Bayesian Synthesis Models part 2

Jingchen (Monika) Hu

Vassar College

Statistical Data Privacy

Outline

Outline

Recap

- Lecture 4
  - Synthesizing continuous outcome variables: Bayesian simple / multiple linear regressions
  - Synthesizing binary outcome variables: Bayesian logistic regression
  - In both cases
    - ★ Prior (default priors in `brms`)
    - ★ MCMC estimation and diagnostics
    - ★ Posterior predictions for synthetic data generation (writing R functions)
    - ★ Simple utility checks

The CE sample

- Our sample is from the 1st quarter of 2019, containing five variables on 5133 CUs

Variable Name	Variable information
UrbanRural	Binary; the urban / rural status of CU: 1 = Urban, 2 = Rural.
Income	Continuous; the amount of CU income before taxes in past 12 months (in *USD*).
Race	Categorical; the race category of the reference person: 1 = White, 2 = Black, 3 = Native American, 4 = Asian, 5 = Pacific Islander, 6 = Multi-race.
Expenditure	Continuous; CU's total expenditures in last quarter (in *USD*).
KidsCount	Count; the number of CU members under age 16.

Plan for this lecture

- Synthesis models for unordered / nominal categorical outcome variables (e.g., Race)
- Synthesis models for count outcome variables (e.g., KidsCount) within **sequential synthesis**
- A joint synthesis model for multivariate categorical data (the DPMPM model (Hu, Reiter, and Wang (2014))) for an ACS sample

Outline

A Bayesian multinomial logistic regression: model specification

- Let Y_i be the Race variable for CU i, taking values from 1 to C (C refers to the number of levels and $C = 6$ in our CE sample)
- We work with a vectorized version of Y_i, denoted as $\tilde{\boldsymbol{Y}}_i$, where $\tilde{\boldsymbol{Y}}_i = (\tilde{Y}_{i1}, \cdots, \tilde{Y}_{iC})$ with $(C-1)$ 0's and one 1 at the occupied category

A Bayesian multinomial logistic regression: model specification

- Let Y_i be the Race variable for CU i, taking values from 1 to C (C refers to the number of levels and $C = 6$ in our CE sample)
- We work with a vectorized version of Y_i, denoted as $\tilde{\boldsymbol{Y}}_i$, where $\tilde{\boldsymbol{Y}}_i = (\tilde{Y}_{i1}, \cdots, \tilde{Y}_{iC})$ with $(C-1)$ 0's and one 1 at the occupied category

- A Bayesian multinomial logistic regression model:

$$\{\tilde{\boldsymbol{Y}}_i\} \overset{ind}{\sim} \text{Multinomial}(1; p_{i1}, \cdots, p_{iC}), \tag{1}$$

where p_{ic} is the probability of observation i taking $\tilde{Y}_{ic} = 1$ (i.e. $Y_i = c$), and the 1 indicates 1 trial of this multionmial experiment; $\sum_{c'=1}^{C} p_{ic'} = 1$ for each i

A Bayesian multinomial logistic regression: model specification

- The multinomial logistic regression is a generalization of binary logistic regression to the multi-categorical outcome case

A Bayesian multinomial logistic regression: model specification

- The multinomial logistic regression is a generalization of binary logistic regression to the multi-categorical outcome case
- Assume r predictor variables, $X_{i1}, \cdots, X_{ir}$, we can define the **log odds ratio** for category c relative to category 1 as:

$$\log\left(\frac{p_{ic}}{p_{i1}}\right) = \beta_{0c} + \beta_{1c}X_{i1} + \cdots + \beta_{rc}X_{ir}, \tag{2}$$

with $r+1$ parameters, $\{\beta_{0c}, \cdots, \beta_{rc}\}$ for each category $c > 1$

A Bayesian multinomial logistic regression: model specification

- The multinomial logistic regression is a generalization of binary logistic regression to the multi-categorical outcome case
- Assume r predictor variables, $X_{i1}, \cdots, X_{ir}$, we can define the **log odds ratio** for category c relative to category 1 as:

$$\log\left(\frac{p_{ic}}{p_{i1}}\right) = \beta_{0c} + \beta_{1c}X_{i1} + \cdots + \beta_{rc}X_{ir}, \tag{2}$$

  with $r+1$ parameters, $\{\beta_{0c}, \cdots, \beta_{rc}\}$ for each category $c > 1$
- Note that the first category, $c = 1$ of Equation (2), is used as a **baseline**; i.e., Equation (2) $= 0$ when $c = 1$

Discussion question: How many parameters do we have in this model?

A Bayesian multinomial logistic regression: synthesis details

- After some algebra transformation, we can express the probability p_{ic} of record i falling into category c as:

$$p_{ic} = \frac{\exp(\beta_{0c} + \beta_{1c}X_{i1} + \cdots + \beta_{rc}X_{ir})}{\sum_{c'=1}^{C} \exp(\beta_{0c'} + \beta_{1c'}X_{i1} + \cdots + \beta_{rc'}X_{ir})}, \tag{3}$$

  with the constraint that $\exp(\beta_{01} + \beta_{11}X_{i1} + \cdots + \beta_{r1}X_{ir}) = 1$ for the baseline level $c = 1$
- This model is also more specifically referred to as a **baseline-category logit model** (see details in Chapter 8 of Agresti (2012))
- This transformation will be used in our synthesis step later

A Bayesian multinomial logistic regression: synthesis details

- MCMC estimation produces a set of posterior parameter draws, denoted as $\{\beta^*_{0c}, \cdots, \beta^*_{rc}\}$ for $c > 1$
- For notation simplicity, let $p^*_{ic} = h(\beta^*_{0c}, \cdots, \beta^*_{rc}, \boldsymbol{X}_i)$

$$
\begin{aligned}
\text{compute } p^*_{1c} &= h(\beta^*_{0c}, \cdots, \beta^*_{rc}, \boldsymbol{X}_1), \forall c > 1 \rightarrow \\
&\quad \text{sample } \tilde{Y}^*_1 \sim \text{Multinomial}(1; p^*_{11}, \cdots, p^*_{1C}) \\
&\vdots \\
\text{compute } p^*_{ic} &= h(\beta^*_{0c}, \cdots, \beta^*_{rc}, \boldsymbol{X}_i), \forall c > 1 \rightarrow \\
&\quad \text{sample } \tilde{Y}^*_i \sim \text{Multinomial}(1; p^*_{i1}, \cdots, p^*_{iC}) \\
&\vdots \\
\text{compute } p^*_{nc} &= h(\beta^*_{0c}, \cdots, \beta^*_{rc}, \boldsymbol{X}_n), \forall c > 1 \rightarrow \\
&\quad \text{sample } \tilde{Y}^*_n \sim \text{Multinomial}(1; p^*_{n1}, \cdots, p^*_{nC}).
\end{aligned}
$$

