Panel Data (Chamberlain/Mundlak) regression using Stan

Alessandro Varacca, Thomas Heckelei

July 2024

But what does panel data regression mean? (1)

$$y_{it} \sim \mathcal{D}(\boldsymbol{\theta}; \boldsymbol{\vartheta}_i)$$

- where $\boldsymbol{\theta}$ indicates a vector of *global* parameters, while $\boldsymbol{\vartheta}_i$ stands for a vector of *individual* parameters;
- As with other **linear** regression models, we want to specify a functional for $\mathbb{E}[y|X = x]$ **independently** of the the (conditional) variance $\mathbb{V}(y|X = x)$;
- Again, the most common choice of distribution allowing to do so is the **Normal** distribution;

$$y_{it} \sim \mathsf{N}(\mu_{it}, \sigma_y)$$

- or, alternatively:

$$\begin{aligned} y_{it} &= \mu_{it} + \varepsilon_{it} \\ \mu_{it} &= \alpha_i + \eta_{it} \\ \varepsilon_{it} &\sim \mathsf{N}(0, \sigma_y) \end{aligned}$$

But what does panel data regression mean? (2)

- In classical textbook examples, $\eta_{it} = \alpha_0 + \sum_{p=1}^{P} \beta_p x_{it,p}$;
- Hence:

$$y_{it} = \alpha_0 + \alpha_i + \sum_{p=1}^{P} \beta_p x_{it,p} + \varepsilon_{it}$$

- If $\mathbb{E}[\alpha_i, x_{it,p}] \neq 0$ then the estimates of β_p are inconsistent;
- Traditionally, one way to get around this problem consists of devising methods that **get rid** of α_i (for example, the *within* aka *fixed-effects* estimator);
- Another more elegant and **efficient** way consists of modelling the association between α_i and $x_{it,p}$;
- First works date back to Chamberlain (1982) and Mundlak (1978). Since we focus on the first one, we will hereafter refer to the estimated slops as β_p^{CH}.

But what does panel data regression mean? (3)

- Idea: make the distribution of α_i dependent on $x_{it,p}$! How?
- The approach is analogous to what you should have seen when discussing the *random effects* (RE) estimator, i.e., we give α_i a distribution;
- Unlike the RE case, however, we do not assume that $\mathbb{E}[\alpha_i | X = x] = 0$ but

$$\alpha_i \sim \mathsf{N}(\mu_i, \sigma_\alpha)$$

- where:

$$\mu_i = \mathbb{E}[\alpha_i | X = x] = \sum_{p=1}^{P} \sum_{t=1}^{T} \gamma_{tp} x_{i,tp}$$

- Therefore, the conditional mean of α_i depends on a vector of individual characteristics, each observed in different periods;
- It can be shown that $\mathbb{E}\left[\hat{\beta}_p^{FE}\right] = \mathbb{E}\left[\hat{\beta}_p^{CH}\right]$, but $\mathbb{V}\left(\hat{\beta}_p^{FE}\right) > \mathbb{V}\left(\hat{\beta}_p^{CH}\right)$.

How many parameters?

- Plenty more than standard linear regression, unfortunately!
  1. σ_y
  2. α_0
  3. all the β_p
  4. all the α_i
  5. all the γ_{tp}
  6. σ_α
- Total: $P + N + (T \times P) + 3$
- Clearly, calibration is going to be a lot more challenging...
- For this reason, to make our lives easier, we will begin by standardizing all our variables.

Load data and libraries

```
library(tidyverse)
library(rstan)
library(plm)

setwd("your working directory")
source("aux_functions.R")

dat <- read.csv("panel_fadn.csv")

set.seed(123)
```

Explore the data (1)

```
summary(dat, digits=2)[,1:4]
```

```
##        id            year         code_alt     cult_land
##  Min.   :  1   Min.   :2015   Min.   :1.0   Min.   :  0.1
##  1st Qu.:106   1st Qu.:2016   1st Qu.:2.0   1st Qu.:  6.1
##  Median :210   Median :2018   Median :2.0   Median : 13.0
##  Mean   :210   Mean   :2018   Mean   :2.1   Mean   : 30.3
##  3rd Qu.:315   3rd Qu.:2019   3rd Qu.:3.0   3rd Qu.: 33.3
##  Max.   :420   Max.   :2020   Max.   :3.0   Max.   :653.8
```

```
summary(dat, digits=2)[,5:8]
```

```
##   forest_land     kw_machinery     workers          earn
##  Min.   :  0.0   Min.   :   9   Min.   :0.25   Min.   :7.1e-02
##  1st Qu.:  0.0   1st Qu.:  66   1st Qu.:0.91   1st Qu.:1.0e+01
##  Median :  0.0   Median : 117   Median :1.00   Median :2.4e+01
##  Mean   :  3.4   Mean   : 185   Mean   :1.34   Mean   :5.3e+01
##  3rd Qu.:  0.2   3rd Qu.: 213   3rd Qu.:1.71   3rd Qu.:5.7e+01
##  Max.   :460.6   Max.   :1961   Max.   :5.36   Max.   :1.4e+03
```

