Supplemental Analyses

Jennah Gosciak

May 17, 2022

```
## load data
df <- read_dta("../00_data/sample1.dta")
df_samp <- df %>%
  sample_n(1000)

df_samp <- df_samp %>%
  mutate(across(c("age", "age_fbirth"), ~ . - mean(., na.rm = T))) %>%
  mutate(l_incwage = if_else(incwage <= 0, log(1), log(incwage)),
         l_wkswork1 = if_else(wkswork1 <= 0, log(1), log(wkswork1)),
         incwage_mod = if_else(incwage <= 0, 1, incwage))

df_samp %>%
  group_by(samesex) %>%
  summarize(n = n())
```

```
## # A tibble: 2 x 2
##   samesex     n
##     <dbl> <int>
## 1       0   473
## 2       1   527
```

Model with Gamma distribution

- I explored using the gamma distribution with a log link function since the income data is right-skewed and positive.
- This should be similar to the implementation in `rstanarm`
- Because it takes so long to run, I only ran the model on 1,000 observations.

```
writeLines(readLines("linear_gamma.stan"))
```

```
## #include quantile_functions.stan
## data {
##   int<lower = 0> N; // number of observations
##   int<lower = 0> K; // number of predictors
##   matrix[N, K] X;   // matrix of predictors
##   vector[N] y;      // outcomes
##   int<lower = 0, upper = 1> prior_only;   // ignore data?
##   vector[K + 1] m;                        // prior medians
##   vector<lower = 0>[K + 1] scale;             // prior IQRs
##   real r;
```

```
## }
## parameters {
##   vector[K] beta;
##   real alpha;
##   real<lower = 0> shape;
## }
##
## transformed parameters {
##   vector[N] mu = alpha + X * beta;
## }
##
## model { // log likelihood, equivalent to target += normal_lpdf(y | alpha + X * beta, sigma)
##   if (!prior_only) {
##     for (i in 1:N) target += gamma_lpdf(y | shape, shape/exp(mu[i]));
##   }
##   target += normal_lpdf(alpha | m[1],   scale[1]);
##   target += normal_lpdf(beta | m[2:K + 1], scale[2:K + 1]);
##   target += exponential_lpdf(shape | r); // exponential
## }
##
## generated quantities {
##   vector[N] log_lik;
##   vector[N] yrep;
##   {
##     for (n in 1:N) {
##       log_lik[n] = gamma_lpdf(y[n] | shape, shape / exp(mu[n]));
##       yrep[n] = gamma_rng(shape, shape / exp(mu[n]));
##     }
##   }
## }
```

```
# use normal priors
m <- rep(-0.01, 9)
s<- rep(0.1, 9)

stan_data <- list(N = nrow(df_samp), K = 8,
                                        y = df_samp$incwage,
                                        X = df_samp[, c("cnum_mt2", "age", "age_fbirth", "f_boy",
                                                  "s_boy", "r_black", "hisp", "r_oth")],
                                        prior_only = TRUE, m = m,
                                        scale = s, r = 1)
```

