Bayesian Statistics: Laboratory 4

Vincenzo Gioia

DEAMS

University of Trieste

vincenzo.gioia@units.it

Building D, room 2.13

Office hour: Friday, 15 - 17

05/05/2023

Load the following packages

```
library(rstan)
library(rstanarm)
library(ggplot2)
library(bayesplot)

theme_set(bayesplot::theme_default())

set.seed(1)
```

Bayesian Inference in Stan

- During this lab and the following one, we consider a *real* problem, albeit the data are simulated, and we focus on the following Bayesian data analysis workflow
  - Model building
  - Model checking
  - Model expansion
  - Model comparison
- We leverage **Stan** to carry out Bayesian inference for the Coackroaches' example

Outline

Section 1

Cockroaches' example: Problem, goals and materials

Cockroaches' example: The problem

- Imagine that you are a statistician working as an independent contractor
- One of your clients is a company that owns many residential buildings throughout New York City
- The property manager explains that they are concerned about the number of cockroach complaints that they receive from their buildings
- They tried to solve the problem with monthly visits from a pest inspector but this solution is
  - expensive
  - not very effective due to the difficulty in finding the tenants at home

Cockroaches' example: Possible solution and goals

- Possible alternative: deploy long term bait stations installed throughout the apartment building
- The property manager
  - asks you to explore the relationship between roaches and bait stations to shed light on the effectiveness of this solution
  - would also like to learn how these results generalize to buildings they haven't treated, so they can understand the potential costs of pest control at buildings not recorded for the experiment and also the ones they are acquiring

Cockroaches' example: Experiment and first attempt

- A subset of the company's buildings (10) have been randomly selected for an experiment:
  - At the beginning of each month, a pest inspector randomly places a **number of bait stations throughout the building** (thereafter `traps`)
  - At the end of the month, the manager records the total **number of cockroach complaints in that building** (thereafter `complaints`)
- This is done for 12 successive months, for a total of 120 observations
- At first, we will **model the number of complaints as a function of the number of traps**, ignoring variation over time and across buildings

Cockroaches' example: Data

- Load and explore the dataset in the file `pest_data.RDS`
- The dataset includes 14 variables and 120 observations:
  - What is the structure of your data?
  - What kind of variables do you have?
  - What is the outcome variable and the explanatory ones?

```
data <- readRDS('pest_data.RDS')
str(data)
```

Cockroaches' example: Data structure

```
## 'data.frame':    120 obs. of  14 variables:
##  $ mus                  : num  0.369 0.359 0.282 0.129 0.452 ...
##  $ building_id          : int  37 37 37 37 37 37 37 37 37 37 ...
##  $ wk_ind               : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ date                 : Date, format:  ...
##  $ traps                : num  8 8 9 10 11 11 10 10 9 9 ...
##  $ floors               : num  8 8 8 8 8 8 8 8 8 8 ...
##  $ sq_footage_p_floor   : num  5149 5149 5149 5149 5149 ...
##  $ live_in_super        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ monthly_average_rent: num  3847 3847 3847 3847 3847 ...
##  $ average_tenant_age   : num  53.9 53.9 53.9 53.9 53.9 ...
##  $ age_of_building      : num  47 47 47 47 47 47 47 47 47 47 ...
##  $ total_sq_foot        : num  41192 41192 41192 41192 41192 ...
##  $ month                : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ complaints           : num  1 3 0 1 0 0 4 3 2 2 ...
```

Cockroaches' example: Data description

The variables we will be using in this lab and in the following labs:

- `complaints`: Number of complaints per building per month
- `building_id`: Unique building identifier
- `traps`: Number of traps used per month per building
- `date`: Date at which the number of complaints are recorded
- `month`: Month of the year
- `live_in_super`: Whether the building has a live-in -superintendent
- `age_of_building`: Age of the building
- `total_sq_foot`: Total square footage of the building
- `average_tenant_age`: Average age of the tenants per building
- `monthly_average_rent`: Average monthly rent per building
- `floors`: Number of floors per building

Cockroaches' example: Data

Create a new data frame only containing such variables

```
Svar <- c("complaints", "building_id", "traps", "date",
          "live_in_super","age_of_building", "month",
          "total_sq_foot", "average_tenant_age",
          "monthly_average_rent", "floors")
pest_data <- data[, Svar]
```

Number of buildings

```
N_buildings <- length(unique(pest_data$building_id))
N_buildings
```

```
## [1] 10
```

Cockroaches' example: Exploratory analysis

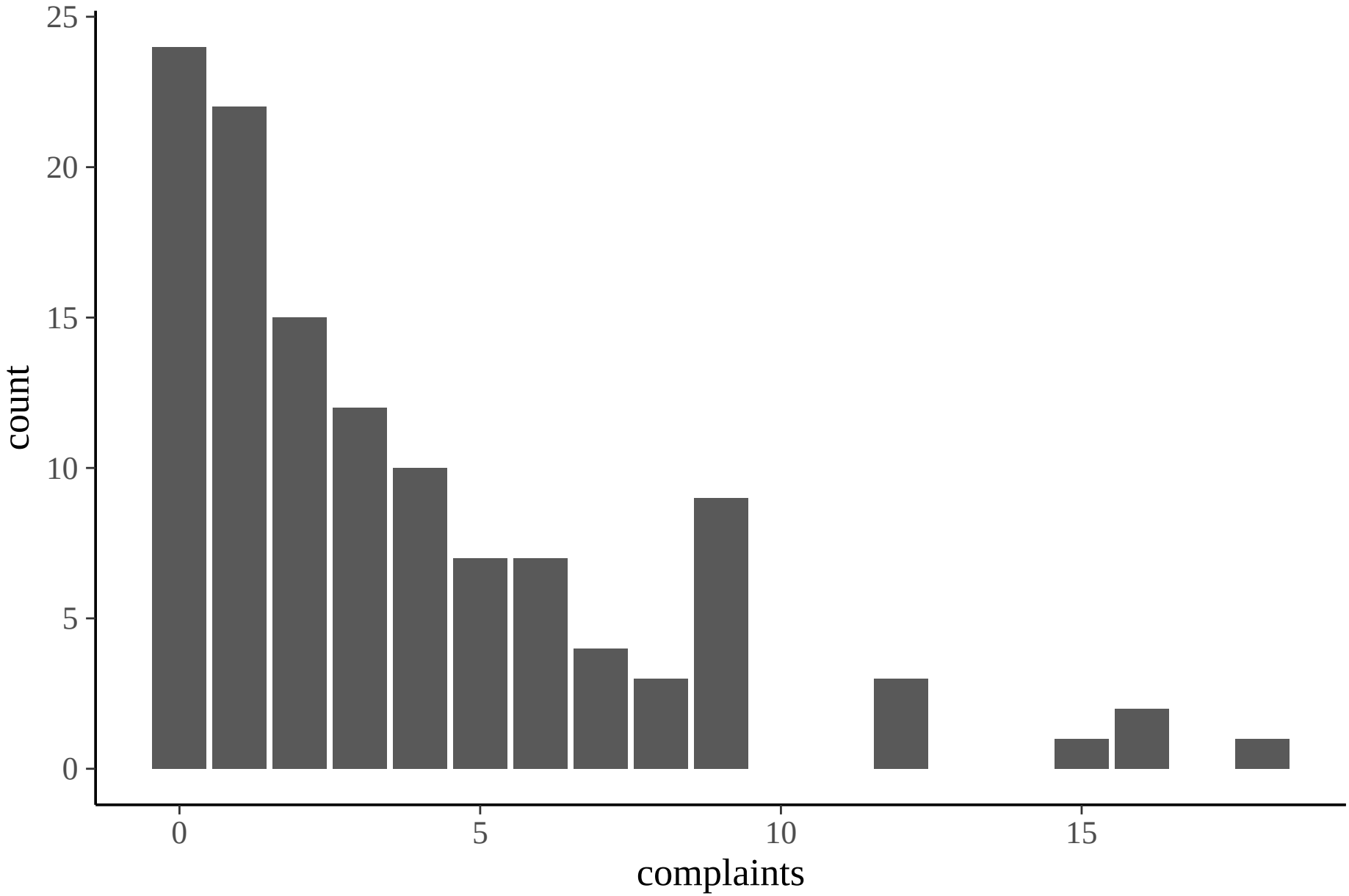
Make some plots for:

- Exploring the distribution of the number of complaints
- Analysing the relation between complaints and traps

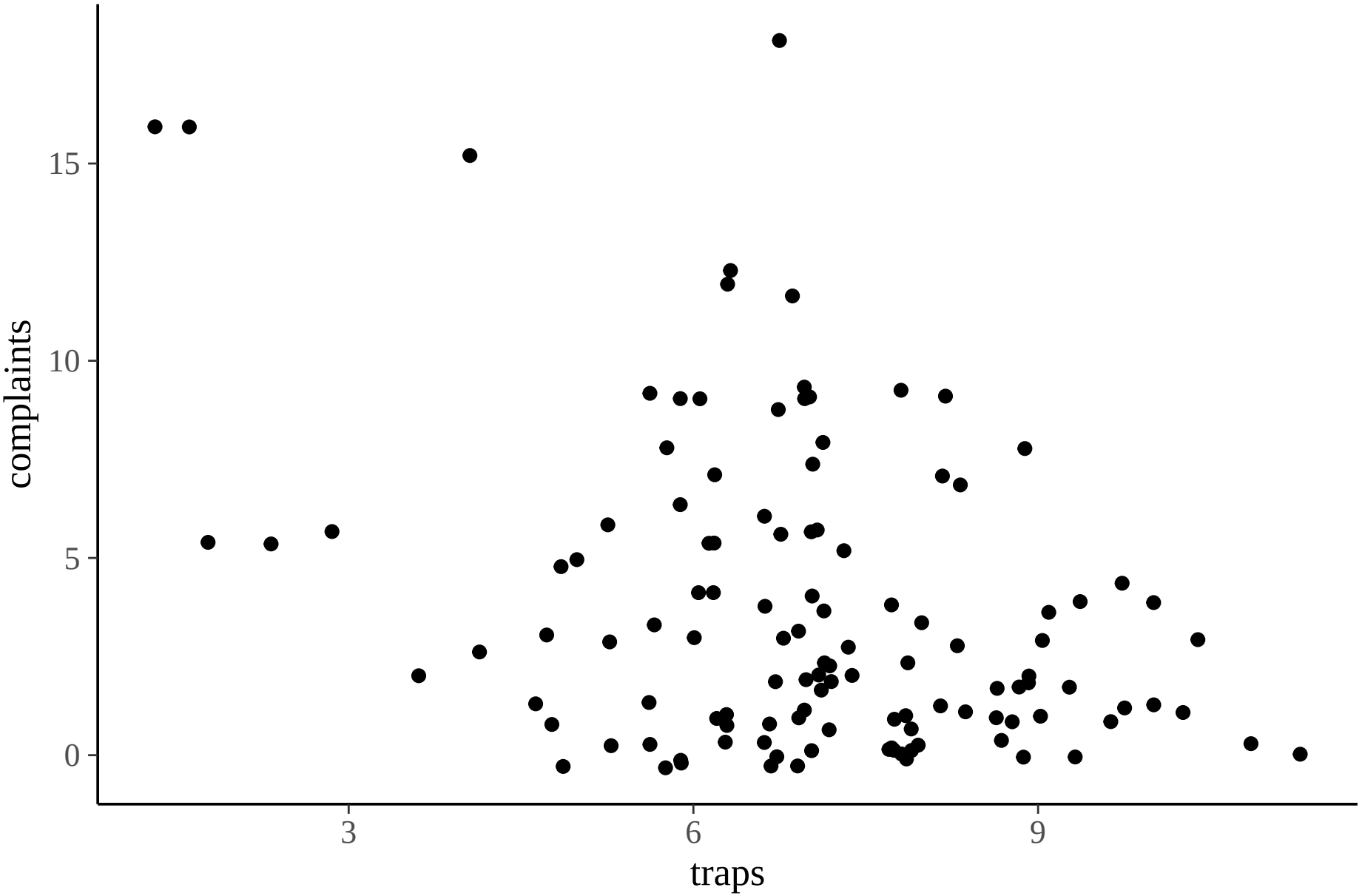
```
ggplot(pest_data, aes(x = complaints)) +
  geom_bar()

ggplot(pest_data, aes(x = traps, y = complaints)) +
    geom_jitter()
```

Cockroaches' example: Exploratory analysis



Cockroaches' example: Exploratory analysis



Section 2

Cockroaches' example: Poisson regression model

Cockroaches' example: (simple) Poisson regression model

- How we could analyse the number of complaints?
- Knowing that the number of complaints over a month is unlikely to be zero and that rarely there are a large number of complaints over a month, a sensible probability distribution assumption to model the outcome variable `complaints` could be the Poisson distribution
- We start modelling the number of complaints using the number of traps through a Poisson regression model

$$\begin{aligned} \text{complaints}_i &\sim \text{Poisson}(\lambda_i), \quad i = 1, \ldots, 120 \\ \lambda_i &= \exp(\eta_i) \\ \eta_i &= \alpha + \beta_1 \, \text{traps}_i \end{aligned}$$

- Our model's mean parameter is the rate of complaints per 1 month

Cockroaches' example: (simple) Poisson regression model

- Of course, the model can be equivalently written in a different way

$$\begin{aligned} \text{complaints}_{b,t} &\sim \text{Poisson}(\lambda_{b,t}), \quad b = 1, \ldots, 10, \quad t = 1, \ldots, 12 \\ \lambda_{b,t} &= \exp\left(\eta_{b,t}\right) \\ \eta_{b,t} &= \alpha + \beta_1 \, \text{traps}_{b,t} \end{aligned}$$

- Here, b is an index for the building and t for the time

Cockroaches' example: (simple) Poisson regression model

Organise the data: arrange the data into a list to match Stan requirements

```
stan_dat <- list(
  N = nrow(pest_data),
  complaints = pest_data$complaints,
  traps = pest_data$traps
)
str(stan_dat)
```

```
## List of 3
##  $ N         : int 120
##  $ complaints: num [1:120] 1 3 0 1 0 0 4 3 2 2 ...
##  $ traps     : num [1:120] 8 8 9 10 11 11 10 10 9 9 ...
```

Cockroaches' example: (simple) Poisson regression model

- Our model is saved into `simple_poisson_regression.stan`. Take a look to it. It includes 5 blocks:
  - functions
  - data
  - parameters
  - model
  - generated quantities
- At first, we will explore the data, parameters and model blocks, which are needed for sampling from the posterior
- Then, we analyse the functions and generated quantities blocks, which are needed for the posterior predictive checks (PPCs)

Cockroaches' example: (simple) Poisson regression model

- Data block:
  - sample size (constrain to be $>= 1$)
  - outcome (a vector of integer of length `N`)
  - covariate (a vector of length `N`)

```
data{
  int<lower=1> N;
  int<lower=0> complaints[N];
  vector<lower=0>[N] traps;
}
```

Cockroaches' example: (simple) Poisson regression model

- Parameters block: our model accept two parameters `alpha` and `beta1`, both real and unconstrained

```
parameters {
  real alpha;
  real beta1;
}
```

Cockroaches' example: (simple) Poisson regression model

- Model block:
  - Prior specification: we considered two weakly informative prior, since we expect negative slope on traps and a positive intercept, that is

$$\alpha \sim \mathcal{N}(\log(4), 1) \qquad \beta_1 \sim \mathcal{N}(-0.25, 1)$$