Adjust the data

- Before exploring prior calibration in case of panel data models, we make some adjustment to our initial dataset:

```
dat_tr <- dat %>%
  mutate(forest_land_sh = forest_land / cult_land,
         earn_ha = earn / cult_land,
         kw_machinery_ha = kw_machinery / cult_land,
         workers_ha = workers / cult_land)
```

Standardize the data (1)

- When variables incorporate both individual and time variation, standardization can be performed with respect to the
  1. *grand mean/sd*;
  2. *individual mean/sd*;
  3. *time mean/sd*.
- These means can be defines as:

$$\bar{z} = NT^{-1}\sum_{t=1}^{T}\sum_{i=1}^{N} z_{it}; \quad \bar{z}_i = T^{-1}\sum_{t=1}^{T} z_{it}; \quad \bar{z}_t = N^{-1}\sum_{i=1}^{N} z_{it}$$

- Standard deviations are computed through the differences between z_{it} the corresponding mean.
- Each choice leads to a different estimate of β_p which requires transforming the coefficient back to data scale;
- For simplicity and consistency with the previous applications, we will use option (1).

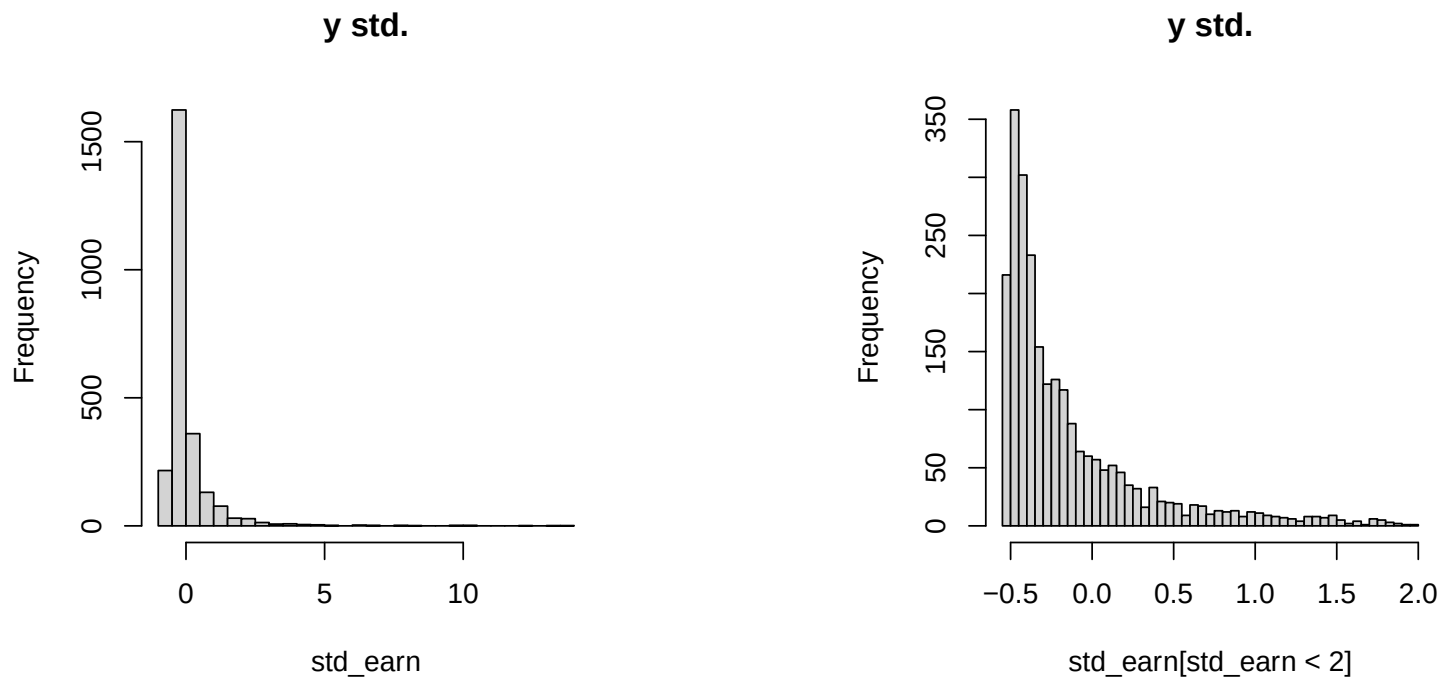
Standardize the data (2)

```
X <- dat_tr %>%
  select(cult_land, forest_land_sh, kw_machinery_ha,
         workers_ha) %>%
  mutate_all(scale) %>%
  as.matrix()

U_chmb <- dat_tr %>%
  select(year, cult_land, forest_land_sh, kw_machinery_ha,
         workers_ha) %>%
  group_by(year) %>%
  pivot_wider(names_from = year,
              values_from = cult_land:workers_ha) %>%
  unnest(everything()) %>%
  mutate_all(scale) %>%
  as.matrix()

std_earn <- as.vector(scale(dat$earn))
```

- Recall that this forces $\alpha_0 = 0$.

