# Bayesian Statistics: Laboratory 5


Vincenzo Gioia

DEAMS

University of Trieste

vincenzo.gioia@units.it

Building D, room 2.13

Office hour: Friday, 15 - 17


16/05/2023

# Outline

1. Cockroaches' example: Recap from the previous Lab

2. Varying intercept Negative Binomial hierarchical model

3. Negative Binomial hierarchical model: Varying intercept and slope

Section 1

# Cockroaches' example: Recap from the previous Lab

# Recap from the previous Lab

- A property manager ask us to explore the relationship between cockroaches complaints and bait stations to shed light on the effectiveness of the solution to install long term bait stations throughout the apartment buildings

- We are analysing data about complaints for the presence of cockroaches in 10 residential buildings

- As a first attempt in a Bayesian data analysis workflow, we modelled the number of complaints as a function of the number of traps by using a (simple) Poisson regression model

- Diagnostic analysis of the model highlights some residual variability

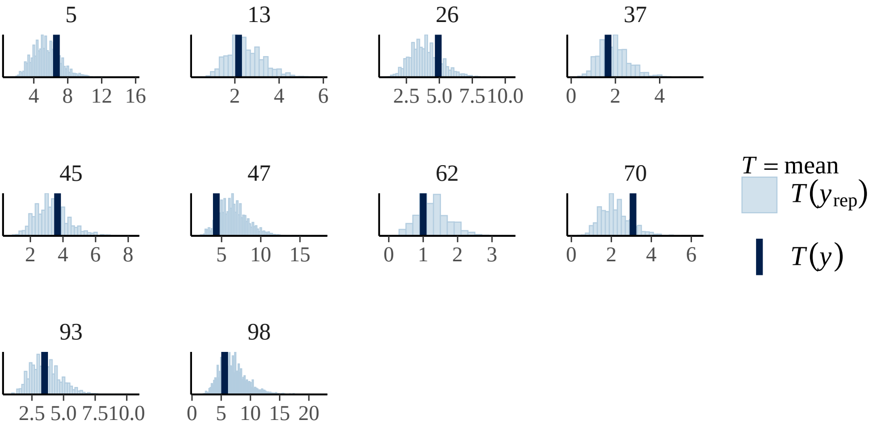
# Recap from the previous Lab

- We added one predictor and an exposure term to the Poisson model, but the situation did not improve. The Poisson distribution assumption on the model variance turned out to be too restrictive to model our data

- Then, we considered the Negative Binomial model. It gave better results, but examining the standardised residual plot we noticed that some of them were still very large

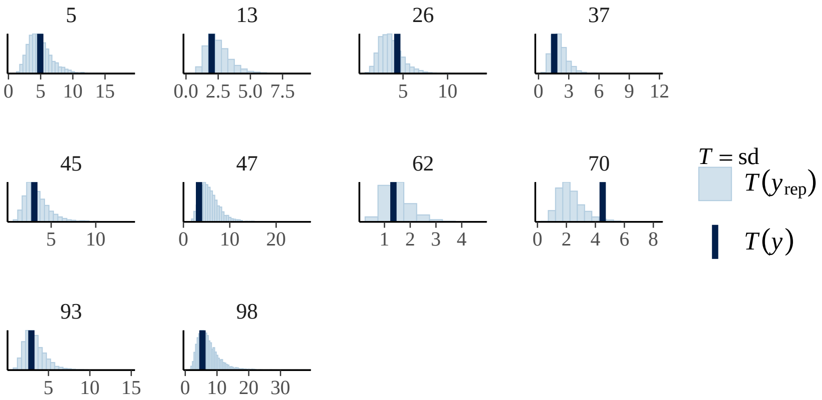
- A posterior predictive check considering that the data are clustered by building show that we are getting plausible predictions for most building means but some are estimated better than others and some have larger uncertainties than we might expect

# Recap from the previous Lab



# Recap from the previous Lab

- Considering as statistic the standard deviation instead of the mean, it seems we are not doing bad to estimate the within-building variability



Section 2

# Varying intercept Negative Binomial hierarchical model

# Varying intercept Negative Binomial hierarchical model

- If we explicitly model the variation across buildings we may be able to get much better estimates. Then, we consider for modelling the variation across buildings a varying intercept Negative Binomial hierarchical model

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

- Recall that `super` is `live_in_super`, `age` is `age_of_building`, `ata` is `average_tenant_age` and `mar` is `monthly_average_rent`

# Varying intercept Negative Binomial hierarchical model

- Equivalently, we will write our model in Stan as

$$
\begin{aligned}
complaints_{b,t} &\sim \mathrm{Neg-Binomial}(\lambda_{b,t}, \phi) \\
\lambda_{b,t} &= \exp\left(\eta_{b,t}\right) \\
\eta_{b,t} &= \mu_b + \beta \, traps_{b,t} + \log(sqfoot_b) \\
\beta &\sim \mathcal{N}(-0.25, 1) \\
\mu_b &\sim \mathcal{N}(\alpha + \text{building\_data}_b \zeta, \sigma_\mu) \\
\alpha &\sim \mathcal{N}(\log(4), 1) \\
\zeta_k &\sim \mathcal{N}(0, 1), \quad k = 1, \ldots, 4 \\
\sigma_\mu &\sim \mathcal{TN}(0, 1, 0, +\infty) \\
\phi^{-1} &\sim \mathcal{TN}(0, 1, 0, +\infty)
\end{aligned}
$$

- Here, $\mu_b$ is the $b$-th element of the vector $\mu$ which has one element per building

# Load packages and data

```
library(rstan)
library(bayesplot)
theme_set(bayesplot::theme_default())

set.seed(123)

data <- readRDS('pest_data.RDS')

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]
```

# Prepare the dataset

- We arrange the variables containing information at the building level (super, age, ata, and mat) into a matrix called building_data, such that the number of rows corresponds to the number of buildings

```
idx_row1 <- !duplicated(pest_data$building_id)
auxD <- pest_data[idx_row1, ]
bld <- matrix(NA, nrow = 10, ncol = 4)
bld[, 1] <- scale(auxD[, "live_in_super"], scale = F)
bld[, 2] <- scale(auxD[, "age_of_building"], scale = F) / 10
bld[, 3] <- scale(auxD[, "average_tenant_age"],
                  scale = F) / 10
bld[, 4] <- scale(auxD[, "monthly_average_rent"],
                  scale = F) / 1000
colnames(bld) <- c("super", "age", "ata", "mat")
```

# Prepare the dataset

```
bld
```

```
##       super   age         ata         mat
##  [1,]  -0.3 -0.24  0.39534708  0.15969341
##  [2,]  -0.3 -0.64 -0.03573347 -0.12373990
##  [3,]  -0.3 -0.04 -0.20656914 -0.65858726
##  [4,]  -0.3  0.16 -0.25872497  0.33198120
##  [5,]   0.7 -0.04 -0.47869016 -0.01064589
##  [6,]  -0.3 -0.04 -0.13793737 -0.30945229
##  [7,]  -0.3  0.96 -0.87817272  0.18343880
##  [8,]   0.7 -0.14  1.52587390  0.17640191
##  [9,]  -0.3  1.06  0.85317813  0.15298211
## [10,]   0.7 -1.04 -0.77857128  0.09792791
```

