

BDA Course 2026

Lecture 8

Objectives of this Lecture

- Understand the role of model checking in applied Bayesian statistics
- Discuss posterior predictive checks
- Leave-one-out cross-validation
- PSIS-LOO_CV (Fast cross-validation)
- Model comparison and selection

The map is not the territory

- Models are simplified representations of reality
 - They can only capture certain aspects of reality and ignore others
 - Often details are omitted intentionally to keep the model simple and tractable
- Model checking helps us understand the limitations of our model and if they are adequate for our purposes

Model checking

- External validation
 - compare to domain knowledge not used in modeling
 - compare predictions to completely new observations
- Internal validation
 - Compare predictions to the observed data
 - posterior predictive checking
 - cross-validation predictive checking

Posterior predictive distribution

- Distribution of a new observation $\tilde{y}$ given the observed data:

$$p(\tilde{y} \mid y) = \int p(\tilde{y} \mid \theta) p(\theta \mid y) d\theta$$

- Computationally, for $s = 1, \dots, S$:
 - draw $\theta^{(s)}$ from $p(\theta \mid y)$
 - draw $\tilde{y}^{(s)}$ from $p(\tilde{y} \mid \theta^{(s)})$
- $\tilde{y}$: any new, not yet observed data we want to predict (any size, possibly new x values)
- y^{rep} : a replication of the original experiment, i.e. the same x values and the same number of observations as y , used for model checking

Posterior predictive checking

- Compute predictions y^{rep} and compare them to the observed data y
- Assess how well the model captures the data, where it succeeds and where it fails
- If necessary, revise data, model or assumptions and repeat

Posterior predictive checking

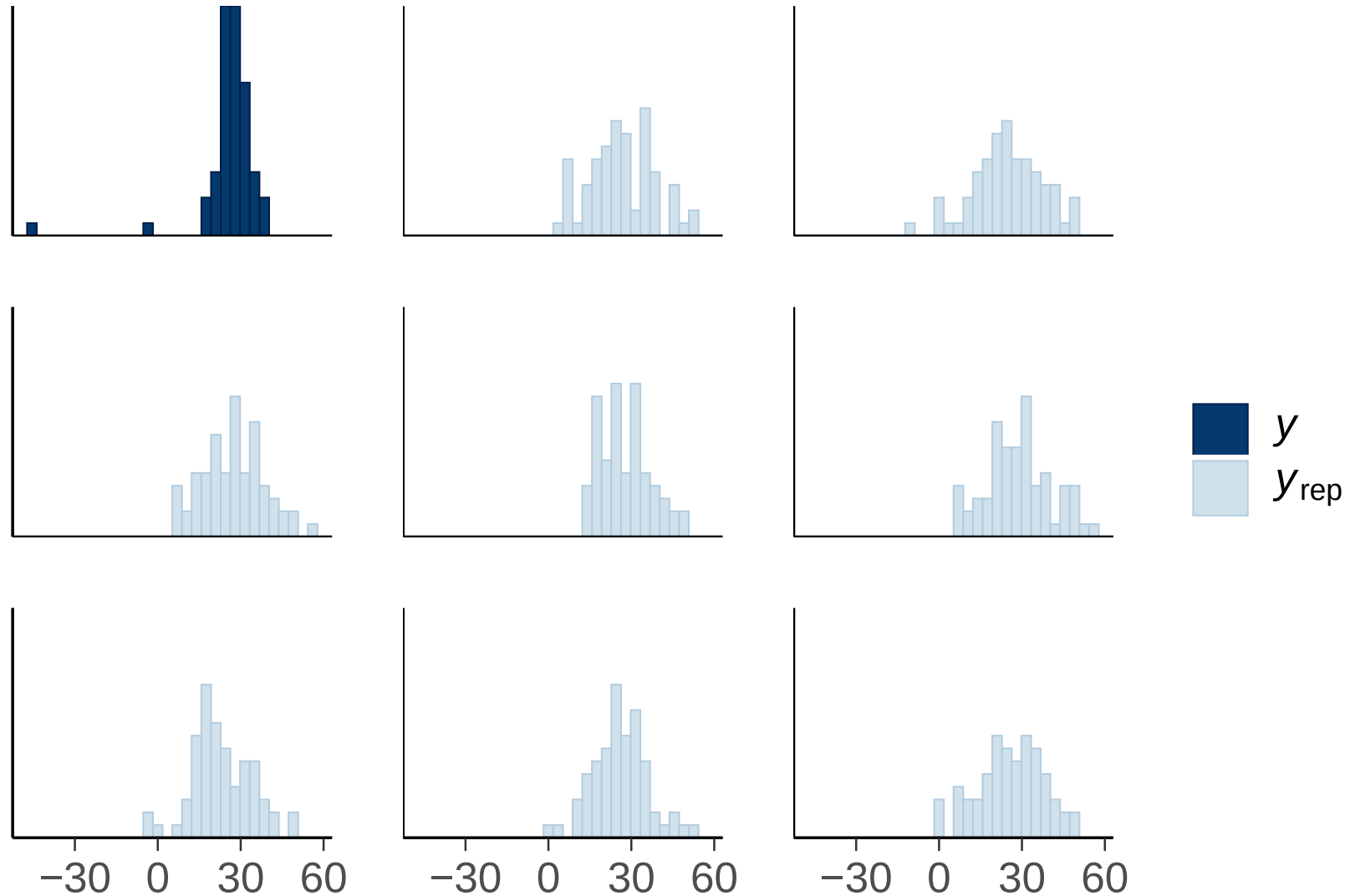
- This is a very general idea that can be adapted to different models and data
- There are some standard checks (often visual) and also recommendations for good practices
 - Gabry, Simpson, Vehtari, Betancourt, and Gelman (2019). Visualization in Bayesian workflow. <https://doi.org/10.1111/rssa.12378>.
 - Säilynoja, Johnson, Martin, and Vehtari (2026). Recommendations for visual predictive checks in Bayesian workflow. <https://journals.aau.dk/index.php/jovi/article/view/11478>
 - EABM: predictive checks https://arviz-devs.github.io/EABM/Chapters/Prior_posterior_predictive_checks.html
 - Graphical posterior predictive checks:
 - Bayesplot <http://mc-stan.org/bayesplot/articles/graphical-ppcs.html>
 - ArviZ <https://python.arviz.org/projects/plots/en/latest/gallery/index.html#predictive-checks>
 - brms demos https://avehtari.github.io/BDA_R_demos/demos_rstan/brms_demo.html

Simon Newcomb's light of speed experiment in 1882

Newcomb measured ($n = 66$) the time required for light to travel from his laboratory on the Potomac River to a mirror at the base of the Washington Monument and back, a total distance of 7422 meters.

