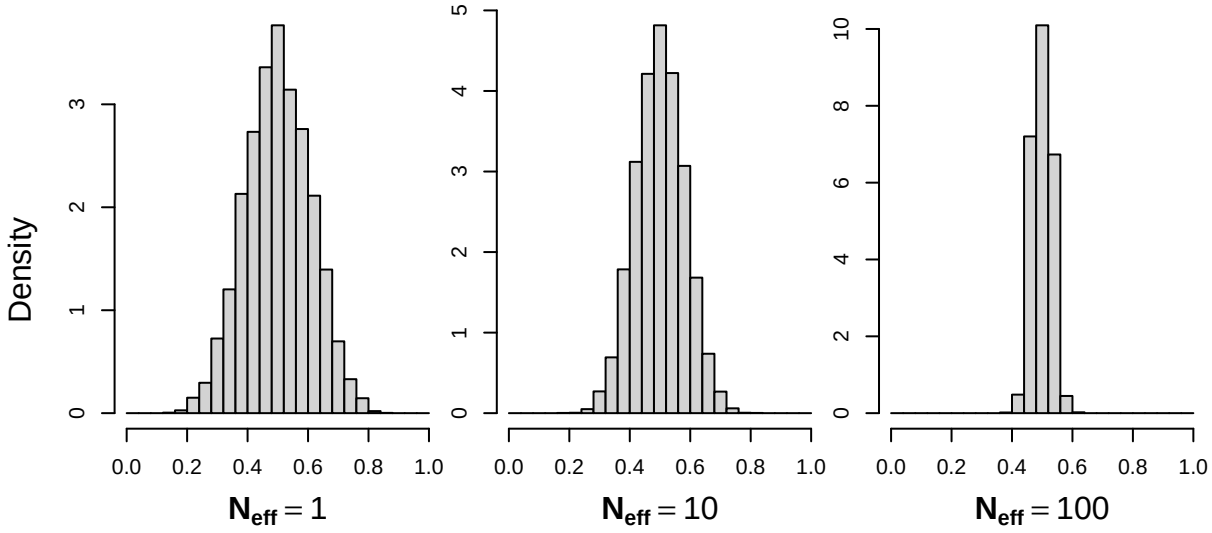


Figure 3.24: Histograms showing $\mu(\mathbf{y})$ with $N = 100$ for $N_{eff} = 1$ (left), 10 (center), and 100 (right).

scenario.

**Example 2: Normal base distribution**

In our second example, suppose we observe $\mathbf{x}$ of size $n = 20$, represented by the histogram in Figure 3.25.

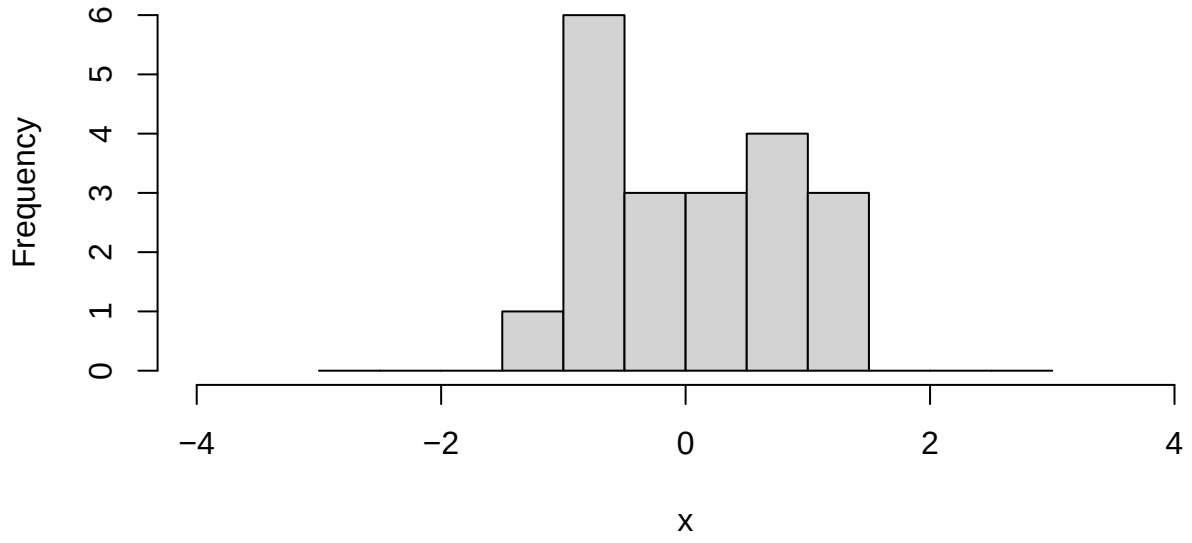


Figure 3.25: Histogram showing sample $\mathbf{x}$ from a supposed Normal$(0, 1)$ distribution.

It is believed that $\mathbf{x}$ comes from a standard Normal distribution, so we can use that as our base distribution $F$. Using this base distribution, we can synthesize several populations of size $N = 100$ using different values of $N_{\text{eff}}$. In this case, we will consider $N_{\text{eff}} = 20, 100, 500$. Figure 3.26 shows the resulting histograms from combining 10000 synthetic populations for each value of $N_{\text{eff}}$. While all of these histograms closely resemble the standard Normal distribution, it is clear that the sample has an effect, especially for $N_{\text{eff}} = 20$.

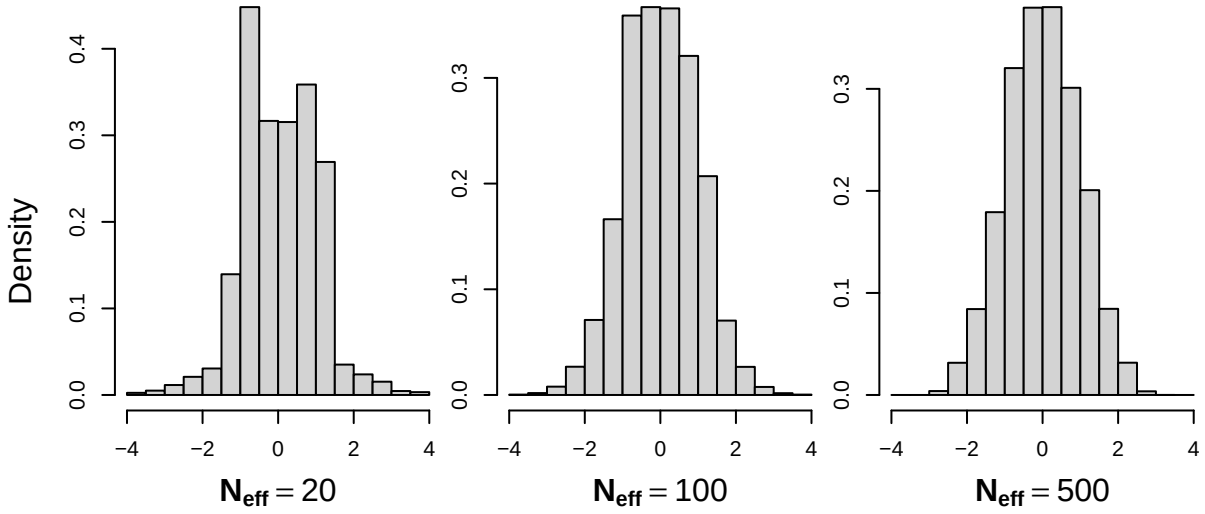


Figure 3.26: Histograms showing $\mathbf{y}$ for $N_{eff} = 20$ (left), 100 (center), 500 (right).

While it may not be clear from 3.26, increasing and decreasing $N_{\text{eff}}$ has the same effect on the populations that it had in the previous example; specifically, lowering $N_{\text{eff}}$ allows the populations to deviate more from the base distribution via clustering. To show this, Figure 3.27 shows Empirical CDFs from 50 sampled populations (here we look at the $\mathbf{u}$ values instead of the $\mathbf{y}$ values themselves) for each value of $N_{\text{eff}}$. Note that $N_{\text{eff}} = 20$ allows for significant deviations from a straight line, while $N_{\text{eff}} = 500$ results in very little deviation.

Finally, we can plot $\mu(\mathbf{y})$ like we did for the previous example. Figure 3.28 shows the distribution of these means, where we can clearly see that increasing $N_{\text{eff}}$ results in much less deviation from the base distribution's mean of 0.

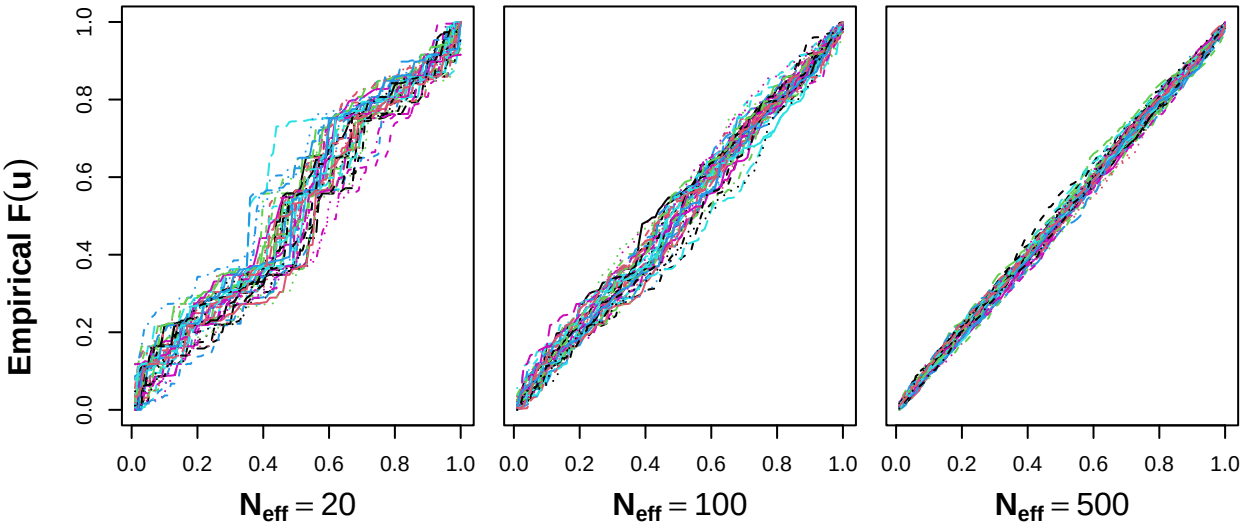


Figure 3.27: 50 Empirical CDFs of $\mathbf{y}$ for $N_{eff} = 20$ (left), 100 (center), 500 (right).

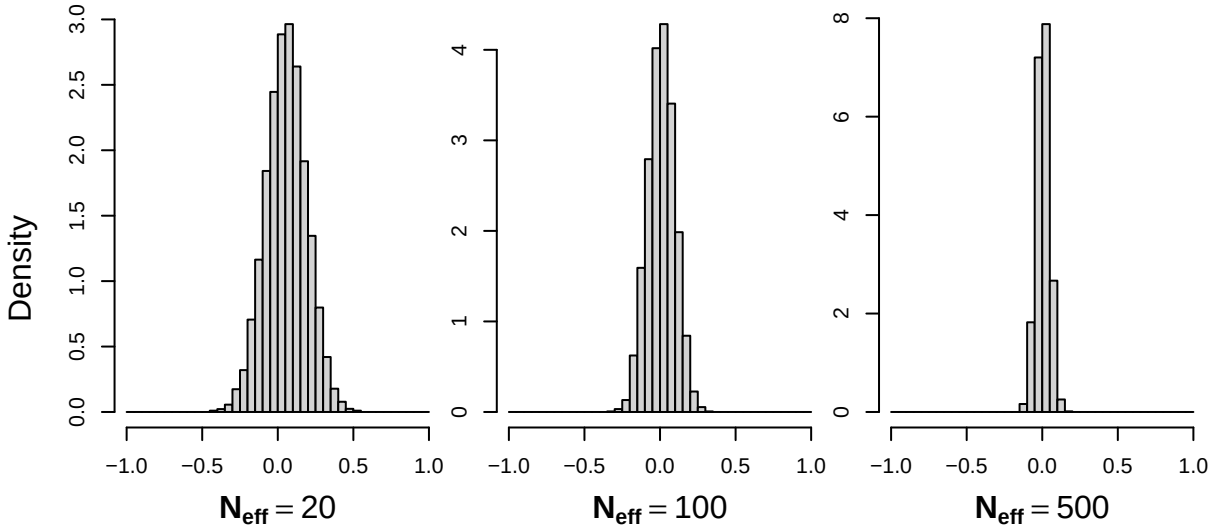


Figure 3.28: Histograms showing $\mu(\mathbf{y})$ for $N_{eff} = 20$ (left), 100 (center), 500 (right).

For the full Dirichlet Spacing prior that we implemented in these two examples, we are limited to fake examples or examples with a small sample and population size. Sampling from the full Dirichlet Spacing prior with a large sample and population size is computationally intensive; thus, we will instead turn to an approximation, discussed in the next section.

## 3.4 Binned Dirichlet Spacing Prior

In Section 3.3, we mentioned that the full formulation of the Dirichlet Spacing prior could run into issues with large populations. One way around that problem is explored within this section. To begin with, the density of the full Dirichlet Spacing prior can be expressed as

$$f(\mathbf{y}|G_\theta, \alpha) \propto \prod_{i=1}^{N+1} \left[G_\theta(y_{(i)}) - G_\theta(y_{(i-1)})\right]^{\alpha-1} \times \prod_{j=1}^{N} g_\theta(y_j), \tag{3.4}$$

assuming we know base distribution parameters $\theta$. One property of the Dirichlet distribution which we can use to our advantage is that the conditional distribution of observations is also Dirichlet.

*Proof.* Suppose $\mathbf{x} = (x_1, x_2, \dots, x_k)$ follows a Dirichlet distribution with parameter $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_k)$. WLOG, let $\mathbf{x}_{(1)} = (x_1, x_2, \dots, x_a)$ and $\mathbf{x}_{(2)} = (x_{a+1}, x_{a+2}, \dots, x_k)$. Let $\alpha_0 = \sum_{i=a+1}^{k} \alpha_i$. The joint density of $\mathbf{x} = (x_1, x_2, \dots, x_k)$ is

$$f_X(x_1, x_2, \dots, x_k) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \prod_{i=1}^{k} {x_i}^{\alpha_i - 1}$$

with $x_i \in (0, 1)$ and $\sum_{i=1}^{k} x_i = 1$, where $\Gamma(\cdot)$ is the Gamma function. Similarly, the joint density of $\mathbf{x}_{(1)} = (x_1, x_2, \dots, x_a)$ is

$$f_{X_{(1)}}(x_1, x_2, \dots, x_a) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{\Gamma(\alpha_0) \prod_{i=1}^{a} \Gamma(\alpha_i)} \prod_{i=1}^{a} {x_i}^{\alpha_i - 1} \Big(1 - \sum_{j=1}^{a} x_j\Big)^{\alpha_0 - 1}.$$

Taking the ratio, we get

$$\begin{aligned} f_{\mathbf{X}_{(2)}|\mathbf{X}_{(1)}}(x_{(2)}|x_{(1)}) &= \frac{\Gamma(\alpha_0)\prod_{i=1}^{a}\Gamma(\alpha_i)}{\prod_{i=1}^{k}\Gamma(\alpha_i)} \frac{\prod_{i=1}^{k} {x_i}^{\alpha_i-1}}{\prod_{i=1}^{a} {x_i}^{\alpha_i-1}\Big(1-\sum_{j=1}^{a}x_j\Big)^{\alpha_0-1}} \\ &= \frac{\Gamma(\sum_{i=a+1}^{k}\alpha_i)}{\prod_{i=a+1}^{k}\Gamma(\alpha_i)} \prod_{i=a+1}^{k} {x_i}^{\alpha_i-1}\Big(1-\sum_{j=1}^{a}x_j\Big)^{-(\alpha_0-1)} \\ &= \frac{\Gamma(\sum_{i=a+1}^{k}\alpha_i)}{\prod_{i=a+1}^{k}\Gamma(\alpha_i)} \prod_{i=a+1}^{k} \left[x_i\Big(1-\sum_{j=1}^{a}x_j\Big)^{-1}\right]^{\alpha_i-1} \Big(1-\sum_{j=1}^{a}x_j\Big)^{-[(k-a)-1]} \end{aligned}$$

□

Thus, $\mathbf{x}_{(2)}|\mathbf{x}_{(1)}$ follows a scaled (or truncated) Dirichlet distribution, where the scaling parameter is $1-\mathbf{1}^T x_{(1)}$. This means that the values for $p$ values that form our cumulative sums and define the population $\mathbf{y}$ are still distributed Dirichlet, but instead of summing to 1, they instead sum to the appropriate value (e.g., the set of $p_i, \dots, p_j$ that fall between $x_{(1)} = 0.2$ and $x_{(2)} = 0.35$ sum to $x_{(2)} - x_{(1)} = 0.15$). This lets us break down the full Dirichlet Spacing prior into pieces; for example we can break the support of our base distribution $G$ into bins $\mathbf{B} = \{B_1, B_2, \dots, B_c\}$, where the bin boundaries are $\mathbf{b} = \{b_0, b_1, \dots, b_c\}$ (let $b_0 = -\infty$ and $b_c = \infty$ so the whole support is covered). Inside of each bin, we can sample from a Dirichlet distribution to get our values for $p$ that define the spacings between observations.

There is one problem with this however: we do not know how many observations are inside of each bin. To answer this question, we need to return to the full prior, but think about it in terms of $\mathbf{N} = \{N_1, N_2, \dots, N_c\}$, the number of elements within each bin, instead of considering the full prior on $\mathbf{y}$. When we only care about $\mathbf{N}$, the prior on $\mathbf{N}$ simplifies to

$$f(N_1, N_2, \dots, N_c|G_\theta, \alpha) \propto \prod_{i=1}^{c} \left[G_\theta(b_i) - G_\theta(b_{i-1})\right]^{N_i\alpha-1} \times \prod_{j=1}^{c-1} g_\theta(b_j), \tag{3.5}$$

where $\mathbf{b} = \{b_0, b_1, \ldots, b_c\}$ is as described above.

The following pseudocode will sample from (3.5), the prior for $\mathbf{N}$, when $\alpha$ is known and we do not need realizations of $\mathbf{y}$:

**INPUT**: $\alpha$ - concentration parameter

$\mathbf{B}$ - bins

$G$ - base distribution

$N$ - population size

$T$ - number of iterations

**OUTPUT**: $T$ realizations of $\mathbf{N}$

Initialize $\mathbf{N}$

**FOR** $i$ **IN** $1, \ldots, T$

Propose $\mathbf{N}^*$ from $p(\mathbf{N}^*|\mathbf{N})$

Calculate $a = \min\left(1, \frac{f(\mathbf{N}^*|G,\alpha)}{f(\mathbf{N}|G,\alpha)} \times \frac{p(\mathbf{N}|\mathbf{N}^*)}{p(\mathbf{N}^*|\mathbf{N})}\right)$

Set $\mathbf{N} = \mathbf{N}^*$ with probability $a$

**SAVE N**

**RETURN** $T$ realizations of $\mathbf{N}$

where $b_0, b_1, \ldots, b_c$ are the bin boundaries as described above. Since we have the posterior for $\mathbf{N}$, we could model the elements within each bin using the scaled Dirichlet distribution and original base distribution to recover the full Dirichlet Spacing prior (if we let a sample $\mathbf{x}$ define the bins). Alternatively, if the bins are small enough that exact values are not important, we can use an arbitrary set of bins $\mathbf{B}$ and employ a much simpler distribution (e.g., continuous uniform) inside of the bins.

Regardless of our choice, the joint prior for $\mathbf{y}, \mathbf{N}$ takes the form

$$\pi(\mathbf{y}, \mathbf{N}|G_\theta, \alpha) \propto f(\mathbf{y}|\mathbf{N}) \times \text{BD-Sp}(\mathbf{N}|G_\theta, \alpha), \tag{3.6}$$

where BD-Sp$(\mathbf{N}|G_\theta, \alpha)$ is the Binned Dirichlet Spacing prior, expressed in (3.5), and $f$ is our choice of distribution for $\mathbf{y}$ inside of the bins. Of course, it is up to us whether parameters $\theta$ of the base distribution are fixed or modeled with priors themselves, and the same applies to $\alpha$. It is important to realize that this joint prior for $\mathbf{y}, \mathbf{N}$ does not get *updated* by our sample in the same way that the full Dirichlet Spacing prior does, so learning parameters such as $\theta$ or $\alpha$ is impossible without an additional likelihood on the data. We will build off the joint prior above (3.6) throughout the remainder of this chapter, as well as Chapters 4 and 5. However, we first look at whether the choice of distribution for $f(\mathbf{y}|\mathbf{N})$ is important, and how to include a hyperprior for $\alpha$.

### 3.4.1 Choice of Bin Distribution

When we introduced the Binned Dirichlet Spacing prior, we mentioned that one option was use the scaled Dirichlet distribution inside of the bins to form the full Dirichlet Spacing prior. However, that is certainly not the only option; two other likely candidates are a continuous uniform distribution and the base distribution used inside of the BD-Sp prior. To choose, we recommend assessing sensitivity with applications. For example, in this section we compare the resulting distribution of $\mathbf{y}$ using all three options mentioned above to see if this choice has a serious effect on the resulting distribution of $\mathbf{y}$.

For our comparison, we use the Blacksburg PUMS data (specifically incomes). To assist in our sensitivity analysis, we would like to compare outcomes to those using the full Dirichlet Spacing prior. For this, we cannot use the full sample ($n > 2000$) because of computational

reasons; instead, we will use a subset ($n = 100$) of the sample data. As mentioned in the previous section, the BD-Sp prior for $\mathbf{y}, \mathbf{N}$ does not get updated by observing a sample. Because of this, we will use a Multinomial likelihood on the sample data of the form

$$f(\mathbf{x}|\mathbf{y}) \propto \text{Multi}(\mathbf{k}|n, \vec{\rho}),$$

where $\mathbf{k} = \{\sum_{i=1}^{n} 1_{\{x_i \in B_1\}}, \ldots \sum_{i=1}^{n} 1_{\{x_i \in B_c\}}\}$ are the sample counts in each bin, $n$ is the sample size, and $\vec{\rho}$ is the population proportion within each bin (e.g., $\rho_1 = \frac{1}{N} \sum_{i=1}^{N} 1_{\{y_i \in B_1\}}$). For the BD-Sp prior (3.5), we let $\mathbf{B} = \{(-\infty, 5000), [5000, 10000), \ldots, [195000, 200000), [200000, \infty)\}$. In turn, we will synthesize populations of size $N = 500$, with $\alpha = 0.4$. The base distribution is Gamma($k = 1.299, \theta = 54688$), with parameters estimated from the national distribution of incomes.

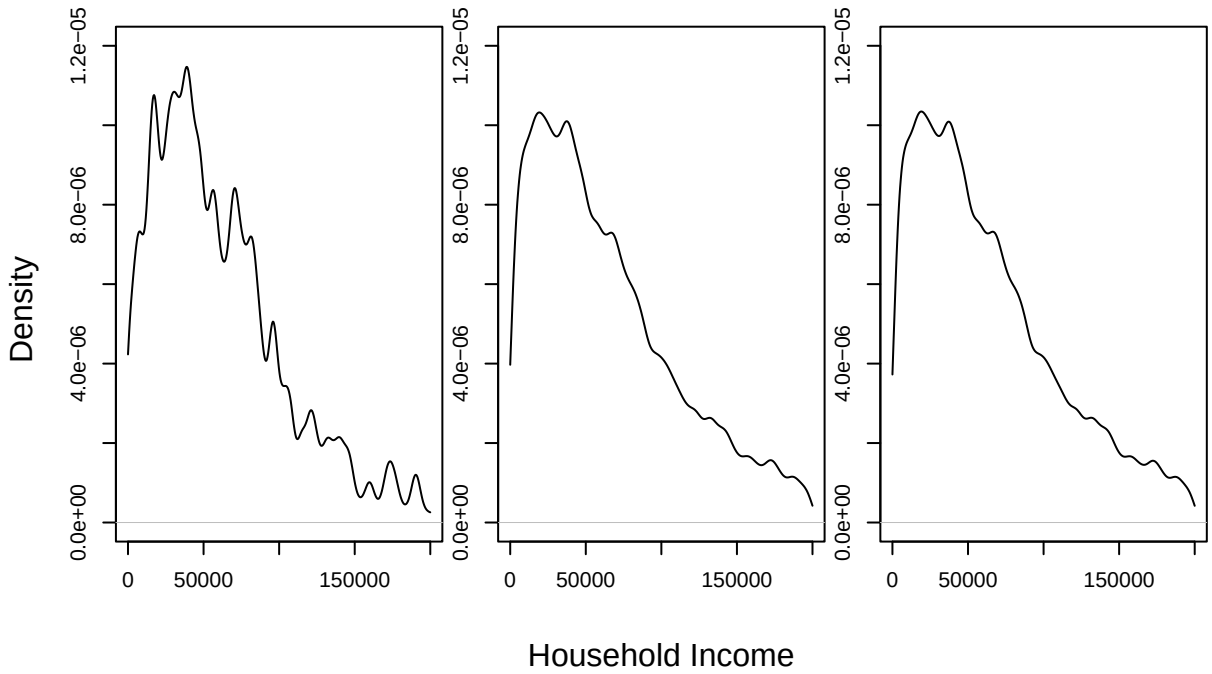


Figure 3.29: Synthetic income densities for three methods: full Dirichlet-Sp (left), uniform within gaps (middle), and base distribution within gaps (right).

Figure 3.29 shows the densities of $\mathbf{y}$ created by sampling from the full Dirichlet Spacing

prior (left), BD-Sp with uniform within bins (middle) and BD-Sp with base distribution within bins (right). Here we see that the full Dirichlet Spacing prior is capturing more local variability in the density of incomes than the binned methods. This difference is due to how a sample is considered in the population. The full Dirichlet Spacing prior ensures that observed sample values are a subset of each resulting population, thus the population density will be jagged around sampled values. The populations created with the Binned Dirichlet Spacing prior do not contain the observed sample, and thus the resulting populations are much smoother. If we enforce that the observed sample are a part of each population with the binned methods, we get the densities found below in Figure 3.30.

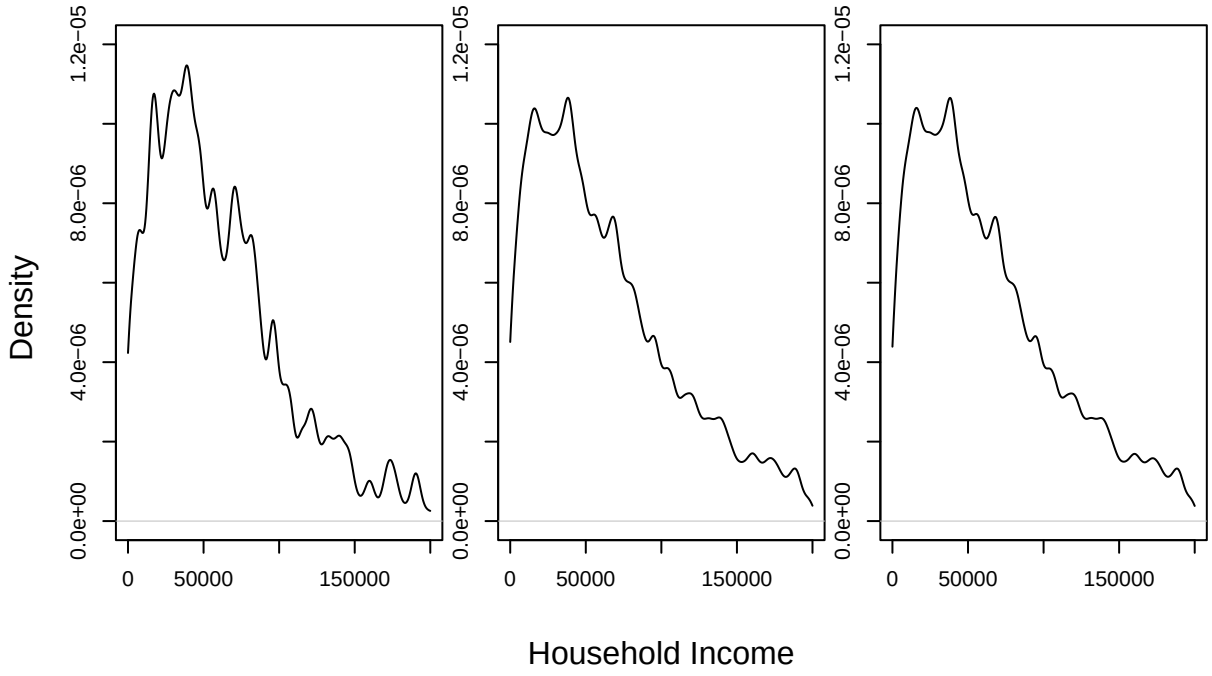


Figure 3.30: Synthetic income densities for three methods: full Dirichlet-Sp (left), uniform within gaps (middle), and base distribution within gaps (right), where the gap-based methods include the random sample.