# Prepare Stan data list

```r
stan_dat_hier <- with(
  pest_data,
  list(N = length(traps),
       complaints = complaints,
       traps = traps,
       sqfoot = total_sq_foot/1e4,
       building_idx = as.integer(factor(
         building_id, levels = unique(building_id))),
       B = length(unique(building_id)),
       building_data = bld,
       K = ncol(bld)))
```

# Prepare Stan data list

```
str(stan_dat_hier)
```

```
## List of 8
##  $ 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 ...
##  $ sqfoot       : num [1:120] 4.12 4.12 4.12 4.12 4.12 ...
##  $ building_idx : int [1:120] 1 1 1 1 1 1 1 1 1 1 ...
##  $ B            : int 10
##  $ building_data: num [1:10, 1:4] -0.3 -0.3 -0.3 -0.3 0.7 -0.3 -0.3 0.7
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:4] "super" "age" "ata" "mat"
##  $ K            : int 4
```

# Working in Stan: it's your turn

- Create a new stan file and call it as `hier_NB_regression.stan`

- Use the code in `multiple_NB_regression.stan`. We will modify such file implementing the hierarchical Negative Binomial regression model

- The **functions**, **transformed data** and **transformed parameters** blocks do not require to be modified

- Then, we will work only on **data**, **parameters**, **model** and **generating quantities** blocks

# Working in Stan: it's your turn

- The **data** block requires to add the building information data, that is the matrix `building_data` and the identifier `building_idx`
- **To complete**: add such information to the data block (remember to remove `super`)

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

  // To complete the building-level data
}
```

# Working in Stan: it's your turn

- The **parameter** block requires some changes
  - No changes for `inv_phi` and `alpha`
  - Now, `beta_1` is simply `beta`, while `beta_2` must be removed
  - **To complete**: Introduce the $K$-dimensional parameter `zeta` and the $B$-dimensional parameter `mu`, as well as `sigma_mu`

```
parameters {
  real<lower=0> inv_phi;
  real alpha;
  real beta;
  // To complete
}
```

# Working in Stan: it's your turn

- The **model** block requires some changes
  - Same priors for `inv_phi`, `alpha` and `beta` (previously `beta_1`)
  - **To complete**: Specify the priors for the vector `zeta`, for which $\zeta_k \sim \mathcal{N}(0, 1)$; for the `mu` vector, for which $\mu_b \sim \mathcal{N}(\alpha + \text{building\_data}_b \zeta, \sigma_\mu)$, where `sigma_mu` ($\sigma_\mu$) is distibuted according to the half-normal
  - **To complete**: Modify suitably the log-likelihood accounting for the varying intercept (you need to include `mu[building_idx]`)

```
model {
  alpha ~ normal(log(4), 1);
  beta ~ normal(-0.25, 1);
  inv_phi ~ normal(0, 1);
  //To complete
}
```

# Working in Stan: it's your turn

- The **generated quantities** block requires only a suitable change in `eta_n`, accounting for the varying intercept (**To complete**)

```
generated quantities {
  int y_rep[N];
  for (n in 1:N) {
    // To complete  specifying eta_n
    y_rep[n] = neg_binomial_2_log_safe_rng(eta_n, phi);
  }
}
```

# Varying intercept Negative Binomial hierarchical model

- Compile

```
comp_model_NB_hier <- stan_model('hier_NB_regression.stan')
```

- Sampling (we will start with 2000 iterations for each chain)

```
fit_NB_hier <- sampling(comp_model_NB_hier,
                        data = stan_dat_hier,
                        chains = 4, iter = 2000)
```

- Do you have any divergent transitions?

# Varying intercept Negative Binomial hierarchical model

```
## Warning: There were 646 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.

## Warning: Examine the pairs() plot to diagnose sampling problems

## Warning: The largest R-hat is 1.21, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#r-hat

## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior mean
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#bulk-ess

## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior vari
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#tail-ess
```

# Varying intercept Negative Binomial hierarchical model

- We get a bunch of warnings from Stan about divergent transitions, which is an indication that there may be regions of the posterior that have not been explored by the Markov chains
- For more details about divergences see https://arxiv.org/abs/1701.02434 and http://mc-stan.org/bayesplot/articles/visual-mcmc-diagnostics.html
- However, consider to increase the the number of iterations to 4000

```
fit_NB_hier <- sampling(comp_model_NB_hier,
                        data = stan_dat_hier,
                        chains = 4, iter = 4000, refresh = 0)
```

# Varying intercept Negative Binomial hierarchical model

```
## Warning: There were 804 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.

## Warning: Examine the pairs() plot to diagnose sampling problems

## Warning: The largest R-hat is 1.05, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#r-hat

## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#bulk-ess

## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior vari
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#tail-ess
```

# Varying intercept Neg. Bin. hierarchical model: diagnostics

- Here, we have divergent transitions because we need to reparametrise our model
- We will solve the problem by transforming some of the parameters, preserving the structure of the model, so that it is easier for Stan to sample from the parameter space
- Before considering the reparametrisation, we will first go through:
  - Examining the fitted parameter values, including the ESS
  - Traceplots and scatterplots that reveal particular patterns in locations of the divergences

# Varying intercept Neg. Bin. hierarchical model: diagnostics

- Print the posterior summary for the parameters that are of most interest

```
print(fit_NB_hier, pars = c('sigma_mu', 'beta', 'alpha', 'phi', 'mu'))
```

- We can see that the ESSs are low for many of the parameters relative to the total number of samples (see next slide)
- This alone is not indicative of the need to reparametrise, but it indicates that we should look further at the trace plots and pairs plots

# Varying intercept Neg. Bin. hierarchical model: diagnostics

```
## Inference for Stan model: hier_NB_regression.
## 4 chains, each with iter=4000; warmup=2000; thin=1;
## post-warmup draws per chain=2000, total post-warmup draws=8000.