  - Likelihood: implemented by using the `poisson_log(eta)`, which is more efficient and stable than `poisson(exp(eta))`

```
model {
  beta1 ~ normal(-0.25, 1);
  alpha ~ normal(log(4), 1);
  complaints ~ poisson_log(alpha + beta1 * traps);
  // Alternatively
  // complaints ~ poisson(exp(alpha + beta1 * traps));
}
```

Cockroaches' example: (simple) Poisson regression model

- Model block can be equivalently specified by means of target statement

```
model {
 target += normal_lpdf(alpha | log(4), 1);
 target += normal_lpdf(beta1 | -0.25, 1);
 target += poisson_log_lpmf(complaints| alpha + beta1 * traps);
}
```

- Or by mixing sampling notation and target statement

```
model {
 beta1 ~ normal(-0.25, 1);
 alpha ~ normal(log(4), 1);
 target += poisson_log_lpmf(complaints|alpha + beta1 * traps);
}
```

Cockroaches' example: (simple) Poisson regression model

- Then, we compile the model

```
comp_model_P <- stan_model('simple_poisson_regression.stan')
```

- Sample draws from the posterior

```
fit_P1 <- sampling(comp_model_P,
                   data = stan_dat,
                   refresh = 0)
```

- Print the posterior summary of the regression parameters

```
print(fit_P1, pars = c('alpha','beta1'),
      probs = c(0.1, 0.5, 0.9))
```

Cockroaches' example: (simple) Poisson regression model

- Negative β_1 implies that a high number of traps appears to be associated with fewer complaints in the following month

```
## Inference for Stan model: simple_poisson_regression.
## 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   10%   50%   90% n_eff
## alpha  2.58    0.01 0.16  2.38  2.57  2.78   731
## beta1 -0.19    0.00 0.02 -0.22 -0.19 -0.16   753
##       Rhat
## alpha    1
## beta1    1
##
## Samples were drawn using NUTS(diag_e) at Fri May 05 17:55:17 2023.
## 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).
```

Cockroaches' example: (simple) Poisson regression model

- Alternative: use the `stan_glm()` wrapper of the **rstanarm** package

```
fit2 <- stan_glm(complaints ~ traps,
                 data = pest_data,
                 family = poisson,
                 prior_intercept = normal(log(4), 1),
                 prior = normal(-0.25, 1),
                 refresh = 0, warmup = 1000,
                 chains = 4, iter = 2000)
round(summary(fit2)[1 : 2, 1 : 8], 2)
```

```
##                mean mcse   sd   10%   50%   90%
## (Intercept)    2.61    0 0.16  2.40  2.61  2.81
## traps         -0.20    0 0.02 -0.23 -0.20 -0.17
##                n_eff Rhat
## (Intercept)     2553    1
## traps           2244    1
```

Cockroaches' example: (simple) Poisson regression model

- MLE

```
fit3 <- glm(complaints ~ traps,
            data = pest_data,
            family = poisson)
round(summary(fit3)$coefficients[, 1 : 2], 2)
```

```
##             Estimate Std. Error
## (Intercept)     2.61       0.16
## traps          -0.20       0.02
```

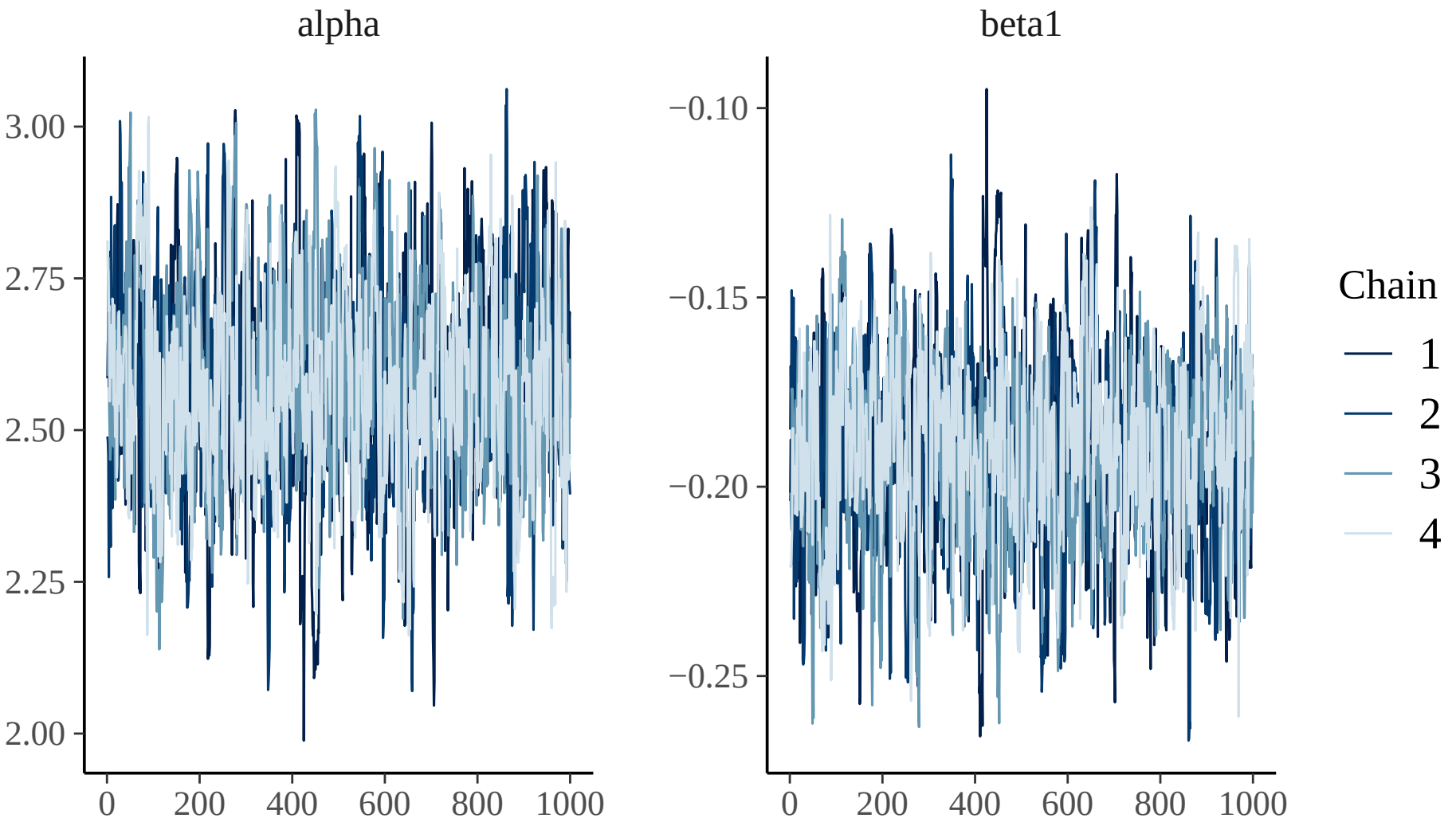
Cockroaches' example: (simple) Poisson regression model

- Visualizing Markov chain Monte Carlo (MCMC) draws from the posterior distribution of the parameters (traceplot and autocorrelation function, histogram and scatterplot)

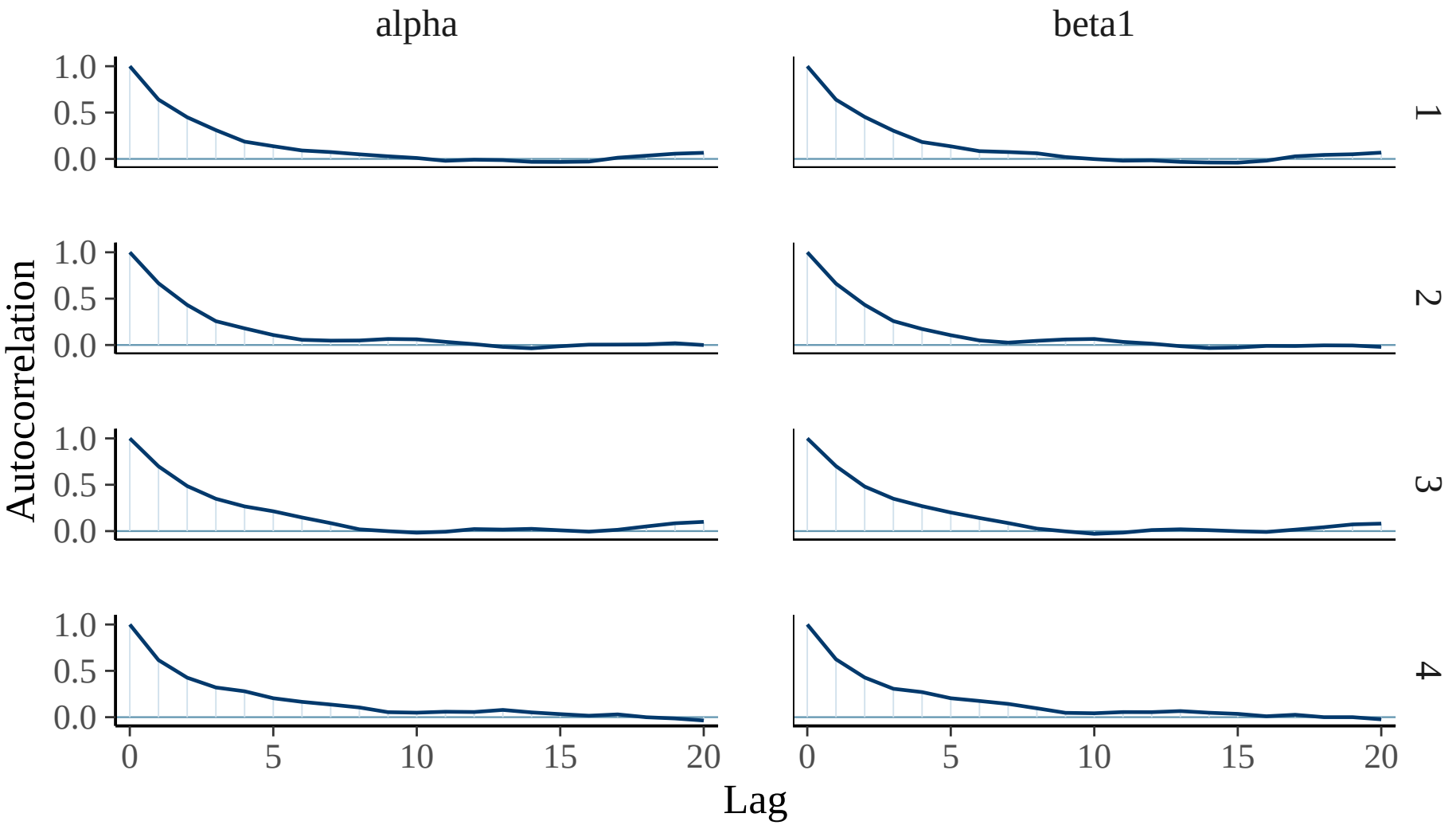
```
res_array <- as.array(fit_P1, pars = c('alpha', 'beta1'))
res_matrix <- as.matrix(fit_P1, pars = c('alpha', 'beta1'))
mcmc_trace(res_array)
mcmc_acf(res_array)
mcmc_acf(res_matrix)
mcmc_hist(res_array)
mcmc_scatter(res_array, alpha = 0.2)
```

- See in the R script, equivalent or alternative plots according to the way of extracting the sample draws. In addition take a look to: https://cran.r-project.org/web/packages/bayesplot/vignettes/plotting-mcmc-draws.html

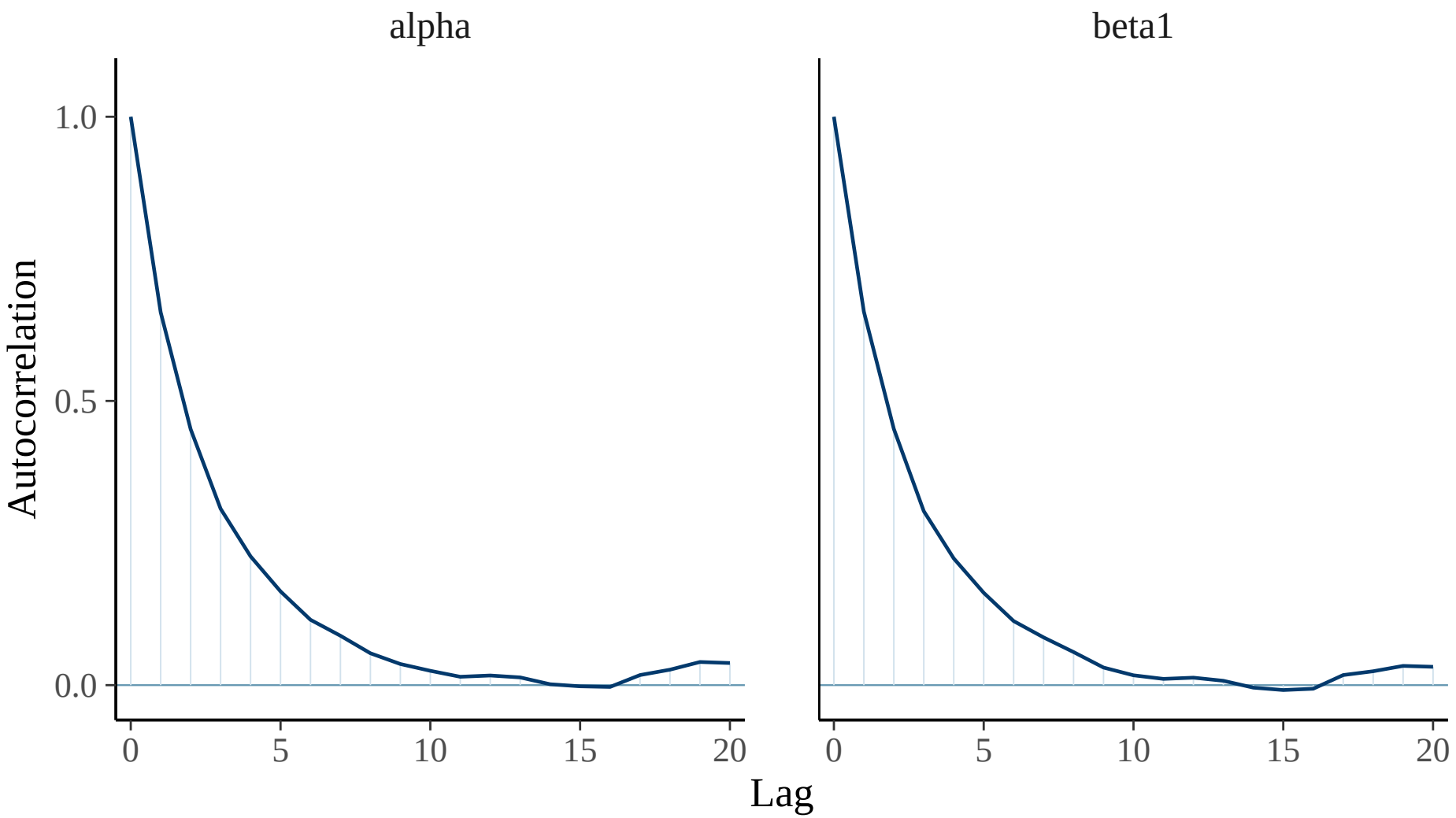
Cockroaches' example: (simple) Poisson regression model