A Bayesian multinomial logistic regression: synthesis details

- Here $p_{i1}^* = 1$ for every i
- From the vectorized $\tilde{Y}_i^*$, we can obtain scalar Y_i^*, and therefore we have simulated one synthetic vector $(Y_i^*)_{i=1,\cdots,n}$

A Bayesian multinomial logistic regression: model estimation

- We wish to synthesize categorical variable Race with LogIncome as the single predictor
- The multinomial logistic regression synthesis model can then be set up as follows.

$$\tilde{Y}_i \overset{ind}{\sim} \text{Multinomial}(1; p_{i1}, \cdots, p_{iC}), \tag{4}$$

$$\log\left(\frac{p_{ic}}{p_{i1}}\right) = \beta_{0c} + \beta_{1c} X_i, \tag{5}$$

  where $\tilde{Y}_i$ is the vectorized Race outcome variable and X_i is the LogIncome predictor variable
- $\{\beta_{0c}, \beta_{1c}\}$ are two parameters in this multinomial logistic regression model for category $c > 1$

Discussion question: How many parameters do we have in this model?

A Bayesian multinomial logistic regression: model estimation

- Load the CE dataset and create LogIncome and LogExpenditure

```
CEdata <- readr::read_csv(file = "CEdata.csv")
CEdata$LogIncome <- log(CEdata$Income)
CEdata$LogExpenditure <- log(CEdata$Expenditure)
```

- To streamline our synthesis process, we create the **design matrix $\boldsymbol{X}$** based on the chosen model

```
ml_ff <- stats::as.formula(Race ~ 1 + LogIncome)
ml_model <- stats::model.frame(ml_ff, CEdata)
ml_X <- data.frame(stats::model.matrix(ml_ff, ml_model))
```

A Bayesian multinomial logistic regression: model estimation

- Load the CE dataset and create LogIncome and LogExpenditure

```
CEdata <- readr::read_csv(file = "CEdata.csv")
CEdata$LogIncome <- log(CEdata$Income)
CEdata$LogExpenditure <- log(CEdata$Expenditure)
```

- To streamline our synthesis process, we create the **design matrix X** based on the chosen model

```
ml_ff <- stats::as.formula(Race ~ 1 + LogIncome)
ml_model <- stats::model.frame(ml_ff, CEdata)
ml_X <- data.frame(stats::model.matrix(ml_ff, ml_model))
```

- We use the **default priors**

A Bayesian multinomial logistic regression: model estimation

- We use `family = categorical(link = "logit")` in the `brm()` function to fit the multinomial logistic regression model

```
ml_fit <- brms::brm(data = CEdata,
                    family = categorical(link = "logit"),
                    Race ~ 1 + LogIncome,
                    iter = 5000,
                    warmup = 3000,
                    thin = 1,
                    chains = 1,
                    seed = 823)
```

A Bayesian multinomial logistic regression: model estimation

- The key to applying Bayesian synthesis models is to save posterior parameter draws of estimated parameters
- These draws will be used to generate synthetic data given the posterior predictive distribution
- We use the `posterior_samples()` function to retrieve the posterior parameter draws in `post_ml`

A Bayesian multinomial logistic regression: model estimation

```
post_ml <- brms::posterior_samples(x = ml_fit)
post_ml[1:3, ]
```

```
##   b_mu2_Intercept b_mu3_Intercept b_mu4_Intercept b_mu5_Intercept
## 1        1.418636       -3.519348       -4.413770       -7.419548
## 2        1.513357       -3.375913       -4.443419       -7.537551
## 3        1.014357       -3.034133       -3.916273       -6.332770
##   b_mu6_Intercept b_mu2_LogIncome b_mu3_LogIncome b_mu4_LogIncome
## 1       -2.269950      -0.3389376      -0.1799465       0.1323971
## 2       -2.188688      -0.3490482      -0.1875901       0.1371984
## 3       -3.150873      -0.2951873      -0.1782070       0.0929380
##   b_mu5_LogIncome b_mu6_LogIncome      lp__
## 1       0.2182136     -0.14867769 -3423.668
## 2       0.2262599     -0.15525579 -3423.847
## 3       0.0928044     -0.06041435 -3417.185
```

A Bayesian multinomial logistic regression: MCMC diagnostics

- Don't forget to do them!

A Bayesian multinomial logistic regression: synthesis

- Suppose we use one set of posterior draws at the `index` iteration of the MCMC

```
ml_synthesize <- function(X, post_draws, index, n, C, seed){
  set.seed(seed)
  log_p_allC <- matrix(NA, nrow = n, ncol = C)
  for (c in 2:C){
    name_Intercept_c <- paste0("b_mu", c, "_Intercept")
    name_LogIncome_c <- paste0("b_mu", c, "_LogIncome")
    log_p_c <- as.matrix(X) %*%
      t(post_draws[index, c(name_Intercept_c, name_LogIncome_c)])
    log_p_allC[, c] <- log_p_c
  }
  log_p_allC[, 1] <- rep(0, n)
```

A Bayesian multinomial logistic regression: synthesis

```
  p_allC <- exp(log_p_allC) / (1 + exp(log_p_allC))

  synthetic_Y <- rep(NA, n)
  for (i in 1:n){
    synthetic_Y[i] <- which(stats::rmultinom(n = 1, size = 1,
                                            prob = p_allC[i, ]) == 1)
  }
  data.frame(X, synthetic_Y)
}
```