Standardize the data (3)

```
par(mfrow=c(1,2), pty="s")
hist(std_earn, 50, main = "y std.")
hist(std_earn[std_earn<2], 50, main = "y std.")
```




```
par(mfrow=c(1,1), pty="m")
```

Prior calibration

- Since our model involves calibrating the slope coefficients of two covariate sets, we will need more than one `Stan` program:

```
rstan_options(auto_write = TRUE)
chmb_prior <- stan_model("chamberlain_regression_prior_1.stan")
chmb_prior_re <- stan_model("chamberlain_regression_prior_2.stan")
chmb_prior_re_sl <- stan_model("chamberlain_regression_prior_3.stan")
chmb_prior_re_sl_b <- stan_model("chamberlain_regression_prior_4.stan")
```

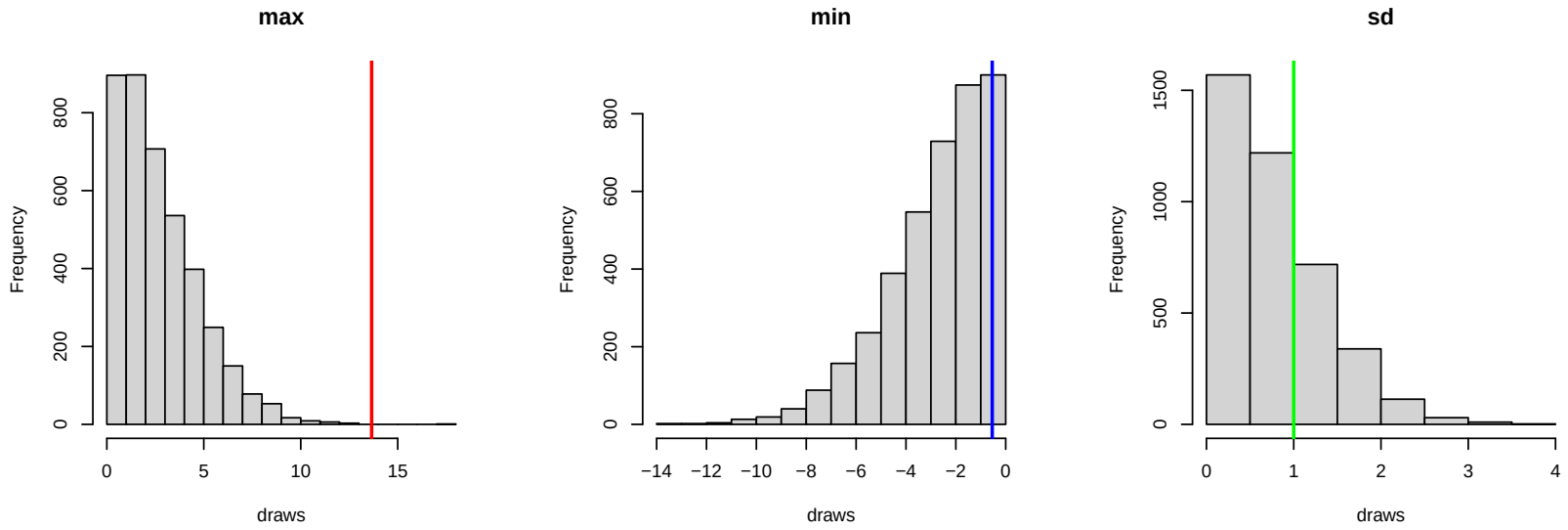
- Thanks to standardization, we can easily start at $\sigma_y \sim \mathrm{N}_+(0, 1)$ (see linear regression case study).

Prior calibration: σ_y (1)

```
dat_list <- list(
  y = std_earn,
  X = X,
  U = U_chmb,
  J = length(unique(dat$id)),
  N = nrow(dat),
  K = ncol(X),
  L = ncol(U_chmb),
  jj = dat$id,
  sd_sigma_y = 1,
  sd_inter = 0,
  mu_inter = 0
)
fit_prior <- sampling(chmb_prior,
                      data = dat_list,
                      algorithm = "Fixed_param")
fit_prior_smpl <- extract(fit_prior)

y_sim <- fit_prior_smpl$y
max_y <- apply(y_sim, 1, max); min_y <- apply(y_sim, 1, min)
sd_y  <- apply(y_sim, 1, sd)
```

Prior calibration: σ_y (2)

```
par(mfrow=c(1,3), pty="s")
hist(max_y, main = "max", xlab = "draws")
abline(v=max(std_earn),col="red", lwd=2)
hist(min_y, main = "min", xlab = "draws")
abline(v=min(std_earn),col="blue", lwd=2)
hist(sd_y, main = "sd", xlab = "draws")
abline(v=sd(std_earn),col="green", lwd=2)
```




```
par(mfrow=c(1,1), pty="m")
```