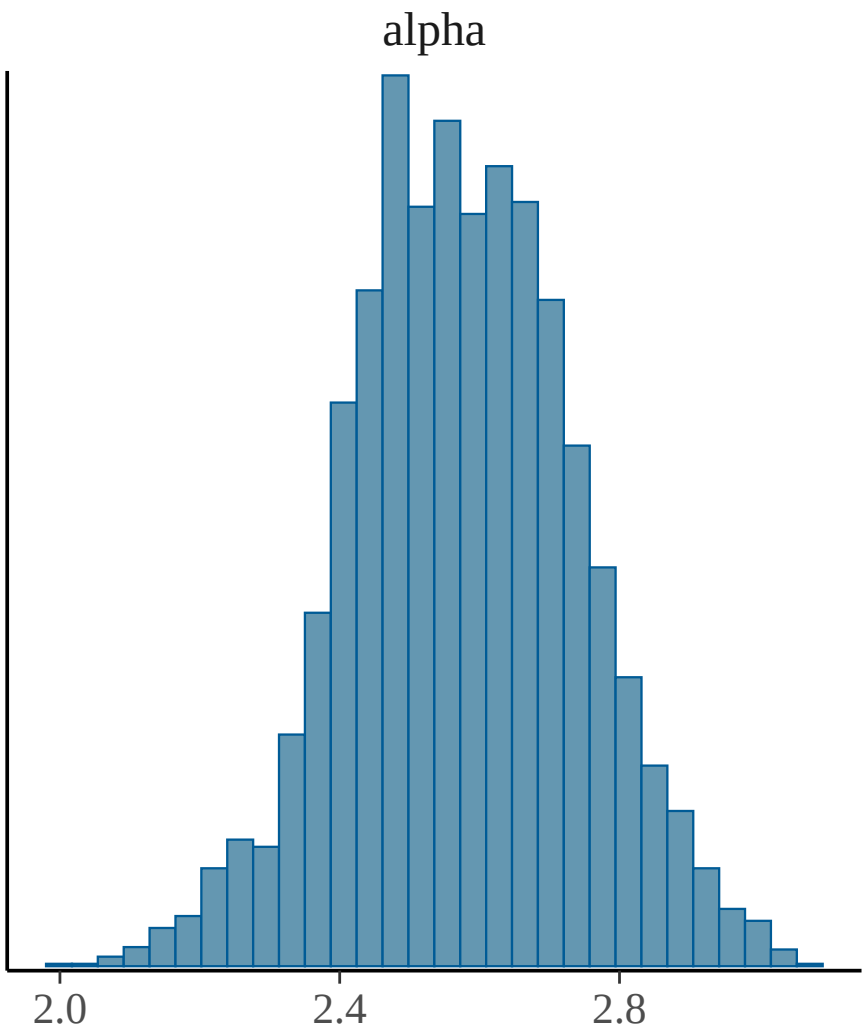
Cockroaches' example: (simple) Poisson regression model

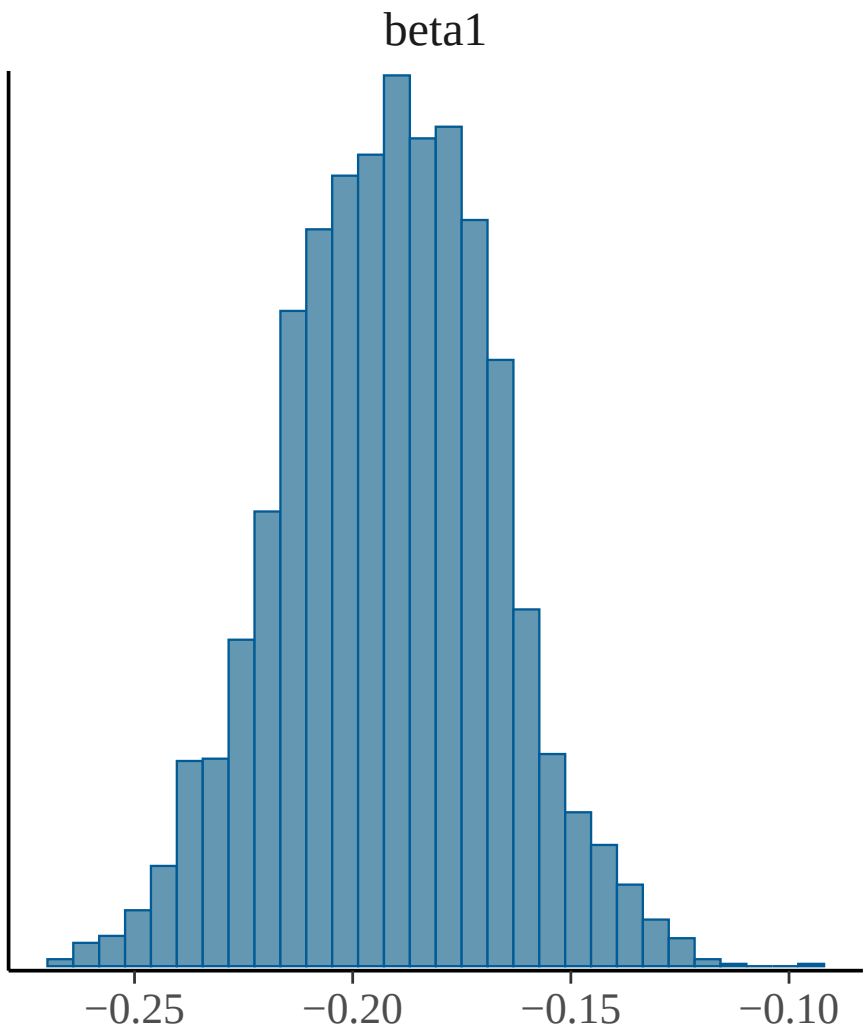


Cockroaches' example: (simple) Poisson regression model

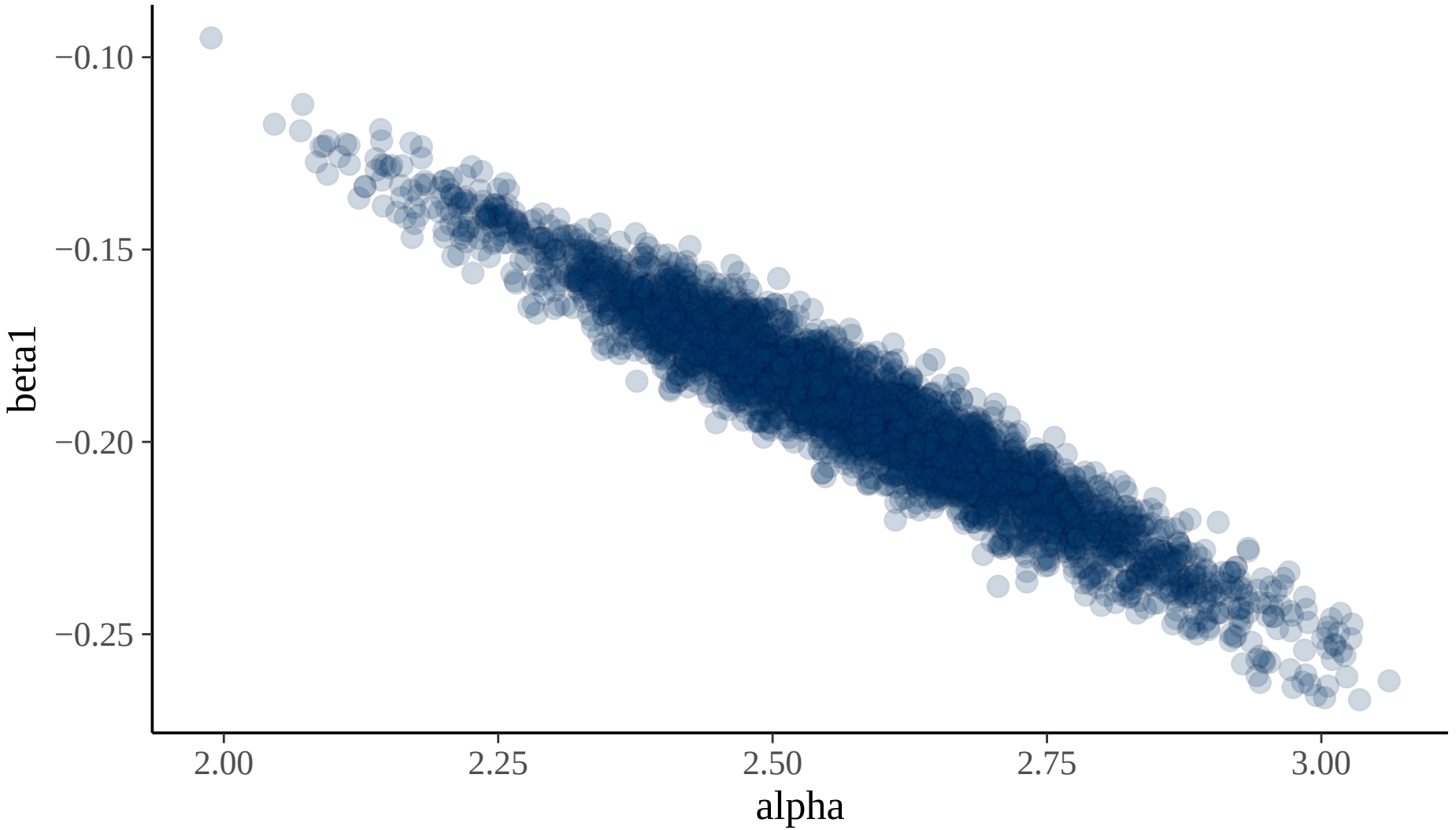


Cockroaches' example: (simple) Poisson regression model





Cockroaches' example: (simple) Poisson regression model



Cockroaches' example: (simple) Poisson regression model

- The plots above allow to inspect the draws from the posterior distribution, but they don't say anything about how well the model fits
- For such purposes, posterior predictive checks (PPCs) aim to compare the observed data with the simulated data from the posterior predictive distribution
- Idea: if a model is a good fit then we should be able to use it to generate data that looks a lot like the data we observed
- In the following, we define y^{rep} as the replicated data that could have been observed (in-sample replication). We usually distinguish between y^{rep} and $\tilde{y}$, which is any future observable value or vector of observable quantities (out-of-sample replication)

Cockroaches' example: (simple) Poisson regression model

- Recall that if we disallow improper priors, then Bayesian modeling is generative

- To generate the data used for PPCs we simulate from the posterior predictive distribution, that is the distribution of the outcome variable implied by a model after using the observed data y to update our beliefs about unknown model parameters θ

- The posterior predictive distribution for observation y^{rep}, conditionally on X (a matrix of predictor variables), is

$$p(y^{rep}|y, X) = \int_{\Theta} p(y^{rep}|\theta, X)\pi(\theta|y, X)d\theta$$

- For each draw $s = 1, \ldots, S$, of the parameters from the posterior distribution, $\theta^{(s)} \sim \pi(\theta|y, X)$, we draw $\tilde{y}^{rep\,(s)}$ from the posterior predictive distribution by simulating from the data model conditional on parameters $\theta^{(s)}$ (and X)

Cockroaches' example: (simple) Poisson regression model

- Sample predicted values from the model for posterior predictive checks is carried out in the **generated quantities** block
- For each observation we simulate y^{rep} using the simulated parameters
- The result will be a matrix of draws y^{rep} with `nrow = iter` and `ncol = n` (the sample size)

```
generated quantities{
  int y_rep[N];
  for (n in 1:N) {
    real eta_n = alpha + beta1 * traps[n];
    y_rep[n] = poisson_log_safe_rng(eta_n);
  }
}
```

Cockroaches' example: (simple) Poisson regression model

- The **generated quantities** block above make use of a user defined function that we specify in the **functions** block

- This is done to avoid potential numerical problems during warmup which could appear using `poisson_log_rng()`

```
functions{
 int poisson_log_safe_rng(real eta) {
    real pois_rate = exp(eta);
    if (pois_rate >= exp(20.79))
      return -9;
    return poisson_rng(pois_rate);
  }
}
```

Cockroaches' example: (simple) Poisson regression model

- Then, we extract the generated quantity `y_rep` (Each row of the matrix is a draw from the posterior predictive distribution, i.e. a vector with one element for each of the data points in y)

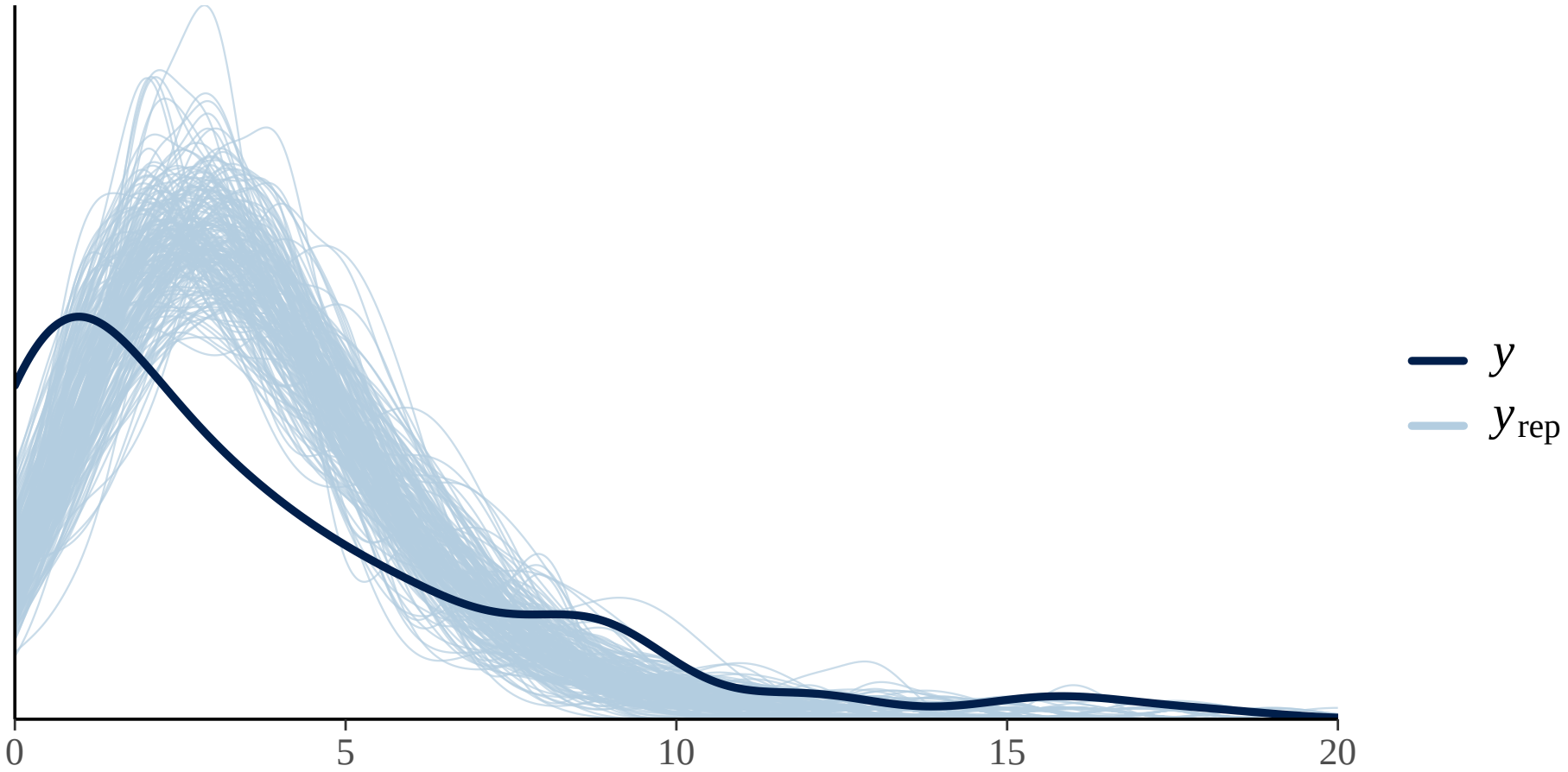
- There are a lot of graphical posterior predictive checking within the **bayesplot** package, all with the prefix `ppc_`. See http://mc-stan.org/bayesplot/articles/graphical-ppcs.html

