UNIVERSITÀ
DEGLI STUDI DI TRIESTE

Dipartimento di scienze economiche,
aziendali, matematiche e statistiche
“Bruno de Finetti”

Bayesian Statistics

Model checking and diagnostics

Leonardo Egidi

A.A. 2019/20

Motivations

Once we have accomplished the first two steps of a Bayesian analysis—constructing a probability model and computing the posterior distribution of all estimands—we should not ignore the relatively easy step to assessing the fit of the model to the data and to our substantive knowledge.

It is worth to remind that we use the term *model* to encompass the sampling distribution, the prior distribution, any hierarchical structure, and issues such as which explanatory variables have been included in a regression.

Motivations

It is not correct to ask 'Is our model true or false?', since probability models in most data analysis will not be perfectly true.

The more relevant question is 'Do the model's deficiencies have a noticeable effect on the substantive inferences?'. Remember the George E.P. Box quote:

All models are wrong, but some are useful.

How to judge when assumptions of convenience can be made safely is a central task of Bayesian sensitivity analysis. Failures in the model lead to practical problems by creating false inferences about estimands of interest.

The external validation paradigm

We can check a model by external validation using the model to make predictions about future/hypothetical data, and then collecting those data and comparing to their predictions.

Bayesian analysis uses *posterior predictive checking* to check the joint posterior predictive distribution of future data given the data at hand, $p(\tilde{y}|y)$.

The idea is the following: if the model fits, then replicated data under the model should look similar to observed data. To put in another way, the observed data should look *plausible* under the posterior predictive distribution.

Posterior predictive checking

The basic technique for checking the fit of a model is to draw simulated values from the joint posterior predictive distribution of replicated data and compare these samples to the observed data. Any systematic differences between the simulation and the data indicate potential failings of the model.

We define y^{rep} as the replicated data that *could have been observed*. We distinguish between y^{rep} and $\tilde{y}$:

- $\tilde{y}$ is any future observable value or vector of observable quantities (out-of-sample replication)
- y^{rep} is specifically a replication just like y (in-sample replication)

Posterior predictive checking

The posterior predictive distribution of y^{rep} given the current state of knowledge is:

$$p(y^{\text{rep}}|y) = \int \underbrace{p(y^{\text{rep}}|\theta)}_{\textit{Likelihood}} \underbrace{\pi(\theta|y)}_{\textit{hyp. Posterior}} d\theta. \tag{1}$$

We measure the *discrepancy* between model and data by defining some test quantities $T(y, \theta)$, the aspects of the data we wish to check. T is a scalar summary of parameters and data that is used to compare data to predictive simulations.

Test (or discrepancy) quantities play the role in Bayesian model checking that test statistics play in classical testing.

Classical p-value vs Bayesian p-value

Lack of fit of the data with respect to the ppd can be measured by the *tail-area* probability, or p-value, of the test quantity, and computed using posterior simulations of (θ, y^{rep}). We define the p-value mathematically, first for classical inference.

The classical p-value for the test statistic $T(y)$ is

$$p_C = \Pr(T(y^{\text{rep}}) \geq T(y)|\theta), \qquad (2)$$

where the probability is taken over the distribution of y^{rep} with θ fixed.

Classical p-value vs Bayesian p-value

The Bayesian p-value is the probability that the replicated data could be more extreme than the observed data, as measured by the test quantity:

$$p_B = \Pr(T(y^{\text{rep}}, \theta) \geq T(y, \theta)|y), \qquad (3)$$

where the probability is taken over the posterior distribution of θ and the ppd of y^{rep}:

$$\begin{aligned} p_B &= \int \int |T(y^{\text{rep}}, \theta) \geq T(y, \theta)| \, p(y^{\text{rep}}|\theta)\pi(\theta|y)dy^{\text{rep}}\, d\theta \\ &= \int p_C \pi(\theta|y)d\theta, \end{aligned}$$

where $|\cdot|$ denotes the indicator function. Thus the Bayesian p-value is an average of the classical p-value over θ.

Bayesian p-values in practice

In practice, we usually compute the ppd (1) using *simulation*. Specifically, this happens with a two-steps procedure:

- Suppose to have S simulations $\theta^{(s)}$, $s = 1, \ldots, S$ from the posterior distribution.
- We generate S draws $y^{\text{rep}(s)}$ from $p(y^{\text{rep}}|\theta^{(s)})$.
- We compute now $T(y^{\text{rep}}, \theta)$: the estimated Bayesian p-value for (3) is the proportion of these S simulations for which the test quantity equals or exceeds its realized values; that is, for which $T(y^{\text{rep}(s)}, \theta) \geq T(y, \theta)$.

Bayesian p-values in practice

Thus, we almost never have a closed form for (1). What we do, is performing something similar to Monte Carlo simulation, and approximating the integral in (1) by the sum over the S draws:

$$\sum_{s=1}^{S} p(y^{\text{rep}(s)}|\theta^{(s)})\pi(\theta^{(s)}|y). \tag{4}$$

The resulting estimation of (3) is then equal to:

$$\frac{1}{S}\sum_{s=1}^{S} |T(y^{\text{rep}(s)}, \theta^{(s)}) \geq T(y, \theta^{(s)})|. \tag{5}$$

Indice

Eight schools: model checking

Consider again the eight schools example about the effects of special coaching programs on test scores in each of eight high-schools:

$$y_{ij} \sim \mathcal{N}(\theta_j, \sigma_y^2)$$
$$\theta_j \sim \mathcal{N}(\mu, \tau^2)$$

The example is based on many assumptions:

1. normality of the estimates y_j given θ_j and σ_j, where the σ_j are assumed known;
2. exchangeability of the prior distribution of the θ_j's;
3. normality of the prior distribution of each θ_j given μ and τ.

The exchangeability assumption means that we will let the data tell us about the relative ordering and similarity of effects in the eight schools.

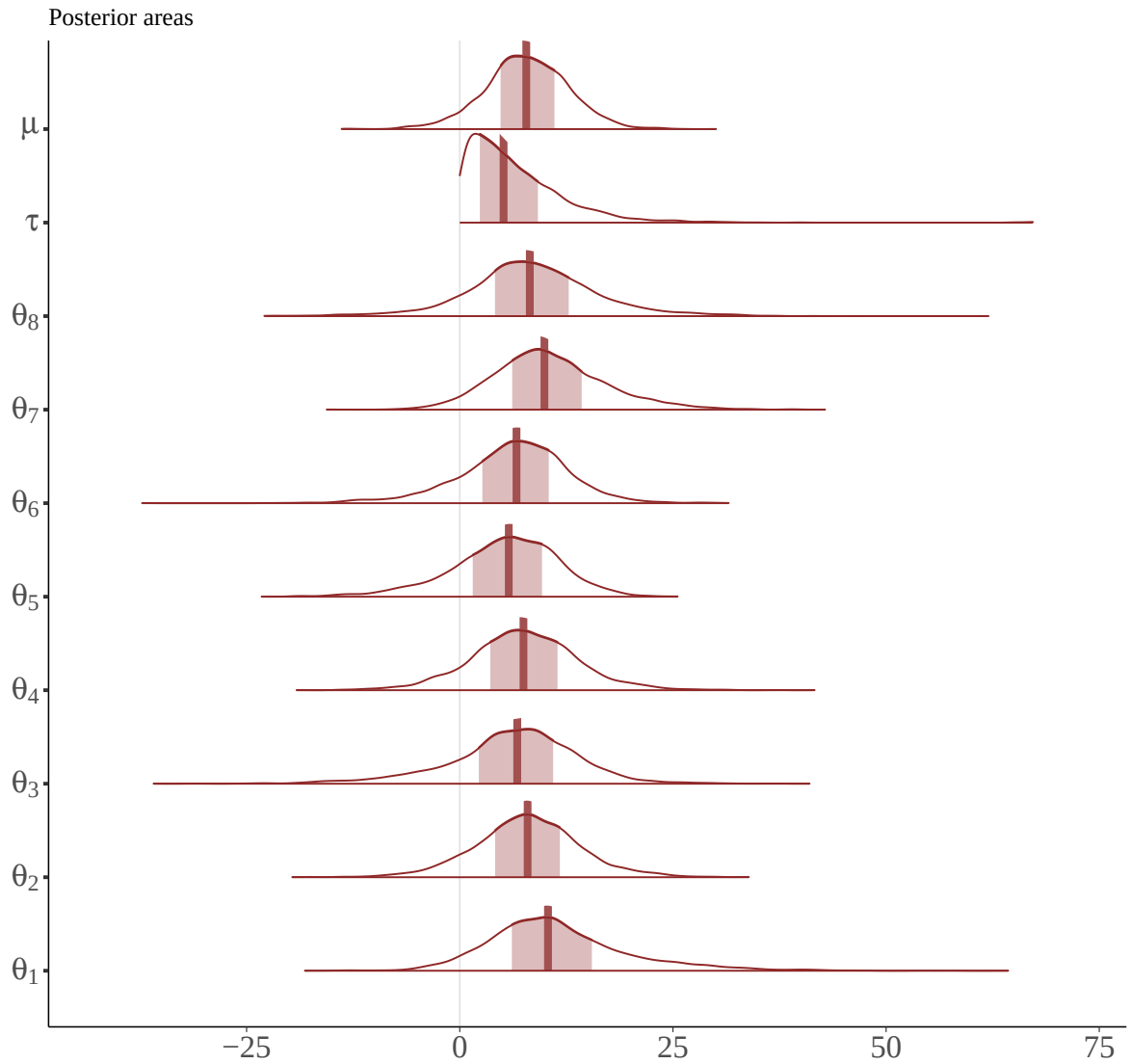
Eight schools: Stan model

```
data {
  int<lower=0> J; // number of schools
  real y[J]; // estimated treatment effects
  real<lower=0> sigma[J]; // s.e. of effect estimates
}
parameters {
  real mu;
  real<lower=0> tau;
  real eta[J];
}
transformed parameters {
  real theta[J];
  for (j in 1:J)
    theta[j] = mu + tau * eta[j];
}
```

Eight schools: Stan model (cont.)

```
model {
  target += normal_lpdf(eta | 0, 1);
  target += normal_lpdf(y | theta, sigma);
}
generated quantities {
  real y_rep[J];
  for (j in 1:J)
     y_rep[j] = normal_rng(theta[j], sigma[j]);
}
```

Eight schools: estimation



Replications

We simulate the ppd of a hypothetical replication of the experiment. In Stan, we do this by coding the cycled instruction:

```
y_rep[j] = normal_rng(theta[j], sigma[j]);
```

We have now S draws for the replicated vector $y^{\text{rep}} = (y_1^{\text{rep}}, \ldots, y_8^{\text{rep}})$. We should now visualize this distribution over the S draws and detect eventual deficiencies of the model.

We will perform many kinds of pp checks. The main tool here is visualization. All the plots are obtained with the `bayesplot` package.

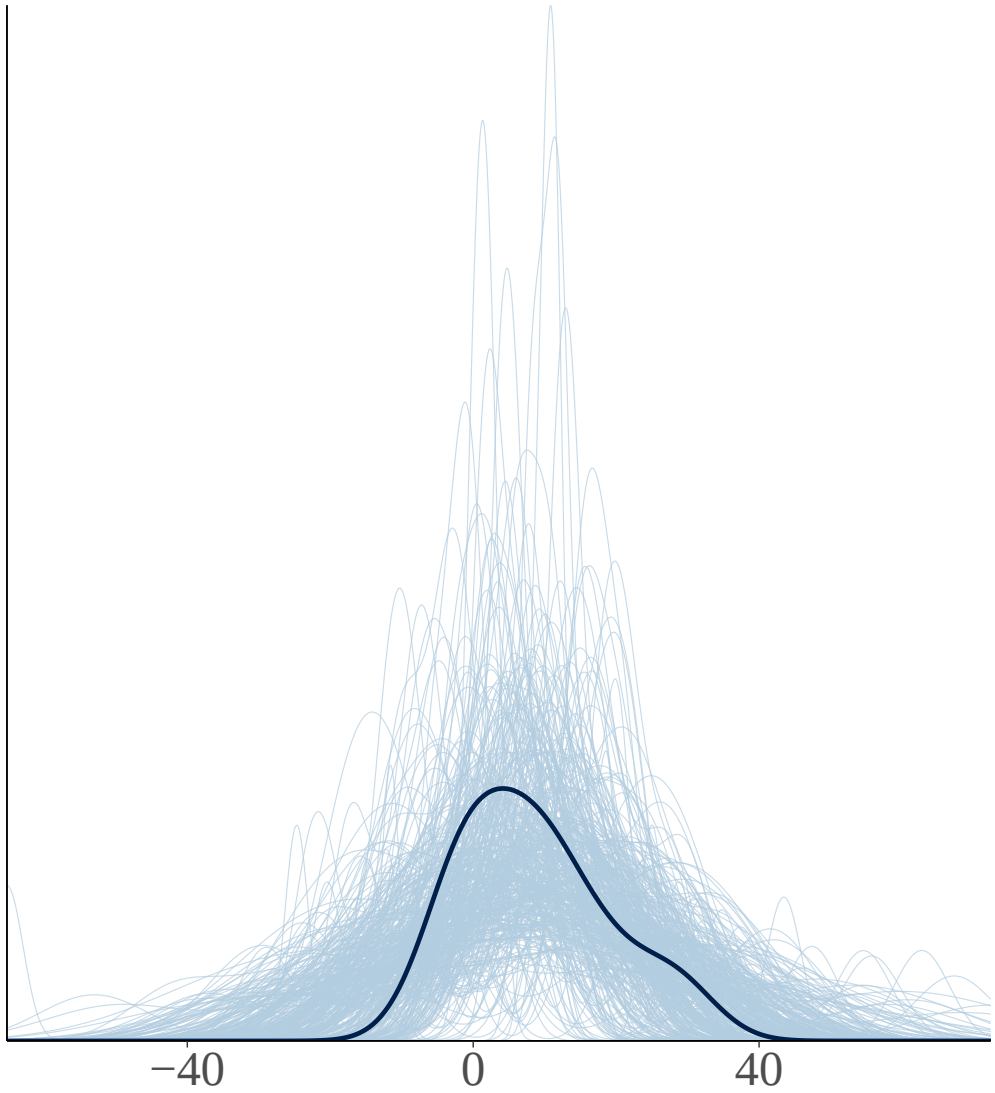
Graphical posterior predictive checks

The basic idea of graphical model checking is to *display the data alongside simulated data* from the fitted model, and to look for systematic discrepancies between real and simulated data. Essentially, we may recognize three kinds of graphical display:

- direct display of all the data
- display of data summaries or parameter inferences
- graphs of residuals or other measures of discrepancy between model and data.