##           mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
## sigma_mu  0.24    0.01 0.17  0.05  0.11  0.20  0.33  0.68   226 1.03
## beta     -0.23    0.00 0.06 -0.34 -0.27 -0.23 -0.19 -0.11   745 1.01
## alpha     1.25    0.02 0.41  0.40  0.99  1.24  1.51  2.07   743 1.01
## phi       1.59    0.01 0.35  1.03  1.34  1.55  1.78  2.40  1124 1.00
## mu[1]     1.26    0.02 0.52  0.21  0.92  1.24  1.60  2.30   864 1.01
## mu[2]     1.20    0.02 0.50  0.21  0.89  1.18  1.53  2.21   935 1.01
## mu[3]     1.41    0.02 0.46  0.48  1.09  1.43  1.72  2.32   796 1.01
## mu[4]     1.41    0.02 0.47  0.51  1.11  1.39  1.72  2.35   638 1.02
## mu[5]     1.06    0.01 0.41  0.25  0.80  1.05  1.33  1.89   803 1.01
## mu[6]     1.17    0.02 0.47  0.21  0.87  1.19  1.47  2.08   819 1.01
## mu[7]     1.45    0.02 0.50  0.45  1.13  1.43  1.78  2.45   825 1.01
## mu[8]     1.24    0.02 0.42  0.43  0.97  1.22  1.50  2.13   658 1.01
## mu[9]     1.42    0.02 0.55  0.27  1.08  1.44  1.77  2.49   803 1.01
## mu[10]    0.83    0.01 0.35  0.18  0.61  0.80  1.06  1.55   782 1.01
```

# Varying intercept Neg. Bin. hierarchical model: diagnostics

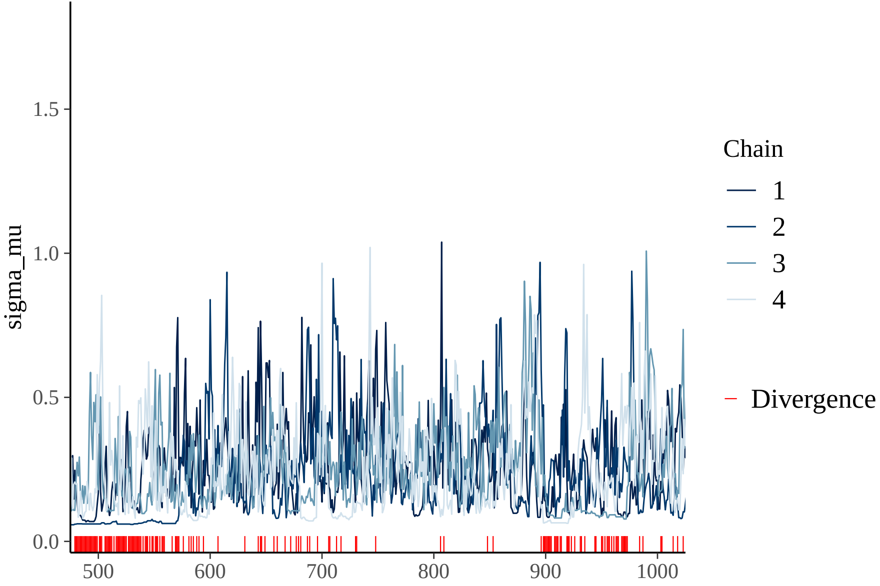
- Take a look to the traceplot of $\sigma_\mu$ to see if the divergent transitions form a pattern

```
mcmc_trace(
  as.array(fit_NB_hier, pars = 'sigma_mu'),
  np = nuts_params(fit_NB_hier),
  window = c(500, 1000)
)
```

- The `np` argument can be used to add a rug plot of the divergences to a trace plot of parameter draws. Adding a rug plot we see the samples where the sampler gets stuck at said parameter value for $\sigma_\mu$

- The `window` argument allows to zoom in

# Varying intercept Neg. Bin. hierarchical model: diagnostics



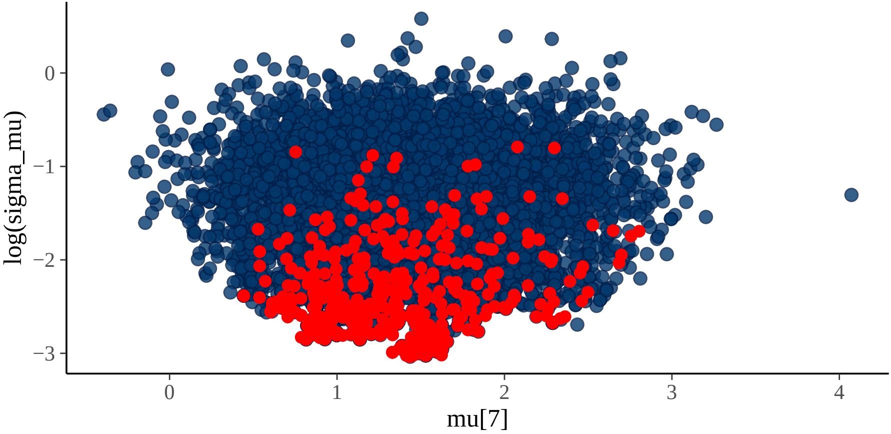
# Varying intercept Neg. Bin. hierarchical model: diagnostics

- Consider an element of $\mu$ (here $\mu_7$ ) and $\log(\sigma_\mu)$ and produce the scatterplot
- The `transform` argument allows to take the logarithm for `sigma_mu`
- Here, the `np` argument can be used to identify the divergences of the parameter draws
- We assign the plot to an object for comparing it with another plot later

```
scatter_with_divs <- mcmc_scatter(
  as.array(fit_NB_hier),
  pars = c("mu[7]", 'sigma_mu'),
  transform = list('sigma_mu' = "log"),
  np = nuts_params(fit_NB_hier))
scatter_with_divs
```

# Varying intercept Neg. Bin. hierarchical model: diagnostics

- We can see a cloud-like shape, with most of the divergences clustering towards the bottom



# Varying intercept Neg. Bin. hierarchical model: diagnostics

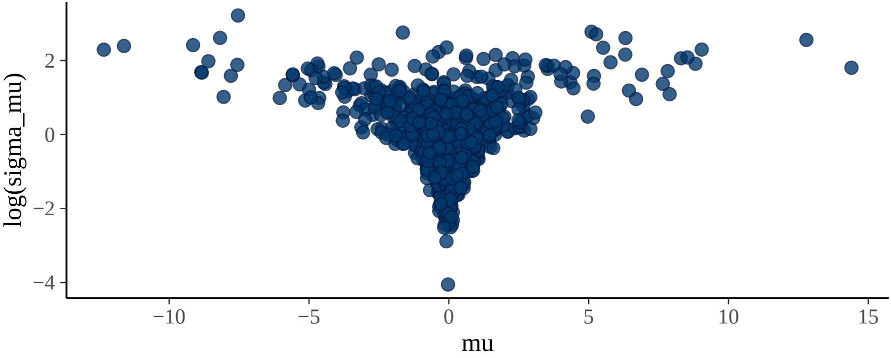
- We actually want this to look more like a funnel than a cloud
- The divergences are indicating that the sampler ca not explore the narrowing neck of the funnel
- One way to see why we should expect some version of a funnel is to look at some simulations from the prior, which we can do without MCMC and thus with no risk of sampling problems

```
N_sims <- 1000
log_sigma <- rep(NA, N_sims)
theta <- rep(NA, N_sims)
for (j in 1 : N_sims) {
  log_sigma[j] <- rnorm(1, mean = 0, sd = 1)
  theta[j] <- rnorm(1, mean = 0, sd = exp(log_sigma[j]))
}
draws <- cbind("mu" = theta, "log(sigma_mu)" = log_sigma)
mcmc_scatter(draws)
```

# Varying intercept Neg. Bin. hierarchical model: diagnostics

- We should not expect the posterior to look exactly like the prior if the data are at all informative

- The sampler is going to have to confront the funnel geometry, unless the data are highly informative about the parameter and the posterior concentrates away from the narrow neck of the funnel



# Varying intercept Neg. Bin. hierarchical model: diagnostics

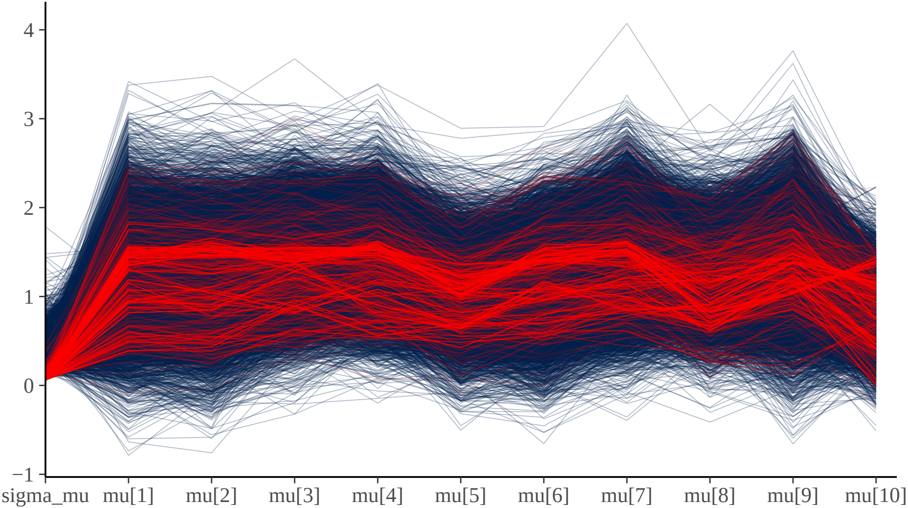
- Another graphical tools: Parallel coordinates plot