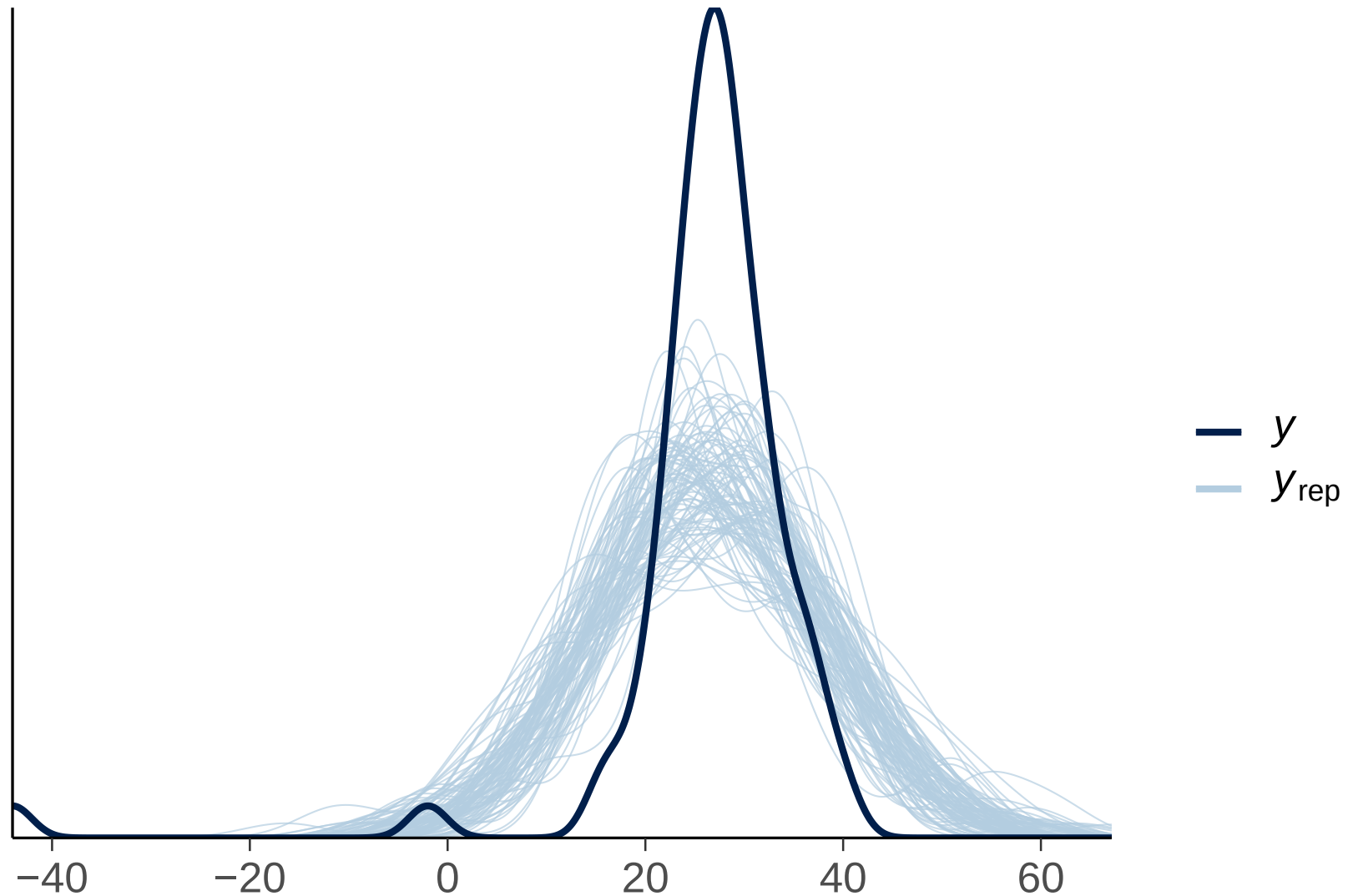
Posterior predictive checking – bayesplot

```
ppc_hist(y, yrep[1:8,])
```



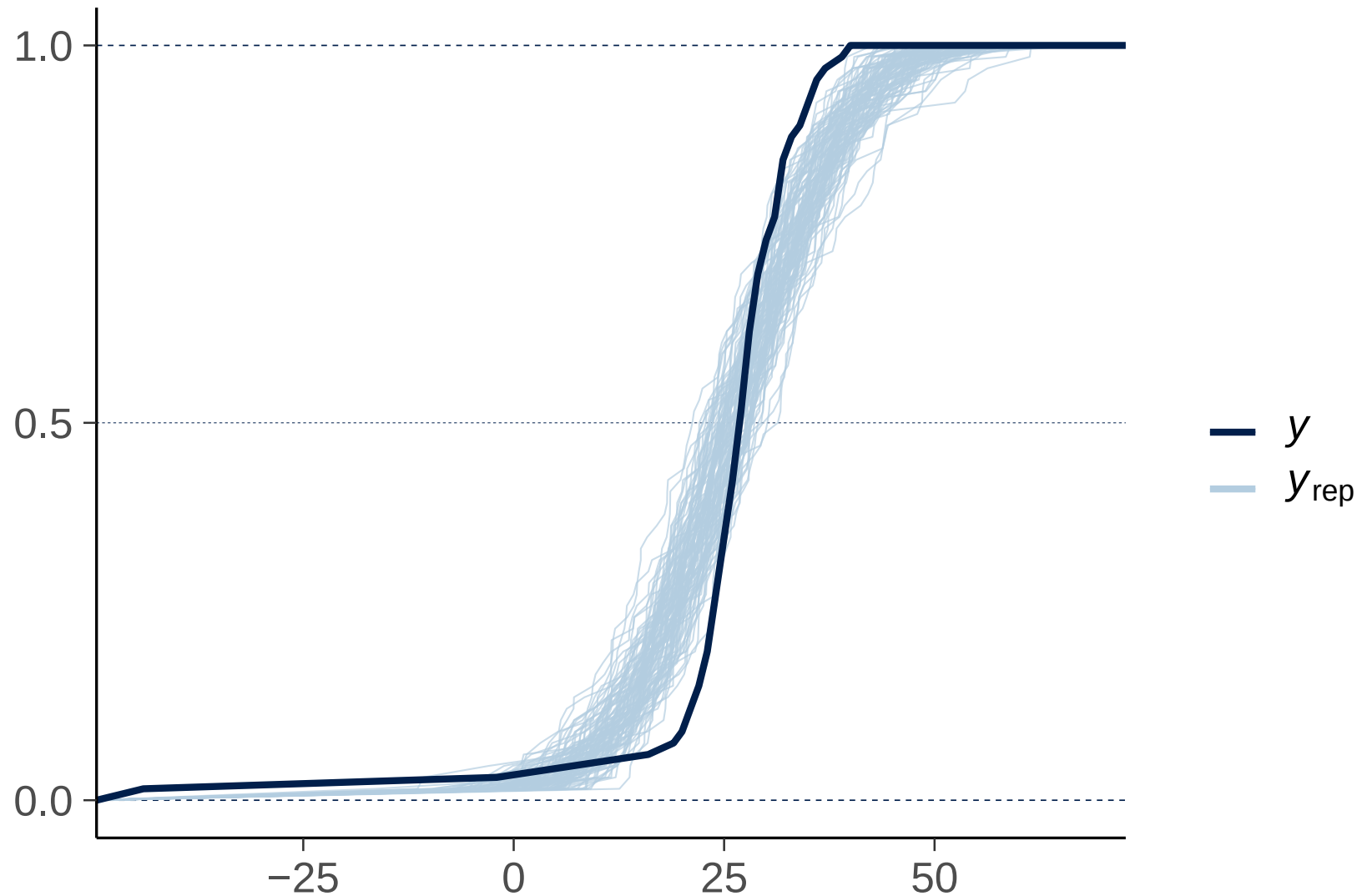
Posterior predictive checking – bayesplot

```
ppc_dens_overlay(y, yrep[1:100,])
```



Posterior predictive checking – bayesplot

```
ppc_ecdf_overlay(y, yrep[1:100,])
```



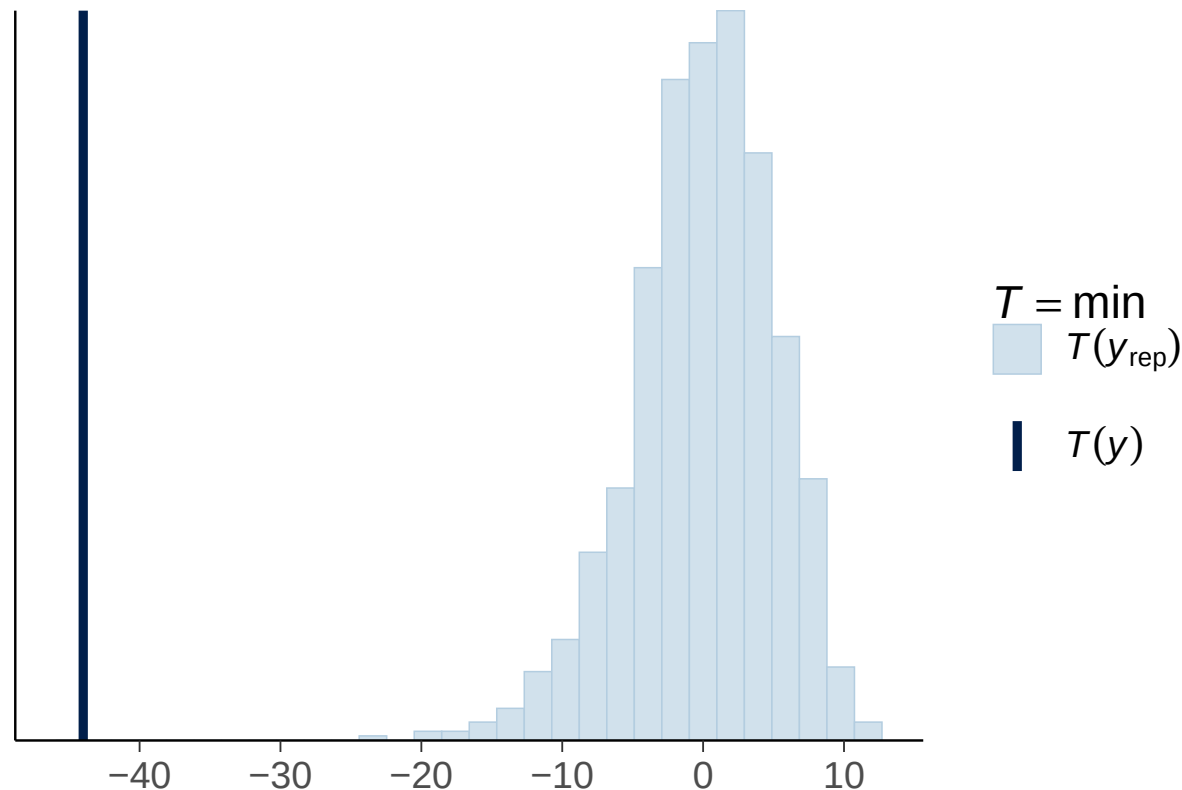
Posterior predictive checking with test statistic