- We start by comparing the distribution of y and the distributions of some of the simulated datasets (first 200 rows) in the `y_rep` matrix

```
y_rep <- as.matrix(fit_P1, pars = "y_rep")
ppc_dens_overlay(y = stan_dat$complaints, y_rep[1 : 200,])
```

Cockroaches' example: (simple) Poisson regression model

- The replicated datasets are not as dispersed as the observed data. They don't seem to capture the rate of zero in the observed data



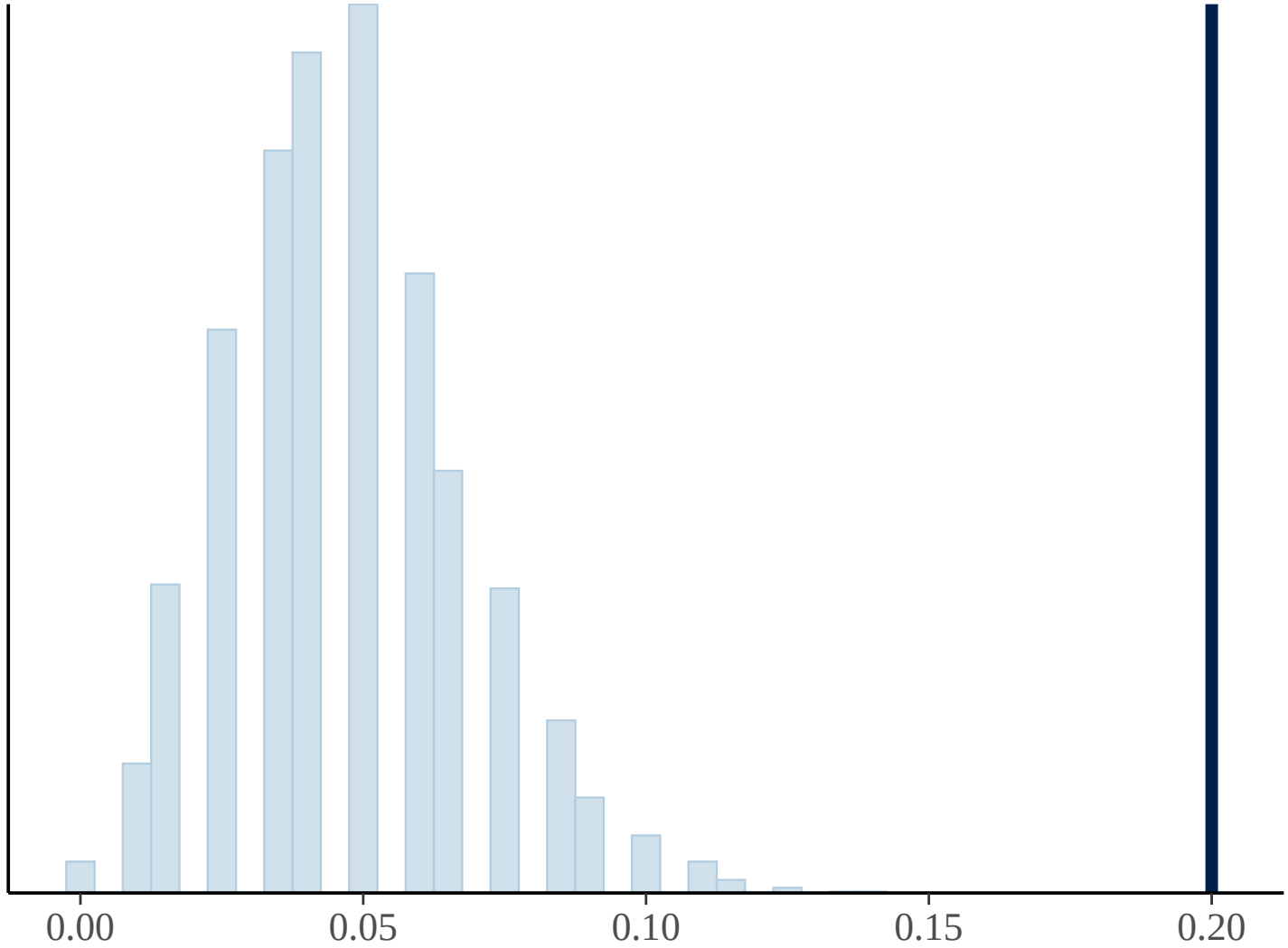
Cockroaches' example: (simple) Poisson regression model

- Plot of the observed proportion of zeros and histogram of the proportion of zeros in each of the simulated datasets

```
prop_zero <- function(x) mean(x == 0)
ppc_stat(pest_data$complaints, y_rep,
         stat = "prop_zero", binwidth = 0.005)
```

Cockroaches' example: (simple) Poisson regression model

- Clearly the model does not capture this feature of the data



T = prop_zero

$T(y_{\text{rep}})$

$T(y)$

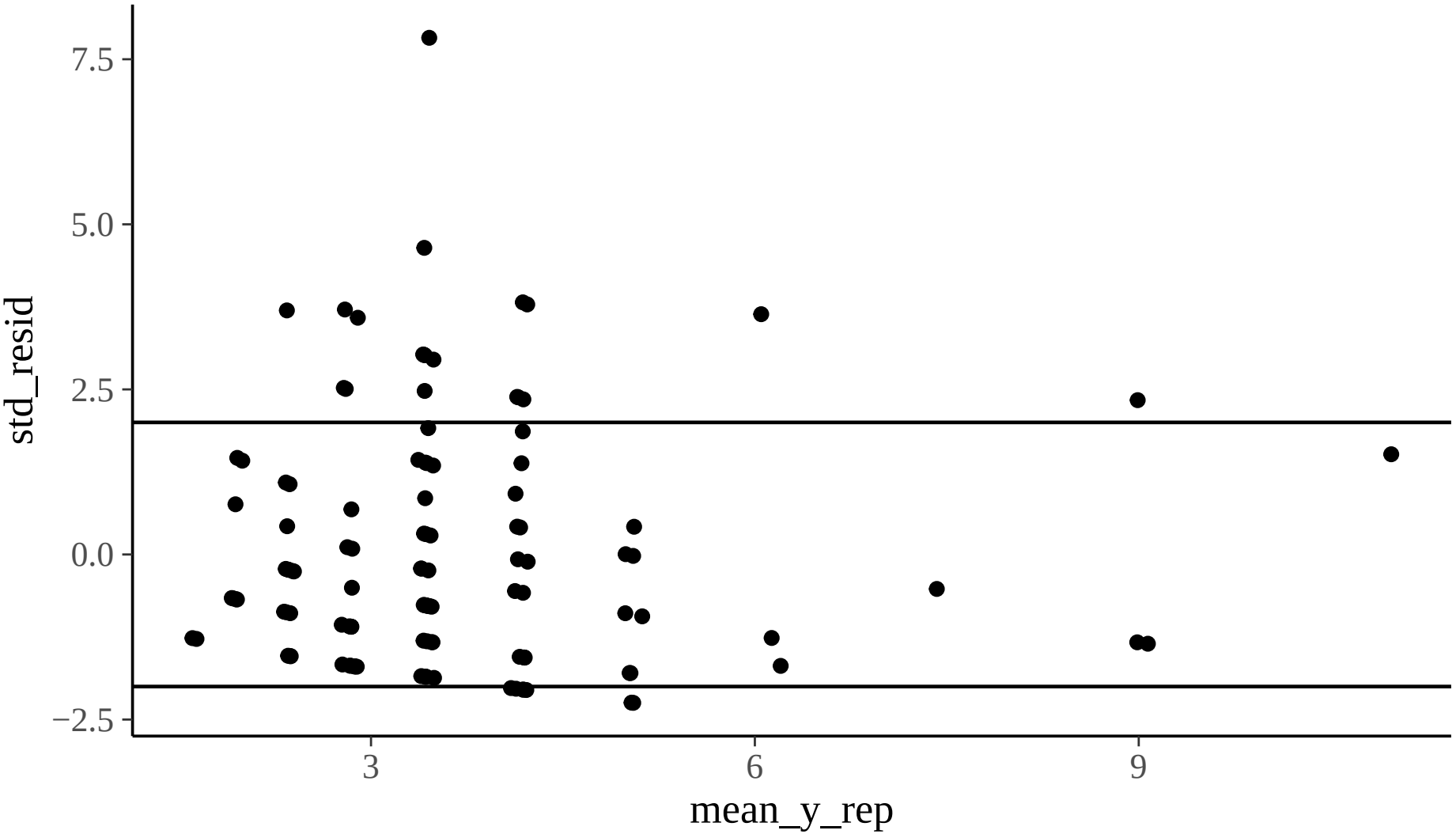
Cockroaches' example: (simple) Poisson regression model

- Plot of the standardised residuals of the observed vs predicted number of complaints

```
mean_y_rep <- colMeans(y_rep)
std_resid <- (stan_dat$complaints - mean_y_rep) /
             sqrt(mean_y_rep)
ggplot() +
  geom_point(mapping = aes(x = mean_y_rep, y = std_resid)) +
  geom_hline(yintercept = c(-2,2))
```

Cockroaches' example: (simple) Poisson regression model

- Residuals are mostly positive, thus the model tends to underestimate the number of complaints



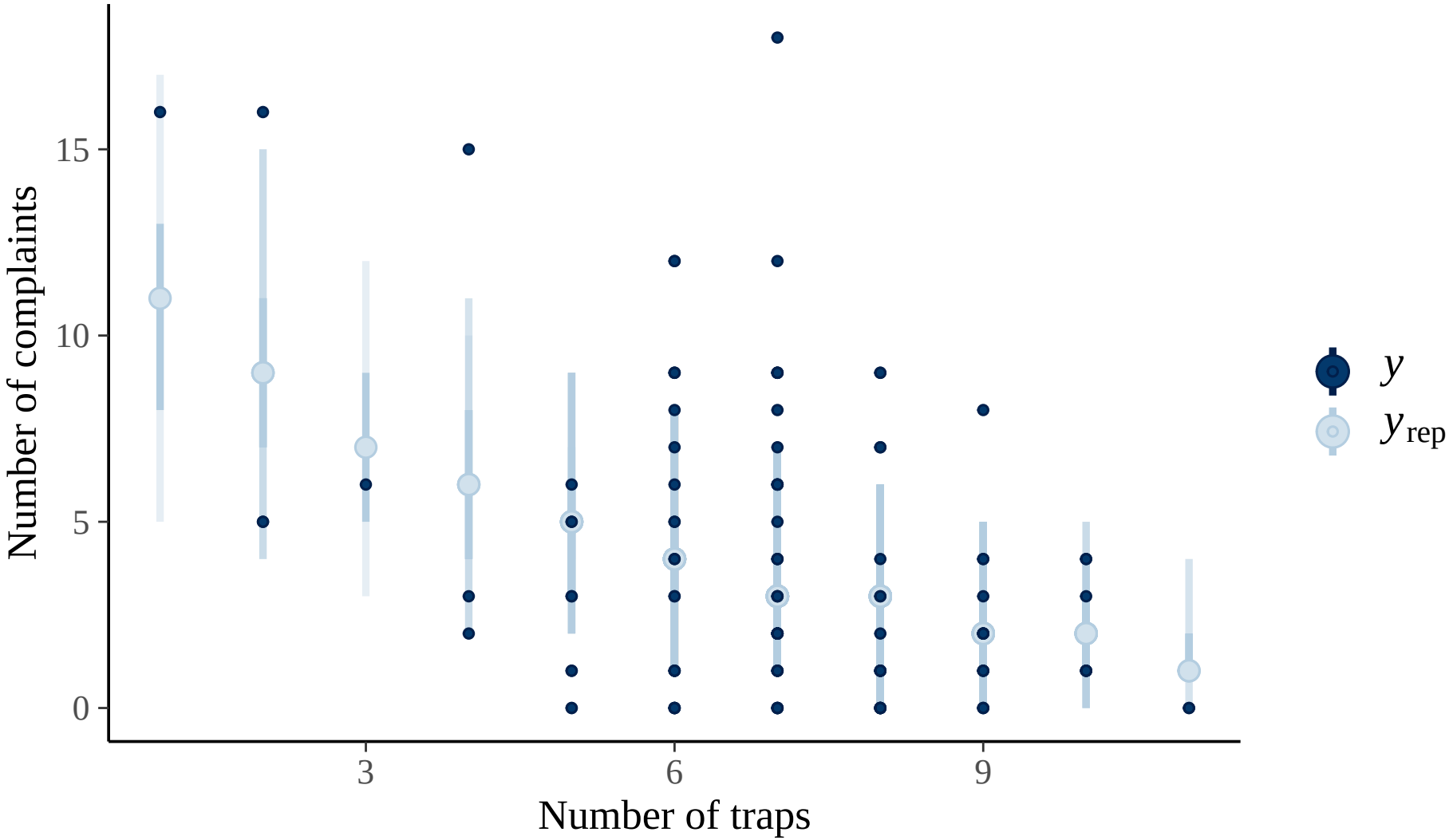
Cockroaches' example: (simple) Poisson regression model

- Plot of the uncertainty intervals for the predicted number of complaints for different numbers of bait stations

```
with(stan_dat, ppc_intervals(
  y = complaints,
  yrep = y_rep,
  x = traps
) + labs(x = "Number of traps",
         y = "Number of complaints"))
```

Cockroaches' example: (simple) Poisson regression model

- The model doesn't seem to fully capture the data, especially the tails of the observed data



Cockroaches' example: Improve the model

- The posterior predictive checks (PPCs) highlight some deficiency of our modelling choice. What to do if a PPC fails? There is not a unique answer. However, some tips may be the following ones:
  - **Extend the model: augment the predictors**, include eventual hierarchies
  - **Change the sampling distribution**
  - Change the priors
  - Transform your data, for instance using logarithmic scale
- Of course, modeling the relationship between complaints and bait stations by means of Poisson regression model is the simplest choice
- Thus, we can expand the model aiming to meet the requirements of our client