From Figure 3.30, we see that some of the local behavior is achieved, but the density is still smoother than the full Dirichlet Spacing prior. Given the computational complexity of the full Dirichlet Spacing prior, it may often be more desirable (or required) to fit a binned

method. Also, we can see that there is very little difference in the end result when we use a uniform distribution in the bins, compared to the base distribution.

For one final comparison, we use the method outlined in Section 3.3.2 to see if there is any difference in model fit between the full Dirichlet Spacing prior and the binned methods. The results are shown in 3.31.

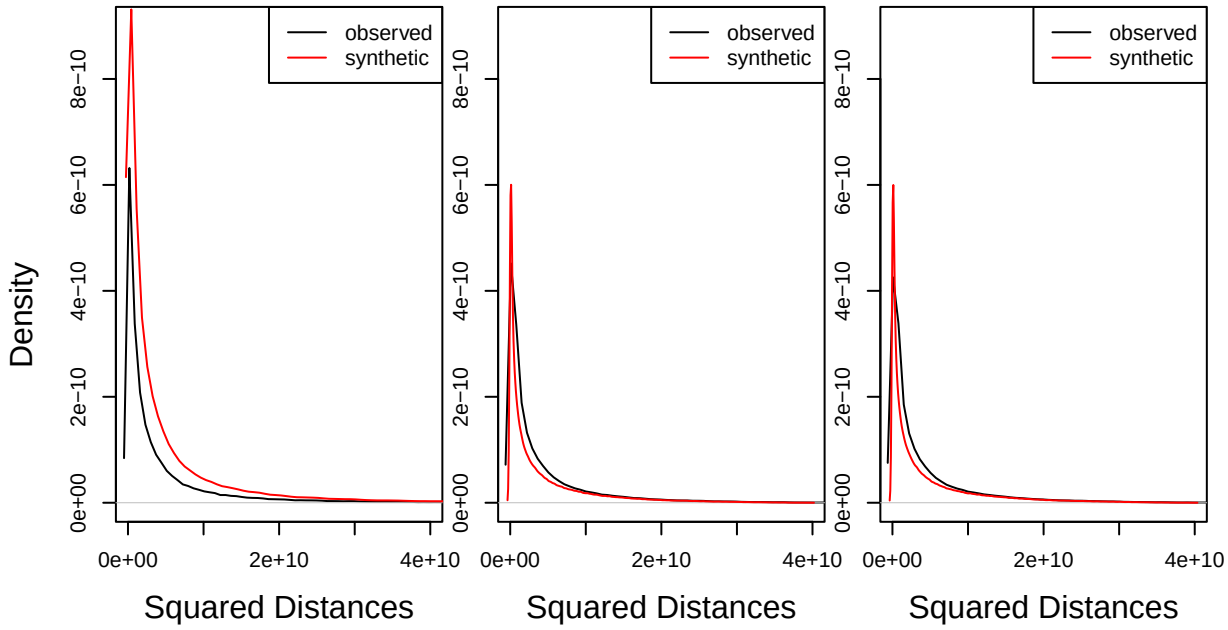


Figure 3.31: Comparison of distances between $\mathbf{y}^{obs}$ and $\mathbf{y}^{rep,m}$ for 3 models: full Dirichlet Spacing prior (left), uniform within gaps (middle), and base distribution within gaps (right).

Here we are not looking for a *true* model (there is none, since we allowed $\alpha$ to be fixed at a constant which we know is not correct), instead we are mostly looking to see if there are major differences within the binned methods, and between the binned methods and the full Dirichlet Spacing prior. Visually, the choice of distribution within bins does not seem to make a difference; however, there does appear to be a significant difference between the binned methods and the full Dirichlet Spacing prior.

As mentioned above, we chose a value for $\alpha$ instead of attempting to learn plausible values

of the parameter. In the next section we will explore placing a hyperprior on $\alpha$.

### 3.4.2 Estimating $\alpha$

Previously we mentioned the possibility of including a hyperprior on $\alpha$. Within this section, we formalize the posterior for $\mathbf{y}, \mathbf{N}, \alpha$ when we do so, and show results from simulation studies where we know the true $\alpha$ and try to learn it via MCMC.

If we append a hyperprior on $\alpha$ to our binned method, and also include a Multinomial likelihood from $\mathbf{x}$, we get

$$\pi(\mathbf{y}, \mathbf{N}, \alpha | G_\theta, \mathbf{x}) \propto \text{Multi}(\mathbf{k}|n, \vec{\rho}) \times f(\mathbf{y}|\mathbf{N}) \times \text{BD-Sp}(\mathbf{N}|G_\theta, \alpha) \times \pi(\alpha),$$

where $\pi(\alpha)$ is our hyperprior on $\alpha$. For our purposes, we mainly expect $\alpha \in (0, 1)$; while it can theoretically take on values higher than 1, it would not make sense in our situation since that would imply that populations are *less* clustered (more *regularly* spaced) than i.i.d. sampling.

To evaluate the performance of this setup, we conduct a simulation study using a Gamma($k = 1.299, \theta = 54688$) base distribution, and a Unif$(0.1, 1)$ hyperprior on $\alpha$. We create populations $\mathbf{y}$ of size $N = 10000$ with true $\alpha = \frac{N_{\text{eff}}+1}{N+1} \in (0.2, 0.9)$, and sample $\mathbf{x}$ of size $n = 2000$ from $\mathbf{y}$. We then tally the sample members within bins defined by $-\infty, 5000, \ldots, 200000, \infty$, and construct equal-tailed credible intervals of $\alpha$, checking whether the true $\alpha$ lives in this interval.

Figure 3.32 shows the results of this simulation study. Here we see the effectiveness of estimating $\alpha = \frac{N_{\text{eff}}+1}{N+1}$ via posterior medians. Our overall coverage rate (from 1000 simulations) for our 90% equal-tailed credible intervals was 89.8%.

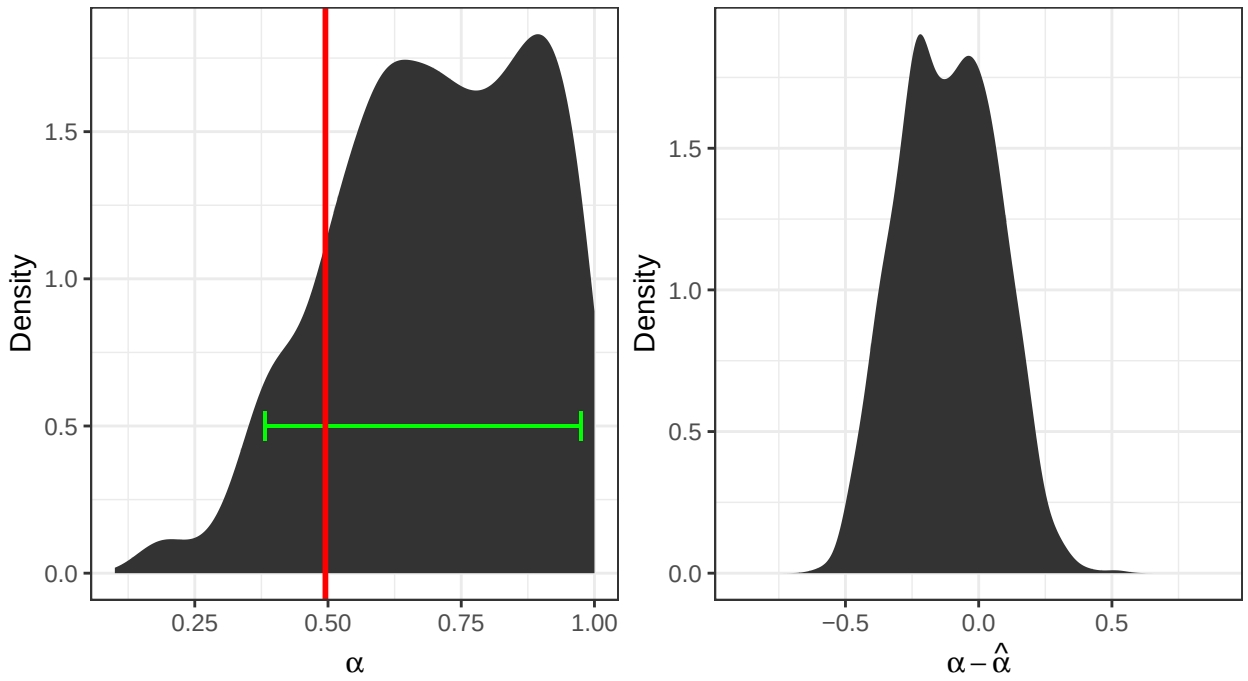


Figure 3.32: An example posterior for $\alpha$ (left), showing equal-tailed confidence interval (green) and the true $\alpha$ (red); the density of $\alpha - \hat{\alpha}$ (right).

### 3.4.3 Blacksburg Application

In Section 3.3, we limited our scope to mock examples because the full Dirichlet Spacing prior is too computationally intensive for real-world examples. The Binned Dirichlet Spacing prior however is not. In this section, we model the population of Blacksburg household incomes using a Binned Dirichlet Spacing prior on the population.

In our introduction (see 1.3), we briefly covered some data sources available to us for Blacksburg. Here, we specifically look at the household incomes from the American Community Survey's PUMS (US Census Bureau 2019c), which provides income data for a sample from the PUMA containing Blacksburg. For now, we assume this sample is a simple random sample (SRS) from Blacksburg. The only other piece of information needed is the population size ($N = 8127$), which in this case is taken from the Montgomery County parcel and tax records (Virginia Tech Library Maps & GIS Division 2019).

To make use of the Binned Dirichlet Spacing prior, we also need to specify a base distribution. To model Blacksburg incomes, we will use a Gamma distribution with parameters estimated using the national distribution of household incomes, from the 2019 ACS 5-Year estimates (US Census Bureau 2019a). Specifically, we pick values for the parameters so that the mass within each income bin is as close as possible to the national distribution. Figure 3.33 overlays the national distribution of incomes with the estimated Gamma distribution which we will use as our base distribution.

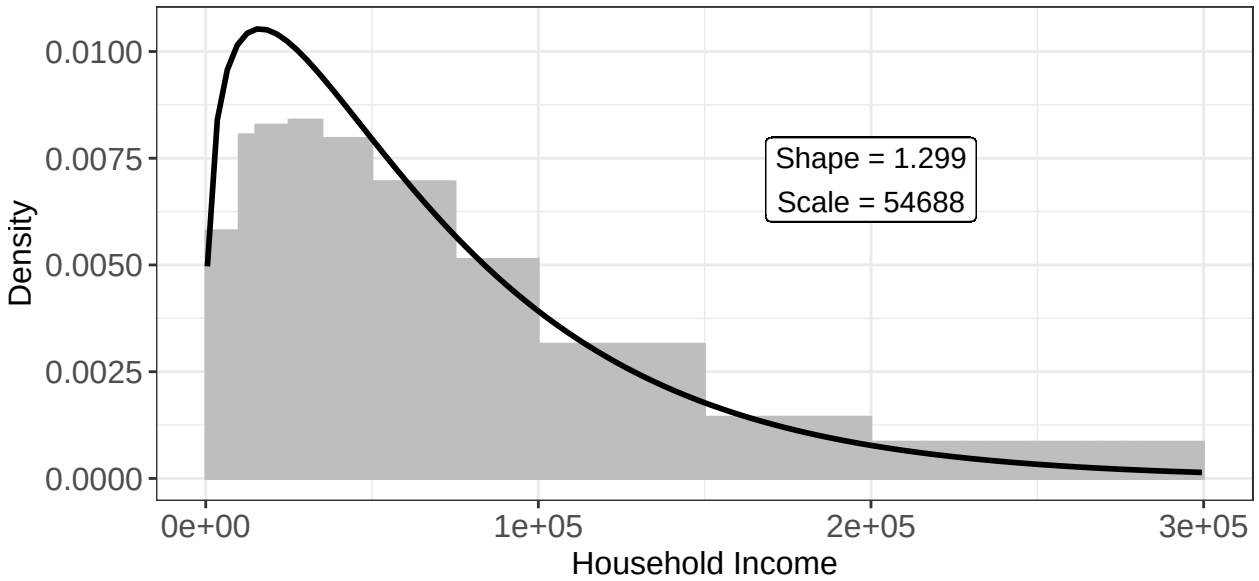


Figure 3.33: Histogram showing distribution of national household incomes (from binned data) and corresponding Gamma distribution parameter estimates.

If we compare the resulting Gamma distribution that we intend to use as our base distribution with the distribution of household incomes within our PUMS data, we see that the distributions are not overly dissimilar (see Figure 3.34). Given the nature of the Dirichlet Spacing prior that allows for significant deviations from the base distribution, the dissimilarity should not be an issue for generating reasonable synthetic populations.

The full posterior for $\mathbf{N}, \alpha$ which sample from to create our synthetic incomes (or in this

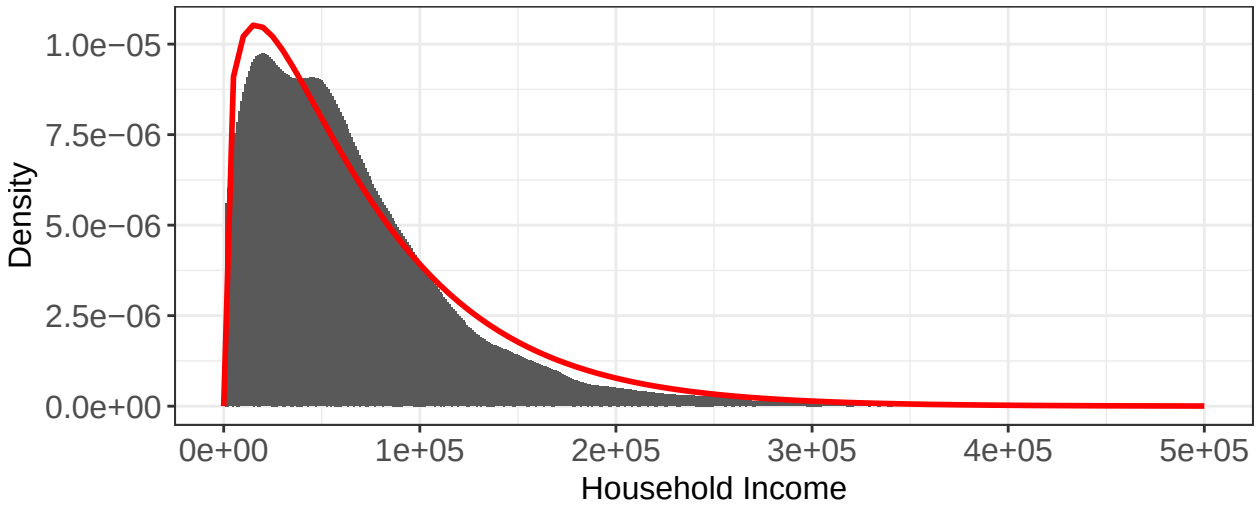


Figure 3.34: Histogram showing distribution of household incomes from PUMS with Gamma distribution from national distribution overlaid in red.

case, binned incomes) is

$$f(\mathbf{N}, \alpha|\mathbf{x}) \propto \text{Multi}(\mathbf{k}|n, \vec{\rho}) \times \text{BD-Sp}(\mathbf{N}|G, \alpha) \times \pi(\alpha), \tag{3.7}$$

where $\mathbf{k} = \{\sum_{i=1}^{n} 1_{\{x_i \in B_1\}}, \ldots \sum_{i=1}^{n} 1_{\{x_i \in B_c\}}\}$ are the sample counts in each bin, $n$ is the sample size, and $\vec{\rho}$ is the population proportion within each bin (e.g., $\rho_1 = \frac{1}{N}\sum_{i=1}^{N} 1_{\{y_i \in B_1\}}$). For this example, we will use a Unif$(0, 1)$ continuous prior on $\alpha$.

Since our sample is quite large ($n = 3185$), we generate the posterior for $N_1, N_2, \ldots$ bin counts, without generating $\mathbf{y}$ directly. In this case, we will use bins $\mathbf{B} = \{[0, 5000), [5000, 10000), \ldots, [195000, 200000), [200000, \infty)\}$. Figure 3.35 shows, for each bin, the proportion of the sample within each bin, the mass of the base distribution in each bin.

We initialize our MCMC using the *average* of these two values: the proportion from the sample and the mass of the base distribution. The following pseudocode will sample from the desired distribution (3.7), the combination of the BD-Sp with a Multinomial likelihood

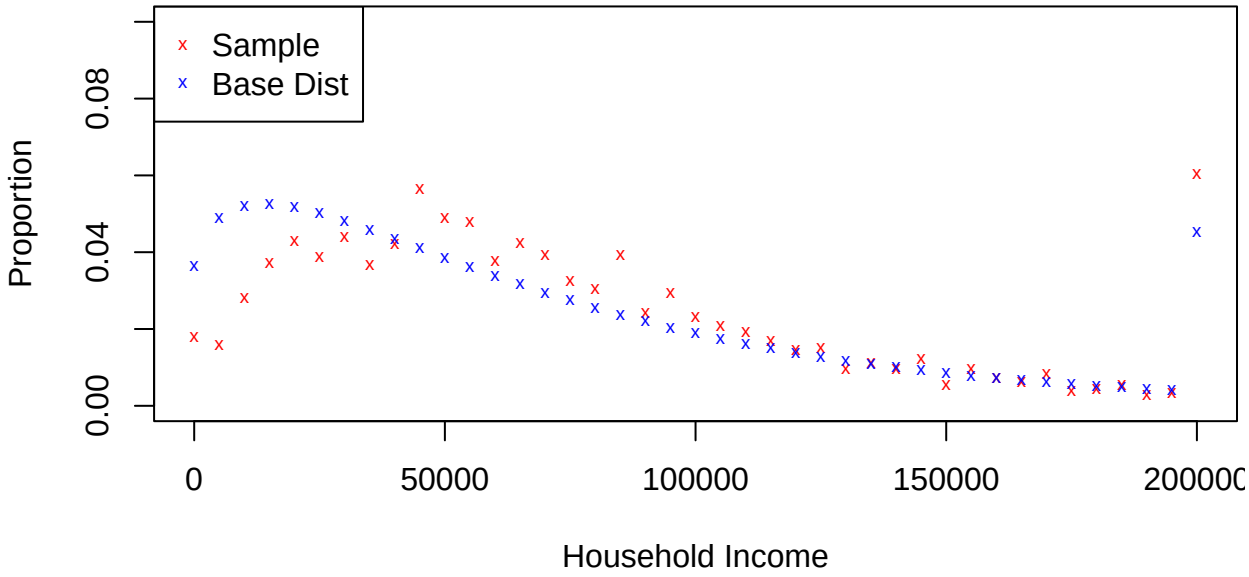


Figure 3.35: Comparison between random sample proportions (red) and base distribution mass (blue) inside each bin (left bin endpoints shown).

from the sample data:

**INPUT**: **x** - sample data

**B** - bins

$G$ - base distribution

$N$ - population size

$T$ - number of iterations

**OUTPUT**: $T$ realizations of **N**

Initialize **N**

**FOR** $i$ **IN** $1, \ldots, T$

Propose $\alpha^*$ from $p(\alpha^*|\alpha)$

Calculate $a = \min\left(1, \frac{f(\mathbf{N},\alpha^*|\mathbf{x})}{f(\mathbf{N},\alpha|\mathbf{x})} \times \frac{p(\alpha|\alpha^*)}{p(\alpha^*|\alpha)}\right)$

Set $\alpha = \alpha^*$ with probability $a$

Propose $\mathbf{N}^*$ from $p(\mathbf{N}^*|\mathbf{N})$

Calculate $a = \min\left(1, \frac{f(\mathbf{N}^*,\alpha|\mathbf{x})}{f(\mathbf{N},\alpha|\mathbf{x})} \times \frac{p(\mathbf{N}|\mathbf{N}^*)}{p(\mathbf{N}^*|\mathbf{N})}\right)$

Set $\mathbf{N} = \mathbf{N}^*$ with probability $a$

**SAVE** $\mathbf{N}, \alpha$

**RETURN** $T$ realizations of $\mathbf{N}$

Figure 3.36 shows 50 realizations of the posterior bin proportions, as well as the sample bin proportions and the base distribution mass within each bin. From this figure, we can see that the resulting posterior follows the Gamma base distribution, while allowing for deviations generally in the direction of the SRS data.

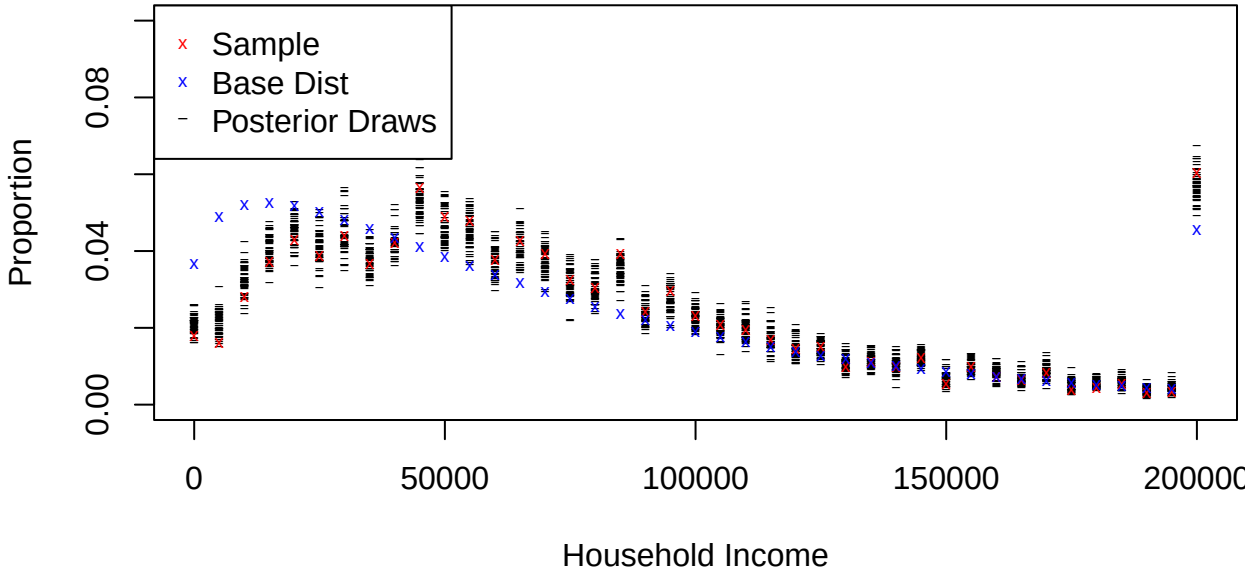


Figure 3.36: Posterior income bin proportions, sample income bin proportions, and base distribution mass within each bin.

Also of interest is the value of $\alpha = \frac{N_{\text{eff}}+1}{N+1}$. Figure 3.37 shows the posterior for $\alpha$. For this application, $\hat{\alpha} = 0.0579405$, while a 90% equal-tailed credible interval is (0.0381504, 0.0879471).

Since we are now modeling the town of Blacksburg, we can visualize this data by creating

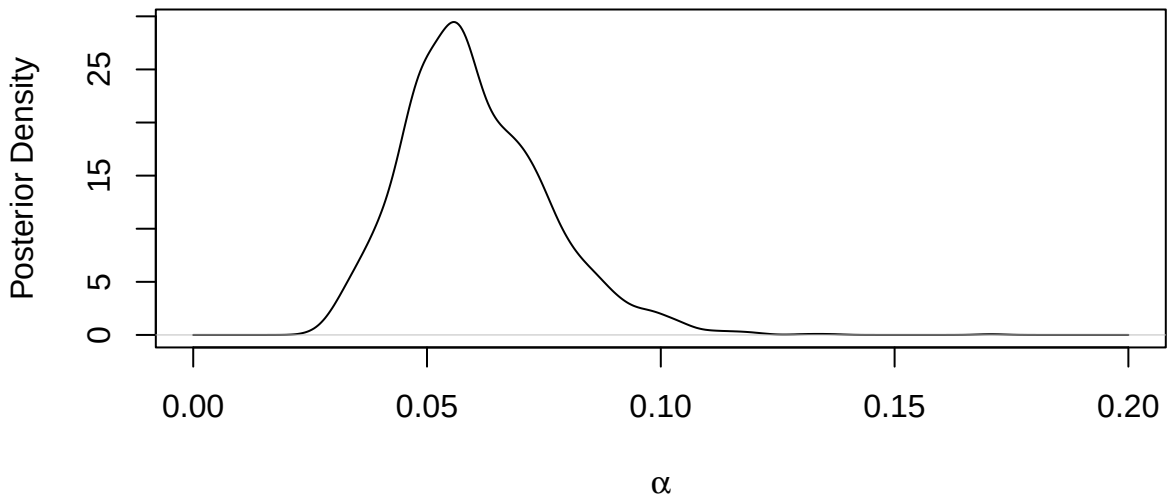


Figure 3.37: Posterior for $\alpha$ from using Binned Dirichlet Spacing prior.

maps of the city. Figure 3.38 shows the results of the Binned Dirichlet Spacing prior, applied to Blacksburg. As we explained in Section 1.3, visualizing the town of Blacksburg in its entirety is difficult, so we zoom in on a small area of the town where several census block groups come together. At this stage of the modeling, we are not including any household characteristics in the model, only aggregated statistics. Thus, we have no choice but to randomly assign incomes to households. Additionally, since we are modeling binned incomes at this stage and we want to plot on a continuous scale, we will randomly assign continuous incomes using a truncated version of the base distribution within each bin (e.g., $y_i \in [5000, 10000) \sim \text{Gamma}(y_i|k, \theta) \times 1_{\{y_i \in [5000,10000)\}}$).

With a framework in place for modeling non-i.i.d. population members, we can move onto more complicated problems: combining multiple data sources in Chapter 4 and modeling multivariate populations in Chapter 5.

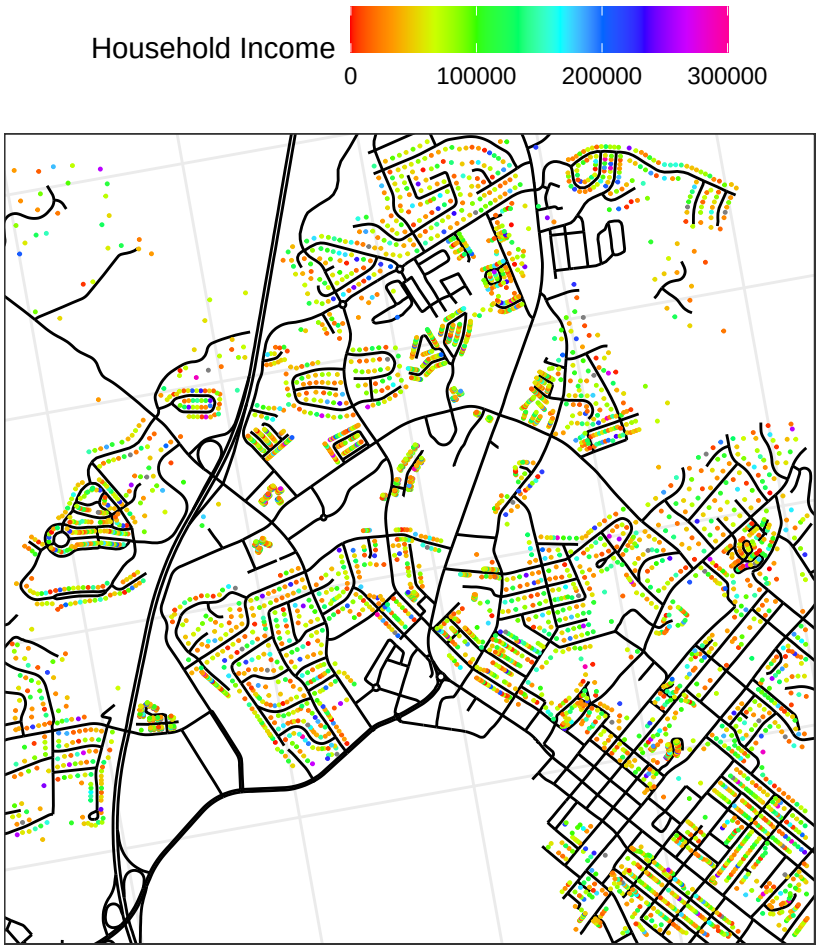


Figure 3.38: One possible realization of Blacksburg, using the Binned Dirichlet Spacing prior.