- Choose a test statistic T that summarizes the aspect of the data you want to check
- compute summary quantity for the observed data $T(y, \theta)$
- compute summary quantity for a replicated data $T(y^{\text{rep}}, \theta)$

Posterior predictive checking with test statistic

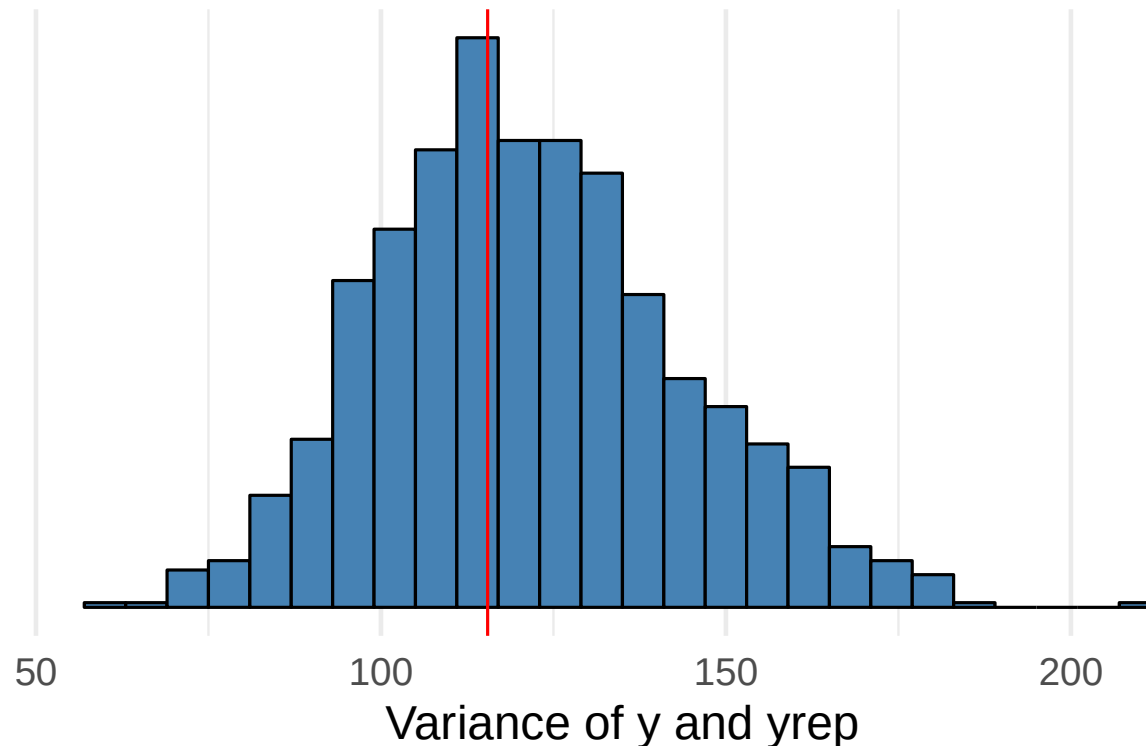
- Compute test statistic for data $T(y, \theta) = \min(y)$
- Compute test statistic $\min(y^{\text{rep}})$ for many replicated datasets

```
ppc_stat(y, yrep, stat = "min")
```



Posterior predictive checking – example

- Good test statistic is (almost) *ancillary* (fi: *aputunnusluku*)
 - ancillary = distribution of $T(y)$ does not depend on the model parameters θ (like the minimum in a normal model)
- Bad test statistic is highly dependent on the parameters
 - e.g. mean or variance for normal model



Posterior predictive p -value

- Probability that replicated data are at least as extreme as observed data

$$p = \Pr(T(y^{\text{rep}}, \theta) \geq T(y, \theta) \mid y)$$

- Discussed in BDA3, but not commonly used, as
 - not calibrated for non-ancillary test statistics
 - the full distribution of the test statistic contains more information than a single number

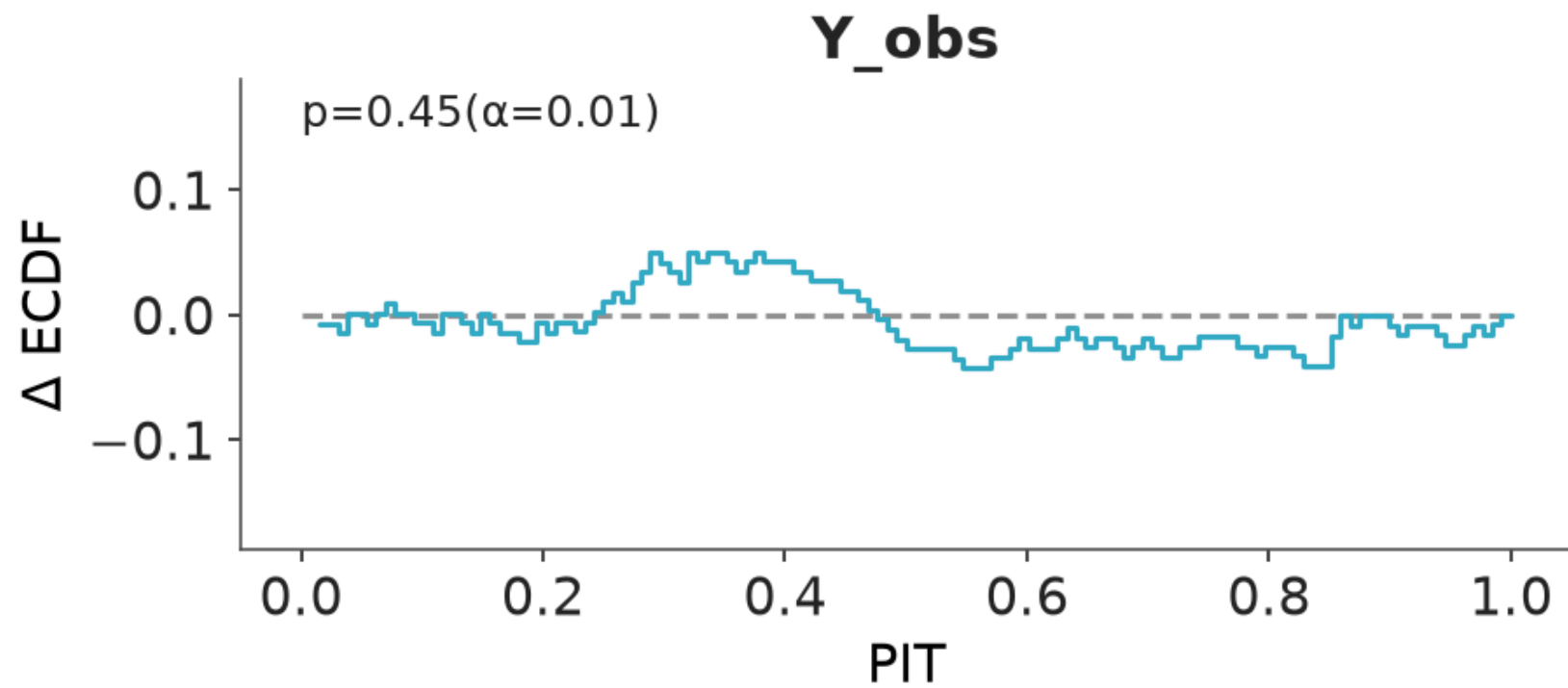
Posterior predictive checking with PIT

- Instead of one p -value for a test statistic, check calibration of each observation
- Posterior predictive PIT (probability integral transform) for y_i