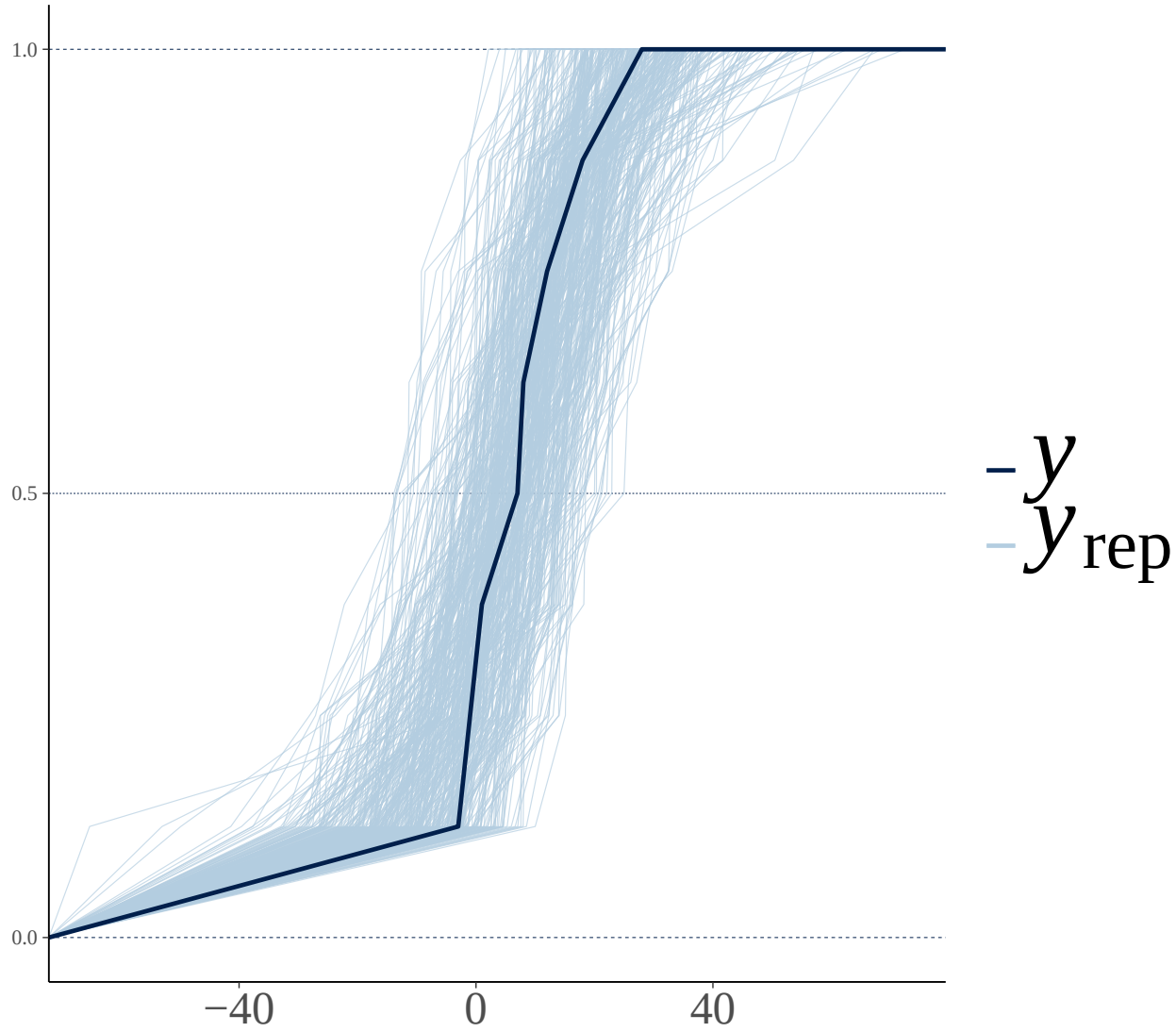
Check 1: distribution of replicated data vs real data

```
ppc_dens_overlay(y,y_rep)
```



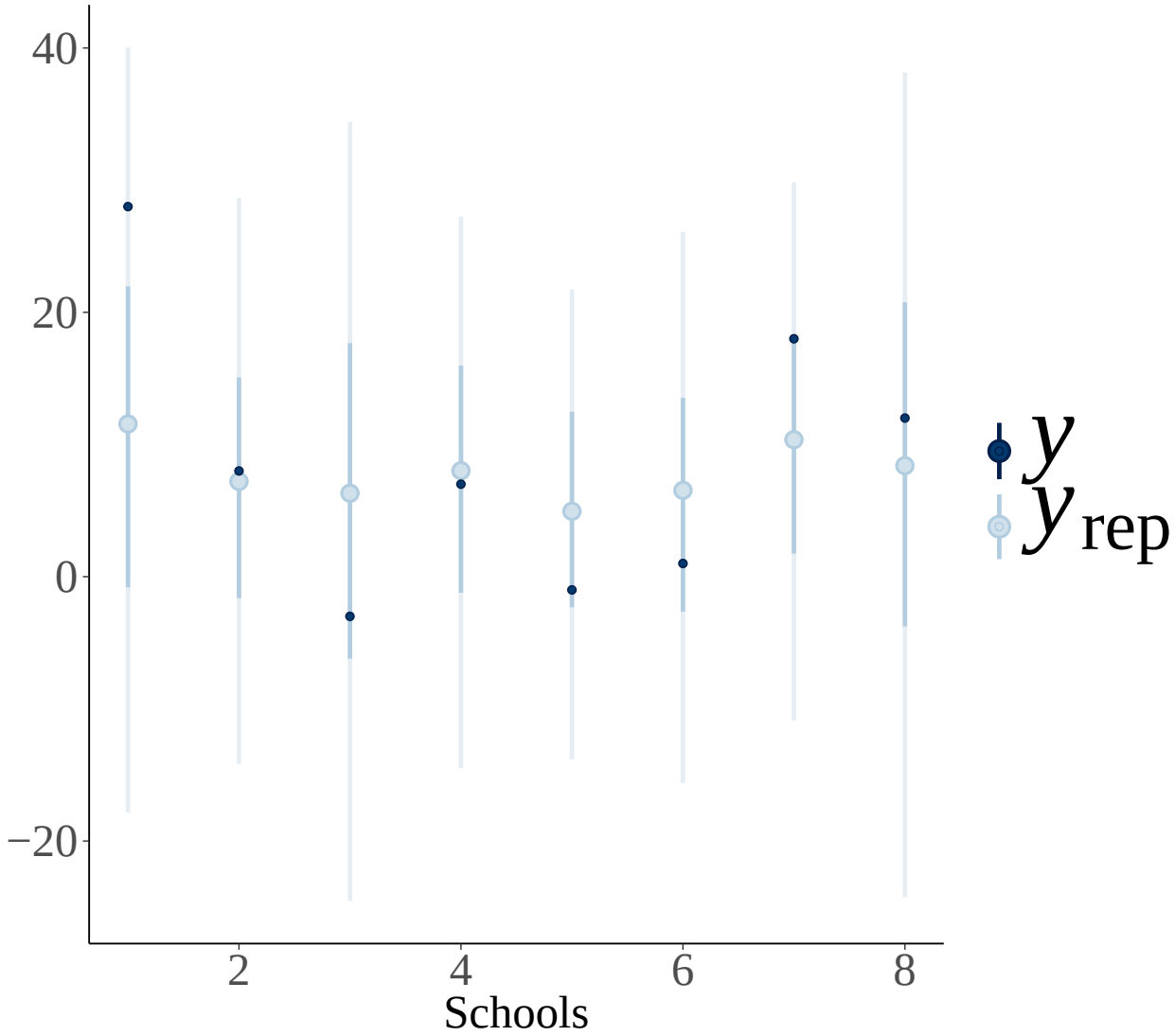
Check 2: empirical distribution function

```
ppc_ecdf_overlay(y,y_rep)
```



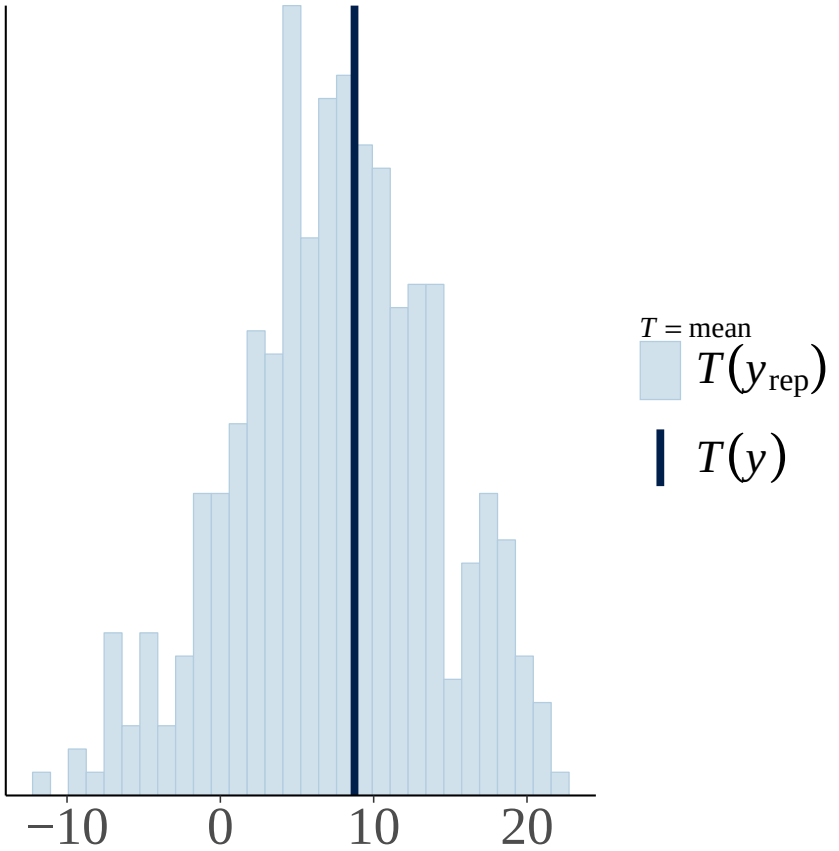
Check 3: predictive intervals vs observed values

```
ppc_intervals(y,y_rep)
```