```
parcoord_with_divs <- mcmc_parcoord(
  as.array(fit_NB_hier, pars = c("sigma_mu", "mu")),
  np = nuts_params(fit_NB_hier)
)
parcoord_with_divs
```

- The optional np argument is used to pass NUTS parameter information, then divergences will be colored in the plot (red trajectories)

# Varying intercept Neg. Bin. hierarchical model: diagnostics

- It provides another evidence that our problems concentrate when `sigma_mu` is small



# Varying intercept Neg. Bin. hierarchical model: NCP

- A possible solution to this problem is represented by the use of the non-centered parameterization for $\mu_b$
- The structure of the model is the same, that is $\mu_b \sim \mathcal{N}(\alpha + \text{building\_data}_b \zeta, \sigma_\mu)$, but we decouple the dependence of the density of each element of `mu` from `sigma_mu`$(\sigma_\mu)$
- Centred: $\mu_b \sim \mathcal{N}(\alpha + \text{building\_data}_b \zeta, \sigma_\mu)$
- Non-centred: $\mu_b = \alpha + \text{building\_data}_b \zeta + \sigma_\mu \tilde{\mu}_b, \quad \tilde{\mu}_b \sim \mathcal{N}(0, 1)$

# Varying intercept Neg. Bin. hierarchical model: NCP

- In Stan, this is done by defining a vector of auxiliary variables in the parameters block `mu_raw`
- We will give a $\mathcal{N}(0, 1)$ prior to `mu_raw`
- We then make `mu` a transformed parameter, thus reparametrising the random intercept $\mu_b$ as

```
transformed parameters {
 vector[J] mu;
 mu = alpha + building_data * zeta + sigma_mu * mu_raw;
}
```

# Varying intercept Neg. Bin. hierarchical model: NCP

- The Stan file hier_NB_regression_ncp.stan uses the non-centered parameterization for mu

- We will examine the ESS of the fitted model to see whether we have fixed the problem with our reparameterization

- Compile

```
comp_model_NB_hier_ncp <- stan_model('hier_NB_regression_ncp.stan')
```

- Sampling

```
fit_NB_hier_ncp <- sampling(comp_model_NB_hier_ncp,
                            data = stan_dat_hier,
                            refresh = 0)
```

- As you can see, no divergent transitions

# Varying intercept Neg. Bin. hierarchical model: NCP

- Large improvement for the ESSs of $\mu$

```
print(fit_NB_hier_ncp, pars = c('sigma_mu','beta','alpha','phi','mu'))

## Inference for Stan model: hier_NB_regression_ncp.
## 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% 97.5% n_eff Rhat
## sigma_mu   0.24    0.00 0.18  0.01  0.10  0.20  0.33  0.68  1308    1
## beta      -0.23    0.00 0.06 -0.35 -0.27 -0.23 -0.19 -0.11  2838    1
## alpha      1.26    0.01 0.43  0.40  0.97  1.26  1.55  2.10  2838    1
## phi        1.58    0.00 0.34  1.04  1.34  1.53  1.77  2.36  4925    1
## mu[1]      1.28    0.01 0.55  0.15  0.92  1.28  1.65  2.35  2763    1
## mu[2]      1.22    0.01 0.53  0.15  0.87  1.23  1.58  2.27  2896    1
## mu[3]      1.40    0.01 0.50  0.43  1.06  1.40  1.74  2.38  3229    1
## mu[4]      1.44    0.01 0.49  0.49  1.11  1.43  1.77  2.39  3094    1
## mu[5]      1.08    0.01 0.42  0.25  0.80  1.08  1.36  1.87  3432    1
## mu[6]      1.18    0.01 0.49  0.19  0.85  1.18  1.51  2.12  2844    1
## mu[7]      1.45    0.01 0.52  0.46  1.10  1.45  1.80  2.49  3239    1
## mu[8]      1.25    0.01 0.43  0.39  0.96  1.25  1.54  2.12  3352    1
## mu[9]      1.42    0.01 0.57  0.27  1.04  1.42  1.79  2.53  2836    1
## mu[10]     0.86    0.01 0.37  0.14  0.61  0.85  1.11  1.61  3619    1
```

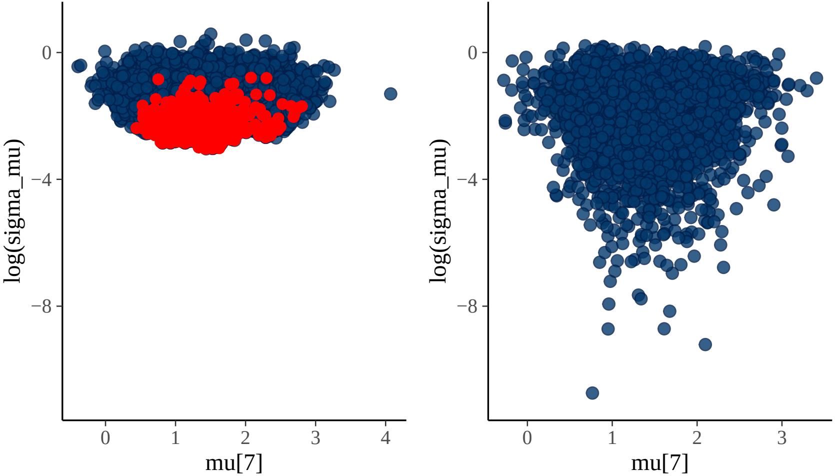
# Varying intercept NB hier. mod.: NCP - diagnostics

- Scatterplot and comparison with the centred parametrisation

