Scientific Research Publishing

Applied Mathematics, 2024, 15, 214-234
https://www.scirp.org/journal/am
ISSN Online: 2152-7393
ISSN Print: 2152-7385

A Bayesian Mixture Model Approach to Disparity Testing

Gary C. McDonald

Department of Mathematics and Statistics, Oakland University, Rochester, USA
Email: mcdonald@oakland.edu

How to cite this paper: McDonald, G.C. (2024) A Bayesian Mixture Model Approach to Disparity Testing. *Applied Mathematics*, **15**, 214-234.
https://doi.org/10.4236/am.2024.153012

Received: February 21, 2024
Accepted: March 26, 2024
Published: March 29, 2024

Abstract

The topic of this article is one-sided hypothesis testing for disparity, *i.e.*, the mean of one group is larger than that of another when there is uncertainty as to which group a datum is drawn. For each datum, the uncertainty is captured with a given discrete probability distribution over the groups. Such situations arise, for example, in the use of Bayesian imputation methods to assess race and ethnicity disparities with certain insurance, health, and financial data. A widely used method to implement this assessment is the Bayesian Improved Surname Geocoding (BISG) method which assigns a discrete probability over six race/ethnicity groups to an individual given the individual's surname and address location. Using a Bayesian framework and Markov Chain Monte Carlo sampling from the joint posterior distribution of the group means, the probability of a disparity hypothesis is estimated. Four methods are developed and compared with an illustrative data set. Three of these methods are implemented in an R-code and one method in WinBUGS. These methods are programed for any number of groups between two and six inclusive. All the codes are provided in the appendices.

Keywords

Bayesian Improved Surname and Geocoding (BISG), Mixture Likelihood Function, Posterior Distribution, Metropolis-Hastings Algorithms, Random Walk Chain, Independence Chain, Gibbs Sampling, WinBUGS

1. Introduction

This article deals with the issue of disparity testing, *i.e.*, testing the hypothesis that the mean of one group is larger than that of a second group when the groups from which samples are drawn are uncertain and given with a probability distribution. McDonald [1] presented relevant approaches and calculations for

such testing, for both frequentist and Bayesian formulations, with two groups. In both formulations, use is made of all possible data configurations along with their corresponding probabilities for small sample sizes (≤6). Elkadry and McDonald [2] provide extensive discussion of the background giving rise to such disparity testing problems, and give an R-code to analyze sample sizes up to 22 using all possible data configurations. McDonald and Oakley [3] use a Bayesian framework and Markov Chain Monte Carlo sampling from the joint posterior distribution of the group means to estimate the probability of a disparity hypothesis. They provide the applicable R-codes and a WinBUGS code, and greatly extend sample size limitations of previous methods given in the literature. McDonald and Willard [4], using a frequentist approach, employ a bootstrap methodology to generate summary statistics for the population of p-values arising from all possible configurations of the data to assess the credibility of the disparity hypothesis. Using their provided R-code, previous limitations on sample sizes are substantially eliminated. The p-value calculations for statistical hypotheses testing are presented in numerous texts (e.g., see Navidi [5]).

The primary motivation for this work is the use of imputation of race/ethnicity of an individual based on available information such as surname and address. For example, in testing for disparity among race/ethnicity loan applicants where such information is not directly available, imputation methods are being used to assign one of six race/ethnicity groups to an application. There has been considerable research in developing such imputation methodology following the work of Fiscella and Fremont [6]. See, for example, the articles by Adjaye-Gbewonyo, *et al.* [7], Brown, *et al.* [8], Consumer Financial Protection Bureau [9] [10], Elliott, *et al.* [11], McDonald and Rojc [12] [13] [14], Zhang [15], and Zavez, *et al.* [16] for a wide variety of applications of such imputation methods in healthcare and finance using geocoding and surname analysis. Voicu, *et al.* [17] include the first name information, along with surname and geocoding, to improve the accuracy of race/ethnicity classification. To assess disparity of, say, auto loan rates extended to two race/ethnicity groups, a one-sided statistical hypothesis test could be used to assess the plausibility of the mean rate of one group being larger than that of another. This article addresses the issue of such statistical testing with imputed data as noted above.

This article expands the models used in references given above from two groups based on a binomial model to one using c groups ($2 \le c \le 6$) based on a mixture model within a Bayesian context. The methodologies developed in McDonald and Oakley [3] are applied with the more encompassing mixture likelihood function. Thus, this work is directly applicable to the output of proxy methods such as that from the Bayesian Improved Surname Geocoding (BISG) (see, e.g., Elliott, *et al.* [11]) used extensively by the Consumer Financial Protection Bureau (CFPB) and others. Using BISG, the CFPB [9] ordered Ally Financial Inc. and Ally Bank to pay $80 million in damages to African-American, Hispanic, and Asian and Pacific Islander consumers harmed by Ally's alleged discriminatory

auto loan pricing, and $18 million in civil money penalties. The Wall Street Journal provides a website (published in 2015), http://graphics.wsj.com/ally-settlement-race-calculator/, implementing BISG for six race/ethnicity groups.

A Bayesian approach to the statistical hypothesis testing problem is utilized here. With this approach, the group means of the variable of interest (e.g., auto loan interest rates) are modeled with a probability distribution from which the probability of the disparity hypothesis can be calculated. Prior knowledge on the group means is combined with the likelihood function of the sample data to yield a so-called posterior probability distribution of the group means. Several approaches to these calculations are herein described and illustrated using computational codes given in the appendices. The illustrative calculations utilize one set of assumptions on the model and data ("noninformative" prior knowledge on the group means, and normal likelihood function for the sample data). However, the approach and computational tools given in the appendices are easily adapted to other assumptions. Hence the robustness of conclusions can be assessed with several other specifications of model assumptions. Some robustness considerations are explicitly considered in Sections 4 and 5.

