Week 11: Splines

Toufic Ayoub

03/04/23

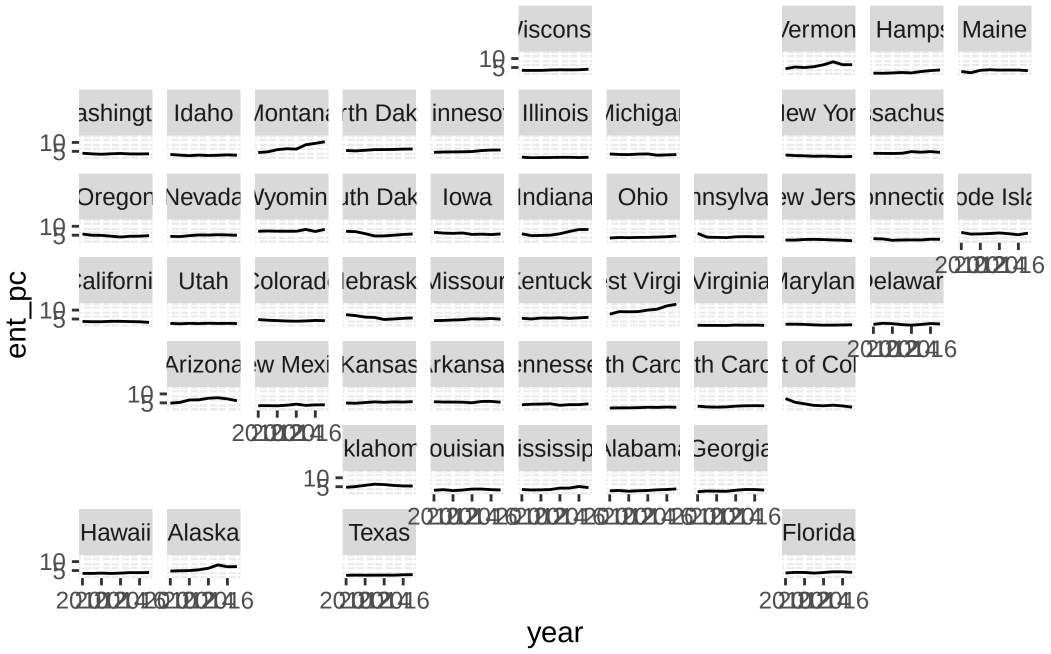
```
library(tidyverse)
library(here)
library(rstan)
library(tidybayes)
source(here("code/getsplines.R"))
library(geofacet)
```

```
d <- read_csv(here("fc_entries.csv"))
```

Question 1

- Make a plot highlighting trends over time by state. Might be a good opportunity to use `geofacet`. Describe what you see in a couple of sentences.

```
d |>
  ggplot(aes(year, ent_pc)) +
  geom_line() +
  facet_geo(~state)
```



We see that for the majority of states, the trends are relatively flat and somewhat similar; indicating a low state variation. However, there are some exceptional cases, such as West Virgina and District of Columbia. These two deviate from the flat trends we see for most of the states, the former trending positively over time while the later trending negatively. It would be interesting to see what factors may cause such a stark difference between states like this.

Question 2

- Fit a hierarchical second-order P-Splines regression model to estimate the (logged) entries per capita over the period 2010-2017. The model you want to fit is

$$
\begin{aligned}
y_{st} &\sim N(\log \lambda_{st}, \sigma^2_{y,s}) \\
\log \lambda_{st} &= \alpha_k B_k(t) \\
\Delta^2 \alpha_k &\sim N(0, \sigma^2_{\alpha,s}) \\
\log \sigma_{\alpha,s} &\sim N(\mu_\sigma, \tau^2)
\end{aligned}
$$

- Where $y_{s,t}$ is the logged entries per capita for state s in year t. Use cubic splines that have knots 2.5 years apart and are a constant shape at the boundaries. Put standard normal priors on standard deviations and hyperparameters.

```r
years = unique(d$year)
N = length(years)

y = log(d |> select(state, year, ent_pc) |> pivot_wider(names_from = "state", values_from

res = getsplines(years, 2.5)
B = res$B.ik
K = ncol(B)
S = length(unique(d$state))

stan_data <- list(N = N,
                  S = S,
                  K = K,
                  y = y,
                  B = B)

mod <- stan(data = stan_data,
             file = here("labs/11.stan"),
             seed = 243)
```

Question 3

- Project forward entries per capita to 2030. Pick 4 states and plot the results (with 95% CIs). Note the code to do this in R is in the lecture slides.