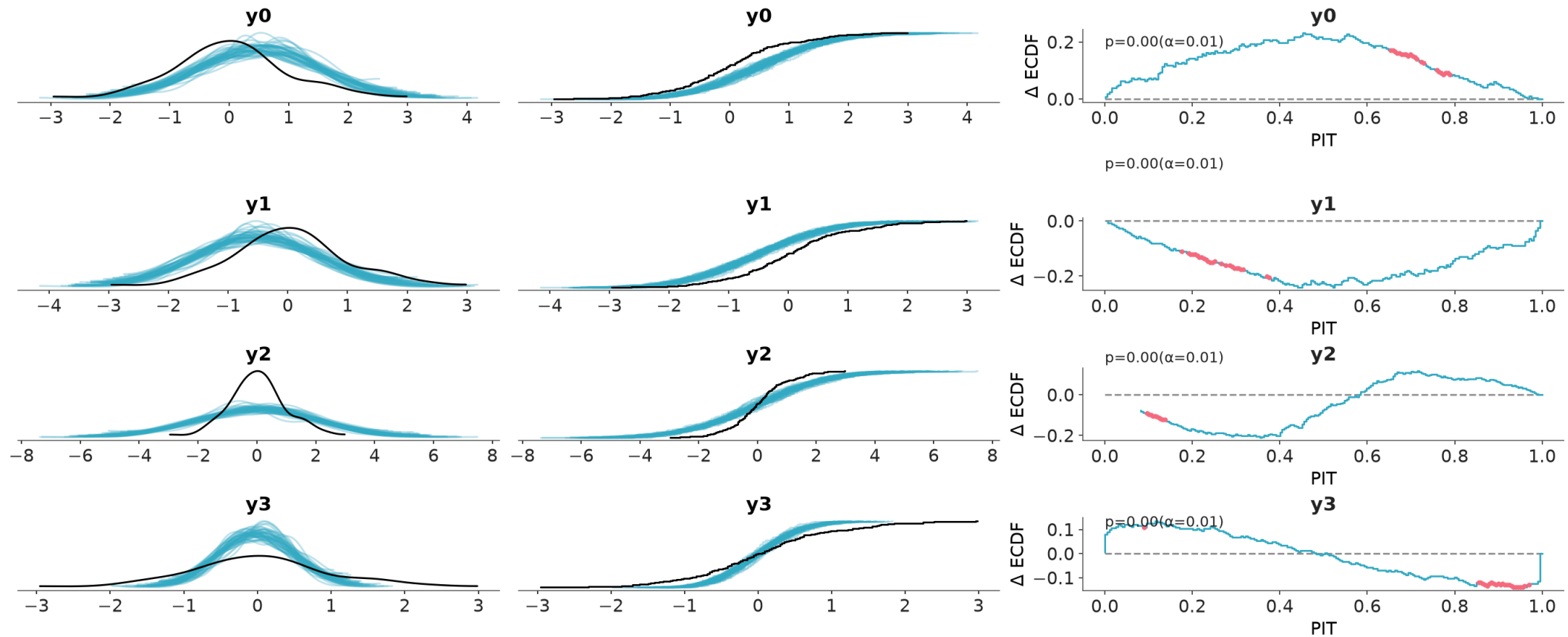
$$\text{PIT}_i = \Pr(y_i^{\text{rep}} \leq y_i \mid y)$$

- If the model is calibrated, PIT_i are (approximately) uniformly distributed
 - the test statistic is $T(y_i^{\text{rep}}) = y_i^{\text{rep}}$, so no need to choose one
 - discrete y : use randomized PIT

PPC PIT – example



PPC PIT – interpretation



Posterior predictive samples – what you need

- Stan: generated quantities block with `_rng` functions
- rstanarm, brms: `posterior_predict(fit)`
- PyMC: `pm.sample_posterior_predictive(idata)`
- bambi: `model.predict(idata, kind="response")`

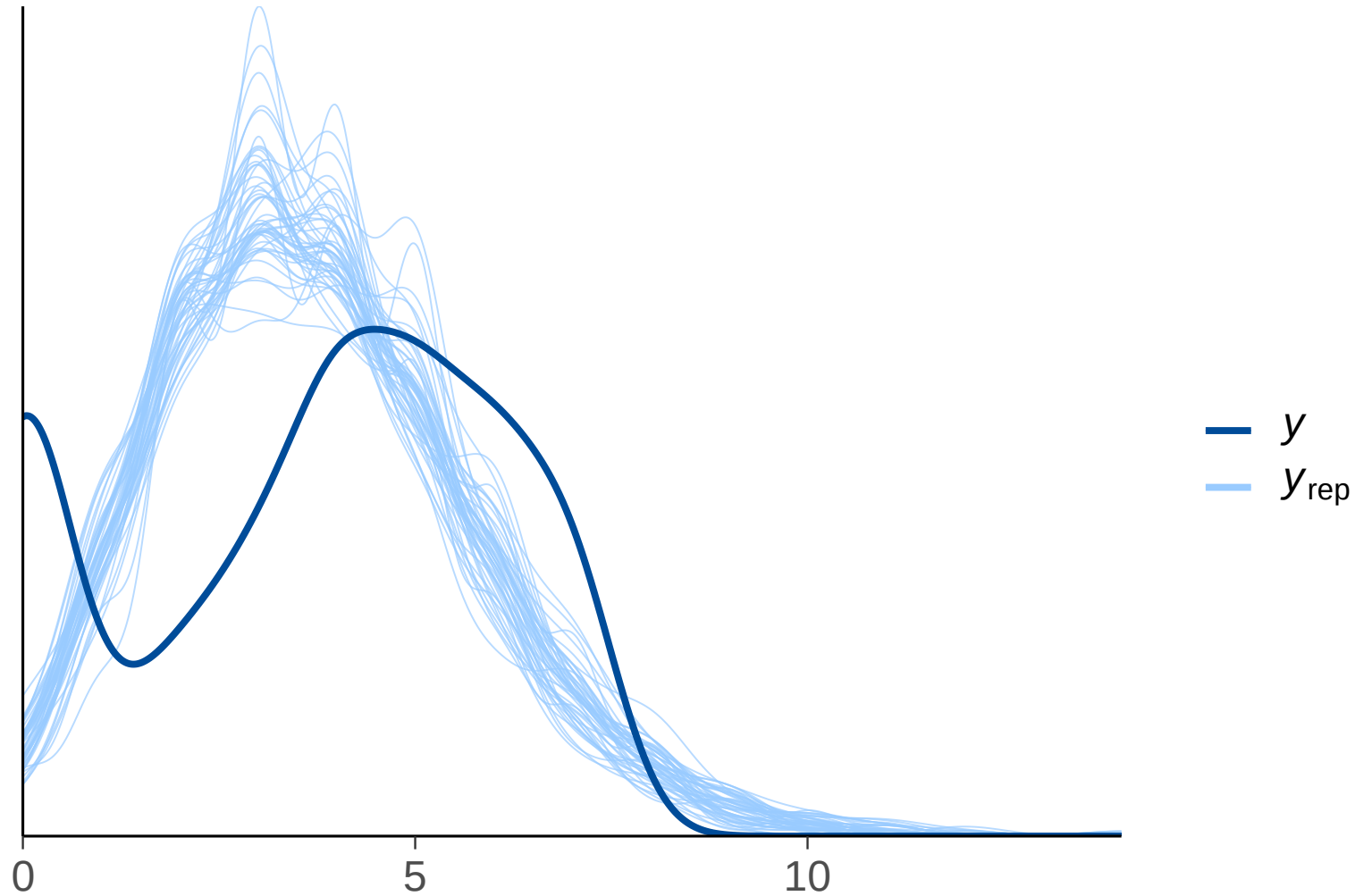
Posterior predictive checking – Stan code

- demo demos_rstan/ppc/poisson-ppc.Rmd

```
data {  
  int<lower=1> N;  
  array[N] int<lower=0> y;  
}  
parameters {  
  real<lower=0> lambda;  
}  
model {  
  lambda ~ exponential(0.2);  
  y ~ poisson(lambda);  
}  
  
generated quantities {  
  real log_lik[N];  
  array[N] int y_rep;  
  for (n in 1:N) {  
    y_rep[n] = poisson_rng(lambda);  
    log_lik[n] = poisson_lpmf(y[n] | lambda);  
  }  
}
```