Cockroaches' example: Exploratory analysis

- Manager's intuition: they expect there are other reasons that one building might have more cockroaches complaints than another
- A possible candidate for expanding our model is the live-in-superintendent variable (`super`): the live-in-superintendent is responsible for keeping a building in good condition
- The following plot suggests that our guess is sensible, as it is apparent that the number of complaints vary as function of the number of traps and the presence or not of the live-in-superintendent in the building

```
ggplot(pest_data, aes(x = traps, y = complaints,
                      color = factor(live_in_super,
                                     label = c(FALSE, TRUE)))) +
  scale_color_discrete(name = "Live-in super") +
  geom_jitter() +
  theme(legend.position = "bottom")
```

Cockroaches' example: Exploratory analysis

Cockroaches' example: Improve the model

- Recall that our model's mean parameter is a rate of complaints per 1 month
- However, we are modelling a process that occurs over an area as well as over time
- Since the square footage of each building is in our availability, we can insert such information in the model
- The Poisson regression model allows the inclusion of an offset/exposure variable, allowing to interpret the model's parameter as a rate of events per unit exposure (in this case the rate of complaints per square foot per 1 month)

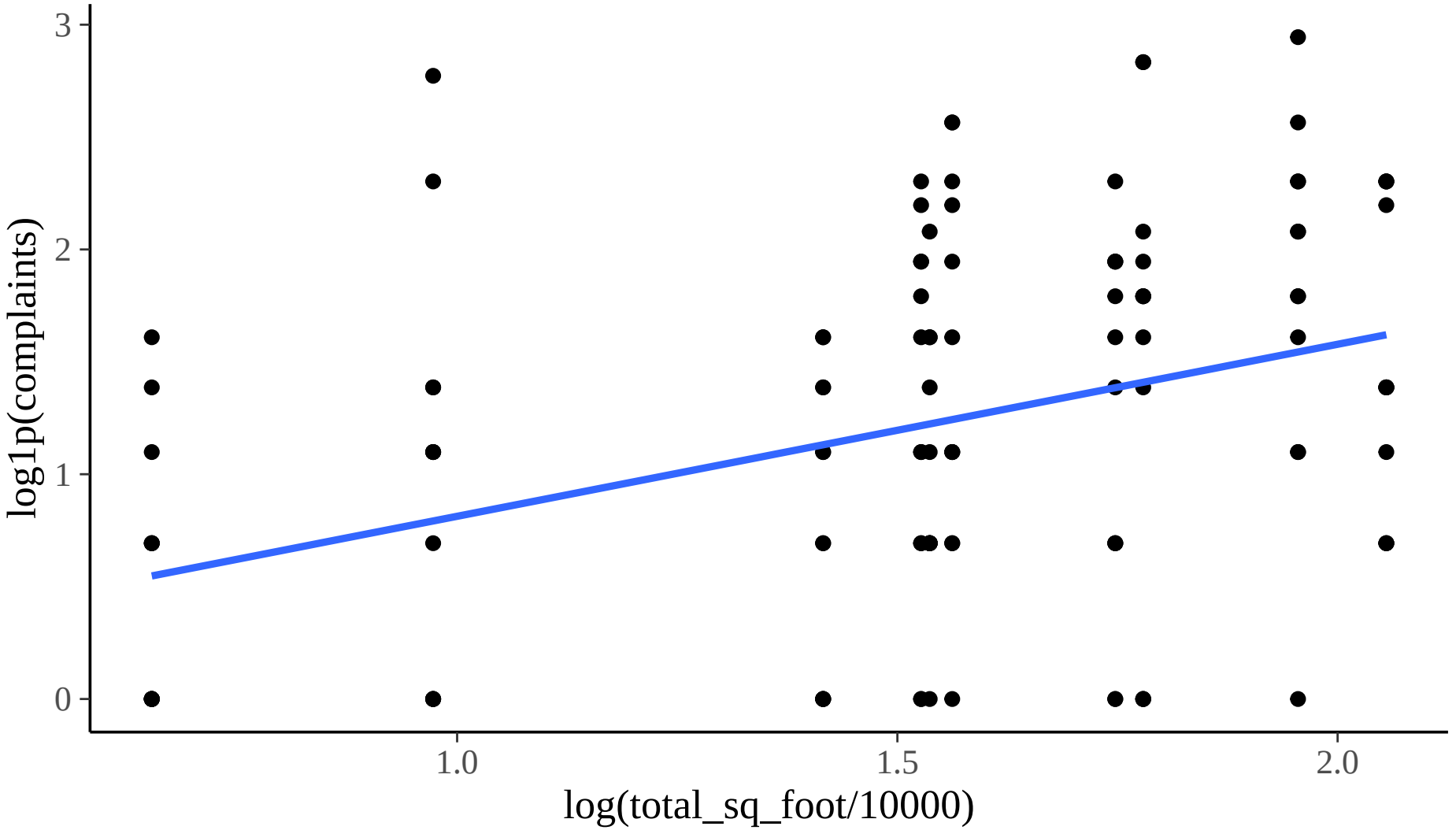
Cockroaches' example: Exploratory analysis

- A very simple check motivates the inclusion of such term in the model by noticing that there is a relationship between the square footage of the building and the number of complaints received

```
ggplot(pest_data, aes(x = log(total_sq_foot/1e4),
                      y = log1p(complaints))) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

- Note:
  - We scaled `total_sq_foot` by 10000 for simplicity
  - We are using `log1p(.x)` which returns the base-10 logarithm of the value `(1+.x)` (So `log1p(0)=0`)

Cockroaches' example: Exploratory analysis



Cockroaches' example: (multiple) Poisson regression model

- From the preliminary analysis above, it appears sensible adding to the simplest model the live-in superintendent (thereafter `super`) as explanatory variable and the total square footage of the building (divided by 10000 and thereafter `sqfoot`) as an offset

- The model we will implement

$$\begin{aligned} \text{complaints}_i &\sim \text{Poisson}(\text{sqfoot}_i \lambda_i), \quad i = 1, \ldots, 120 \\ \lambda_i &= \exp(\eta_i) \\ \eta_i &= \alpha + \beta_1 \, \text{traps}_i + \beta_2 \, \text{super}_i) \end{aligned}$$

Cockroaches' example: (multiple) Poisson regression model

- The latter model corresponds to

$$\begin{aligned} \text{complaints}_i &\sim \text{Poisson}(\lambda_i), \quad i = 1, \ldots, 120 \\ \lambda_i &= \exp(\eta_i) \\ \eta_i &= \alpha + \beta_1 \, \text{traps}_i + \beta_2 \, \text{super}_i + \log(\text{sqfoot}_i) \end{aligned}$$

- Or using the building and the time indices, equivalently

$$\begin{aligned} \text{complaints}_{b,t} &\sim \text{Poisson}(\lambda_{b,t}), \quad b = 1, \ldots, 10, \quad t = 1, \ldots, 12 \\ \lambda_{b,t} &= \exp(\eta_{b,t}) \\ \eta_{b,t} &= \alpha + \beta_1 \, \text{traps}_{b,t} + \beta_2 \, \text{super}_b + \log(\text{sqfoot}_b) \end{aligned}$$

Cockroaches' example: It's your turn

- Starting from `simple_Poisson_regression.stan` create a new stan file
  - Save it as `multiple_Poisson_regression.stan`
  - Insert in the **data block** `super` and `sqfoot`. For the latter:
    - If you consider `sqfoot` then you need consider the **transformed data** block for taking its logarithm by creating the variable `log_sqfoot`
    - Otherwise, fell free to specify directly the transformed variable in the **data block** as `log_sqfoot`
  - Modify suitably the **parameters block** considering now α, β_1, β_2
  - For the **model block**:
    - Consider the same priors for α and β_1, while consider $\beta_2 \sim \mathcal{N}(-0.5, 1)$
    - Modify suitably the likelihood to take into account the new linear predictor formulation, including also the offset variable
  - Modify suitably the generated quantities block

Cockroaches' example: (multiple) Poisson regression model

- Then, add to the list of data the new variables (according to the choice above you can include directly the logarithm of the offset variable or the original variable)

```
# To complete; the final structure you will obtain could be
str(stan_dat)
```

```
## List of 5
##  $ N          : int 120
##  $ complaints: num [1:120] 1 3 0 1 0 0 4 3 2 2 ...
##  $ traps      : num [1:120] 8 8 9 10 11 11 10 10 9 9 ...
##  $ super      : num [1:120] 0 0 0 0 0 0 0 0 0 0 ...
##  $ sqfoot     : num [1:120] 4.12 4.12 4.12 4.12 4.12 ...
```

Cockroaches' example: (multiple) Poisson regression model

- Then, you can compile the model

```
comp_model_P2 <- stan_model('multiple_poisson_regression.stan')
```

- Sample draws from the posterior distribution

```
fit_P2 <- sampling(comp_model_P2, data = stan_dat, refresh=0)
```

- See the posterior summary of the regression parameters

```
print(fit_P2, pars = c('alpha','beta1','beta2'))
```

Cockroaches' example: (multiple) Poisson regression model

```
## Inference for Stan model: multiple_poisson_regression.
## 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%
## alpha  1.19    0.01 0.21  0.78  1.05  1.19  1.33
## beta1 -0.21    0.00 0.03 -0.27 -0.23 -0.21 -0.19
## beta2 -0.28    0.00 0.12 -0.53 -0.36 -0.28 -0.19
##       97.5% n_eff Rhat
## alpha  1.62   726 1.01
## beta1 -0.16   755 1.00
## beta2 -0.05   978 1.01
##
## Samples were drawn using NUTS(diag_e) at Fri May 05 17:56:07 2023.
## 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).
```

Cockroaches' example: (multiple) Poisson regression model

- As above, you can visualise the Markov chain Monte Carlo (MCMC) draws from the posterior distribution of the parameters by suitably extracting such draws and using the graphical tools of the **bayesplot** package (`mcmc_trace`, `mcmc_acf`, `mcmc_hist`, `mcmc_scatter`)

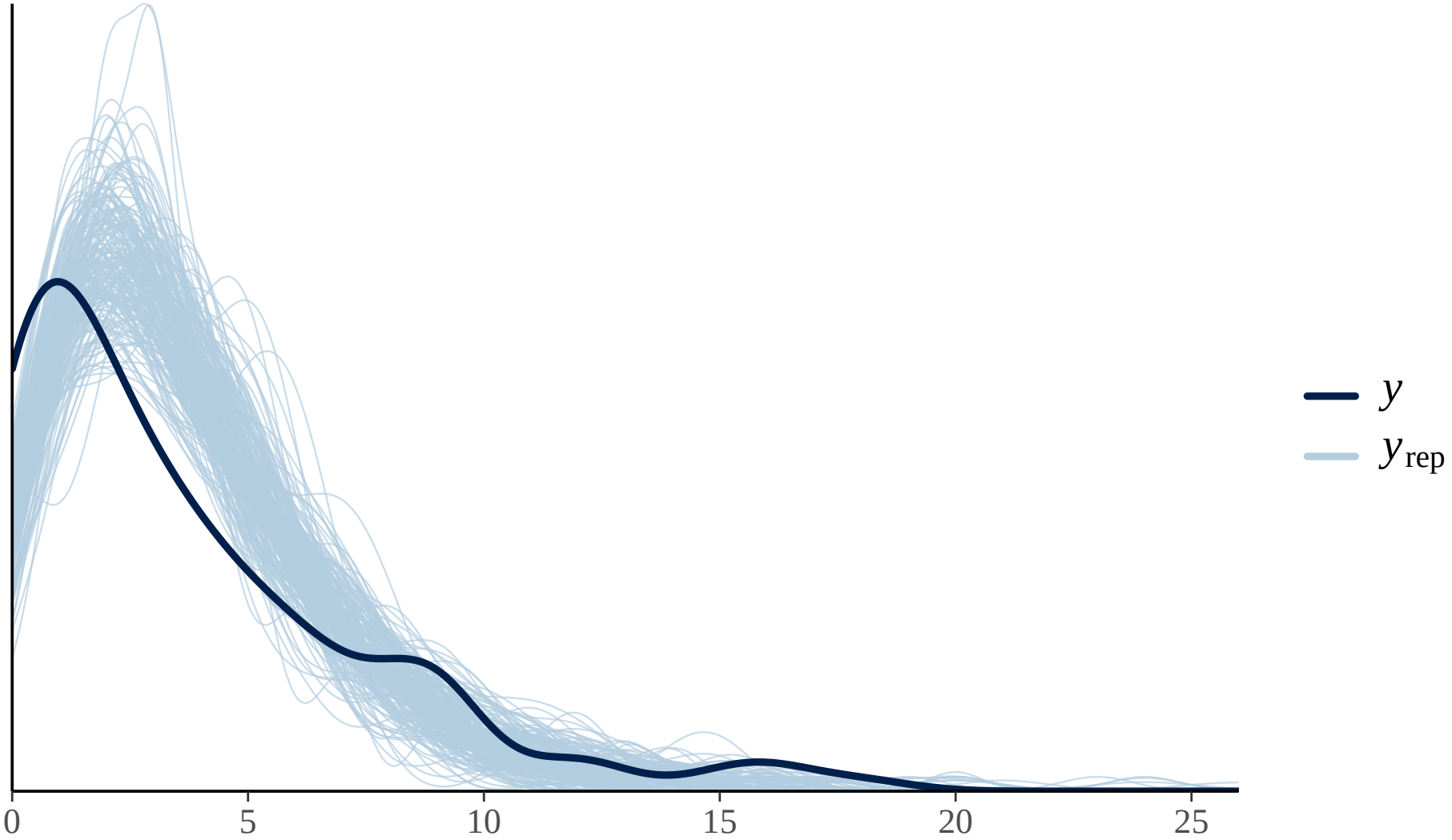
- However, our main interest is in evaluating if the inclusion of the new explanatory variable `super` as well as the offset are responsible for a better fit. Thus, we will explore the PPCs introduced above

- We start extracting the generated quantity and comparing the distributions of the posterior predictive simulations with the distribution of y