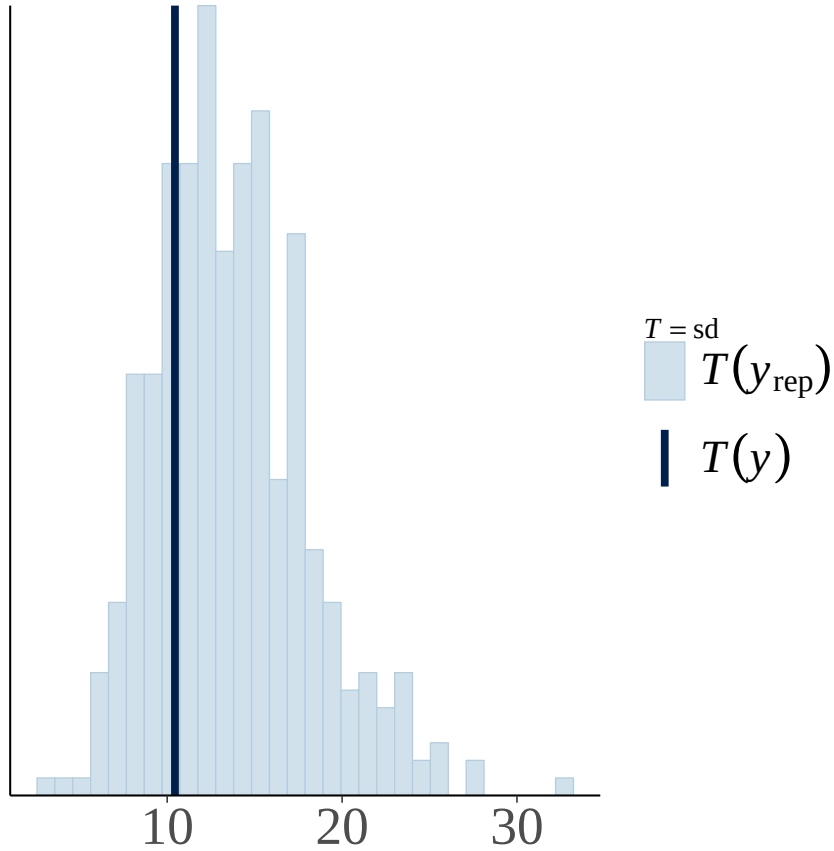


Check 4: statistics

```
ppc_stat(y,y_rep)
```



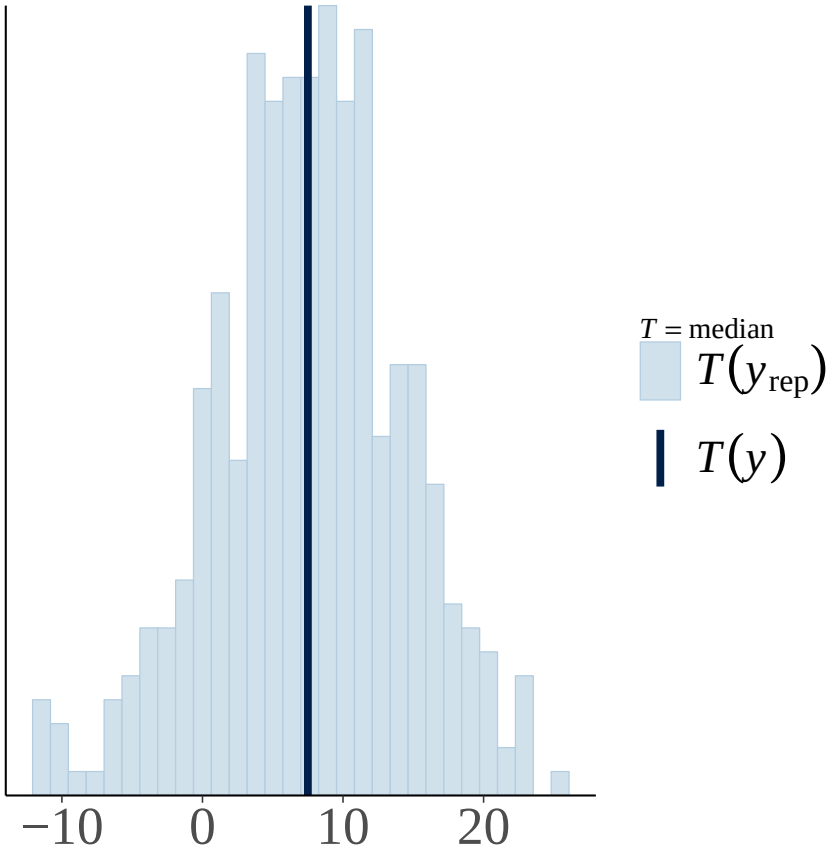
(a) $T(y) = \bar{y}$



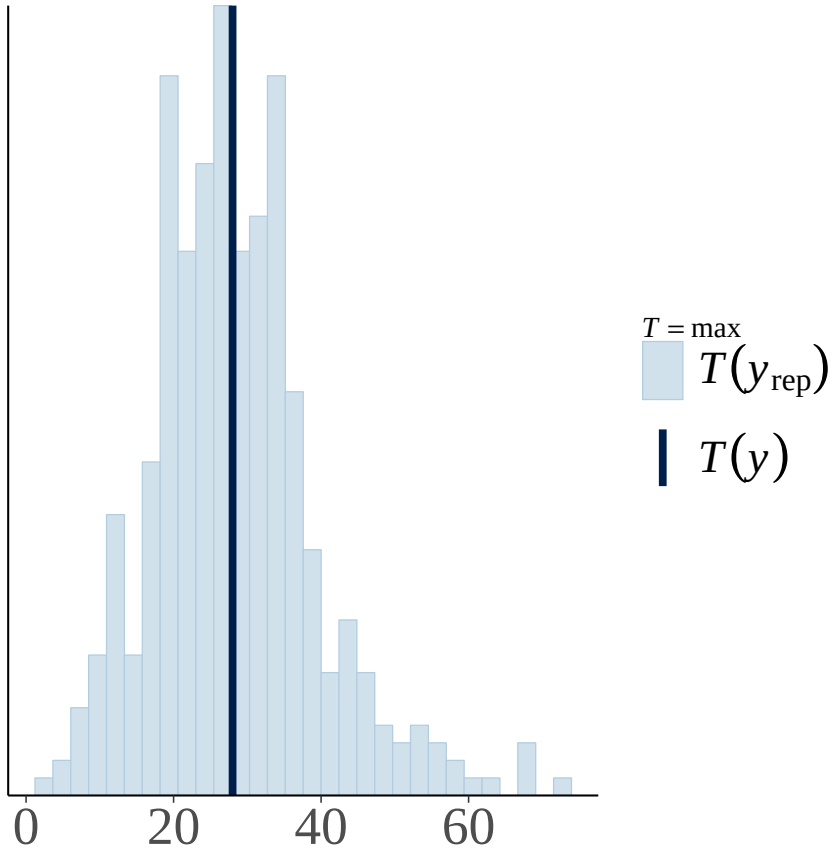
(b) $T(y) = \mathrm{sd}(y)$

Check 4: statistics

```
ppc_stat(y,y_rep)
```



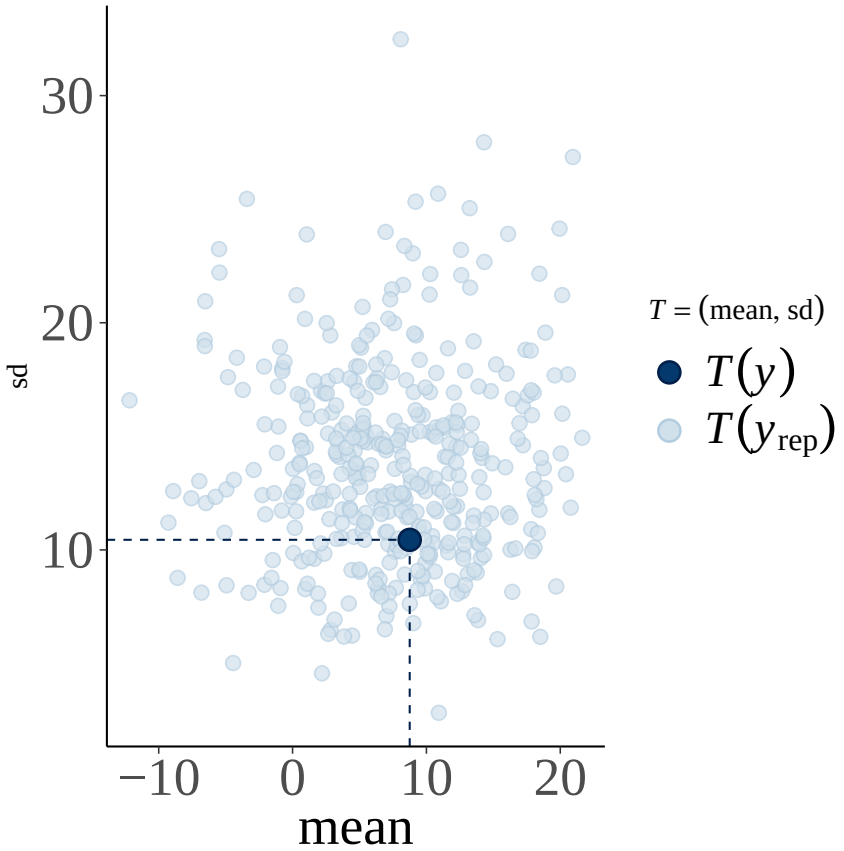
(c) $T(y) = \text{Me}(y)$



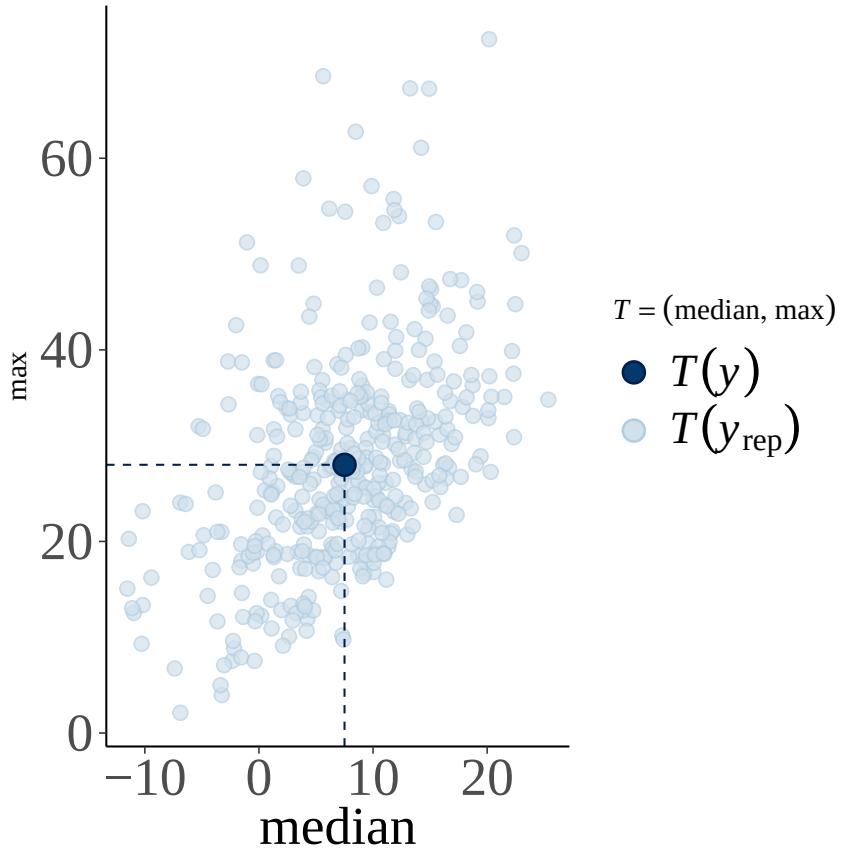
(d) $T(y) = \max(y)$

Check 5: bivariate statistics

```
ppc_stat_2d(y,y_rep)
```



(e) Mean and sd



(f) Median and max

Comments for the pp checks

- The graphical summaries suggest that the model generates predicted results similar to the observed data in the study. Observed test statistics fall within their replicated distributions (Check 4), and the distribution of the data is coherent with the replicated ones (Check 1 and 2).
- As a further measure of discrepancy, we may compute the estimated Bayesian p-value (3) from check 4: in each of the four considered statistics, $p_B \approx 0.5$. Remember that a model is suspect if p_B is close to 0 or 1. If a p-value is close to 0 or 1, it is not so important exactly how extreme it is!

Indice

Pest control example

Remind the simple Poisson regression for the cockroaches:

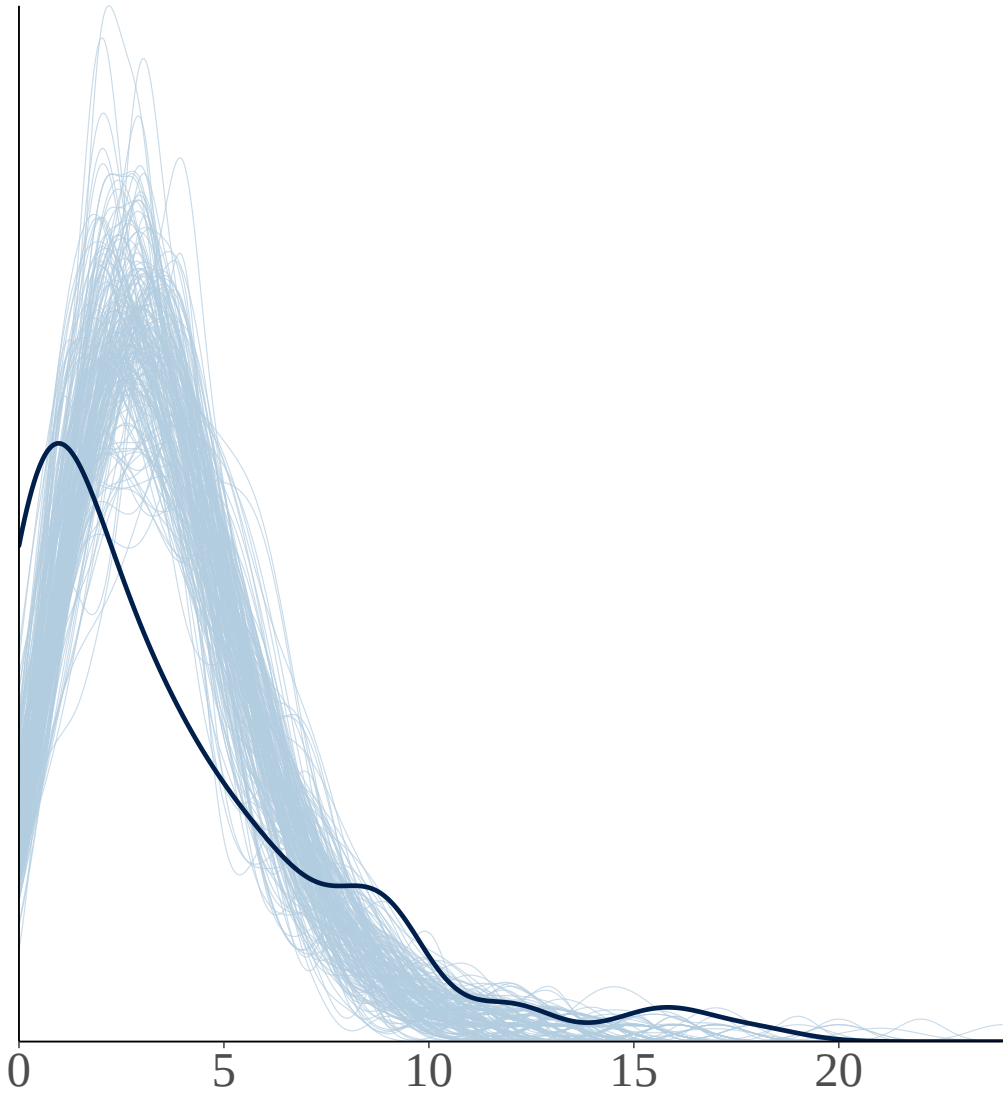
$$\begin{aligned} \text{complaints}_i &\sim \text{Poisson}(\lambda_i) \\ \lambda_i &= \exp(\eta_i) \\ \eta_i &= \alpha + \beta \, \text{traps}_i \end{aligned}$$