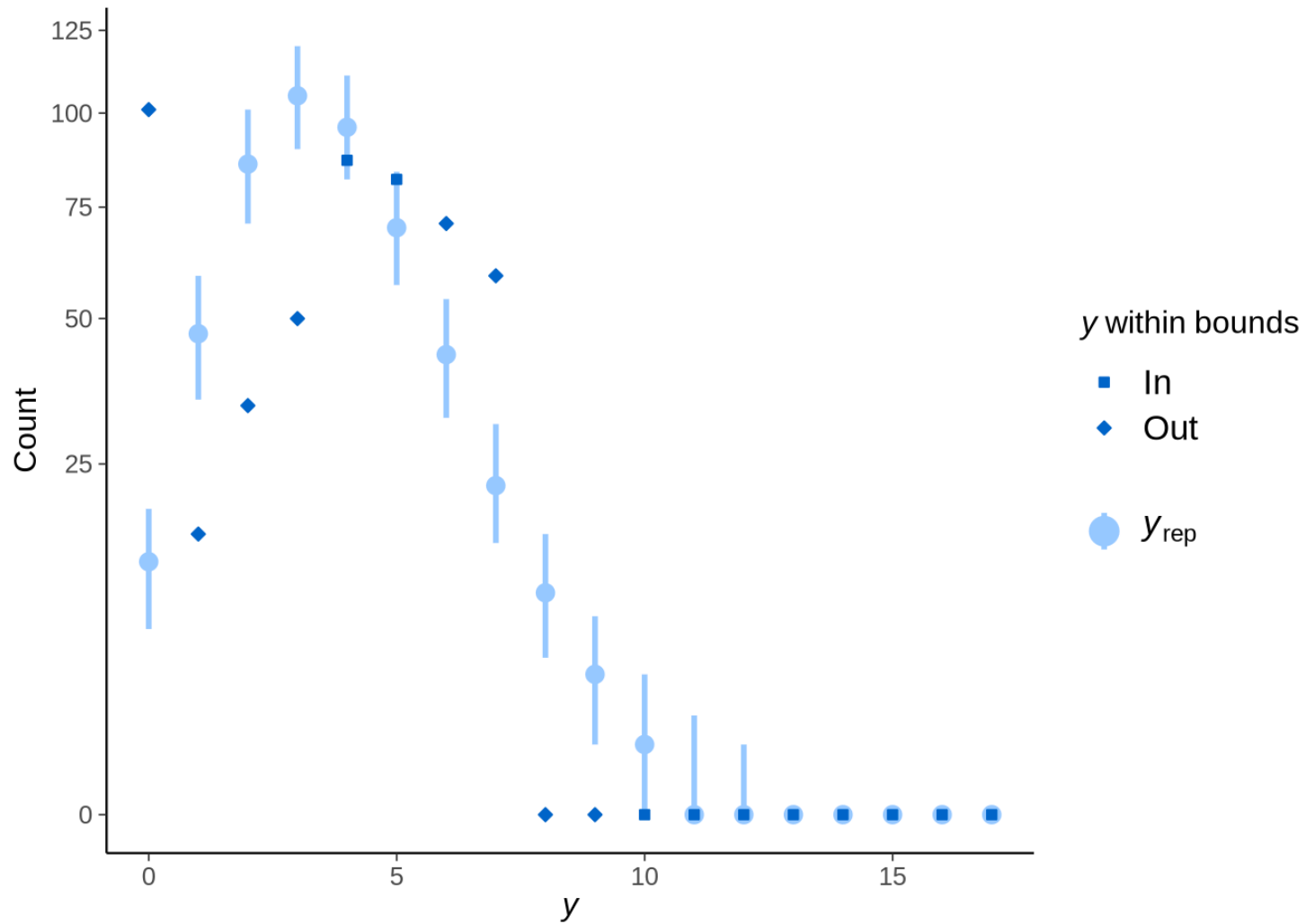
PPC for count data – Poisson model

```
ppc_dens_overlay(y, yrep[1:50,])
```



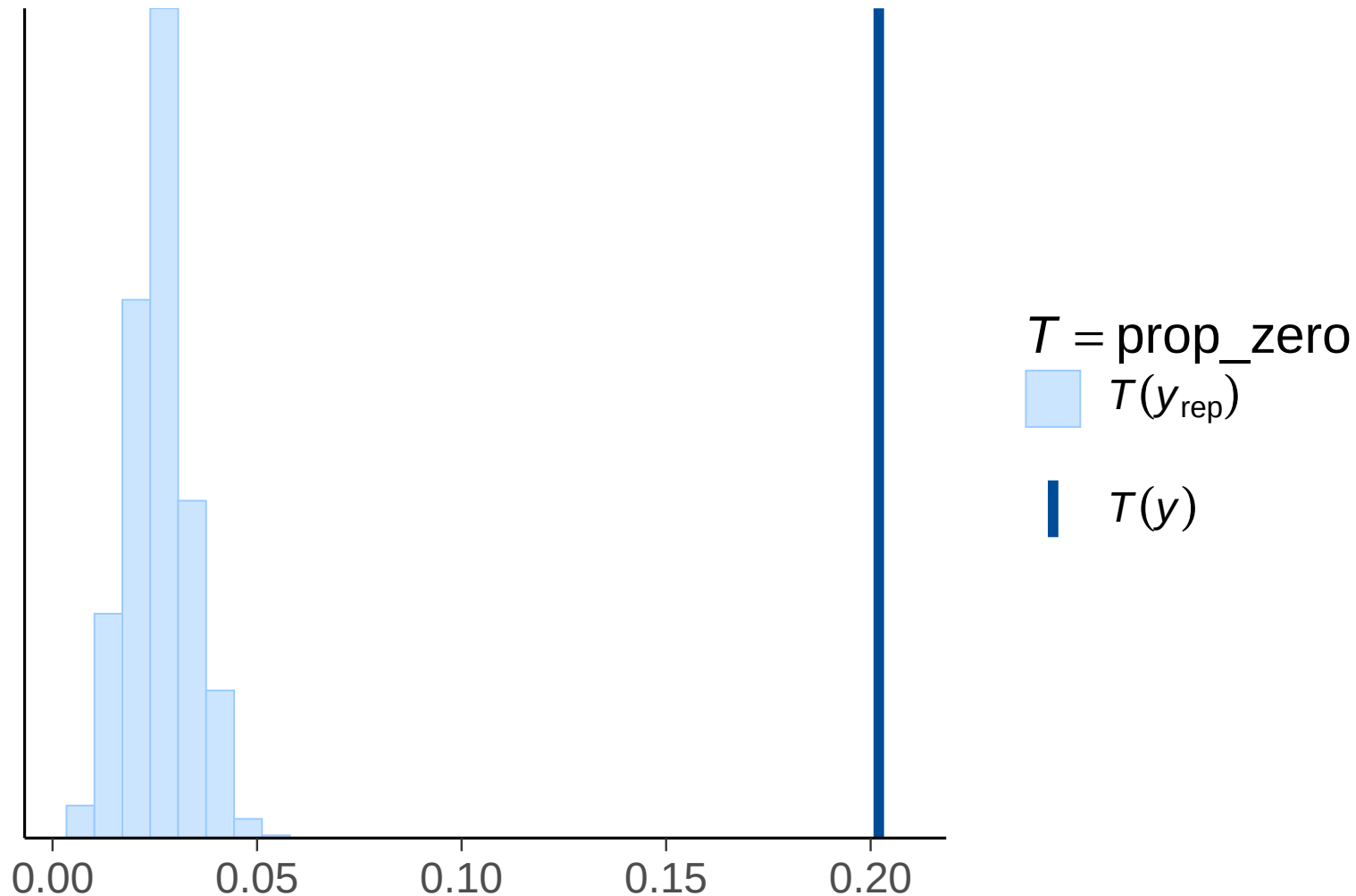
PPC for count data – Poisson model

```
ppc_rootogram(y, yrep, style="discrete")
```