```
y_rep2 <- as.matrix(fit_P2, pars = "y_rep")
ppc_dens_overlay(stan_dat$complaints, y_rep2[1 : 200, ])
```

Cockroaches' example: (multiple) Poisson regression model

- It seems we are not able to capture the smaller counts well



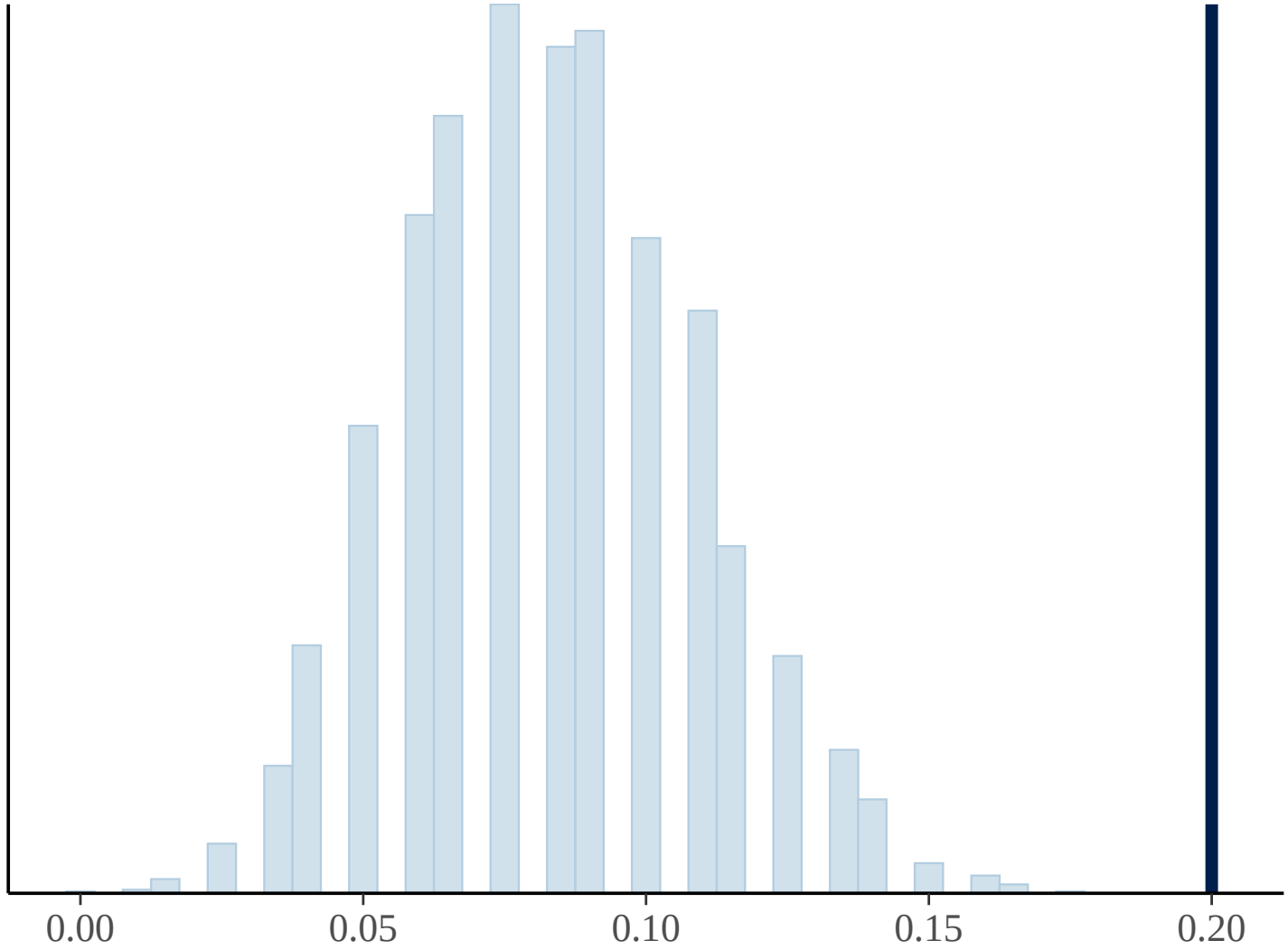
Cockroaches' example: (multiple) Poisson regression model

- Plot of the observed proportion of zeros and histogram of the proportion of zeros in each of the simulated datasets

```
ppc_stat(pest_data$complaints, y_rep2,
         stat = "prop_zero", binwidth = 0.005)
```

Cockroaches' example: (multiple) Poisson regression model

- Again we are not able to capture this feature of the data: we are still underestimating the proportion of zeroes in the data



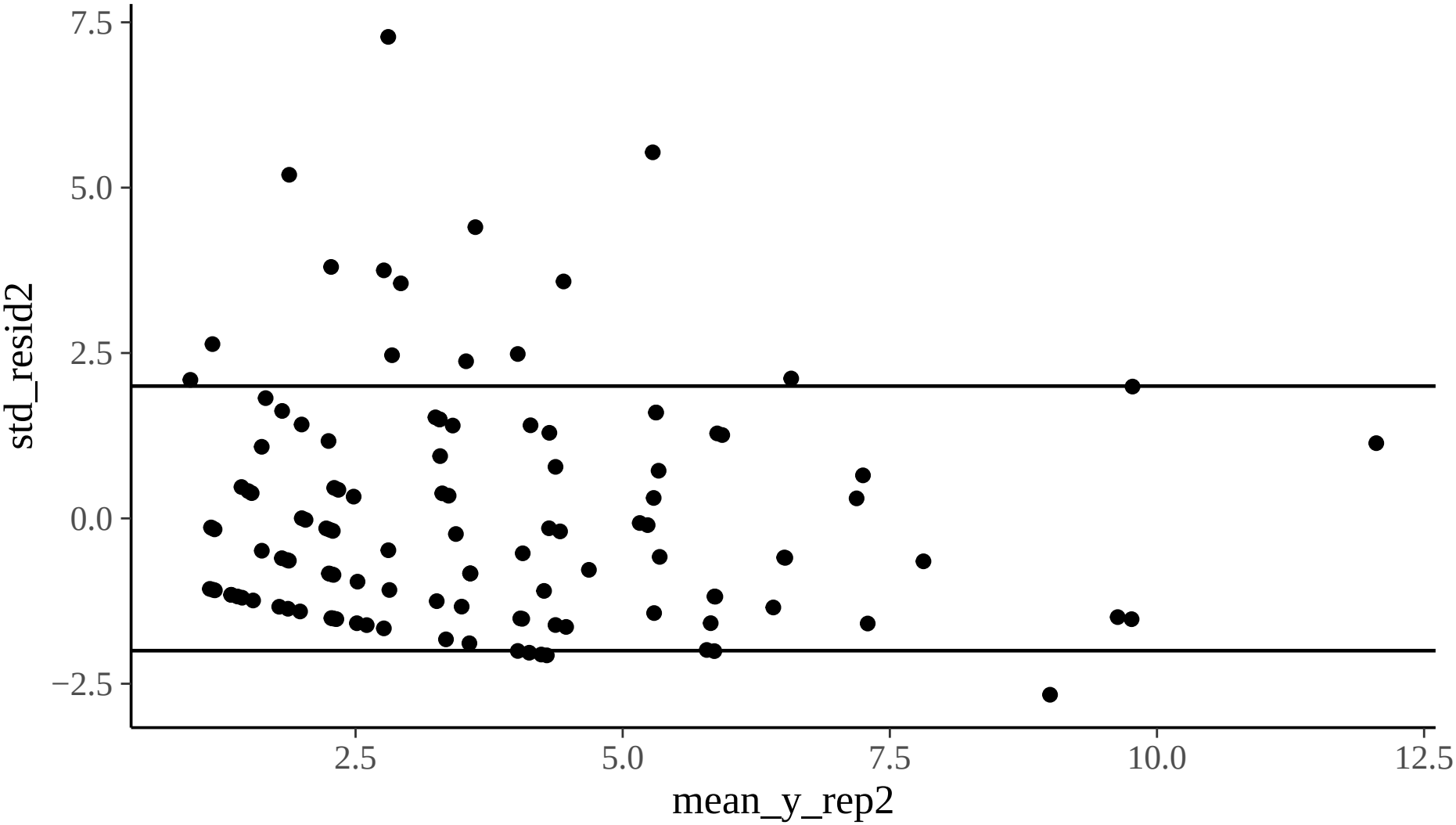
Cockroaches' example: (multiple) Poisson regression model

- Plot of the standardised residuals of the observed vs predicted number of complaints

```
mean_y_rep2 <- colMeans(y_rep2)
std_resid2 <- (stan_dat$complaints - mean_y_rep2) /
              sqrt(mean_y_rep2)
ggplot() +
  geom_point(mapping = aes(x = mean_y_rep2, y = std_resid2)) +
  geom_hline(yintercept = c(-2,2))
```

Cockroaches' example: (multiple) Poisson regression model

- Again the residuals are mostly positive (underestimate of the number of complaints)



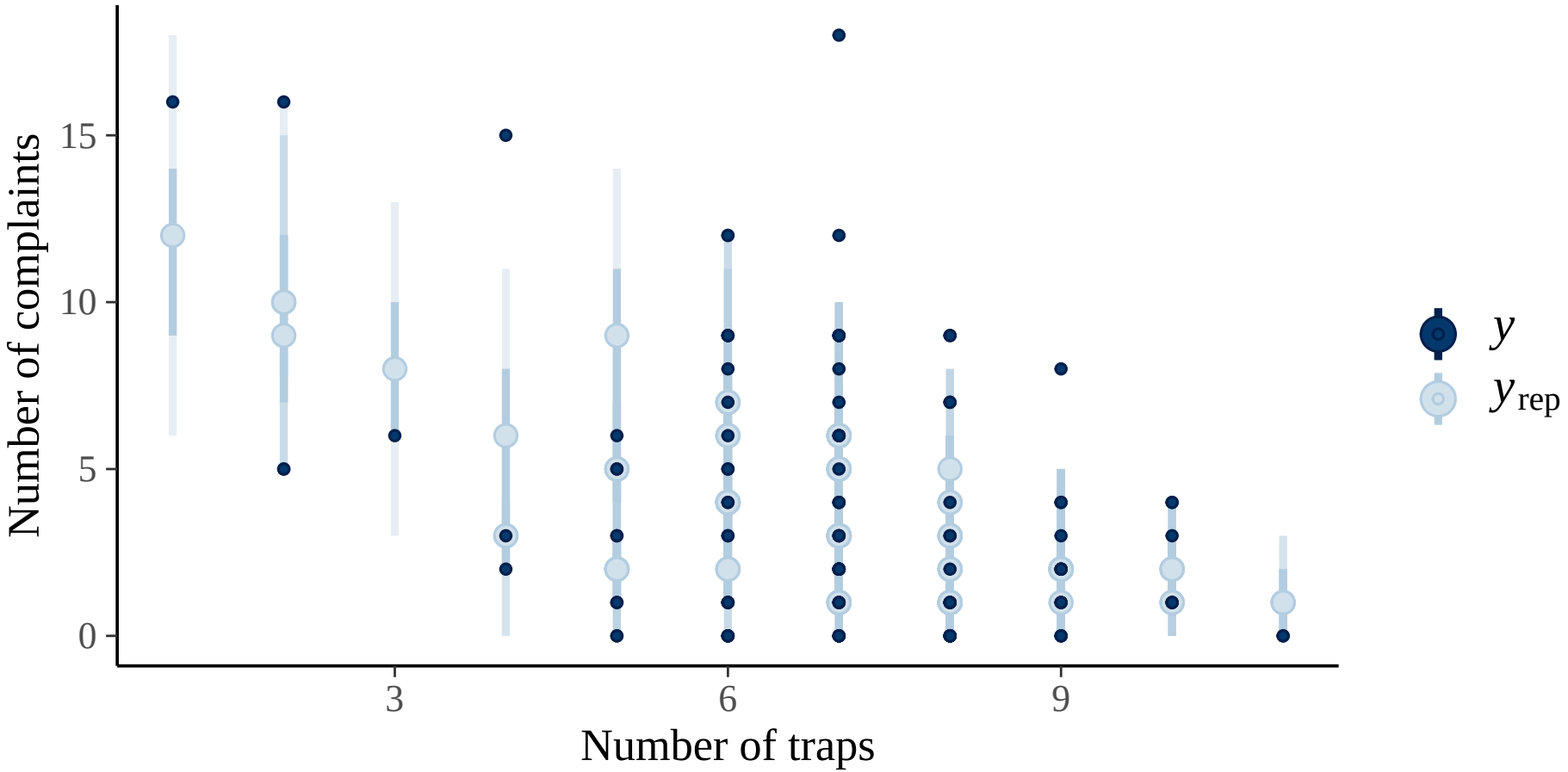
Cockroaches' example: (multiple) Poisson regression model

- Plot of the uncertainty intervals for the predicted number of complaints for different numbers of bait stations

```
with(stan_dat, ppc_intervals(
  y = complaints,
  yrep = y_rep2,
  x = traps
) +
  labs(x = "Number of traps", y = "Number of complaints"))
```

Cockroaches' example: (multiple) Poisson regression model

- We've increased the tails a bit more at the larger numbers of traps but we still have some large observed numbers of complaints that the model would consider extremely unlikely events



Cockroaches' example: Improve the model

- However, some questions arise naturally:
  - Is the Poisson distribution assumption sensible?
  - Is there overdispersion?
- We could consider a different probability distribution for our modelling task: any thoughts?

Cockroaches' example: Improve the model

- We saw from the PPCs that the model:
  - Doesn't fit the data as we would like
  - Underpredict low and high number of complaints
  - Overpredict the medium number of complaints
- This is an indication of overdispersion, where the variance is larger than the mean, and this can be due to the omission of relevant explanatory variables
- The Poisson model doesn't fit overdispersed count data well because the same parameter control both the expected counts and the variance of these counts
- The natural alternative to the Poisson model assumption for counts data is the Negative Binomial model

Section 3

Cockroaches' example: Negative Binomial regression model

Cockroaches' example: Negative Binomial regression model

- A very quick introduction to the Negative Binomial distribution
- It is a discrete probability distribution modelling the number of failures until the ϕ-th success ($\phi > 0$) in a sequence of Bernoulli trials with probability of success $p \in (0, 1)$. So $Y \sim \text{Neg} - \text{Binomial}(\phi, p)$ has pmf

$$P(Y = y) = \binom{y + \phi - 1}{y} p^{\phi}(1 - p)^{y}, \quad y = 0, 1, 2, \ldots$$

$$\mathbb{E}(Y) = \phi(1 - p)/p \qquad Var(Y) = \phi(1 - p)/p^2$$

Cockroaches' example: Negative Binomial regression model

- From the latter the alternative parametrisation in terms of mean ($\lambda = \mathbb{E}[Y]$) can be derived, which is particularly useful in Negative - Binomial regression, after extending the possibility that ϕ takes positive real values. Since $p = \frac{\phi}{\phi+\lambda}$ we have

$$Var(Y) = \frac{\lambda}{p} = \frac{\lambda(\phi+\lambda)}{\phi} = \lambda + \frac{\lambda^2}{\phi} > \lambda$$

$$P(Y = y) = \frac{\Gamma(y+\phi)}{y!\Gamma(\phi)}\left(\frac{\phi}{\phi+\lambda}\right)^{\phi}\left(\frac{\lambda}{\phi+\lambda}\right)^{y} \quad y = 0, 1, 2, \ldots$$

- ϕ is called precision parameters (sometimes $1/\phi$ is called similarly)

Cockroaches' example: Negative Binomial regression model

- See the Stan documentation: https://mc-stan.org/docs/2_20/functions-reference/neg-binom-2-log.html
- The pmf of the Negative Binomial we will use is `neg_binomial_2_log` (reals η, reals ϕ)
- It is parametrised in terms of its log-mean, η, and the precision, ϕ, such that

$$\mathbb{E}[Y] = \lambda = \exp(\eta) \quad \text{Var}[Y] = \lambda + \lambda^2/\phi = \exp(\eta) + \exp(\eta)^2/\phi.$$

- As ϕ gets larger, the term λ^2/ϕ approaches zero and so the variance of Negative Binomial approaches λ, so the Negative Binomial gets closer and closer to the Poisson

Cockroaches' example: Negative Binomial regression model

- Our model will be

$$\begin{aligned}
\text{complaints}_i &\sim \text{Neg-Binomial}(\lambda_i, \phi), \quad i = 1, \ldots, 120 \\
\lambda_i &= \exp(\eta_i) \\
\eta_i &= \alpha + \beta_1 \, \text{traps}_i + \beta_2 \, \text{super}_i + \log(\text{sqfoot}_i)
\end{aligned}$$

- Or equivalently,

$$\begin{aligned}
\text{complaints}_{b,t} &\sim \text{Neg-Binomial}(\lambda_{b,t}, \phi) \quad b = 1, \ldots, 10, \quad t = 1, \ldots, 12 \\
\lambda_{b,t} &= \exp(\eta_{b,t}) \\
\eta_{b,t} &= \alpha + \beta_1 \, \text{traps}_{b,t} + \beta_2 \, \text{super}_b + \log(\text{sqfoot}_b)
\end{aligned}$$

Cockroaches' example: It's your turn

- Open the file `multiple_NB_regression_void.stan` (containing only **functions block**) and leveraging `multiple_poisson_regression.stan`, implement:
  - The **data block** (no changes)
  - Modify suitably the **parameters block** by adding ϕ^{-1} as parameter and get ϕ in the **transformed parameters block**
  - For the **model block**:
    - Consider the same priors for α, β_1 and β_2 while consider $\phi^{-1} \sim \mathcal{TN}(0, 1, 0, +\infty)$ (also called half - normal distribution)
    - Modify suitably the likelihood accounting the new distribution
  - Modify suitably the generated quantities block accounting for the user-defined function
- Save the file as `multiple_NB_regression.stan`

Cockroaches' example: It's your turn

- In the **functions block** we use a customised function for generating from the Negative Binomial distribution, which leverages on the fact that the Negative - Binomial distribution arise from the Poisson-Gamma mixture model, that is

$$y|\lambda \sim Poisson(\lambda)$$

$$\lambda \sim \text{Gamma}(\alpha = \phi, \beta = \text{p}/1 - \text{p})$$

$$y \sim \text{Neg} - \text{Binomial}(\phi, p)$$