Section 2 of this article describes the data set to be used subsequently to illustrate the computations of the various methodologies herein presented. It also applies a method labeled Laplace (Albert [18]) to develop a multivariate normal distribution approximation to the posterior distribution of the group means. Section 3 details the use of "BayesMix", an R-code, integrating several Metropolis-Hastings algorithms so as to generate draws from the posterior distribution of the group means. Using these draws, the mean values of the group means are calculated to assess the probabilities of linear contrasts of these mean values. The probability of a disparity hypothesis is calculated. Section 4 uses the publicly available software package WinBUGS to calculate quantities similar to those of Section 3. Gill [19] and Christensen, *et al.* [20] provide excellent introductions to Bayesian modeling emphasizing the use of both R and WinBUGS to analyze real data. Section 5 provides a summary and the concluding remarks.

2. Normal Approximation to the Posterior Distribution

2.1. The Data

To illustrate the following methodologies a simulated data set consisting of n = 18 observations, given in **Appendix A**, will be used. The data can be entered into the programs given in the **Appendices B-D** in several different manners, e.g., excel spreadsheet, list format, etc. The excel spreadsheet format will be used here with the R-code BayesMix given in **Appendix B**. The excel spreadsheet with the example data for this article is labeled "TestData.xlsx". Two R packages are required for using this data format with the BayesMix: "readxl" and "LearnBayes". The data of **Appendix A** were randomly drawn from a normal distribution in c = 6 groups of three—with means of 2, 4, 6, 8, 10, and 12, and with standard dev-

iations of 1. The probabilities for the six categories of each datum are given as q = (q_1, ⋯, q_6). For each datum the category from which it was drawn is assigned the max probability. The objective of the analysis is to assess stochastic properties of the group means based on the data and group probabilities. These analyses will utilize algorithms given in Albert [18] and utilized in McDonald and Oakley [3] for c = 2. The user input required to execute BayesMix is given in Table 1. The specific values of the variables given in **Appendix B** used in this article are noted. The vector A, as given here, is used to formulate the disparity hypothesis "mu[2] is greater than mu[1]" for which its probability is estimated. Similarly, the contrast vector A1 is used for the hypothesis "mu[3] is greater than the average of mu[4] and mu[5]". The quantities mu[i] refer to the group[i] means.

The output of BayesMix contains many data summaries and diagnostics useful in checking the analyses structures and calculations. The user can suppress one or more of these if so wished. However, the primary output is Table 2 (less the column WinBUGS).

2.2. Laplace

As described in the above references, the R-code Laplace computes the posterior mode of the vector mu = (mu[1], ⋯, mu[c]), the associated variance-covariance matrix, and an estimate of the logarithm of the normalizing constant for a general posterior density, and an indication of algorithm convergence (true or false). Three inputs are required: logpost, a function that defines the logarithm of the posterior density (denoted by loglike in **Appendix B**); mode, an initial guess at the posterior mode; par, a list of parameters associated with the function logpost.

The probability (posterior) density for the data is a mixture distribution given by

$$f\left(x/q,\mu,\sigma\right)=\sum_{i=1}^{c}q_i\cdot\mathrm{dnorm}\left(x,\mu_i,\sigma\right) \tag{1}$$

where $0 \le q_i \le 1$, $\Sigma q_i = 1$, $|x| < \infty$, $0 < \sigma < \infty$, and $\mathrm{dnorm}\left(x,\mu_i,\sigma\right)$ is the normal probability density with mean and sigma equal to $\left(\mu_i,\sigma\right)$ evaluated at datum *x*.

Table 1. User required input for BayesMix (**Appendix B**).

Variable	Description	Values used in this article
sig	Std dev of mixture normal densities	1
mcn	Sample size of MCMC chains	52,000
sca	Scale parameter for MCMC random walk	1.2
dis	Burn-in for MCMC draws	2000
A	Vector to compare two group means	(−1, 1, 0, 0, 0, 0)
A1	Vector specifying linear transform of group means	(0, 0, 1, −0.5, −0.5, 0)

Table 2. Output (to four dp) of BayesMix (**Appendix B**) and of WinBUGS (**Appendix C**) with **Appendix A** data.

	Lap	Ran	Ind	RanRed	IndRed	WinBUGS
Draws	NA	52,000	52,000	50,000	50,000	50,000
est1	2.7351	2.7107	2.7327	2.7117	2.7320	2.728
sd1	0.5977	0.6289	0.6190	0.6312	0.6184	0.6154
est2	4.8257	4.8337	4.8220	4.8307	4.8211	4.833
sd2	0.6052	0.6328	0.6214	0.6320	0.6212	0.6515
est3	7.0174	7.1319	7.1520	7.1305	7.1504	7.146
sd3	0.7207	0.8304	0.8338	0.8294	0.8321	0.8727
est4	7.7113	7.6265	7.6050	7.6289	7.6036	7.605
sd4	0.6837	0.8464	0.7738	0.8466	0.7733	0.8089
est5	10.1977	10.3075	10.3140	10.3063	10.3149	10.27
sd5	0.7285	0.8609	0.8218	0.8634	0.8207	0.9251
est6	11.8836	11.8867	11.8753	11.8856	11.8753	11.87
sd6	0.5824	0.6040	0.5982	0.6016	0.5985	0.603
P(mu[2] > mu[1])	0.9934	0.9904	0.9901	0.9904	0.9902	0.9905
P(Y > 0)	0.0146	NA	NA	0.0491	0.0485	0.0533
Accept. Rate	NA	0.2191	0.8091	NA	NA	NA

The initial guess mode, a vector of length c, is denoted by mu in BayesMix. This initial guess can affect the convergence of the Laplace algorithm indicated by "true" or "false" in the output. One method for specifying this guess is to put each of the mu values equal to the average of the data. This has worked well for many of the calculations made in preparing this manuscript, but not all. There have been instances when the output of Laplace indicated "false" for convergence. In these cases, a good strategy is to rerun the Laplace function with the initial guess restated with the mode values output for the false convergence output. Another strategy that has worked very well, and is included in BayesMix, is to form the initial guess of mu[i] by taking the average of the data for which the category[i] has the maximum value over the c groups.