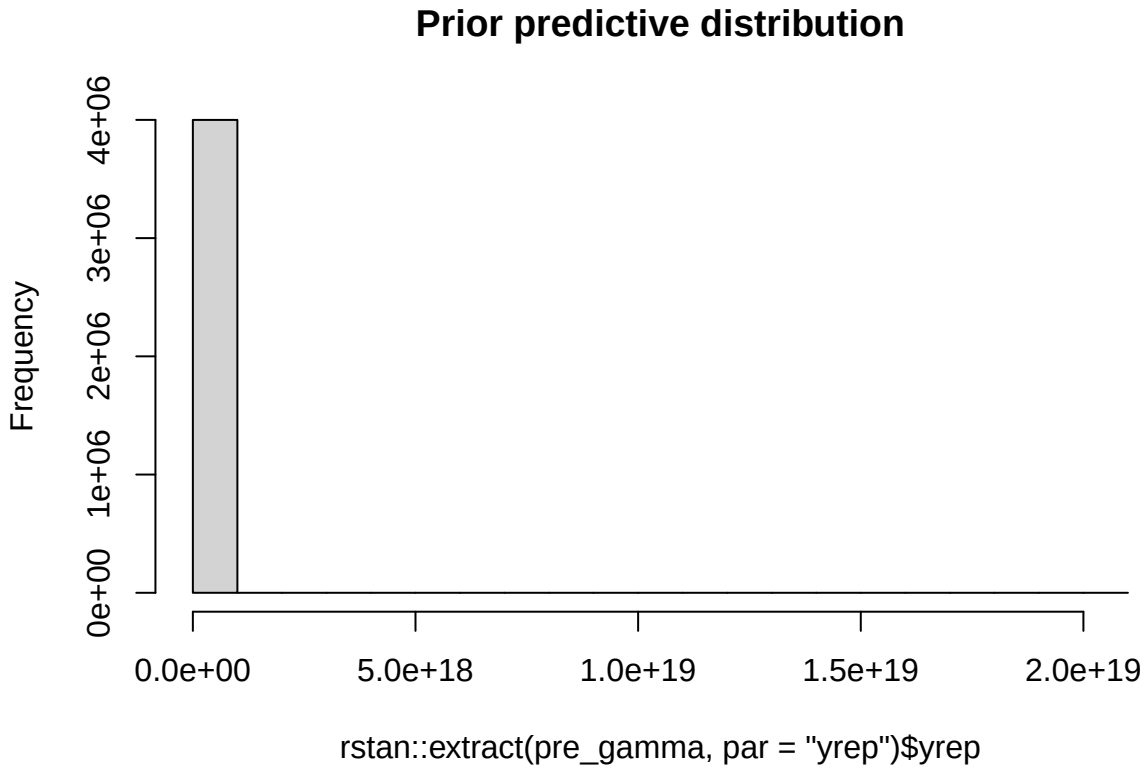
```
pre_gamma <- stan("linear_gamma.stan", data = stan_data, seed = 12345)
```

```
# print output
print(pre_gamma, pars = c("alpha", "beta", "shape"))
```

```
## Inference for Stan model: linear_gamma.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##          mean se_mean   sd  2.5%   25%   50%  75% 97.5% n_eff Rhat
## alpha   -0.11    0.01 1.01 -2.05 -0.79 -0.13 0.57  1.86  7895    1
```

```
## beta[1] -0.08    0.01 0.99 -2.02 -0.73 -0.10 0.59  1.90  7723    1
## beta[2] -0.10    0.01 0.99 -1.99 -0.75 -0.11 0.56  1.86  8044    1
## beta[3] -0.11    0.01 1.01 -2.06 -0.80 -0.12 0.58  1.86  9162    1
## beta[4] -0.10    0.01 1.01 -2.08 -0.78 -0.11 0.58  1.85  8555    1
## beta[5] -0.12    0.01 1.01 -2.17 -0.78 -0.13 0.54  1.89  8612    1
## beta[6] -0.10    0.01 1.00 -2.09 -0.78 -0.09 0.57  1.86  7752    1
## beta[7] -0.09    0.01 1.02 -2.11 -0.79 -0.08 0.61  1.91  8154    1
## beta[8] -0.11    0.01 1.02 -2.10 -0.80 -0.11 0.59  1.92  6893    1
## shape    1.99    0.02 1.97  0.05  0.56  1.38 2.78  7.16  7238    1
##
## Samples were drawn using NUTS(diag_e) at Mon May 16 22:36:55 2022.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
hist(rstan::extract(pre_gamma, par = "yrep")$yrep,
     main = "Prior predictive distribution")
```



The prior predictive distribution generates large, extreme values of y. At the same time, most observations are clustered toward 0.

```
# set prior to false, run the full dist
stan_data$prior_only <- FALSE
```

