Final Exam Questions 1 and 2

Michal Malyska

11/04/2020

```
knitr::opts_chunk$set(
    echo = TRUE,
    message = FALSE,
    warning = FALSE,
    cache = TRUE
)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------- tidyverse 1.3
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.4
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
## -- Conflicts ------------------------------------------------------------------- tidyverse_conflicts
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
## The following object is masked from 'package:base':
##
##     date
```

```
library(here)
```

```
## here() starts at /Users/michalmalyska/Desktop/University/Grad School/Classes/STA2201 - Applied Statis
##
## Attaching package: 'here'
## The following object is masked from 'package:lubridate':
##
##     here
```

```
library(corrr)
library(rstan)
```

```
## Loading required package: StanHeaders
## rstan (Version 2.19.2, GitRev: 2e1f913d3ca3)
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
```

```
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
##
## Attaching package: 'rstan'
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
options(mc.cores = parallel::detectCores())
library(tidybayes)
library(tidyposterior)
library(loo)
```

```
## This is loo version 2.2.0
```

```
## - Online documentation and vignettes at mc-stan.org/loo
```

```
## - As of v2.0.0 loo defaults to 1 core but we recommend using as many as possible. Use the 'cores' ar
```

```
##
## Attaching package: 'loo'
```

```
## The following object is masked from 'package:rstan':
##
##     loo
```

```
library(bayesplot)
```

```
## This is bayesplot version 1.7.1
```

```
## - Online documentation and vignettes at mc-stan.org/bayesplot
```

```
## - bayesplot theme set to bayesplot::theme_default()
```

```
##    * Does _not_ affect other ggplot2 plots
```

```
##    * See ?bayesplot_theme_set for details on theme setting
```

```
theme_set(theme_bw())
set.seed(2718)
```

Question 1

a)

Since we are modelling a binary outcome variable (whether the purchase was made online or not), I decided to use a bayesian logistic regression model with the following specification:

$$
\begin{aligned}
y_i &= \mathbb{I}_{online} \\
y_i|\pi_i &\sim Bernoulli(\pi_i) \\
\pi_i &= \text{logit}^{-1}(\beta_0 + \beta^{date} date_i + \beta^{amount} amount_i + \alpha^{age}_{j[i]} + \alpha^{income}_{k[i]} + \\
&\quad + \alpha^{category}_{l[i]} + \alpha^{location}_{m[i]} + \beta^{sd\times age}_j date_i age_{j[i]} + \beta^{sd\times income}_k date_i income_{k[i]}) \\
\alpha^{age}_j &\sim N(0, \sigma^2_{age}) \\
\alpha^{income}_k &\sim N(0, \sigma^2_{income}) \\
\alpha^{category}_l &\sim N(0, \sigma^2_{category}) \\
\alpha^{location}_m &\sim N(0, \sigma^2_{location})
\end{aligned}
$$

Where:

$$
\begin{aligned}
date_i &= \mathbb{I}_{date>March16th} \\
j &\in \{1, 2, 3, 4\}(\text{age groups}) \\
k &\in (1, 2, 3, 4)(\text{income groups}) \\
l &\in \text{Categories} \\
m &\in \text{Locations}
\end{aligned}
$$

Where βs indexed by J and K are just a shorthand notation to write out there is a separate β for each of the interactions. This makes for a total of 11 β coefficients and 4 α coefficient corresponding to different group intercepts.

The choice of priors would then be that of weakly informative on everything.

$$
\begin{aligned}
\beta &\sim N(0, 1), \forall \beta \\
\sigma^2_{age} &\sim N_+(0, 1) \\
\sigma^2_{income} &\sim N_+(0, 1) \\
\sigma^2_{category} &\sim N_+(0, 1) \\
\sigma^2_{location} &\sim N_+(0, 1)
\end{aligned}
$$

An argument can be made for setting means of priors for both α variables to be empirical means for their respective groups and for many other potential modelling changes. But I think this is a good start that could be built upon. Depending on the number of categories and locations I would assess whether those variables should be considered / using different modelling strategies based on neighbourhoods.

The choice of priors could also probably be improved to be more / less informative, but I have no real knowledge so I don't want to use strongly informative priors.

How to assess whether the average change in proportion of online purchases is higher for 18-29 year olds in the second income bracket versus 45-64 year olds in the same income bracket:

A simplified approach is to not include the interactions at all and then just look at the posterior distribution of the outcomes for the two groups pre and post social distancing and look at the difference between those estimates. However I feel like that is not perfectly proper.

The reason I included the interaction terms in the model was to answer this question. It is enough to look at the magnitude of the posterior point estimates for the interaction betas for the respecitve ages and date. The larger the β the larger the change in proportion of online purchases for the respective age group.

This is all under the assumption (which would need to be checked) that people's spending Amounts and other variables were not affected by the social distancing.

For a full solution I would once again take the posterior distributions of outcomes take the differences for each group and then compare how big the differences are.

b)

Based on the posterior samples, how to estimate expected change in proportion of online purchases for people aged 18-29 in the first income bracket and construct a 95% credible interval?

First of all I would take the data for all the 18-29 people in the first income bracket, create a copy of it with just the social distancing variable flipped. Then make predictions using samples from the posterior distributions for coefficients. Transform the predictions to be probabilities. Finally, take the difference in probabilities for the rows differing only in social distancing. The output is a large number of samples from the differences, which we then use to calculate the credible interval. The vague pseudocode would look something like:

```
# make stan data
stan_data <- list(things)

# run model
mod1 <- stan(data = stan_data,
             file = stan_model_dir,
             iter = 100000000,
             seed = 2718)

# do some model checks

# Extract posteriors
sim <- extract(mod1)

# Get the data into the required format

data_model <- data_orig %>%
    filter(age == "18-29", income_bracket == 1) %>%
    mutate(sd_pos = 0,
           sd_neg = 1)

#
model_post <- tibble(
    beta0 = sim$beta[,1],
    beta1 = sim$beta[,2],
    beta2 = sim$beta[,3],
    beta3 = sim$beta[,4],
    ...,
    beta11 = sim$beta[,11],
    alpha_age = sim$alpha_age[,age_bracket],
    ...,
)

# iterate over rows in the data model
for (row_values in data_model_rows) {
    # make predictions
    model_post <- model_post %>%
        mutate(pred_1 = beta0 + # prediction for social distancing
                beta1 * as.numeric(row_values[1]) +
                beta2 * as.numeric(row_values[2]) +
```

```
                beta3 * as.numeric(row_values[sd_pos_idx]) + #interaction (SD == 1)
                ... +
                alpha_age +
                ...
                , # prediction for no social distancing
               pred_2 = beta0 +
                beta1 * as.numeric(row_values[1]) +
                beta2 * as.numeric(row_values[2]) + # no interaction (SD == 0)
                ... +
                alpha_age +
                ...
               ,
               prop_1 = inverse_logit(pred_1),
               prop_2 = inverse_logit(pred_2),
               diff = prop_1 - prop_2)
}

model_post %>%
    ggplot(aes(x = diff)) +
    geom_violin(draw_quantiles = c(0.025, 0.5, 0.975))+
    ggtitle("Posterior for difference in proportions")

CI <- model_post %>%
    summarise(q2.5 = quantile(0.025, diff),
              q97.5 = quantile(0.975, diff)) %>%
    mutate(CI = str_c("(", q2.5, ", ", q97.5 , ")")) %>%
    pull(CI)
```

This would asssume that there is no change in other variables.

Another approach would be to just use the posterior samples to obtain predictions for the filtered data filtered on age and income in the pre-social distancing period, get a proportion. Then obtain a different proportion for just the post social distancing period rows filtered on age and income. Calculate the difference. Repeat the process for every sample from posterior of parameters.

Easier approach (pretty good):

For the second part of getting "prediction intervals" I would sample 30 rows from the dataset filtered on the age and income bracket. Then for each of the 30 rows I would get a prediction using approach similar as above.

Repeat that for every sample from the posterior parameters. This way I get a reasonably good idea of what the predictions on 30 parameters would look like using the posterior distribution.

Calculate the quantiles to get the prediction interval

If no data was observed for that particular group I would first need to sample a new alpha value from it's predictive posterior distribution and the proceed as before.

Code would look almost identical except for the predictions:

```
mutate(pred_1 = rnorm(1, mean = beta0 + ..., sd = sigma_y),
       pred_2 = rnorm(1, mean = beta0 + ..., sd = sigma_y))
```

Which now would be themselves draws from a normal (predictive posterior) distribution.