# Chapter 4

# Using Additional Data

If we are fortunate, there are data sources available to us beyond a simple random sample. While Chapter 3 focused on developing priors for populations in the simple case, where we only have one simple random sample $\mathbf{x}$ to inform our synthetic populations $\mathbf{y}$, this chapter focuses on how we expand our modeling strategy to include other sources of information, whether that be by incorporating data-based likelihoods or updating our prior on populations.

Population $\mathbf{y} = \{y_1, y_2, \dots, y_N\}$, where $y_i$ is $p \times 1$, $p = 1$

Data $\mathbf{x} = \{\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG}, \dots\}$

There is no limit to possible data sources that *could* be available, and thus discussing every possible data source and the appropriate likelihood to use for each is impractical. Instead, we will focus on the data available to us for our specific application of creating synthetic populations of Blacksburg, VA. In Section 1.3, we explored some of the data available for Blacksburg. In addition to a random sample (US Census Bureau 2019c), we also introduced median information and marginal information for variables of interest (US Census Bureau 2019a). Throughout this chapter, we will repeatedly introduce new data sources and likelihoods to model them, building off our previous implementation.

We begin by modeling $\mathbf{y}$ when there is zero data available to us. In this case, our only option (if we want to use the Dirichlet-Spacing prior) is to use a fixed $\alpha = \frac{N_{\text{eff}}+1}{N+1}$, since there is no data available to learn this parameter. From there, we continue by modeling $\mathbf{y}$ when our only source of data is a random sample $\mathbf{x}$. Henceforth, we will denote this type of data $\mathbf{x}^{SRS}$, since we are going to be working with multiple types of data at once. From there, we will incrementally add median information for the Census block groups, and then add regression information utilizing a secondary variable.

Additionally, in Chapter 3, we introduced the idea of modeling $\mathbf{y}$ hierarchically. Specifically, we simplified the Dirichlet distribution to let it govern $\mathbf{N} = \{N_1, N_2, \dots, N_k\}$, the number of elements within each "bin", where the bins could be arbitrarily defined. In order to apply the Dirichlet-Spacing prior to larger problems (such as the population of Blacksburg), we are essentially forced to use this binned version. Since we are using other data sources, and some of them may use binned information, it may make sense to use the same bins. If multiple data sources have different bins, it may be necessary to use a set of bins that can be combined into any of the sets of bins from data sources. Thus, we will let

$$\pi(\mathbf{y}, \mathbf{N}|G, \alpha) \propto g(\mathbf{y}|\mathbf{N}) \times \text{BD-Sp}(\mathbf{N}|G, \alpha), \tag{4.1}$$

where $G$ is our (fixed) base distribution, and BD-Sp is the binned Dirichlet-Spacing prior. Additionally, we will incorporate a hyperprior on $\alpha$, instead of letting it be a fixed constant which we estimate. For our base distribution, we will use the Gamma distribution with parameters estimated from the distribution of national incomes.

As in Chapter 3.3, we use MCMC algorithms to sample from our posterior for $\mathbf{y}$. In the interest of computational efficiency, we will be specifying a proposal distribution that will

cause cancellation with part of our posterior. Specifically, we will use

$$p(\mathbf{y}, \mathbf{N}) = p(\mathbf{y})p(\mathbf{N}|\mathbf{y}) = \prod_{i=1}^{N} g(y_i|\vec{\theta}) \frac{N_{k\,:\,y_i \in B_k}}{N} \tag{4.2}$$

as our proposal, where $g$ is the base distribution (Gamma in this case). Essentially, this proposal choice translates to us cycling through $i = 1, \ldots, N$ population members and proposing a new value for each MCMC iteration. We have to keep in mind that $\mathbf{N}$ is also a random vector, and thus the second part of the proposal accounts for the fact that we are changing $\mathbf{N}$ as well. We could center the proposal for $y_i$ around the current value, but using the full base distribution enables more terms to cancel. When we combine our prior with this proposal distribution to calculate the acceptance probability, we get cancellation from the base distribution term in the posterior and the base distribution term in the proposal. Needless to say, these proposal choices are checked by acceptance rates and MCMC convergence criteria.

## 4.1 No Data, Prior Only

Before we begin incorporating data sources, we model $\mathbf{y}$ exclusively using our prior, as we would be forced to do if no data describing $\mathbf{y}$ was available to us. This largely serves as a point of comparison for the following sections, so that we can see what effect adding our multiple data sources has on the resulting synthetic populations. Since there is no data, we cannot learn the parameter $\alpha$ and will have to pick a specific value. Here, we let $\alpha = 0.3$.

Figure 4.1 shows MCMC samples from the prior distribution influencing $\mathbf{y}$. With no data, our distribution of incomes will closely follow the base distribution. We can see this reflected in the distribution of the populations mirroring the base distribution nearly perfectly. Using the Dirichlet-Spacing prior allows the populations to vary more (in terms of bin proportions)

than would be expected from i.i.d. sampling from the base distribution.

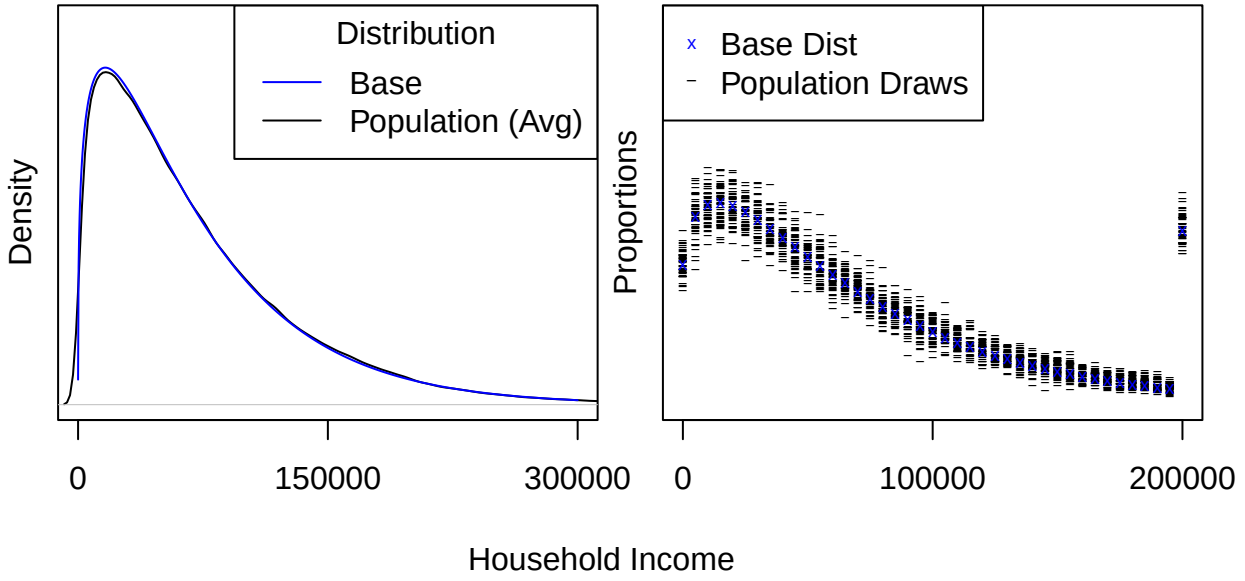


Figure 4.1: Comparison of density of incomes between base distribution and 2000 synthetic populations (left) and comparison of proportions within each bin for the base distribution and 50 synthetic populations (right).

Since we are not incorporating any data, synthetic $\mathbf{y}$ are not matched to real households. However, we can assign income values to households at random. Figure 4.2 shows one realization of $\mathbf{y}$ assigned at random to households within Blacksburg. As before, we show only a relatively small part of the town because visualizing the entire town is difficult.

Throughout this chapter we will add data sources one at a time, continuously comparing to the previous implementation. Now that we have something to compare against, we can begin adding data sources, starting with the random sample data.

## 4.2 Random Sample Data

The process of incorporating random sample data into our sampler will be different depending upon the sample data types (e.g., discrete, continuous, etc.). For Blacksburg, our random

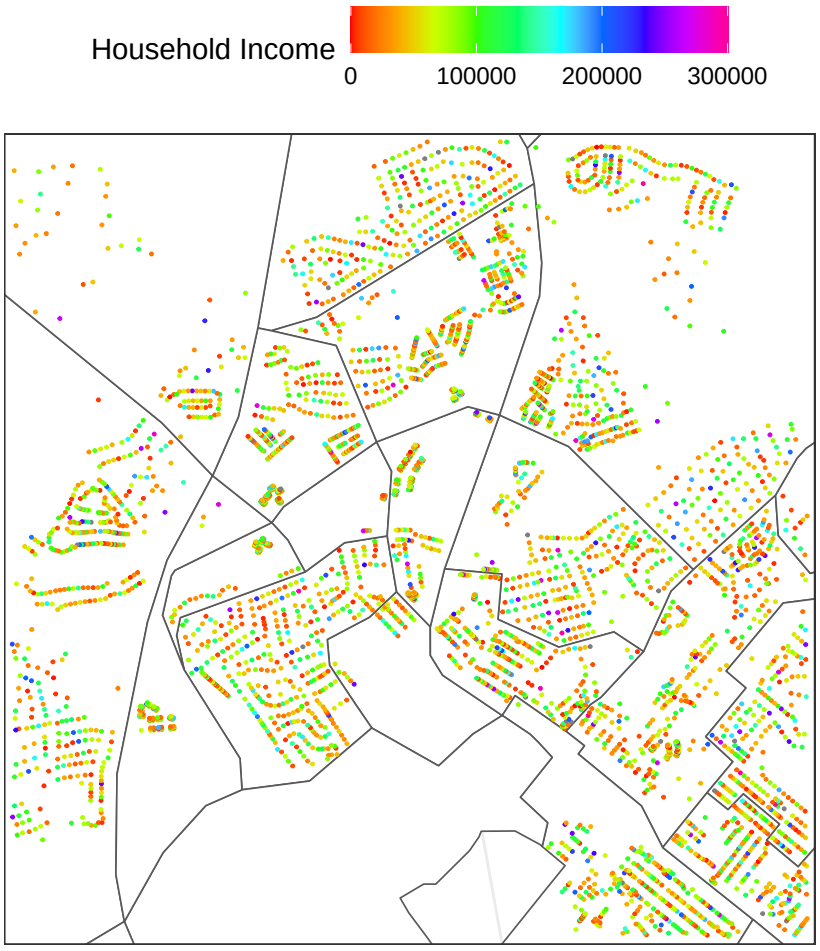


Figure 4.2: One possible realization of Blacksburg, using the Binned Dirichlet Spacing prior with base distribution within bins; Census block group borders shown.

sample comes to us from the American Community Survey, in the form of a Public Use Microdata Sample (2019c). This data of course includes many other variables, but for now we will only be using the income values. The income data are continuous, but we will collapse the incomes into bins

$$\mathbf{B} = \{[0, 5000), [5000, 10000), \ldots, [195000, 200000), [200000, \infty)\}.$$

Figure 4.3 shows how these sample bin proportions compare to our base distribution.

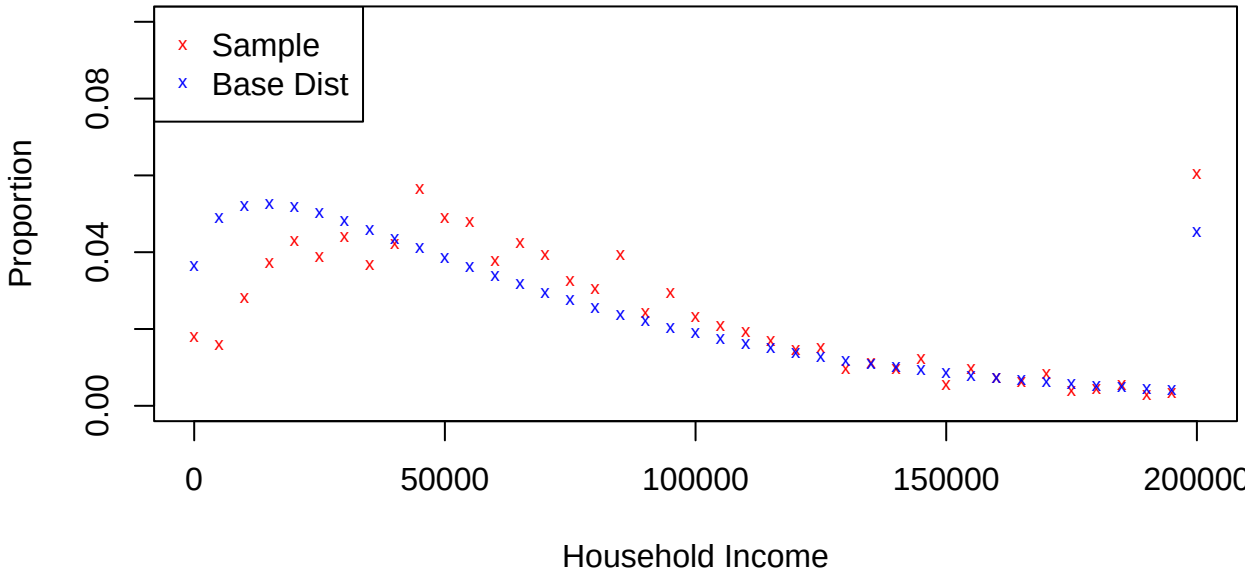


Figure 4.3: Comparison between random sample proportions (red) and base distribution mass (blue) inside each bin (left bin endpoints shown).

In Chapter 3, we stated how to incorporate random sample data into our posterior. Despite this sample likely being taken without replacement, we use the Multinomial likelihood for computational efficiency reasons. Thus,

$$f(\mathbf{x}^{SRS}|\mathbf{y}) \propto \text{Multi}(\mathbf{k}|n, \vec{\rho}),$$

where $\mathbf{k} = \{\sum_{i=1}^{n} 1_{\{x_i \in B_1\}}, \ldots \sum_{i=1}^{n} 1_{\{x_i \in B_c\}}\}$ are the sample counts in each bin, $n$ is the sample size, and $\vec{\rho}$ is the population proportion within each bin (e.g., $\rho_1 = \frac{1}{N}\sum_{i=1}^{N} 1_{\{y_i \in B_1\}}$). This results in a posterior for $\mathbf{y}, \mathbf{N}, \alpha$ of the form

$$f(\mathbf{y}, \mathbf{N}, \alpha|\mathbf{x}^{SRS}) \propto \text{Multi}(\mathbf{k}|n, \vec{\rho}) \times \pi(\mathbf{y}, \mathbf{N}|G, \alpha) \times \pi(\alpha), \tag{4.3}$$

where $\pi(\mathbf{y}, \mathbf{N}|G, \alpha)$ is expressed in Equation (4.1) and $\pi(\alpha) = 1_{\{\alpha \in (0,1)\}}$. To sample from this posterior in Equation (4.3), we will implement the following pseudocode:

**INPUT**: $\mathbf{x}^{SRS}$ - random sample of $\mathbf{y}$

**B** - bins

$G$ - base distribution

$N$ - population size

$T$ - number of iterations

**OUTPUT**: $T$ realizations of $\mathbf{y}$

Initialize $\alpha$, $\mathbf{y}$, and $\mathbf{N}$

**FOR** $i$ **IN** $1, \ldots, T$

Propose $\alpha^*$ from $p(\alpha^*|\alpha)$

Calculate $a = \min\left(1, \frac{f(\mathbf{y},\mathbf{N},\alpha^*|\mathbf{x}^{SRS})}{f(\mathbf{y},\mathbf{N},\alpha|\mathbf{x}^{SRS})} \times \frac{p(\alpha|\alpha^*)}{p(\alpha^*|\alpha)}\right)$

Set $\alpha = \alpha^*$ with probability $a$

**FOR** $j$ **IN** $1, \ldots, N$

Jointly propose $\mathbf{y}^*, \mathbf{N}^*$ from $p(\mathbf{y}^*, \mathbf{N}^*|\mathbf{y}, \mathbf{N})$

Calculate $a = \min\left(1, \frac{f(\mathbf{y}^*,\mathbf{N}^*,\alpha|\mathbf{x}^{SRS})}{f(\mathbf{y},\mathbf{N},\alpha|\mathbf{x}^{SRS})} \times \frac{p(\mathbf{y},\mathbf{N}|\mathbf{y}^*,\mathbf{N}^*)}{p(\mathbf{y}^*,\mathbf{N}^*|\mathbf{y},\mathbf{N})}\right)$

Set $\mathbf{y} = \mathbf{y}^*$ and $\mathbf{N} = \mathbf{N}^*$ with probability $a$

**SAVE** $\mathbf{y}$

**RETURN** $T$ realizations of $\mathbf{y}$

Exact code that implements this process can be found in Appendix 7.2.1. Here we use Equation (4.2) as our proposal $p(\mathbf{y}^*, \mathbf{N}^*|\mathbf{y}, \mathbf{N})$. Utilizing SRS data in our sampler should pull the population bin proportions towards the sample bin proportions. The results of fitting model (4.3) on the Blacksburg data are shown in Figure 4.4, which indeed show the posterior bin proportions pulled in the direction we expected.

Additionally, we can see that the average distribution of incomes across all synthetic populations is pulled towards the distribution of incomes from the sample. Both of these effects

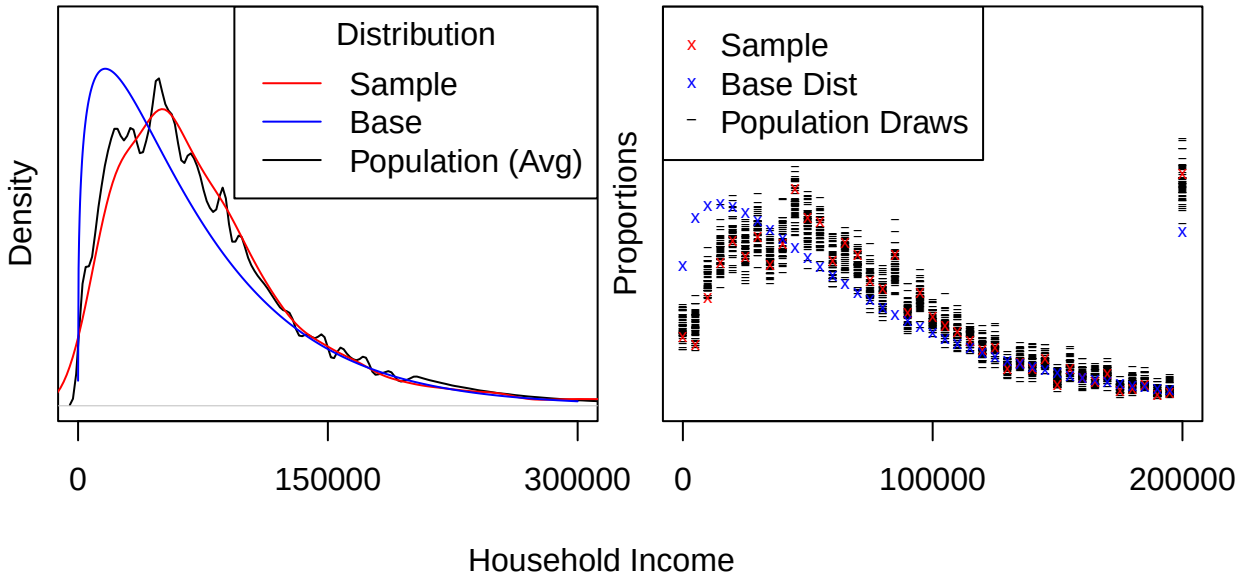


Figure 4.4: Comparison of density of incomes between sample and 2000 synthetic populations (left) and comparison of proportions within each bin for sample, base distribution, and 50 synthetic populations (right), when incorporating the random sample information into our sampler.

are expected, since the likelihood for $\mathbf{x}^{SRS}$ has nothing to compete with other than the prior, meaning this likelihood has a large impact on the overall distribution of incomes. As we add in additional data sources, the effect this data source has on the overall distribution should be lessened.

Figure 4.5 shows one realization of $\mathbf{y}$, again randomized within Blacksburg since there is no geographic data included in the model at this time. Visually, there does not appear to be much difference between this implementation and the previous, which did not include $\mathbf{x}^{SRS}$. We know the overall density of incomes has changed, but visually it is difficult to see on a map.

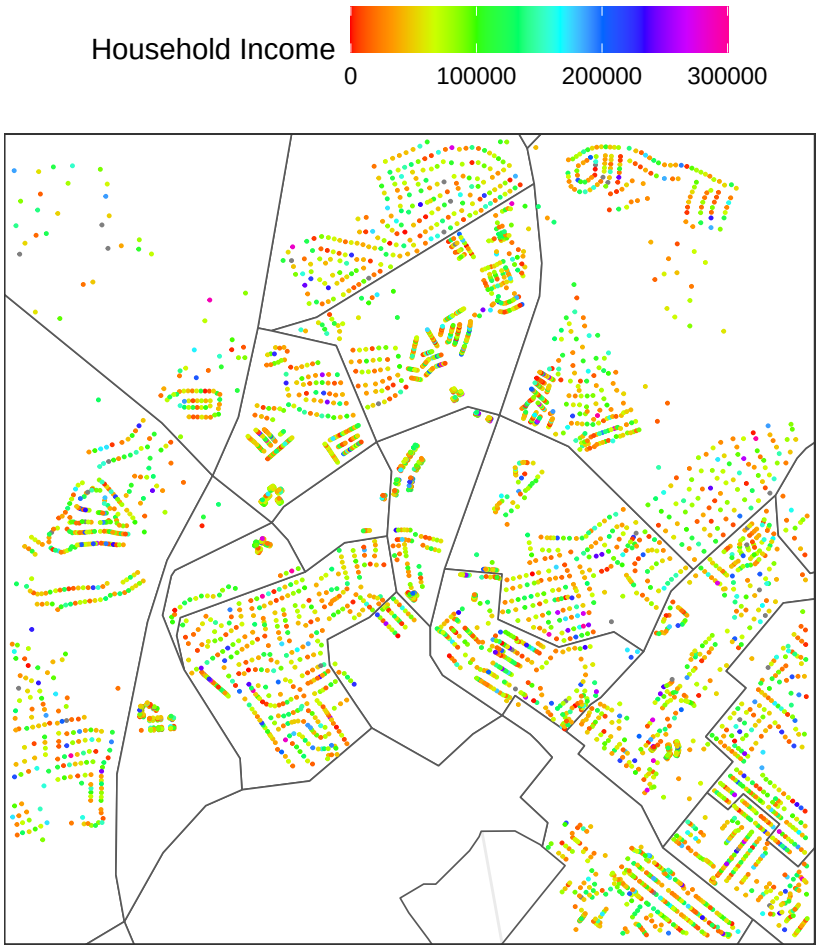


Figure 4.5: One possible realization of Blacksburg, using the posterior for $\mathbf{y}$ with the SRS likelihood; Census block group borders shown.

## 4.3 Median Data

The next type of data that we will add is also from the American Community Survey (2019a), which provides estimates and margins of error for the median income within each census block group that overlaps with Blacksburg. Figure 4.6 shows the data available to us. We calculate the standard errors assuming that they are using confidence intervals based on the central limit theorem, dividing the margin of error by $Z_{0.95}$ (the margin of error they provide is for a 90% interval).

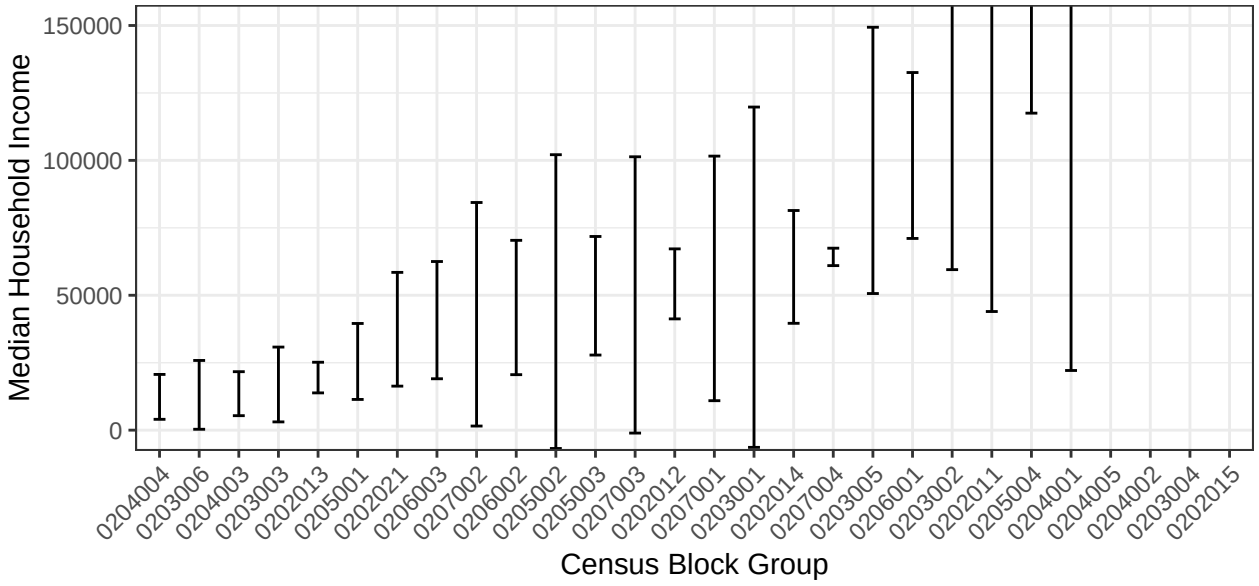


Figure 4.6: Estimated block group median household incomes with error bar representing 90% margin of error.

We model the median of each geographic area using

$$m_i \sim \mathrm{N}(\mu_i, \sigma(m_i)),$$

where $m_i$ and $\sigma(m_i)$ are the median and standard error of the median for a block group. Combining the information about every group mean or median yields the following likelihood:

$$f(\mathbf{x}^{MD}|\mathbf{y}) \propto \prod_{i=1}^{g} \mathrm{N}(m_i|\mu_i, \sigma(m_i)).$$

Combining this likelihood with the posterior we used in Section 4.2 yields

$$f(\mathbf{y}, \mathbf{N}, \alpha|\mathbf{x}^{SRS}, \mathbf{x}^{MD}) \propto \mathrm{N}(\mathbf{m}|\vec{\mu}, \mathrm{diag}(\vec{\sigma})) \times \mathrm{Multi}(\mathbf{k}|n, \vec{\rho}) \times \pi(\mathbf{y}, \mathbf{N}|G, \alpha) \times \pi(\alpha), \quad (4.4)$$

where $\pi(\mathbf{y}, \mathbf{N}|G, \alpha)$ is expressed in equation (4.1) and $\pi(\alpha) = 1_{\{\alpha \in (0,1)\}}$. This new posterior combines what we learned in Section 4.2 with block group median information and informs us

of plausible values of $\mu_i$, the population medians for each census block group. To incorporate the median data into our model and generate samples, we provide the following pseudocode:

**INPUT**: $\mathbf{x}^{SRS}$ - random sample of $\mathbf{y}$

$\mathbf{x}^{MD}$ - medians (or means) with standard errors

$\mathbf{B}$ - bins

$G$ - base distribution

$N$ - population size

$T$ - number of iterations

**OUTPUT**: $T$ realizations of $\mathbf{y}$

Initialize $\alpha$, $\mathbf{y}$, and $\mathbf{N}$

**FOR** $i$ **IN** $1, \ldots, T$

Propose $\alpha^*$ from $p(\alpha^*|\alpha)$

Calculate $a = \min\left(1, \frac{f(\mathbf{y},\mathbf{N},\alpha^*|\mathbf{x}^{SRS},\mathbf{x}^{MD})}{f(\mathbf{y},\mathbf{N},\alpha|\mathbf{x}^{SRS},\mathbf{x}^{MD})} \times \frac{p(\alpha|\alpha^*)}{p(\alpha^*|\alpha)}\right)$

Set $\alpha = \alpha^*$ with probability $a$

**FOR** $j$ **IN** $1, \ldots, N$

Jointly propose $\mathbf{y}^*, \mathbf{N}^*$ from $p(\mathbf{y}^*, \mathbf{N}^*|\mathbf{y}, \mathbf{N})$

Calculate $a = \min\left(1, \frac{f(\mathbf{y}^*,\mathbf{N}^*,\alpha|\mathbf{x}^{SRS},\mathbf{x}^{MD})}{f(\mathbf{y},\mathbf{N},\alpha|\mathbf{x}^{SRS},\mathbf{x}^{MD})} \times \frac{p(\mathbf{y},\mathbf{N}|\mathbf{y}^*,\mathbf{N}^*)}{p(\mathbf{y}^*,\mathbf{N}^*|\mathbf{y},\mathbf{N})}\right)$

Set $\mathbf{y} = \mathbf{y}^*$ and $\mathbf{N} = \mathbf{N}^*$ with probability $a$

**SAVE** $\mathbf{y}$

**RETURN** $T$ realizations of $\mathbf{y}$

Code that implements this process can be found in Appendix 7.2.2. Once again, we use Equation (4.2) as our proposal $p(\mathbf{y}^*, \mathbf{N}^*|\mathbf{y}, \mathbf{N})$. Figure 4.7 shows the same information as Figure 4.6, with medians from our synthetic populations overlaid.