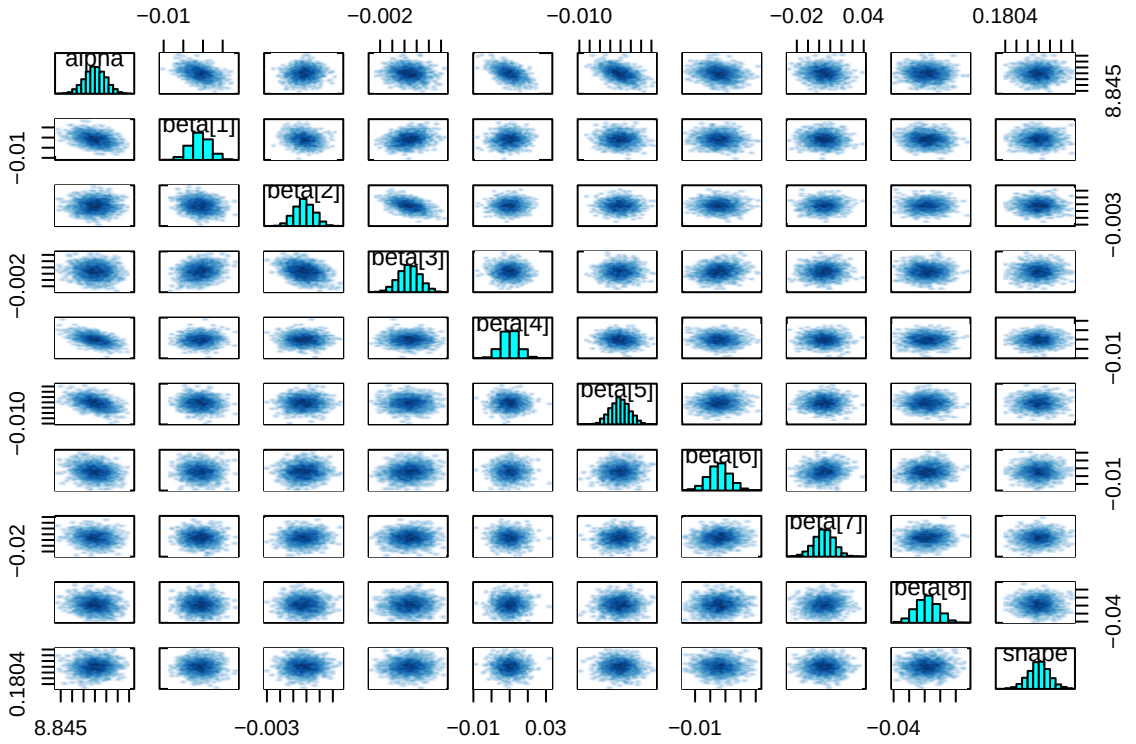
```r
post_gamma <- stan("linear_gamma.stan", data = stan_data, seed = 1234)
```

```r
# print output
print(post_gamma, pars = c("alpha", "beta", "shape"))
```

```
## Inference for Stan model: linear_gamma.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##         mean se_mean   sd  2.5%   25%  50%  75% 97.5% n_eff Rhat
## alpha   8.86       0 0.00  8.85  8.86 8.86 8.86  8.87  2337    1
## beta[1] 0.01       0 0.00  0.00  0.01 0.01 0.01  0.02  3796    1
## beta[2] 0.00       0 0.00  0.00  0.00 0.00 0.00  0.00  4501    1
## beta[3] 0.00       0 0.00  0.00  0.00 0.00 0.00  0.00  4537    1
## beta[4] 0.01       0 0.00  0.00  0.01 0.01 0.01  0.02  3472    1
## beta[5] 0.01       0 0.00  0.00  0.01 0.01 0.01  0.02  3387    1
## beta[6] 0.01       0 0.01 -0.01  0.00 0.01 0.01  0.02  3112    1
## beta[7] 0.00       0 0.01 -0.01  0.00 0.00 0.01  0.02  3122    1
## beta[8] 0.00       0 0.01 -0.02 -0.01 0.00 0.01  0.03  3652    1
## shape   0.18       0 0.00  0.18  0.18 0.18 0.18  0.18  4893    1
##
## Samples were drawn using NUTS(diag_e) at Tue May 17 01:48:50 2022.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

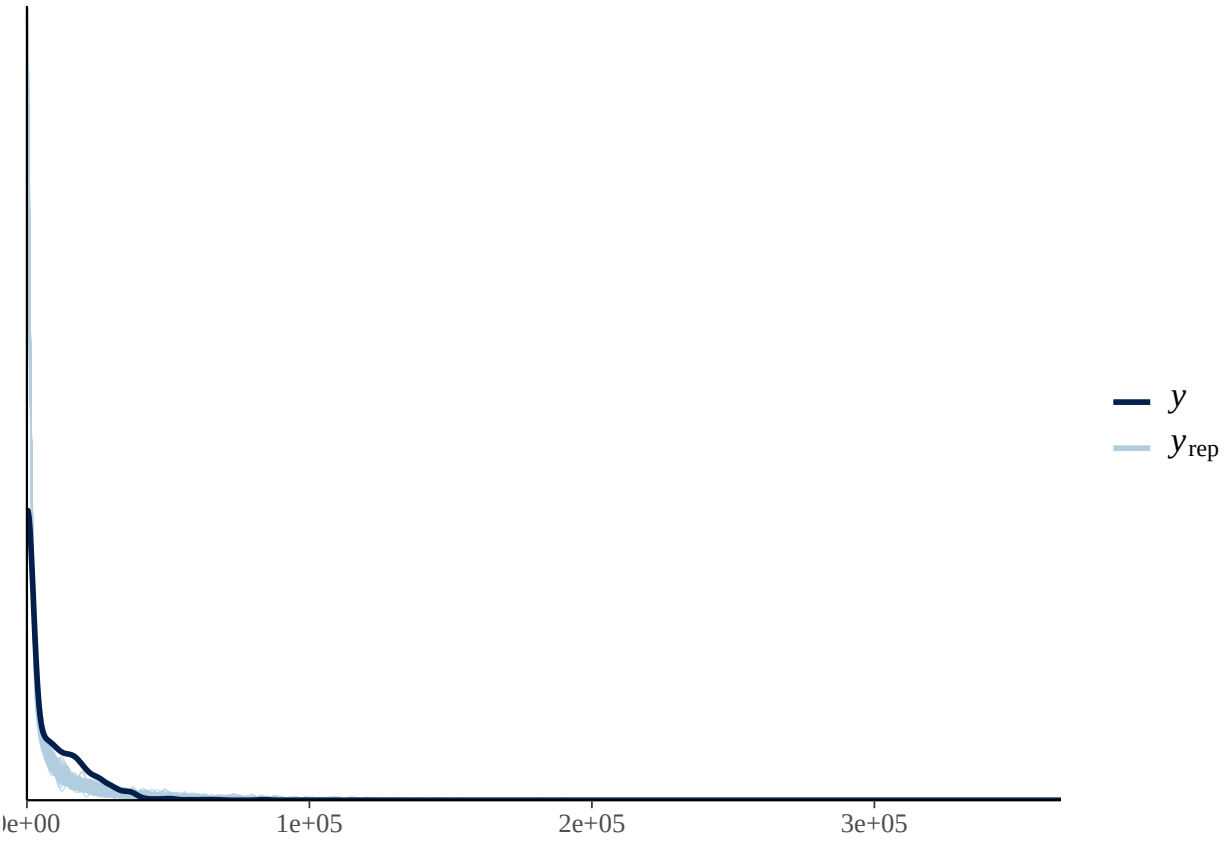
The posterior distribution when using a log link and the gamma distribution is not conclusive about the effect of additional children on earnings. The coefficient on `cnum_mt2` is on average 0 and the 95% credible interval is between -0.01 and 0.01. Almost all the predictors ahve values close to 0. The small values for the coefficients, as indicated by the marginal distributions, is likely due to the conversion from log earnings to earnings in dollars.