Harder Approach (probably better):

Instead of just sampling 30 rows from the filtered dataset, I would try to find some joint model of explanatory variables for that filtering, perhaps as a copula for easy sampling. Then sample 30 rows at a time from that and proceed as before to get the prediction interval.

c)

Since we can assume that the differences in propensity for online purchases can be explained fully by age groups we know that the actual predictor can be factored in the same way as our estimator. That is:

$$\pi^* = \frac{\sum_G \pi_g N_g}{N}$$

Thus it is enough to show that $\hat{\pi}_g^{ps}$ is an unbiased estimator for an arbitrary G, and since it is the MLE we know it is unbiased. (It will be the mean rate of purchases within each group to estimate π_g) So that $\hat{\pi}_g$ is y_g/N_g which is of course the MLE estimator for binomial probability, which, again, we know is unbiased.

d)

I will assume that μ_α and σ_α are fixed since we are not given priors for those. (i.e. they are hyperparameters), Also in my notation π_g is the estimate (missing the hat)

Then we have :

$$\mathbb{E}(\pi_g^{mr}|y) = \mathbb{E}(\pi_g^{mr}|y_g)$$

Note that I am restricting the posterior to be only dependent on the y when it belongs to the age group g since we assume age groups are independent.

$$p(\pi_g|y_g) \propto p(y_g|\pi_g)p(\pi_g)$$

Since N_g is a constant. Now observe that both of the distributions are normal.

$$y_g|\pi_g \sim N(\pi_g N_g, \pi_g(1-\pi_g)N_g) = N(\mu_g, \sigma_g^2)$$

by the normal approximation to a binomial, and from the question we know that:

$$\pi_g \sim N(\mu_\alpha, \sigma_\alpha^2)$$

So we have a product of a Normal likelihood with a normal prior, for which we know the posterior distribution:

$$\pi_g|y \sim N(\mu_{post}, \sigma_{post}^2)$$

with

$$\sigma_{post,g}^2 = \left(\frac{1}{\sigma_{prior}^2} + \frac{1}{\sigma_{likelihood}/n}\right)^{-1}$$
$$\mu_{post,g} = \sigma_{post}^2 \left(\frac{\mu_{prior}}{\sigma_{prior}^2} + \frac{\bar{y}}{\sigma_{likelihood}^2/n}\right)$$

substituting:

$$\begin{aligned}\mu_{prior} &= \mu_\alpha \\ \sigma_{prior} &= \sigma_\alpha \\ n &= N_g \\ \bar{y} &= \frac{y_g}{N_g} \\ \sigma_{likelihood} &= \sigma_g\end{aligned}$$

(since y_g is already a count)

We get:

$$\mu_{post,g} = \frac{\mu_\alpha \sigma_g^2 + y_g \sigma_\alpha^2}{\sigma_g^2 + N_g \sigma_\alpha^2}$$

therefore:

$$\mathbb{E}(\pi_g^{mr}|y_g)\mu_{post,g} = \frac{\mu_\alpha \sigma_g^2 + y_g \sigma_\alpha^2}{\sigma_g^2 + N_g \sigma_\alpha^2}$$

Note that this does make the simplifying assumption that we take π_g to be equal to α_g. If we want to be absolutely proper the solution is not tractable if we model $\pi_g = \text{logit}^{-1}(\alpha_g)$ since then π_g follows logit-normal distribution which does not have a tractable mean. Nevertheless the model would then look like:

$$p(\pi_g|y_g) \propto p(y_g|\pi_g)p(\pi_g) = p(y_g|\alpha_g)p(\alpha_g)$$

with :

$$p(y_g|\alpha_g) = N(\text{logit}^{-1}(\alpha_g)N_g, \text{logit}^{-1}(\alpha_g)(1 - \text{logit}^{-1}(\alpha_g)N_g)$$

which is the same as saying:

$$p(\pi_g|y_g) \propto p(y_g|\pi_g)p(\pi_g)$$

$$\pi_g \sim \text{logit-Normal}(\mu_\alpha, \sigma_\alpha)$$

but in this setting there is no expression for the expected value.

e)

Bias is just the expectation of a difference so :

$$\mathbb{E}(\hat{\pi}^{mrp}|\pi^*) - \pi^*$$

Now if we can still make the assumption we made in c that the differences in propensities are fully captured by age we can see that:

$$\pi^* = \sum_G (\pi_g^*)\frac{N_g}{N}, \text{ with: } y_g = \pi_g^* N_g$$

Which leads to:

$$\begin{aligned}
\mathbb{E}(\hat{\pi}^{mrp}|\pi^*) - \pi^* &= \mathbb{E}(\hat{\pi}^{mrp}|y) - \pi^* = \\
&= \mathbb{E}(\frac{\sum_G \pi_g^{mr} N_g}{N}|\pi^*) - \pi^* = \\
&= \sum_G \frac{\mathbb{E}(\pi_g^{mr} N_g|\pi_g^*)}{N} - \pi^* = \\
&= \sum_G \frac{\mathbb{E}(\pi_g^{mr} N_g|\pi_g^*) - \pi_g^* N_g}{N} = \\
&= \sum_G \frac{\mathbb{E}(\pi_g^{mr}|y_g^*) - \pi_g^*}{N/N_g}
\end{aligned}$$

Which is just a weighted average of respective group biases with weights being the group population proportions. Since we know that each of the respective group estimators π_g^{mr} are biased (since they are not equal to π_g^{ps}) but rather moved by a prior, we know this is not 0. (unless we did a perfect job choosing priors and the data is perfect)

Why would we prefer a bayesian estimator over MLE?

It can potentially have a much lower variance leading to a lower mean squared error, as well as being able to incorporate some prior knowledge in case the data might not be reliable (correct for sampling errors etc.)

Without making that assumption the best we can get is:

$$\mathbb{E}(\hat{\pi}^{mrp}|\pi^*) - \pi^* = \mathbb{E}(\mathbb{E}(\hat{\pi}^{mrp}|y)|\pi^*) - \pi^*$$

Where we need to take the expectation over y given π^*. The bias of that would similarily likely be non-zero.

Question 2

```
mmr_data <- read_csv("~/Desktop/University/Grad School/Classes/STA2201 - Applied Statistics/applied-sta

mmr_pred <- read_csv("~/Desktop/University/Grad School/Classes/STA2201 - Applied Statistics/applied-sta
```

a)

```
country_region_list <- mmr_pred %>%
    group_by(iso) %>%
    slice(1) %>%
    arrange(iso) %>%
    select(iso, region)

# the iso country of each country
iso.c <- country_region_list$iso

# number of countries
C <- length(iso.c)

# the region that country c belongs to (name)
region.c <- country_region_list$region
```

```r
# a list of all unique regions
regions <- unique(region.c)

# number of regions
R <- length(regions)

# the region index that country c belongs to
r.c <- as.numeric(factor(region.c, levels = regions))
c.i <- as.numeric(factor(mmr_data$iso, levels = iso.c))

N <- nrow(mmr_data)

y <- log(mmr_data$PM_na)

mmr_data <- mmr_data %>%
    mutate(log_gdp = as.numeric(scale(log(GDP))),
           log_gfr = as.numeric(scale(log(GFR))),
           SAB = as.numeric(scale(SAB)))

X <- model.matrix(~ log_gdp + log_gfr + SAB, mmr_data)

P <- ncol(X)

stan_data <- list( N = N,
                   P = P,
                   C = C,
                   R = R,
                   country = c.i,
                   region = r.c,
                   y = y,
                   X = X)

mod1 <- stan(file = here::here("code/models/exam_Q2_M1.stan"),
             data = stan_data,
             iter = 1000,
             seed = 2718)
```

Please note I did not see the line that indexes the regions by observation so I used indexing by country. Stan code is a tiny bit messier than expected because of this but I think it's incredible that it's able to run

```
mu = X * beta + eta_country[country] + eta_region[region[country]];
```

without writing a for-loop over n.

For all priors I used weakly informative priors with $N(0,1)$ where applicable and $N_+(0,1)$ on variances

b)

First Look at ess (n_eff) and also Rhat to check for convergence just in case.

```r
summary(mod1)$summary[1:10,c("n_eff", "Rhat")]
```

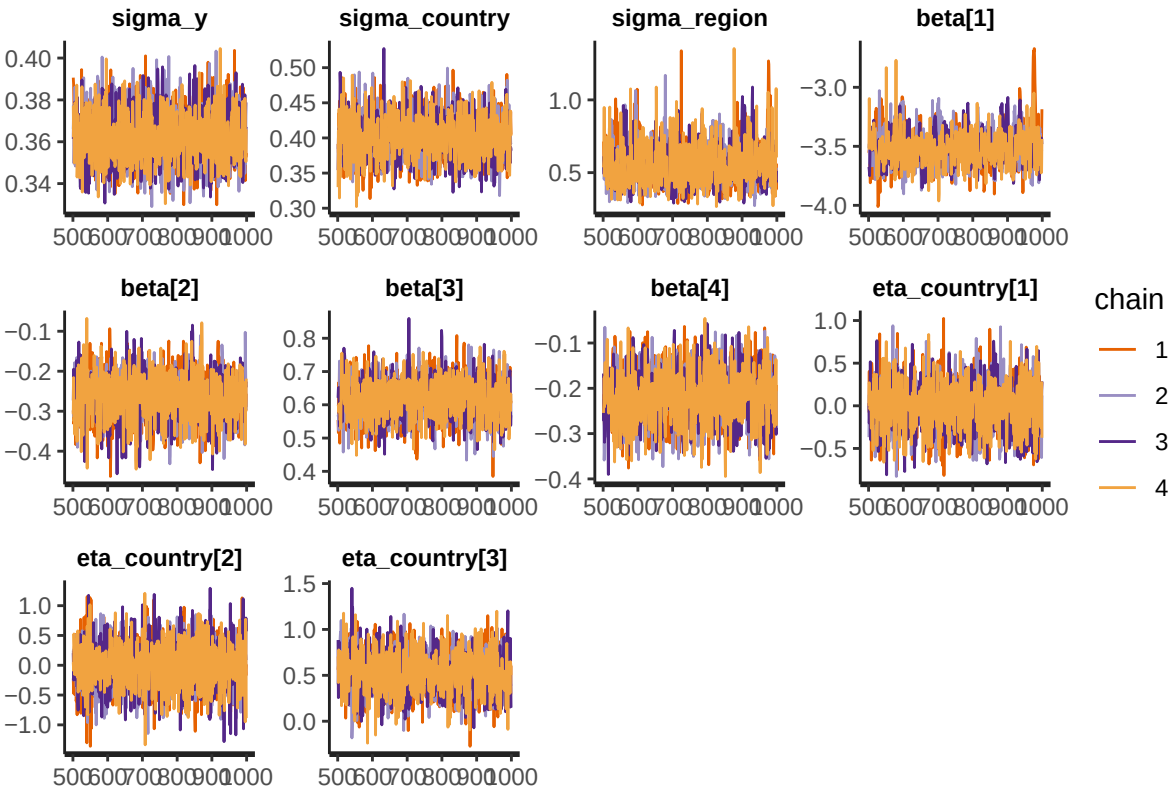
```
##                    n_eff      Rhat
## sigma_y        1722.1652 0.9996344
## sigma_country  1016.4303 1.0028177
## sigma_region    905.0031 1.0008337
```

```
## beta[1]          551.3226 1.0025602
## beta[2]          964.0316 1.0040268
## beta[3]         1322.6723 0.9996121
## beta[4]         1185.1941 1.0010080
## eta_country[1] 2406.4161 1.0013066
## eta_country[2] 3793.8048 0.9996937
## eta_country[3] 1591.9093 1.0006120
```

The effective sample sizes for β_0 and β_1 which are beta[1] and beta[2] in code and for σ_{region} are a bit on the lower side but not too low. I could run the code with more than 1000 iterations to fix this. But since they are still well above 100 it's fine.