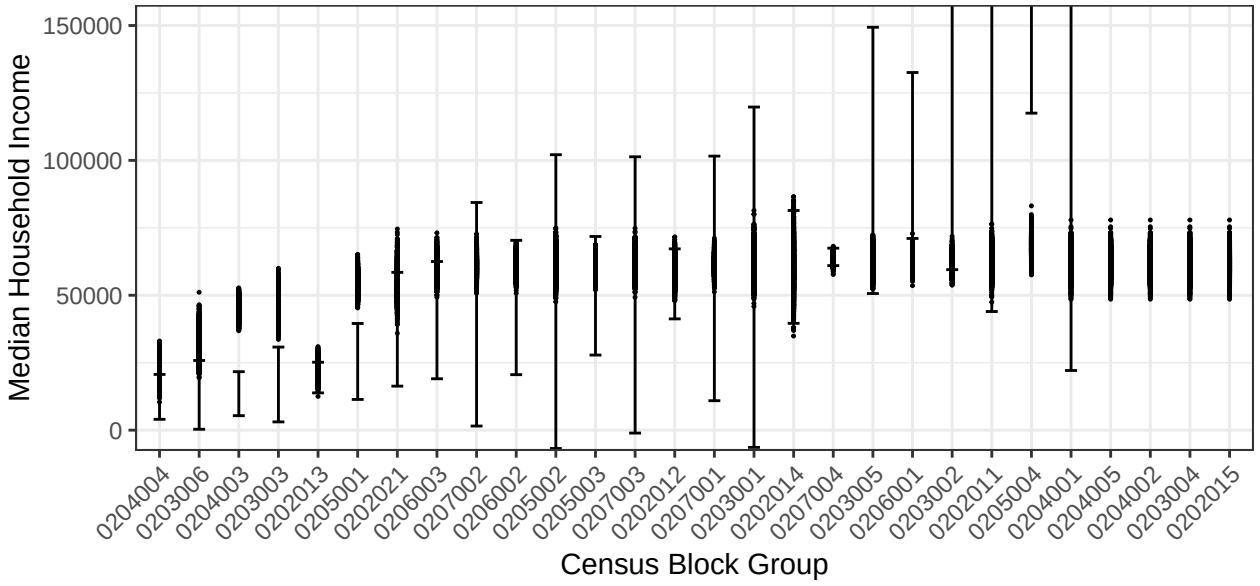


Figure 4.7: Medians with margin of error (90%), overlaid with medians from 2000 synthetic populations.

It is rather clear to see that the synthetic population medians are not strictly adhering to the data. But, because of data challenges, the model is fitting as expected. First, the margins of error provided by the ACS are quite large (and completely missing for 4 of 28 block groups). The ACS is answered by a relatively small percentage of the population, so the margins of error they provide are unfortunately quite large. In fact, some block groups have zero respondents, which is why a few of the block groups have no median information at all. It makes sense for our model to minimize the influence of highly variable data.

Second, not all of our data $\mathbf{x}$ can be adhered to at the same time. Essentially, the likelihoods from different data sources (in this case, $\mathbf{x}^{SRS}$ and $\mathbf{x}^{MD}$) disagree with each other. In this case, the Multinomial likelihood from $\mathbf{x}^{SRS}$ is heavily influencing the number of houses within each income bin. Given a certain distribution of households in the income bins, satisfying the medians becomes *impossible*. To illustrate, we run the sampler using *only* median information, ignoring $\mathbf{x}^{SRS}$. Figure 4.8 shows these results, where we see a much stronger agreement between the median data and out synthetic populations. This dis-

agreement between data sources is unfortunately bound to happen. Here we simply include both likelihoods with equal weight and let them compete with each other; however, if we had more trust in one data source over another we could down-weight specific likelihoods to reflect our level of trust.

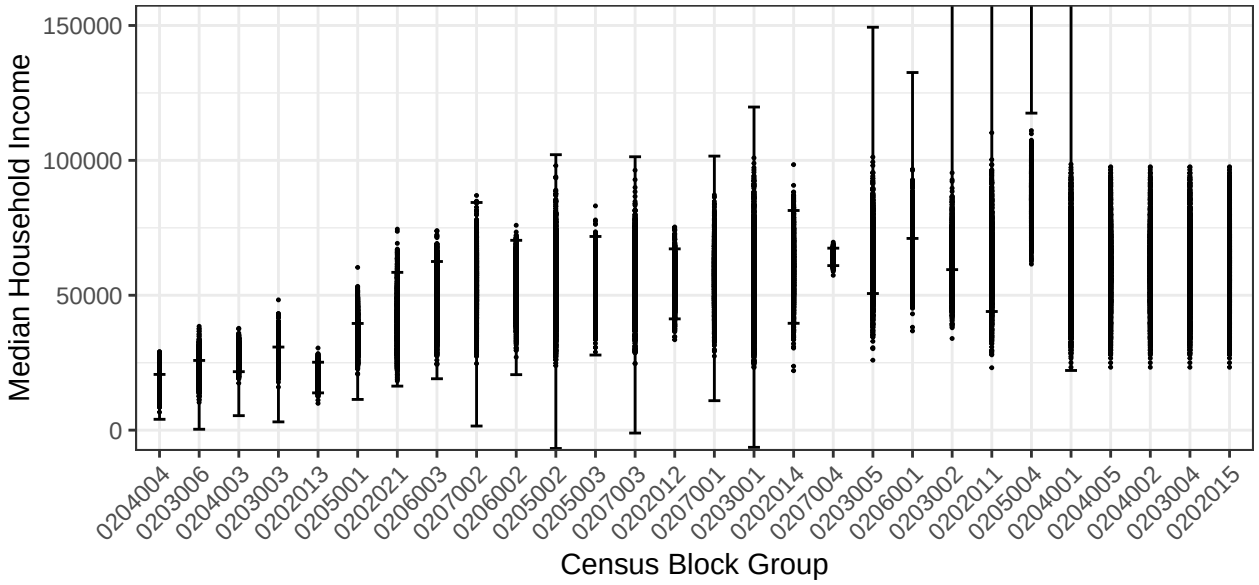


Figure 4.8: Medians with margin of error (90%), overlaid with medians from 2000 synthetic populations.

Returning to the results with both likelihoods included, Figure 4.9 shows the average density of the synthetic populations and the distribution of the proportions within each bin. Comparing these results to 4.4, which only utilized the likelihood for $\mathbf{x}^{SRS}$, we note two observations. Firstly, the average density of the populations $\mathbf{y}$ is relatively unchanged, and secondly, some of the effect that $\mathbf{x}^{SRS}$ had on the distribution of the proportions within each bin has been negated.

For the first time, we are now incorporating some sort of location information. In this case, we only have block groups specified, so instead of randomizing over the entire town of Blacksburg, we can instead randomize inside of the block groups. Figure 4.10 shows these results; unfortunately, it is not visually obvious that these census block groups have quite

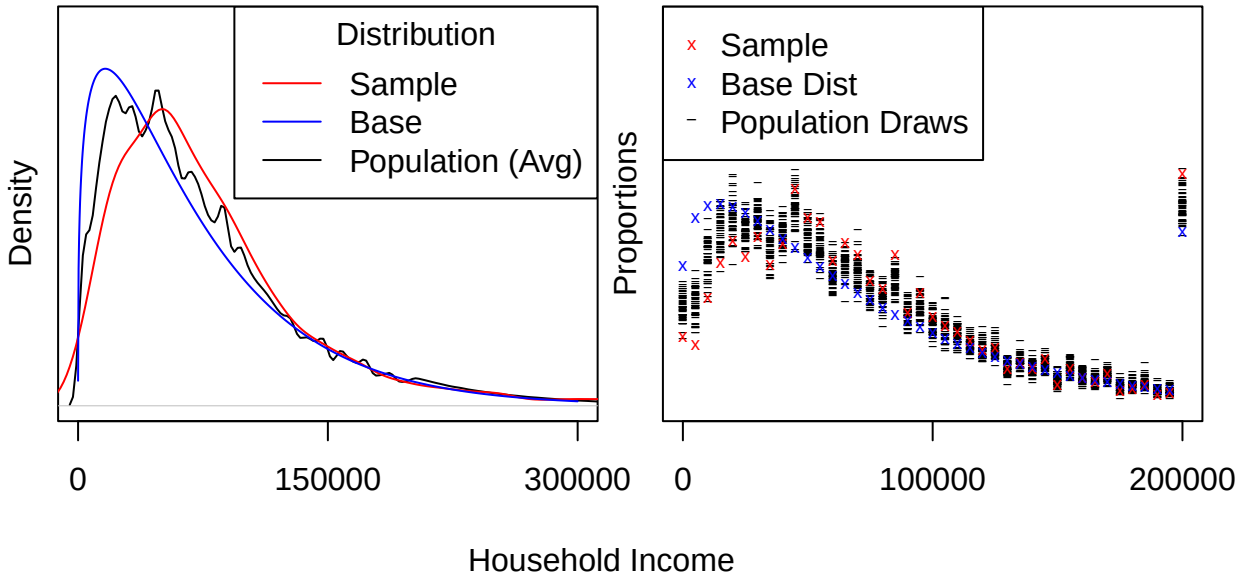


Figure 4.9: Comparison of density of incomes between sample and 2000 synthetic populations (left) and comparison of proportions within each bin for sample, base distribution, and 50 synthetic populations (right), when including the random sample and median information into our sampler.

different medians.

To further investigate these block group medians, we show the same geographic area with the block groups colored by their median income averaged over all of our draws from the posterior. Figure 4.11 shows these results.

## 4.4 Regression Data

The final data that we incorporate into the sampler is regression data $\mathbf{x}^{REG}$. From the parcel records, we have a property value for every parcel in Blacksburg. We use the fact that property value and income are correlated to influence plausible values for each household's income. For Blacksburg, $\mathbf{x}^{SRS}$ provides us everything we need in order to construct this regression model. Figure 4.12 shows the relationship between our variable of interest income and property value within the random sample. Notably, we take the square root of each

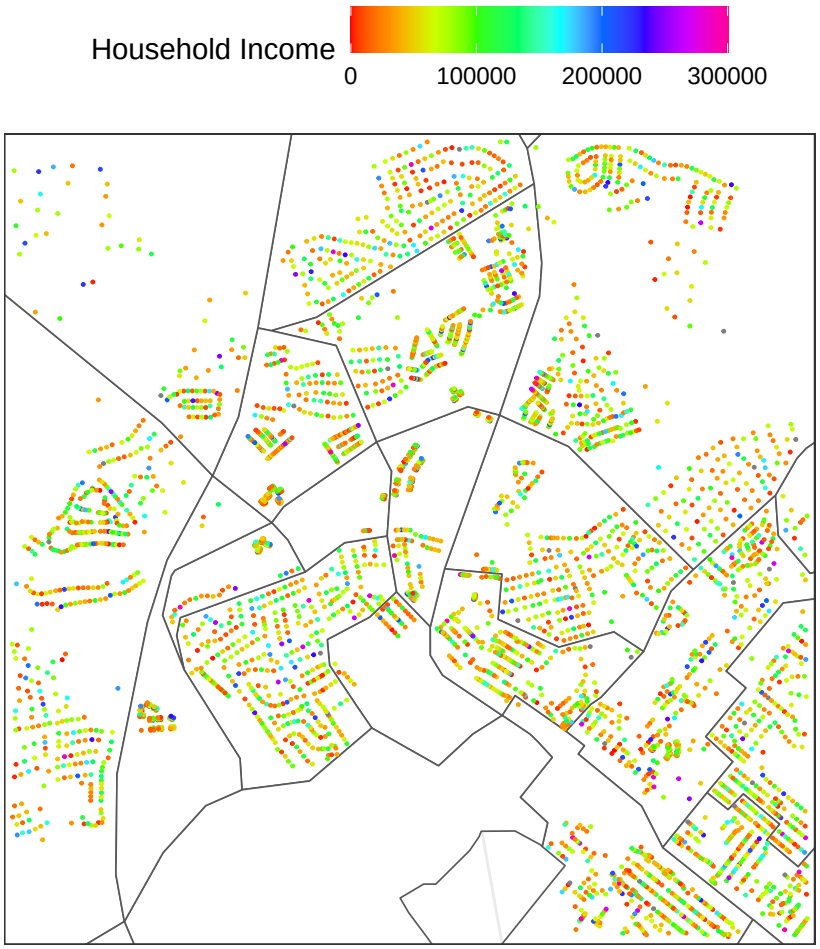


Figure 4.10: One possible realization of Blacksburg, using the posterior for $\mathbf{y}$ with the SRS and median likelihoods; Census block group borders shown.

variable because doing so provided us with a stronger linear relationship.

Since we are now dealing with two variables within the population (though only one is being synthesized), we must introduce some new notation. We will denote $\mathbf{y}^u$ the unknown variable of interest, income, and $\mathbf{y}^k$ the known variable, property value. The goal is to use a regression model $\mathbf{y}^k \sim \mathbf{y}^u$ with parameters $\vec{\beta}$ and $\tau$. However, since we do not have population income values, we cannot fit this model. Instead, we can estimate our parameters using a similar regression model on the sample data, $\mathbf{x}^k \sim \mathbf{x}^u$. We use this information to construct a

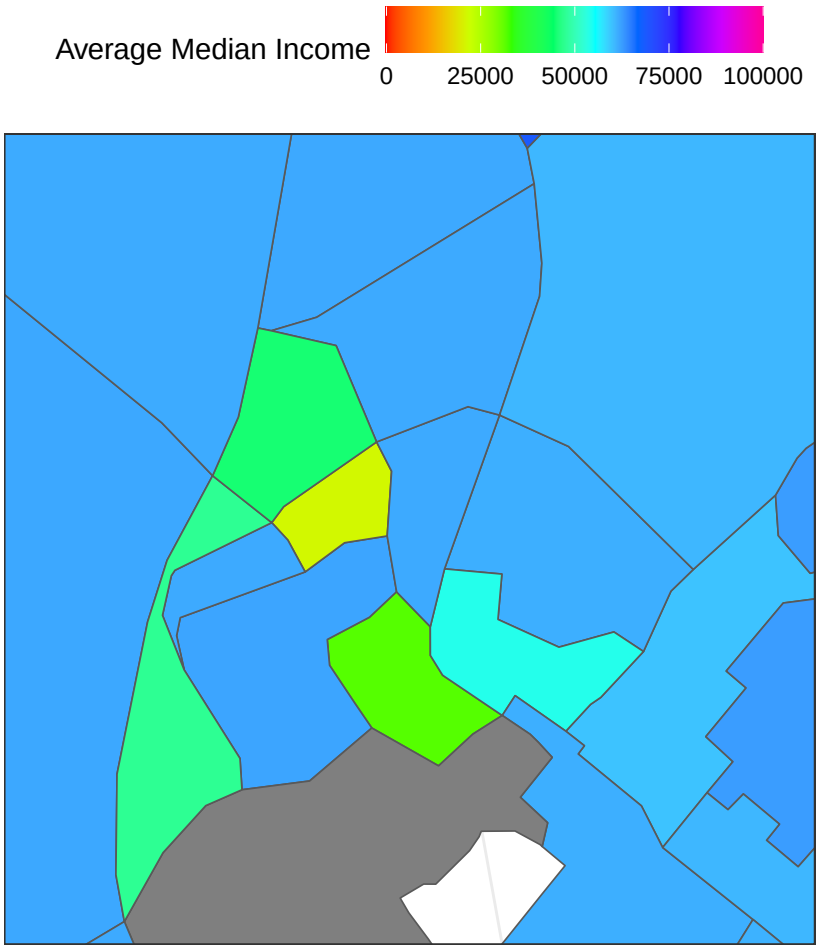


Figure 4.11: Average of block group medians, using the posterior for $\mathbf{y}$ with the SRS and median likelihoods.

multivariate prior on $(\mathbf{y}^u, \mathbf{y}^k)$ of the form

$$\pi(\mathbf{y}^u, \mathbf{y}^k, \mathbf{N}|\alpha, \vec{\beta}, \tau) \propto \pi(\mathbf{y}^u, \mathbf{N}|\alpha) f(\mathbf{y}^k|\mathbf{y}^u, \vec{\beta}, \tau).$$

Here, $\pi(\mathbf{y}^u, \mathbf{N}|\alpha)$ is the prior we used in the previous sections (see Section 4.3) of this chapter.

Thus, our new prior becomes

$$\pi(\mathbf{y}^u, \mathbf{y}^k, \mathbf{N}|\alpha, \vec{\beta}, \tau) = \mathrm{N}(\mathbf{y}^k|\beta_0\mathbf{1} + \beta_1\mathbf{y}^u, \mathrm{diag}(\tau)) \times g(\mathbf{y}^u|\mathbf{N}) \times \text{BD-Sp}(\mathbf{N}|G, \alpha). \quad (4.5)$$


Figure 4.12: Scatterplot showing relationship between household income and property value from our random sample (n = 2071).

This will function as an *updated prior* on $\mathbf{y}^u$ that incorporates information from known variable $\mathbf{y}^k$. Here we are only modeling one variable of interest. Note that we could do something similar for multiple unknown variables of interest, but in this chapter we are limiting the scope to synthesizing single variables. Combining the updated prior in Equation (4.5) with the likelihood from Section 4.3 yields the posterior

$$\pi(\mathbf{y}^u, \mathbf{y}^k, \mathbf{N}, \alpha | \mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG}) \propto \mathrm{N}(\mathbf{m}|\vec{\mu}, \mathrm{diag}(\vec{\sigma})) \times \mathrm{Multi}(\mathbf{k}|n, \vec{\rho}) \times \pi(\mathbf{y}^u, \mathbf{y}^k, \mathbf{N}|\alpha, \vec{\beta}, \tau) \times \pi(\alpha),$$

where $\pi(\mathbf{y}^u, \mathbf{y}^k, \mathbf{N}|\alpha, \vec{\beta}, \tau)$ is defined in Equation (4.5), and $\pi(\alpha) = 1_{\{\alpha \in (0,1)\}}$.

The following pseudocode will sample from the posterior (**??**) of $\{\mathbf{y}^u, \mathbf{N}\}$, incorporating the updated prior described above.

**INPUT**: $\mathbf{x}^{SRS}$ - random sample of $\mathbf{y}$

$\mathbf{x}^{MD}$ - medians (or means) with standard errors

$\mathbf{x}^{REG}$ - all information needed to construct regression model above

**B** - bins

$G$ - base distribution

$N$ - population size

$T$ - number of iterations

**OUTPUT**: $T$ realizations of $\mathbf{y}$

Initialize $\alpha$, $\mathbf{y}$, and $\mathbf{N}$

**FOR** $i$ **IN** $1, \ldots, T$

Propose $\alpha^*$ from $p(\alpha^*|\alpha)$

Calculate $a = \min\left(1, \frac{f(\mathbf{y}^u, \mathbf{y}^k, \mathbf{N}, \alpha^*|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})}{f(\mathbf{y}^u, \mathbf{y}^k, \mathbf{N}, \alpha|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})} \times \frac{p(\alpha|\alpha^*)}{p(\alpha^*|\alpha)}\right)$

Set $\alpha = \alpha^*$ with probability $a$

**FOR** $j$ **IN** $1, \ldots, N$

Jointly propose $\mathbf{y}^{u*}, \mathbf{N}^*$ from $p(\mathbf{y}^{u*}, \mathbf{N}^*|\mathbf{y}^u, \mathbf{N})$

Calculate $a = \min\left(1, \frac{f(\mathbf{y}^u, \mathbf{y}^k, \mathbf{N}, \alpha^*|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})}{f(\mathbf{y}^u, \mathbf{y}^k, \mathbf{N}, \alpha|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})} \times \frac{p(\mathbf{y}^u, \mathbf{N}|\mathbf{y}^{u*}, \mathbf{N}^*)}{p(\mathbf{y}^{u*}, \mathbf{N}^*|\mathbf{y}^u, \mathbf{N})}\right)$

Set $\mathbf{y} = \mathbf{y}^*$ and $\mathbf{N} = \mathbf{N}^*$ with probability $a$

**SAVE** $\mathbf{y}$

**RETURN** $T$ realizations of $\mathbf{y}$

Code that implements this process can be found in Appendix 7.2.3.

We incorporate $\mathbf{x}^{REG}$ information into our sampler by applying the above algorithm where our regression describes the relationship between property value and income, after applying a square root transformation to both. We kept income on the original scale for all other parts of the prior and likelihoods. One positive side-effect of incorporating regression information

into our prior is that it helps move block group medians. In Figure 4.13, we can see that compared to Figure 4.7, the block group medians are significantly improved.

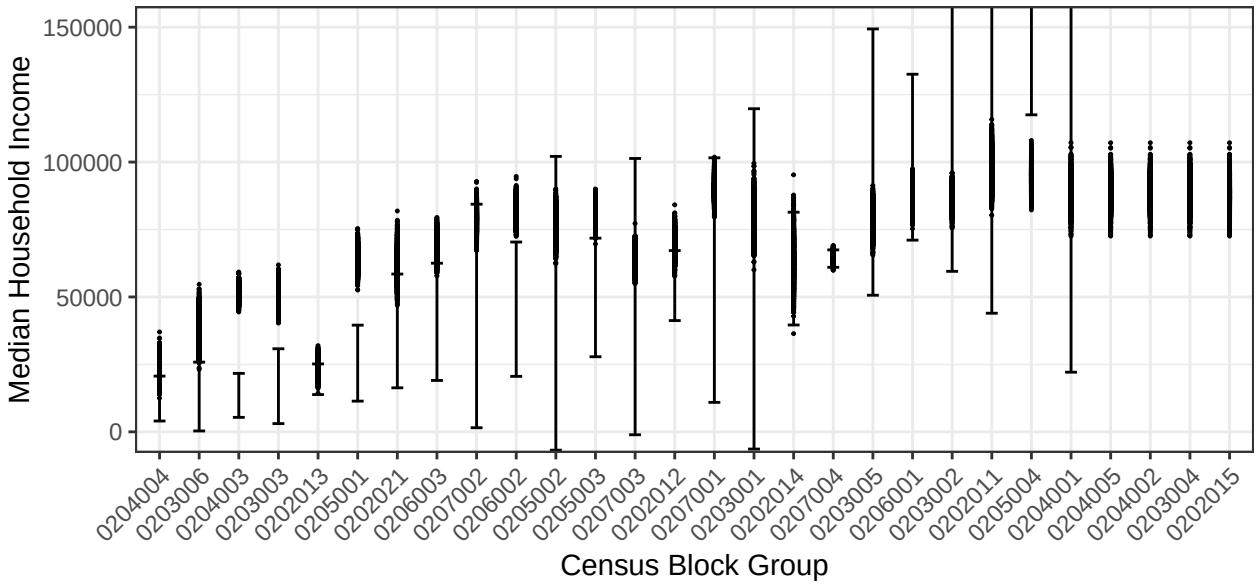


Figure 4.13: Estimated block group medians with margin of error (90%) from ACS, overlaid with medians from 2000 synthetic populations.

Additionally, from Figure 4.14, we can see that while the overall density of incomes is relatively unchanged, the bin proportions are affected substantially. In general, adding in the regression information results in population draws where the proportion of the population within higher income bins is increased, and the proportion of the population within lower income bins is decreased. This behavior is not only expected, but desired, as it makes sense that Blacksburg is skewed slightly compared to our sample. Remember, the sample data comes from the entire PUMA, which includes a large rural area outside of Blacksburg. If we believe that incomes are higher in Blacksburg than the surrounding area, then seeing this distribution skew is a good thing.

From Figure 4.15, we can see that the relationship between income and housing price from the sample is being followed within the synthetic populations. For this plot, we picked 50 synthetic populations at random from the posterior and plotted all population members

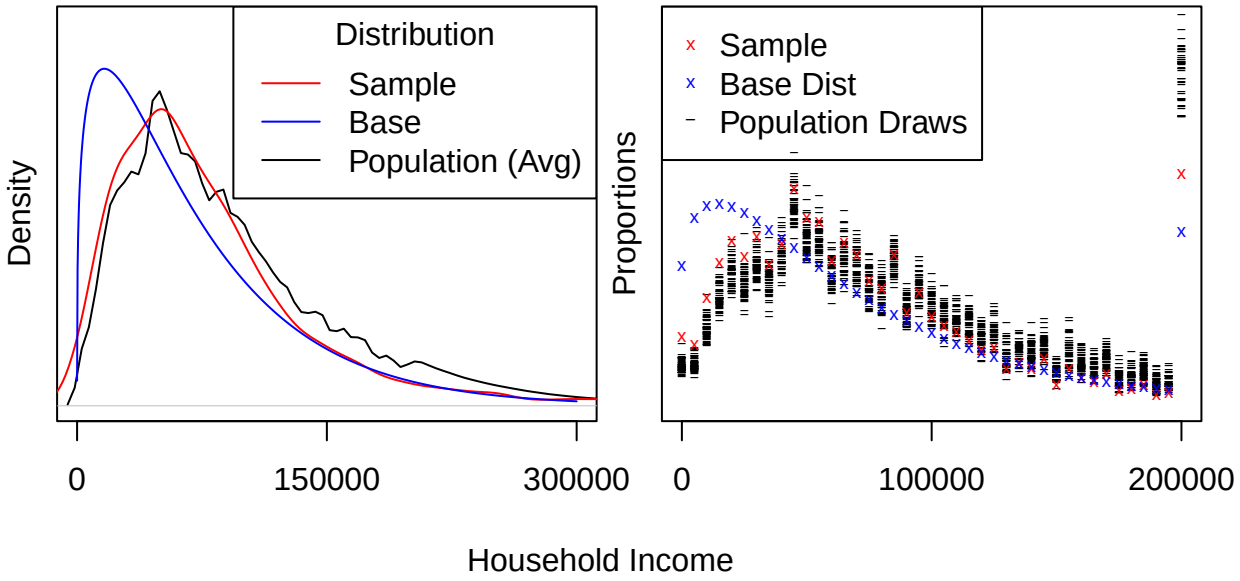


Figure 4.14: Comparison of density of incomes between random sample and 2000 synthetic populations (left) and comparison of proportions within each bin for sample, base distribution, and 50 synthetic populations (right).

underneath the sample data. As desired, the populations follow the same linear relationship as the sample.