```
scatter_no_divs <- mcmc_scatter(
  as.array(fit_NB_hier_ncp),
  pars = c("mu[7]", 'sigma_mu'),
  transform = list('sigma_mu' = "log"),
  np = nuts_params(fit_NB_hier_ncp)
)
bayesplot_grid(scatter_with_divs, scatter_no_divs,
               grid_args = list(ncol = 2), ylim = c(-11, 1))
```

# Varying intercept NB hier. mod.: NCP - diagnostics

- Scatterplot and comparison with the centred parametrisation

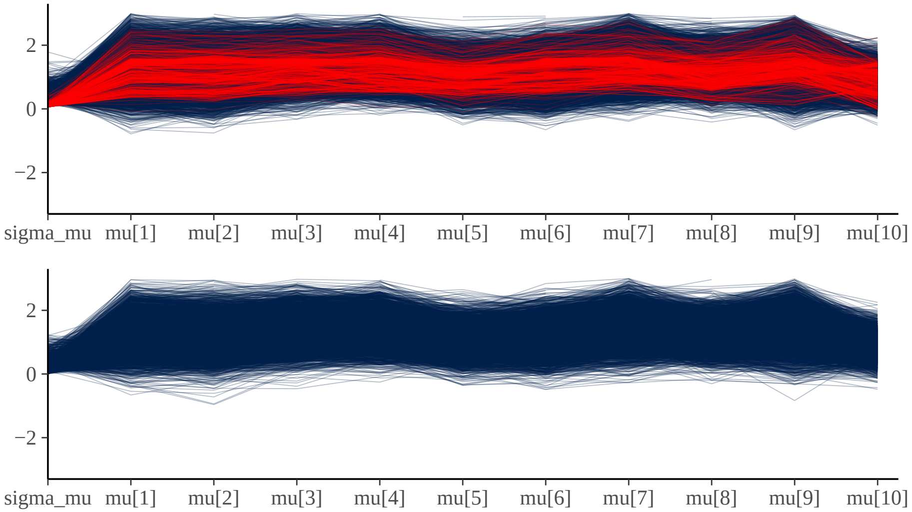


# Varying intercept NB hier. mod.: NCP - diagnostics

- Parallel coordinate plot and comparison with the centered parametrisation

```
parcoord_no_divs <- mcmc_parcoord(
  as.array(fit_NB_hier_ncp, pars = c("sigma_mu", "mu")),
  np = nuts_params(fit_NB_hier_ncp)
)
bayesplot_grid(parcoord_with_divs, parcoord_no_divs,
               ylim = c(-3, 3))
```

# Varying intercept NB hier. mod.: NCP - diagnostics



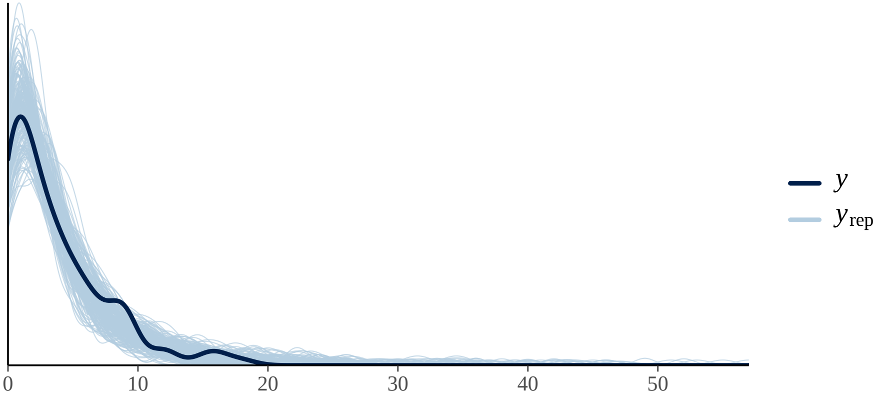
# Varying intercept NB hier. model: PPCs

- Having solved the problem of divergent transitions, we can analyse the PPCs, that is we use the posterior predictive distribution to assess the model fit
- At first, we extract the generated quantity $y^{rep}$ and, then, we compare the distribution of $y$ with the distributions of the first 200 simulated datasets

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

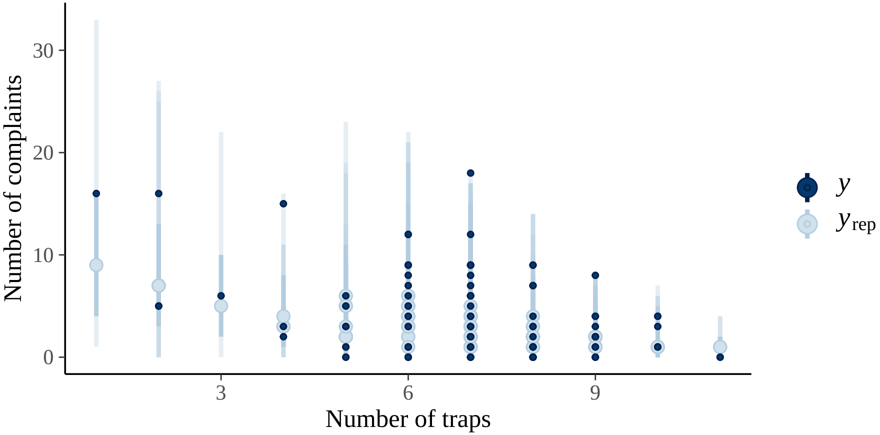
# Varying intercept NB hier. model: PPCs

- The distribution of the generated data looks like the distribution of the data we observed (however, no large difference w.r.t. to the same comparison without considering the hierarchical model)



# Varying intercept NB hier. model: PPCs

- Plot of predictions by number of bait stations, with uncertainty intervals (very similar to the same plot without considering for the hierarchical model)



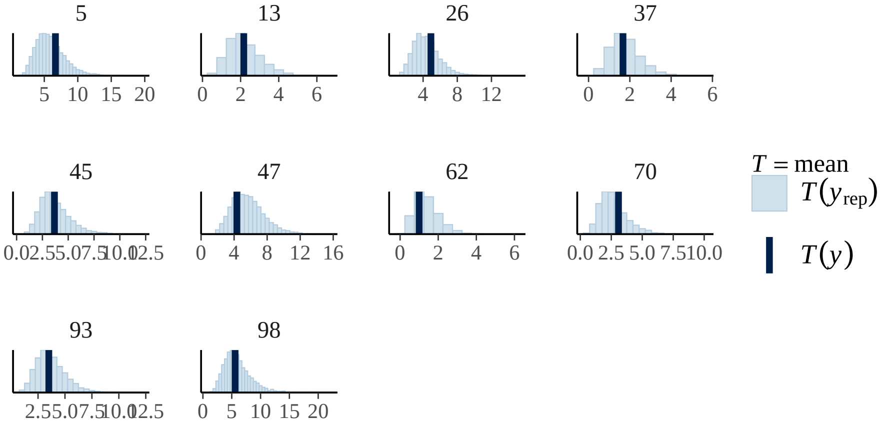
# Varying intercept NB hier. model: PPCs

- PPCs: means by building
- If we have captured the building-level means well, then the posterior distribution of means by building should match well with the observed means of the quantity of building complaints by month

```
ppc_stat_grouped(
  y = stan_dat_hier$complaints,
  yrep = y_rep,
  group = pest_data$building_id,
  stat = 'mean',
  binwidth = 0.5
)
```

# Varying intercept NB hier. model: PPCs

- Without considering the hierarchical model we were not doing bad with the building-specific means, but now they are all well captured by our model (note the improvement for the building number 70)



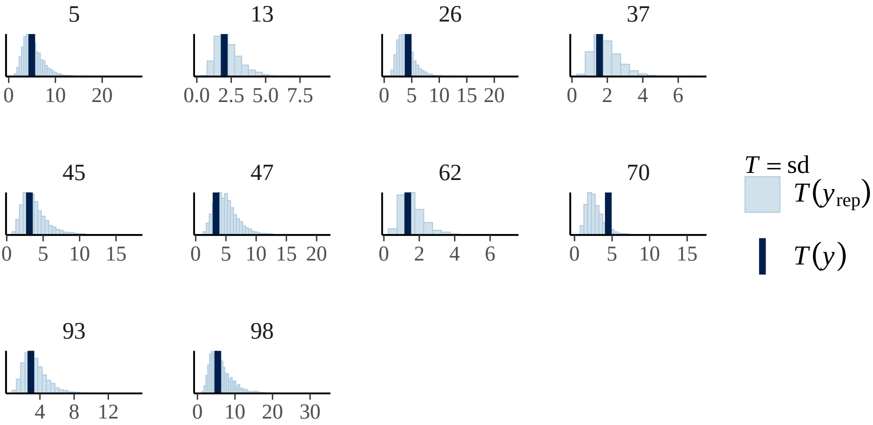
# Varying intercept NB hier. model: PPCs

- PPCs: within-building variability