Now take a look at traceplots for those:

```
traceplot(mod1)
```



Traceplots show nice mixing, and nothing to be particularily worried about.

Overall I would say the model is acceptable, but could be run for a few more iterations to get better sample numbers.

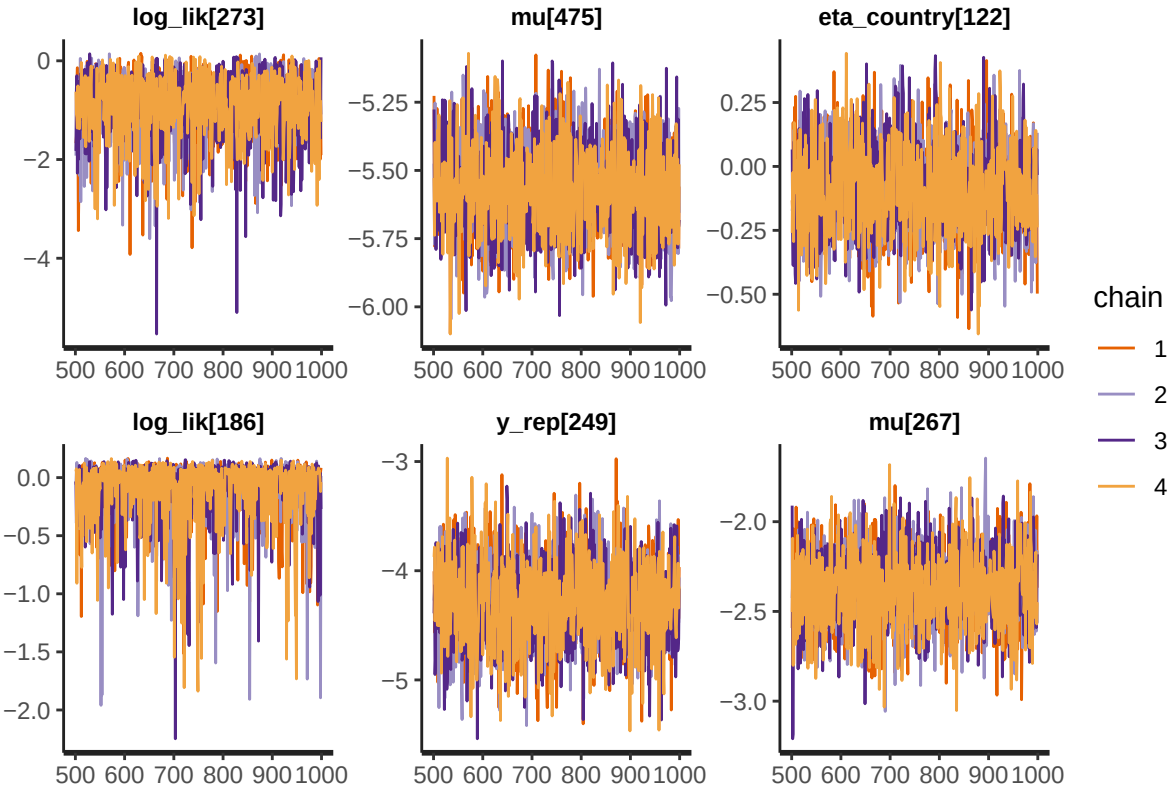
Let's take a quick look at a few (5) random traceplots and summaries just to be sure:

```
set.seed(1)

n_estimates <- length(summary(mod1)$summary[,c("mean")])

ridx <- sample(1:n_estimates, size = 6)

summary(mod1)$summary[ridx,c("n_eff", "Rhat")]
```

```
##                    n_eff      Rhat
```

```
## log_lik[273]      2699.808 0.9996870
## mu[475]           3114.318 0.9995096
## eta_country[122] 1912.047 0.9990704
## log_lik[186]      1310.703 1.0011543
## y_rep[249]        2067.210 1.0009485
## mu[267]           2591.405 1.0000380
```

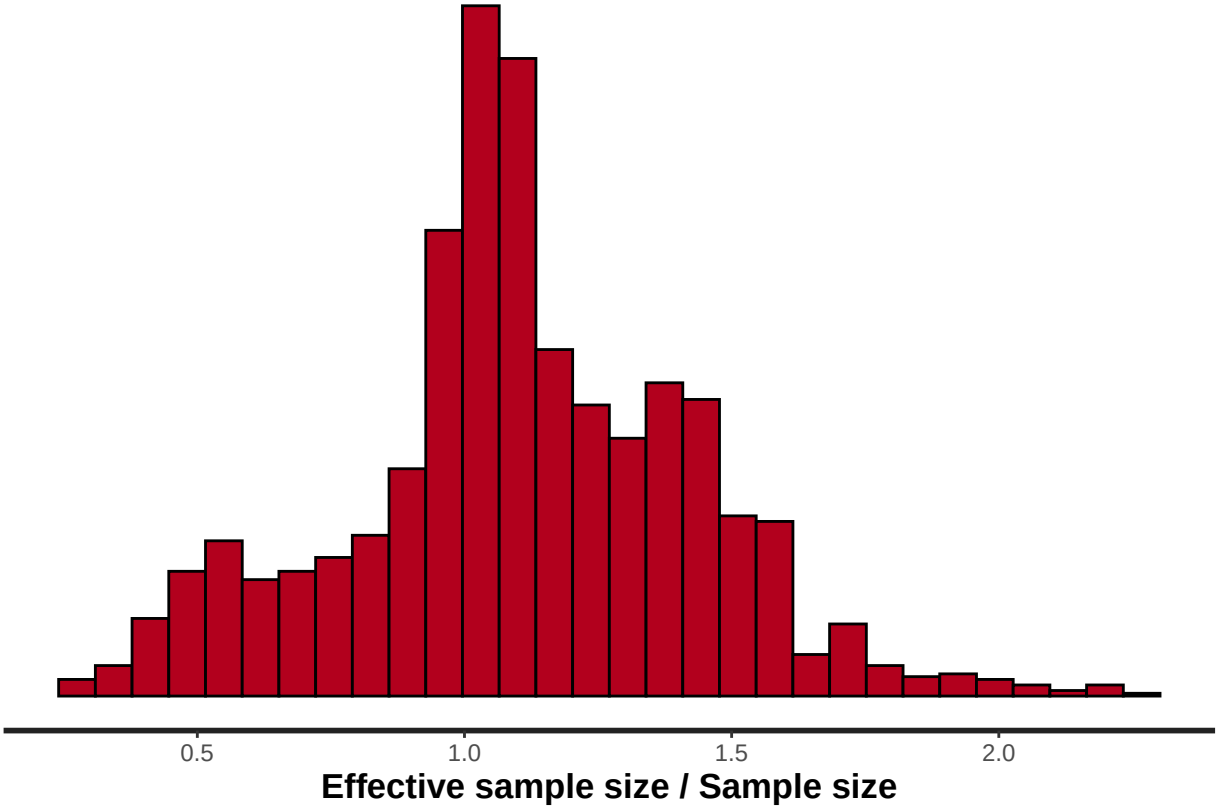
```
ridxnames <- names(summary(mod1)$summary[ridx, 1])

traceplot(mod1, pars = ridxnames)
```



These all look quite good, with reasonable mixing, and very close to convergence, and OK effective sample sizes.

To further show the effective sample sizes, I use the stan ess plotting function to show a histogram of proportion of effective sample size / total samples.

```
stan_ess(mod1)
```



It shows clearly that there are not that many problems (there is not many values close to 0). Worst case sceanrio I would run the chain for 2000 or even 3000 iterations instead of the 1000 to be safe.