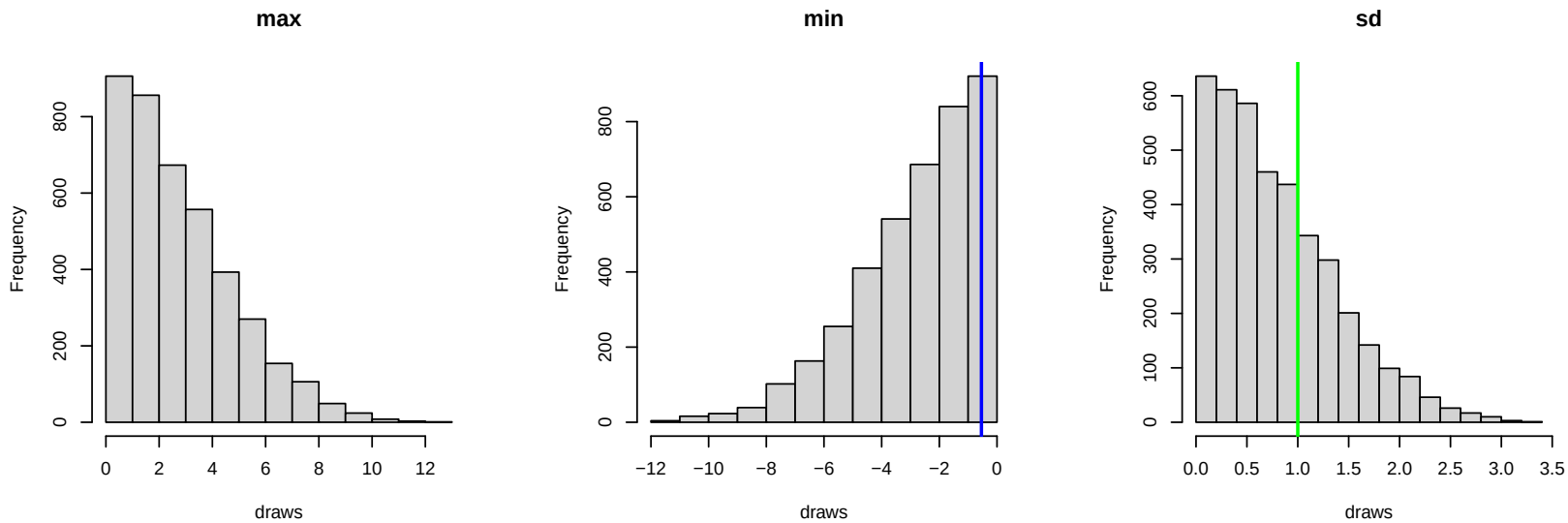
Prior calibration: β_p (1)

- Given the way we standardized our outcome and covariates, it also make sense to set $\beta_p \sim \mathrm{N}(0, 1)$ (see linear regression case study);

```
dat_list$sd_beta <- 1

fit_prior <- sampling(chmb_prior,
                      data = dat_list,
                      algorithm = "Fixed_param")
fit_prior_smpl <- extract(fit_prior)

y_sim <- fit_prior_smpl$y

max_y <- apply(y_sim, 1, max)
min_y <- apply(y_sim, 1, min)
sd_y  <- apply(y_sim, 1, sd)
```

Prior calibration: β_p (2)

```
par(mfrow=c(1,3), pty="s")
hist(max_y, main = "max", xlab = "draws")
abline(v=max(std_earn),col="red", lwd=2)
hist(min_y, main = "min", xlab = "draws")
abline(v=min(std_earn),col="blue", lwd=2)
hist(sd_y, main = "sd", xlab = "draws")
abline(v=sd(std_earn),col="green", lwd=2)
```




```
par(mfrow=c(1,1), pty="m")
```

Prior calibration: β_p (3)

- This configuration seems reasonable, although the observed maximum values still fall in the right tail of the PrPD;
- This may be hard to directly address while keeping the minimum and the standard deviation coherent with the observed ones;
- The distribution of NFI is right-skewed and a Normal model for y_{it} might not be the best choice;
- We may try to circumvent this problem through the individual heterogeneity components.

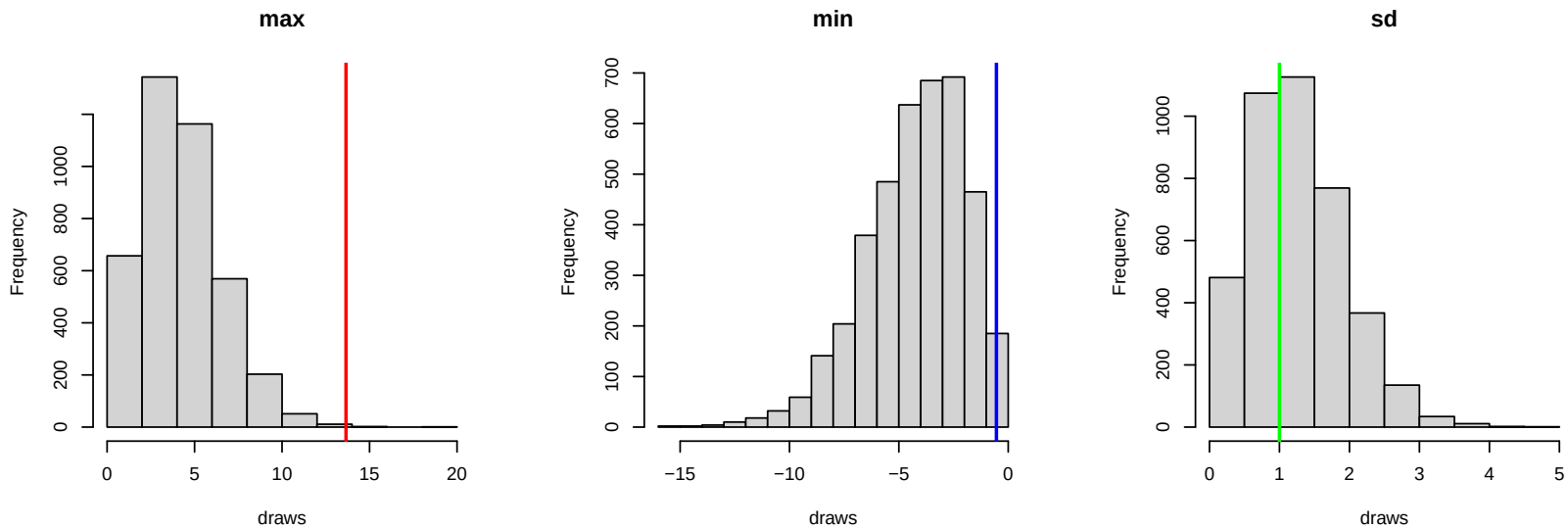
Prior calibration: σ_α (1)