```r
pairs(post_gamma, pars = c("alpha", "beta", "shape"))
```



The marginal distribution of the parameters all seem approximately normal, even for shape.

```
pp_check(as.numeric(stan_data$y),
  rstan::extract(post_gamma, par = "yrep")$yrep[sample(1:length(stan_data$y), size = 150), ],
  ppc_dens_overlay
)
```



The posterior predictive distribution does appear to closely follow the observed values of y, which are less smooth. However, it's difficult to tell in the tails if the observed values and the predicted values diverge.

```
# comparison of predictions
yrep <- rstan::extract(post_gamma, "yrep")[[1]]
low  <- apply(yrep, MARGIN = 2, FUN = quantile, probs = 1 / 3)
high <- apply(yrep, MARGIN = 2, FUN = quantile, probs = 2 / 3)

y <- stan_data$y

c(too_low = mean(y < low),
  just_right = mean(y > low & y < high),
  too_high = mean(y > high))
```

```
##    too_low just_right   too_high
##      0.477      0.084      0.439
```

The model is predicting too many extreme values (bottom and top third percentiles). Only 8% of observed y-values fall in only the middle third percentile.

```
loo_gamma <- loo(post_gamma)
loo_gamma
```

```
##
## Computed from 4000 by 1000 log-likelihood matrix
```

```
##
##          Estimate    SE
## elpd_loo  -7618.5 125.8
## p_loo         0.0   0.0
## looic     15236.9 251.5
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

All the Pareto k values are less than 0.5, which indicates that importance sampling can generate a good estimate. However, `p_loo << p`, which might be evidence that the model is misspecified.