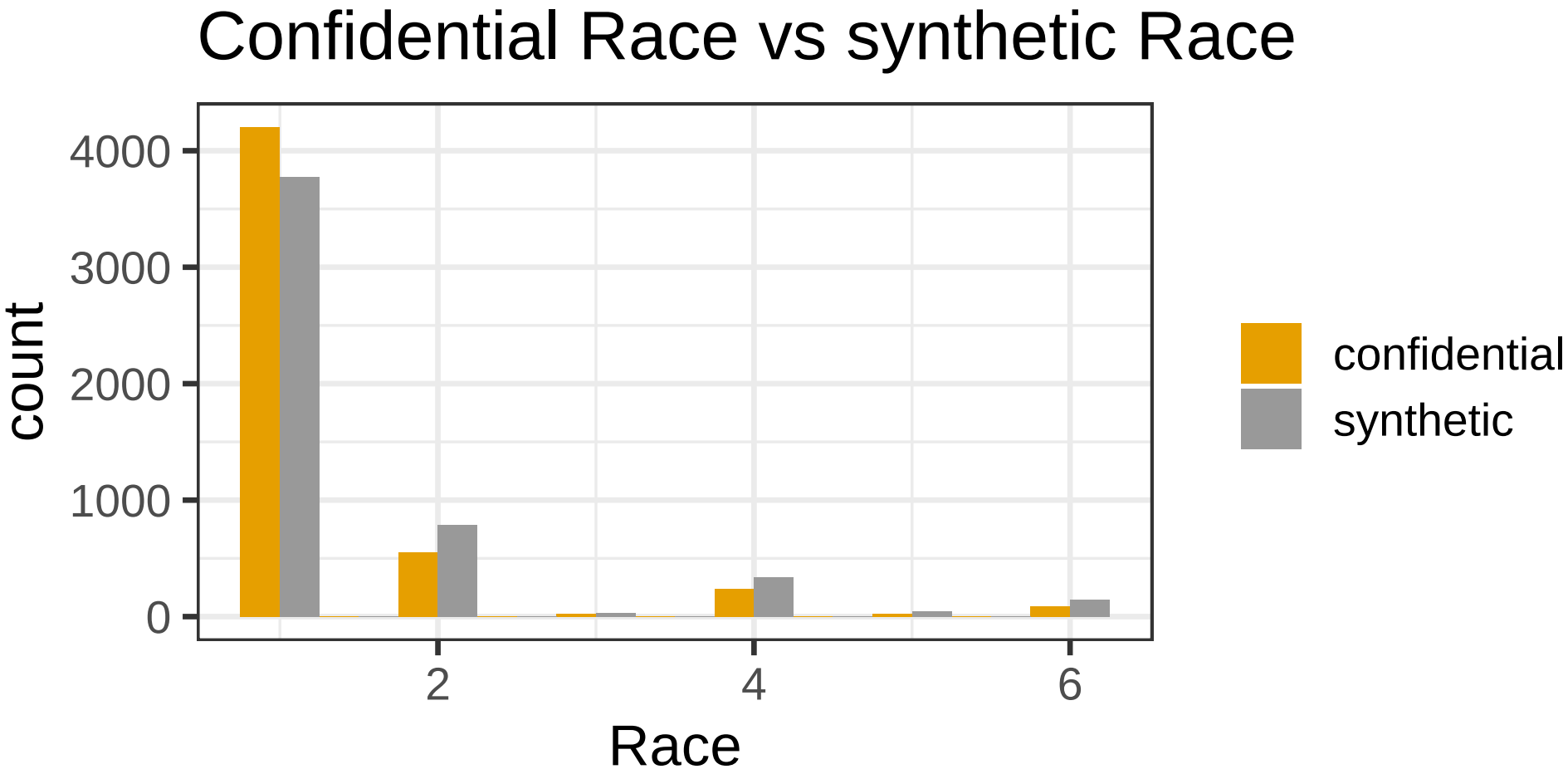
Dicussion question: Discuss the details of this function - why do we use `c in 2:C`?

A Bayesian multinomial logistic regression: synthesis

- Perform synthesis for the CE dataset

```
n <- nrow(CEdata)
ml_synthetic_one <- ml_synthesize(X = ml_X,
                                  post_draws = post_ml,
                                  index = 1,
                                  n = nrow(ml_X),
                                  C = 6,
                                  seed = 983)
names(ml_synthetic_one) <- c("Intercept", "LogIncome", "Race")
```

A Bayesian multinomial logistic regression: utility check



Additional Bayesian models for categorical variables

- Unordered / nominal categorical variables
  - Dirichlet-multinomial conjugate model

Additional Bayesian models for categorical variables

- Unordered / nominal categorical variables
  - Dirichlet-multinomial conjugate model
- Ordered categorical variables
  - Probit model
  - Cumulative logit model
  - Proportional odds model

Outline

Within a sequential synthesis process

- Recall Poisson regression for count variable with predictors for the bike sharing dataset from Lecture 3
- In this section, suppose we need to synthesize KidsCount for privacy protection and we wish to use LogIncome and LogExpenditure as predictors
- Moreover, LogExpenditure is also considered sensitive, which is to be synthesized with predictor LogIncome (as done in Lecture 4)
- We will therefore utilize a sequential synthesis approach

Dicussion question: Discuss the process of this sequential synthesis task. Pay attention to what data is used in what step.

Within a sequential synthesis process

1. We first build a Bayesian synthesis model for LogExpenditure using predictor LogIncome. We estimate this model on the **confidential data**, and generate synthetic LogExpenditure using confidential LogIncome via linear regression synthesis model
2. We next build a Bayesian synthesis model for KidsCount using predictors LogIncome and LogExpenditure. We estimate this model on the **confidential data**, and generate synthetic KidsCount using **synthetic** LogExpenditure from step 1 and **confidential** LogIncome.