```
ppc_stat_grouped(
  y = stan_dat_hier$complaints,
  yrep = y_rep,
  group = pest_data$building_id,
  stat = 'sd',
  binwidth = 0.5
)
```

# Varying intercept NB hier. model: PPCs

- The model is also able to estimate properly the within-building variability (note the improvement for the building number 70)



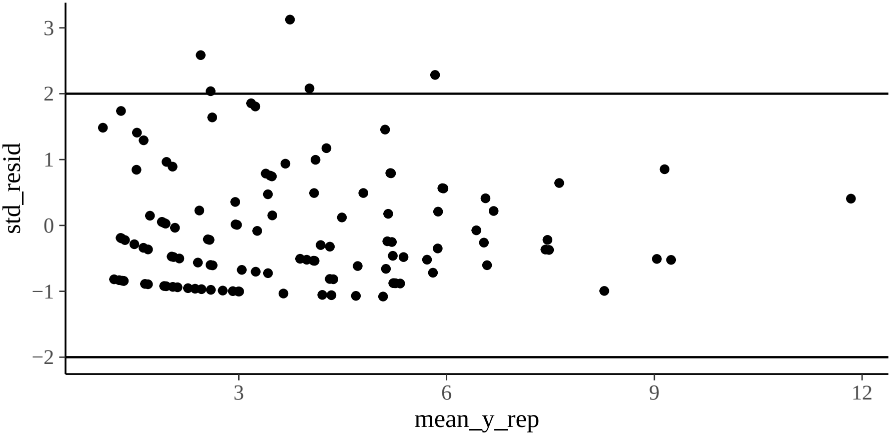
# Varying intercept NB hier. model: PPCs

- Plot of the standardised residuals

```
mean_y_rep <- colMeans(y_rep)
mean_inv_phi <- mean(as.matrix(fit_NB_hier_ncp,
                                pars = "inv_phi"))
std_resid <- (stan_dat_hier$complaints - mean_y_rep) /
  sqrt(mean_y_rep + mean_y_rep^2 * mean_inv_phi)
ggplot() +
  geom_point(mapping = aes(x = mean_y_rep, y = std_resid)) +
  geom_hline(yintercept = c(-2, 2))
```

# Varying intercept NB hier. model: PPCs

- Looks OK and we reduced a bit the large standardized residuals (compare this plot with the same plot without considering the hierarchical model)



# Exercise

- **What if instead to consider the NCP we repramétrise $\sigma_\mu$ by using $\sigma_\mu^{-1}$ and using the half-normal prior for $\sigma_\mu^{-1}$?**
- You will see that the problem of divergenent transition is fixed, but compare the ESSs of the fitted model with those obtained buy using the NCP
- Follow similar steps to the reparametrisation used for $\phi$

Section 3

# Negative Binomial hierarchical model: Varying intercept and slope

# An extended dataset for expanding the model

- We receive some new data that extends the number of time points for which we have observations for each building
- This will let us explore how to expand the model a bit more with varying slopes in addition to the varying intercepts
- Load the dataset saved as `pest_data_long.RDS` and see the structure

```
stan_dat_hier_long <- readRDS('pest_data_long.RDS')
str(stan_dat_hier_long)
```

# An extended dataset for expanting the model

```
str(stan_dat_hier_long)
```

```
## List of 10
##  $ N            : int 360
##  $ complaints   : num [1:360] 1 0 1 1 0 1 2 0 4 4 ...
##  $ traps        : num [1:360] 7 7 7 7 7 7 6 7 8 8 ...
##  $ sqfoot       : num [1:360] 4.12 4.12 4.12 4.12 4.12 ...
##  $ mo_idx       : int [1:360] 1 2 3 4 5 6 7 8 9 10 ...
##  $ M            : num 36
##  $ building_idx : int [1:360] 1 1 1 1 1 1 1 1 1 1 ...
##  $ B            : int 10
##  $ building_data: num [1:10, 1:4] -0.3 -0.3 -0.3 -0.3 0.7 -0.3 -0.3 0.7 -0.3 0.7
##   ..- attr(*, "scaled:center")= Named num [1:4] 0.3 49.4 3687.3 49.9
##   .. ..- attr(*, "names")= chr [1:4] "live_in_super" "age_of_building" "monthly_a
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:10] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:4] "live_in_super" "age_of_building" "monthly_average_rent" "a
##  $ K            : num 4
```

# Neg. Bin. hierarchical model: Varying intercept and slope

- Likely, if the levels of complaints differ by building, so does the coefficient for the effect of bait stations. So, we add these varying coefficients to our model

$$
\begin{aligned}
complaints_{b,t} &\sim \mathrm{Neg-Binomial}(\lambda_{b,t}, \phi) \\
\lambda_{b,t} &= \exp\left(\eta_{b,t}\right) \\
\eta_{b,t} &= \mu_b + \kappa_b \, traps_{b,t} + log(sqfoot)_i \\
\mu_b &\sim \mathcal{N}(\alpha + \text{building\_data}_b \zeta, \sigma_\mu) \\
\kappa_b &\sim \mathcal{N}(\beta + \text{building\_data}_b \gamma, \sigma_\kappa) \\
\alpha &\sim \mathcal{N}(\log(4), 1) \\
\zeta_k &\sim \mathcal{N}(0, 1), \quad k = 1, \ldots, 4 \\
\sigma_\mu &\sim \mathcal{TN}(0, 1, 0, +\infty) \\
\beta &\sim \mathcal{N}(-0.25, 1) \\
\gamma_k &\sim \mathcal{N}(0, 1), \quad k = 1, \ldots, 4 \\
\sigma_\kappa &\sim \mathcal{TN}(0, 1, 0, +\infty) \\
\phi^{-1} &\sim \mathcal{TN}(0, 1, 0, +\infty)
\end{aligned}
$$

# Neg. Bin. hierarchical model: Varying intercept and slope

- Compile

```
comp_model_NB_hier_slopes <- stan_model('hier_NB_regression_ncp_slopes.stan')
```

- Sampling

```
fit_NB_hier_slopes <-
  sampling(
    comp_model_NB_hier_slopes,
    data = stan_dat_hier_long,
    chains = 4, refresh = 0,
    control = list(adapt_delta = 0.95)
  )
```

```
## Warning: There were 1 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.