Hierarchical linear model

- Allowing intercepts to shift based on state
- Resample the data to 3,000 observations

```
df_samp <- df %>%
  sample_n(3000)

df_samp <- df_samp %>%
  mutate(across(c("age", "age_fbirth"), ~ . - mean(., na.rm = T))) %>%
  mutate(l_incwage = if_else(incwage <= 0, log(1), log(incwage)),
         l_wkswork1 = if_else(wkswork1 <= 0, log(1), log(wkswork1)),
         incwage_mod = if_else(incwage <= 0, 1, incwage))
```

```
# set generic priors
m <- rep(0, 6)
s<- rep(1, 6)

# set states as ordered factor
states <- as.integer(as.factor(df_samp$statefip))


stan_data <- list(N = nrow(df_samp), K = 5, J = 51,
                                        states = states,
                                        y = df_samp$l_incwage,
                                        X = df_samp[, c("cnum_mt2", "age",
                                                        "r_black", "hisp",
                                                        "r_oth")],
                                        prior_only = FALSE, m = m,
                                        scale = s, r = 1)

post_mlm <- stan("linear_mlm.stan", data = stan_data, seed = 1234)
```

```
print(post_mlm, pars = c("beta"))
```

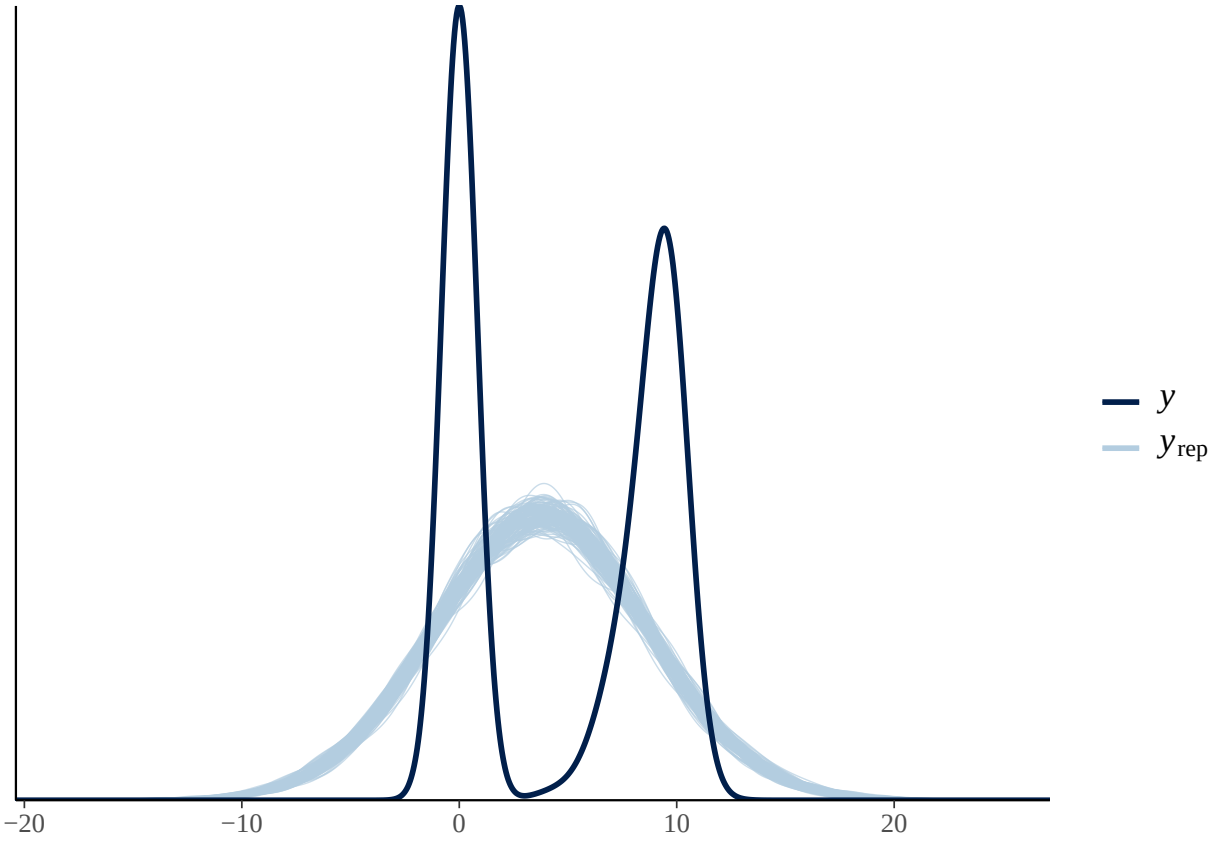
```
## Inference for Stan model: linear_mlm.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
```

```
##
##         mean se_mean   sd  2.5%  25%  50%  75% 97.5% n_eff Rhat
## beta[1] 0.41    0.00 0.17  0.09 0.29 0.41 0.52  0.75  3913    1
## beta[2] 0.12    0.00 0.02  0.07 0.10 0.12 0.13  0.17  6790    1
## beta[3] 2.41    0.00 0.27  1.89 2.22 2.40 2.59  2.95  5332    1
## beta[4] 0.75    0.00 0.31  0.14 0.54 0.75 0.97  1.35  4535    1
## beta[5] 0.52    0.01 0.51 -0.48 0.18 0.51 0.87  1.51  6083    1
##
## Samples were drawn using NUTS(diag_e) at Tue May 17 10:26:18 2022.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
loo_mlm <- loo(post_mlm)
loo_mlm
```

```
##
## Computed from 4000 by 3000 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -8894.6 18.3
## p_loo        36.8  0.8
## looic     17789.2 36.6
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
## visualize density
pp_check(as.numeric(stan_data$y),
  rstan::extract(post_mlm, par = "yrep")$yrep[sample(1:length(stan_data$y), size = 150), ],
  ppc_dens_overlay
)
```




```
# comparison of predictions
yrep <- rstan::extract(post_mlm, "yrep")[[1]]
low  <- apply(yrep, MARGIN = 2, FUN = quantile, probs = 1 / 3)
high <- apply(yrep, MARGIN = 2, FUN = quantile, probs = 2 / 3)

y <- stan_data$y

c(too_low = mean(y < low),
  just_right = mean(y > low & y < high),
  too_high = mean(y > high))
```

```
##    too_low just_right   too_high
## 0.43300000 0.05866667 0.50833333
```