Laplace estimates of the mode values of the mu variables along with their standard deviations are given in Table 2 in the column denoted Lap. The estimates and standard deviations for mu[i] are denoted by esti and sdi, respectively. Since A is specified as (−1, 1, 0, 0, 0, 0), the vector (Ahi, Alo) = (2, 1) and the probability of mu[2] being greater than mu[1] is approximated with the normal distribution and the Laplace variance-covariance output as 0.9934. Also, A1 is specified by (0, 0, 1, −0.5, −0.5, 0), so Y = mu[3] − 0.5(mu[4] + mu[5]). Thus, Y > 0 is equivalent to mu[3] greater than the average of mu[4] and mu[5], the estimated probability of which is given in Table 2 as 0.0146. The entries in the other columns of Table 2, and the row "Accept Rate", are explained in subse-

quent sections of this article.

While the Laplace algorithm with the normal approximation may not be most appropriate for the posterior distribution, the Laplace output provides very useful input to the algorithms to be utilized subsequently. The output of Laplace is explicitly used in the "proposal" and "start" inputs for the two Markov Chain Monte Carlo (MCMC) algorithms to be described in Section 3. BayesMix, which generated **Table 2**, ran in about 6.5 minutes on a laptop computer with Windows 10.

3. Exploring the Posterior Distribution with Metropolis-Hastings Algorithms

In this Section, random draws from the posterior distribution of the mean vector mu are used to estimate the probability of disparity as well as the cumulative probability distribution of any contrast of the category means. As was developed by McDonald and Oakley [3], these draws are obtained using the popular MCMC methods in the Bayesian literature. As stated in Section 3 of the McDonald and Oakley reference, these methods consist of specifying an initial value for the parameter vector of interest and then generating a chain of values each of which depends only on the previous value in the chain. A rule for generating the sequence is a function of a proposal density and an acceptance probability. Not all generated values in the sequence are accepted as they must pass a probability criterion to be retained in the final sample. These components, the proposal density and acceptance probability criterion, are constructed so that the accepted simulated draws converge to draws from a random variable having the target posterior distribution. Different choices of the proposal density may result in slightly different distribution summaries.

Two such proposal density implementations described in detail by Albert [18], and executed in the R package LearnBayes, are incorporated here in BayesMix given in **Appendix B**. The R functions "rwmetrop" and "indepmetrop" implement the so-called random walk and independence Metropolis-Hastings algorithms for special choices of the proposal densities. The methodology descriptions given in the following two subsections follow closely that of Section 3 of McDonald and Oakley [3] where the application is developed for $c = 2$ with a binomial likelihood. Here the applications are built for $2 \le c \le 6$ with the likelihood function consisting of a mixture of normal densities given in Equation (1). The R-code BayesMix can be extended in a straightforward manner for $c > 6$, *i.e.*, for more than six groups. However, the use of WinBUGS, described in Section 4, can be easily applied to cases of $c > 6$ by suitably stating the number of groups and appropriately entering the data in **Appendix C** and **Appendix D**.

Since the MCMC algorithms converge to sampling from the posterior distribution, initial draws may not be from the stationary distribution of the Markov chain, *i.e.*, not drawn from the target posterior distribution. Thus, it is common practice to discard an initial portion of the draws and utilize the remainder. This

practice is frequently referred to as "burn-in." There are several diagnostics that can be viewed to help decide when the chain has progressed sufficiently far to stabilize on the posterior distribution (e.g., see Section 9.6 of Albert and Hu [21]; Section 4.4.2 of Lunn *et al.* [22]). **Table 2** provides results based on 52,000 random draws using MCMC algorithms described in the following two subsections. Results are also provided following a burn-in of 2000 draws, *i.e.*, results based on the last 50,000 random draws. As noted, the burn-in deletions result in very minor changes in the posted estimates for this illustrative data set. The two chains utilized in subsections 3.1 and 3.2 are described fully in Albert [18].

3.1. Random Walk Metropolis Chain

Within LearnBayes, the function rwmetrop (logpost, proposal, start, m, par) requires five inputs: logpost, function defining the log posterior density; proposal, a list containing var, an estimated variance-covariance matrix, and scale, the Metropolis scale factor; start, a vector giving the starting value of the parameter; m, the number of iterations of the chain; par, the data used in the function logpost. The output is par, a matrix of the simulated values where each row corresponds to a value of the vector parameter; accept, the acceptance rate of the algorithm. A summary of the outputs of rwmetrop run with the specification of the data described in Section 2.1 with 52,000 draws from the posterior distribution is given in **Table 2** in the column designated Ran.

As noted in the output, the acceptance rate here is 0.2191. The input "scale" should be chosen so that the acceptance rate is around 25%. Acceptance rates are discussed on page 121 of Albert [18] and on page 253 of Rizzo [23]. The acceptance rate is a decreasing function of scale.

The results following a burn-in of 2000 are given in **Table 2** column RanRed. The results in this column are thus based on 50,000 draws and differ very little than those in column Ran.

3.2. Independence Metropolis Chain

The function indepmetrop (logpost, proposal, start, m, data) also requires five inputs: logpost, as above; proposal, a list containing mu, an estimated mean, and var, an estimated variance-covariance matrix for the normal proposal density; start, array with a single row that gives the starting value for the parameter vector; m, the number of iterations of the chain; data, data used in the function logpost. A summary of the outputs of indepmetrop run with the illustrative data set with 52,000 draws is given in **Table 2** in the column designated Ind. Column IndRed provides analogous results for the 50,000 draws following the burn-in of 2000. As is the case with rwmetrop algorithm, there is negligible difference in the results with and without burn-in deletion.

4. Applying WinBUGS for Bayesian Simulation

Another approach to generating draws from the posterior distribution is to use a MCMC software package, WinBUGS, designed specifically for Bayesian compu-

tation. WinBUGS is the MS Windows operating system version of BUGS, Bayesian Analysis Using Gibbs Sampling. There is a free download of this Bayesian software package available at the site (http://www.mrc-bsu.cam.ac.uk/bugs/overview/contents.shtml). Clear discussions of the applicability, limitations, and use of WinBUGS are given in Hahn [24], Woodworth [25], Lunn *et al.* [22], and in many other references. The setup of this approach for the illustrative example considered here, where c = 6 and n = 18, is given in **Appendix C**. Note that in WinBUGS the normal distribution density is specified by dnorm(μ, τ), where μ is the mean and τ is the precision (= 1/variance). This differs from the specification in R, as given in Equation (1), where the normal distribution density at x is denoted by dnorm(x, μ, σ) with μ as the mean and σ as the standard deviation.