We fit the model in Stan and we obtain the following posterior estimates (R output):

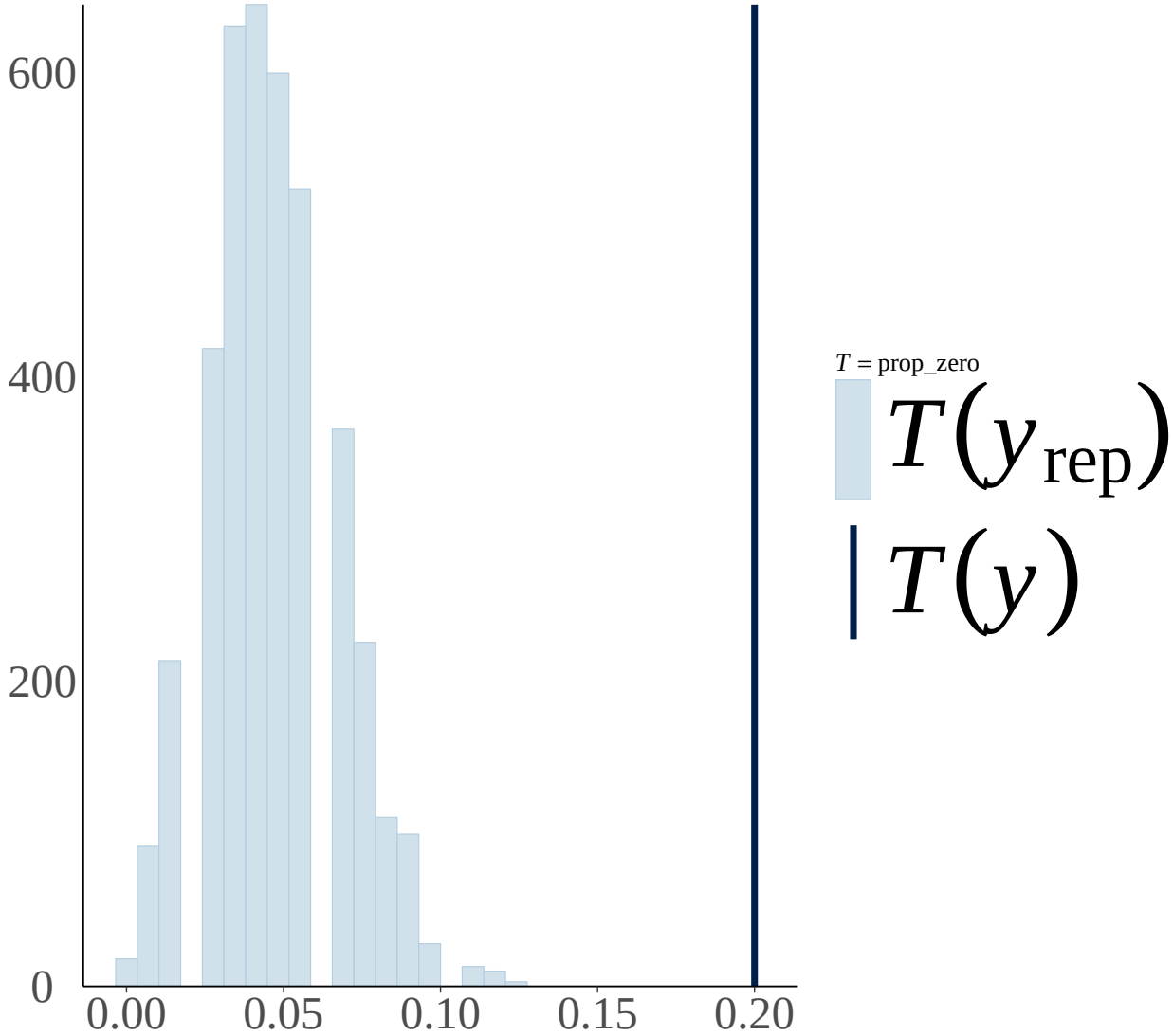
	mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
alpha	2.58	0.15	2.28	2.48	2.58	2.69	2.88	979	1
beta	-0.19	0.02	-0.24	-0.21	-0.19	-0.18	-0.15	997	1

Pest control: pp check. Densities

We check the model via some simulated data:



Pest control: pp check. Proportion of zeros

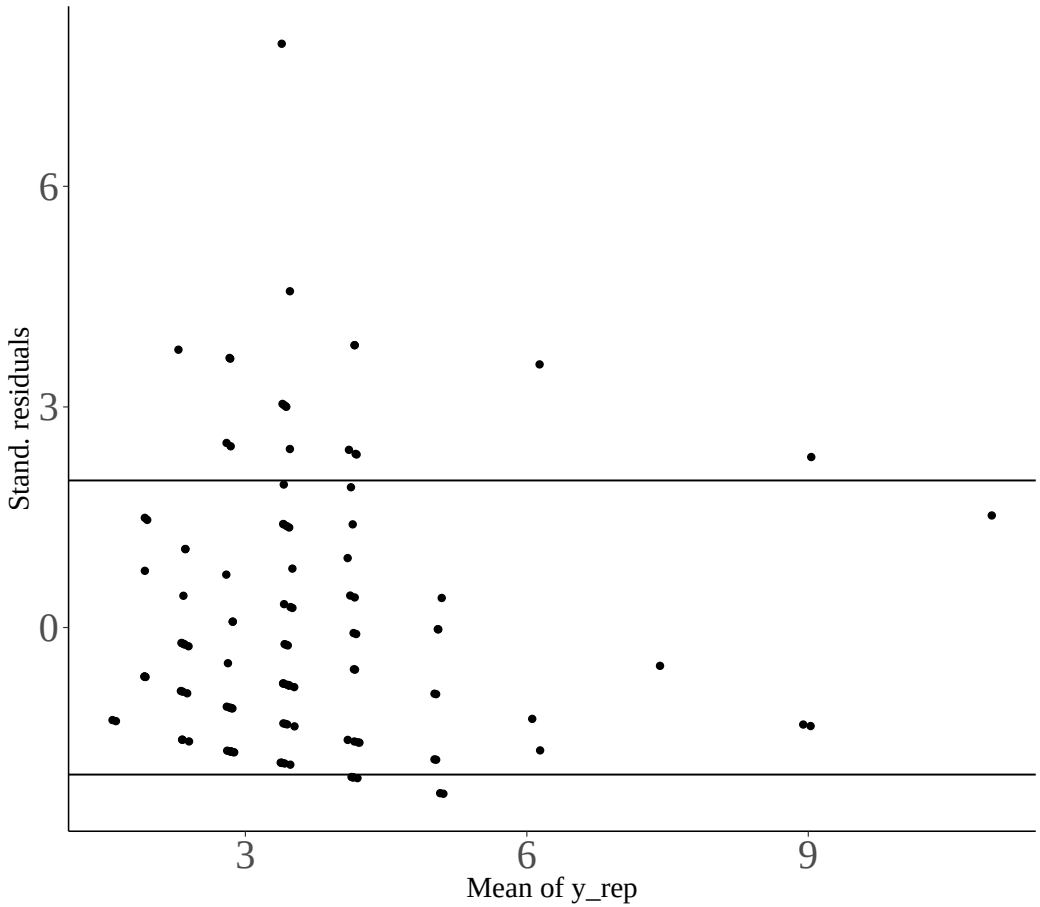


Pest control: pp check.

Comments:

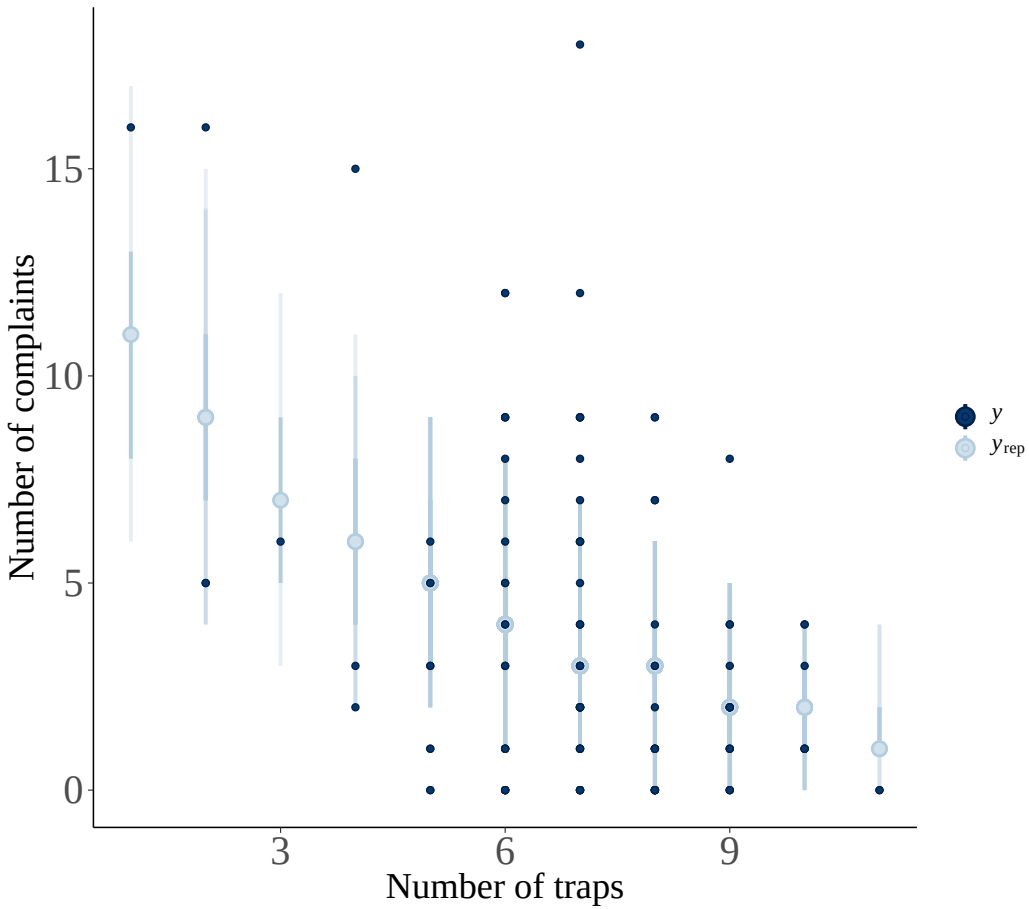
- We immediately realize that replicated distributions are far from the observed data distribution, and that the proportion of zero assumed by the Poisson model is quite underestimated...It is clear that the model does not capture this feature of the data well at all.
- Maybe the Poisson distribution distribution is not suited in this case...let's still explore the standardised residuals of the observed vs predicted number of complaints.
- We can also view how the predicted number of complaints varies with the number of traps.

Pest control: pp check. Residuals



It looks as though we have more positive residuals than negative $\Rightarrow$ the model tends to underestimate the number of complaints.

Pest control: pp check. Predictive intervals



We can see that the model does not seem to fully capture the data.

Strategies when a pp check fails

What to do if a pp check 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.

In what follows, we do not include further predictors, but we will work on the choice of the sampling distribution and, finally, we will consider further hierarchies.