Hierarchical IV model

```
# set generic priors
m <- rep(0, 9)
s<- rep(1, 9)

# set states as ordered factor
df_samp$states <- as.integer(as.factor(df_samp$statefip))
```

```r
# subset the data
df_child <- df_samp %>%
  filter(cnum_mt2 == 1)

df_nochild <- df_samp %>%
  filter(cnum_mt2 == 0)
```

```r
# reset covariates
cov <- c("age", "r_black", "hisp", "r_oth")

# set stan data
stan_data_iv <- list(N = nrow(df_samp),
                     N_child = nrow(df_child),
                     N_nochild = nrow(df_nochild),
                     K = 4,
                     J = 51,
                     states_child = df_child$states,
                     states_nochild = df_nochild$states,
                     X_child_s = df_child[, c("samesex", cov)],
                     X_nochild_s = df_nochild[, c("samesex", cov)],
                     X_child = df_child[, c(cov)],
                     X_nochild = df_nochild[, c(cov)],
                     y_child = df_child$l_incwage,
                     y_nochild = df_nochild$l_incwage,
                     prior_only = FALSE,
                     m = rep(-0.1, 5),
                     scale = rep(0.3, 5))

post_iv_mlm <- stan("iv_bin_mlm.stan",
                    data = stan_data_iv, seed = 1234)
```

```r
print(post_iv_mlm, pars = c("beta1", "beta2"))
```

```
## Inference for Stan model: iv_bin_mlm.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##          mean se_mean   sd  2.5%   25%  50%  75% 97.5% n_eff Rhat
## beta1[1] 0.05       0 0.03 -0.01  0.03 0.05 0.07  0.11  7642    1
## beta1[2] 1.20       0 0.22  0.77  1.05 1.20 1.35  1.64  7274    1
## beta1[3] 0.32       0 0.25 -0.16  0.15 0.32 0.49  0.81  7394    1
## beta1[4] 0.05       0 0.29 -0.53 -0.14 0.05 0.24  0.61  8246    1
## beta2    5.86       0 0.18  5.51  5.74 5.86 5.98  6.22  5070    1
##
## Samples were drawn using NUTS(diag_e) at Tue May 17 10:43:00 2022.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```r
loo_iv_mlm <- loo(post_iv_mlm)
loo_iv_mlm
```