- For the time being, let us assume that $\gamma_{tp} = 0$ for all p and t;
- Since $\text{sd}(\mathbf{y}) = 1$ it make sense to set $\sigma_\alpha \sim \text{N}_+(0, 1)$:

```
dat_list$sd_sigma_a <- 1

fit_prior_re <- sampling(chmb_prior_re,
                         data = dat_list,
                         algorithm = "Fixed_param")
fit_prior_smpl <- extract(fit_prior_re)
y_sim <- fit_prior_smpl$y

max_y <- apply(y_sim, 1, max)
min_y <- apply(y_sim, 1, min)
sd_y  <- apply(y_sim, 1, sd)
```

Prior calibration: σ_α (2)

```
par(mfrow=c(1,3), pty="s")
hist(max_y, main = "max", xlab = "draws")
abline(v=max(std_earn),col="red", lwd=2)
hist(min_y, main = "min", xlab = "draws")
abline(v=min(std_earn),col="blue", lwd=2)
hist(sd_y, main = "sd", xlab = "draws")
abline(v=sd(std_earn),col="green", lwd=2)
```




```
par(mfrow=c(1,1), pty="m")
```

Prior calibration: σ_α (3)

- This setting seems OK as well, although maximum values are not quite there yet.
- Being NFI asymmetric, tweaking the additive **zero-mean** individual heterogeneity and σ_y can only help so much.

Prior calibration: γ_{tp} (1)

- We now study the PrPD when $\mathbb{E}[\alpha_i|X = x] \neq 0$;
- Using the Chamberlain approach, for $t_0 = 2015$ and $T = 2020$, we need to include 30 covariates and that many coefficients;
- When either P or T (or both) are large, this modelling approach can lead to overfit and, as a consequence, (very) high variance;
- One way to mitigate this issue is called *regularization*: a modelling strategy that forces some coefficients to small values;
- This be easily implemented in any Bayesian regression model by simply imposing a prior on the variance of the slope terms:

$$\gamma_{tp} \sim \mathsf{N}(0, \sigma_\gamma)$$
$$\sigma_\gamma \sim \mathsf{N}_+(0, \rho_\gamma)$$

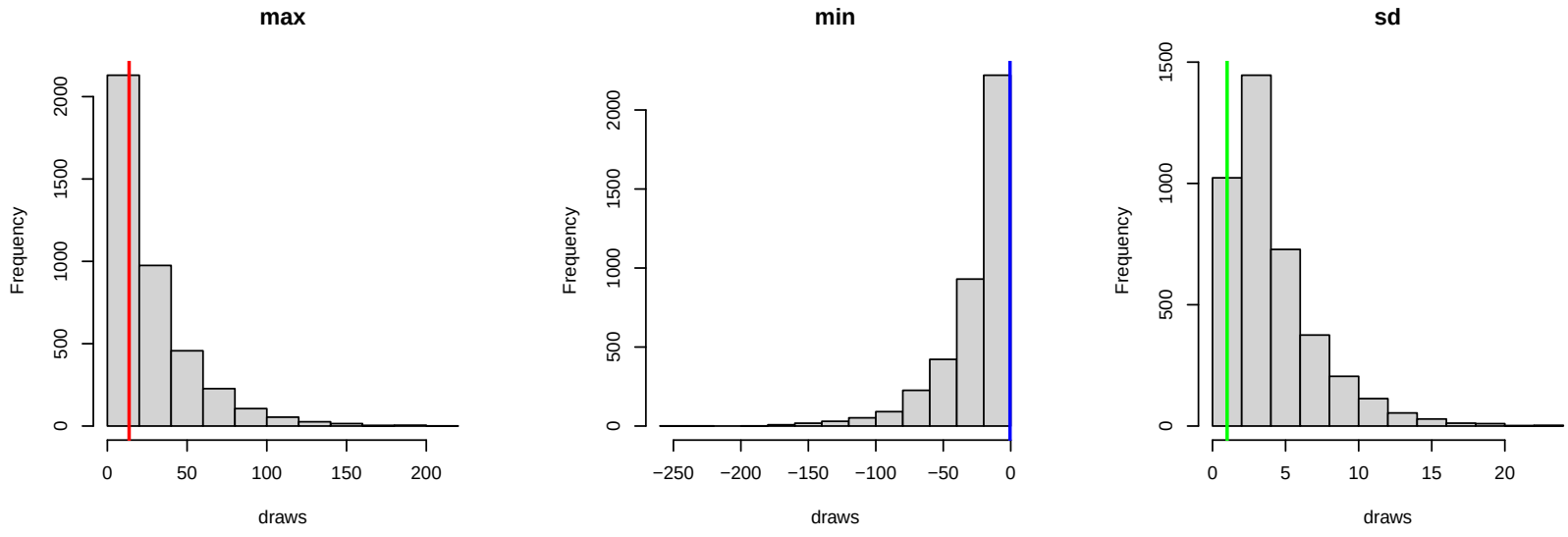
Prior calibration: γ_{tp} (2)