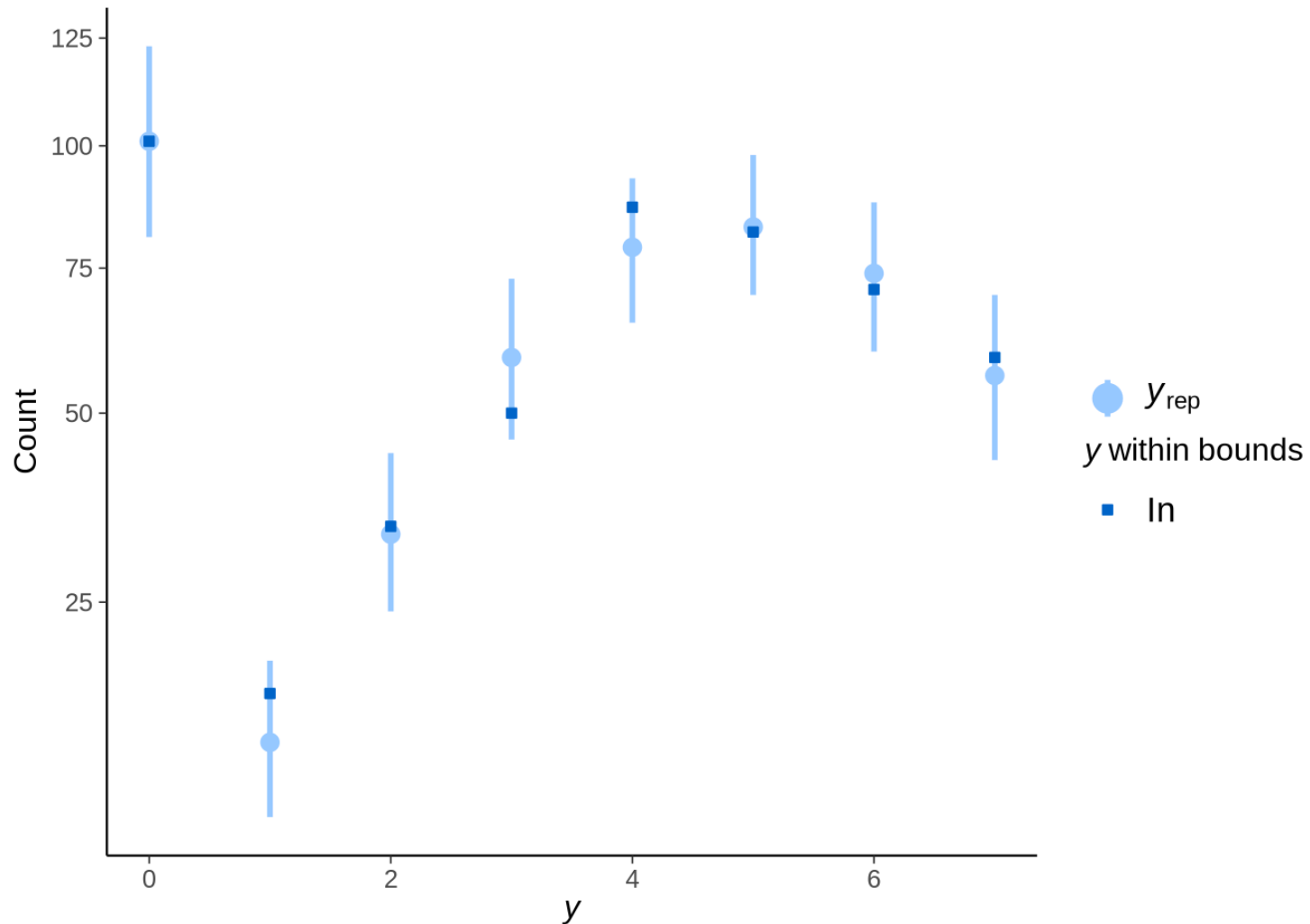
PPC for count data – Poisson model

```
prop_zero <- function(x) mean(x == 0)  
ppc_stat(y, yrep, stat = "prop_zero")
```



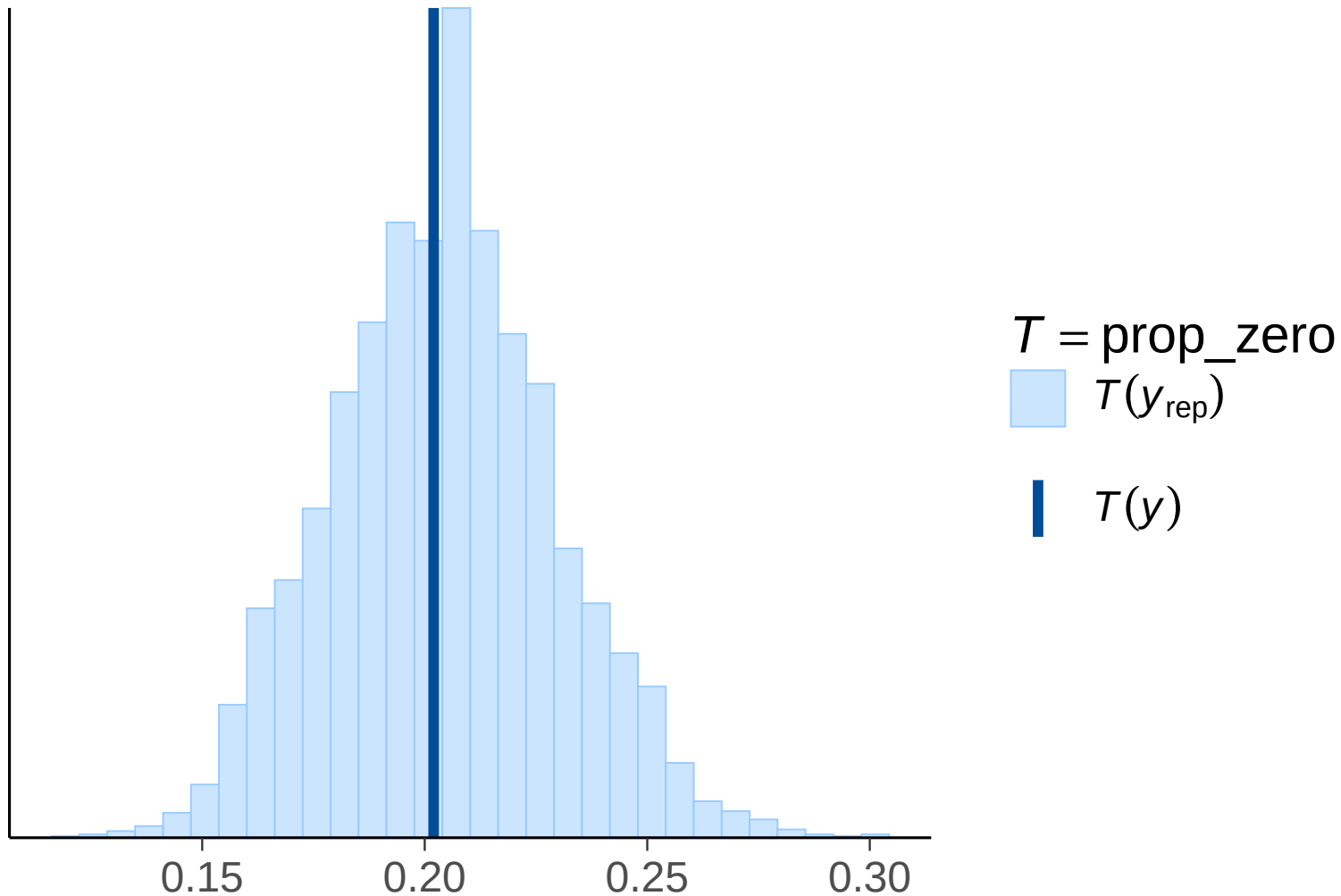
PPC for count data – hurdle truncated Poisson model

```
ppc_rootogram(y, yrep2, style="discrete")
```



PPC for count data – hurdle truncated Poisson model

```
prop_zero <- function(x) mean(x == 0)  
ppc_stat(y, yrep2, stat = "prop_zero")
```