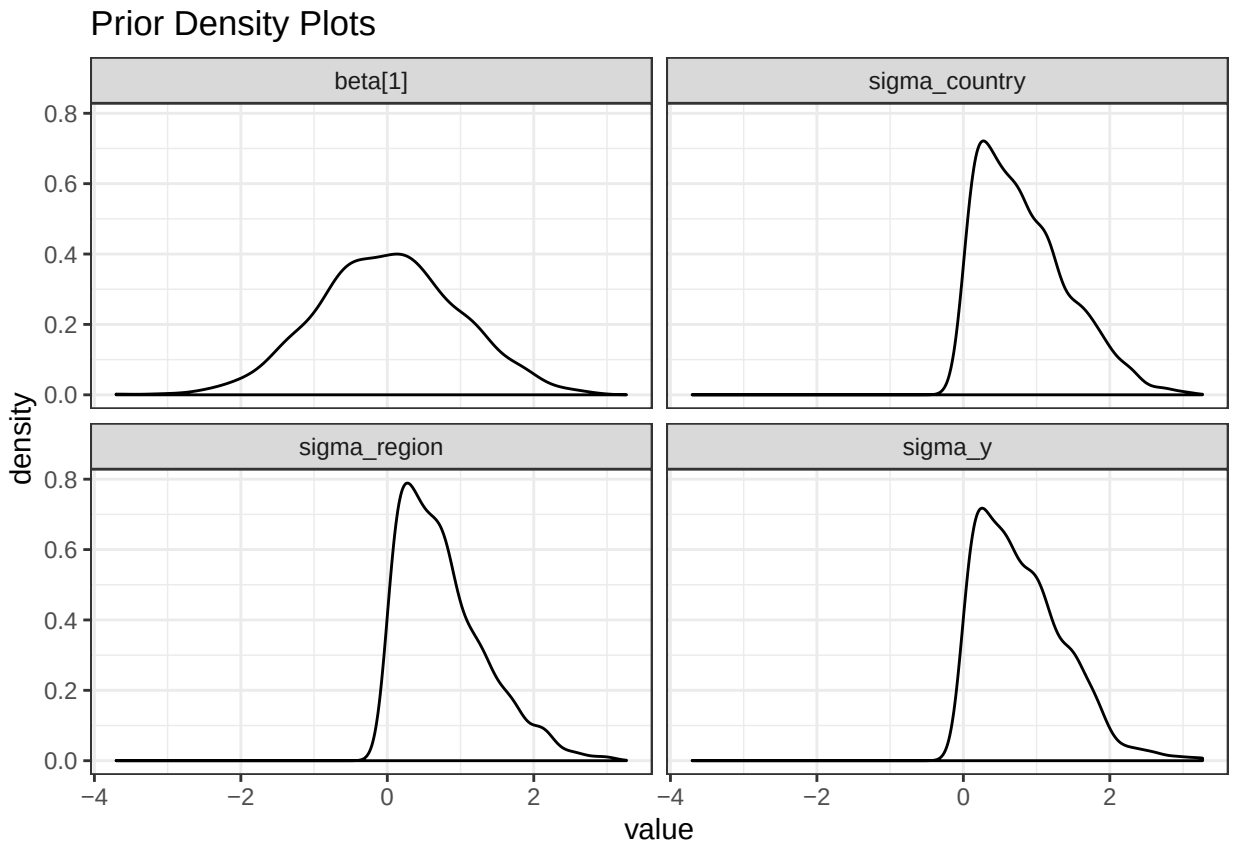
In reality if I had more compute and wanted to do this for real I could easily run this for 1h with more than 4 chains (parallel compute is pretty much free). Like 1M samples only taking 1/100 or even 1/1000 with 20 chains somewhere on an EC2 instance. This should solve all the potential problems here.

c)

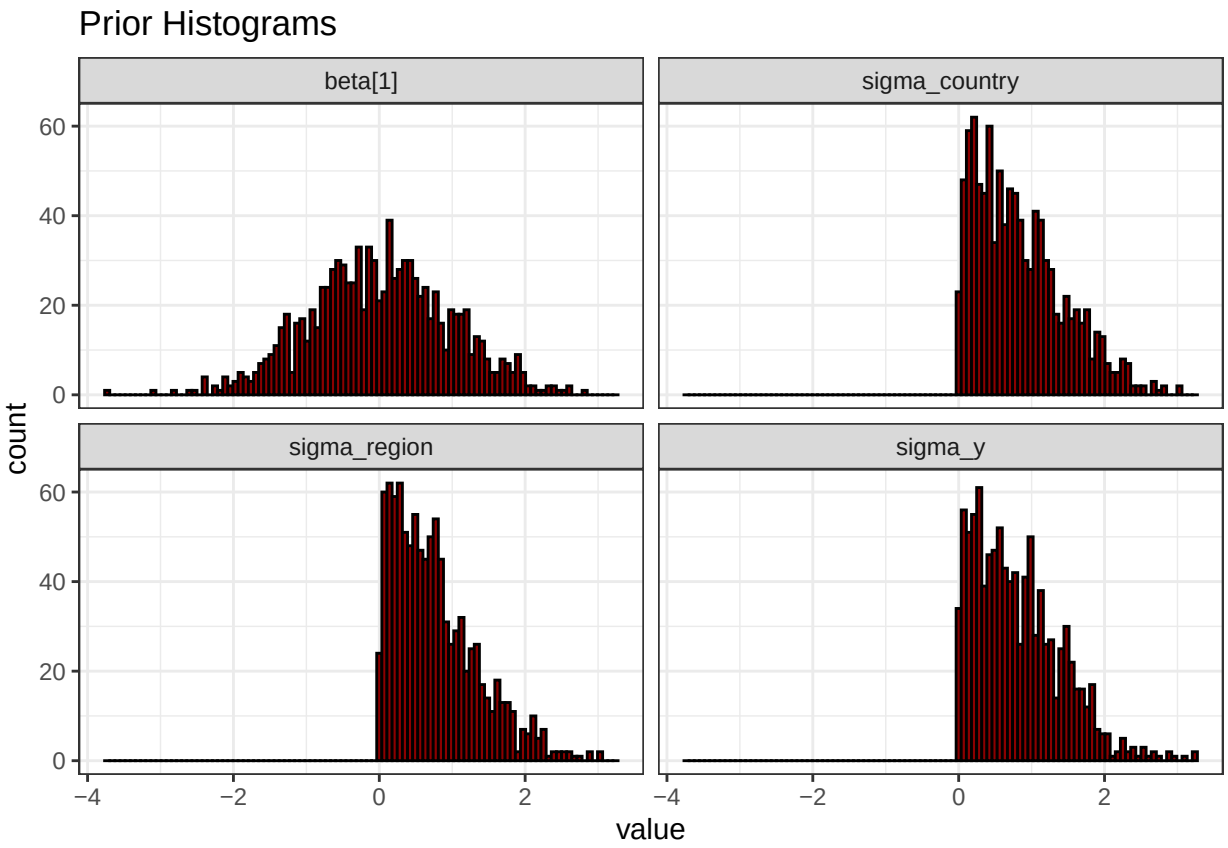
Prior plots

First, the priors for the four parameters of interest:

```
pars <- c("sigma_y", "sigma_country", "sigma_region", "beta[1]")

prior_draws <- tibble(sigma_y = abs(rnorm(1000, 0,1)),
                      sigma_country = abs(rnorm(1000, 0, 1)),
                      sigma_region = abs(rnorm(1000, 0, 1)),
                      beta1 = rnorm(1000, 0 ,1)) %>%
    pivot_longer(everything()) %>%
    mutate(name = if_else(name == "beta1", "beta[1]", name))

prior_draws %>%
    ggplot(aes(x = value, facet = name)) +
    geom_density() +
    facet_wrap(name~.) +
    ggtitle("Prior Density Plots")
```




```
prior_draws %>%
    ggplot(aes(x = value, facet = name)) +
    geom_histogram(bins = 100, fill = "darkred", color = "black") +
    facet_wrap(name~.) +
    ggtitle("Prior Histograms")
```

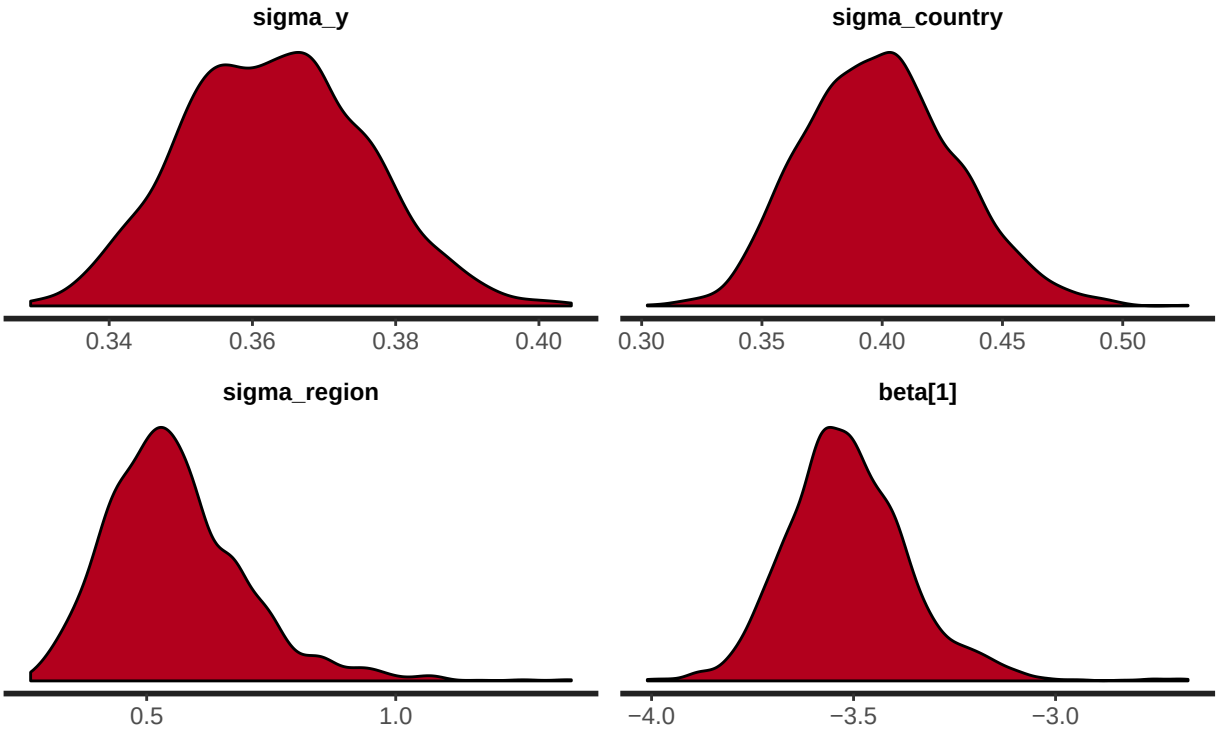


Posterior plots:

We want to see posterior plots for the first four parameters we sampled:

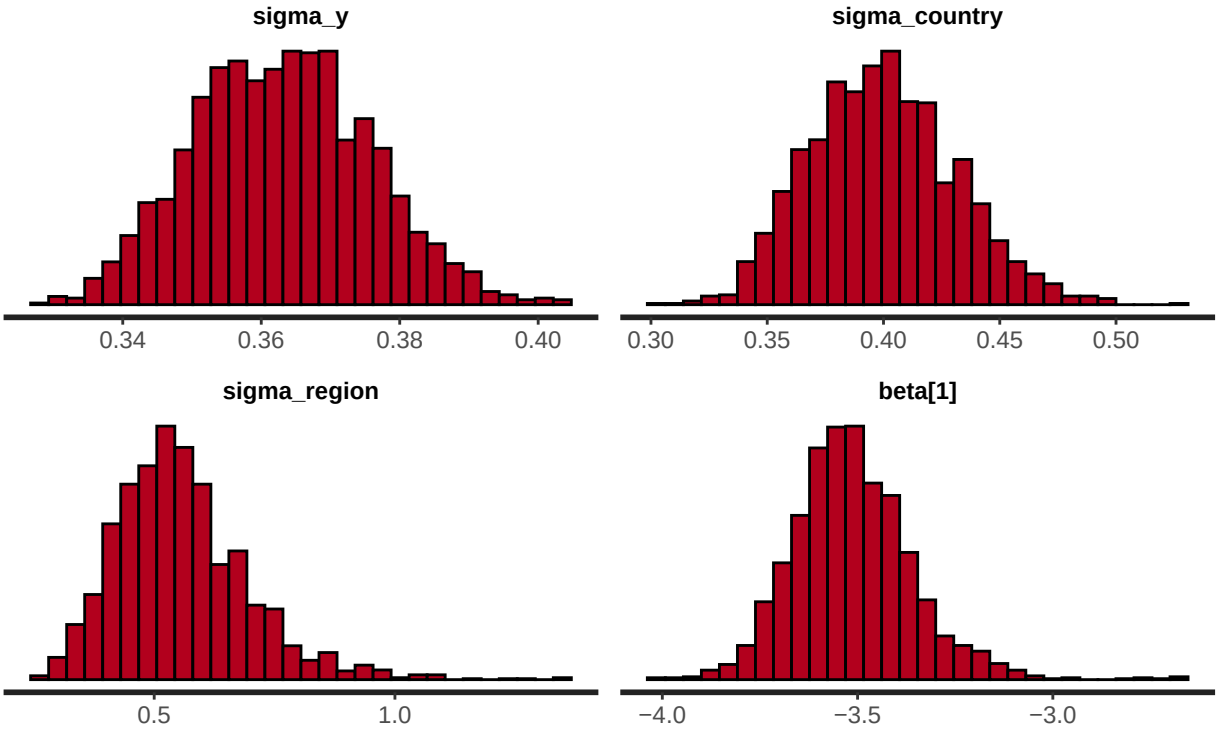
```
stan_dens(mod1, pars = pars) +
    ggtitle("Posterior Density Plots")
```

Posterior Density Plots




```
stan_hist(mod1, pars = pars) +
    ggtitle("Posterior Histograms")
```

Posterior Histograms



Interpret estimates of β_1 and β_3

Since the explanatory variables are standardized the meaning of β_1 is that an increase of log GDP by one standard deviation results in an increase of -0.2651346 in $log(PM^{NA})$ which is a decrease of 0.2651346

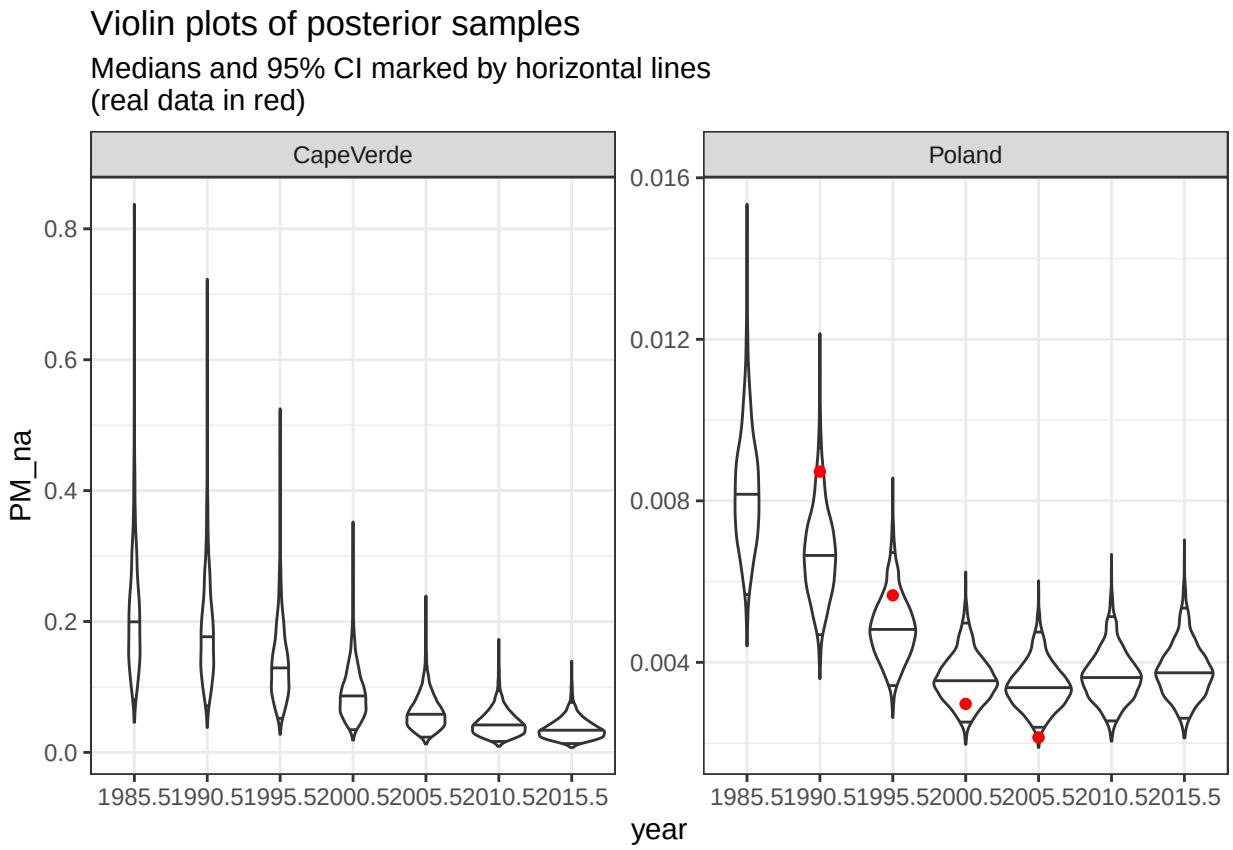
Similarily an increase of 1 SD in SAB results in an increase of -0.2198875 in $log(PM^{NA})$ which is a decrease of 0.2198875

d)

For the country with observation I chose my homeland - Poland, for the country with no observations I chose Cape Verde because I wish I was there right now instead of being stuck at home in quarantine.

```
country_with_obs <- "POL"
country_without_obs <- "CPV"

country_with_obs_iso <- which(iso.c == country_with_obs)
country_without_obs_iso <- which(iso.c == country_without_obs)

sim <- extract(mod1)

pol_idx <- which(c.i == country_with_obs_iso)

# Get data for predictions
pred_data <- mmr_pred %>% filter(iso %in% c(country_without_obs, country_with_obs)) %>%
    mutate(log_gdp = as.numeric(scale(log(GDP))),
           log_gfr = as.numeric(scale(log(GFR))),
           SAB = as.numeric(scale(SAB))) %>%
    select(log_gdp, log_gfr, SAB, iso, mid.date)

# Get the coefficient values
model_post <- tibble(
    beta0 = sim$beta[,1],
    beta1 = sim$beta[,2],
    beta2 = sim$beta[,3],
    beta3 = sim$beta[,4],
    eta_pol = sim$eta_country[,country_with_obs_iso],
    eta_cpv = sim$eta_country[,country_without_obs_iso],
    eta_dr = sim$eta_region[,r.c[country_with_obs_iso]],
    eta_wa = sim$eta_region[,r.c[country_without_obs_iso]],
    sigma_y = sim$sigma_y
)

pred_data <- pred_data %>%
    mutate(name = str_c(iso, mid.date))

years <- unique(pred_data$mid.date)

# Poland
for (year in years) {
    varname = str_c("Poland_", year)
    values <- pred_data %>% filter(iso == "POL", mid.date == year) %>% t() %>% as_tibble() %>% pull()
    model_post <- model_post %>%
        mutate(!!varname := beta0 +
                beta1 * as.numeric(values[1]) +
                beta2 * as.numeric(values[2]) +
```

```
                beta3 * as.numeric(values[3]) +
                eta_pol +
                eta_dr)
}

# CV
for (year in years) {
    varname = str_c("CapeVerde_", year)
    values <- pred_data %>% filter(iso == "CPV", mid.date == year) %>% t() %>% as_tibble() %>% pull()
    model_post <- model_post %>%
        mutate(!!varname := beta0 +
                beta1 * as.numeric(values[1]) +
                beta2 * as.numeric(values[2]) +
                beta3 * as.numeric(values[3]) +
                eta_cpv +
                eta_wa)
}

post_samps <- model_post %>%
    select(-beta0, -beta1, -beta2, -beta3, -sigma_y) %>%
    select_at(vars(!starts_with("eta"))) %>%
    pivot_longer(everything(), names_to = c("country", "year"), names_sep = "_",
                 values_to = "PM_na") %>%
    mutate(PM_na = exp(PM_na))

pol <- mmr_data %>% filter(iso == "POL") %>% select(PM_na, mid.date) %>%
    mutate(country = "Poland",
           real = TRUE) %>%
    rename(year = mid.date)

post_samps <- post_samps %>%
    mutate(real = FALSE) %>%
    rbind(pol)

# post_samps %>%
#   filter(country == "CapeVerde") %>%
#   ggplot(aes(x = year, y = PM_na)) +
#   geom_violin(draw_quantiles = c(0.025, 0.5, 0.975)) +
#   ggtitle("Violin plots of posterior samples for Cape Verde") +
#   labs(subtitle = "Medians and 95% CI marked by horizontal lines")

post_samps %>%
    ggplot(aes(x = year, y = PM_na, facet = country)) +
    geom_violin(draw_quantiles = c(0.025, 0.5, 0.975), data = subset(post_samps, real == FALSE)) +
    ggtitle("Violin plots of posterior samples") +
    labs(subtitle = "Medians and 95% CI marked by horizontal lines \n(real data in red)") +
    facet_wrap(country~., scales = "free") +
    geom_point(data = subset(post_samps, real == TRUE), color = "red")
```




```
# pol_data %>%
#   ggplot(aes(x = year, y = PM_na)) +
#   geom_violin(draw_quantiles = c(0.025, 0.5, 0.975)) +
#   geom_point(data = subset(pol_data, real == TRUE), color = "red") +
#   ggtitle("Violin plots of posterior samples for Poland") +
#   labs(subtitle = "Medians and 95% CI marked by horizontal lines \n real observations in red")
```

```
final_table <- post_samps %>% group_by(country, year) %>%
    filter(real != TRUE) %>%
    summarise(q2.5 = quantile(PM_na, 0.025),
              q97.5 = quantile(PM_na, 0.975),
              point_estimate = median(PM_na)) %>%
    mutate(CI = str_c("(", round(q2.5, digits = 4), ", ", round(q97.5, digits = 4), ")"))

final_table %>%
    select(-q2.5, -q97.5) %>%
    kableExtra::kable(format = "latex")
```