- Coherently with the above choices, we begin with $\rho_\gamma = 1$:

```
dat_list$sd_sigma_g <- 1

fit_prior_re_sl <- sampling(chmb_prior_re_sl,
                            data = dat_list,
                            algorithm = "Fixed_param")
fit_prior_smpl <- extract(fit_prior_re_sl)
y_sim <- fit_prior_smpl$y

max_y <- apply(y_sim, 1, max)
min_y <- apply(y_sim, 1, min)
sd_y  <- apply(y_sim, 1, sd)
```

Prior calibration: γ_{tp} (3)

```
par(mfrow=c(1,3), pty="s")
hist(max_y, main = "max", xlab = "draws")
abline(v=max(std_earn),col="red", lwd=2)
hist(min_y, main = "min", xlab = "draws")
abline(v=min(std_earn),col="blue", lwd=2)
hist(sd_y, main = "sd", xlab = "draws")
abline(v=sd(std_earn),col="green", lwd=2)
```




```
par(mfrow=c(1,1), pty="m")
```

Prior calibration: γ_{tp} (4)

- Introducing μ_i make the PrPD leptokurtic (i.e., it thickens the tail of the starting normal distribution model);
- Turns it to something similar to a Student't thnaks to the 'deviant' behavour/characteristic of some individuals;
- See how the choice of a Normal model for y is not too restrictie after all;
- What we can't still caoture, however, is symmetry. For that we need a different distribution family.

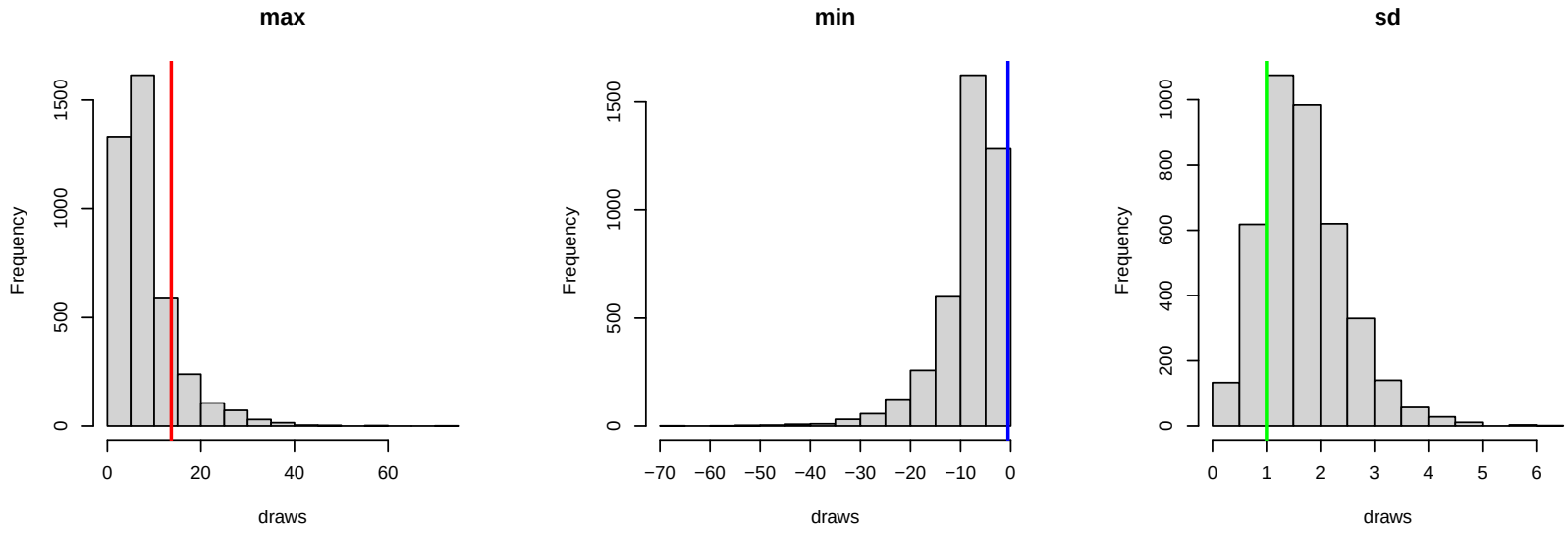
Prior calibration: γ_{tp} (5)