Output from WinBugs is given in Table 2 based on a sample of 50,000 draws from the posterior distribution following a burn-in of an initial 2000 draws (similar to that done for the RanRed and IndRed entries). A "count1" node is included to estimate P(mu[2] > mu[1]) as shown in Table 2. A similar "count2" node is included to estimate P(Y > 0) for a user specified transformation of the group means, A1 = (0, 0, 1, −0.5, −0.5, 0), in this example. To execute WinBugs, initial values are required for mu, group, and sigma. For mu, a reasonable choice for mu inits are the mode values given by Laplace. For group, a reasonable choice is to assign that group which has the largest probability for the datum. In case several groups share the largest value, choose one of those at random. For sigma, simply use a reasonable guess (e.g., use sigma = 1 as done here). The robustness of the results can easily be checked by running WinBUGS with other choices. The run time for WinBUGS as given in **Appendix C** is about forty seconds.

Appendix D provides a modification of the **Appendix C** WinBUGS program by treating sigma as an unknown value, common to all six groups, and to be estimated along with the other nodes. With this program, the mean of the 50,000 sigma draws is 0.9886 with a standard deviation of 0.3024. The output for the other nodes is very close to those given in Table 2 for WinBUGS with sigma fixed at one (*i.e.*, the output using the **Appendix C** program). The P(m[2] > mu[1]) is estimated to be 0.9811, and P(Y > 0) to be 0.0689. A further extension of **Appendix D** can be made easily to accommodate the case where the group sigma values are not assumed known or to be equal.

5. Summary and Concluding Remarks

Table 2 shows a great deal of consistency with the entries, especially among the last five columns of the table. The results for the Laplace column also are in close agreement with those of the other columns, with perhaps the row entries corresponding to the standard deviations (*i.e.*, sdi's) where the Laplace values are uniformly a bit smaller than the others. The last five columns are based on statistics calculated from the random draws from the posterior distribution, whereas the Laplace column is based on a numerical fit to the data. These observa-

tions are based on the one data set given in **Appendix A** and may not generalize to other data sets. All the results given in this article apply to a mixture likelihood function of normal densities. Other densities could be substituted with appropriate changes in Equation (1) and appropriate modifications to the computer codes given in the Appendices.

An important question to be addressed with the methodologies used in this article is simply one of sample size. What is the tradeoff between the sample size (n) and the computing time? To partially address this issue, a sample of size n = 72 was constructed by pooling together four copies of the data given in **Appendix A**. The R-code in **Appendix B** and the WinBUGS code in **Appendix C** were run with exactly the same inputs used in Sections 3 and 4, with the expanded data set, generating an analogous Table 2. The laptop computer time for BayesMix was approximately forty-five minutes and for WinBUGS approximately forty seconds. The corresponding computer times with n = 18 (**Appendix A**) were approximately nine minutes for BayesMix and forty seconds for WinBUGS. There was no meaningful difference in computing times for WinBUGS between the two data sets. The output for the larger sample size closely matched that for the smaller sample size with the exception of the standard deviations (sdi's) whose values with the larger data set were approximately half of the values with those obtained with the smaller set. For BayesMix, the computer time was approximately five times longer for n = 72 vs. n = 18. For much larger sample sizes, the bootstrap approach developed by McDonald and Willard [4] could be adapted to limit the required computing time.

As with all analyses, especially Bayesian, the robustness of the conclusions with respect to choices of model and prior specifications should be explored. With WinBUGS, a feature called "chains" facilitates running multiple MCMCs with different prior initializations given in the inits list. The priors used in this article were chosen to be so-called "noninformative" (e.g., distributions with very large variances). Other graphical diagnostics provided by WinBUGS are given in **Appendix E** for mu[1] and are very helpful in addressing the issue of stability of MCMC draws from a stationary posterior distribution. These diagnostics, along with similar ones for the other nodes, support the assessment that the MCMC draws are from a stable stationary posterior distribution. Chapter 6 of Hahn [24] and Chapter 14 of Gill [19] provide extensive discussion of assessing MCMC performance in WinBUGS along with some interfaces to R packages.

In conclusion, the Bayesian methods herein presented with appropriate computer codes provide a statistically sound basis for drawing conclusions about group disparity using race and ethnicity uncertainty data such as that arising with BISG and similarly based methodologies.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] McDonald, G.C. (2018) Statistical Testing When the Populations from Which Samples Are Drawn Are Uncertain. *Health Services and Outcomes Research Methodology*, **18**, 155-174. https://doi.org/10.1007/s10742-018-0182-7

[2] Elkadry, A. and McDonald, G.C. (2020) Hypothesis Testing When Data Sources Are Uncertain. *Journal of Statistical Theory and Practice*, **14**, Article No. 66. https://doi.org/10.1007/s42519-020-00132-5

[3] McDonald, G.C. and Oakley, R.H. (2023) Extending Computations for Disparity Testing When Data Sources Are Uncertain. *Health Services and Outcomes Research Methodology*, **23**, 207-226. https://doi.org/10.1007/s10742-022-00286-8

[4] McDonald, G.C. and Willard, J.F. (2023) Bootstrap Approach to Disparity Testing with Source Uncertainty in the Data. *Health Services and Outcomes Research Methodology.* https://doi.org/10.1007/s10742-023-00318-x

[5] Navidi, W. (2024) Statistics for Engineers and Scientists. 6th Edition, McGraw Hill, New York.

