Week 11: Splines

J. Arturo Esquivel

30/03/23

Overview

In this lab you'll be fitting a second-order P-Splines regression model to foster care entries by state in the US, projecting out to 2030.

```r
library(tidyverse)
library(here)
library(rstan)
library(tidybayes)
source(here("getsplines.R"))
```

Here's the data

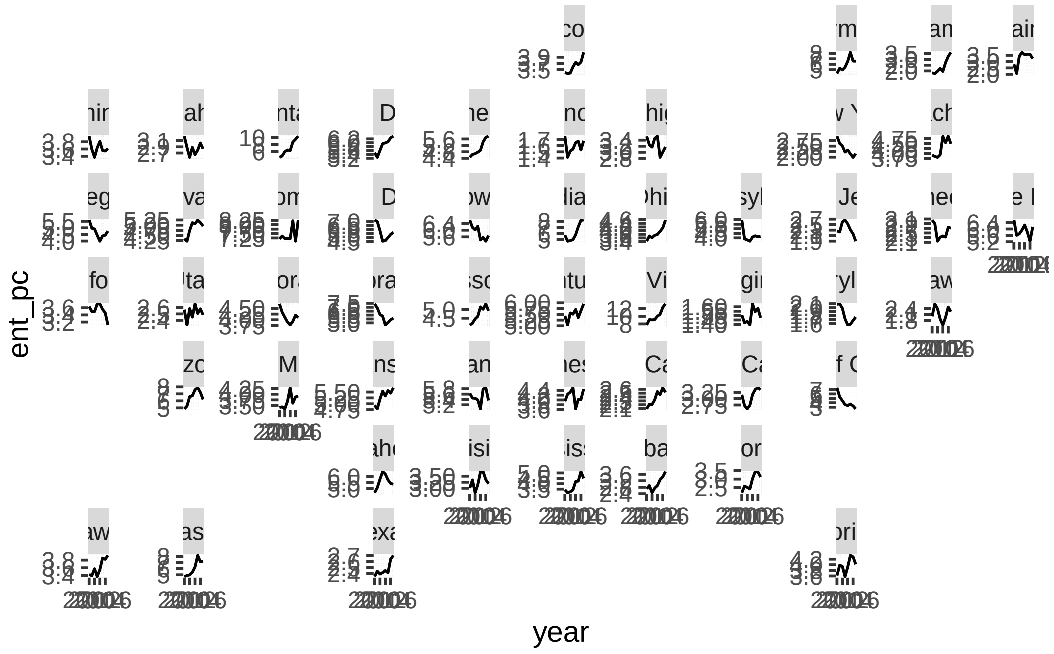
```r
d <- read_csv(here("fc_entries.csv"))
```

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.

```r
library(geofacet)

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



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 = "ent_pc") |>
  select(-year) |>
  as.matrix()/1000)
```

```r
res <- getsplines(years, 2.5)
B <- res$B.ik
K <- ncol(B)

stan_data <- list(N = N, y = y, K = K, S = length(unique(d$state)),
                  B = B)

mod <- stan(data = stan_data, file = "lab11.stan",
            verbose = FALSE,
            refresh = 0)
```

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
proj_years <- 2018:2030
# splines for whole period
B.ik_full <- getsplines(c(years, proj_years), 2.5)$B.ik
#K <- ncol(B.ik) # number of knots in sample
K_full <- ncol(B.ik_full)
proj_steps <- K_full - K# number of projection steps