- To reduce the kurtosis, let's set $\rho_\gamma = 0.25$

```
dat_list$sd_sigma_g <- .25

fit_prior_re_sl <- sampling(chmb_prior_re_sl,
                            data = dat_list,
                            algorithm = "Fixed_param")
fit_prior_smpl <- extract(fit_prior_re_sl)
y_sim <- fit_prior_smpl$y

max_y <- apply(y_sim, 1, max)
min_y <- apply(y_sim, 1, min)
sd_y  <- apply(y_sim, 1, sd)
```

Prior calibration: γ_{tp} (6)

```
par(mfrow=c(1,3), pty="s")
hist(max_y, main = "max", xlab = "draws")
abline(v=max(std_earn),col="red", lwd=2)
hist(min_y, main = "min", xlab = "draws")
abline(v=min(std_earn),col="blue", lwd=2)
hist(sd_y, main = "sd", xlab = "draws")
abline(v=sd(std_earn),col="green", lwd=2)
```




```
par(mfrow=c(1,1), pty="m")
```

Estimation

```
chmb <- stan_model("chamberlain_regression.stan")

fit <- sampling(chmb,
                data = dat_list,
                chains = 4,
                cores = 4)
sample_fit <- extract(fit)

beta <- sample_fit$beta
alpha <- sample_fit$alpha
y_pred <- sample_fit$y_rep

max_y_pr <- apply(y_pred, 1, max)
min_y_pr <- apply(y_pred, 1, min)
sd_y_pr  <- apply(y_pred, 1, sd)
```

Exploring the results (2)

- Since $\mathbf{x}_{it}$ and y_{it} have been standardized, we need to map the estimated coefficients back to the original scale of the data:

```
X_unsc <- dat_tr %>%
  select(cult_land, forest_land_sh, kw_machinery_ha, workers_ha) %>%
  as.matrix()

sc_factor <- sd(dat$earn)/apply(X_unsc, 2, sd)
beta_backsc <- sweep(beta, 2, sc_factor, "*")

fe_reg_unsc <- plm(earn ~ cult_land
                        + forest_land_sh
                        + kw_machinery_ha
                        + workers_ha, data = dat_tr)
```

Exploring the results (3)

```
t(round(apply(beta_backsc, 2, quantile, c(.025, .975)), 3))
```

```
##
##            2.5%   97.5%
##    [1,]   0.582   1.090
##    [2,]  19.310 109.901
##    [3,]  -0.494   0.172
##    [4,] -10.684  23.587
```

```
round(confint(fe_reg_unsc), 3)
```

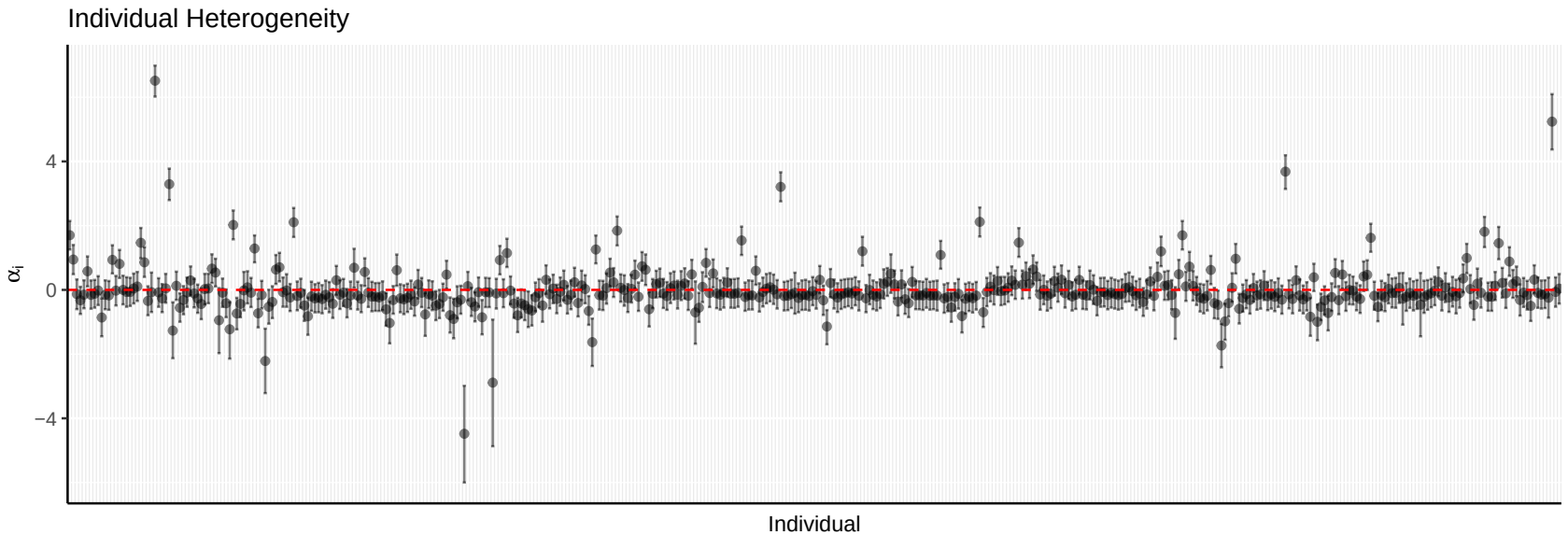
```
##                    2.5 %  97.5 %
## cult_land          0.582   1.103
## forest_land_sh    16.998 113.985
## kw_machinery_ha   -0.510   0.174
## workers_ha       -11.449  24.576
```