[6] Fiscella, K. and Fremont, A.M. (2006) Use of Geocoding and Surname Analysis to Estimate Race and Ethnicity. *Health Services Research*, **41**, 1482-1500. https://doi.org/10.1111/j.1475-6773.2006.00551.x

[7] Adjaye-Gbewonyo, D., Bednarczyk, R.A., Davis, R.L. and Omer, S.B. (2014) Using the Bayesian Improved Surname Geocoding Method (BISG) to Create a Working Classification of Race and Ethnicity in a Diverse Managed Care Population: A Validation Study. *Health Services Research*, **49**, 268-283. https://doi.org/10.1111/1475-6773.12089

[8] Brown, D.P., Knapp, C., Baker, K. and Kaufmann, M. (2016) Using Bayesian Imputation to Assess Racial and Ethnic Disparities in Pediatric Performance Measures. *Health Services Research*, **51**, 1095-1108. https://doi.org/10.1111/1475-6773.12405

[9] Consumer Financial Protection Bureau (2013, December 20) CFPB and DOJ Order Ally to Pay $80 Million to Consumers Harmed by Discriminatory Auto Loan Pricing. https://www.consumerfinance.gov/enforcement/actions/ally-financial-ally-bank/

[10] Consumer Financial Protection Bureau (2014) Using Publicly Available Information to Proxy for Unidentified Race and Ethnicity. https://www.consumerfinance.gov/data-research/research-reports/using-publicly-available-information-to-proxy-for-unidentified-race-and-ethnicity/

[11] Elliott, M.N., Morrison, P.A., Fremont, A., McCaffrey, D.F., Pantoja, P. and Lurie, N. (2009) Using the Census Bureau's Surname List to Improve Estimates of Race/Ethnicity and Associated Disparities. *Health Services and Outcomes Research Methodology*, **9**, 69-83. https://doi.org/10.1007/s10742-009-0047-1

[12] McDonald, K.M. and Rojc, K.J. (2015) Automotive Finance Regulation: Warning Lights Flashing. *The Business Lawyer*, **70**, 617-624.

[13] McDonald, K.M. and Rojc, K.J. (2017) Accelerating Regulation of Automotive Finance. *The Business Lawyer*, **72**, 559-566.

[14] McDonald, K.M. and Rojc, K.J. (2022) Ladies and Gentlemen: Rev Your Regulatory Engines! *The Business Lawyer*, **77**, 581-590.

[15] Zhang, Y. (2018) Assessing Fair Lending Risks Using Race/Ethnicity Proxies. *Management Science*, **64**, 178-197. https://doi.org/10.1287/mnsc.2016.2579

[16] Zavez, K., Harel, O. and Aseltine, R.H. (2022) Imputing Race and Ethnicity in Healthcare Claims Databases. *Health Services and Outcomes Research Methodolo-*

gy, **22**, 493-507. https://doi.org/10.1007/s10742-022-00273-z

[17] Voicu, I. (2018) Using First Name Information to Improve Race and Ethnicity Classification. *Statistics and Public Policy*, **5**, 1-13. https://doi.org/10.1080/2330443X.2018.1427012

[18] Albert, J. (2009) Bayesian Computation with R. 2nd Edition, Springer, New York. https://doi.org/10.1007/978-0-387-92298-0

[19] Gill, J. (2015) Bayesian Methods: A Social and Behavioral Sciences Approach. 3rd Edition, CRC Press, Boca Raton.

[20] Christensen, R., Johnson, W., Branscum, A. and Hanson, T.E. (2011) Bayesian Ideas and Data Analysis. CRC Press, Boca Raton. https://doi.org/10.1201/9781439894798

[21] Albert, J. and Hu, J. (2020) Probability and Bayesian Modeling. CRC Press, Boca Raton. https://doi.org/10.1201/9781351030144

[22] Lunn, D., Jackson, C., Best, N., Thomas, A. and Spiegelhalter, D. (2013) The BUGS Book: A Practical Introduction to Bayesian Analysis. CRC Press, Boca Raton. https://doi.org/10.1201/b13613

[23] Rizzo, M.L. (2008) Statistical Computing with R. Chapman & Hall/CRC, Boca Raton.

[24] Hahn, E.D. (2014) Bayesian Methods for Management and Business: Pragmatic Solutions for Real Problems. Wiley, Hoboken.

[25] Woodworth, G.G. (2004): Biostatistics: A Bayesian Introduction. Wiley, Hoboken.

Appendix A. Excel Spreadsheet Data, TestData.xlsx

z	q1	q2	q3	q4	q5	q6
2.701	0.7	0.1	0.05	0.05	0.05	0.05
1.915	0.8	0.025	0.1	0.025	0.025	0.025
3.569	0.6	0.05	0.05	0.2	0.05	0.05
4.817	0.1	0.7	0.05	0.05	0.05	0.05
4.395	0.025	0.8	0.1	0.025	0.025	0.025
5.213	0.05	0.6	0.05	0.2	0.05	0.05
6.355	0.1	0.05	0.7	0.05	0.05	0.05
8.216	0.025	0.1	0.8	0.025	0.025	0.025
5.909	0.05	0.05	0.6	0.2	0.05	0.05
6.683	0.1	0.05	0.05	0.7	0.05	0.05
8.067	0.025	0.1	0.025	0.8	0.025	0.025
8.512	0.05	0.05	0.2	0.6	0.05	0.05
10.05	0.1	0.05	0.05	0.05	0.7	0.05
11.467	0.025	0.1	0.025	0.025	0.8	0.025
8.71	0.05	0.05	0.2	0.05	0.6	0.05
11.742	0.1	0.05	0.05	0.05	0.05	0.7
12.293	0.025	0.1	0.025	0.025	0.025	0.8
11.663	0.05	0.05	0.2	0.05	0.05	0.6