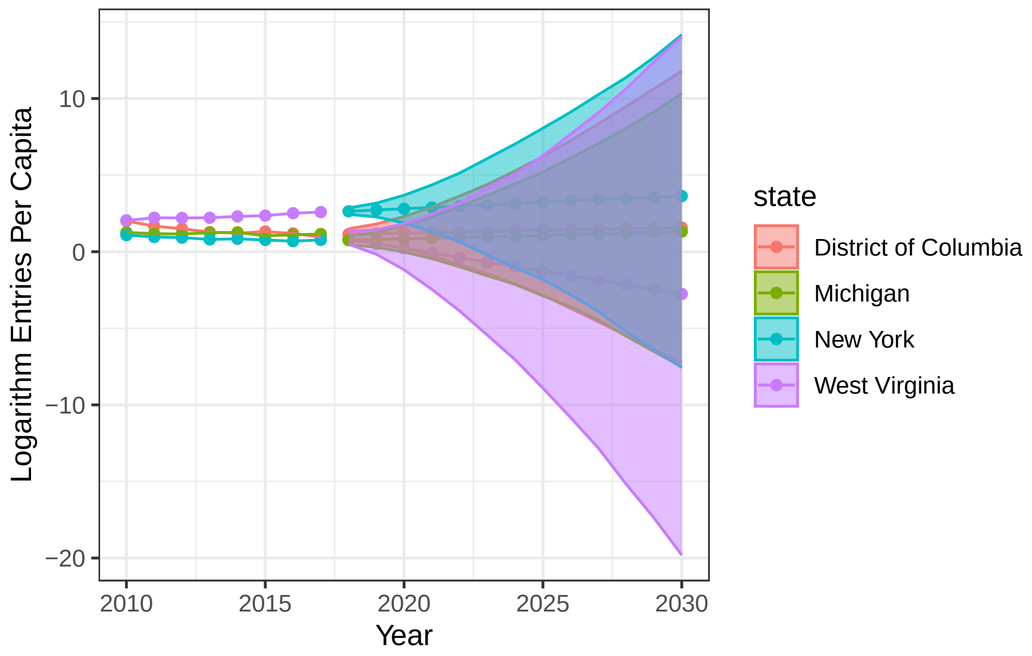
```r
B.ik = B
proj_years = 2018:2030


B.ik_full <- getsplines(c(years, proj_years), 2.5, degree=3)$B.ik
K <- ncol(B.ik)
K_full <- ncol(B.ik_full)
proj_steps <- K_full - K

alphas <- extract(mod)[["alpha"]]
sigmas <- extract(mod)[["sigma_alpha"]]
sigma_ys <- extract(mod)[["sigma_y"]]
nsims <- nrow(alphas)
```

```r
alphas_proj <- array(NA, c(nsims, proj_steps, 51))
set.seed(1098)


for(j in 1:51){
  first_next_alpha <- rnorm(n = nsims,
                            mean = 2*alphas[,K,j] - alphas[,K-1,j],
                            sd = sigmas[,j])
  second_next_alpha <- rnorm(n = nsims,
                             mean = 2*first_next_alpha - alphas[,K,j],
                             sd = sigmas[,j])

  alphas_proj[,1,j] <- first_next_alpha
  alphas_proj[,2,j] <- second_next_alpha

  for(i in 3:proj_steps){
    alphas_proj[,i,j] <- rnorm(n = nsims,
                               mean = 2*alphas_proj[,i-1,j] - alphas_proj[,i-2,j],
                               sd = sigmas[,j])
  }
}

y_proj <- array(NA, c(nsims, 13, 51))

for(i in 1:13){
  for(j in 1:51){
    all_alphas <- cbind(alphas[,,j], alphas_proj[,,j] )
    this_lambda <- all_alphas %*% as.matrix(B.ik_full[length(years)+i, ])
    y_proj[,i,j] <- rnorm(n = nsims, mean = this_lambda, sd = sigma_ys[,j])
  }
}
```