A Bayesian Poisson regression: model specification

- Let Y_i be a count random variable, which follows a Poisson distribution with record-specific parameter λ_i:

$$Y_i \overset{ind}{\sim} \text{Poisson}(\lambda_i), \tag{6}$$

where $\lambda_i > 0$ is the rate parameter of the Poisson model for record i

A Bayesian Poisson regression: model specification

- Let Y_i be a count random variable, which follows a Poisson distribution with record-specific parameter λ_i:

$$Y_i \overset{ind}{\sim} \text{Poisson}(\lambda_i), \tag{6}$$

  where $\lambda_i > 0$ is the rate parameter of the Poisson model for record i

- We can specify the logarithm of the rate parameter, $\log(\lambda_i)$ as a linear function of the predictor variables
- Assume r predictor variables, $X_{i1}, \cdots, X_{ir}$

$$\log(\lambda_i) = \beta_0 + \beta_1 X_{i1} + \cdots + \beta_r X_{ir}. \tag{7}$$

A Bayesian Poisson regression: synthesis details

- MCMC estimation of the Poisson regression synthesis model produces posterior parameter draws, denoted as $\{\beta_0^*, \beta_1^*, \cdots, \beta_r^*\}$
- If none of $X_{i1}, \cdots, X_{ir}$ are synthesized for privacy protection, then to simulate synthetic values for all n observations, we perform the following.

$$\begin{aligned}
\text{compute } \log(\lambda_1^*) &= \beta_0^* + \beta_1^* X_{11} + \cdots + \beta_r^* X_{1r} \rightarrow \\
&\quad \text{sample } Y_1^* \sim \text{Poisson}(\lambda_1^*) \\
&\vdots \\
\text{compute } \log(\lambda_i^*) &= \beta_0^* + \beta_1^* X_{i1} + \cdots + \beta_r^* X_{ir} \rightarrow \\
&\quad \text{sample } Y_i^* \sim \text{Poisson}(\lambda_1^*) \\
&\vdots \\
\text{compute } \log(\lambda_n^*) &= \beta_0^* + \beta_1^* X_{n1} + \cdots + \beta_r^* X_{nr} \rightarrow \\
&\quad \text{sample } Y_n^* \sim \text{Poisson}(\lambda_n)
\end{aligned}$$

A Bayesian Poisson regression: synthesis details

- We will then have simulated one synthetic vector $(Y_i^*)_{i=1,\cdots,n}$.
- Discussion question: What happens if the last variable, X_{ir}, is also sensitive and synthesized beforehand?

A Bayesian Poisson regression: synthesis details

- In such case, we do the following to simulate synthetic Y_i^* for all n observations:

$$\begin{aligned}
\text{compute } \log(\lambda_1^*) &= \beta_0^* + \beta_1^* X_{11} + \cdots + \beta_{r-1}^* X_{1(r-1)} + \beta_r^* X_{1r}^* \rightarrow \\
&\quad \text{sample } Y_1^* \sim \text{Poisson}(\lambda_1^*) \\
&\vdots \\
\text{compute } \log(\lambda_i^*) &= \beta_0^* + \beta_1^* X_{i1} + \cdots + \beta_{r-1}^* X_{i(r-1)} + \beta_r^* X_{ir}^* \rightarrow \\
&\quad \text{sample } Y_1^* \sim \text{Poisson}(\lambda_1^*) \\
&\vdots \\
\text{compute } \log(\lambda_n^*) &= \beta_0^* + \beta_1^* X_{n1} + \cdots + \beta_{r-1}^* X_{n(r-1)} + \beta_r^* X_{nr}^* \rightarrow \\
&\quad \text{sample } Y_n^* \sim \text{Poisson}(\lambda_n^*)
\end{aligned}$$

A Bayesian Poisson regression: synthesis details

- Both **synthetic** $(X^*_{ir})_{i=1,\cdots,n}$ and **confidential** $(X_{i1}, \cdots, X_{i(r-1)})_{i=1,\cdots,n}$ are used, due to the process of sequential synthesis
- We will have simulated one synthetic vector $(Y^*_i)_{i=1,\cdots,n}$

A Bayesian Poisson regression: model estimation

- We wish to synthesize count variable KidsCount with LogIncome and LogExpenditure as two predictors
- Moreover, LogExpenditure is sensitive and synthesized given LogIncome

A Bayesian Poisson regression: model estimation

- We wish to synthesize count variable KidsCount with LogIncome and LogExpenditure as two predictors
- Moreover, LogExpenditure is sensitive and synthesized given LogIncome

- Let Y_i be the KidsCount of CU i, X_{i1} be the LogIncome and X_{i2} be the LogExpenditure of CU i

$$Y_i \overset{ind}{\sim} \text{Poisson}(\lambda_i), \tag{8}$$

$$\log(\lambda_i) = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2}, \tag{9}$$

Discussion question: How many parameters do we have in this model? What are they?

A Bayesian Poisson regression: model estimation

- We need to first fit the SLR synthesis model from Lecture 4
- To streamline our synthesis process, we create the **synthetic design matrix $\boldsymbol{X}^*$** based on the chosen model

```
CEdata$LogExpenditure_synthetic <- SLR_synthetic_one$LogExpenditure
Poisson_ff <- stats::as.formula(KidsCount ~ 1 + LogIncome +
                                  LogExpenditure_synthetic)
Poisson_model <- stats::model.frame(Poisson_ff, CEdata)
Poisson_X_star <- data.frame(stats::model.matrix(Poisson_ff,
                                                 Poisson_model))
```

- We use the **default priors**

A Bayesian Poisson regression: model estimation

- We use `family = poisson(link = "log")` in the `brm()` function to fit the Poisson regression model

```
Poisson_fit <- brms::brm(data = CEdata,
                         family = poisson(link = "log"),
                         KidsCount ~ 1 + LogIncome + LogExpenditure,
                         iter = 5000,
                         warmup = 3000,
                         thin = 1,
                         chains = 1,
                         seed = 398)
```

- Note that here the model expression is `KidsCount ~ 1 + LogIncome + LogExpenditure` with `LogExpenditure`; previously for the synthetic design matrix, we use `LogExpenditure_synthetic`

A Bayesian Poisson regression: model estimation

- The key to applying Bayesian synthesis models is to save posterior parameter draws of estimated parameters
- These draws will be used to generate synthetic data given the posterior predictive distribution
- We use the `posterior_samples()` function to retrieve the posterior parameter draws in `post_Poisson`

```
post_Poisson <- brms::posterior_samples(x = Poisson_fit)
post_Poisson[1:3, ]
```

```
##   b_Intercept b_LogIncome b_LogExpenditure      lp__
## 1   -4.564058   0.1734750        0.2244817 -5216.570
## 2   -4.378532   0.1684752        0.2043459 -5215.492
## 3   -4.603462   0.1706977        0.2292895 -5214.753
```

A Bayesian Poisson regression: MCMC diagnostics

- Don't forget to do them!