Appendix B. BayesMix Implementing Laplace, Metropolis-Hastings Algorithms

```
#BayesMix
set.seed(210)
####User supplied input
#an slsx spreadsheet with rows being observations
#n is the sample size, i.e., the length of z--extracted from spreadsheet
#c is the number of multinomial categories (2<=c<=6)--extracted from spreadsheet
#z are the response data
#q are the multinomial probabilities associated with z, given in order of z
#sig is the standard deviation of the normal mixture densities
#mcn is the sample size of the MCMC chains
#sca is the scale parameter in proposal.2 for MCMC random walk
#dis is the number of initial MCMC draws to be dropped
#setup is for estimating Pr(mu[2]>mu[1]). Other probs can be easily inserted.
sig=1; mcn=52000; sca=1.2; dis=2000
library(readxl)
dfex<-read_excel("C:/Users/Gary/Documents/Multinomial BISG R Codes/TestData.xlsx")
dfm<-data.matrix(dfex)
colnames(dfm)<-NULL
```

```
dim(dfm)
n<-nrow(dfm)
c<-ncol(dfm)-1
df<-data.frame(dfm)
head(df,5)
tail(df,5)
#Calculate the prob that a linear transformation of the mu's, Y, is > 0
#Y <- sum(A[i]*mu[i])
#Input A to specifiy the linear transformation (length = c)
#A, here, is restricted to comparison of one mean to another mean
#For the more general linear trans., enter A1 (length=c) in the third to last
#section of this code for P(Y>0) estimates using Laplace and the two MCMCs after
#deleting the burn-ins
A<-c(-1,1,0,0,0,0)
Ahi<-which.max(A)
Alo<-which.min(A)
c(Ahi,Alo)
A<-as.matrix(A)
Atr<-t(A)
dim(A)
dim(Atr)
#####################################################################
message("The sample size is n = ",n)
message("The number of categories is c = ",c)
#ck is sum of multinomial probabilities for each response data, z
#each ck should equal 1
ck<-NULL
c1<-c+1
for (i in 1:n) {ck[i] <- sum(df[i,2:c1])}
table(ck)
#####################################################################
s<-NULL;logs<-NULL;H<-matrix(0,ncol=c,nrow=n);Hm<-NULL
#mu is starting vector of mode values of length c
for (i in 1:n){H[i,]<-dfm[i,2:ncol(dfm)]}
for (i in 1:n){Hm[i]<-which.max(H[i,])}
Hm
table(Hm)
dfH<-data.frame(Hm,dfm[,1])
Hmu<-aggregate(.~Hm,data=dfH,mean)
mu<-Hmu[,2]
#mu<-c(rep(mean(dfm[,1]),c))
loglike<-function(mu,df){
    for (i in 1:n){
```

```
        s[i]=0
        for (j in 1:c){
            s[i]<-s[i]+df[i,j+1]*dnorm(df[i,1],mu[j],sig)
            }
        }
for (k in 1:n){logs[k]<-log(s[k])}
loglk<-sum(logs)
return(loglk)
}
loglike(mu,df)
########################################################################
library(LearnBayes)
#see Albert's book, Sec. 5.5
init<-mu
fit=laplace(loglike,init,df)
fit
lap.means<-c(fit$mode)
lap.var<-diag(fit$var)
lap.sds<-sqrt(lap.var)
print("OUTPUT SUMMARY FROM laplace")
round(rbind(lap.means,lap.sds),4)
stdz<-(lap.means[Ahi]-lap.means[Alo])/
        sqrt(lap.var[Ahi]+lap.var[Alo]-2*fit$var[Ahi,Alo])
message("Estimate that Pr(mu[",Ahi,"]-mu[",Alo,"]>0) = ",
        round(pnorm(stdz),4))
########################################################################
#Now adding rwmetrop & indepmetrop
message("Sample size for MCMC chains is mcn = ",mcn)
#random walk Metropolis chain
#see Albert's book Secs. 6.8 and 6.9
#choose scale=sca so that accept rate is around 25%
#see Albert, page 121, for discussion of accept rates
#choose scale=sca so that accept rate is in (0.15,0.5)
#see Rizzo, page 253, for discussion
#accept rate is a decreasing function of scale
proposal.2=list(var=fit$var,scale=sca)
start=fit$mode
fit2=rwmetrop(loglike,proposal.2,start,mcn,df)
post.means.rw=apply(fit2$par,2,mean)
post.sds.rw=apply(fit2$par,2,sd)
message("OUTPUT SUMMARY FROM rwmetrop")
message("The acceptance rate for random walk = ",round(fit2$accept,4))
message("with ",mcn," draws from the MCMC rw posterior")
round(rbind(post.means.rw,post.sds.rw),4)
```

```
diffrwH_L<-fit2$par[,Ahi]-fit2$par[,Alo]
PrEst2<-length(diffrwH_L[diffrwH_L>0])/mcn
message("Estimate that Pr(mu[",Ahi,"]-mu[",Alo,"]>0) = ",round(PrEst2,4))
##################################
#Metropolis independence chain
#see Albert's book Sec. 6.9
proposal.3=list(var=fit$var,mu=fit$mode)
fit3=indepmetrop(loglike,proposal.3,start,mcn,df)
post.means.indep=apply(fit3$par,2,mean)
post.sds.indep=apply(fit3$par,2,sd)
message("OUTPUT SUMMARY FROM indepmetrop")
message("The acceptance rate for indepmetrop = ",round(fit3$accept,4))
message("with ",mcn," draws from the MCMC indep posterior")
round(rbind(post.means.indep,post.sds.indep),4)
diffindH_L<-fit3$par[,Ahi]-fit3$par[,Alo]
PrEst3<-length(diffindH_L[diffindH_L>0])/mcn
message("Estimate that Pr(mu[",Ahi,"]-mu[",Alo,"]>0) = ",round(PrEst3,4))
#########################################################################
#DISCARD FIRST 'DIS' DRAWS FROM RANDOM WALK & INDEPENDENT
message("Initial draws discarded, burn-in, is dis = ",dis)
newsize<-mcn-dis
dis1<-dis+1
#in particular, look at the draws from random walk
df1<-data.frame(fit2$par[,1])
for (i in 2:c){df1<-cbind(df1,fit2$par[,i])}
dim(df1)
df2<-data.frame(df1[dis1:mcn,])
dim(df2)
if (c==2){colnames(df2)<-c("mu1","mu2")}
if (c==3){colnames(df2)<-c("mu1","mu2","mu3")}
if (c==4){colnames(df2)<-c("mu1","mu2","mu3","mu4")}
if (c==5){colnames(df2)<-c("mu1","mu2","mu3","mu4","mu5")}
if (c==6){colnames(df2)<-c("mu1","mu2","mu3","mu4","mu5","mu6")}
avg2<-apply(df2,2,mean)
stddev2<-apply(df2,2,sd)
message("The number of draws after discarding the burn-in = ",newsize)
message("output from rw after deleting dis = ",dis," burn-ins")
round(rbind(avg2,stddev2),4)
diffrw.dis<-df2[,Ahi]-df2[,Alo]
PrEst4<-length(diffrw.dis[diffrw.dis>0])/newsize
message("Estimate that Pr(mu[",Ahi,"]-mu[",Alo,"]>0) = ",round(PrEst4,4))
print("mean and sd of diffrw.dis")
c(mean(diffrw.dis),sd(diffrw.dis))
```

```
#
#look at draws from independent walk
df3<-data.frame(fit3$par[,1])
for (i in 2:c){df3<-cbind(df3,fit3$par[,i])}
dim(df3)
df4<-data.frame(df3[dis1:mcn,])
dim(df4)
if (c==2){colnames(df4)<-c("mu1","mu2")}
if (c==3){colnames(df4)<-c("mu1","mu2","mu3")}
if (c==4){colnames(df4)<-c("mu1","mu2","mu3","mu4")}
if (c==5){colnames(df4)<-c("mu1","mu2","mu3","mu4","mu5")}
if (c==6){colnames(df4)<-c("mu1","mu2","mu3","mu4","mu5","mu6")}
avg3<-apply(df4,2,mean)
stddev3<-apply(df4,2,sd)
message("The number of draws after discarding the burn-in =",newsize)
message("output from indep after deleting dis = ",dis," burn-ins")
round(rbind(avg3,stddev3),4)
diffindep.dis<-df4[,Ahi]-df4[,Alo]
PrEst5<-length(diffindep.dis[diffindep.dis>0])/newsize
message("Estimate that Pr(mu[",Ahi,"]-mu[",Alo,"]>0) = ",round(PrEst5,4))
print("mean and sd of diffindep.dis")
c(mean(diffindep.dis),sd(diffindep.dis))
#############################################################################
#Assemble results in a table using given A
if (c==2){Lap<-c(NA,fit$mode[1],lap.sds[1],fit$mode[2],lap.sds[2],
      pnorm(stdz),NA)}
if (c==3){Lap<-c(NA,fit$mode[1],lap.sds[1],fit$mode[2],lap.sds[2],
      fit$mode[3],lap.sds[3],pnorm(stdz),NA)}
if (c==4){Lap<-c(NA,fit$mode[1],lap.sds[1],fit$mode[2],lap.sds[2],
      fit$mode[3],lap.sds[3],fit$mode[4],lap.sds[4],pnorm(stdz),NA)}
if (c==5){Lap<-c(NA,fit$mode[1],lap.sds[1],fit$mode[2],lap.sds[2],
      fit$mode[3],lap.sds[3],fit$mode[4],lap.sds[4],
      fit$mode[5],lap.sds[5],pnorm(stdz),NA)}
if (c==6){Lap<-c(NA,fit$mode[1],lap.sds[1],fit$mode[2],lap.sds[2],
      fit$mode[3],lap.sds[3],fit$mode[4],lap.sds[4],
      fit$mode[5],lap.sds[5],fit$mode[6],lap.sds[6],pnorm(stdz),NA)}
Lap<-round(Lap,4)
Ran<-c(post.means.rw[1],post.sds.rw[1])
for (i in 2:c){Ran<-c(Ran,post.means.rw[i],post.sds.rw[i])}
Ran<-c(mcn,Ran,PrEst2,fit2$accept)
Ran<-round(Ran,4)
Ind<-c(post.means.indep[1],post.sds.indep[1])
for (i in 2:c){Ind<-c(Ind,post.means.indep[i],post.sds.indep[i])}
```

```
Ind<-c(mcn,Ind,PrEst3,fit3$accept)
Ind<-round(Ind,4)
RanRed<-c(avg2[1],stddev2[1])
for (i in 2:c){RanRed<-c(RanRed,avg2[i],stddev2[i])}
RanRed<-c(newsize,RanRed,PrEst4,NA)
RanRed<-round(RanRed,4)
IndRed<-c(avg3[1],stddev3[1])
for (i in 2:c){IndRed<-c(IndRed,avg3[i],stddev3[i])}
IndRed<-c(newsize,IndRed,PrEst5,NA)
IndRed<-round(IndRed,4)
tab<-matrix(rbind(Lap,Ran,Ind,RanRed,IndRed),ncol=5,byrow=TRUE)
colnames(tab)<-c('Lap','Ran','Ind','RanRed','IndRed')
if (c==2){rownames(tab)<-c('Draws','est1','sd1','est2','sd2',
      'P(mu(Ahi)>mu(Alo))','Acc. Rate')}
if (c==3){rownames(tab)<-c('Draws','est1','sd1','est2','sd2',
      'est3','sd3','P(mu(Ahi)>mu(Alo))','Acc. Rate')}
if (c==4){rownames(tab)<-c('Draws','est1','sd1','est2','sd2',
      'est3','sd3','est4','sd4','P(mu(Ahi)>mu(Alo))','Acc. Rate')}
if (c==5){rownames(tab)<-c('Draws','est1','sd1','est2','sd2',
      'est3','sd3','est4','sd4','est5','sd5',
      'P(mu(Ahi)>mu(Alo))','Acc. Rate')}
if (c==6){rownames(tab)<-c('Draws','est1','sd1','est2','sd2',
      'est3','sd3','est4','sd4','est5','sd5','est6','sd6',
      'P(mu(Ahi)>mu(Alo))','Acc. Rate')}
tab
#############################################################################
#Calculate Pr(Y>0) using Laplace approximation
#A1 specifies the coefficients of the linear transformation (length=c)
A1<-c(0,0,1,-0.5,-0.5,0)
A1tr<-t(A1)
C<-A1tr%*%fit$var
C
D<-C%*%A1
#D is the variance of Y
D
D<-as.numeric(D)
Std<-sqrt(D)
lap.means<-as.matrix(lap.means)
#fg is Laplace estimate of the mean of Y divided by its sd
fg<-A1tr%*%lap.means/Std
dim(fg)
dim(lap.means)
proba<-pnorm(fg)
```

```
message("Estimate that Pr(Y>0) = ",round(proba,4))
################################
#Using random walk posterior draws after discard of burn-n
df2<-as.matrix(df2)
E<-df2%*%A1
probb<-length(E[E>0])/newsize
message("Estimate that Pr(Y>0) = ",round(probb,4))
################################
#Using indept. walk posterior draws after discard of burn-in
df4<-as.matrix(df4)
F<-df4%*%A1
probc<-length(F[F>0])/newsize
message("Estimate that Pr(Y>0) = ",round(probc,4))
dim(fg)
dim(lap.means)
proba<-pnorm(fg)
message("Estimate that Pr(Y>0) = ",round(proba,4))
################################
#Using random walk posterior draws after discard of burn-n
df2<-as.matrix(df2)
E<-df2%*%A1
probb<-length(E[E>0])/newsize
message("Estimate that Pr(Y>0) = ",round(probb,4))
################################
#Using indept. walk posterior draws after discard of burn-in
df4<-as.matrix(df4)
F<-df4%*%A1
probc<-length(F[F>0])/newsize
message("Estimate that Pr(Y>0) = ",round(probc,4))
```