Pest example. Negative binomial model

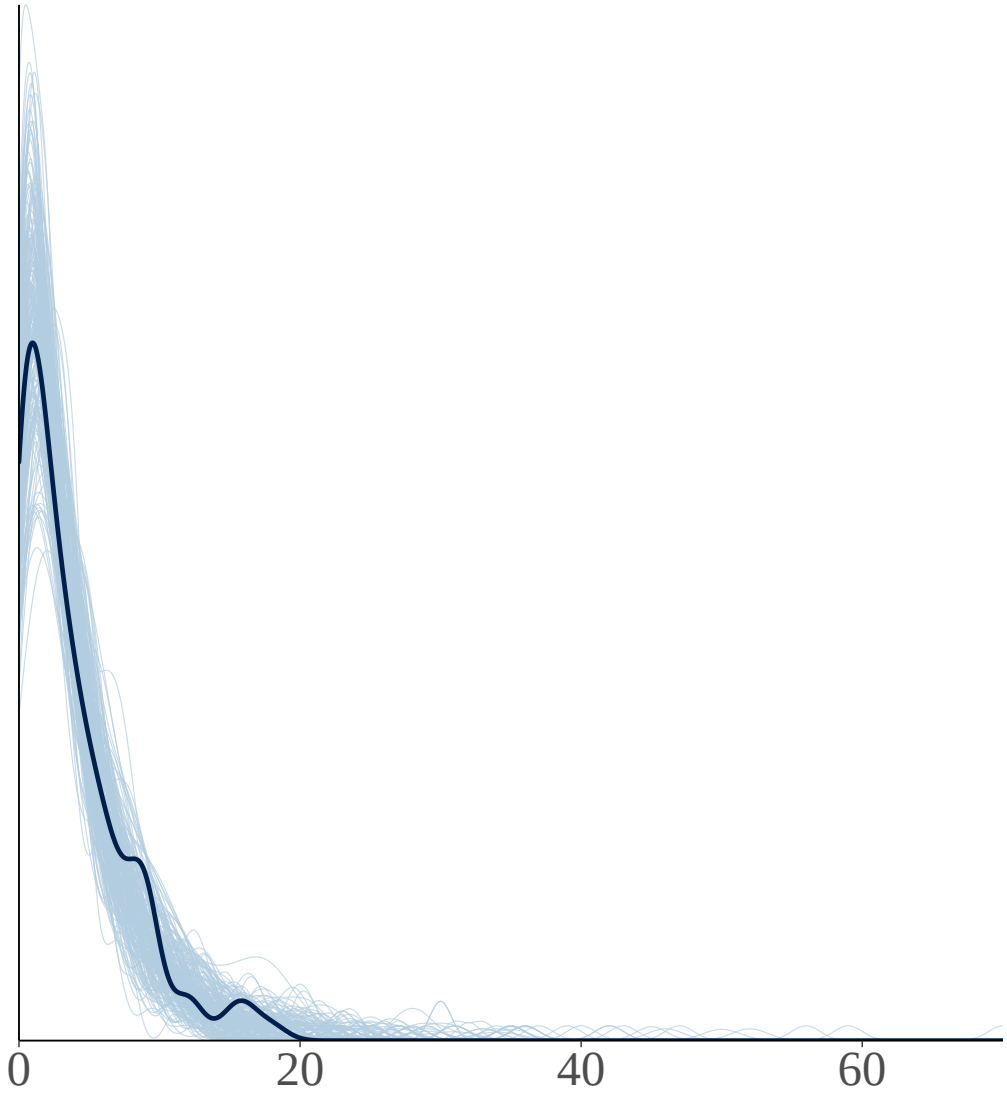
As already seen, negative binomial distribution may capture the *overdispersion* in the data with the parameter ϕ:

$$\text{complaints}_i \sim \text{Neg-Binomial}(\lambda_i, \phi)$$
$$\lambda_i = \exp(\eta_i)$$
$$\eta_i = \alpha + \beta \, \text{traps}_i$$

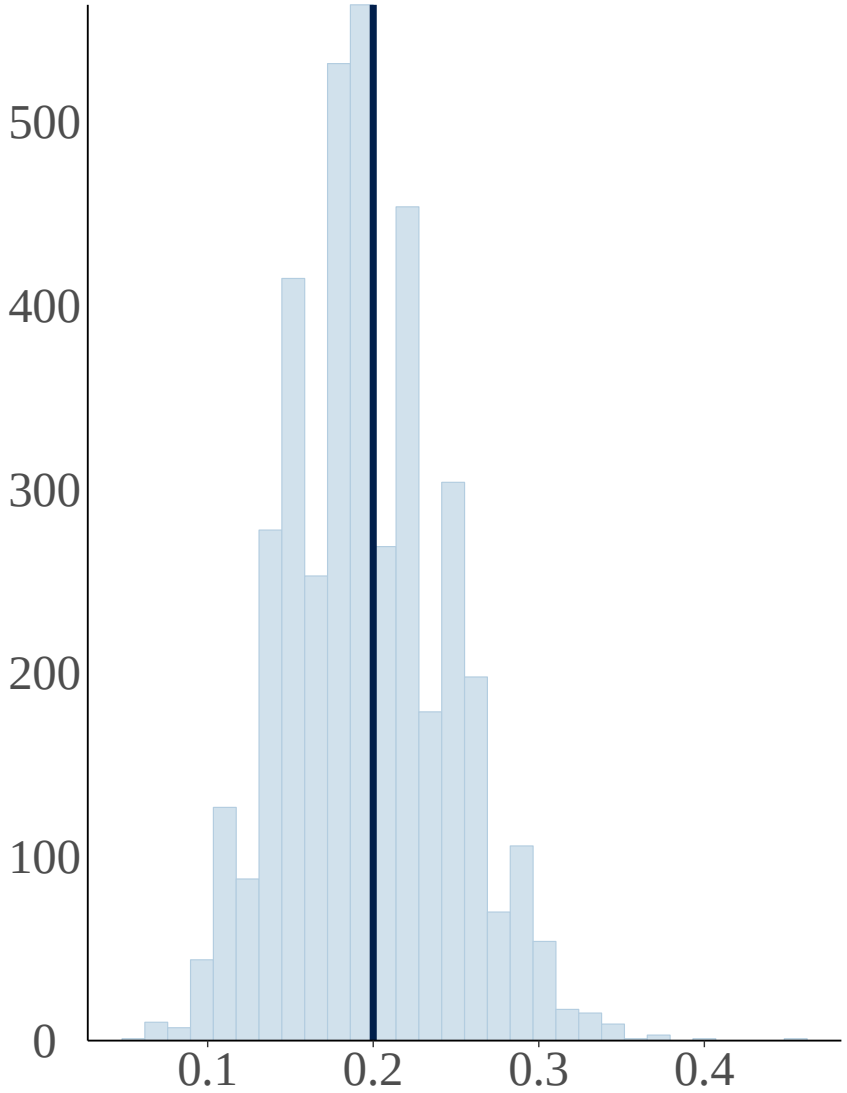
We fit also the negative-binomial model in Stan:

	mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
alpha	2.49	0.34	1.81	2.26	2.49	2.73	3.16	1177	1
beta	-0.18	0.05	-0.27	-0.21	-0.18	-0.15	-0.09	1167	1

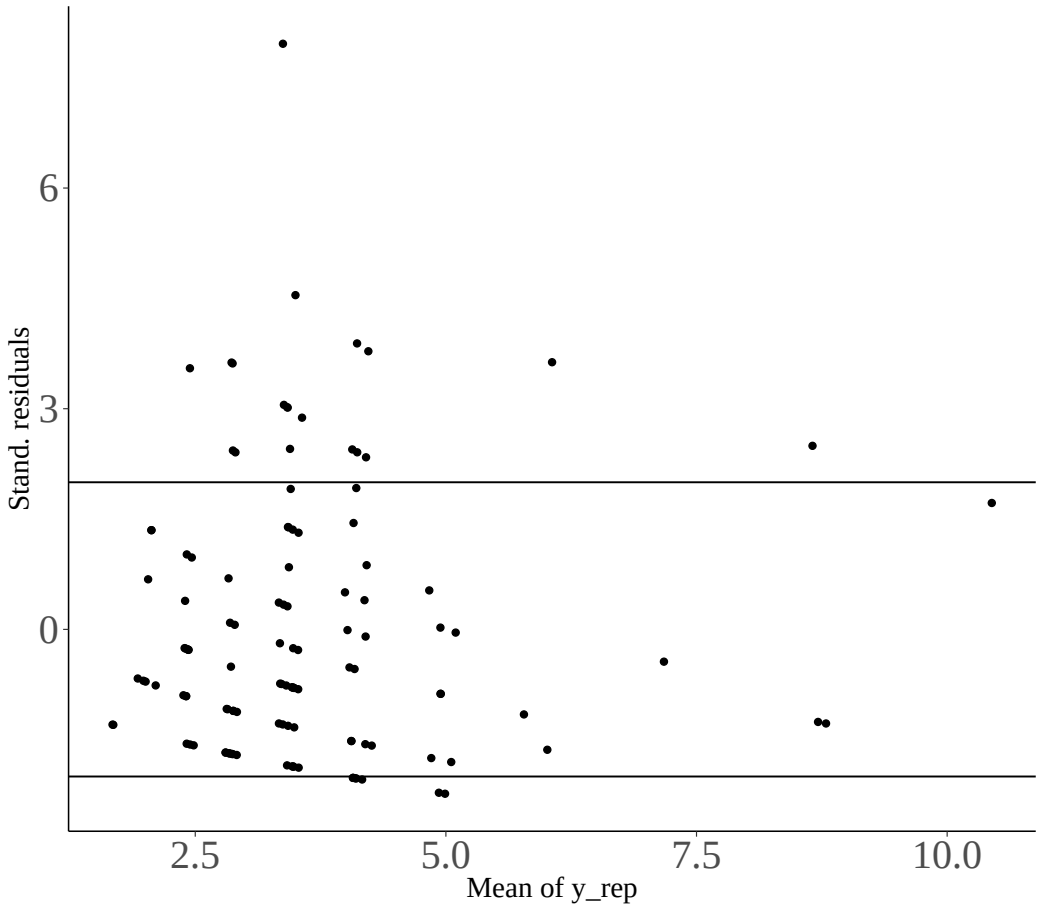
Pest control, NB model. PP check: densities



Pest control, NB model: pp check. Proportion of zeros



Pest control, NB model: pp check. Residuals



It looks as though we have more positive residuals than negative $\Rightarrow$ the model tends to underestimate the number of complaints.

Comments for pp check, NB model

- It appears that our model now captures both the number of small counts better as well as the tails. The negative binomial model does a better job in capturing the number of zeros.
- However, we still have some very large standardized residuals. This might be because we are currently ignoring that the data are clustered by buildings, and that the probability of roach issue may vary substantially across buildings.

Pest control: Hierarchical modelling

Let's add a hierarchical intercept parameter, α_b at the building level to our model. Thus, for the i-th complaint in the b-th building we have:

$$\begin{aligned}
\text{complaints}_{ib} &\sim \text{Neg-Binomial}(\lambda_{ib}, \phi) \\
\lambda_{ib} &= \exp\left(\eta_{ib}\right) \\
\eta_{ib} &= \alpha_{b(i)} + \beta\,\text{traps}_i + \beta_{\text{super}}\,\text{super}_i + \text{log_sq_foot}_i \\
\alpha_b &\sim \mathcal{N}(\mu, \sigma_\alpha^2)
\end{aligned}$$

One of our predictors varies only by building, so we can rewrite the above model more efficiently like so:

$$\begin{aligned}
\eta_{ib} &= \alpha_{b(i)} + \beta\,\text{traps}_i + \text{log_sq_foot}_i \\
\alpha_b &\sim \mathcal{N}(\mu + \beta_{\text{super}}\,\text{super}_i, \sigma_\alpha^2)
\end{aligned}$$

Pest control: Hierarchical modelling

We have more information at the building level as well, like the average age of the residents, the average age of the buildings, and the average per-apartment monthly rent so we can add that data into a matrix called `building_data`, which will have one row per building and four columns:

- `live_in_super`: An indicator for whether the building has a live-in super
- `age_of_building`: The age of the building
- `average_tenant_age`: The average age of the tenants per building
- `monthly_average_rent`: The average monthly rent per building

We'll write the Stan model like:

$$\begin{aligned} \eta_{ib} &= \alpha_{b(i)} + \beta\,\text{traps}_i + \text{log\_sq\_foot}_i \\ \alpha_b &\sim \mathcal{N}(\mu + \texttt{building\_data}\,\zeta, \sigma_\alpha^2) \end{aligned} \tag{6}$$

Model fit in Stan

We fit the model in Stan, at the end we obtain these warnings:

```
Warning messages:
1: There were 915 divergent transitions after warmup.
Increasing adapt_delta above 0.8 may help.
```

What happened? 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. We will return to this issue later...

In this example we will see that we have divergent transitions because we need to reparametrize our model - i.e., we will retain the overall structure of the model, but transform some of the parameters so that it is easier for Stan to sample from the parameter space.