## Warning: Examine the pairs() plot to diagnose sampling problems
```

- The adapt_delta argument: https://mc-stan.org/rstanarm/reference/adapt_delta.html

# Neg. Bin. hierarchical model: Varying intercept and slope

- Print posterior summary: part 1

```
print(fit_NB_hier_slopes, pars = c('kappa','beta','alpha','phi','sigma_mu',

## Inference for Stan model: hier_NB_regression_ncp_slopes.
## 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% 97.5% n_eff Rhat
## kappa[1]    -0.01    0.00 0.08 -0.14 -0.07 -0.02  0.03  0.16  1132 1.00
## kappa[2]    -0.42    0.00 0.10 -0.64 -0.48 -0.41 -0.35 -0.25  1718 1.00
## kappa[3]    -0.59    0.00 0.10 -0.79 -0.65 -0.59 -0.52 -0.39  4477 1.00
## kappa[4]    -0.22    0.00 0.07 -0.36 -0.26 -0.22 -0.18 -0.09  3457 1.00
## kappa[5]    -0.60    0.00 0.09 -0.78 -0.66 -0.60 -0.54 -0.42  5890 1.00
## kappa[6]    -0.44    0.00 0.11 -0.67 -0.50 -0.43 -0.37 -0.25  2916 1.00
## kappa[7]    -0.31    0.00 0.07 -0.44 -0.35 -0.31 -0.27 -0.18  4536 1.00
## kappa[8]    -0.23    0.00 0.15 -0.57 -0.32 -0.23 -0.13  0.04  1417 1.00
## kappa[9]     0.08    0.00 0.06 -0.03  0.04  0.08  0.12  0.20  5624 1.00
## kappa[10]   -0.72    0.00 0.16 -1.00 -0.82 -0.73 -0.62 -0.36  1355 1.00
## beta        -0.35    0.00 0.07 -0.48 -0.38 -0.34 -0.31 -0.22  1779 1.00
```

# Neg. Bin. hierarchical model: Varying intercept and slope

- Print posterior summary: part 2

```
## alpha         1.40    0.01 0.30  0.79  1.21  1.41  1.60  1.97  3107 1.00
## phi           1.61    0.00 0.20  1.27  1.48  1.60  1.74  2.04  4243 1.00
## sigma_mu      0.52    0.02 0.43  0.02  0.18  0.41  0.77  1.58   624 1.00
## sigma_kappa   0.13    0.00 0.09  0.02  0.07  0.11  0.17  0.38   535 1.01
## mu[1]         0.25    0.02 0.73 -1.45 -0.15  0.35  0.75  1.43  1108 1.00
## mu[2]         1.66    0.01 0.53  0.74  1.29  1.61  1.99  2.85  1605 1.00
## mu[3]         2.13    0.00 0.32  1.51  1.91  2.13  2.34  2.77  4709 1.00
## mu[4]         1.49    0.01 0.52  0.48  1.16  1.48  1.80  2.58  3802 1.00
## mu[5]         2.39    0.01 0.42  1.56  2.11  2.39  2.67  3.24  6538 1.00
## mu[6]         1.91    0.01 0.40  1.20  1.65  1.88  2.13  2.78  2848 1.00
## mu[7]         2.68    0.00 0.25  2.20  2.51  2.68  2.84  3.18  4684 1.00
## mu[8]        -0.51    0.02 0.96 -2.29 -1.15 -0.56  0.09  1.51  1561 1.00
## mu[9]         0.21    0.01 0.57 -0.90 -0.18  0.20  0.60  1.32  5689 1.00
## mu[10]        1.79    0.03 1.11 -0.94  1.19  1.95  2.56  3.55  1049 1.00
```

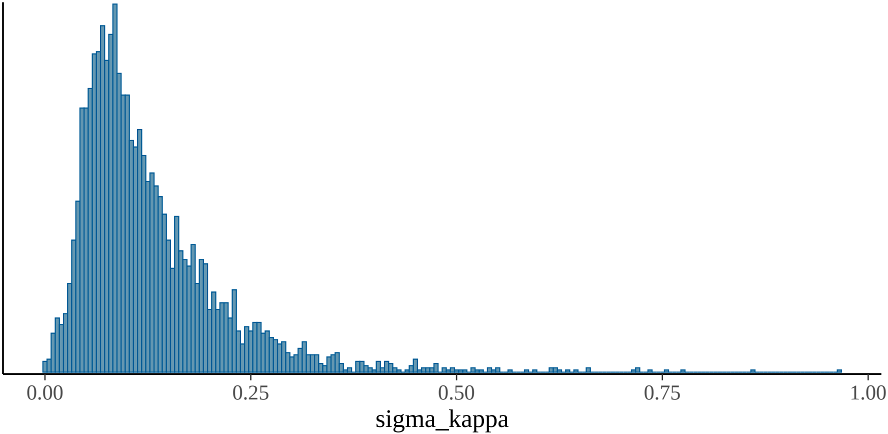
# Neg. Bin. hierarchical model: Varying intercept and slope

- Take a look to the marginal posterior distribution for `sigma_kappa` to see if the model inferss building-to-building differences in

```
mcmc_hist(
  as.matrix(fit_NB_hier_slopes, pars = "sigma_kappa"),
  binwidth = 0.005
)
```

# NB hierarchical model: Varying intercept and slope

- While the model can't specifically rule out zero from the posterior, it does have mass at small non-zero numbers, so we should leave in the hierarchy over `kappa`.



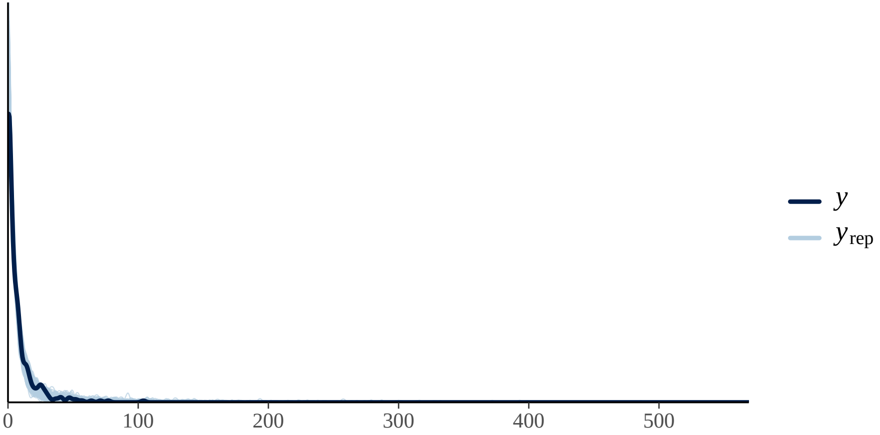
# NB hierarchical model: Varying intercept and slope - PPCs

- Examine the distribution of the generated data, comparing it with that of the original data