```
functions {
  int neg_binomial_2_log_safe_rng(real eta, real phi) {
    real gamma_rate = gamma_rng(phi, phi / exp(eta));
    if (gamma_rate >= exp(20.79))
      return -9;
    return poisson_rng(gamma_rate);
  }
}
```

Cockroaches' example: Negative Binomial regression model

- Compile the model

```
comp_model_NB <- stan_model('multiple_NB_regression.stan')
```

- Sampling

```
fit_NB <- sampling(comp_model_NB, data = stan_dat, refresh=0)
```

- Print a posterior summary of the regression parameters

```
print(fit_NB, pars = c('alpha','beta1','beta2'))
```

Cockroaches' example: Negative Binomial regression model

```
## Inference for Stan model: multiple_NB_regression.
## 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%
## alpha  1.31    0.01 0.43  0.48  1.02  1.30  1.59
## beta1 -0.22    0.00 0.06 -0.34 -0.26 -0.22 -0.19
## beta2 -0.29    0.01 0.22 -0.73 -0.44 -0.28 -0.14
##       97.5% n_eff Rhat
## alpha  2.16  1207    1
## beta1 -0.11  1259    1
## beta2  0.15  1746    1
##
## Samples were drawn using NUTS(diag_e) at Fri May 05 17:56:57 2023.
## 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).
```

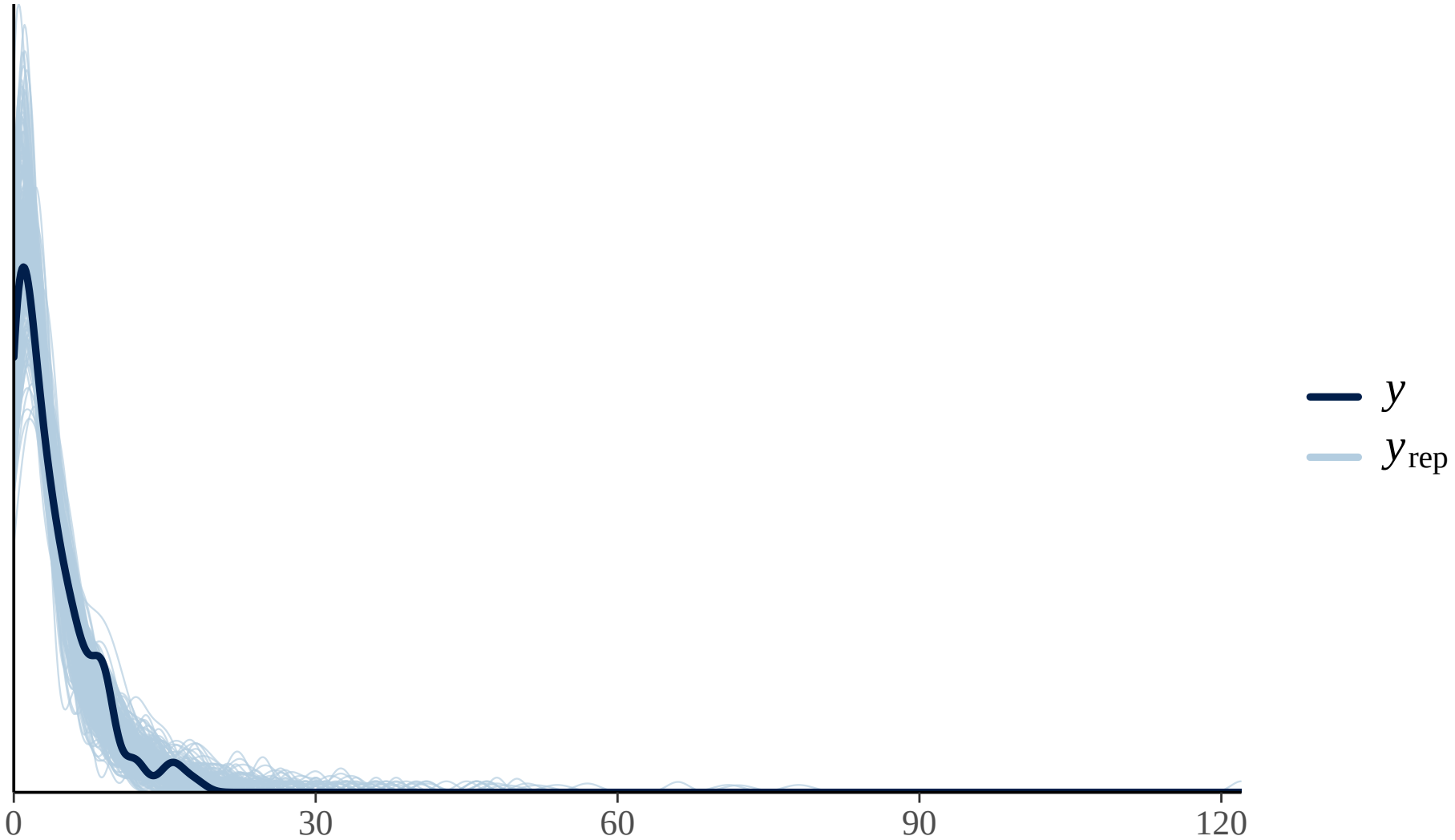
Cockroaches' example: Negative Binomial regression model

- As above, we will evaluate our model fit by means of PPCs
- Thus, after extracting the generated quantity we can compare the distibution of the data with the distribution of the posterior predictive simulations

```
y_rep3 <- as.matrix(fit_NB, pars = "y_rep")
ppc_dens_overlay(stan_dat$complaints, y_rep3[1 : 200,])
```

Cockroaches' example: Negative Binomial regression model

- It appears that our model now captures both the number of small counts better as well as the tails



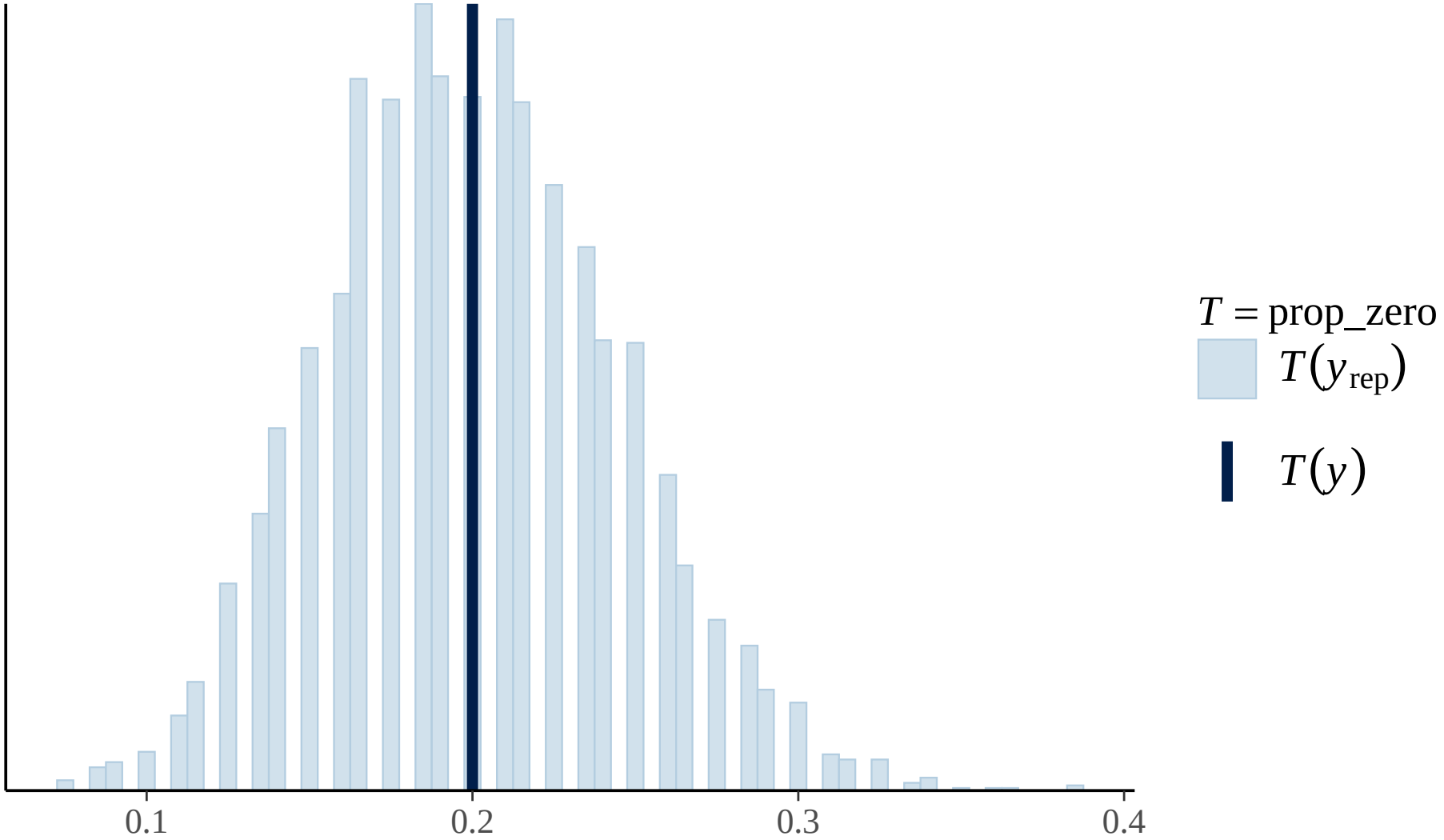
Cockroaches' example: Negative Binomial regression model

- Plot of the observed proportion of zeros and histogram of the proportion of zeros in each of the simulated datasets

```
ppc_stat(stan_dat$complaints, y_rep3,
         stat = "prop_zero", binwidth = 0.005)
```

Cockroaches' example: Negative Binomial regression model

- It appears that the Negative Binomial model is responsible for a better job capturing the number of zeroes



Cockroaches' example: Negative Binomial regression model

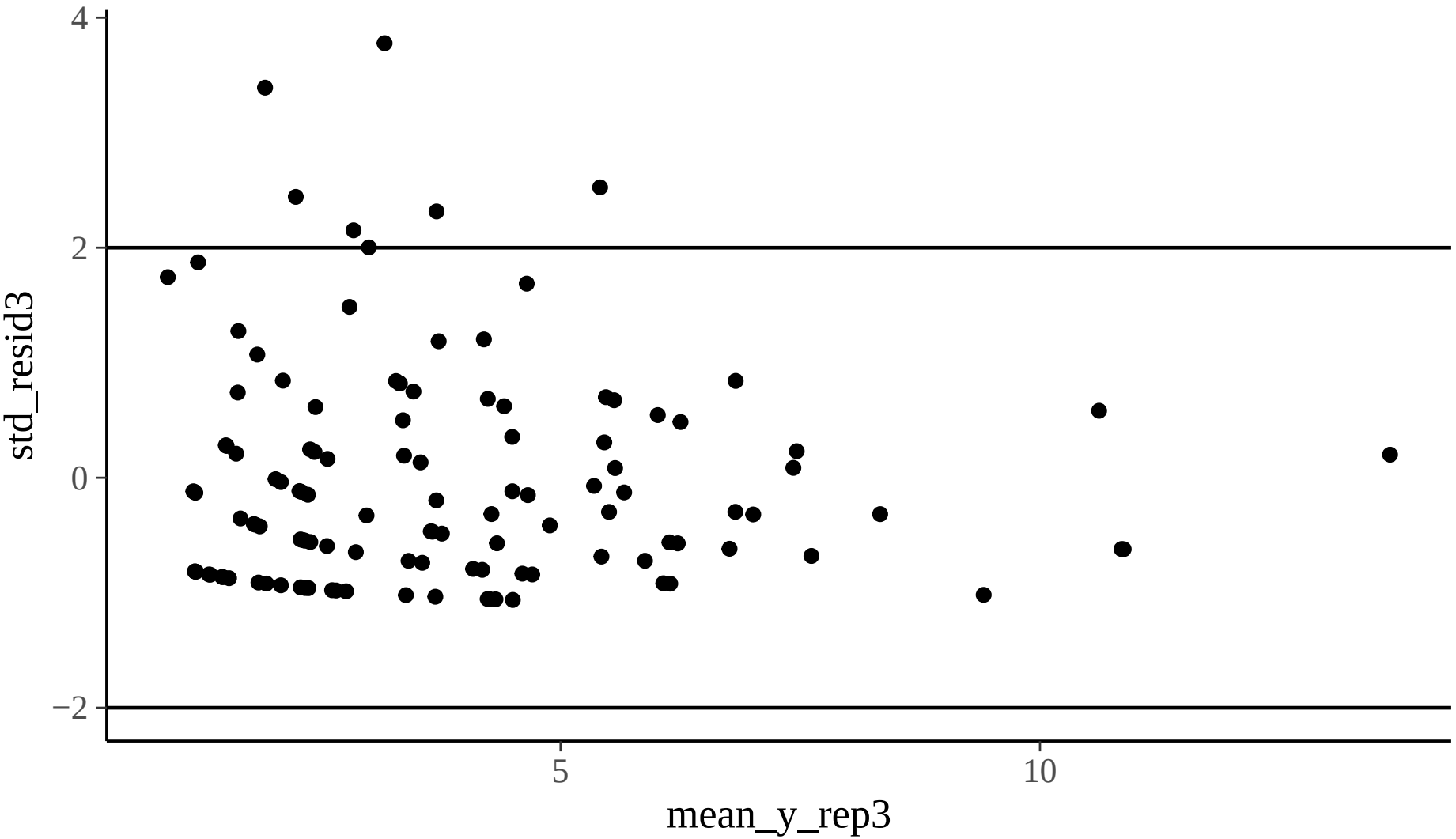
- Plot of the standardised residuals of the observed vs predicted number of complaints