country	year	point_estimate	CI
CapeVerde	1985.5	0.2002490	(0.086, 0.4434)
CapeVerde	1990.5	0.1773900	(0.0765, 0.3935)
CapeVerde	1995.5	0.1292823	(0.057, 0.2833)
CapeVerde	2000.5	0.0864661	(0.0385, 0.1923)
CapeVerde	2005.5	0.0583414	(0.0257, 0.1303)
CapeVerde	2010.5	0.0419681	(0.0185, 0.094)
CapeVerde	2015.5	0.0338037	(0.0146, 0.076)
Poland	1985.5	0.0081675	(0.0058, 0.0115)
Poland	1990.5	0.0066613	(0.0047, 0.0093)
Poland	1995.5	0.0048158	(0.0035, 0.0067)
Poland	2000.5	0.0035421	(0.0025, 0.005)
Poland	2005.5	0.0033811	(0.0024, 0.0047)
Poland	2010.5	0.0036330	(0.0026, 0.0051)
Poland	2015.5	0.0037442	(0.0027, 0.0053)

e)

To compute the MRNA I will just compute the value of the adjustment factor of

$$Adjust = \frac{\#Deaths * (1 - \text{prop Aids})}{Births}$$

for each country year and then multiply by the values to get samples and actual numbers for both countries.

```
adjust_df <- mmr_pred %>%
    filter(iso %in% c("POL", "CPV")) %>%
    mutate(adjust = Deaths * (1 - prop.AIDS) / Births) %>%
    select(Country, mid.date, adjust) %>%
    rename(country = Country,
           year = mid.date) %>%
    mutate(country = str_remove(country, " "),
           year = as.character(year))

final_df_new <- left_join(final_table, adjust_df) %>%
    mutate(q2.5 = q2.5 * adjust,
           q97.5 = q97.5 * adjust,
           point_estimate = point_estimate * adjust,
           CI = str_c("(", round(q2.5, digits = 4), ", ", round(q97.5, digits = 4), ")"))

final_df_new %>%
    filter(year == "2010.5") %>%
    select(-adjust, -q2.5, -q97.5) %>%
    kableExtra::kable(format = "latex")
```

country	year	point_estimate	CI
CapeVerde	2010.5	0.000704	(3e-04, 0.0016)
Poland	2010.5	0.000066	(0, 1e-04)

f)

Model specification

$$
\begin{aligned}
y_i|\eta_{c[i]}^{country}, \eta_{r[i]}^{region} &\sim N(\beta_0 + \eta_{c[i]}^{country} + \eta_{r[i]}^{region} + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \beta_3 x_{i,3},\ \sigma_{VR}^2 \mathbb{I}_{VR}(i) + \sigma_{NVR}^2(1 - \mathbb{I}_{VR}(i))) \\
\eta_{c[i]}^{country} &\sim N(0, (\sigma_\eta^{country})^2),\ \text{for } c = 1, 2 \ldots, C \\
\eta_{r[i]}^{region} &\sim N(0, (\sigma_\eta^{region})^2),\ \text{for } c = 1, 2 \ldots, R
\end{aligned}
$$

Where:

- σ_{VR}^2 is the variance for VR countries
- σ_{NVR}^2 is the variance for non-VR countries

```
mmr_data <- mmr_data %>%
    mutate(VR = if_else(data.type == "VR", 2, 1))

VR <- mmr_data$VR

stan_data <- list( N = N,
                   P = P,
                   C = C,
                   R = R,
                   VR = VR,
                   country = c.i,
                   region = r.c,
                   y = y,
                   X = X)

mod2 <- stan(file = here::here("code/models/exam_Q2_M2.stan"),
             data = stan_data,
             iter = 2000,
             seed = 2718)
```

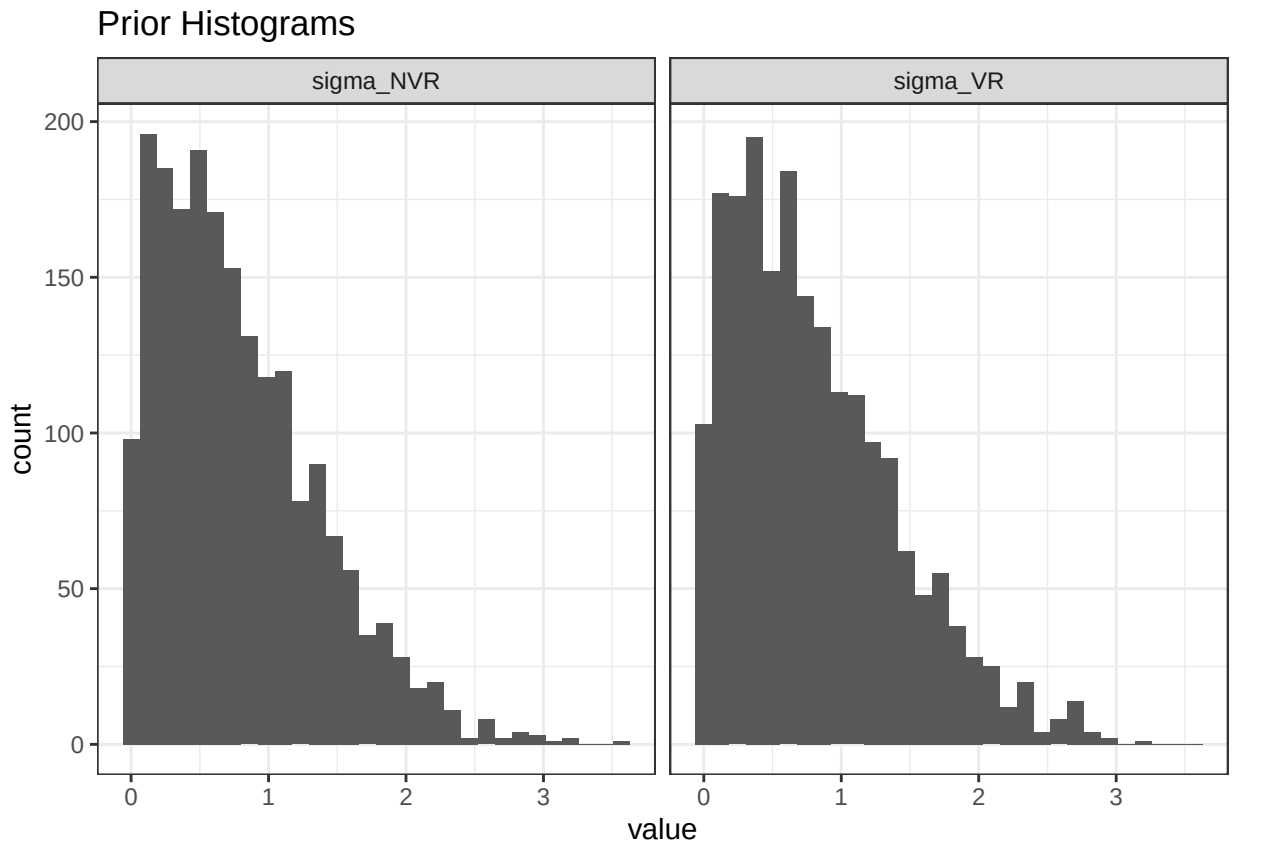
```
summary(mod2)$summary[1:10, c("50%", "n_eff", "Rhat")]
```

```
##                          50%    n_eff      Rhat
## sigma_y[1]      0.329863381 2758.548 1.0003381
## sigma_y[2]      0.385999599 3313.130 0.9997615
## sigma_country   0.401405374 1834.404 1.0000467
## sigma_region    0.536599116 1939.024 1.0011021
## beta[1]        -3.533686083  834.139 1.0096864
## beta[2]        -0.260806497 2137.444 0.9998811
## beta[3]         0.629911435 2027.892 0.9999025
## beta[4]        -0.214163068 1587.045 1.0016703
## eta_country[1] -0.004207526 3885.287 1.0010375
## eta_country[2] -0.005815771 5315.397 0.9994080
```

All Rhats are close to 1, and the n_eff are not bad, significantly higher than 100.