```
y_rep2 <- as.matrix(fit_NB_hier_slopes, pars = "y_rep")
ppc_dens_overlay(
  y = stan_dat_hier_long$complaints,
  yrep = y_rep2[1:200,]
)
```

# NB hierarchical model: Varying intercept and slope - PPCs

- The model still looks well calibrated

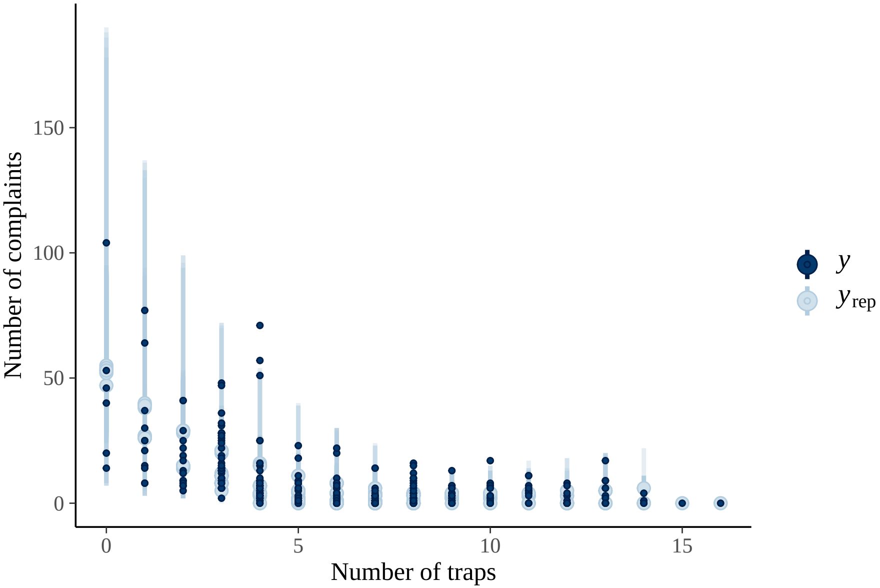


# NB hierarchical model: Varying intercept and slope - PPCs

- Plot of predictions by number of bait stations, with uncertainty intervals

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

# NB hierarchical model: Varying intercept and slope - PPCs



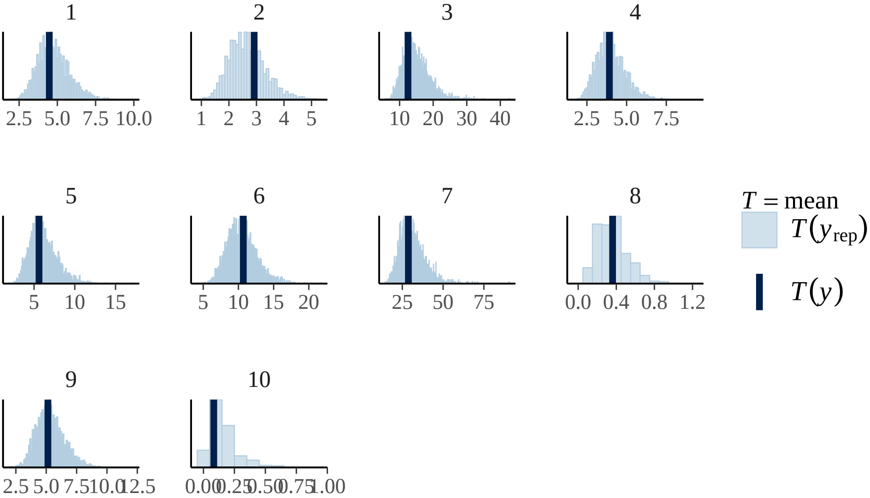
# NB hierarchical model: Varying intercept and slope - PPCs

- Means and sd by building

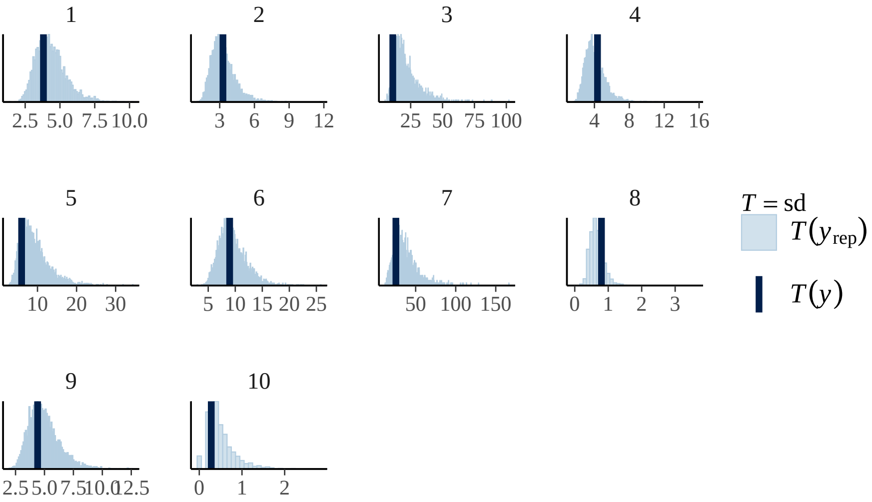
```
ppc_stat_grouped(
  y = stan_dat_hier_long$complaints,
  yrep = y_rep2,
  group = stan_dat_hier_long$building_idx,
  stat = 'mean',
  binwidth = 0.1)

ppc_stat_grouped(
  y = stan_dat_hier_long$complaints,
  yrep = y_rep2,
  group = stan_dat_hier_long$building_idx,
  stat = 'sd',
  binwidth = 0.1)
```

# NB hierarchical model: Varying intercept and slope - PPCs



# NB hierarchical model: Varying intercept and slope - PPCs

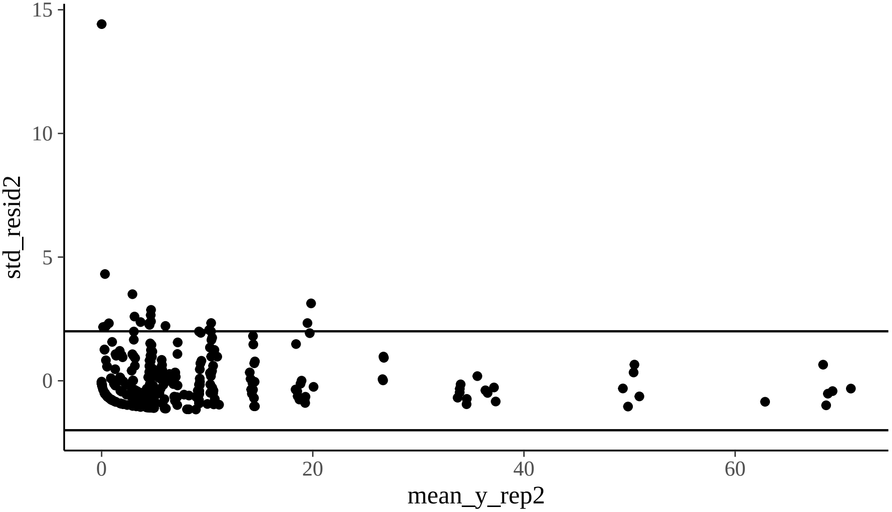


# NB hierarchical model: Varying intercept and slope - PPCs

- Standardised residuals:

```
mean_y_rep2 <- colMeans(y_rep2)
mean_inv_phi2 <- mean(as.matrix(fit_NB_hier_slopes,
                                 pars = "inv_phi"))
std_resid2 <- (stan_dat_hier_long$complaints - mean_y_rep2) /
  sqrt(mean_y_rep2 + mean_y_rep2^2*mean_inv_phi2)
ggplot() +
  geom_point(mapping = aes(x = mean_y_rep2, y = std_resid2)) +
  geom_hline(yintercept = c(-2,2))
```

# NB hierarchical model: Varying intercept and slope - PPCs



# Exercise

- Analysing the ESSs, it is apparent that those related to `sigma_mu` and `sigma_kappa` are to low

- Following the request of the previous exercise, consider a possible reparametrisation of $\sigma_\mu$ and $\sigma_\kappa$, by considering $\sigma_\mu^{-1}$ and $\sigma_\kappa^{-1}$

## Next Lab

- We will consider that the cockroaches complaints vary over time

- We will compare all the models built until now