- Very close, but Chamberlain approch, in this case, a touch more efficient.

Exploring the results: α_i

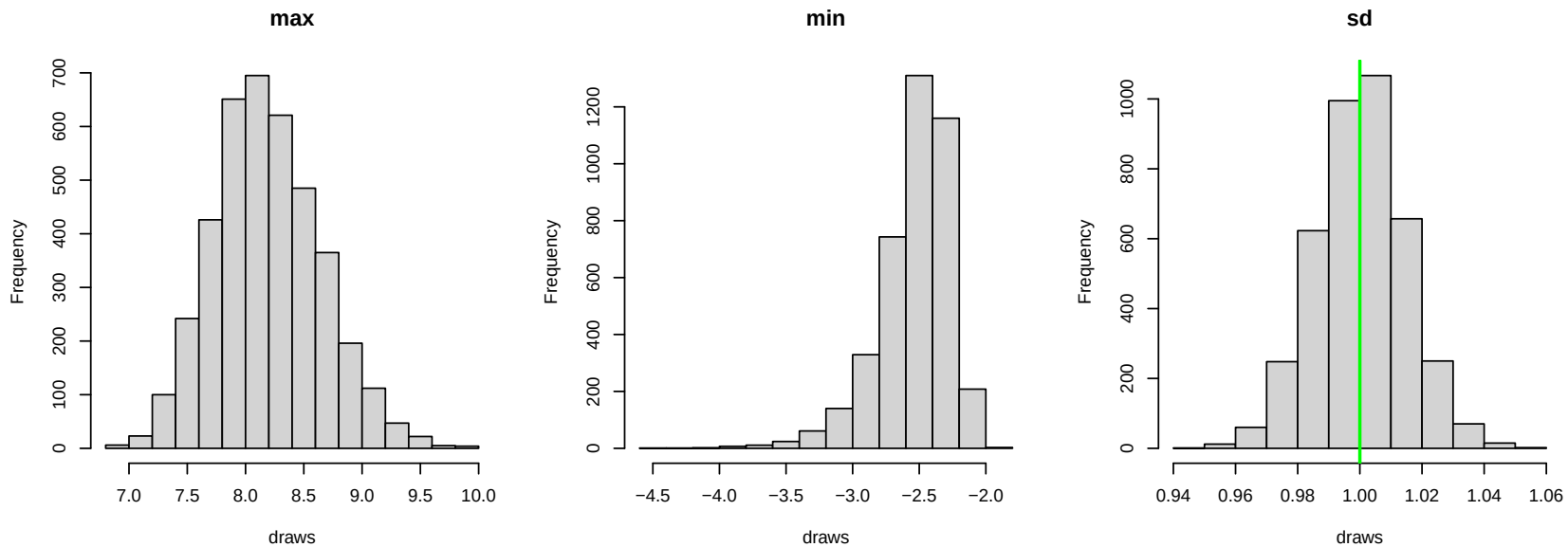
```
data.frame(alpha) %>%
  pivot_longer(everything(), values_to = "individual") %>%
  mutate(ind = as.numeric(gsub("X", "", name))) %>%
  group_by(name) %>%
  summarise(low = quantile(individual, .025),
            upp = quantile(individual, .975),
            med = quantile(individual, .5)) %>%
  ggplot(aes(x = name)) +
  geom_point(aes(y = med), alpha = .5) +
  geom_errorbar(aes(ymin = low, ymax = upp), alpha = .5) +
  geom_hline(yintercept = 0, col = "red", lty = 2) +
  ggtitle("Individual Heterogeneity") +
  ylab(expression(alpha[i])) +
  xlab("Individual") +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.line = element_line(),
        title = element_text(size = 10))
```

Exploring the results: α_i



Model checking (internal consistency - 1)

```
par(mfrow=c(1,3), pty="s")
hist(max_y_pr, main = "max", xlab = "draws")
abline(v=max(std_earn),col="red", lwd=2)
hist(min_y_pr, main = "min", xlab = "draws")
abline(v=min(std_earn),col="blue", lwd=2)
hist(sd_y_pr, main = "sd", xlab = "draws")
abline(v=sd(std_earn),col="green", lwd=2)
```




```
par(mfrow=c(1,1), pty="m")
```

Model checking (internal consistency - 2)

```
plot_prior_dens(y_pred, std_earn, main = "PPD")
```