# get your posterior samples
alphas <- extract(mod)[["alpha"]]
sigmas <- extract(mod)[["sigma_alpha"]] # sigma_alpha
sigma_ys <- extract(mod)[["sigma_y"]]
nsims <- nrow(alphas)
```

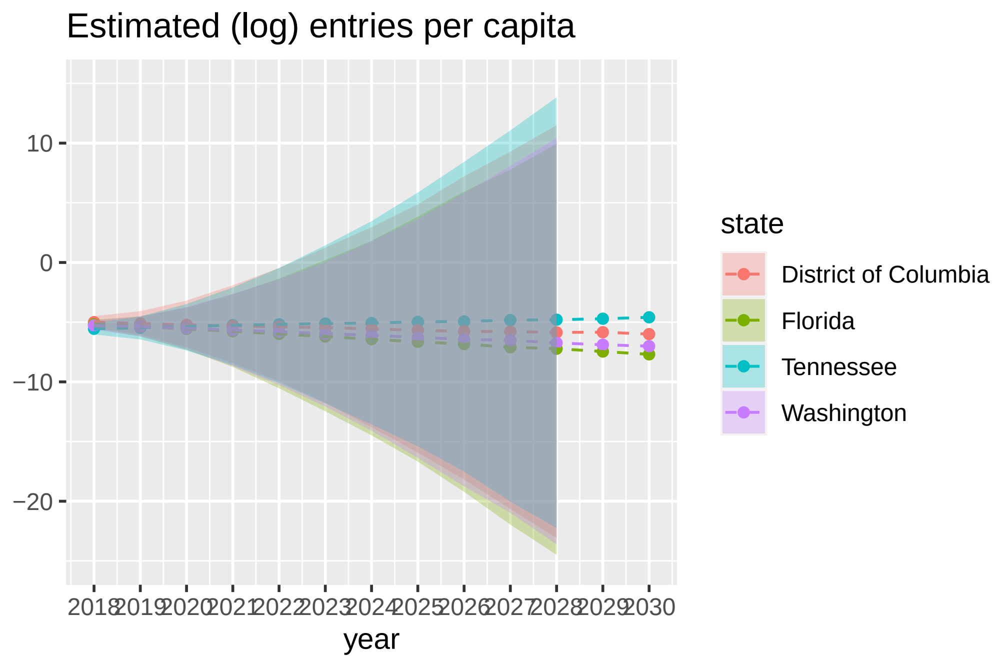
```r
set.seed(10)
states <- sample(length(unique(d$state)), 4)
# first, project the alphas
alphas_proj <- array(NA, c(nsims, proj_steps, 4))
# project the alphas
for(j in 1:length(states)){
  first_next_alpha <- rnorm(n = nsims,
                            mean = 2*alphas[,K,j] - alphas[,K-1,j],
                            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
# now project the rest
for(i in 3:proj_steps){ #!!! not over years but over knots
  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, length(proj_years), 4))
for(i in 1:length(proj_years)){ # now over years
for(j in 1:4){
  all_alphas <- cbind(alphas[,,j], alphas_proj[,,j] )
  this_lambda <- all_alphas %*% as.matrix(B.ik_full[length(years)+i, ])
  y_proj[,i,j] <- exp(rnorm(n = nsims, mean = this_lambda, sd = sigma_ys[,j]))
}
}
```

```r
for(i in 1:4){
  proj <- data.frame(y_proj[,,i])
  names(proj) <- proj_years

  proj <- proj |>
    pivot_longer(names_to = "year", cols = 1:13) |>
    group_by(year) |>
    summarise(m = median(value),
              l = quantile(value, 0.025),
              u = quantile(value, 0.975)) |>
    mutate(state = unique(d$state)[states[i]], year = as.integer(year))
  if(i == 1){
    full_proj <- proj
  }else{
    full_proj <- bind_rows(full_proj, proj)
  }
}


full_proj |>
  ggplot(aes(year, log(m))) +
geom_point(aes(color = state)) +
geom_line(aes(color = state), lty = 2) +
geom_ribbon(aes(ymin = log(l),
                ymax = log(u),
                fill =  state), alpha = 0.3) +
labs(title = "Estimated (log) entries per capita", y="") +
ylim(-25,15) +
scale_x_continuous(breaks = 2018:2030)
```


Estimated (log) entries per capita
year
state
District of Columbia
Florida
Tennessee
Washington