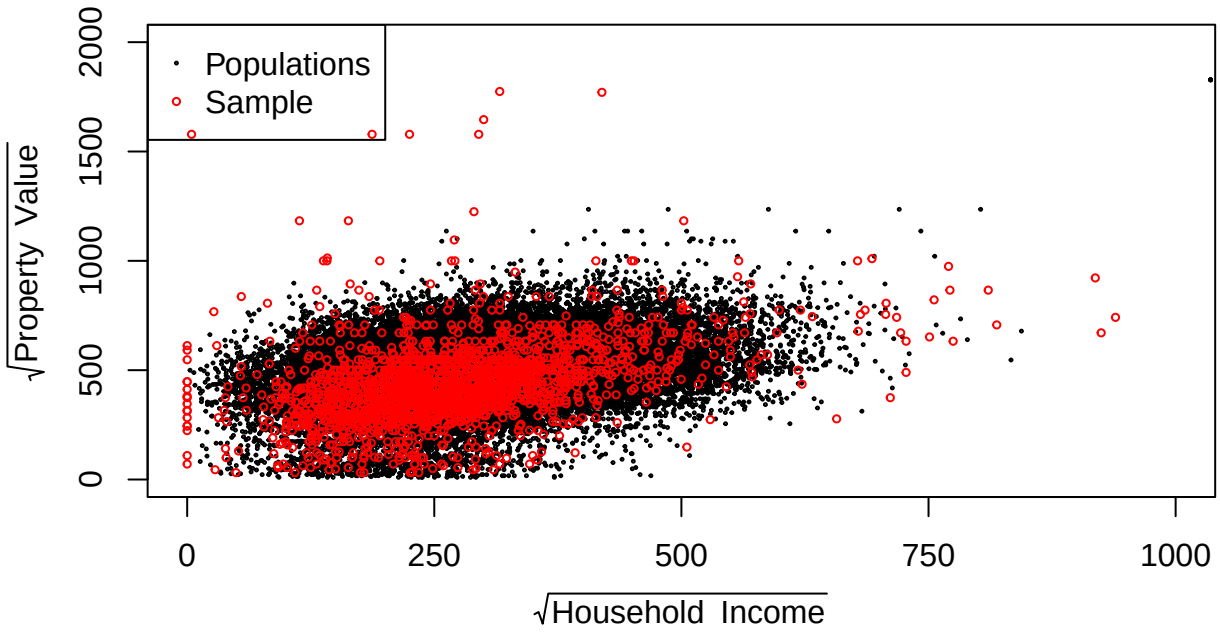


Figure 4.15: Relationship between the square roots of household income and property value in the sample (red) and 2000 synthetic poplations (black).

Figure 4.16 shows a single draw from the posterior displayed on the map of Blacksburg. Compared to Section 4.3, we can clearly see the incomes have been increased overall; there are much more greens and blues and less red on the map. However, when looking at a single draw of the posterior such as this one, the block groups do not stand out as being heterogeneous.

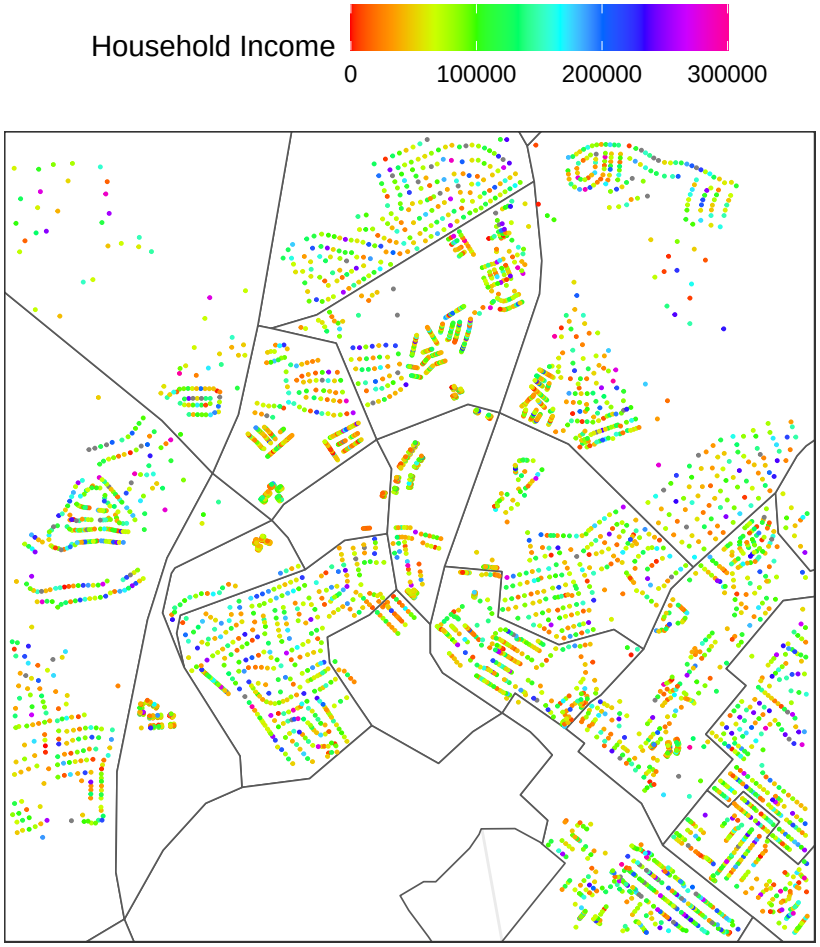


Figure 4.16: One possible realization of household incomes for Blacksburg, using the posterior for **y** with the SRS and median likelihoods, and updated prior using regression information; Census block group borders shown.

Figure 4.17 shows the same map, but this time we take the average of household income over all draws from the posterior. This plot is very interesting in that not only can we see some of the block groups without the aid of the border lines, we are even able to clearly

differentiate some smaller neighborhoods within block groups. While we have not explicitly incorporated spatial correlation into the model, it makes sense that the housing prices $\mathbf{y}^k$ are spatially correlated. Since this variable is now correlated with household income via our new prior, it causes our variable of interest $\mathbf{y}^u$ to also become spatially correlated, allowing us to see clearly see neighborhoods.

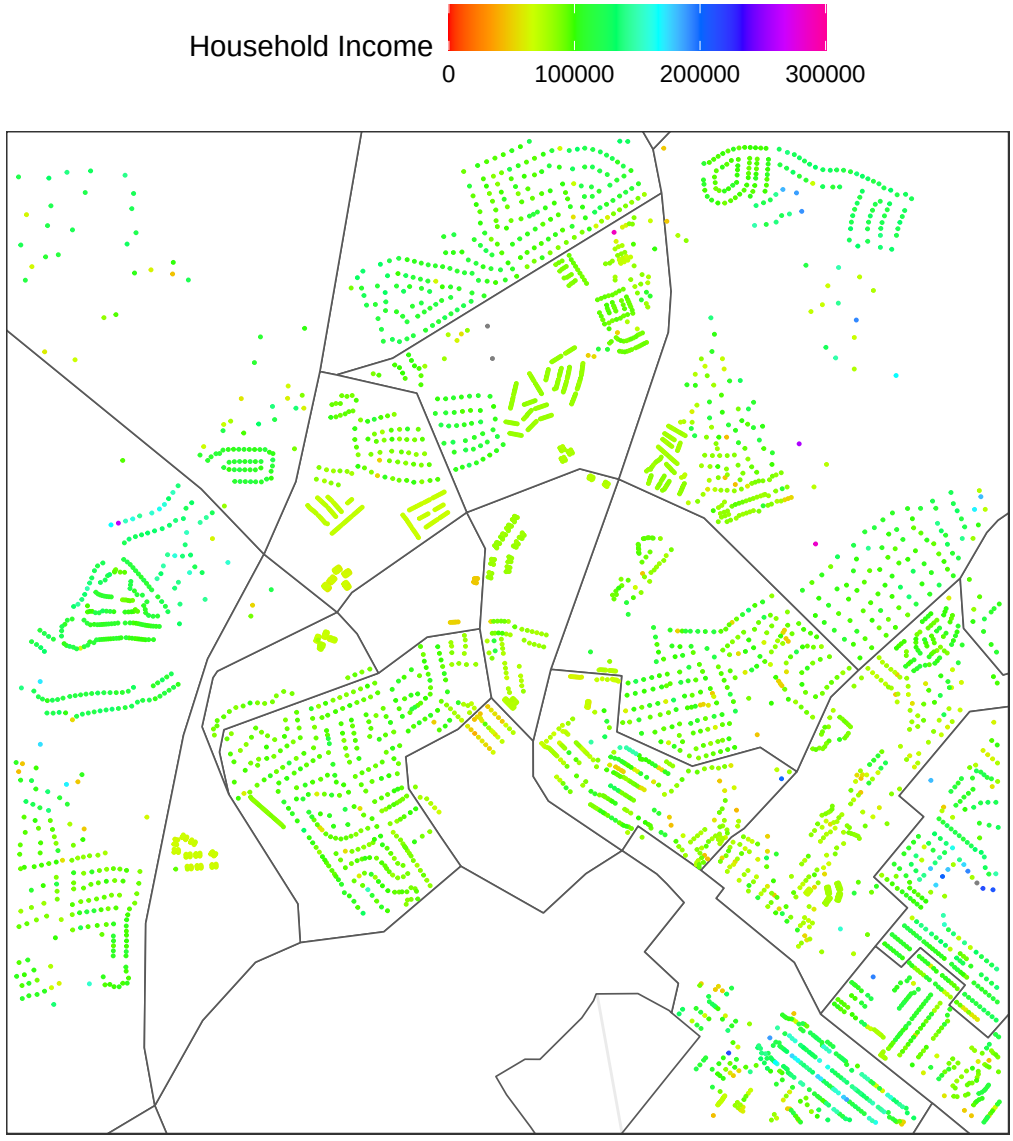


Figure 4.17: Posterior expectation of household incomes for Blacksburg, using the SRS and median likelihoods, and updated prior using regression information; Census block group borders shown.

Figure 4.18 shows the same area, but only overlaying the block group medians. Here we see much more variation in the average block group median incomes than we saw in Section 4.3.


Figure 4.18: Average of block group medians for 2000 synthetic populations, using the posterior for **y** with the SRS and median likelihoods, and updated prior using regression information.

## 4.5 MCMC Diagnostics

Before we move onto the next chapter, where we cover how to synthesize multiple variables of interest jointly, we show MCMC diagnostics from the samplers discussed in this chapter. Since most of the samplers are very similar and only build off each other in complexity, we will limit our scope to the most complex sampler, implemented in Section 4.4.

It is worth mentioning that our MCMC implementation does not have many tunable parameters. Since the population members are proposed with the base distribution to cause terms in the posterior to cancel, our only parameter that has a tunable proposal is $\alpha$ (here we propose with a continuous uniform within 0.08 of the current value). Beyond this, we thin our results by only saving every 5th MCMC iteration.

Figure 4.19 shows trace plots for four quantities of interest. On the top is the mean of the population $\mu(\mathbf{y})$ and parameter $\alpha$. On the bottom is $N_1$ and $N_{41}$, the number of population members in the lowest and highest income bins: $[0, 5000)$ and $[200000, \infty)$. These trace plots are not indicative of any issues with the MCMC.

Likewise, Figure 4.20 shows autocorrelation plots for the same four quantities. While some of these autocorrelations do not decay as quickly as we would like, this problem is easily fixed by increasing the amount by which we thin the MCMC results.

Additionally, we look at the trace plots for the median of a select few census block groups. Considering we are examining group medians, these trace plots shown in Figure 4.21 are surprisingly free of sticking, which often occurs with quantities such as medians, which are not guaranteed to change even when numerous new $y$ values are accepted between iterations.

Finally, we also look at autocorrelation plots for these four block group medians as well. Figure 4.22 shows these results. Surprisingly, these autocorrelations decay even faster than

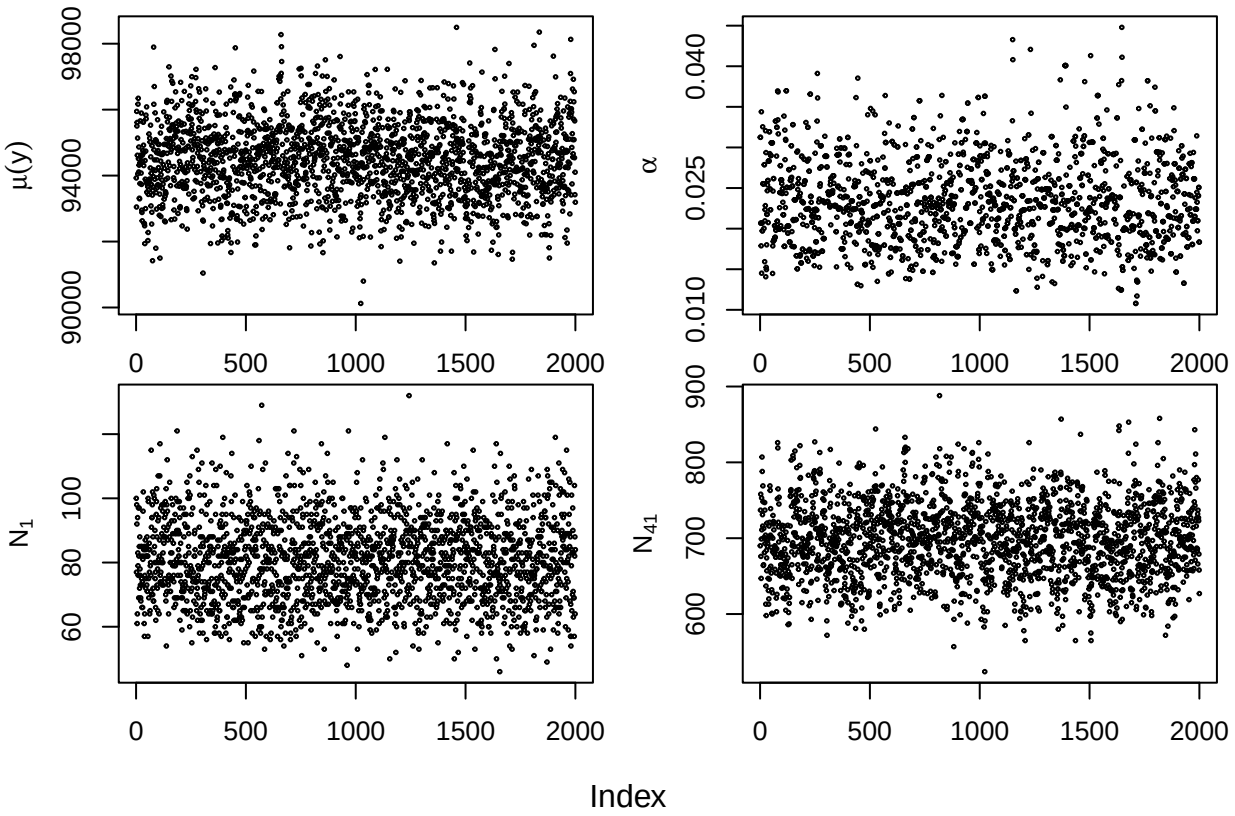


Figure 4.19: Trace plots for four quantities of interest: $\mu(\mathbf{y})$ (top left), $\alpha$ (top right), $N_1$ (bottom left), and $N_{41}$ (bottom right).

our four quantities of interest in Figure 4.20.

Taken as a whole, these MCMC diagnostics are not indicative of any serious problems.

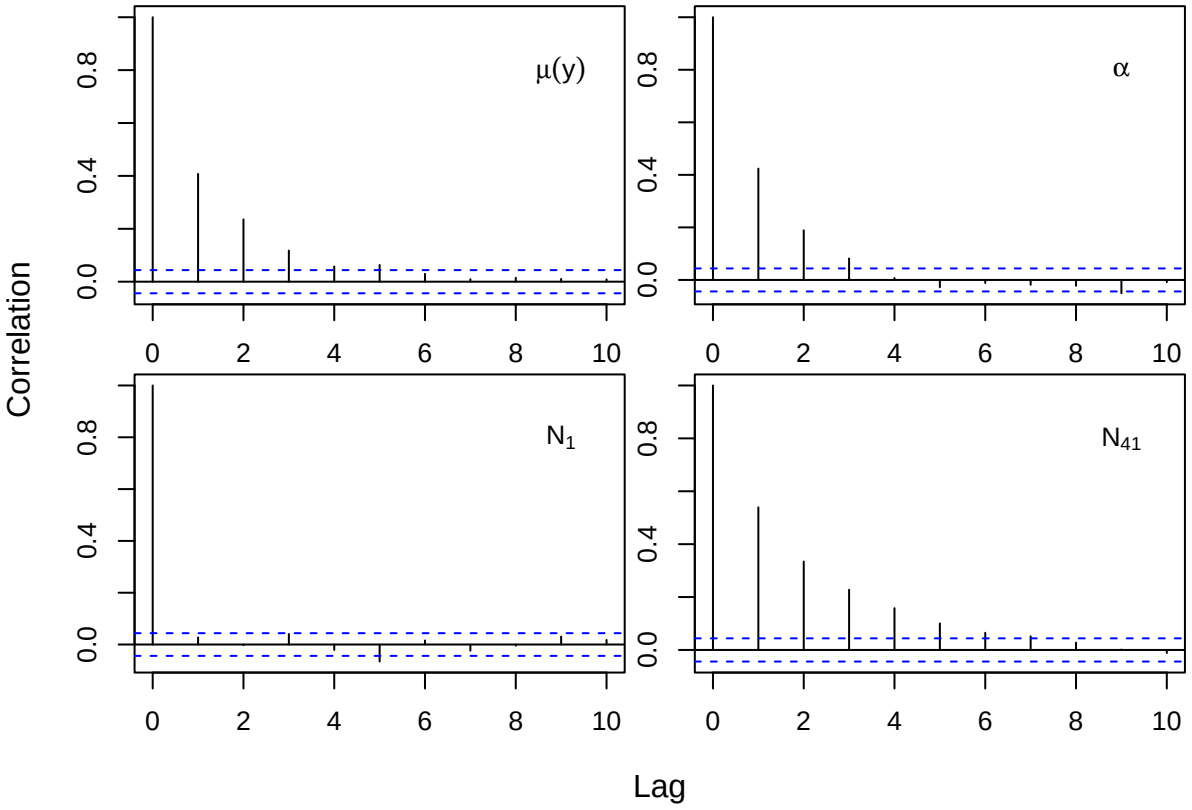


Figure 4.20: ACF plots for four quantities of interest: $\mu(\mathbf{y})$ (top left), $\alpha$ (top right), $N_1$ (bottom left), and $N_{41}$ (bottom right).

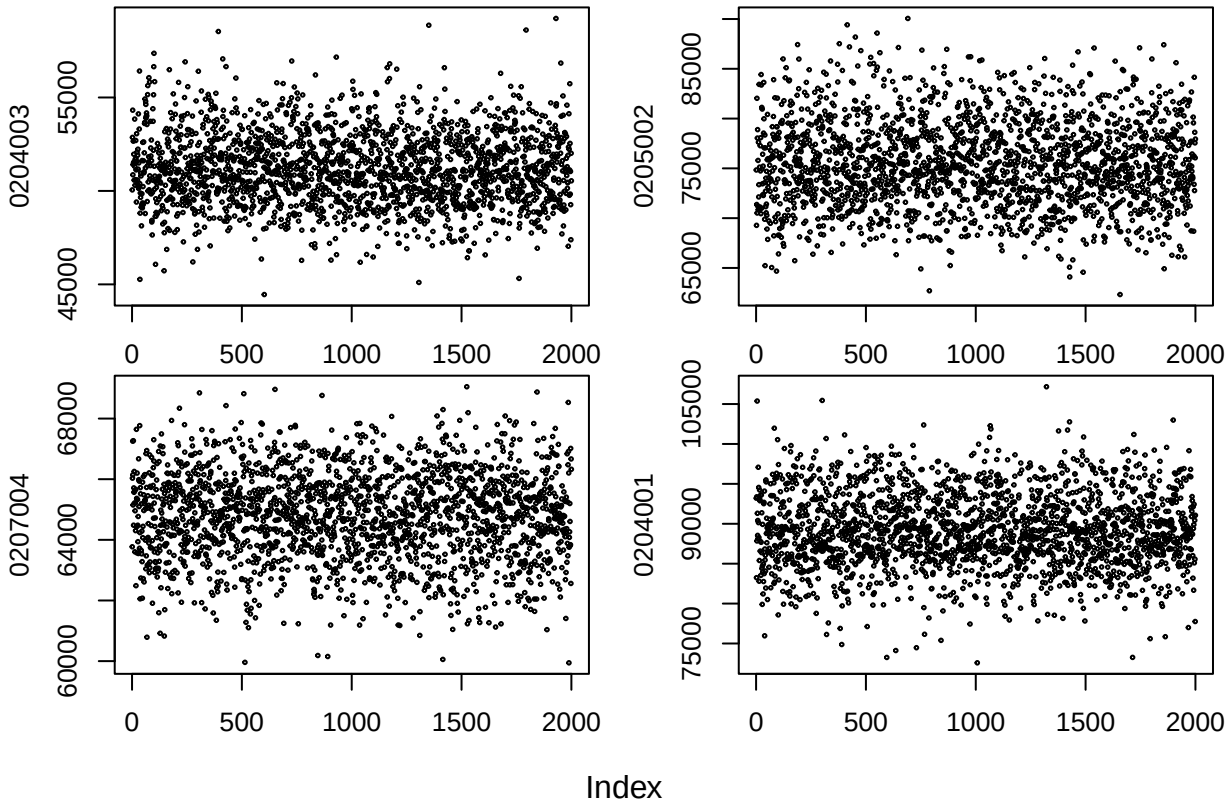


Figure 4.21: Trace plots for four select census block group medians.

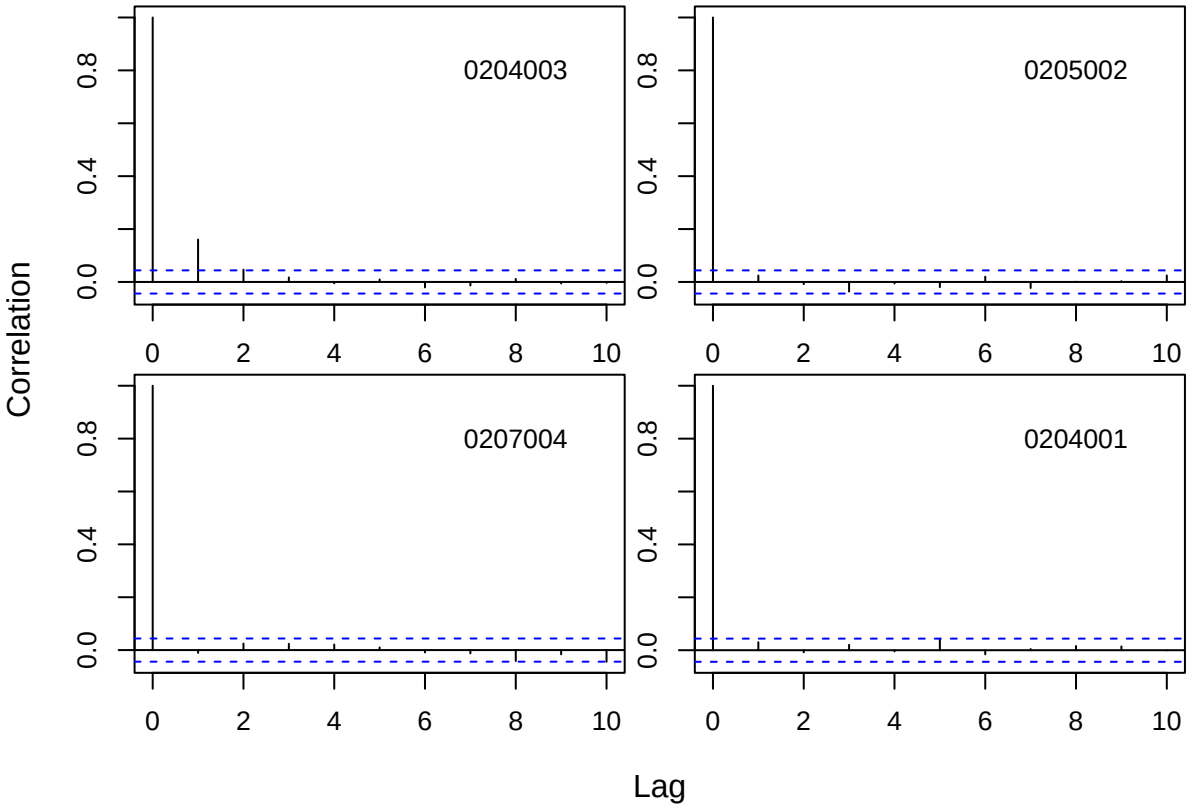


Figure 4.22: ACF plots for four select census block group medians.

# Chapter 5

# Multivariate Populations

In chapter 4, we focused on including additional data sources into our models for synthetic populations. Within this chapter, we focus on expanding the modeling effort to a multivariate $\mathbf{Y}$, using the same sources of information as in Chapter 4.

Population $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\}$, where $\mathbf{y}_i$ is $p \times 1,\ p > 1$

Data $\mathbf{x} = \{\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG}, \dots\}$

## 5.1 Methods

In Section 4.4, we introduced the idea of using two population variables. In that case, we let $\mathbf{y}^u$ denote an unknown variable of interest, while $\mathbf{y}^k$ represented a known variable of the population. Now, we will model multiple variables of interest $\mathbf{y}_1^u, \mathbf{y}_2^u, \dots$ and, potentially, multiple known variables as well.

Building off of how we modeled $\mathbf{y}^k$ in Section 4.4, we will jointly model additional variables of interest in much the same way. For example,

$$f(\mathbf{y}_2^u, \mathbf{y}_1^u, \mathbf{y}^k, \gamma, \phi, \dots) = f(\mathbf{y}_2^u | \mathbf{y}_1^u, \mathbf{y}^k, \gamma, \phi) f(\mathbf{y}_1^u | -) \tag{5.1}$$

$$\propto \mathrm{N}(\mathbf{y}_2^u | \gamma_0 \mathbf{1} + \gamma_1 \mathbf{y}_1^u + \gamma_2 \mathbf{y}^k, \mathrm{diag}(\phi)) f(\mathbf{y}_1^u | -) \tag{5.2}$$

would be one way of modeling an additional continuous variable of interest if we believe that it is correlated linearly with our primary variable of interest $\mathbf{y}_1^u$ and known variable $\mathbf{y}^k$. Just as we did in Section 4.4, we are forced to estimate $\gamma, \phi$ from our sample data. Since we have already seen one example of how to add continuous variables as we did in section 4.4, we will instead focus on categorical variables.

One way of including additional $\mathbf{y}^u$ variables that we will discuss is the conditional implementation which we used in 4.4, whereby additional variables are added one at a time using their conditional distributions given the variables which are already present in the model. For example, if

$$\pi(\mathbf{y}_1^u, \mathbf{y}^k, \mathbf{N}, \alpha | \mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})$$

represents our joint posterior before adding in $\mathbf{y}_2^u$, then we can include $\mathbf{y}_2^u$ by incorporating the distribution $f(\mathbf{y}_2^u|\cdot)$ into our posterior. The form of $f(\mathbf{y}_2^u|\cdot)$ will of course depend on what variables we believe are influential for $\mathbf{y}_2^u$, as well as the data type of $\mathbf{y}_2^u$.

## 5.2 Blacksburg Application

For our analysis of the Blacksburg data, we focus on a variable *couple type* calculated from the ACS PUMS data (US Census Bureau 2019c). This variable takes two levels: *single/non-romantic* (or 0), meaning the housing unit is occupied by either 1 person or multiple individuals that are not romantically involved, and *couple* (or 1), meaning the occupants are a married or unmarried couple.

We model this new variable $\mathbf{y}_2^u$ via logistic regression using $\mathbf{y}_1^u$ (income) and $\mathbf{y}^k$ (housing value). This logistic regression is visualized in Figure 5.1, where each predictor is displayed with the logistic regression prediction line, conditioned on the other predictor being equal

to the sample median value. Here we see that couple type has a strong relationship with household income, and a weak relationship with property value.

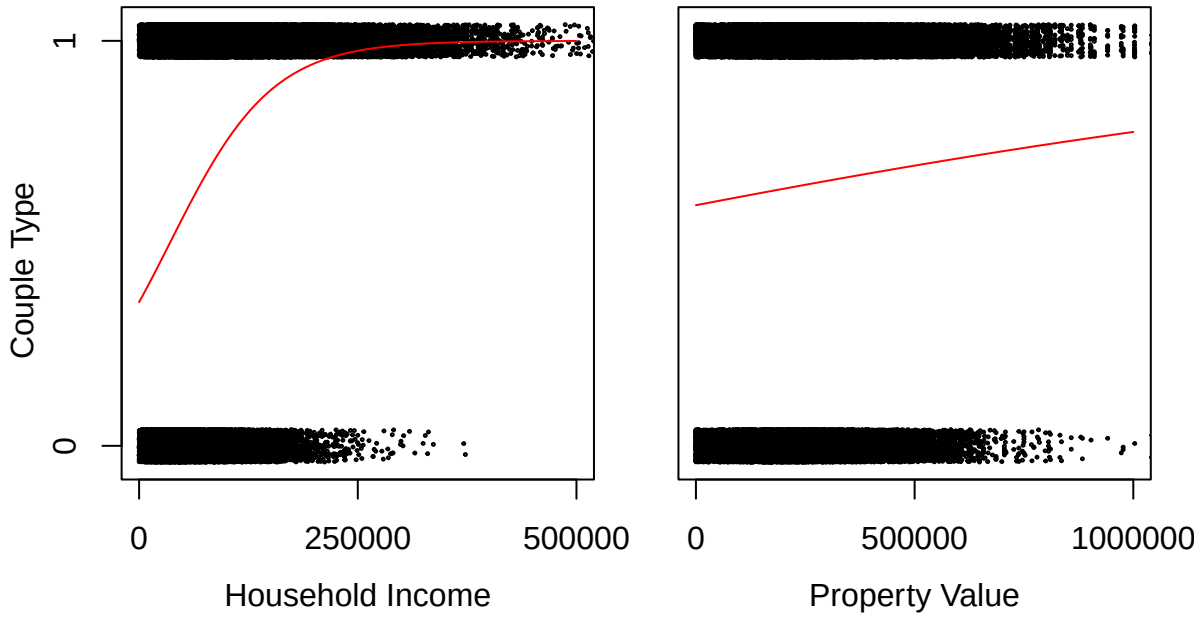


Figure 5.1: Logistic regression from sample data for couple type predicted with household income and housing value.