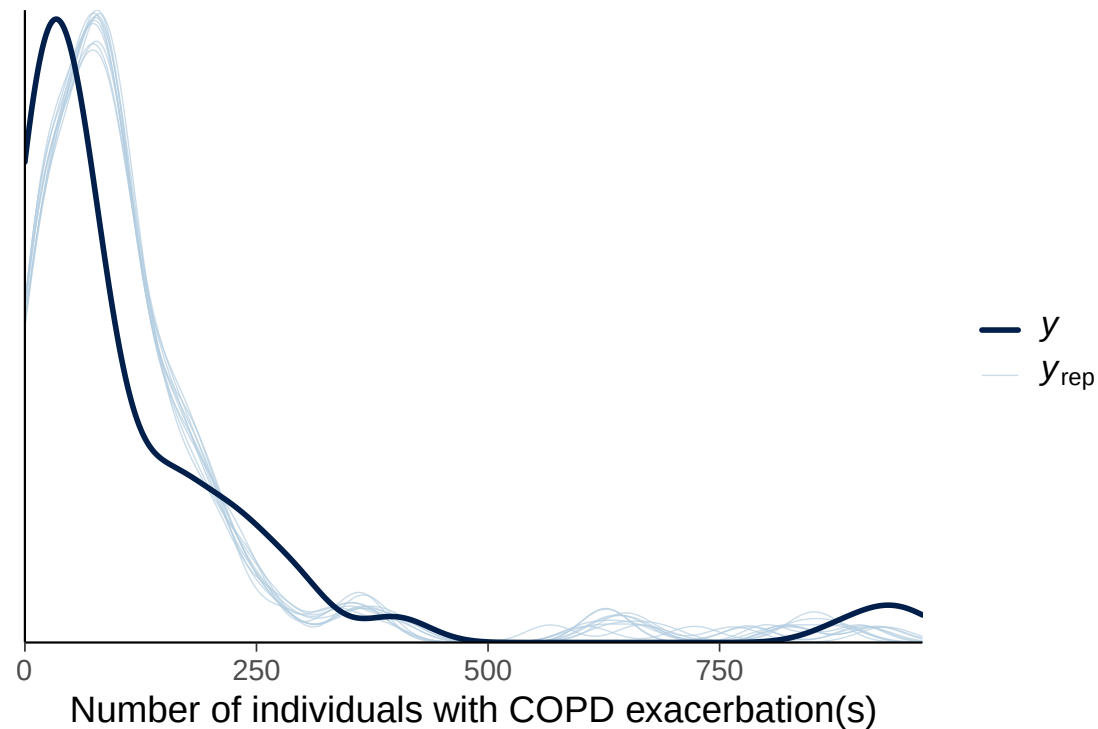


Posterior predictive checking

meta-analysis: Pharmacologic treatments for chronic obstructive pulmonary disease

Pooled over studies, separate for treatments

```
fit_pooled <- brm(exac | trials(total) ~ 0 + treatment,  
  prior = prior(student_t(7, 0, 1.5), class='b'),  
  family=binomial(), data=dat.baker2009)
```

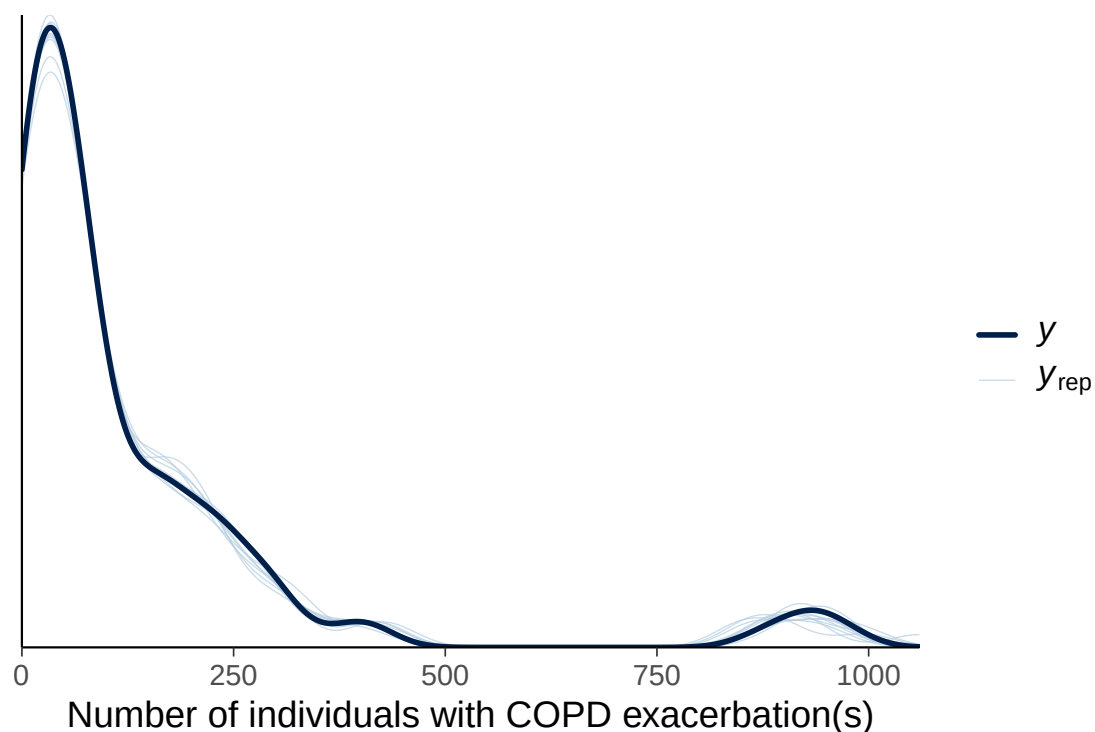


Posterior predictive checking

meta-analysis: Pharmacologic treatments for chronic obstructive pulmonary disease

Hierarchical for studies, hierarchical for treatments

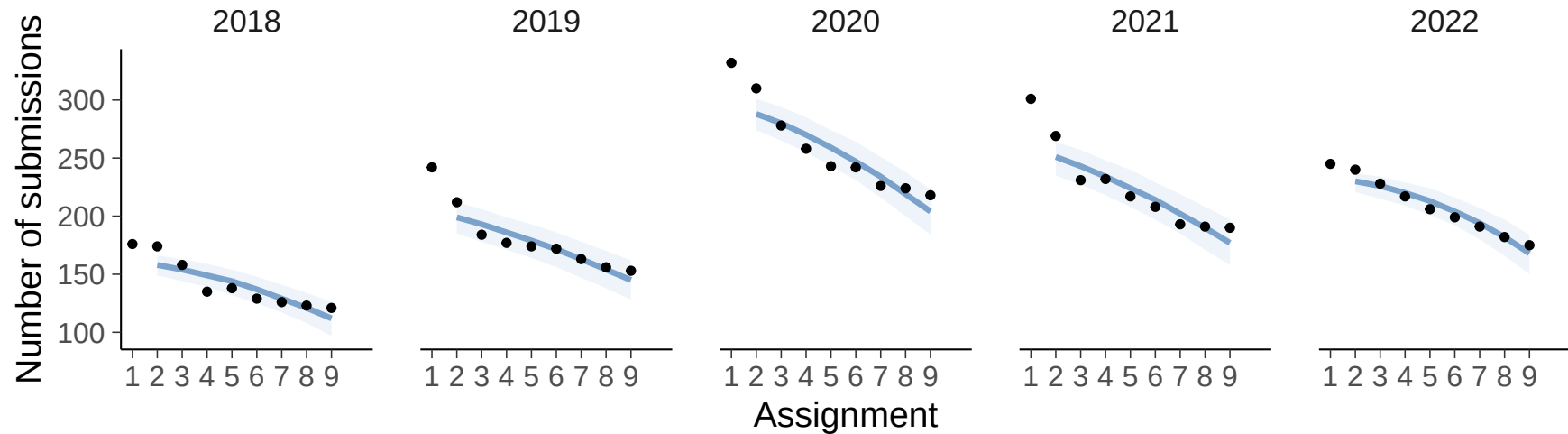
```
fit_hier <- brm(exac | trials(total) ~ (1 | treatment) + (1 | study),  
  family=binomial(), data=dat.baker2009)
```



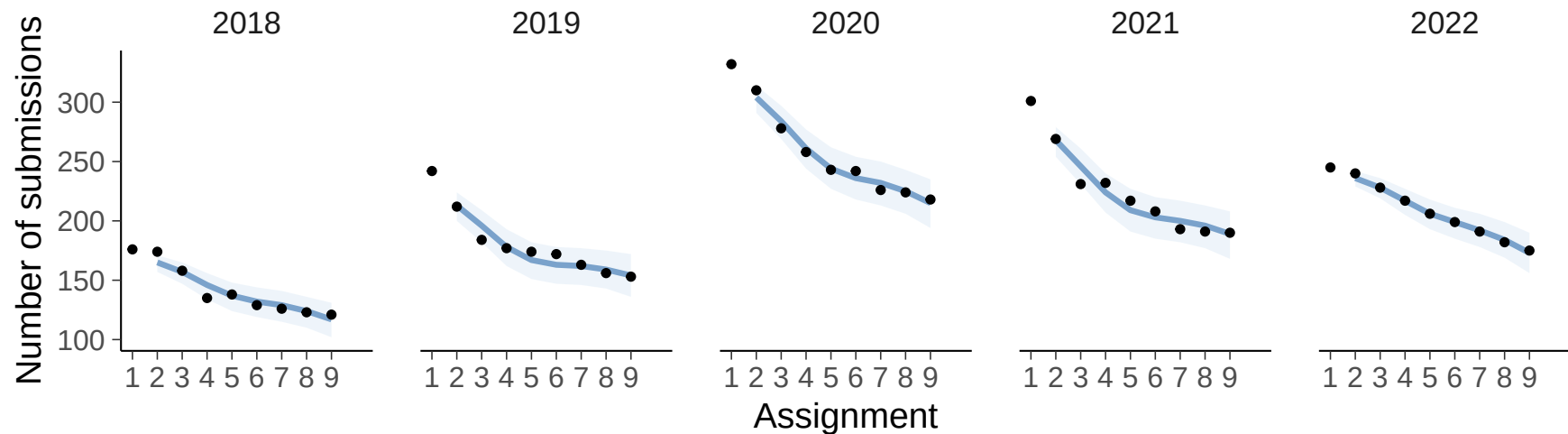
- For very flexible models posterior predictive checks may become very optimistic.