```
inv_phi <- as.matrix(fit_NB, pars = "inv_phi")
mean_inv_phi <- mean(inv_phi)
mean_y_rep3 <- colMeans(y_rep3)
std_resid3 <- (stan_dat$complaints - mean_y_rep3) /
  sqrt(mean_y_rep3 + mean_y_rep3 ^ 2 * mean_inv_phi)
ggplot() +
  geom_point(mapping = aes(x = mean_y_rep3, y = std_resid3)) +
  geom_hline(yintercept = c(-2,2))
```

- Note the change w.r.t. the Poisson-type of standarzized residuals

Cockroaches' example: Negative Binomial regression model

- It appears better, but we still have some large standardised residuals



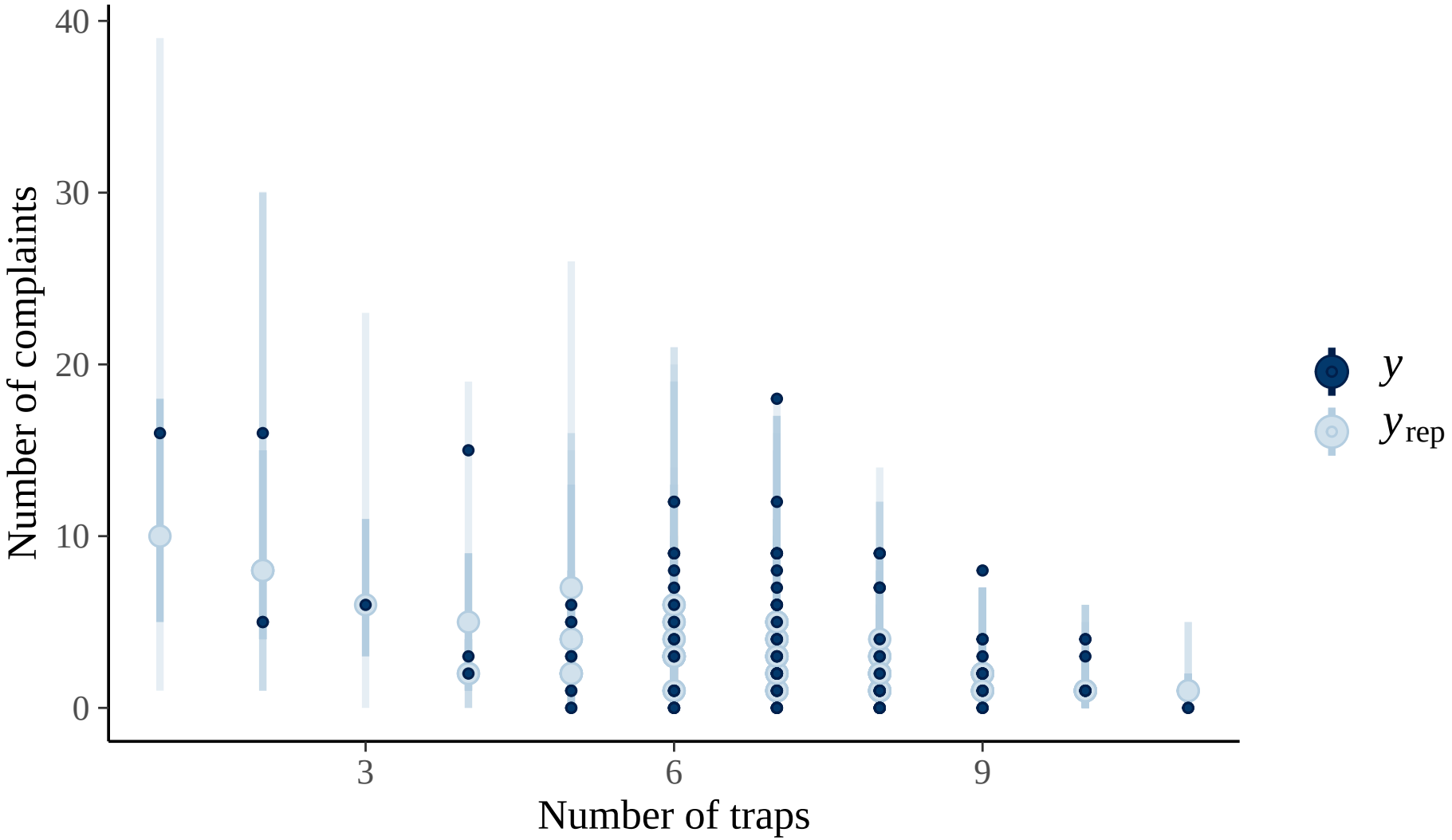
Cockroaches' example: Negative Binomial regression model

- Plot uncertainty intervals for the predicted number of complaints for different numbers of bait stations

```
with(stan_dat, ppc_intervals(
  y = complaints,
  yrep = y_rep3,
  x = traps
) +
  labs(x = "Number of traps", y = "Number of complaints"))
```

Cockroaches' example: Negative Binomial regression model

- It appears that the model achieves a large improvements in capturing the data, especially the tails



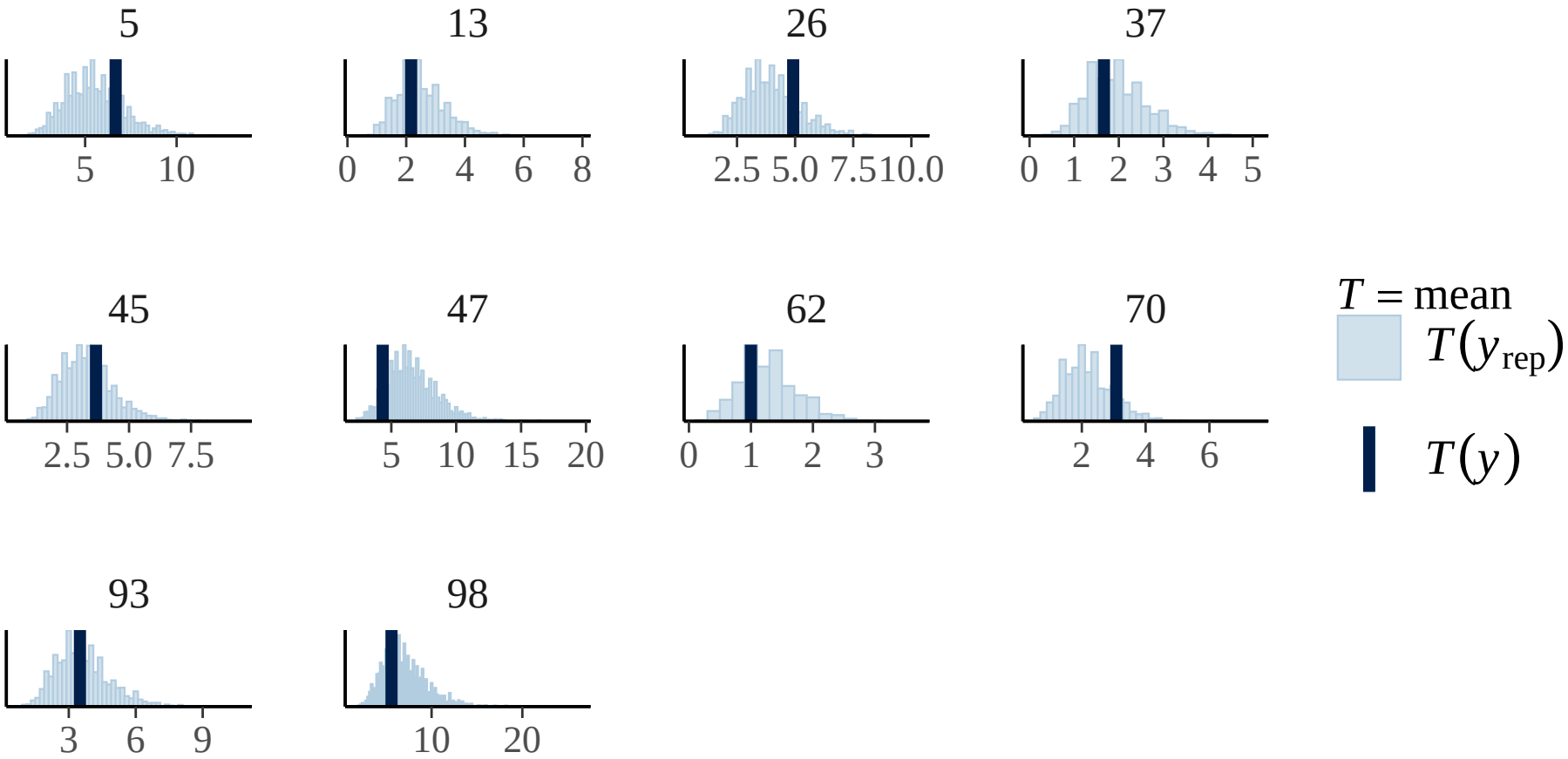
Cockroaches' example: Negative Binomial regression model

- However, data are clustered by building and currently we are missing such an information
- A posterior predictive check can help us understanding if it would be a good idea to add the building information into the model

```
with(pest_data, ppc_stat_grouped(
  y = complaints,
  yrep = y_rep3,
  group = building_id,
  stat = 'mean',
  binwidth = 0.2
))
```

Cockroaches' example: Negative Binomial regression model

- We're getting plausible predictions for most building means
- However, some are estimated better than other and some have larger uncertainties than we might expect

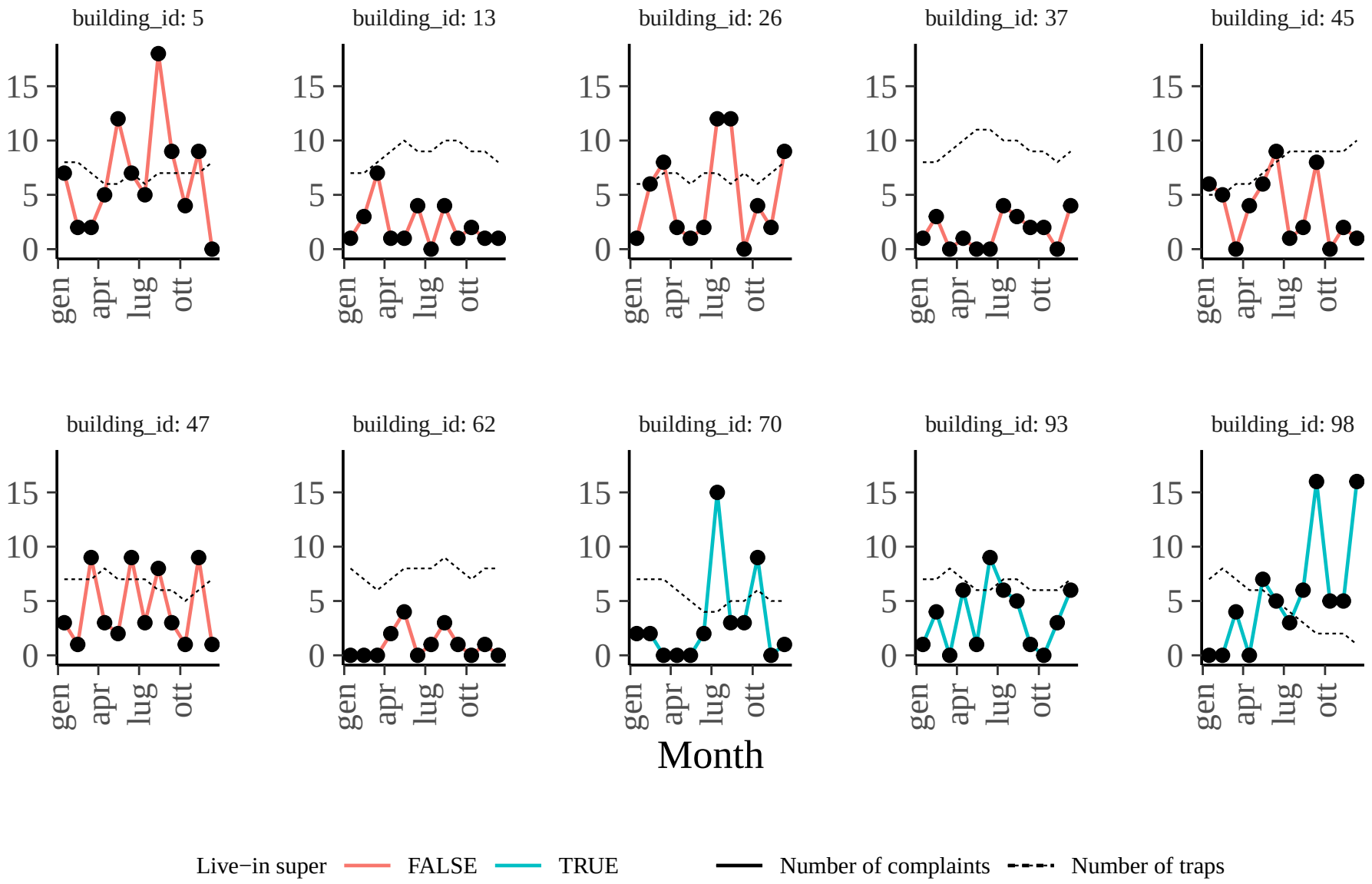


Cockroaches' example: Exploratory analysis

- Time series plot of the traps and the complaints for each building

```
ggplot(pest_data, aes(x = date, y = complaints,
                      color = live_in_super == TRUE)) +
  geom_line(aes(linetype = "Number of complaints")) +
  geom_point(color = "black") +
  facet_wrap(~ building_id, scales = "free",
             ncol = 5, labeller = label_both) +
  geom_line(aes(y = traps, linetype = "Number of traps"),
            color = "black", size = 0.25) +
  scale_x_date(name = "Month", date_labels = "%b") +
  scale_y_continuous(name = "",
                     limits = range(pest_data$complaints)) +
  scale_linetype_discrete(name = "") +
  scale_color_discrete(name = "Live-in super") +
  theme(legend.position="bottom", legend.box = "horizontal",
        legend.title = element_text(size = 7),
        legend.text = element_text(size = 7),
        strip.text.x = element_text(size = 7),
        axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```

Cockroaches' example: Exploratory analysis



Cockroaches' example: Next Lab

- We will take into account the variation across buildings by means of hierarchical modelling. We will start implementing a varying intercept Negative Binomial hierarchical model

$$
\begin{aligned}
\textit{complaints}_{b,t} &\sim \text{Neg} - \text{Binomial}(\lambda_{b,t}, \phi) \quad b = 1, \ldots, 10, \quad t = 1, \ldots, 12 \\
\lambda_{b,t} &= \exp\left(\eta_{b,t}\right) \\
\eta_{b,t} &= \mu_b + \beta_1 \, \textit{traps}_{b,t} + \log(\textit{sqfoot}_b) \\
\mu_b &\sim \mathcal{N}(\alpha + \beta_2 \, \text{super}_b + \beta_3 \, \text{age}_b + \beta_4 \, \text{ata}_b + \beta_5 \, \text{mar}_b, \sigma_\mu)
\end{aligned}
$$

- age for `age_of_building`, ata for `average_tenant_age` and `mar` is `monthly_average_rent`

- Then, we will also consider some varying intercept and slope models

- Final step: compare all the models we built this and the next lab