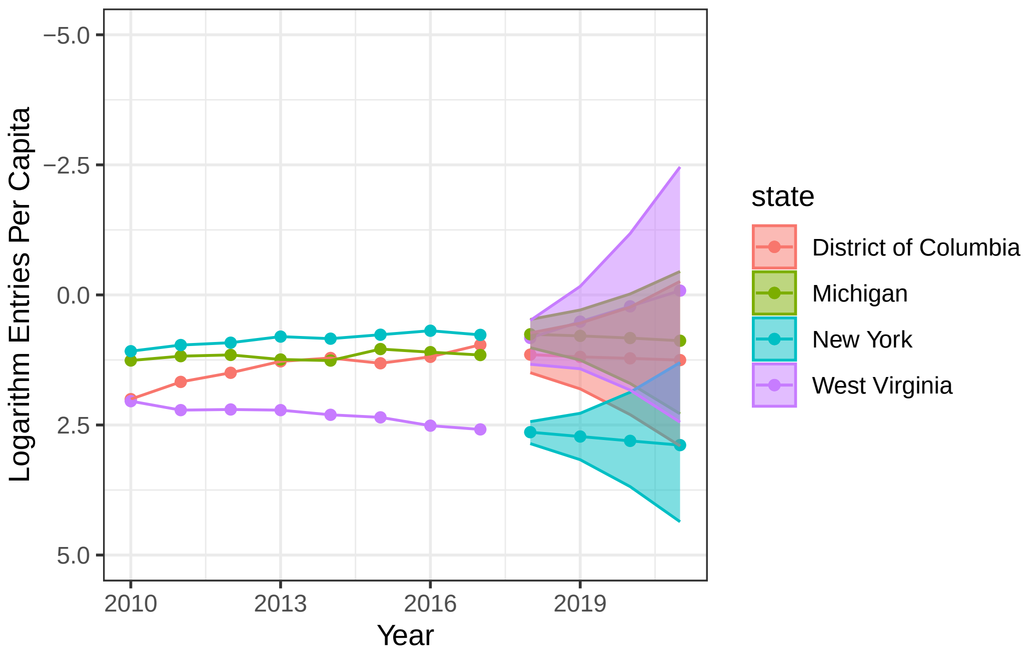
```r
projection_for = function(state, state_id){
  state_proj = y_proj[,,state_id]

proj_d = as.tibble(state_proj) |>
          summarize(across(everything(), list(med = median,
                                              lower = ~quantile(., probs = 0.025),
                                              upper = ~quantile(., probs = 0.975)))) |>
          pivot_longer(cols = everything(),
                       names_to = c(".value", "column"),
                        names_sep = "_")

proj_d =  as.tibble(t(as.matrix(proj_d)))
proj_d <- proj_d[2:nrow(proj_d),]
colnames(proj_d) <- c("med", "lower", "upper")

proj_d = proj_d |> mutate(year = proj_years)
proj_d = proj_d |> mutate(state = rep(state, nrow(proj_d)))
}
```

```
i = which(unique(d$state) %in%
            c("West Virginia", "District of Columbia", "Michigan", "New York"))

# i = 9, 23, 33, 49

WV = projection_for("West Virginia", 9)
DC = projection_for("District of Columbia", 23)
MI = projection_for("Michigan", 33)
NY = projection_for("New York", 49)
```

```
full = bind_rows(WV, DC, MI, NY)
colnames(full) = c("med", "lower", "upper", "year", "state")

d |>
  filter(state %in% c("West Virginia", "District of Columbia", "Michigan", "New York")) |>
  ggplot(aes(x=year, color=state)) +
  geom_point(aes(y=log(ent_pc))) +
  geom_line(aes(y=log(ent_pc))) + theme_bw() +
  geom_line(data = full, aes(x = year, y = as.numeric(med), color = state)) +
  geom_point(data = full, aes(x = year, y = as.numeric(med), color = state)) +
  geom_ribbon(data = full, aes(x = year, ymin = as.numeric(lower),
                              ymax = as.numeric(upper), fill = state), alpha = 0.5) +
  xlab("Year") + ylab("Logarithm Entries Per Capita")
```



```
d |>
  filter(state %in% c("West Virginia", "District of Columbia", "Michigan", "New York")) |>
  ggplot(aes(x=year, color=state)) +
  geom_point(aes(y=log(ent_pc))) +
  geom_line(aes(y=log(ent_pc))) + theme_bw() +
  geom_line(data = full, aes(x = year, y = as.numeric(med), color = state)) +
  geom_point(data = full, aes(x = year, y = as.numeric(med), color = state)) +
  geom_ribbon(data = full, aes(x = year, ymin = as.numeric(lower),
                              ymax = as.numeric(upper), fill = state), alpha = 0.5) +
  xlab("Year") + ylab("Logarithm Entries Per Capita") +  xlim(2010,2021)+ ylim(5,-5)
```


−5.0
−2.5
0.0
2.5
5.0
Logarithm Entries Per Capita
2010
2013
2016
2019
Year
state
District of Columbia
Michigan
New York
West Virginia