Model fit in Stan

	mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
sigma_alpha	0.25	0.17	0.05	0.13	0.22	0.34	0.69	182	1.03
beta	-0.23	0.06	-0.35	-0.27	-0.22	-0.19	-0.11	715	1.00
mu	1.25	0.42	0.43	0.98	1.22	1.53	2.12	849	1.00
phi	1.54	0.36	0.99	1.29	1.49	1.75	2.38	302	1.01
alpha[1]	1.28	0.54	0.21	0.95	1.24	1.62	2.37	1007	1.00
alpha[2]	1.23	0.52	0.21	0.91	1.20	1.56	2.31	914	1.00
alpha[3]	1.39	0.49	0.51	1.05	1.38	1.71	2.41	397	1.01
alpha[4]	1.43	0.48	0.53	1.09	1.39	1.75	2.42	561	1.00
alpha[5]	1.07	0.42	0.25	0.76	1.08	1.33	1.94	880	1.01
alpha[6]	1.16	0.48	0.22	0.86	1.16	1.45	2.16	914	1.00
alpha[7]	1.43	0.52	0.49	1.07	1.39	1.77	2.51	434	1.01
alpha[8]	1.27	0.42	0.45	1.00	1.29	1.52	2.12	1156	1.00
alpha[9]	1.40	0.55	0.29	1.05	1.41	1.74	2.51	1077	1.00
alpha[10]	0.86	0.37	0.17	0.60	0.85	1.11	1.62	644	1.01

Model fit in Stan

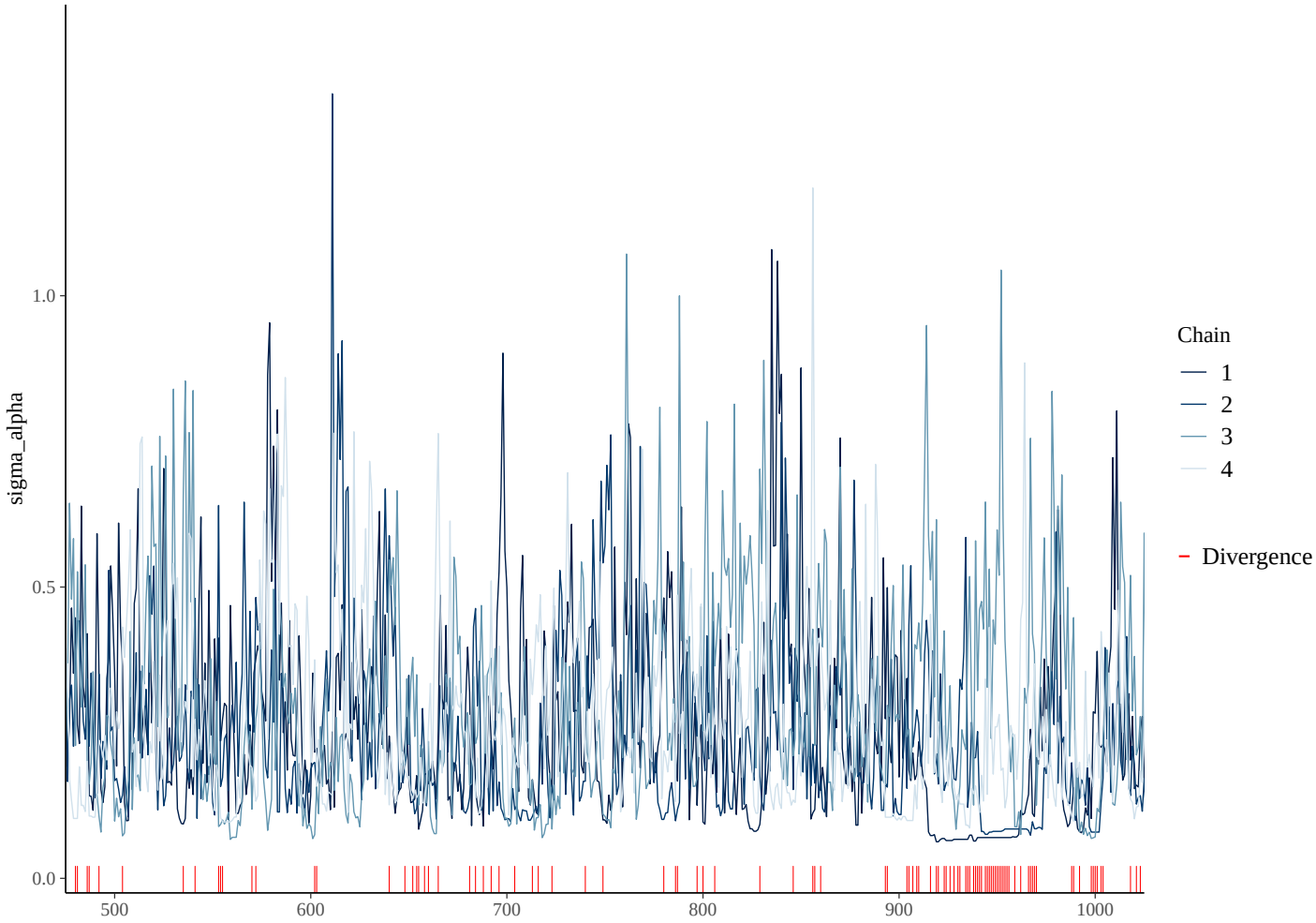
Before we go through exactly how to do this reparameterization, we will first go through what indicates that this is something that reparameterization will resolve. We will go through:

1. Examining the fitted parameter values, including the effective sample size
2. Traceplots and scatterplots that reveal particular patterns in locations of the divergences.

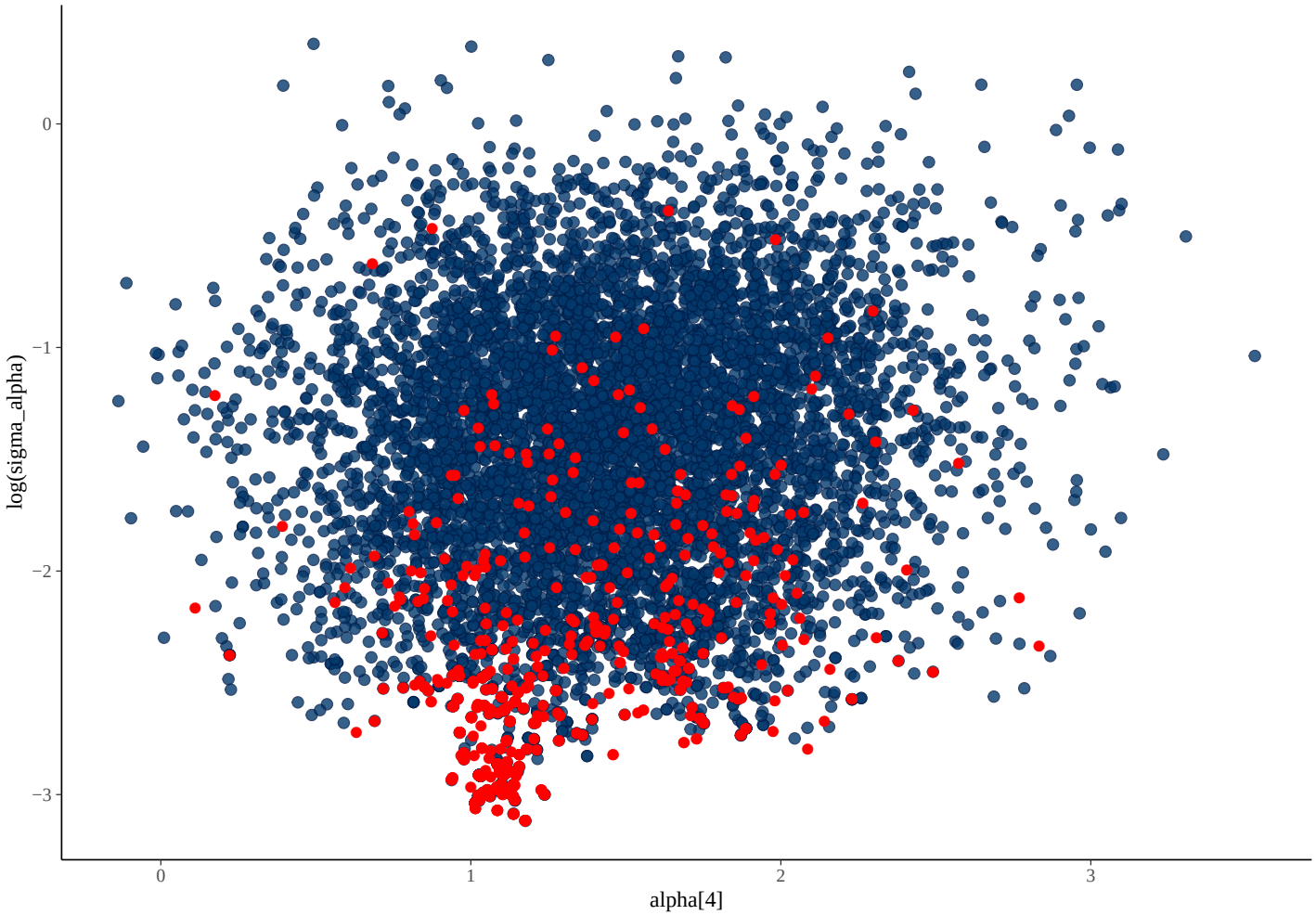
The effective samples are quite low for many of the parameters relative to the total number of samples. This alone isn't indicative of the need to reparameterize, but it indicates that we should look further at the trace plots and pairs plots.

Model fit in Stan

First let's look at the traceplots to see if the divergent transitions form a pattern.

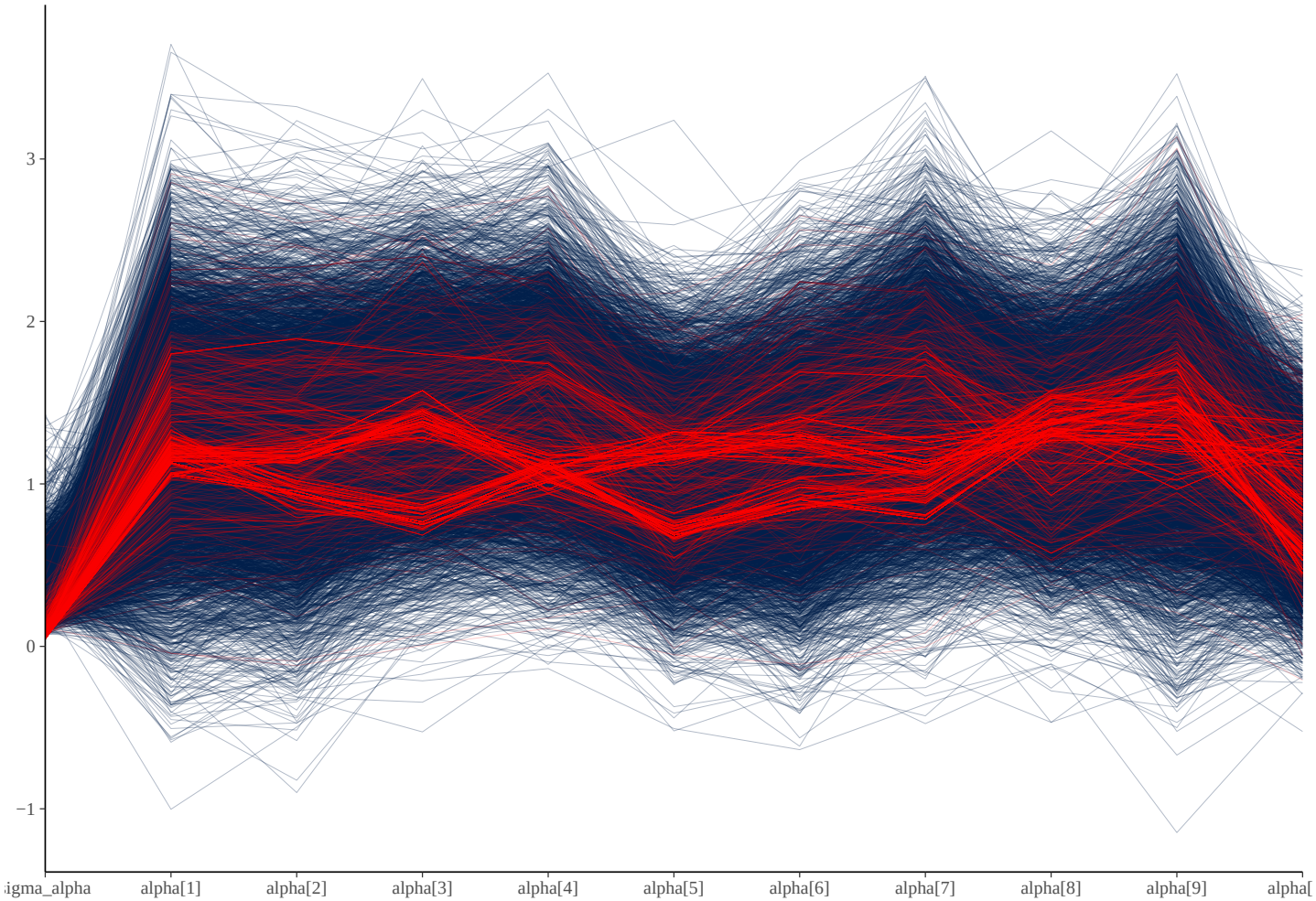


Model fit in Stan



Model fit in Stan

Another way to look at the divergences is via a parallel coordinates plot:



Model fit in Stan

Comments:

- Looks as if the divergent parameters, the little red bars underneath the traceplots correspond to samples where the sampler gets stuck at one parameter value for σ_α.
- What we have in the scatterplot, is a cloud-like shape, with most of the divergences clustering towards the bottom. We'll see a bit later that we actually want this to look more like a funnel than a cloud, but the divergences are indicating that the sampler can't explore the narrowing neck of the funnel.
- From the parallel plot, again, we see evidence that our problems concentrate when σ_α is small.