Appendix C. WinBUGS, Assuming $\sigma = 1$ for the c = 6 Groups

```
#dcat.WinBUGS_newdata.n=18
model {
    for (i in 1:n) {
        group[i] ~ dcat(p[i,])
        y[i] ~ dnorm(mu[group[i]],1)
    }
#prior distributions
#I(a,b) is truncation at (a,b)
for (i in 1:6) {mu[i] ~ dnorm(0,0.001)I(0,14)}
#computed variables
diff1 <- mu[2]-mu[1]
count1<-step(diff1)
```

```
diff2 <- mu[3]-0.5*(mu[4]+mu[5])
count2<-step(diff2)
}
#data
list(y=c(2.701,1.915,3.569,4.817,4.395,5.213,
6.355,8.216,5.909,6.683,8.067,8.512,
10.050,11.467,8.710,11.742,12.293,11.663),n=18,
p = structure(
        .Data = c(.7,.1,.05,.05,.05,.05,
                  .8,.025,.1,.025,.025,.025,
                  .6,.05,.05,.2,.05,.05,
                  .1,.7,.05,.05,.05,.05,
                  .025,.8,.1,.025,.025,.025,
                  .05,.6,.05,.2,.05,.05,
                  .1,.05,.7,.05,.05,.05,
                  .025,.1,.8,.025,.025,.025,
                  .05,.05,.6,.2,.05,.05,
                  .1,.05,.05,.7,.05,.05,
                  .025,.1,.025,.8,.025,.025,
                  .05,.05,.2,.6,.05,.05,
                  .1,.05,.05,.05,.7,.05,
                  .025,.1,.025,.025,.8,.025,
                  .05,.05,.2,.05,.6,.05,
                  .1,.05,.05,.05,.05,.7,
                  .025,.1,.025,.025,.025,.8,
                  .05,.05,.2,.05,.05,.6),
        .Dim = c(18,6)
        )
)
#inits
list(mu=c(2.735,4.826,7.017,7.711,10.198,11.884),group=c(1,1,1,2,2,2,3,3,3,
4,4,4,5,5,5,6,6,6))
```