From the logistic regression performed on the sample values, we extract paramaters $\gamma_1$, $\gamma_2$, $\gamma_3$. We use these parameters to form a conditional prior for $\mathbf{y}_2^u$ of the form

$$\pi(\mathbf{y}_2^u|\mathbf{y}_1^u, \mathbf{y}^k, \vec{\gamma}) \propto \text{Bern}\left(\mathbf{y}_2^u \middle| \frac{\exp(\gamma_1\mathbf{1} + \gamma_2\mathbf{y}_1^2 + \gamma_3\mathbf{y}^k)}{1 + \exp(\gamma_1\mathbf{1} + \gamma_2\mathbf{y}_1^2 + \gamma_3\mathbf{y}^k)}\right),$$

and we can append our posterior from the Section 4.4 with this new prior in order to model our new variable. Doing so yields the joint posterior

$$\begin{aligned}\pi(\mathbf{y}_1^u, \mathbf{y}_2^u, \mathbf{y}^k, \mathbf{N}, \alpha|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG}) &\propto f(\mathbf{x}^{SRS}|\mathbf{y}) \times f(\mathbf{x}^{MD}|\mathbf{y}) \times \pi(\mathbf{y}_2^u|\mathbf{y}_1^u, \mathbf{y}^k, \vec{\gamma}) \\ &\times \pi(\mathbf{y}^k|\mathbf{y}^u, \vec{\beta}, \tau) \times \pi(\mathbf{y}_1^u, \mathbf{N}|\alpha) \times \pi(\alpha), \qquad (5.3)\end{aligned}$$

where

$$
\begin{aligned}
f(\mathbf{x}^{SRS}|\mathbf{y}) &= \text{Multi}(\mathbf{k}|n, \vec{\rho}), \\
f(\mathbf{x}^{MD}|\mathbf{y}) &= \text{N}(\mathbf{m}|\vec{\mu}, \text{diag}(\vec{\sigma})), \\
\pi(\mathbf{y}_2^u|\mathbf{y}_1^u, \mathbf{y}^k, \vec{\gamma}) &= \text{Bern}\left(\mathbf{y}_2^u \middle| \frac{\exp(\gamma_1 \mathbf{1} + \gamma_2 \mathbf{y}_1^2 + \gamma_3 \mathbf{y}^k)}{1 + \exp(\gamma_1 \mathbf{1} + \gamma_2 \mathbf{y}_1^2 + \gamma_3 \mathbf{y}^k)}\right), \\
\pi(\mathbf{y}^k|\mathbf{y}_1^u, \vec{\beta}, \tau) &= \text{N}(\mathbf{y}^k|\beta_0 \mathbf{1} + \beta_1 \mathbf{y}_1^u, \text{diag}(\tau)), \\
\pi(\mathbf{y}_1^u, \mathbf{N}|\alpha) &= g(\mathbf{y}_1^u|\mathbf{N}) \times \text{BD-Sp}(\mathbf{N}|G, \alpha), \\
\pi(\alpha) &= \text{Unif}(0, 1).
\end{aligned}
$$

Here, all of these pieces are defined just as they were in their respective sections within 4. We create an MCMC sampler to draw from the posterior (5.3) using the following pseudocode:

**INPUT**: $\mathbf{x}^{SRS}$ - random sample of $\mathbf{y}$

$\mathbf{x}^{MD}$ - medians (or means) with standard errors

$\mathbf{x}^{REG}$ - all information needed to construct regression model above

$\mathbf{B}$ - bins

$G$ - base distribution

$N$ - population size

$T$ - number of iterations

**OUTPUT**: $T$ realizations of $\mathbf{y}$

Initialize $\alpha$, $\mathbf{y}_1^u$, $\mathbf{y}_2^u$, and $\mathbf{N}$

**FOR** $i$ **IN** $1, \ldots, T$

Propose $\alpha^*$ from $p(\alpha^*|\alpha)$

Calculate $a = \min\left(1, \frac{\pi(\mathbf{y}_1^u, \mathbf{y}_2^u, \mathbf{y}^k, \mathbf{N}, \alpha^*|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})}{\pi(\mathbf{y}_1^u, \mathbf{y}_2^u, \mathbf{y}^k, \mathbf{N}, \alpha|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})} \times \frac{p(\alpha|\alpha^*)}{p(\alpha^*|\alpha)}\right)$

Set $\alpha = \alpha^*$ with probability $a$

**FOR** $j$ **IN** $1, \ldots, N$

Jointly propose $\{y_1^u\}_j^*, \mathbf{N}^*$ from $p(\{y_1^u\}_j^*, \mathbf{N}^*|\{y_1^u\}_j, \mathbf{N})$

Calculate $a = \min\left(1, \frac{f(\mathbf{y}_1^{u^*}, \mathbf{y}_2^u, \mathbf{y}^k, \mathbf{N}, \alpha|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})}{f(\mathbf{y}_1^u, \mathbf{y}_2^u, \mathbf{y}^k, \mathbf{N}, \alpha|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})} \times \frac{p(\{y_1^u\}_j, \mathbf{N}^*|\{y_1^u\}_j^*, \mathbf{N})}{p(\{y_1^u\}_j^*, \mathbf{N}^*|\{y_1^u\}_j, \mathbf{N})}\right)$

Set $\mathbf{y}_1^u = \mathbf{y}_1^{u^*}$ and $\mathbf{N} = \mathbf{N}^*$ with probability $a$

Propose $\{y_2^u\}_j^*$ from $p(\{y_2^u\}_j^*|\{y_2^u\}_j)$

Calculate $a = \min\left(1, \frac{f(\mathbf{y}_1^u, \mathbf{y}_2^{u^*}, \mathbf{y}^k, \mathbf{N}, \alpha|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})}{f(\mathbf{y}_1^u, \mathbf{y}_2^u, \mathbf{y}^k, \mathbf{N}, \alpha|\mathbf{x}^{SRS}, \mathbf{x}^{MD}, \mathbf{x}^{REG})} \times \frac{p(\{y_2^u\}_j|\{y_2^u\}_j^*)}{p(\{y_2^u\}_j^*|\{y_2^u\}_j)}\right)$

Set $\mathbf{y}_2^u = \mathbf{y}_2^{u^*}$ with probability $a$

**SAVE y**

**RETURN** $T$ realizations of $\mathbf{y}$

Exact code that implements this process can be found in Appendix 7.3.1.

Figure 5.2 shows the effect of adding in the additional variable on the census block group medians. Comparing this plot to Figure 4.13, we see that the medians are relatively unaffected by the inclusion of this second variable of interest.

Likewise, in Figure 5.3, we see the average distribution of incomes and bin proportions from the synthetic populations. Compared to Figure 4.14, these distributions look nearly identical to the version of the sampler that did not include couple type. This means that the marginal distribution of $\mathbf{y}_1^u$ is relatively unaffected by adding an additional population variable.

Our final visual comparison between this new implementation and our implementation without couple type is shown in Figure 5.4. This figure shows the relationship between income and housing value both within the sample and synthetic populations. Comparing this plot to

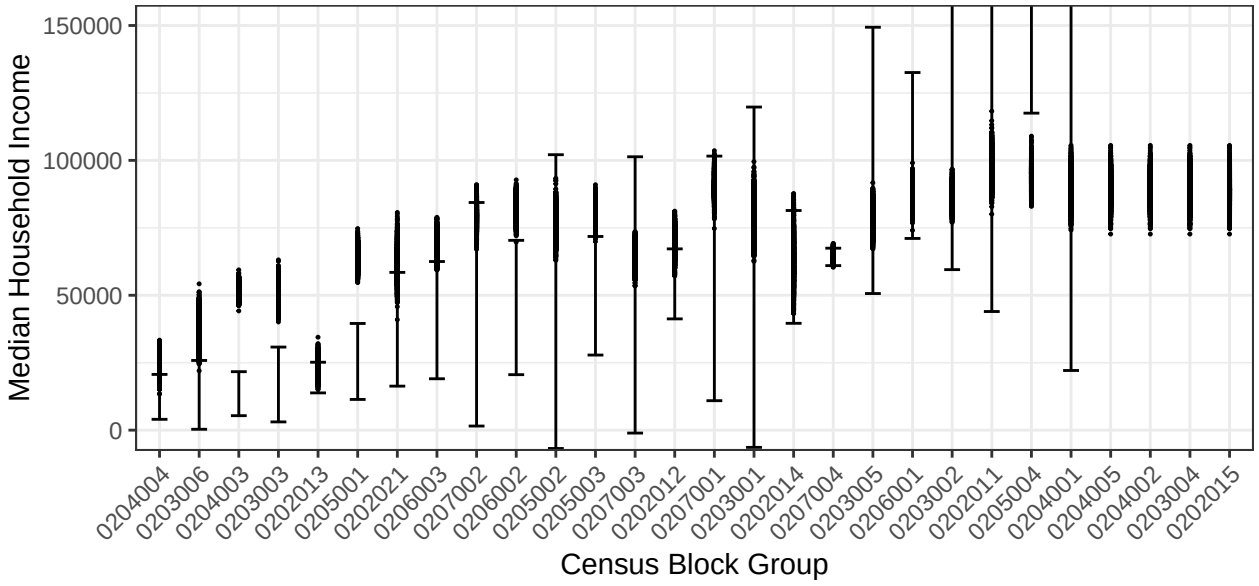


Figure 5.2: Medians with margin of error (90%), overlaid with medians from 2000 synthetic populations.

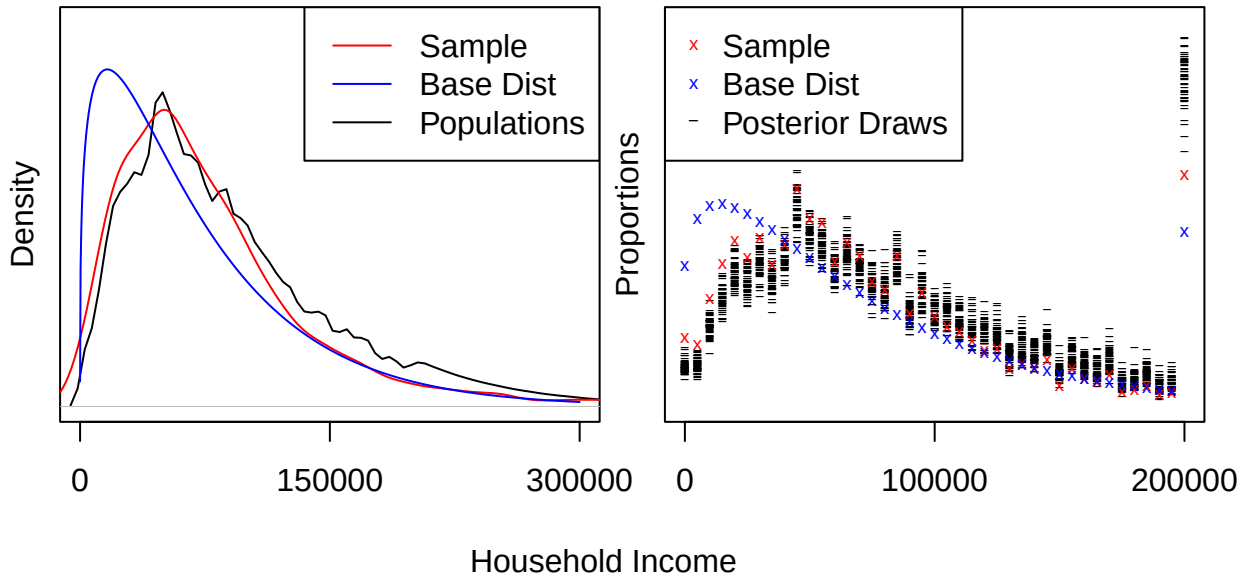


Figure 5.3: Comparison of density of incomes between sample and 2000 synthetic populations (left) and comparison of proportion of population within each bin for sample, base distribution, and 50 synthetic populations (right).

Figure 4.15 from the previous implementation, we see that the overall relationship between these two variables looks essentially the same.

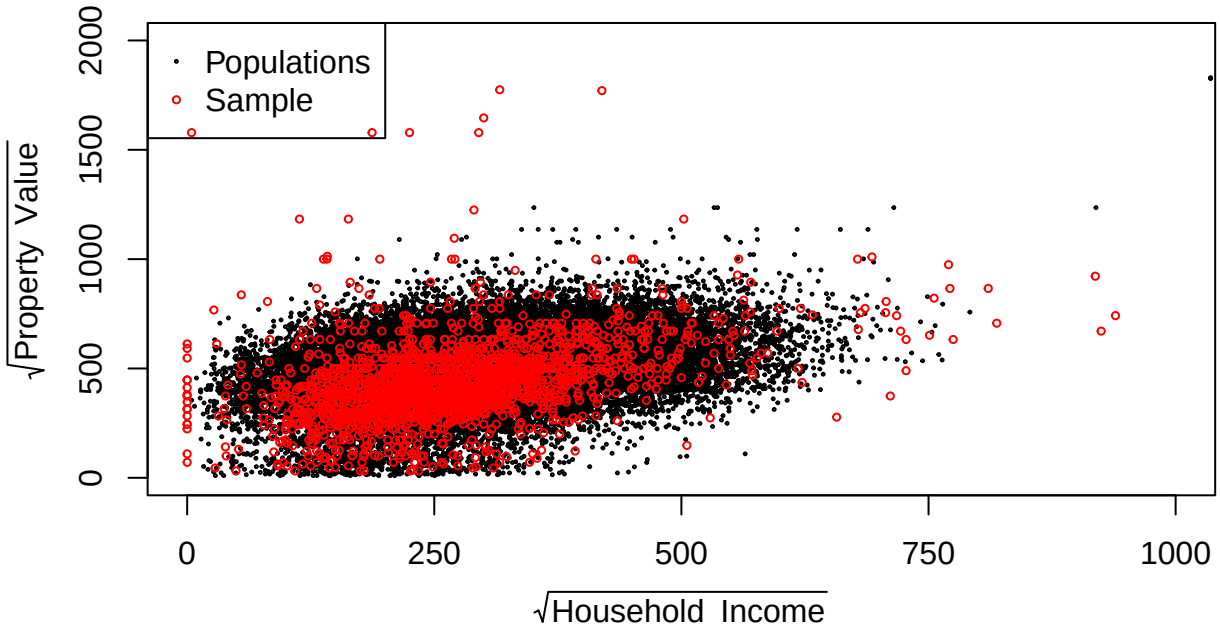


Figure 5.4: Relationship between the square roots of income and property value in the sample (red) and 5 synthetic poplations (black).

We have visually shown that adding in an additional variable $\mathbf{y}_2^u$ does not negatively impact our first variable of interest, income. Ideally, including additional variables could actually aid us in modeling $\mathbf{y}_1^u$; this is possible since $\mathbf{y}_1^u$ is connected to any other variables of interest we include via conditional priors. However, without any likelihoods to influence $\mathbf{y}_2^u$, the marginal distribution of $\mathbf{y}_1^u$ will remain unchanged.

Now, we investigate the relationship between our new variable of interest, couple type, with the other variables in the model in order to confirm that the relationship that we observed in the sample data (see Figure 5.1) is propagated into our synthetic populations.

To do so, we select several synthetic populations and calculate, for each separately, the logistic regression relationship between $\mathbf{y}_2^u$ and $(\mathbf{y}_1^u, \mathbf{y}^k)$. Additionally, we combine a large number of synthetic populations in order to get *average* behavior for this relationship. Figure 5.5 shows these relationships, where we can see that the *average* relationship across synthetic

populations matches the relationship within the sample data nearly perfectly. Additionally, this relationship is relatively constant across individual synthetic populations.

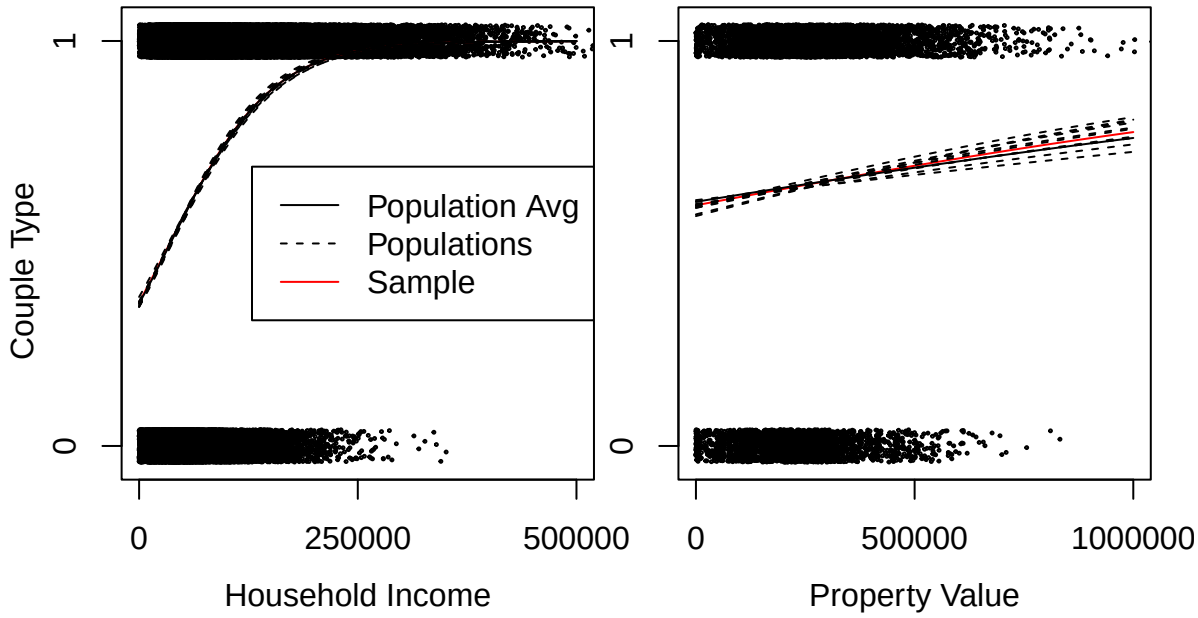


Figure 5.5: Logistic regression relationship between couple type and predictor variables household income and housing price for sample data, average population behavior, and 10 random synthetic populations.

As we did in Chapter 4, we will now overlay our results onto the map of Blacksburg. Figure 5.6 shows the posterior expectation of each household's income, where we the only change compared to Section 4.4 is that we incorporated an additional variable of interest, couple type, via an additional conditional prior. These results appear nearly identical to Figure 4.17.

Likewise, Figure 5.7 shows the summarized block group medians from the posterior. Again, these results are nearly identical to what we observed in Section 4.4.

Finally, we can visualize our new variable, couple type. Figure 5.8 shows one realization of the posterior for our new variable couple type.

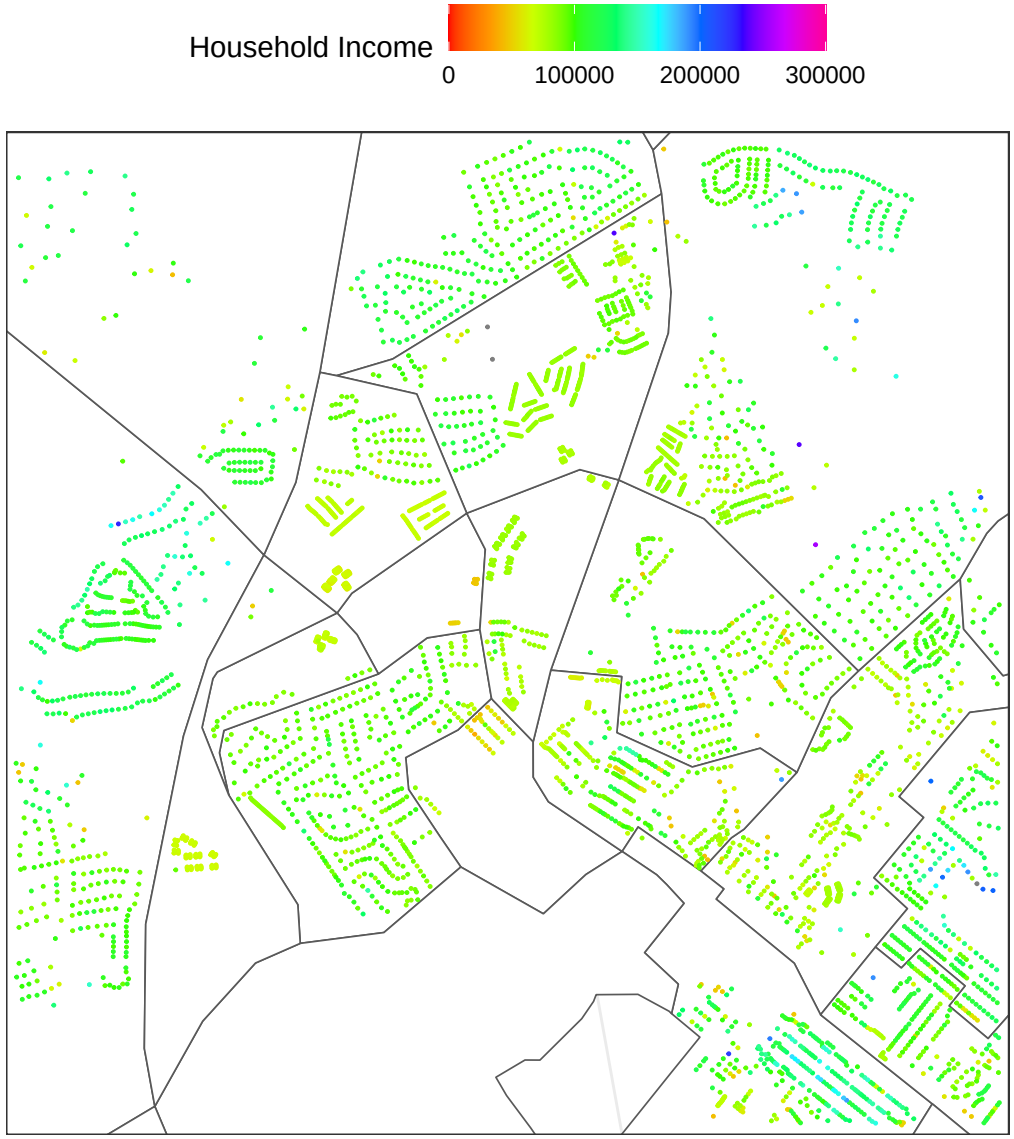


Figure 5.6: Posterior expectation for Blacksburg household incomes, using the SRS and median likelihoods, and updated prior using property value and couple type; Census block group borders shown.


Figure 5.7: Average of block group medians, using the posterior for $\mathbf{y}$ with the SRS and median likelihoods, and updated prior using regression information.

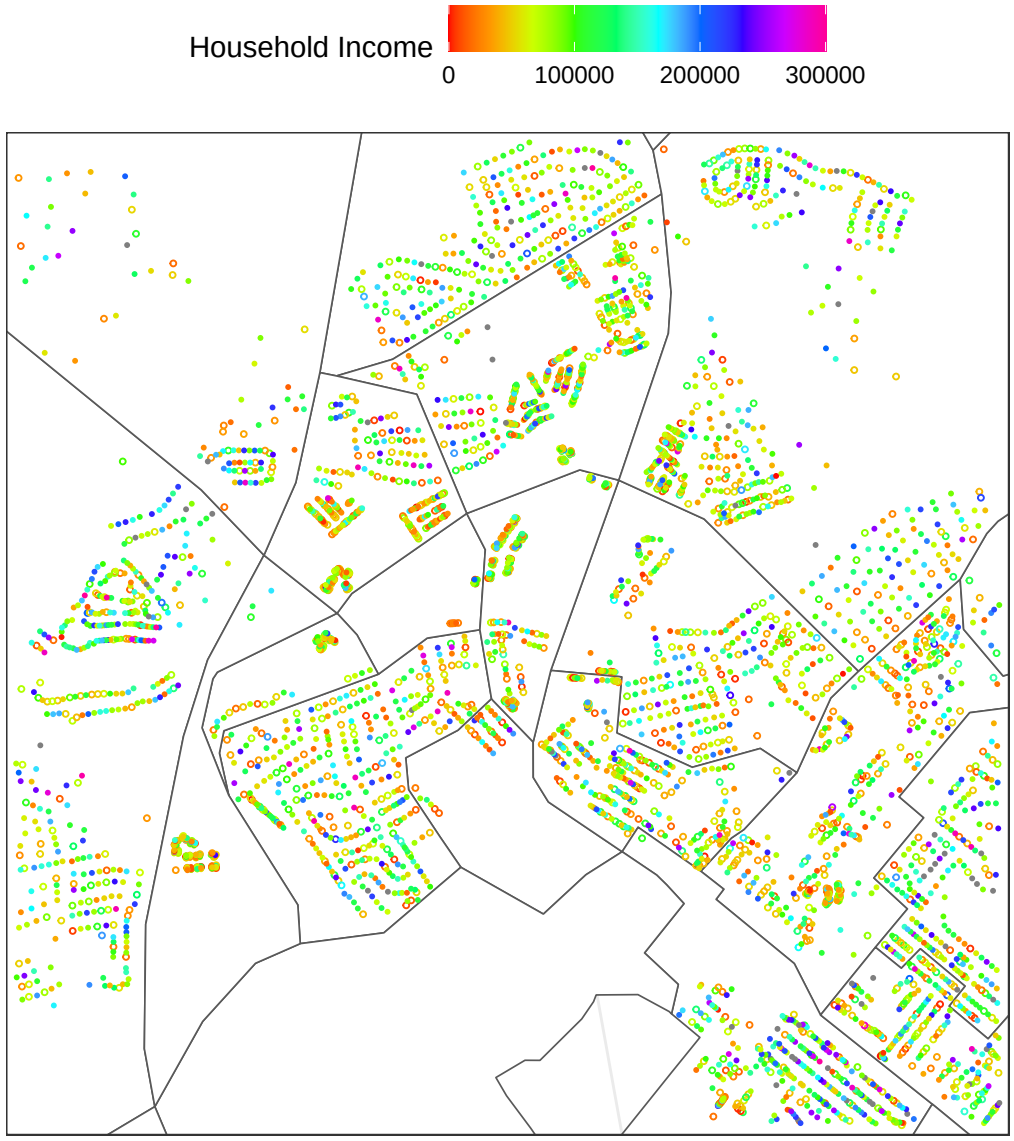


Figure 5.8: One realization of the posterior for household incomes, showing the couple type of each household: couples (filled in circles) and singles (empty circles).

# Chapter 6

# Conclusion

This chapter will conclude this dissertation by providing a brief summary of each chapter, specifically in relation to our goals and how each chapter enhances our ability to model the primary motivating example revisited throughout this document: modeling incomes of Blacksburg, VA. In addition, we outline some of the limitations and problems with the method we develop, as well as potential opportunities for improving upon our method via future work. Finally, we briefly discuss the role of ethics in statistics and machine learning as it applies to this dissertation.

## 6.1 Summary

Within Chapter 1, our introduction, we introduced our goal of modeling populations in order to perform inference on full populations. This runs counter to traditional statistics in which inference is performed parametrically. In addition, we establish the consideration of a population $\mathbf{y}$ as a random variable, allowing us to perform Bayesian inference on populations via Bayesian methods. In addition, we motivated our desire of synthetic population modeling in the scope of Agent-based Models (ABMs). Finally, we introduced the data available for Blacksburg, which we use throughout this thesis in our motivating examples.