Model fit in Stan: non-centered parametrization

CENTERED

$$\eta_{ib} = \alpha_{b(i)} + \beta x_i$$
$$\alpha_b \sim \mathcal{N}(\mu + \zeta z_b, \sigma_\alpha^2)$$

NON-CENTERED

$$\eta_{ib} = \alpha_{b(i)} + \beta x_i$$
$$\alpha_b = \mu + \zeta z_b + \sigma_\alpha \tilde{\alpha}_b$$
$$\tilde{\alpha}_b \sim \mathcal{N}(0, 1)$$

We should use the non-centered parameterization for α_b. We define a vector of auxiliary variables in the parameters block, `alpha_raw` that is given a $\mathcal{N}(0, 1)$ prior in the model block. We then make `alpha` a transformed parameter. We can reparameterize the random intercept α_b, which is distributed:

$$\alpha_b \sim \mathcal{N}(\mu + \texttt{building\_data}\,\zeta, \sigma_\alpha^2)$$

Model fit in Stan: non-centered parametrization

In the `transformed parameters` block we define now:

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

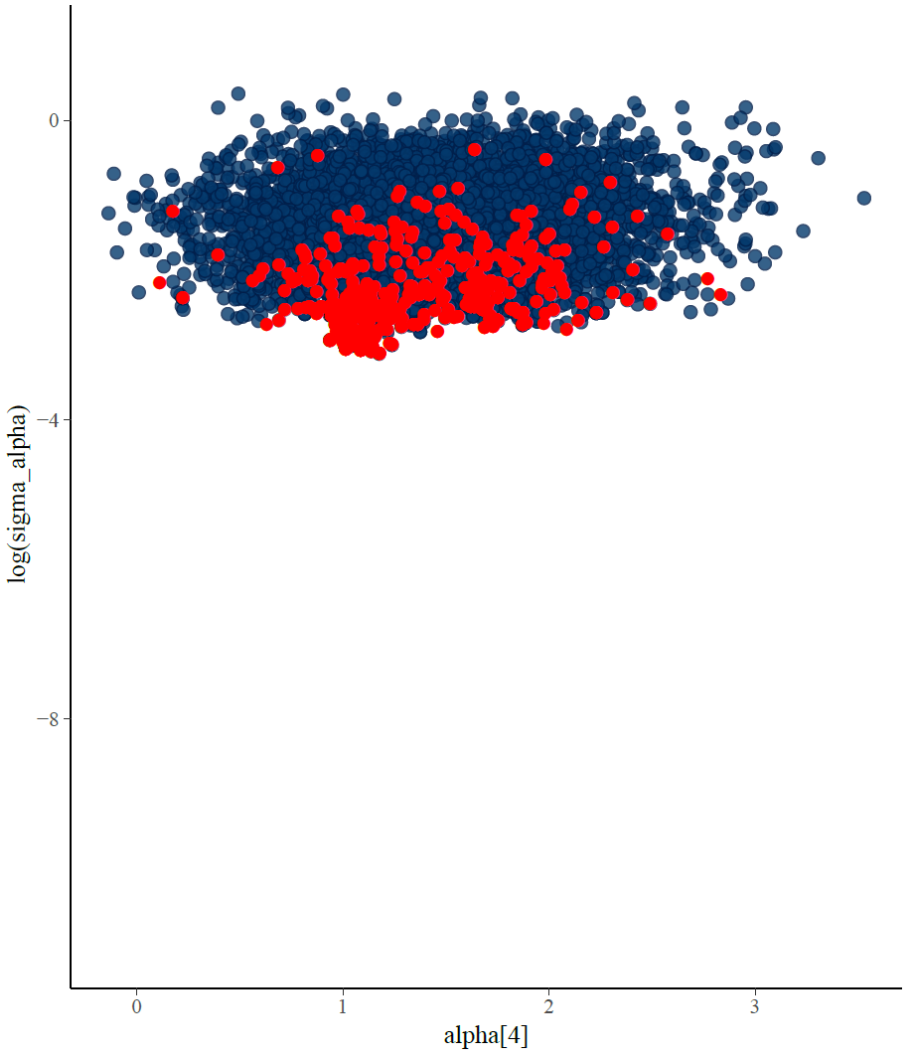
This gives `alpha` a $\mathcal{N}(\mu + \texttt{building\_data}\,\zeta, \sigma_\alpha^2)$ distribution, but it decouples the dependence of the density of each element of `alpha` from `sigma_alpha` (σ_α).

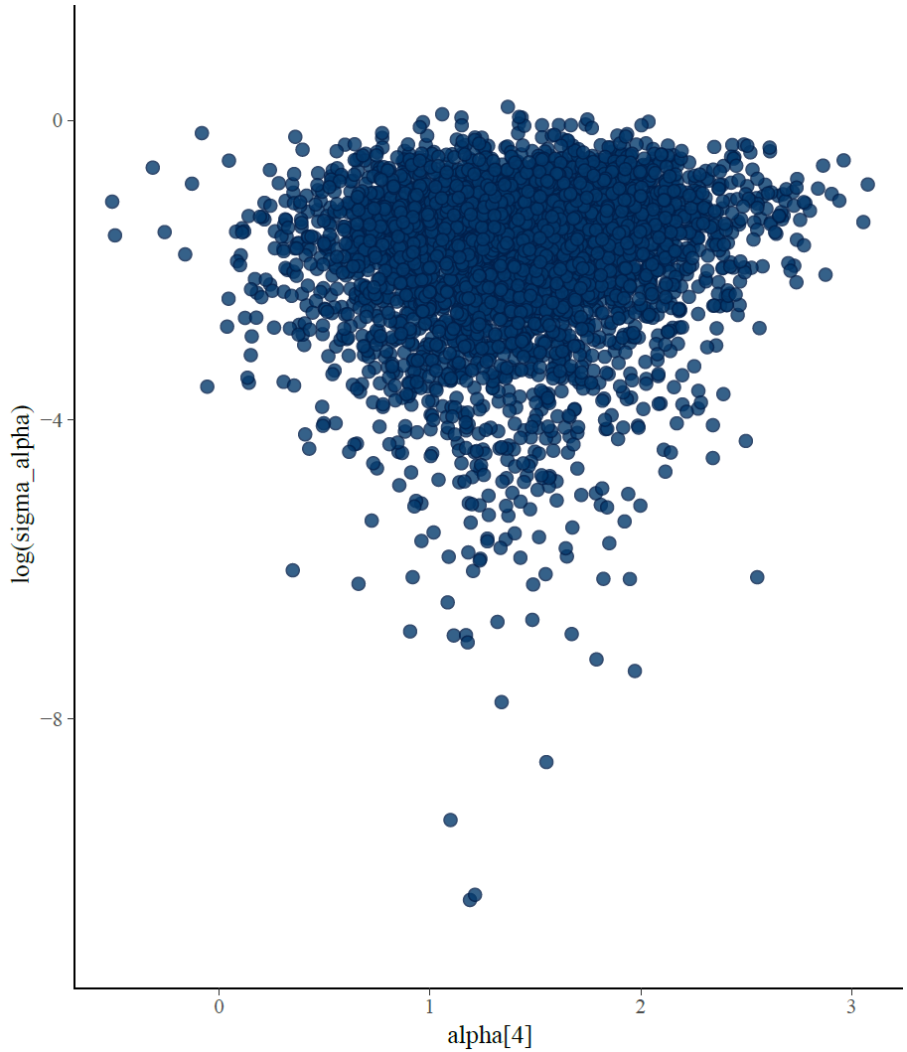
We fit this new model version in Stan. We will examine the effective sample size of the fitted model to see whether we've fixed the problem with our reparameterization.

Model fit in Stan: non-centered parametrization

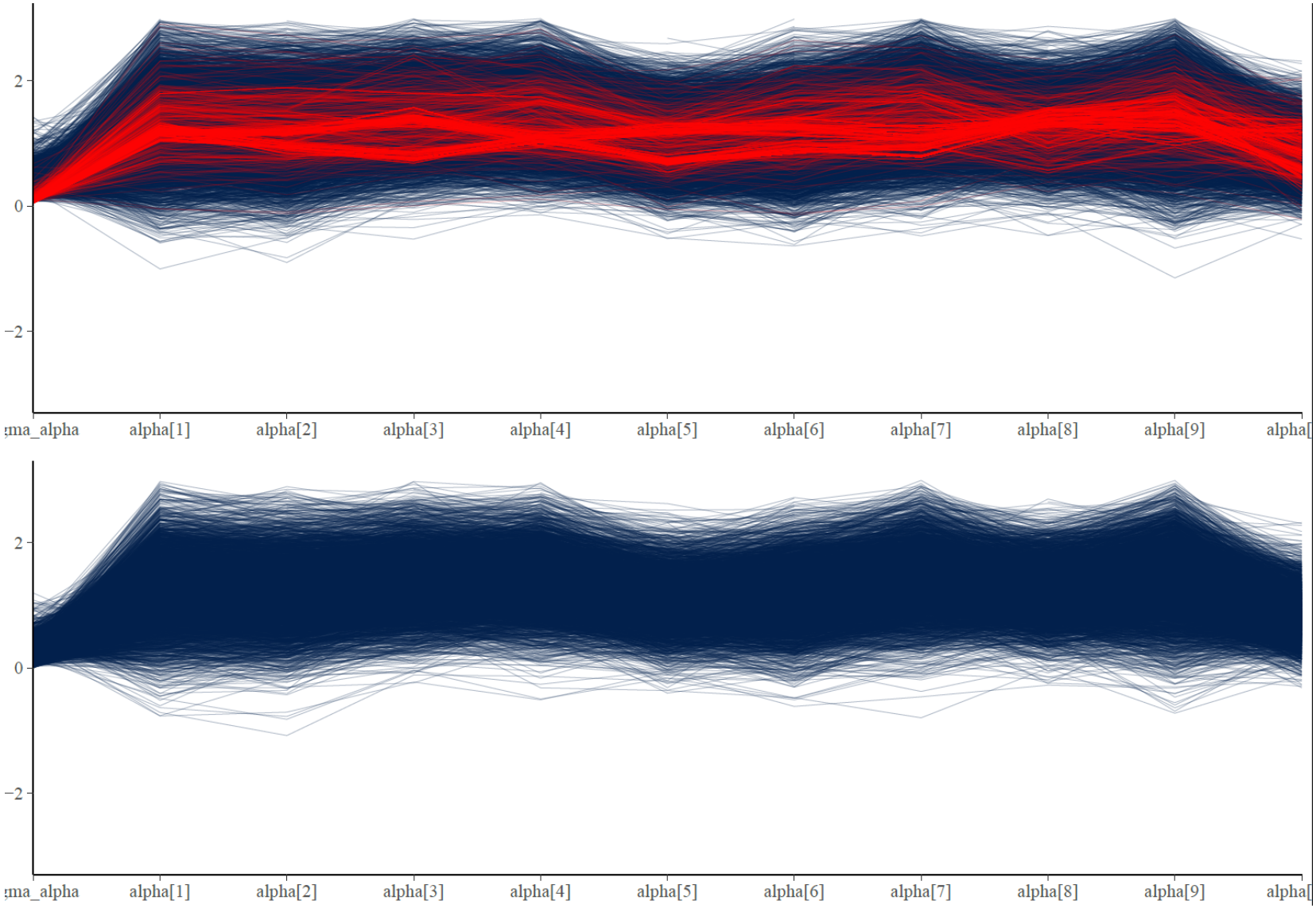
	mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
sigma_alpha	0.23	0.17	0.01	0.10	0.20	0.32	0.63	1447	1
beta	-0.23	0.06	-0.35	-0.27	-0.23	-0.19	-0.11	2649	1
mu	1.25	0.44	0.40	0.95	1.24	1.54	2.12	2555	1
phi	1.58	0.34	1.03	1.34	1.54	1.77	2.35	4256	1
alpha[1]	1.27	0.56	0.15	0.90	1.27	1.64	2.37	2566	1
alpha[2]	1.21	0.53	0.19	0.86	1.21	1.56	2.28	2551	1
alpha[3]	1.38	0.49	0.42	1.05	1.38	1.71	2.38	2672	1
alpha[4]	1.42	0.49	0.46	1.08	1.42	1.74	2.39	2783	1
alpha[5]	1.08	0.42	0.26	0.81	1.07	1.34	1.92	3162	1
alpha[6]	1.17	0.49	0.22	0.85	1.17	1.49	2.12	2502	1
alpha[7]	1.45	0.52	0.42	1.10	1.44	1.79	2.49	2996	1
alpha[8]	1.23	0.43	0.40	0.94	1.23	1.52	2.10	3481	1
alpha[9]	1.41	0.58	0.25	1.03	1.42	1.80	2.51	2780	1
alpha[10]	0.86	0.37	0.17	0.61	0.85	1.11	1.60	3417	1