Appendix D. WinBUGS, Assuming a Common Unknown σ for the c = 6 Groups

```
#dcat.WinBUGS_newdata.n=18; the number of groups is c = 6
model {
        for (i in 1:n) {
                group[i] ~ dcat(p[i,])
                y[i] ~ dnorm(mu[group[i]],tau)
        }
#prior distributions
sigma ~ dgamma(0.0001,0.0001)
tau <- 1/pow(sigma,2)
```

```
#I(a,b) is truncation at (a,b)
for (i in 1:6){mu[i] ~ dnorm(0,0.001)I(0,14)}
#computed variables
diff1 <- mu[2]-mu[1]
count1<-step(diff1)
diff2 <- mu[3]-0.5*(mu[4]+mu[5])
count2<-step(diff2)
}
#data
list(y=c(2.701,1.915,3.569,4.817,4.395,5.213,
6.355,8.216,5.909,6.683,8.067,8.512,
10.050,11.467,8.710,11.742,12.293,11.663),n=18,
p = structure(
        .Data = c(.7,.1,.05,.05,.05,.05,
                        .8,.025,.1,.025,.025,.025,
                        .6,.05,.05,.2,.05,.05,
                        .1,.7,.05,.05,.05,.05,
                        .025,.8,.1,.025,.025,.025,
                        .05,.6,.05,.2,.05,.05,
                        .1,.05,.7,.05,.05,.05,
                        .025,.1,.8,.025,.025,.025,
                        .05,.05,.6,.2,.05,.05,
                        .1,.05,.05,.7,.05,.05,
                        .025,.1,.025,.8,.025,.025,
                        .05,.05,.2,.6,.05,.05,
                        .1,.05,.05,.05,.7,.05,
                        .025,.1,.025,.025,.8,.025,
                        .05,.05,.2,.05,.6,.05,
                        .1,.05,.05,.05,.05,.7,
                        .025,.1,.025,.025,.025,.8,
                        .05,.05,.2,.05,.05,.6),
        .Dim = c(18,6)
        )
)
#inits
list(mu=c(8,8,8,8,8,8),group=c(1,1,1,2,2,2,3,3,3,
4,4,4,5,5,5,6,6,6),sigma=1)
```

Appendix E. WinBUGS Node Statistics Output and Graphical Diagnostics for mu[1] for Appendix 3 Run-From Top Left to Lower Right: Dynamic Trace, Running Quantiles, Time Series History, Kernel Density, and Autocorrelation Function

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
count	0.9905	0.09721	4.865E-4	1.0	1.0	1.0	2001	50000
mu[1]	2.728	0.6154	0.003137	1.505	2.734	3.916	2001	50000
mu[2]	4.833	0.6515	0.003842	3.584	4.832	6.056	2001	50000
mu[3]	7.146	0.8727	0.006982	5.466	7.136	8.903	2001	50000
mu[4]	7.605	0.8089	0.0057	6.02	7.62	9.12	2001	50000
mu[5]	10.27	0.9251	0.006866	8.619	10.27	11.98	2001	50000
mu[6]	11.87	0.603	0.002958	10.71	11.87	13.03	2001	50000