A Bayesian Poisson regression: synthesis

- Suppose we use one set of posterior draws at the `index` iteration of the MCMC

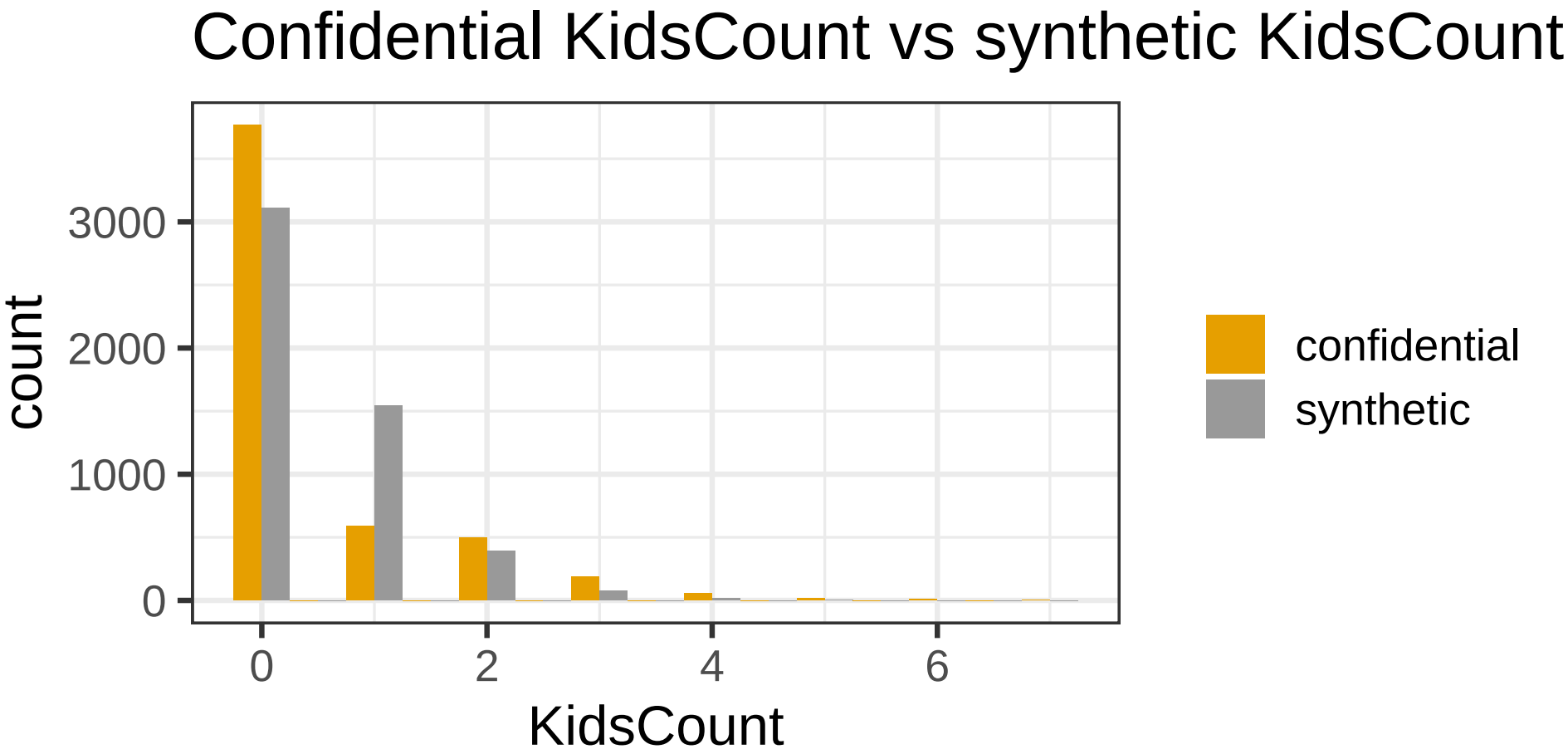
```
Poisson_synthesize <- function(X, post_draws, index, n, seed){
  set.seed(seed)
  lambda_log <- as.matrix(X) %*%
    t(data.matrix(post_draws[index, c("b_Intercept", "b_LogIncome",
                                      "b_LogExpenditure")]))
  synthetic_Y <- stats::rpois(n, exp(lambda_log))
  data.frame(X, synthetic_Y)
}
```

A Bayesian Poisson regression: synthesis

- Now to synthesize KidsCount with **synthetic** LogExpenditure and **confidential** LogIncome, we supply `X_star` which is the synthetic design matrix including the desired values: synthetic for LogExpenditure and confidential for LogIncome

```
n <- nrow(CEdata)
Poisson_ss_synthetic_one <- Poisson_synthesize(X = Poisson_X_star,
                                               post_draws = post_Poisson,
                                               index = 1,
                                               n = nrow(Poisson_X_star),
                                               seed = 653)
names(Poisson_ss_synthetic_one) <- c("Intercept", "LogIncome",
                                     "LogExpenditure", "KidsCount")
```

A Bayesian Poisson regression: utility check



A note on multiple synthetic datasets

- If multiple synthetic datasets are needed, for example $m = 20$, we can then repeat this process m times using m independent MCMC iterations
- A caveat in creating multiple synthetic datasets within sequential synthesis is that we need to create the synthetic design matrix for the second step modeling in the loop over $m = 20$ since that design matrix depends on the synthetic predictor simulated from the first step

Outline

The ACS sample

- Our sample is from 2012 ACS public use microdata, containing ten categorical variables on 10,000 observations

Variable name	Variable information
SEX	1 = male, 2 = female.
RACE	1 = White alone, 2 = Black or African American alone, 3 = American Indian alone, 4 = other, 5 = two or more races, 6 = Asian alone.
MAR	1 = married, 2 = widowed, 3 = divorced, 4 = separated, 5 = never married.
LANX	1 = speaks another language, 2 = speaks only English
WAOB	born in: 1 = US state, 2 = PR and US island areas, oceania and at sea, 3 = Latin America, 4 = Asia, 5 = Europe, 6 = Africa, 7 = Northern America.

The ACS sample

Variable name	Variable information
DIS	1 = has a disability, 2 = no disability.
HICOV	1 = has health insurance coverage, 2 = no coverage.
MIG	1 = live in the same house (non movers), 2 = move to outside US and PR, 3 = move to different house in US or PR.
SCH	1 = has not attended school in the last 3 months, 2 = in public school or college, 3 = in private school or college or home school.
HISP	1 = not Spanish, Hispanic, or Latino, 2 = Spanish, Hispanic, or Latino.