Model fit in Stan: centered vs non-centered parametrization





Model fit in Stan: centered vs non-centered parametrization

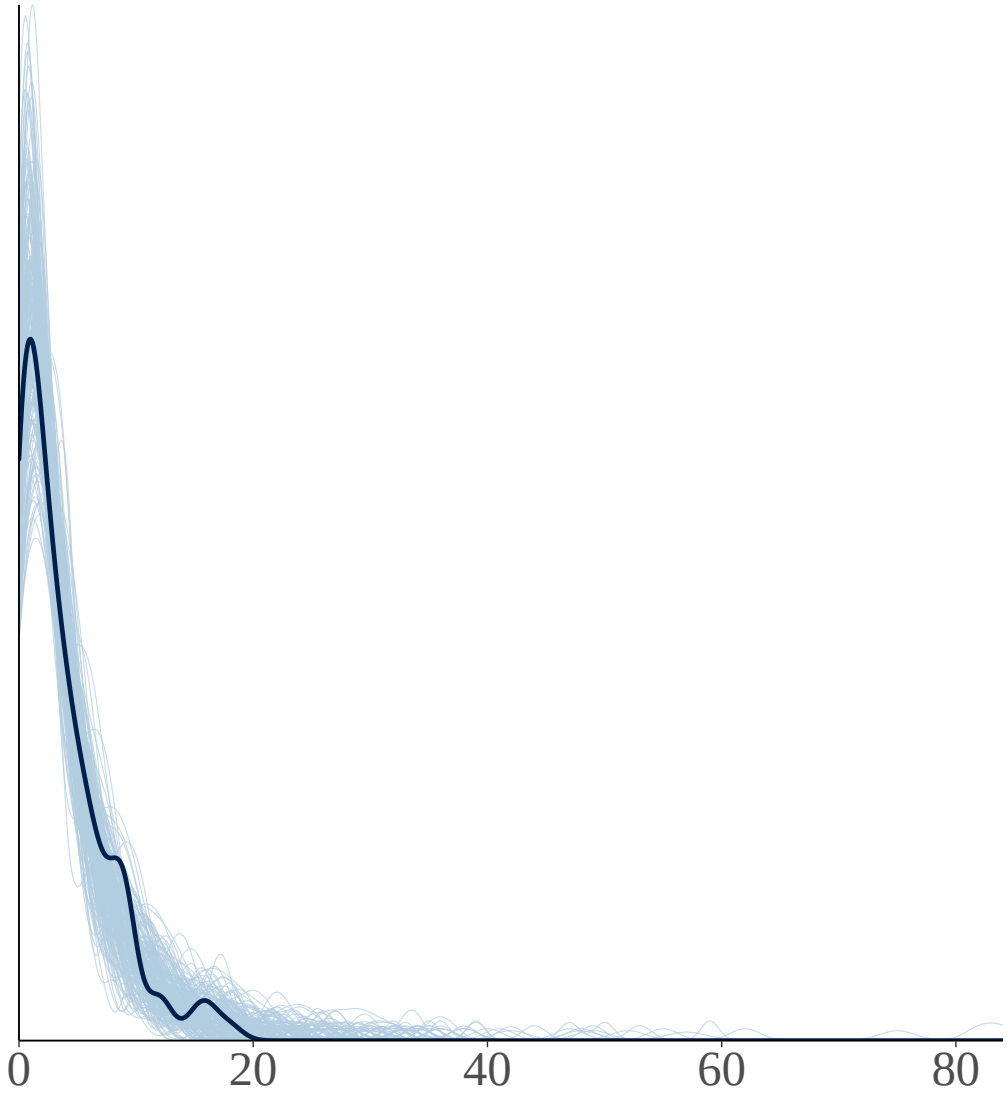


Model fit in Stan: centered vs non-centered parametrization

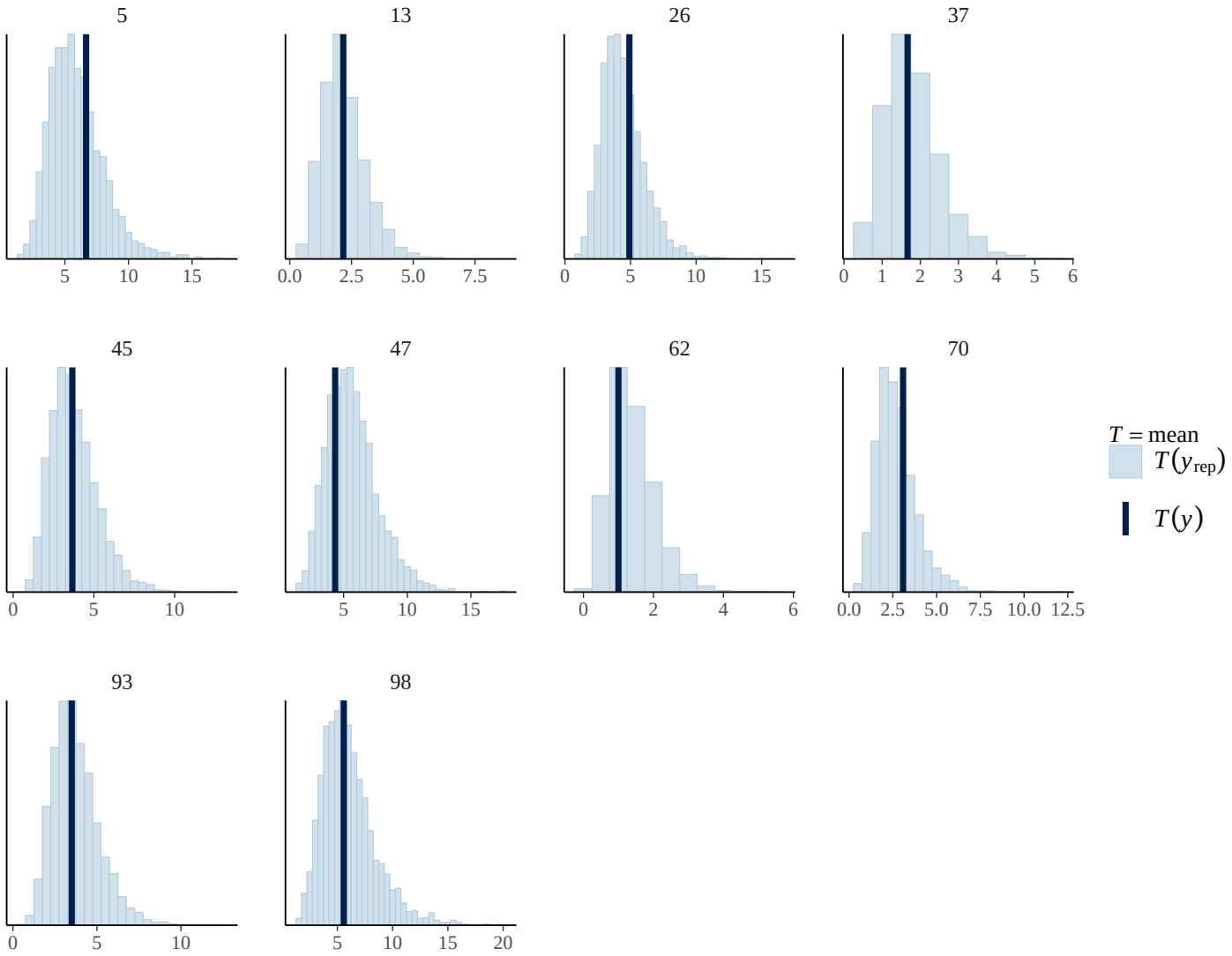
Comments:

- This has improved the effective sample sizes of α.
- No more divergent transitions!

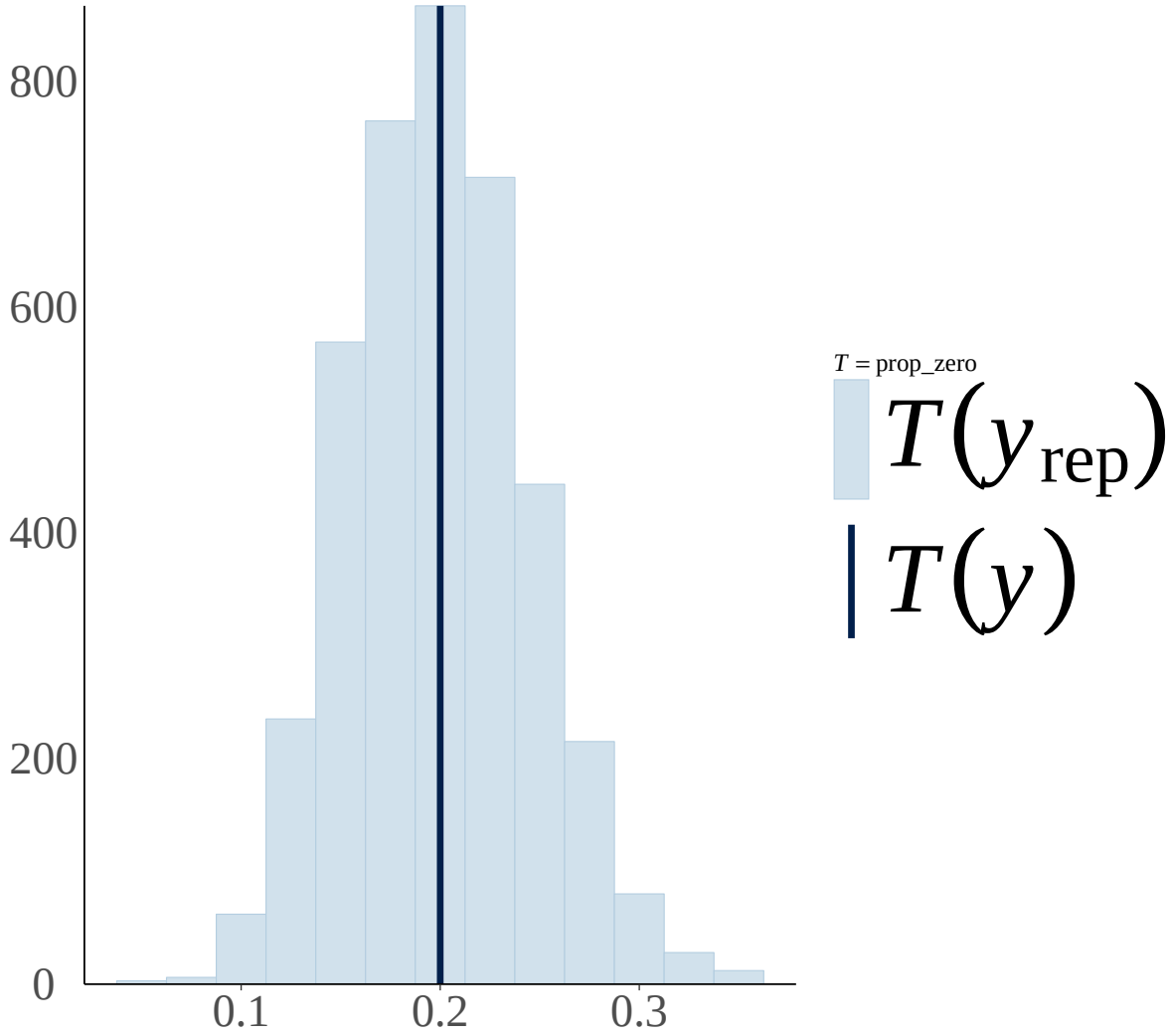
Pest model, hierarchical NB ncp model. PP check: densities



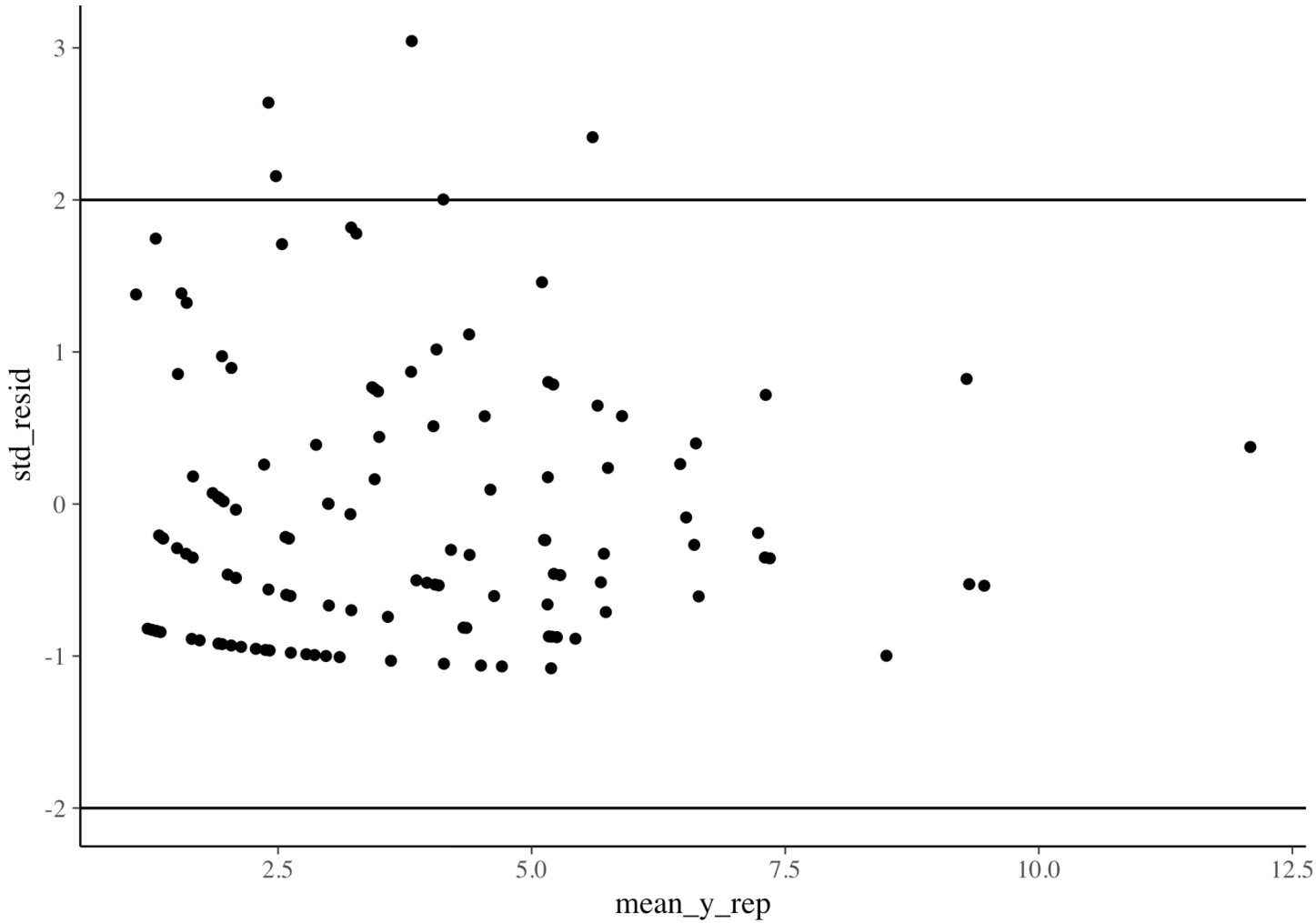
Pest model, hierarchical NB ncp model. PP check: statistics



Pest model, hier NB ncp model. PP check: prop. of zeros



Pest control, hier NB ncp model: pp check. Residuals



Better!

Further readings

Further reading:

- Chapter 6 from *BDA*, A. Gelman et al. (model checking)
- Chapter 20 from the Stan Users Guide (reparametrization)