In Chapter 2, our literature review, we explored existing methods of population synthesis; specifically, we highlighted methods commonly used by practitioners of ABMs, and discussed

how they fall into the three categories of population synthesis methods: Synthetic Reconstruction, Combinatorial Optimization, and Statistical Learning. We specifically focused on one method, Iterative Proportional Fitting (IPF), to highlight its advantages as well as shortcomings. IPF, and variations based on it, remains the most commonly used method of population synthesis, and thus functions as our primary point of comparison. We also performed IPF using some of the Blacksburg data, so that results of IPF for our motivating example can be easily compared against when we start synthesizing populations of Blacksburg in Chapter 3.

In Chapter 3, we began modeling populations, starting with simple examples and culminating in our first synthetic population for Blacksburg incomes at the end of Section 3.4. In the beginning of this chapter, we analyzed simple priors that practitioners could use, such as flat priors, and pointed out their flaws. We introduced the idea of hierarchical priors $f(\mathbf{y}|\theta)\pi(\theta)$ that, while allowing inference consistent with traditional parametric methods, restrict us to synthesizing i.i.d. population members. This led to the development of the Dirichlet Spacing prior in Section 3.3, which we created to solve the problem of dependence between population members. While this prior performs flawlessly, it is computationally expensive to sample from, which led us to the Binned Dirichlet Spacing prior. This prior, which we developed in Section 3.4, integrates the modeling performance that the Dirichlet Spacing prior grants us, in the form of an approximation which is computationally inexpensive in comparison.

Throughout Chapter 3, the focus of our modeling effort was on populations where our only data source was a single random sample. In Chapter 4, we built off of this foundation by incrementally adding in additional data sources one at a time. We did so by including likelihoods for each additional data source. Some data sources, such as the regression information highlighted in Section 4.4, required a slightly more nuanced approach, and we instead

modified our prior specification to include this data source. With the exception of Section 4.4, where we included information on a *known* secondary variable, this chapter's scope was limited to univariate populations.

While modeling a single variable provides a good starting point, in reality any synthetic populations needed for an ABM would require multiple variables. Within Chapter 5, we finally broached the subject of multivariate populations, and discussed one method of incorporating additional variables using a conditional prior specification. We applied this methodology to our Blacksburg example by including a secondary variable, couple type. We concluded by showing that including an additional variable in our sampler did not in any way hinder the modeling performance for our primary variable of interest, income.

## 6.2 Future Work

Throughout this thesis, we encountered a number of issues that afford us an opportunity to continue working on this subject. Perhaps the challenge with the most potential for improvement is the computational complexity of modeling the full Dirichlet Spacing prior that we developed. For small examples, this prior worked exactly how we wanted, but using it to model large populations proved impossible. There are two driving forces behind this problem. The first is that the calculation of the prior density scales linearly with $N$. While not unusual, this led to computational times being higher than desired for real populations such as the population of Blacksburg. The second issue, however, is more serious; at its core, the Dirichlet Spacing prior is a transformation of variables using the Dirichlet distribution and a base distribution. Often, when the Dirichlet concentration parameter $\alpha$ is small, sampling from a Dirichlet distribution will a vector containing values which are numerically zero. Theoretically, this should never happen, but in practice it does because of floating

point precision. These numerical zeroes cause chaos for the Dirichlet Spacing prior. During the development of this prior, we briefly explored avenues to get around this problem, but this effort was abandoned in favor of the Binned Dirichlet Spacing prior. While still based on the Dirichlet distribution, the binning makes this issue less likely to arise, to the point where the problem can be entirely ignored without issue. However, the switch from the full prior to the binned version did negatively effect our modeling performance, and if it were possible to fix these issues and return to the full Dirichlet Spacing prior, it would certainly be advantageous.

One other shortcoming was the lack of real-world applications for the methodology we developed. In the beginning, we hoped to be able to use our methods within an actual ABM in order to quantify uncertainty for ABMs. However, for several reasons, this goal was temporarily abandoned. If possible, it would be interesting to see how the methodology developed in this thesis functions as a way to quantify ABM uncertainty.

Another opportunity for future work comes out of spatial statistics. Briefly, we considered the idea of incorporating a spatial correlation structure to our modeling of real communities such as Blacksburg. Doing so would likely improve the efficacy of our resulting posterior. However, in the interest of time, this idea was pushed aside.

The final, and perhaps most important, opportunity to improve upon the methodology discussed here, is the possible development of a multivariate Dirichlet Spacing prior. In Chapter 5, our answer to the problem of multivariate populations was to include additional variables past the primary variable of interest via a conditional prior. This means that only the primary variable (for Blacksburg, income) was modeled with our Dirichlet Spacing prior. As additional variables are added, this solution does not scale well, computationally or otherwise. In philosophy, the word *telos* is used to describe the ultimate aim or purpose of a

person, or work of art. The *telos* of this thesis was, in the beginning, the development of a multivariate Dirichlet Spacing prior. It is unclear whether a multivariate form of the Dirichlet Spacing prior is possible, or what it would like if it indeed is possible. If research on this topic continues, it is the hope of the author that some day this *telos* will be fulfilled.

## 6.3 Ethical Considerations

This dissertation proposes methods to simulate *human* populations using *human* data for learning about *human*-related subjects and policies, such as the spread of disease and ways to slow it down via Agent-based Models (ABMs). Because of the human focus, it is important to consider the role of ethics in analysis, and how it relates to this work.

In recent years, there has been a considerable amount of concern over the ethical ramifications of various machine learning models introducing bias via data. This is primarily a direct result of the fast growth in data science over the last decade, without much oversight. As is often the case, corporate and legal policy have lagged behind this growth, resulting in a domain where there are comparatively very few rules or laws in place; it is quite easy to find, in mainstream media, examples of unintended bias being introduced to machine learning. In fact, virtually every major technology company has had to issue a statement apologizing for bias it introduced to the public within the past decade. For example, in

- 2016, Microsoft had to pull the plug on Tay, an AI chat bot, after it posted racist and inflammatory messages on Twitter (D. Lee 2016).
- 2017, Amazon had to scrap their recruiting tool that was favoring male candidates (Dastin 2018).
- 2018, as a response to Google getting bad press over an image classification model labeling black people as gorillas, chimpanzees or monkeys, the company opted to simply

remove those labels entirely instead of spending time and money on a real solution (Hern 2018).

- 2020, Twitter apologized for a new AI-powered image cropping tool seeming to prefer cropping black people out of images (Hern 2020).

As a response to these issues and more, the American Statistical Association published "Ethical Guidelines for Statistical Practice" (2022). One of the key takeaways from these guidelines is that significant effort is required to make sure that any statistical or machine learning model does not yield prejudiced results. There is no quick fix when a model is found to be giving unethical results. Instead, we must be conscious of the fact that data can contain prejudice, and actively attempt to make sure any models we create do not propagate that prejudice into statistical results.

While bias from agent-based models has fortunately stayed out of mainstream media, ABMs are certainly not immune to this issue. In the scope of this dissertation, creating synthetic populations for ABMs and other black-box models, we must be aware of any possible bias in sample data, and seek to remove that bias so that it is not propagated into synthetic populations. Allowing prejudice or bias into synthetic populations means that the results of any ABM could be prejudiced or biased themselves. While we did not include any social factors such as race, sex, ethnicity, etc. within our modeling examples, often these kinds of variables will be important for an ABM. If sample data on the population of interest is prejudiced (e.g., black people have lower property values because of prejudiced property assessments), then if we do nothing to fix the bias, our populations will reinforce this prejudice. Only an active effort from the statistician creating these populations can possibly mitigate this risk.

Another potential ethical risk is that of data privacy. While the examples in this dissertation are limited to using publicly available data, that is certainly not the case for all synthetic

populations needed for ABMs. Populations created via methods such as those described within this dissertation are synthetic, but that does not remove the possibility of uniquely identifying individuals. We modeled household incomes for real households that are shown on a map, and certainly we probably got quite close to the truth for several households. If we had modeled sensitive data such as HIPAA protected diagnoses, the possibility exists of correctly matching a diagnosis to a household or individual. This problem can potentially become more severe as multiple data sources are included. As statisticians, we must strive to make data available whenever possible; however, we must also exercise caution to protect confidential data. Again, there is no simple fix for this problem (short of simply never sharing data); the only way of making sure confidential data remains protected is to actively exercise caution and try to protect it.

# References


Bar-Gera, Hillel, Karthik C. Konduri, Bhargava Sana, Xin Ye, and Ram M. Pendyala. 2009. "Estimating Survey Weights with Multiple Constraints Using Entropy Optimization Methods." In.

Barthelemy, Johan, and Thomas Suesse. 2018. *Mipfp: Multidimensional Iterative Proportional Fitting and Alternative Models*. https://github.com/jojo-/mipfp.

Beckman, Richard J., Keith A. Baggerly, and Michael D. McKay. 1996. "Creating Synthetic Baseline Populations." *Transportation Research Part A: Policy and Practice* 30 (6): 415–29. https://doi.org/10.1016/0965-8564(96)00004-3.

Borysov, Stanislav S., Jeppe Rich, and Francisco C. Pereira. 2019. "How to Generate Micro-Agents? A Deep Generative Modeling Approach to Population Synthesis." *Transportation Research Part C: Emerging Technologies* 106: 73–97. https://doi.org/10.1016/j.trc.2019.07.006.

Casati, Daniele, Kirill Müller, Pieter J. Fourie, Alexander Erath, and Kay W. Axhausen. 2015. "Synthetic Population Generation by Combining a Hierarchical, Simulation-Based Approach with Reweighting by Generalized Raking." *Transportation Research Record* 2493 (1): 107–16. https://doi.org/10.3141/2493-12.

Chang, Winston, Joe Cheng, JJ Allaire, Carson Sievert, Barret Schloerke, Yihui Xie, Jeff Allen, Jonathan McPherson, Alan Dipert, and Barbara Borges. 2021. *Shiny: Web Application Framework for r*. https://shiny.rstudio.com/.

Crespi, Catherine M., and W. John Boscardin. 2009. "Bayesian model checking for multivariate outcome data." *Computational Statistics & Data Analysis* 53 (11): 3765–72. https://ideas.repec.org/a/eee/csdana/v53y2009i11p3765-3772.html.

Dastin, Jeffrey. 2018. "Amazon Scraps Secret AI Recruiting Tool That Showed Bias Against

Women." *Reuters*, October. https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G.

Deming, W. Edwards, and Frederick F. Stephan. 1940. "On a Least Squares Adjustment of a Sampled Frequency Table When the Expected Marginal Totals are Known." *The Annals of Mathematical Statistics* 11 (4): 427–44. https://doi.org/10.1214/aoms/1177731829.

Deville, Jean-Claude, Carl-Erik Särndal, and Olivier Sautory. 1993. "Generalized Raking Procedures in Survey Sampling." *Journal of the American Statistical Association* 88 (423): 1013–20. https://doi.org/10.1080/01621459.1993.10476369.

"Ethical Guidelines for Statistical Practice." 2022. *American Statistical Association*. https://www.amstat.org/your-career/ethical-guidelines-for-statistical-practice.

Eubank, S., C. Barrett, R. Beckman, K. Bisset, L. Durbeck, C. Kuhlman, B. Lewis, A. Marathe, M. Marathe, and P. Stretz. 2010. "Detail in Network Models of Epidemiology: Are We There Yet?" *Journal of Biological Dynamics* 4 (5): 446–55. https://doi.org/10.1080/17513751003778687.

Farooq, Bilal, Michel Bierlaire, Ricardo Hurtubia, and Gunnar Flötteröd. 2013. "Simulation Based Population Synthesis." *Transportation Research Part B: Methodological* 58: 243–63.

Fienberg, Stephen E. 2006. "When Did Bayesian Inference Become 'Bayesian'?" *Bayesian Analysis* 1 (1): 1–40.

Fienberg, Stephen E. 1970. "An Iterative Procedure for Estimation in Contingency Tables." *The Annals of Mathematical Statistics* 41 (3): 907–17. https://doi.org/10.1214/aoms/1177696968.

Gallagher, Shannon, Lee F Richardson, Samuel L Ventura, and William F Eddy. 2018. "SPEW: Synthetic Populations and Ecosystems of the World." *Journal of Computational and Graphical Statistics* 27 (4): 773–84.

Gelman, Andrew, John B Carlin, Hal S Stern, and Donald B Rubin. 1995. *Bayesian Data Analysis*. Chapman; Hall/CRC.

Gelman, Andrew, Xiao-Li Meng, and Hal Stern. 1996. "Posterior Predictive Assessment of Model Fitness via Realized Discrepancies." *Statistica Sinica* 6 (4): 733–60. http://www.jstor.org/stable/24306036.

Gibbons, Jean Dickinson et al. 1976. *Nonparametric Methods for Quantitative Analysis*. Holt, Rinehart; Winston.

Haberman, Shelby J. 1974. *The Analysis of Frequency Data*. University of Chicago Press.

Harland, Kirk, A. J. Heppenstall, Dianna Smith, and Mark Birkin. 2012. "Creating Realistic Synthetic Populations at Varying Spatial Scales: A Comparative Critique of Population Synthesis Techniques." *Journal of Artificial Societies and Social Simulation* 15 (February). https://doi.org/10.18564/jasss.1909.

Hermes, Kerstin, and Michael Poulsen. 2012. "A Review of Current Methods to Generate Synthetic Spatial Microdata Using Reweighting and Future Directions." *Computers, Environment and Urban Systems* 36 (4): 281–90. https://doi.org/10.1016/j.compenvurbsys.2012.03.005.

Hern, Alex. 2018. "Google's Solution to Accidental Algorithmic Racism: Ban Gorillas." *The Guardian*, January. https://www.theguardian.com/technology/2018/jan/12/google-racism-ban-gorilla-black-people.

———. 2020. "Twitter Apologises for 'Racist' Image-Cropping Algorithm." *The Guardian*, September. https://www.theguardian.com/technology/2020/sep/21/twitter-apologises-for-racist-image-cropping-algorithm.

Lee, Dave. 2016. "Tay: Microsoft Issues Apology over Racist Chatbot Fiasco." *BBC News*, March. https://www.bbc.com/news/technology-35902104.

Lee, Der-Horng, and Yingfei Fu. 2011. "Cross-Entropy Optimization Model for Population Synthesis in Activity-Based Microsimulation Models." *Transportation Research Record*

2255 (1): 20–27. https://doi.org/10.3141/2255-03.

Müller, Kirill, and Kay W Axhausen. 2010. "Population Synthesis for Microsimulation: State of the Art." *Arbeitsberichte Verkehrs-Und Raumplanung* 638.

———. 2012. "Multi-Level Fitting Algorithms for Population Synthesis." *Arbeitsberichte Verkehrs-Und Raumplanung* 821.

Müller, Kirill, and Kay W. Axhausen. 2011. "Hierarchical IPF: Generating a synthetic population for Switzerland." ERSA conference papers ersa11p305. European Regional Science Association. https://ideas.repec.org/p/wiw/wiwrsa/ersa11p305.html.

Rich, Jeppe, and Ismir Mulalic. 2012. "Generating Synthetic Baseline Populations from Register Data." *Transportation Research Part A: Policy and Practice* 46 (3): 467–79. https://doi.org/10.1016/j.tra.2011.11.002.

Rubin, Donald B. 1984. "Bayesianly Justifiable and Relevant Frequency Calculations for the Applied Statistician." *The Annals of Statistics* 12 (4): 1151–72. http://www.jstor.org/stable/2240995.

Saadi, Ismaïl, Ahmed Mustafa, Jacques Teller, Bilal Farooq, and Mario Cools. 2016. "Hidden Markov Model-based population synthesis." *Transportation Research Part B: Methodological* 90 (C): 1–21. https://doi.org/10.1016/j.trb.2016.04.007.

Stigler, Stephen M. 1986. *The History of Statistics: The Measurement of Uncertainty Before 1900*. Harvard University Press.

Sun, Lijun, and Alexander Erath. 2015. "A Bayesian Network Approach for Population Synthesis." *Transportation Research Part C: Emerging Technologies* 61: 49–62. https://doi.org/10.1016/j.trc.2015.10.010.

Templ, Matthias, Bernhard Meindl, Alexander Kowarik, and Olivier Dupriez. 2017. "Simulation of Synthetic Complex Data: The r Package simPop." *Journal of Statistical Software* 79 (10): 1–38. https://doi.org/10.18637/jss.v079.i10.

US Census Bureau. 2010. "2010 Census- PUMA Reference Map: New River Valley Planning

District Commission." https://www2.census.gov/geo/maps/dc10map/PUMA_RefMap/st51_va/puma5151040/DC10PUMA5151040_001.pdf.

———. 2019a. "American Community Survey 5-Year Data (2009 - 2021)." https://www.census.gov/programs-surveys/acs/data.html.

———. 2019b. "American Community Survey (ACS)." https://www.census.gov/programs-surveys/acs.

———. 2019c. "American Community Survey Public Use Microdata Sample (PUMS)." https://www.census.gov/programs-surveys/acs/microdata.html.

Virginia Tech Library Maps & GIS Division. 2019. "Town of Blacksburg GIS Data." https://sites.google.com/vt.edu/townofblacksburggisdata.

Xie, Yihui. 2015. *Dynamic Documents with R and Knitr*. 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC. http://yihui.name/knitr/.

———. 2016. *Bookdown: Authoring Books and Technical Documents with R Markdown*. Boca Raton, Florida: Chapman; Hall/CRC. https://bookdown.org/yihui/bookdown.

———. 2022. *Bookdown: Authoring Books and Technical Documents with r Markdown*. https://CRAN.R-project.org/package=bookdown.

Yaméogo, Boyam Fabrice, Pascal Gastineau, Pierre Hankach, and Pierre-Olivier Vandanjon. 2021. "Comparing Methods for Generating a Two-Layered Synthetic Population." *Transportation Research Record* 2675 (1): 136–47. https://doi.org/10.1177/0361198120964734.

Ye, Xin, Karthik C. Konduri, Ram M. Pendyala, Bhargava Sana, and Paul Waddell. 2009. "Methodology to Match Distributions of Both Household and Person Attributes in Generation of Synthetic Populations." In.

Zhang, Danqing, Junyu Cao, Sid Feygin, Dounan Tang, Zuo-Jun(Max) Shen, and Alexei Pozdnoukhov. 2019. "Connected Population Synthesis for Transportation Simulation." *Transportation Research Part C: Emerging Technologies* 103: 1–16. https://doi.org/10.1016/j.trc.2018.12.014.

Zhu, Yi, and Joseph Ferreira Jr. 2014. "Synthetic Population Generation at Disaggregated Spatial Scales for Land Use and Transportation Microsimulation." *Transportation Research Record* 2429 (1): 168–77.

# Chapter 7

# Appendix

This appendix is primarily used for R code that is not necessary to understand the content of the dissertation itself.

## 7.1 Algorithms - Hierarchical Priors

These four algorithms are used in Chapter 3.2, Subsections 3.2.1.1, 3.2.1.2, 3.2.2.1, and 3.2.2.2 respectively.

### 7.1.1 Binomial

```
popsim_binomial <- function(obs, N, samples = 1000, a = 0.5, b = 0.5) {

  logprior <- function(N, p, a, b) {
    -lchoose(N, N * p) + dbeta(p, a, b, log = TRUE)
  }

  logpost <- function(N, p, J, n, a, b) {
    dbinom(J, n, p, log = TRUE) + logprior(N, p, a, b)
  }

  J <- sum(obs)
  n <- length(obs)
```

```
13
14   out <- matrix(NA, nrow = samples + 1, ncol = N)
15   out[1, ] <- sample(c(0, 1), size = N, replace = TRUE,
16                      prob = c(1 - mean(obs), mean(obs)))
17
18   current_lp <- logpost(N, mean(out[1, ]), J, n, a, b)
19
20   for(i in 2:(samples + 1)) {
21     current <- out[i - 1, ]
22     for(k in 1:N) {
23       proposal <- current
24       proposal[k] <- as.integer(!current[k])
25       proposal_lp <- logpost(N, mean(proposal), J, n, a, b)
26
27       u <- runif(1)
28       if(!is.na(proposal_lp) & log(u) <= proposal_lp - current_lp) {
29         current <- proposal
30         current_lp <- proposal_lp
31       }
32     }
33     out[i, ] <- current
34   }
35   return(out[-1, ])
36 }
```

### 7.1.2 Multinomial

```
1 popsim_multinomial <- function(obs, N, samples = 1000, alpha = rep(0.5, length
    (unique(obs)))) {
2
3   logprior <- function(K, N, p, alpha) {
```

```
4     -lmnchoose(N, K) + ddirichlet(p, alpha, log = TRUE)
5   }
6
7   logpost <- function(K, N, J, n, alpha) {
8     p <- K / N
9     dmultinom(J, prob = p, log = TRUE) + logprior(K, N, p, alpha)
10  }
11
12  J <- tabulate(obs)
13  uniq <- sort(unique(obs))
14  classes <- max(uniq)
15  n <- length(obs)
16
17  out <- matrix(NA, nrow = samples + 1, ncol = N)
18  out[1, ] <- rep(obs, length.out = N)
19
20  current_lp <- logpost(tabulate(out[1, ], nbins = classes), N, J, n, alpha)
21
22  for(i in 2:(samples + 1)) {
23    current <- out[i - 1, ]
24    for(k in 1:N) {
25      proposal <- current
26      proposal[k] <- sample(uniq[-proposal[k]], 1)
27      proposal_lp <- logpost(tabulate(proposal, nbins = classes), N, J, n,
          alpha)
28
29      u <- runif(1)
30      if(!is.na(proposal_lp) & log(u) <= proposal_lp - current_lp) {
31        current <- proposal
32        current_lp <- proposal_lp
33      }
```

```
    }
    out[i, ] <- current
  }
  return(out[-1, ])
}
```

### 7.1.3 Hypergeometric

```
popsim_hypergeometric <- function(obs, N, samples = 1000, a = 0.5, b = 0.5) {
  logprior <- function(K, N, a, b) {
    lbeta(K + a, N - K + b) - lbeta(a, b)
  }

  logpost <- function(K, N, J, n, a, b) {
    dhyper(J, K, N - K, n, log = TRUE) + logprior(K, N, a, b)
  }

  J <- sum(obs)
  n <- length(obs)

  out <- matrix(NA, nrow = samples + 1, ncol = N)
  out[1, ] <- sample(c(0, 1), size = N, replace = TRUE, prob = c(1 - mean(obs)
      , mean(obs)))

  current_lp <- logpost(sum(out[1, ]), N, J, n, a, b)

  for(i in 2:(samples + 1)) {
    current <- out[i - 1, ]
    for(k in 1:N) {
      proposal <- current
      proposal[k] <- as.integer(!current[k])
```

```
23        proposal_lp <- logpost(sum(proposal), N, J, n, a, b)
24
25        u <- runif(1)
26        if(!is.na(proposal_lp) & log(u) <= proposal_lp - current_lp) {
27          current <- proposal
28          current_lp <- proposal_lp
29        }
30      }
31      out[i, ] <- current
32    }
33    return(out[-1, ])
34  }
```

### 7.1.4 Multivariate Hypergeometric

```
1  popsim_mvhypergeometric <- function(obs, N, samples = 1000, alpha = rep(0.5,
       length(unique(obs)))) {
2
3    logprior <- function(K, alpha) {
4      lmvbeta(K + alpha) - lmvbeta(alpha)
5    }
6
7    logpost <- function(K, J, n, alpha) {
8      dmvhyper(J, K, n, log = TRUE) + logprior(K, alpha)
9    }
10
11   J <- tabulate(obs)
12   uniq <- sort(unique(obs))
13   classes <- max(uniq)
14   n <- length(obs)
15
```

```
  out <- matrix(NA, nrow = samples + 1, ncol = N)
  out[1, ] <- rep(obs, length.out = N)

  current_lp <- logpost(tabulate(out[1, ], nbins = classes), J, n, alpha)

  for(i in 2:(samples + 1)) {
    current <- out[i - 1, ]
    for(k in 1:N) {
      proposal <- current
      proposal[k] <- sample(uniq[-proposal[k]], 1)
      proposal_lp <- logpost(tabulate(proposal, nbins = classes), J, n, alpha)

      u <- runif(1)
      if(!is.na(proposal_lp) & log(u) <= proposal_lp - current_lp) {
        current <- proposal
        current_lp <- proposal_lp
      }
    }
    out[i, ] <- current
  }
  return(out[-1, ])
}
```

## 7.2 Algorithms - Multiple Data Sources

These algorithms are used in Chapter 4, incrementally adding in additional sources of information.