The DPMPM: overview

- A popular joint approach to synthesizing multivariate (unordered / nominal) categorical data is the Dirichlet Process mixture of products of multinomials (DPMPM)
- Hu, Reiter, and Wang (2014) first applied it to a sample of the American Community Survey (ACS) with 14 categorical variables and produced fully synthetic datasets
- Hu and Savitsky (2018) and Drechsler and Hu (2021) utilized DPMPM to synthesize geographic labels for privacy protection and produced partially synthetic data

The DPMPM: overview

- A popular joint approach to synthesizing multivariate (unordered / nominal) categorical data is the Dirichlet Process mixture of products of multinomials (DPMPM)
- Hu, Reiter, and Wang (2014) first applied it to a sample of the American Community Survey (ACS) with 14 categorical variables and produced fully synthetic datasets
- Hu and Savitsky (2018) and Drechsler and Hu (2021) utilized DPMPM to synthesize geographic labels for privacy protection and produced partially synthetic data

- The R package `NPBayesImputeCat` (Wang et al. (2021)) is available to implement the model estimation and synthesis steps

The DPMPM: model specification

- Consider a sample $\mathbf{Y}$ consisting of n records, where each ith record, with $i = 1, \ldots, n$, has r unordered categorical variables
- The basic assumption of the DPMPM is that every record $\mathbf{Y}_i = (Y_{i1}, \cdots, Y_{ir})$ belongs to one of K underlying unobserved **latent classes**
  - A latent class includes observations being classified into that class but the class assignment is unobserved and typically requires model to estimate

The DPMPM: model specification

- Consider a sample $\mathbf{Y}$ consisting of n records, where each ith record, with $i = 1, \ldots, n$, has r unordered categorical variables
- The basic assumption of the DPMPM is that every record $\mathbf{Y}_i = (Y_{i1}, \cdots, Y_{ir})$ belongs to one of K underlying unobserved **latent classes**
  - A latent class includes observations being classified into that class but the class assignment is unobserved and typically requires model to estimate

- Given the latent class assignment z_i of record i each variable Y_{ij} independently follows a multinomial distribution, where d_j is the number of categories of variable j, and $j = 1, \ldots, r$

$$Y_{ij} \mid z_i, \theta \overset{ind}{\sim} \text{Multinomial}(\theta^{(j)}_{z_i 1}, \ldots, \theta^{(j)}_{z_i d_j}; 1) \quad \forall i, j, \tag{10}$$

$$z_i \mid \pi \sim \text{Multinomial}(\pi_1, \ldots, \pi_K; 1) \quad \forall i \tag{11}$$

The DPMPM: model specification

- The marginal probability of $Pr(Y_{i1} = y_{i1}, \cdots, Y_{ip} = y_{ip} \mid \pi, \theta)$ can be expressed as averaging over the latent classes:

$$Pr(Y_{i1} = y_{i1}, \cdots, Y_{ip} = y_{ir} \mid \pi, \theta) = \sum_{k=1}^{K} \pi_k \prod_{j=1}^{r} \theta_{ky_{ij}}^{(j)} \qquad (12)$$

The DPMPM: model specification

- The marginal probability of $Pr(Y_{i1} = y_{i1}, \cdots, Y_{ip} = y_{ip} \mid \pi, \theta)$ can be expressed as averaging over the latent classes:

$$Pr(Y_{i1} = y_{i1}, \cdots, Y_{ip} = y_{ir} \mid \pi, \theta) = \sum_{k=1}^{K} \pi_k \prod_{j=1}^{r} \theta_{ky_{ij}}^{(j)} \qquad (12)$$

- While all variables are considered independent within each of the latent classes, such averaging over latent classes results in dependence among the variables
- The DPMPM effectively clusters records with similar characteristics based on all r variables
- Relationships among these r categorical variables are made explicit after integrating out the latent class assignment z_i

Discussion question: Can you identify the parameters in this model?

The DPMPM: model specification

- The parameters in the DPMPM model include:
  - A vector of probabilities $(\pi_1, \cdots, \pi_K)$
  - A collection of vectors of probabilities $(\theta_{k1}^{(j)}, \ldots, \theta_{kd_j}^{(j)})$, one for each combination of variable j and latent class k.
- We need to provide priors for all these parameters

The DPMPM: model specification

- The parameters in the DPMPM model include:
  - A vector of probabilities $(\pi_1, \cdots, \pi_K)$
  - A collection of vectors of probabilities $(\theta_{k1}^{(j)}, \ldots, \theta_{kd_j}^{(j)})$, one for each combination of variable j and latent class k.
- We need to provide priors for all these parameters

- The choice of K
  - We select a large enough K, which serves as the upper bound of possible latent classes to be used
  - As long as K is set to a large enough value, we allow the DPMPM to fully explore the parameter space and determine the **effective number of occupied latent classes**

The DPMPM: model specification

- To empower the DPMPM to pick the effective number of occupied latent classes, the **truncated stick-breaking representation** (Sethuraman (1994)) of the Dirichlet Process prior is used as

$$\pi_k = V_k \prod_{l<k}(1-V_l) \quad \text{for } k = 1,\ldots,K, \tag{13}$$

$$V_k \overset{iid}{\sim} \text{Beta}(1,\alpha) \quad \text{for } k = 1,\ldots,K-1, \quad V_K = 1, \tag{14}$$

$$\alpha \sim \text{Gamma}(a_\alpha, b_\alpha), \tag{15}$$

$$\boldsymbol{\theta}_k^{(j)} = (\theta_{k1}^{(j)},\ldots,\theta_{kd_j}^{(j)}) \sim \text{Dirichlet}(a_1^{(j)},\ldots,a_{d_j}^{(j)}) \tag{16}$$

$$\text{for } j = 1,\ldots,r, \quad k = 1,\ldots,K. \tag{17}$$

The DPMPM: model specification

- To empower the DPMPM to pick the effective number of occupied latent classes, the **truncated stick-breaking representation** (Sethuraman (1994)) of the Dirichlet Process prior is used as

$$\pi_k = V_k \prod_{l<k}(1-V_l) \text{ for } k = 1,\ldots,K, \tag{13}$$

$$V_k \overset{iid}{\sim} \text{Beta}(1,\alpha) \text{ for } k = 1,\ldots,K-1, \quad V_K = 1, \tag{14}$$

$$\alpha \sim \text{Gamma}(a_\alpha, b_\alpha), \tag{15}$$

$$\boldsymbol{\theta}_k^{(j)} = (\theta_{k1}^{(j)},\ldots,\theta_{kd_j}^{(j)}) \sim \text{Dirichlet}(a_1^{(j)},\ldots,a_{d_j}^{(j)}) \tag{16}$$

$$\text{for } j = 1,\ldots,r, \quad k = 1,\ldots,K. \tag{17}$$

- Dirichlet prior is conjugate for multinomial data model
- A **blocked Gibbs sampler** is implemented for the MCMC sampling procedure (Ishwaran and James (2001))