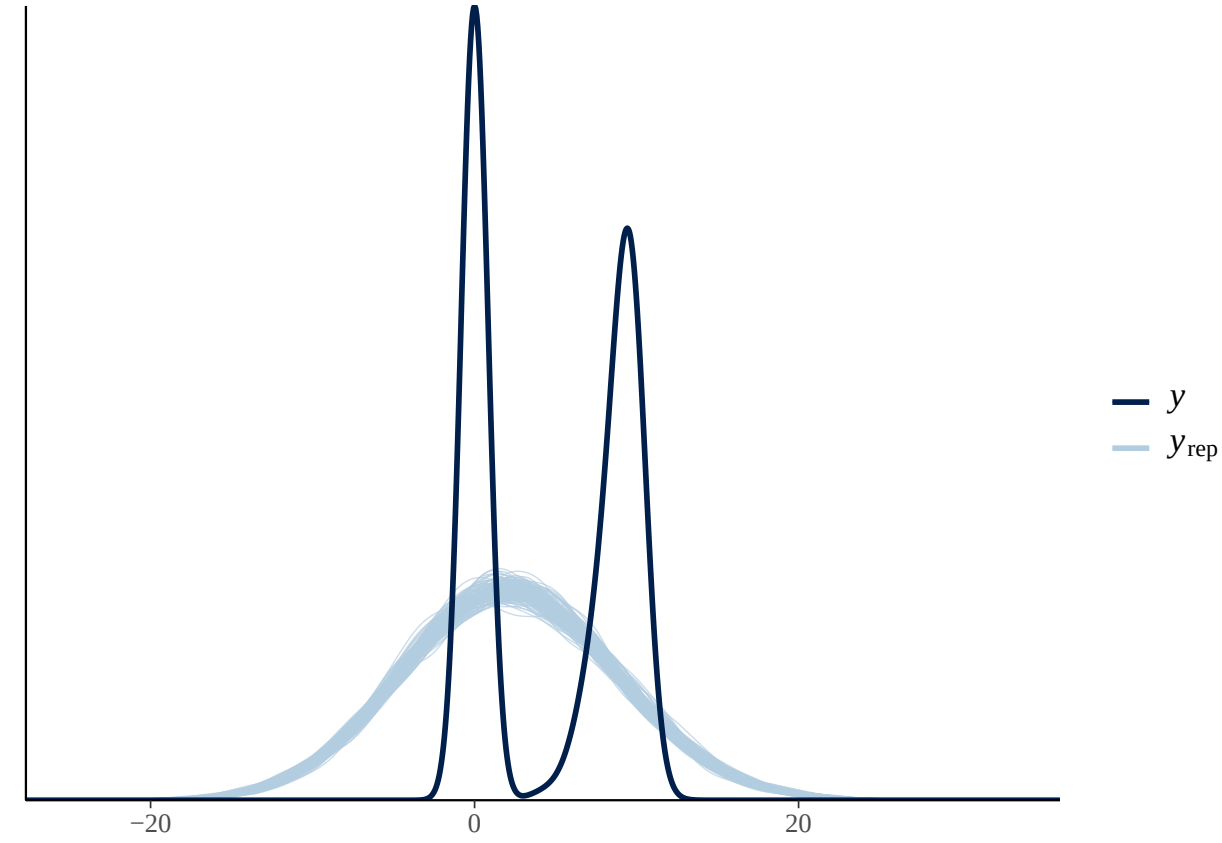
```
##
```

```
## Computed from 4000 by 3000 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo -10952.7 20.1
## p_loo        51.1  1.4
## looic     21905.4 40.2
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
loo_compare(loo_iv_mlm, loo_mlm)
```

```
##        elpd_diff se_diff
## model2     0.0       0.0
## model1 -2058.1      27.2
```

```
# visualize density
pp_check(c(as.numeric(stan_data_iv$y_child),
           as.numeric(stan_data_iv$y_nochild)),
  rstan::extract(post_iv_mlm, par = "yrep")$yrep[sample(1:nrow(df_samp),
                                                        size = 150), ],
  ppc_dens_overlay
)
```



```r
# comparison of predictions
yrep <- rstan::extract(post_mlm, "yrep")[[1]]
low  <- apply(yrep, MARGIN = 2, FUN = quantile, probs = 1 / 3)
high <- apply(yrep, MARGIN = 2, FUN = quantile, probs = 2 / 3)

y <- stan_data$y

c(too_low = mean(y < low),
  just_right = mean(y > low & y < high),
  too_high = mean(y > high))
```

```
##    too_low just_right   too_high
## 0.43300000 0.05866667 0.50833333
```