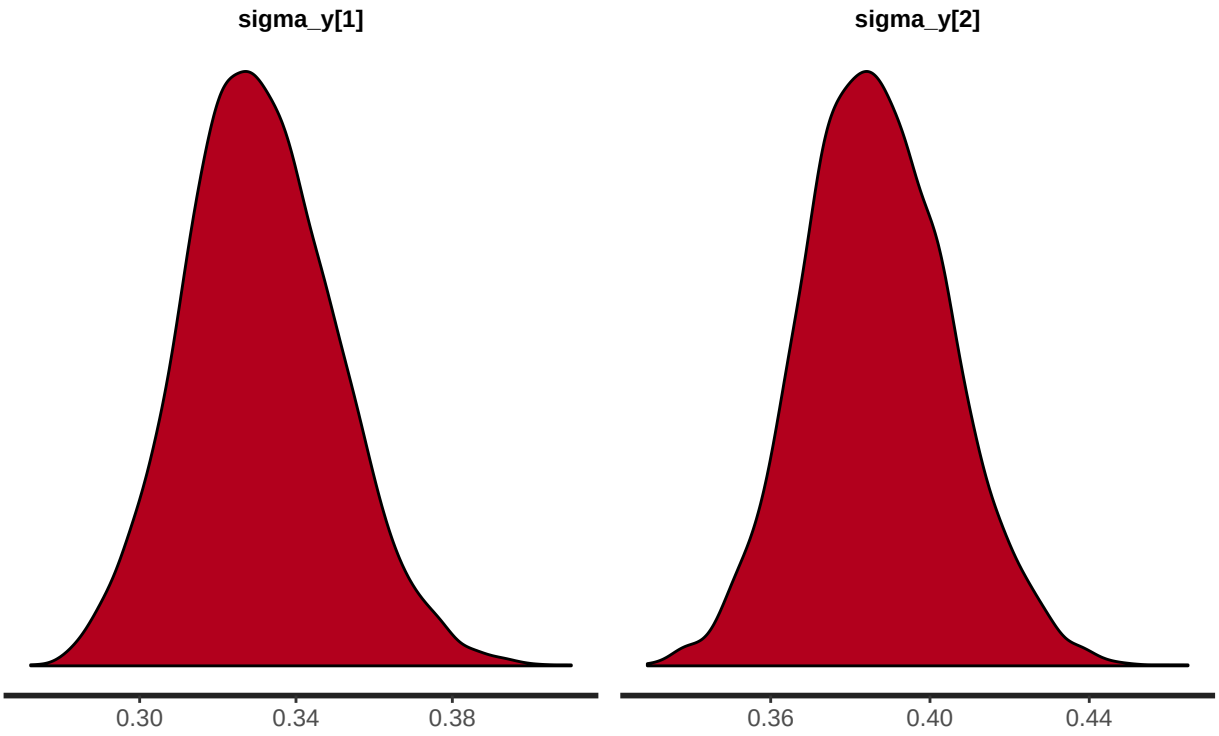
```
pars <- c("sigma_y[1]", "sigma_y[2]")

prior_draws <- tibble(sigma_VR = abs(rnorm(2000, 0,1)),
                      sigma_NVR = abs(rnorm(2000, 0, 1))) %>%
    pivot_longer(everything())

prior_draws %>%
    ggplot(aes(x = value, facet = name)) +
```

```
    geom_histogram() +
    facet_wrap(name~.) +
    ggtitle("Prior Histograms")
```



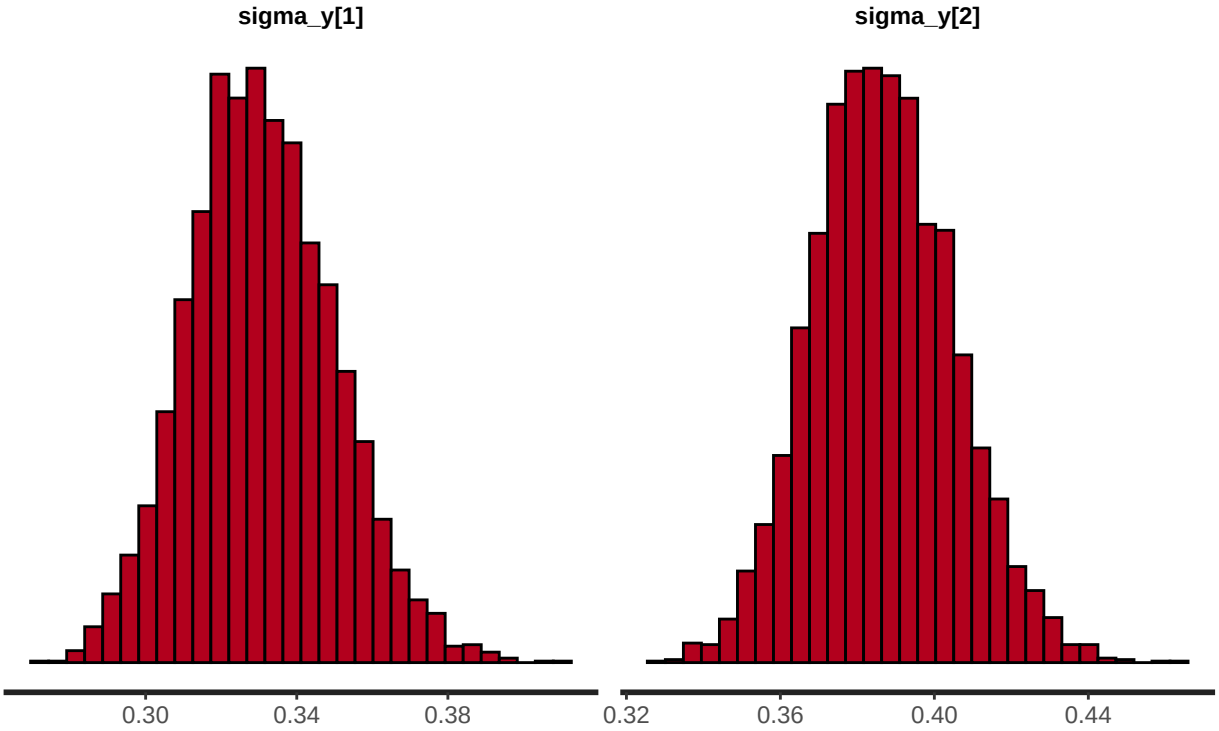

```
stan_dens(mod2, pars = pars) +
    ggtitle("Posterior Density Plots")
```

Posterior Density Plots

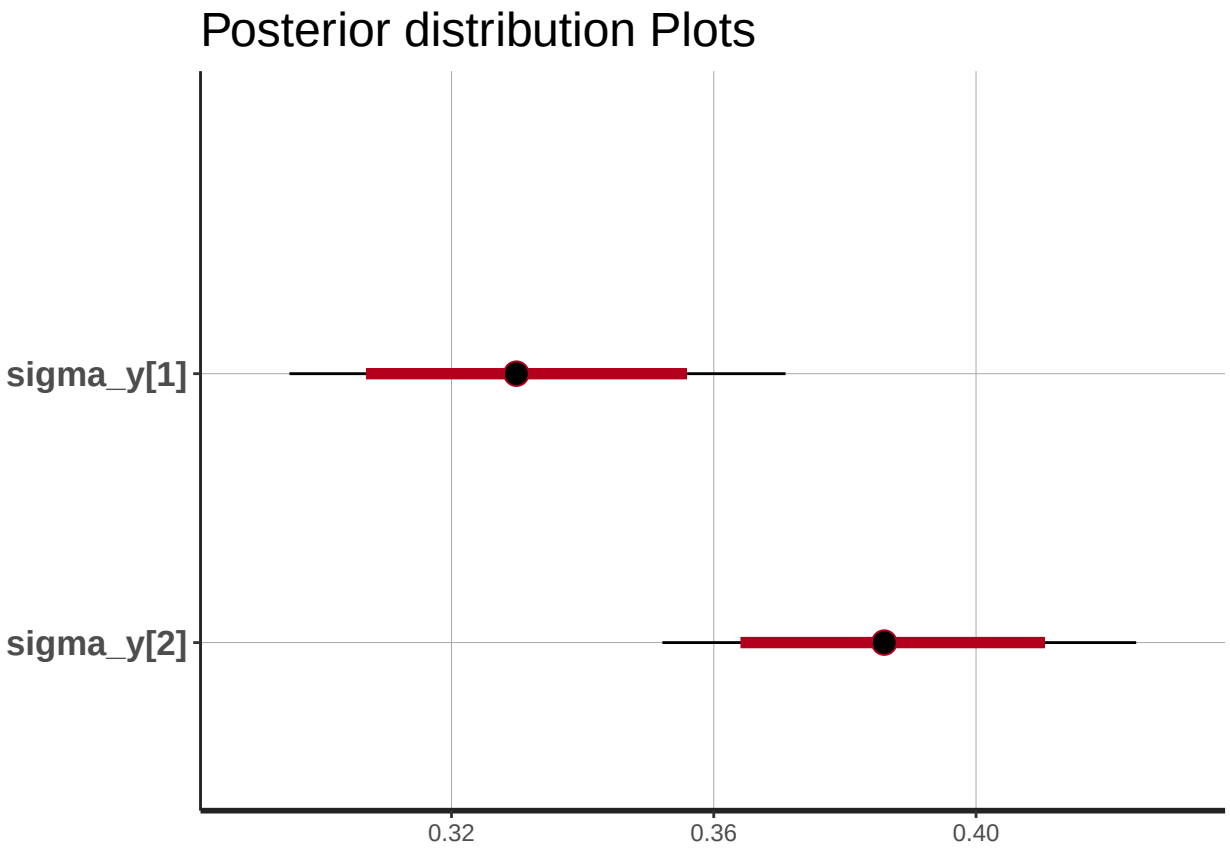



```
stan_hist(mod2, pars = pars) +
    ggtitle("Posterior Histograms")
```