The DPMPM: synthesis details

- When used as a data synthesis model, the confidential dataset is used for model estimation through MCMC
- Sensitive variable values are synthesized as an extra step at chosen MCMC iteration

The DPMPM: synthesis details

- When used as a data synthesis model, the confidential dataset is used for model estimation through MCMC
- Sensitive variable values are synthesized as an extra step at chosen MCMC iteration

- For example, at MCMC iteration l, we first sample a value of the latent class indicator z_i; given the sampled z_i, we then sample synthetic values of sensitive variables using independent draws; this process is repeated for every record that has sensitive values to be synthesized, obtaining one synthetic dataset

The DPMPM: model estimation

- Make sure to install and load the `NPBayesImputeCat` library

```
library(NPBayesImputeCat)
```

- Load the ACS sample and make sure all variables are categorical with a known number of levels

```
ACSdata <- data.frame(readr::read_csv(file = "ACSdata.csv"))

dj <- c(2, 6, 5, 2, 7, 2, 2, 3, 3, 2)
r <- ncol(ACSdata)
# make sure all variables are categorical
for (j in 1:r){
  ACSdata[, j] <- factor(ACSdata[, j], levels = 1:dj[j])
}
```

The DPMPM: model estimation and synthesis

- Create the DPMPM model

```
model <- NPBayesImputeCat::CreateModel(X = ACSdata,
                                       MCZ = NULL,
                                       K = 80,
                                       Nmax = 0,
                                       aalpha = 0.25,
                                       balpha = 0.25,
                                       seed = 221)
```

The DPMPM: model estimation and synthesis

- Use the `DPMPM_nozeros_syn()` function to create `m = 5` partially synthetic datasets

```
m <- 5
ACSdata_syn <- NPBayesImputeCat::DPMPM_nozeros_syn(X = ACSdata,
                                                   dj = dj,
                                                   nrun = 10000,
                                                   burn = 5000,
                                                   thin = 10,
                                                   K = 80,
                                                   aalpha = 0.25,
                                                   balpha = 0.25,
                                                   m = m,
                                                   vars = c("DIS", "HICOV"),
                                                   seed = 221,
                                                   silent = TRUE)
```

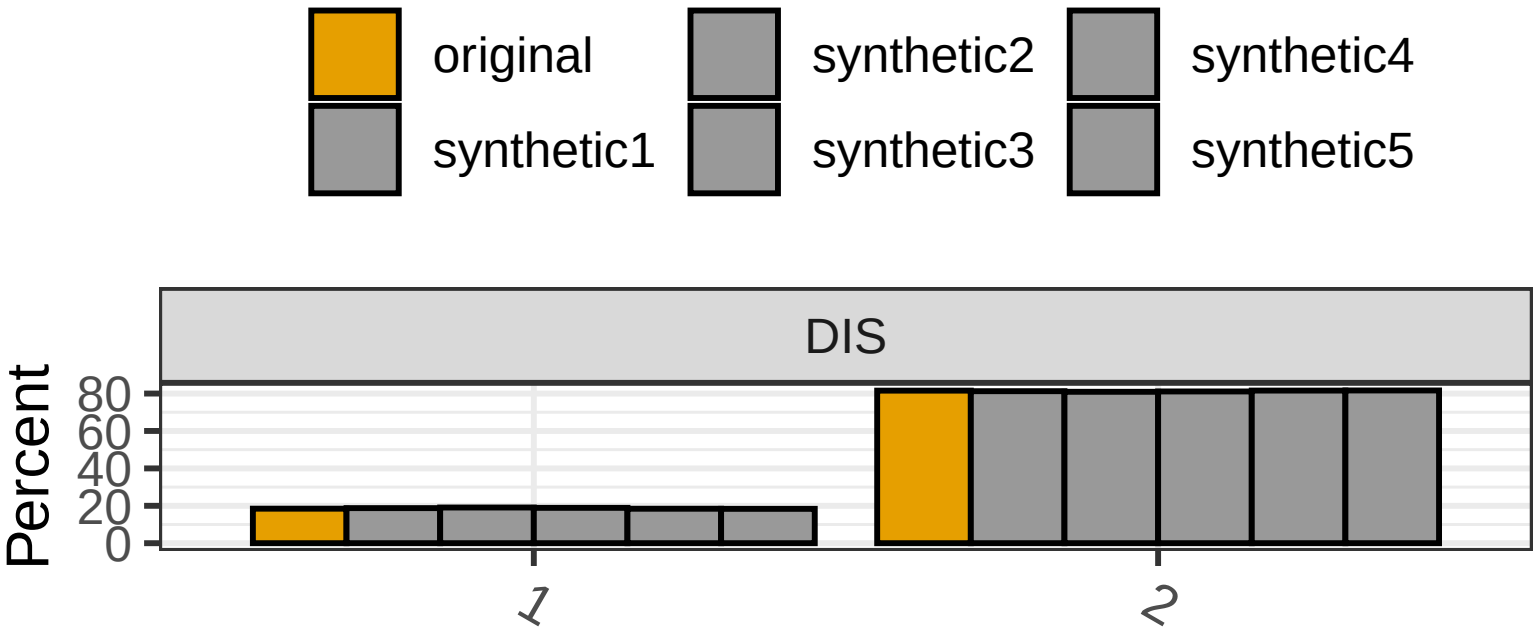
The DPMPM: MCMC diagnostics

- We can focus on diagnosis of `kstar`
- The code below produces a traceplot and an autocorrelation plot

```
library(bayesplot)
kstar_MCMCdiag(kstar = ACSdata_syn$kstar,
               nrun = 10000,
               burn = 5000,
               thin = 50)
```