### 7.2.1 Random Sample Data

```
bdsp.gamma.srs <- function(y.arr, xy.srs, params = list(), mcmc = list()) {

  shape <- params$shape
  scale <- params$scale
  bins <- params$bins
  alphaPrior <- params$alphaPrior

  start <- mcmc$start
  end <- mcmc$end
  thin <- mcmc$thin
  alphaWidth <- mcmc$alphaWidth

  N <- nrow(y.arr)
  n <- nrow(xy.srs)
  nk <- hist(xy.srs$Income, breaks = bins, plot = FALSE)$counts
  baseProbs <- diff(pgamma(bins, shape = shape, scale = scale))

  initModel <- lm(sqrtIncome ~ sqrtValue - 1, data = xy.srs)
  y.arr$sqrtIncome <- predict(initModel, newdata = data.frame(sqrtValue = y.
      arr$sqrtValue))
  y.arr$Income <- y.arr$sqrtIncome ** 2
  Nk <- hist(y.arr$Income, breaks = bins, plot = FALSE)$counts
  alpha <- mean(alphaPrior)

  y.stor <- matrix(y.arr$Income, nrow = (end - start + 1) / thin, ncol = N,
      byrow = TRUE)
  ySqrt.stor <- matrix(y.arr$sqrtIncome, nrow = (end - start + 1) / thin, ncol
      = N, byrow = TRUE)
  Nk.stor <- matrix(Nk, nrow = (end - start + 1) / thin, ncol = length(Nk),
      byrow = TRUE)
  alpha.stor <- rep(alpha, (end - start + 1) / thin)
```

```
28
29  y.acceptance <- 1
30  alpha.acceptance <- 1
31
32  lBinDirichletSp <- function(baseProbs, Nk, alpha) {
33    alphaVec <- alpha * Nk
34    alphaVec[c(1, length(Nk))] <- alpha * (Nk[c(1, length(Nk))] + 0.5)
35    logB <- lgamma(sum(alphaVec)) - sum(lgamma(alphaVec))
36    logK <- sum((alphaVec - 1) * log(baseProbs))
37    return(logB + logK)
38  }
39
40  alpha <- alpha[1]
41  y <- y.stor[1, ]
42  ySqrt <- ySqrt.stor[1, ]
43  Nk <- Nk.stor[1, ]
44  pV <- Nk / N
45
46  for(iter in 2:end) {
47
48    alphaProp <- runif(1, min = max(alphaPrior[1], alpha - alphaWidth), max =
        min(alphaPrior[2], alpha + alphaWidth))
49
50    yProp <- rgamma(N, shape = shape, scale = scale)
51    ySqrtProp <- sqrt(yProp)
52
53    alphaAccept <- lBinDirichletSp(baseProbs, Nk, alphaProp) -
54      lBinDirichletSp(baseProbs, Nk, alpha) +
55      dunif(alpha, min = max(alphaPrior[1], alphaProp - alphaWidth), max = min
          (alphaPrior[2], alphaProp + alphaWidth), log = TRUE) -
```

```
56        dunif(alphaProp, min = max(alphaPrior[1], alpha - alphaWidth), max = min
              (alphaPrior[2], alpha + alphaWidth), log = TRUE)
57
58      if(log(runif(1)) < alphaAccept) {
59        alpha <- alphaProp
60        alpha.acceptance <- 1 + alpha.acceptance
61      }
62
63      for(i in 1:N) {
64
65        yi <- y[i]
66        ySqrti <- ySqrt[i]
67        yiProp <- yProp[i]
68        ySqrtiProp <- ySqrtProp[i]
69        yiBin <- findInterval(yi, bins, all.inside = TRUE)
70        yiPropBin <- findInterval(yiProp, bins, all.inside = TRUE)
71
72        if(yiBin == yiPropBin) {
73
74          yiAccept <- 0
75
76          if(log(runif(1)) < yiAccept) {
77            yi <- yiProp
78            ySqrti <- ySqrtiProp
79            y.acceptance <- 1 + y.acceptance
80          }
81
82        } else {
83
84          NkProp <- Nk; NkProp[yiBin] <- NkProp[yiBin] - 1; NkProp[yiPropBin] <-
                NkProp[yiPropBin] + 1
```

```
85          pVProp <- pV; pVProp[yiBin] <- (pVProp[yiBin] - (1/ N)); pVProp[
                yiPropBin] <- (pVProp[yiPropBin] + (1 / N))
86
87          lTrans <- log(NkProp[yiPropBin]) -
88            log(Nk[yiBin])
89
90          lPriorBins <- lBinDirichletSp(baseProbs, NkProp, alpha) -
91            lBinDirichletSp(baseProbs, Nk, alpha)
92
93          lPriorBase <- log(baseProbs[yiBin]) -
94            log(baseProbs[yiPropBin])
95
96          lMultiLike <- nk[yiBin] * (log(pVProp[yiBin]) - log(pV[yiBin])) +
97            nk[yiPropBin] * (log(pVProp[yiPropBin]) - log(pV[yiPropBin]))
98
99          yiAccept <- lTrans + lPriorBins + lPriorBase + lMultiLike
100
101         if(log(runif(1)) < yiAccept) {
102           yi <- yiProp
103           ySqrti <- ySqrtiProp
104           Nk <- NkProp
105           pV <- pVProp
106           y.acceptance <- 1 + y.acceptance
107         }
108
109       }
110
111       y[i] <- yi
112       ySqrt[i] <- ySqrti
113
114     }
```

```
115
116     if(iter >= start & iter %% thin == 0) {
117       y.stor[(iter - start + 1) / thin, ] <- y
118       ySqrt.stor[(iter - start + 1) / thin, ] <- ySqrt
119       Nk.stor[(iter - start + 1) / thin, ] <- Nk
120       alpha.stor[(iter - start + 1) / thin] <- alpha
121     }
122
123   }
124
125   alpha.acceptance <- alpha.acceptance / end
126   y.acceptance <- y.acceptance / (N * end)
127
128   out <- list(
129     y = y.stor,
130     ySqrt = ySqrt.stor,
131     Nk = Nk.stor,
132     alpha = alpha.stor,
133     acceptance = c(captionpos"captionposalphacaptionpos" = alpha.acceptance,
          captionpos"captionposycaptionpos" = y.acceptance)
134   )
135
136   return(out)
137
138 }
```

### 7.2.2 B.G. Median Data

```
1 bdsp.gamma.srs.med <- function(y.arr, xy.srs, medians, params = list(), mcmc =
      list()) {
2
```

```
3   shape <- params$shape
4   scale <- params$scale
5   bins <- params$bins
6   alphaPrior <- params$alphaPrior
7
8   start <- mcmc$start
9   end <- mcmc$end
10  thin <- mcmc$thin
11  alphaWidth <- mcmc$alphaWidth
12
13  N <- nrow(y.arr)
14  n <- nrow(xy.srs)
15  nk <- hist(xy.srs$Income, breaks = bins, plot = FALSE)$counts
16  baseProbs <- diff(pgamma(bins, shape = shape, scale = scale))
17
18  initModel <- lm(sqrtIncome ~ sqrtValue - 1, data = xy.srs)
19  y.arr$sqrtIncome <- predict(initModel, newdata = data.frame(sqrtValue = y.
        arr$sqrtValue))
20  y.arr$Income <- y.arr$sqrtIncome ** 2
21  Nk <- hist(y.arr$Income, breaks = bins, plot = FALSE)$counts
22  mV <- as.numeric(sapply(split(y.arr$Income, y.arr$Code), median))
23  alpha <- mean(alphaPrior)
24
25  y.stor <- matrix(y.arr$Income, nrow = (end - start + 1) / thin, ncol = N,
        byrow = TRUE)
26  ySqrt.stor <- matrix(y.arr$sqrtIncome, nrow = (end - start + 1) / thin, ncol
        = N, byrow = TRUE)
27  Nk.stor <- matrix(Nk, nrow = (end - start + 1) / thin, ncol = length(Nk),
        byrow = TRUE)
28  mV.stor <- matrix(mV, nrow = (end - start + 1) / thin, ncol = nrow(medians),
        byrow = TRUE)
```

```
29  alpha.stor <- rep(alpha, (end - start + 1) / thin)
30
31  y.acceptance <- 1
32  alpha.acceptance <- 1
33
34  lBinDirichletSp <- function(baseProbs, Nk, alpha) {
35    alphaVec <- alpha * Nk
36    alphaVec[c(1, length(Nk))] <- alpha * (Nk[c(1, length(Nk))] + 0.5)
37    logB <- lgamma(sum(alphaVec)) - sum(lgamma(alphaVec))
38    logK <- sum((alphaVec - 1) * log(baseProbs))
39    return(logB + logK)
40  }
41
42  medianSeq <- list()
43  medianInds <- c(0, cumsum(as.numeric(table(y.arr$Code))))
44  for(i in 1:nrow(medians)) {
45    medianSeq[[i]] <- seq((medianInds + 1)[i], medianInds[i + 1], by = 1)
46  }
47
48  alpha <- alpha[1]
49  y <- y.stor[1, ]
50  ySqrt <- ySqrt.stor[1, ]
51  Nk <- Nk.stor[1, ]
52  pV <- Nk / N
53  mV <- mV.stor[1, ]
54
55  for(iter in 2:end) {
56
57    alphaProp <- runif(1, min = max(alphaPrior[1], alpha - alphaWidth), max =
        min(alphaPrior[2], alpha + alphaWidth))
58
```

```
59      yProp <- rgamma(N, shape = shape, scale = scale)
60      ySqrtProp <- sqrt(yProp)
61
62      alphaAccept <- lBinDirichletSp(baseProbs, Nk, alphaProp) -
63        lBinDirichletSp(baseProbs, Nk, alpha) +
64        dunif(alpha, min = max(alphaPrior[1], alphaProp - alphaWidth), max = min
              (alphaPrior[2], alphaProp + alphaWidth), log = TRUE) -
65        dunif(alphaProp, min = max(alphaPrior[1], alpha - alphaWidth), max = min
              (alphaPrior[2], alpha + alphaWidth), log = TRUE)
66
67      if(log(runif(1)) < alphaAccept) {
68        alpha <- alphaProp
69        alpha.acceptance <- 1 + alpha.acceptance
70      }
71
72      for(i in 1:N) {
73
74        yi <- y[i]
75        ySqrti <- ySqrt[i]
76        yiProp <- yProp[i]
77        ySqrtiProp <- ySqrtProp[i]
78        yiBin <- findInterval(yi, bins, all.inside = TRUE)
79        yiPropBin <- findInterval(yiProp, bins, all.inside = TRUE)
80
81        mVind <- findInterval(i, medianInds, left.open = TRUE)
82        if(is.na(medians$Estimate[mVind])) {
83          lMedianLike <- 0
84        } else {
85          mVi <- mV[mVind]
86          mViProp <- median(replace(y, i, yiProp)[medianSeq[[mVind]]])
87          if(mVi == mViProp) {
```

```
88            lMedianLike <- 0
89          } else {
90            lMedianLike <- dnorm(mViProp, medians$Estimate[mVind], medians$SE[
                  mVind], log = TRUE) -
91                          dnorm(mVi, medians$Estimate[mVind], medians$SE[mVind
                                ], log = TRUE)
92          }
93        }
94
95        if(yiBin == yiPropBin) {
96
97          yiAccept <- lMedianLike
98
99          if(log(runif(1)) < yiAccept) {
100           yi <- yiProp
101           ySqrti <- ySqrtiProp
102           mV[mVind] <- mViProp
103           y.acceptance <- 1 + y.acceptance
104         }
105
106       } else {
107
108         NkProp <- Nk; NkProp[yiBin] <- NkProp[yiBin] - 1; NkProp[yiPropBin] <-
                NkProp[yiPropBin] + 1
109         pVProp <- pV; pVProp[yiBin] <- (pVProp[yiBin] - (1/ N)); pVProp[
              yiPropBin] <- (pVProp[yiPropBin] + (1 / N))
110
111         lTrans <- log(NkProp[yiPropBin]) -
112           log(Nk[yiBin])
113
114         lPriorBins <- lBinDirichletSp(baseProbs, NkProp, alpha) -
```

```
115           lBinDirichletSp(baseProbs, Nk, alpha)
116
117         lPriorBase <- log(baseProbs[yiBin]) -
118           log(baseProbs[yiPropBin])
119
120         lMultiLike <- nk[yiBin] * (log(pVProp[yiBin]) - log(pV[yiBin])) +
121           nk[yiPropBin] * (log(pVProp[yiPropBin]) - log(pV[yiPropBin]))
122
123         yiAccept <- lTrans + lPriorBins + lPriorBase + lMultiLike +
              lMedianLike
124
125         if(log(runif(1)) < yiAccept) {
126           yi <- yiProp
127           ySqrti <- ySqrtiProp
128           Nk <- NkProp
129           pV <- pVProp
130           mV[mVind] <- mViProp
131           y.acceptance <- 1 + y.acceptance
132         }
133
134       }
135
136       y[i] <- yi
137       ySqrt[i] <- ySqrti
138
139     }
140
141     if(iter >= start & iter %% thin == 0) {
142       y.stor[(iter - start + 1) / thin, ] <- y
143       ySqrt.stor[(iter - start + 1) / thin, ] <- ySqrt
144       Nk.stor[(iter - start + 1) / thin, ] <- Nk
```

```
145         mV.stor[(iter - start + 1) / thin, ] <- mV
146         alpha.stor[(iter - start + 1) / thin] <- alpha
147       }
148
149     }
150
151     alpha.acceptance <- alpha.acceptance / end
152     y.acceptance <- y.acceptance / (N * end)
153
154     out <- list(
155       y = y.stor,
156       ySqrt = ySqrt.stor,
157       Nk = Nk.stor,
158       mV = mV.stor,
159       alpha = alpha.stor,
160       acceptance = c(captionpos"captionposalphacaptionpos" = alpha.acceptance,
              captionpos"captionposycaptionpos" = y.acceptance)
161     )
162
163     return(out)
164
165 }
```

### 7.2.3 Regression Data

```
1 bdsp.gamma.srs.med.reg <- function(y.arr, xy.srs, medians, params = list(),
      mcmc = list()) {
2
3     shape <- params$shape
4     scale <- params$scale
5     bins <- params$bins
```

```
6   alphaPrior <- params$alphaPrior
7
8   start <- mcmc$start
9   end <- mcmc$end
10  thin <- mcmc$thin
11  alphaWidth <- mcmc$alphaWidth
12
13  N <- nrow(y.arr)
14  n <- nrow(xy.srs)
15  nk <- hist(xy.srs$Income, breaks = bins, plot = FALSE)$counts
16  baseProbs <- diff(pgamma(bins, shape = shape, scale = scale))
17
18  sqrtMod <- lm(sqrtValue ~ sqrtIncome, data = xy.srs)
19  beta <- as.numeric(coef(sqrtMod))
20  sigma <- as.numeric(summary(sqrtMod)$sigma)
21
22  initModel <- lm(sqrtIncome ~ sqrtValue - 1, data = xy.srs)
23  y.arr$sqrtIncome <- predict(initModel, newdata = data.frame(sqrtValue = y.
        arr$sqrtValue))
24  y.arr$Income <- y.arr$sqrtIncome ** 2
25  Nk <- hist(y.arr$Income, breaks = bins, plot = FALSE)$counts
26  mV <- as.numeric(sapply(split(y.arr$Income, y.arr$Code), median))
27  alpha <- mean(alphaPrior)
28
29  y.stor <- matrix(y.arr$Income, nrow = (end - start + 1) / thin, ncol = N,
        byrow = TRUE)
30  ySqrt.stor <- matrix(y.arr$sqrtIncome, nrow = (end - start + 1) / thin, ncol
        = N, byrow = TRUE)
31  Nk.stor <- matrix(Nk, nrow = (end - start + 1) / thin, ncol = length(Nk),
        byrow = TRUE)
```

```
32  mV.stor <- matrix(mV, nrow = (end - start + 1) / thin, ncol = nrow(medians),
        byrow = TRUE)
33  alpha.stor <- rep(alpha, (end - start + 1) / thin)
34
35  y.acceptance <- 1
36  alpha.acceptance <- 1
37
38  lBinDirichletSp <- function(baseProbs, Nk, alpha) {
39    alphaVec <- alpha * Nk
40    alphaVec[c(1, length(Nk))] <- alpha * (Nk[c(1, length(Nk))] + 0.5)
41    logB <- lgamma(sum(alphaVec)) - sum(lgamma(alphaVec))
42    logK <- sum((alphaVec - 1) * log(baseProbs))
43    return(logB + logK)
44  }
45
46  medianSeq <- list()
47  medianInds <- c(0, cumsum(as.numeric(table(y.arr$Code))))
48  for(i in 1:nrow(medians)) {
49    medianSeq[[i]] <- seq((medianInds + 1)[i], medianInds[i + 1], by = 1)
50  }
51
52  alpha <- alpha[1]
53  y <- y.stor[1, ]
54  ySqrt <- ySqrt.stor[1, ]
55  Nk <- Nk.stor[1, ]
56  pV <- Nk / N
57  mV <- mV.stor[1, ]
58
59  for(iter in 2:end) {
60
```

```
61      alphaProp <- runif(1, min = max(alphaPrior[1], alpha - alphaWidth), max =
            min(alphaPrior[2], alpha + alphaWidth))
62
63      yProp <- rgamma(N, shape = shape, scale = scale)
64      ySqrtProp <- sqrt(yProp)
65
66      alphaAccept <- lBinDirichletSp(baseProbs, Nk, alphaProp) -
67        lBinDirichletSp(baseProbs, Nk, alpha) +
68        dunif(alpha, min = max(alphaPrior[1], alphaProp - alphaWidth), max = min
              (alphaPrior[2], alphaProp + alphaWidth), log = TRUE) -
69        dunif(alphaProp, min = max(alphaPrior[1], alpha - alphaWidth), max = min
              (alphaPrior[2], alpha + alphaWidth), log = TRUE)
70
71      if(log(runif(1)) < alphaAccept) {
72        alpha <- alphaProp
73        alpha.acceptance <- 1 + alpha.acceptance
74      }
75
76      for(i in 1:N) {
77
78        yi <- y[i]
79        ySqrti <- ySqrt[i]
80        yiProp <- yProp[i]
81        ySqrtiProp <- ySqrtProp[i]
82        yiBin <- findInterval(yi, bins, all.inside = TRUE)
83        yiPropBin <- findInterval(yiProp, bins, all.inside = TRUE)
84
85        mVind <- findInterval(i, medianInds, left.open = TRUE)
86        if(is.na(medians$Estimate[mVind])) {
87          lMedianLike <- 0
88        } else {
```

```
89          mVi <- mV[mVind]
90          mViProp <- median(replace(y, i, yiProp)[medianSeq[[mVind]]])
91          if(mVi == mViProp) {
92            lMedianLike <- 0
93          } else {
94            lMedianLike <- dnorm(mViProp, medians$Estimate[mVind], medians$SE[
                mVind], log = TRUE) -
95                          dnorm(mVi, medians$Estimate[mVind], medians$SE[mVind
                              ], log = TRUE)
96          }
97        }
98
99        lPriorReg <- dnorm(y.arr$sqrtValue[i], beta[1] + beta[2] * ySqrtiProp,
            sigma, log = TRUE) -
100                 dnorm(y.arr$sqrtValue[i], beta[1] + beta[2] * ySqrti, sigma
                        , log = TRUE)
101
102       if(yiBin == yiPropBin) {
103
104         yiAccept <- lPriorReg + lMedianLike
105
106         if(log(runif(1)) < yiAccept) {
107           yi <- yiProp
108           ySqrti <- ySqrtiProp
109           mV[mVind] <- mViProp
110           y.acceptance <- 1 + y.acceptance
111         }
112
113       } else {
114
```

```
115         NkProp <- Nk; NkProp[yiBin] <- NkProp[yiBin] - 1; NkProp[yiPropBin] <-
                NkProp[yiPropBin] + 1
116         pVProp <- pV; pVProp[yiBin] <- (pVProp[yiBin] - (1/ N)); pVProp[
                yiPropBin] <- (pVProp[yiPropBin] + (1 / N))
117
118         lTrans <- log(NkProp[yiPropBin]) -
119           log(Nk[yiBin])
120
121         lPriorBins <- lBinDirichletSp(baseProbs, NkProp, alpha) -
122           lBinDirichletSp(baseProbs, Nk, alpha)
123
124         lPriorBase <- log(baseProbs[yiBin]) -
125           log(baseProbs[yiPropBin])
126
127         lMultiLike <- nk[yiBin] * (log(pVProp[yiBin]) - log(pV[yiBin])) +
128           nk[yiPropBin] * (log(pVProp[yiPropBin]) - log(pV[yiPropBin]))
129
130         yiAccept <- lTrans + lPriorBins + lPriorBase + lPriorReg + lMultiLike
                + lMedianLike
131
132         if(log(runif(1)) < yiAccept) {
133           yi <- yiProp
134           ySqrti <- ySqrtiProp
135           Nk <- NkProp
136           pV <- pVProp
137           mV[mVind] <- mViProp
138           y.acceptance <- 1 + y.acceptance
139         }
140
141       }
142
```

```
143         y[i] <- yi
144         ySqrt[i] <- ySqrti
145
146       }
147
148       if(iter >= start & iter %% thin == 0) {
149         y.stor[(iter - start + 1) / thin, ] <- y
150         ySqrt.stor[(iter - start + 1) / thin, ] <- ySqrt
151         Nk.stor[(iter - start + 1) / thin, ] <- Nk
152         mV.stor[(iter - start + 1) / thin, ] <- mV
153         alpha.stor[(iter - start + 1) / thin] <- alpha
154       }
155
156     }
157
158     alpha.acceptance <- alpha.acceptance / end
159     y.acceptance <- y.acceptance / (N * end)
160
161     out <- list(
162       y = y.stor,
163       ySqrt = ySqrt.stor,
164       Nk = Nk.stor,
165       mV = mV.stor,
166       alpha = alpha.stor,
167       acceptance = c(captionpos"captionposalphacaptionpos" = alpha.acceptance,
              captionpos"captionposycaptionpos" = y.acceptance)
168     )
169
170     return(out)
171
172 }
```

## 7.3 Algorithms - Multivariate Populations

This algorithm is used in Chapter 5, where we include an additional binary variable into the population.

### 7.3.1 Additional Binary Variable

```
bdsp.gamma.srs.med.reg.logit <- function(y.arr, xy.srs, medians, params = list
    (), mcmc = list()) {

  shape <- params$shape
  scale <- params$scale
  bins <- params$bins
  alphaPrior <- params$alphaPrior

  start <- mcmc$start
  end <- mcmc$end
  thin <- mcmc$thin
  alphaWidth <- mcmc$alphaWidth

  N <- nrow(y.arr)
  n <- nrow(xy.srs)
  nk <- hist(xy.srs$Income, breaks = bins, plot = FALSE)$counts
  baseProbs <- diff(pgamma(bins, shape = shape, scale = scale))

  sqrtMod <- lm(sqrtValue ~ sqrtIncome, data = xy.srs)
  beta <- as.numeric(coef(sqrtMod))
  sigma <- as.numeric(summary(sqrtMod)$sigma)

```

```
22  initModel <- lm(sqrtIncome ~ sqrtValue - 1, data = xy.srs)
23  y.arr$sqrtIncome <- predict(initModel, newdata = data.frame(sqrtValue = y.
        arr$sqrtValue))
24  y.arr$Income <- y.arr$sqrtIncome ** 2
25  Nk <- hist(y.arr$Income, breaks = bins, plot = FALSE)$counts
26  mV <- as.numeric(sapply(split(y.arr$Income, y.arr$Code), median))
27  alpha <- mean(alphaPrior)
28
29  logMod <- glm(CPLTL ~ Value + Income, data = xy.srs, family = binomial(link
        = captionpos"captionposlogitcaptionpos"))
30  beta2 <- as.numeric(coef(logMod))
31  y.arr$CPLTL <- as.logical(round(predict(object = logMod, newdata = y.arr,
        type = captionpos"captionposresponsecaptionpos")))
32
33  y.stor <- matrix(y.arr$Income, nrow = (end - start + 1) / thin, ncol = N,
        byrow = TRUE)
34  y2.stor <- matrix(y.arr$CPLTL, nrow = (end - start + 1) / thin, ncol = N,
        byrow = TRUE)
35  ySqrt.stor <- matrix(y.arr$sqrtIncome, nrow = (end - start + 1) / thin, ncol
        = N, byrow = TRUE)
36  Nk.stor <- matrix(Nk, nrow = (end - start + 1) / thin, ncol = length(Nk),
        byrow = TRUE)
37  mV.stor <- matrix(mV, nrow = (end - start + 1) / thin, ncol = nrow(medians),
         byrow = TRUE)
38  alpha.stor <- rep(alpha, (end - start + 1) / thin)
39
40  y.acceptance <- 1
41  y2.acceptance <- 1
42  alpha.acceptance <- 1
43
44  lBinDirichletSp <- function(baseProbs, Nk, alpha) {
```

```
45      alphaVec <- alpha * Nk
46      alphaVec[c(1, length(Nk))] <- alpha * (Nk[c(1, length(Nk))] + 0.5)
47      logB <- lgamma(sum(alphaVec)) - sum(lgamma(alphaVec))
48      logK <- sum((alphaVec - 1) * log(baseProbs))
49      return(logB + logK)
50    }
51
52    medianSeq <- list()
53    medianInds <- c(0, cumsum(as.numeric(table(y.arr$Code))))
54    for(i in 1:nrow(medians)) {
55      medianSeq[[i]] <- seq((medianInds + 1)[i], medianInds[i + 1], by = 1)
56    }
57
58    alpha <- alpha[1]
59    y <- y.stor[1, ]
60    y2 <- y2.stor[1, ]
61    ySqrt <- ySqrt.stor[1, ]
62    Nk <- Nk.stor[1, ]
63    pV <- Nk / N
64    mV <- mV.stor[1, ]
65
66    for(iter in 2:end) {
67
68      alphaProp <- runif(1, min = max(alphaPrior[1], alpha - alphaWidth), max =
          min(alphaPrior[2], alpha + alphaWidth))
69      yProp <- rgamma(N, shape = shape, scale = scale)
70      ySqrtProp <- sqrt(yProp)
71      y2Prop <- !y2
72
73      alphaAccept <- lBinDirichletSp(baseProbs, Nk, alphaProp) -
74        lBinDirichletSp(baseProbs, Nk, alpha) +
```

```
75        dunif(alpha, min = max(alphaPrior[1], alphaProp - alphaWidth), max = min
              (alphaPrior[2], alphaProp + alphaWidth), log = TRUE) -
76        dunif(alphaProp, min = max(alphaPrior[1], alpha - alphaWidth), max = min
              (alphaPrior[2], alpha + alphaWidth), log = TRUE)
77
78      if(log(runif(1)) < alphaAccept) {
79        alpha <- alphaProp
80        alpha.acceptance <- 1 + alpha.acceptance
81      }
82
83      for(i in 1:N) {
84
85        yi <- y[i]
86        y2i <- y2[i]
87        ySqrti <- ySqrt[i]
88        yiProp <- yProp[i]
89        y2iProp <- y2Prop[i]
90        ySqrtiProp <- ySqrtProp[i]
91        yiBin <- findInterval(yi, bins, all.inside = TRUE)
92        yiPropBin <- findInterval(yiProp, bins, all.inside = TRUE)
93
94        mVind <- findInterval(i, medianInds, left.open = TRUE)
95        if(is.na(medians$Estimate[mVind])) {
96          lMedianLike <- 0
97        } else {
98          mVi <- mV[mVind]
99          mViProp <- median(replace(y, i, yiProp)[medianSeq[[mVind]]])
100         if(mVi == mViProp) {
101           lMedianLike <- 0
102         } else {
```

```
103             lMedianLike <- dnorm(mViProp, medians$Estimate[mVind], medians$SE[
                    mVind], log = TRUE) -
104               dnorm(mVi, medians$Estimate[mVind], medians$SE[mVind], log = TRUE)
105           }
106         }
107
108         lPriorReg <- dnorm(y.arr$sqrtValue[i], beta[1] + beta[2] * ySqrtiProp,
                sigma, log = TRUE) -
109           dnorm(y.arr$sqrtValue[i], beta[1] + beta[2] * ySqrti, sigma, log =
                TRUE)
110
111         lPriorLogit <- dbern(y2i, exp(beta2 %*% c(1, y.arr$Value[i], yiProp)) /
                (1 + exp(beta2 %*% c(1, y.arr$Value[i], yiProp))), log = TRUE) -
112           dbern(y2i, exp(beta2 %*% c(1, y.arr$Value[i], yi)) / (1 + exp(beta2 %*
                % c(1, y.arr$Value[i], yi))), log = TRUE)
113
114         if(yiBin == yiPropBin) {
115
116           yiAccept <- lPriorReg + lPriorLogit + lMedianLike
117
118           if(log(runif(1)) < yiAccept) {
119             yi <- yiProp
120             ySqrti <- ySqrtiProp
121             mV[mVind] <- mViProp
122             y.acceptance <- 1 + y.acceptance
123           }
124
125         } else {
126
127           NkProp <- Nk; NkProp[yiBin] <- NkProp[yiBin] - 1; NkProp[yiPropBin] <-
                NkProp[yiPropBin] + 1
```

```
128         pVProp <- pV; pVProp[yiBin] <- (pVProp[yiBin] - (1/ N)); pVProp[
                yiPropBin] <- (pVProp[yiPropBin] + (1 / N))
129
130         lTrans <- log(NkProp[yiPropBin]) -
131           log(Nk[yiBin])
132
133         lPriorBins <- lBinDirichletSp(baseProbs, NkProp, alpha) -
134           lBinDirichletSp(baseProbs, Nk, alpha)
135
136         lPriorBase <- log(baseProbs[yiBin]) -
137           log(baseProbs[yiPropBin])
138
139         lMultiLike <- nk[yiBin] * (log(pVProp[yiBin]) - log(pV[yiBin])) +
140           nk[yiPropBin] * (log(pVProp[yiPropBin]) - log(pV[yiPropBin]))
141
142         yiAccept <- lTrans + lPriorBins + lPriorBase + lPriorReg + lPriorLogit
                + lMultiLike + lMedianLike
143
144         if(log(runif(1)) < yiAccept) {
145           yi <- yiProp
146           ySqrti <- ySqrtiProp
147           Nk <- NkProp
148           pV <- pVProp
149           mV[mVind] <- mViProp
150           y.acceptance <- 1 + y.acceptance
151         }
152
153       }
154
155       lPriorLogit <- dbern(y2iProp, exp(beta2 %*% c(1, y.arr$Value[i], yi)) /
            (1 + exp(beta2 %*% c(1, y.arr$Value[i], yi))), log = TRUE) -
```

```
156         dbern(y2i, exp(beta2 %*% c(1, y.arr$Value[i], yi)) / (1 + exp(beta2 %*
                % c(1, y.arr$Value[i], yi))), log = TRUE)
157
158       y2iAccept <- lPriorLogit
159
160       if(log(runif(1)) < y2iAccept) {
161         y2i <- y2iProp
162         y2.acceptance <- 1 + y2.acceptance
163       }
164
165       y[i] <- yi
166       ySqrt[i] <- ySqrti
167       y2[i] <- y2i
168
169     }
170
171     if(iter >= start & iter %% thin == 0) {
172       y.stor[(iter - start + 1) / thin, ] <- y
173       ySqrt.stor[(iter - start + 1) / thin, ] <- ySqrt
174       y2.stor[(iter - start + 1) / thin, ] <- y2
175       Nk.stor[(iter - start + 1) / thin, ] <- Nk
176       mV.stor[(iter - start + 1) / thin, ] <- mV
177       alpha.stor[(iter - start + 1) / thin] <- alpha
178     }
179
180   }
181
182   alpha.acceptance <- alpha.acceptance / end
183   y.acceptance <- y.acceptance / (N * end)
184   y2.acceptance <- y2.acceptance / (N * end)
185
```

```
186   out <- list(
187     y = y.stor,
188     ySqrt = ySqrt.stor,
189     y2 = y2.stor,
190     Nk = Nk.stor,
191     mV = mV.stor,
192     alpha = alpha.stor,
193     acceptance = c(captionpos"captionposalphacaptionpos" = alpha.acceptance,
          captionpos"captionposycaptionpos" = y.acceptance, captionpos"captionposy2
          captionpos" = y2.acceptance)
194   )
195
196   return(out)
197
198 }
```