Posterior Histograms



```r
stan_plot(mod2, pars = pars, point_est = "median") +
    ggtitle("Posterior distribution Plots")
```



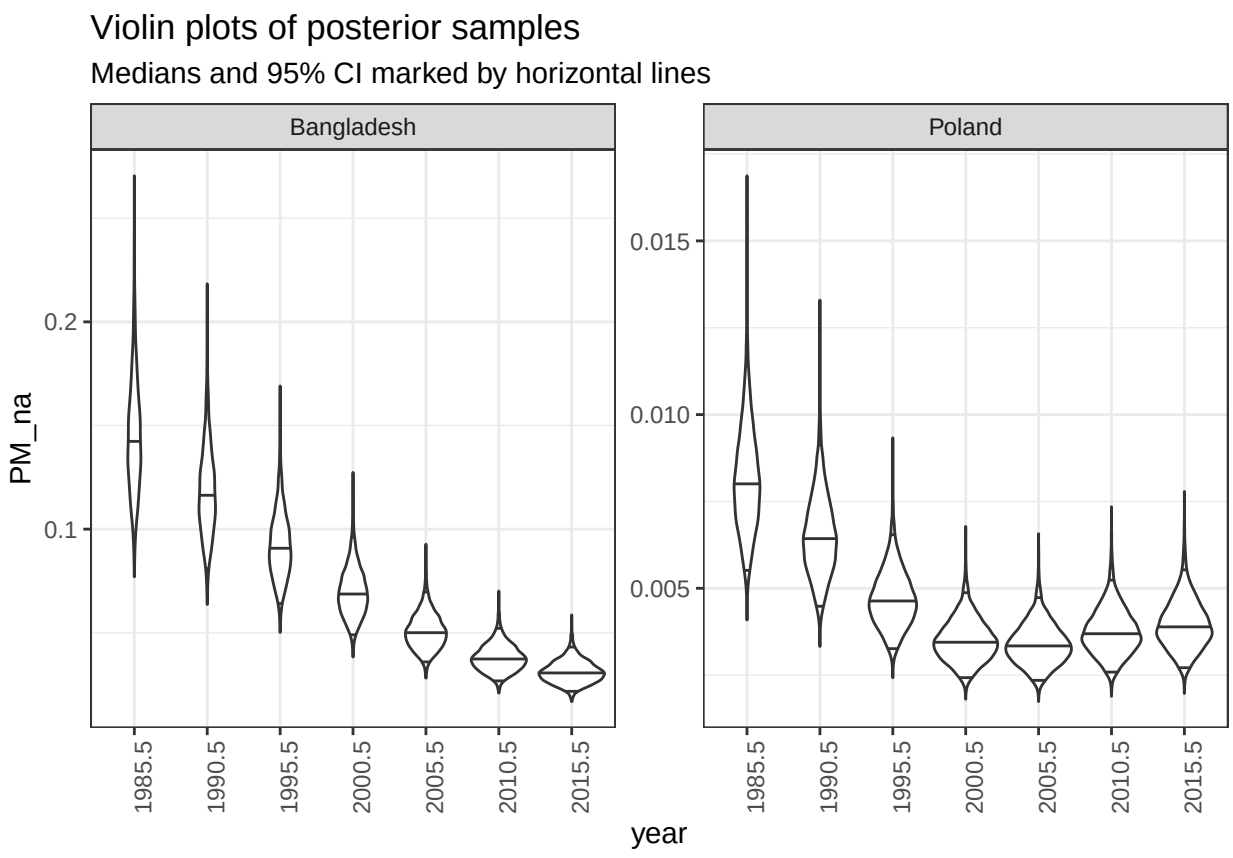
Where sigma_y[1] is non-VR and sigma_y[2] is VR

For VR country I'm gonna use Poland again and for NVR I will use Bangladesh.

```r
country_vr <- "POL"
country_nvr <- "BGD"

country_vr_iso <- which(iso.c == country_vr)
country_nvr_iso <- which(iso.c == country_nvr)

sim <- extract(mod2)

# Get data for predictions
pred_data <- mmr_pred %>% filter(iso %in% c(country_nvr, country_vr)) %>%
    mutate(log_gdp = as.numeric(scale(log(GDP))),
           log_gfr = as.numeric(scale(log(GFR))),
           SAB = as.numeric(scale(SAB))) %>%
    select(log_gdp, log_gfr, SAB, iso, mid.date)

# Get the coefficient values
model_post <- tibble(
    beta0 = sim$beta[,1],
    beta1 = sim$beta[,2],
    beta2 = sim$beta[,3],
    beta3 = sim$beta[,4],
```

```
    eta_pol = sim$eta_country[,country_vr_iso],
    eta_bgd = sim$eta_country[,country_nvr_iso],
    eta_dr = sim$eta_region[,r.c[country_vr_iso]],
    eta_sa = sim$eta_region[,r.c[country_nvr_iso]],
    sigma_vr = sim$sigma_y[,2],
    sigma_nvr = sim$sigma_y[,1]
)

pred_data <- pred_data %>%
    mutate(name = str_c(iso, mid.date))

years <- unique(pred_data$mid.date)

# Poland
for (year in years) {
    varname = str_c("Poland_", year)
    values <- pred_data %>% filter(iso == "POL", mid.date == year) %>% t() %>% as_tibble() %>% pull()
    model_post <- model_post %>%
        mutate(!!varname := beta0 +
                beta1 * as.numeric(values[1]) +
                beta2 * as.numeric(values[2]) +
                beta3 * as.numeric(values[3]) +
                eta_pol +
                eta_dr)
}

# CV
for (year in years) {
    varname = str_c("Bangladesh_", year)
    values <- pred_data %>% filter(iso == "BGD", mid.date == year) %>% t() %>% as_tibble() %>% pull()
    model_post <- model_post %>%
        mutate(!!varname := beta0 +
                beta1 * as.numeric(values[1]) +
                beta2 * as.numeric(values[2]) +
                beta3 * as.numeric(values[3]) +
                eta_bgd +
                eta_sa)
}

post_samps <- model_post %>%
    select(-beta0, -beta1, -beta2, -beta3) %>%
    select_at(vars(!starts_with("eta"))) %>%
    select_at(vars(!starts_with("sigma"))) %>%
    pivot_longer(everything(), names_to = c("country", "year"), names_sep = "_",
                 values_to = "PM_na") %>%
    mutate(PM_na = exp(PM_na))
#
# pol <- mmr_data %>%
#   filter(iso %in% c("POL", "BGD")) %>%
#   select(iso, PM_na, mid.date) %>%
#   mutate(country = if_else(iso == "POL", "Poland", "Bangladesh"),
#          real = TRUE) %>%
#   rename(year = mid.date) %>%
```

```
#   select(-iso)
#
# post_samps <- post_samps %>%
#   mutate(real = FALSE) %>%
#   rbind(pol)

post_samps %>%
    ggplot(aes(x = year, y = PM_na, facet = country)) +
    geom_violin(draw_quantiles = c(0.025, 0.5, 0.975)) +
    ggtitle("Violin plots of posterior samples") +
    labs(subtitle = "Medians and 95% CI marked by horizontal lines") +
    facet_wrap(country~., scales = "free") +
    theme(axis.text.x = element_text(angle = 90, hjust = 1))
```




```
final_table <- post_samps %>% group_by(country, year) %>%
    summarise(q2.5 = quantile(PM_na, 0.025),
              q97.5 = quantile(PM_na, 0.975),
              point_estimate = median(PM_na)) %>%
    mutate(CI = str_c("(", round(q2.5, digits = 4), ", ", round(q97.5, digits = 4), ")"))

final_table %>%
    select(-q2.5, -q97.5) %>%
    kableExtra::kable(format = "latex")
```

country	year	point_estimate	CI
Bangladesh	1985.5	0.1423916	(0.0984, 0.2067)
Bangladesh	1990.5	0.1163220	(0.0817, 0.166)
Bangladesh	1995.5	0.0908142	(0.065, 0.1278)
Bangladesh	2000.5	0.0687417	(0.0496, 0.0957)
Bangladesh	2005.5	0.0500869	(0.0363, 0.0694)
Bangladesh	2010.5	0.0374326	(0.0271, 0.052)
Bangladesh	2015.5	0.0306347	(0.022, 0.043)
Poland	1985.5	0.0080138	(0.0056, 0.0114)
Poland	1990.5	0.0064350	(0.0045, 0.0091)
Poland	1995.5	0.0046451	(0.0033, 0.0065)
Poland	2000.5	0.0034516	(0.0024, 0.0049)
Poland	2005.5	0.0033438	(0.0024, 0.0047)
Poland	2010.5	0.0036938	(0.0026, 0.0052)
Poland	2015.5	0.0038901	(0.0027, 0.0055)