The DPMPM: utility checks

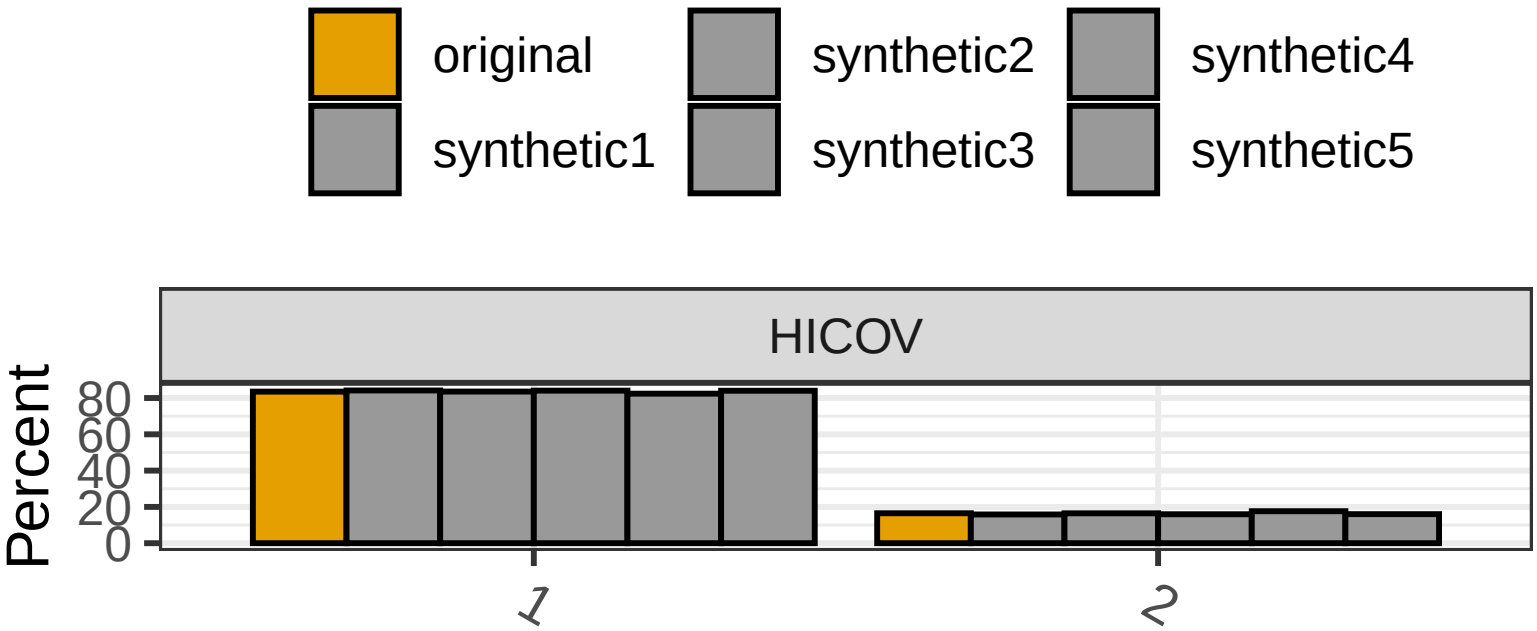
```
## $Plot
```




```
##
## $Comparison
##   level original synthetic1 synthetic2 synthetic3 synthetic
## 1     1    18.46       18.74       19.07       18.89       18.4
## 2     2    81.54       81.26       80.93       81.11       81.5
```

The DPMPM: utility checks

```
## $Plot
```




```
##
## $Comparison
##   level original synthetic1 synthetic2 synthetic3 synthetic
## 1     1     83.5       84.16       83.53       84.05      82.3
## 2     2     16.5       15.84       16.47       15.95      17.6
```

Outline

Summary

- Synthesizing categorical outcome variables
  - Bayesian multinomial logistic regressions
- Synthesizing count outcome variables
  - Bayesian Poisson regressions
  - Examples for sequential synthesis
- Synthesizing multivariate categorical data
  - The DPMPM
  - Use the `NPBayesImputCat` R package

Summary

- Synthesizing categorical outcome variables
  - Bayesian multinomial logistic regressions
- Synthesizing count outcome variables
  - Bayesian Poisson regressions
  - Examples for sequential synthesis
- Synthesizing multivariate categorical data
  - The DPMPM
  - Use the `NPBayesImputCat` R package
- Homework 5: a few R programming exercises
  - Submission on Moodle and prepare to discuss next time
- Lecture 6: Methods for utility evaluation part 1
  - All sections of Woo et al. (2009)

References I

Agresti, A. 2012. Categorical Data Analysis. Wiley; 3rd edition.

Drechsler, J., and J. Hu. 2021. "Synthesizing Geocodes to Facilitate Access to Detailed Geographical Information in Large Scale Administrative Data." Journal of Survey Statistics and Methodology 9 (3): 523–48.

Hu, J., J. P. Reiter, and Q. Wang. 2014. "Disclosure Risk Evaluation for Fully Synthetic Categorical Data." Privacy in Statistical Databases, 185–99.

Hu, J., and T. D. Savitsky. 2018. "Bayesian Data Synthesis and Disclosure Risk Quantification: An Application to the Consumer Expenditure Surveys." arXiv: 1809.10074.

Ishwaran, H., and L. F. James. 2001. "Gibbs Sampling Methods for Stick-Breaking Priors." Journal of the American Statistical Association 96: 161–73.

References II

Sethuraman, J. 1994. "A Constructive Definition of Dirichlet Priors." Statistica Sinica 4: 639–50.

Wang, Q., D. Manrique-Vallier, J. P. Reiter, and J. Hu. 2021. NPBayesImputeCat: Non-Parametric Bayesian Multiple Imputation for Categorical Data. R Package Version 0.3.

Woo, M. J., J. P. Reiter, A. Oganian, and A. F. Karr. 2009. "Global Measures of Data Utility for Microdata Masked for Disclosure Limitation." The Journal of Privacy and Confidentiality 1 (1): 111–24.