Student retention Posterior predictive distributions

Latent hierarchical linear model



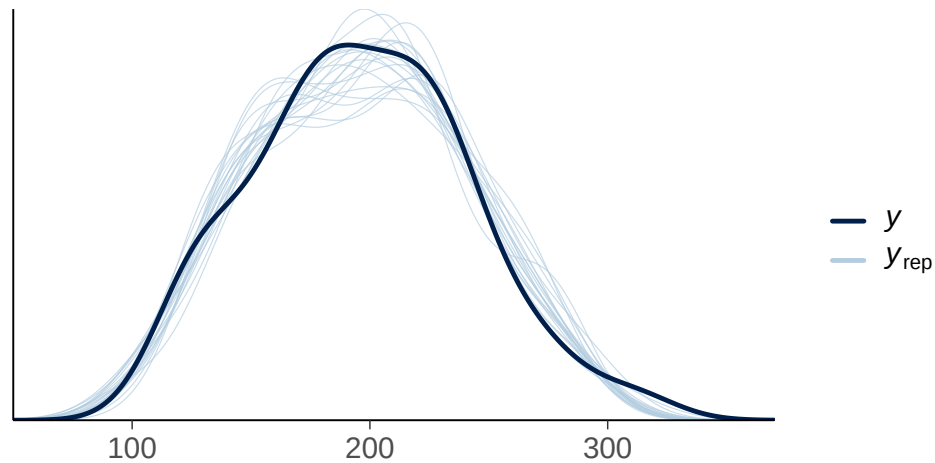
Latent hierarchical linear model + spline



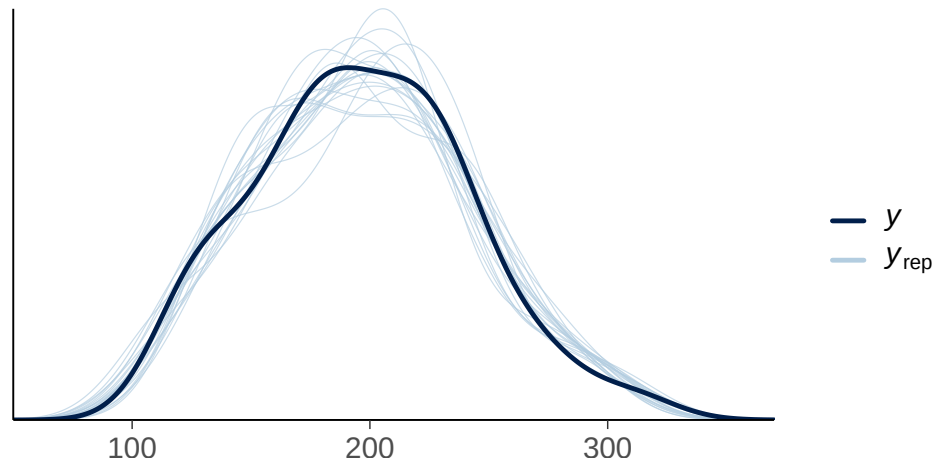
Student retention – Marginal PPC

```
pp_check(fit, ndraws=100)
```

Latent hierarchical linear model



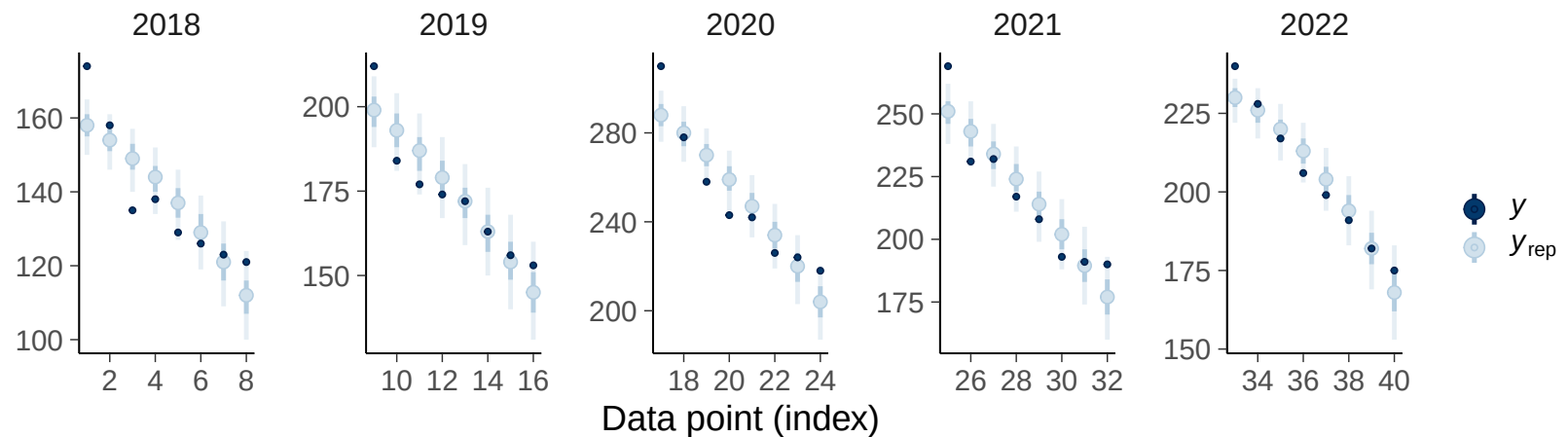
Latent hierarchical linear model + spline



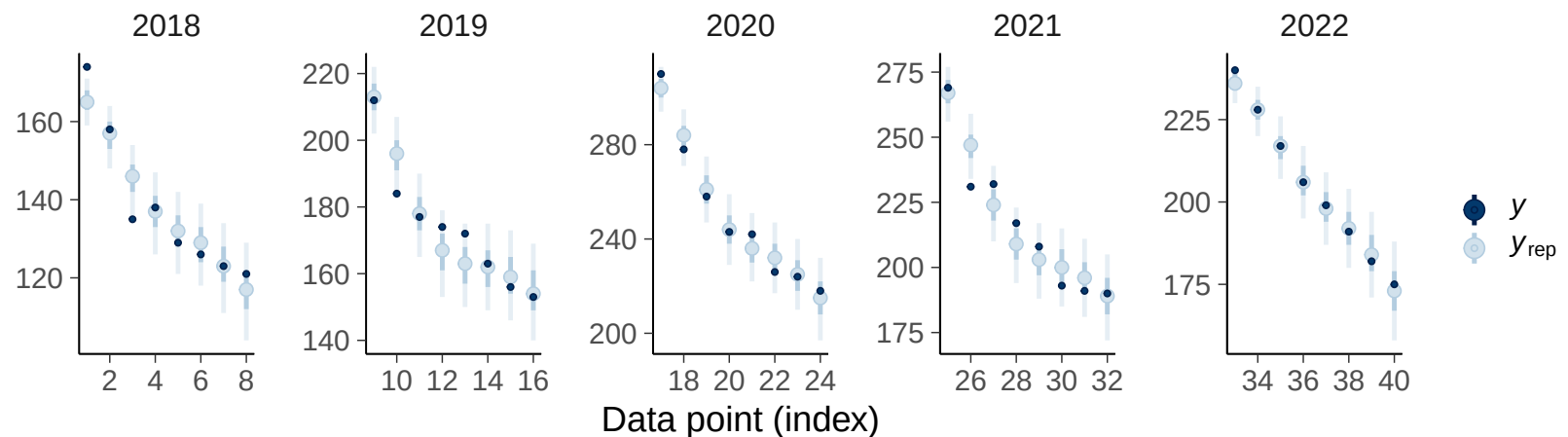
Student retention Posterior predictive intervals

```
pp_check(fit, type = "intervals_grouped", group="year")
```

Latent hierarchical linear model



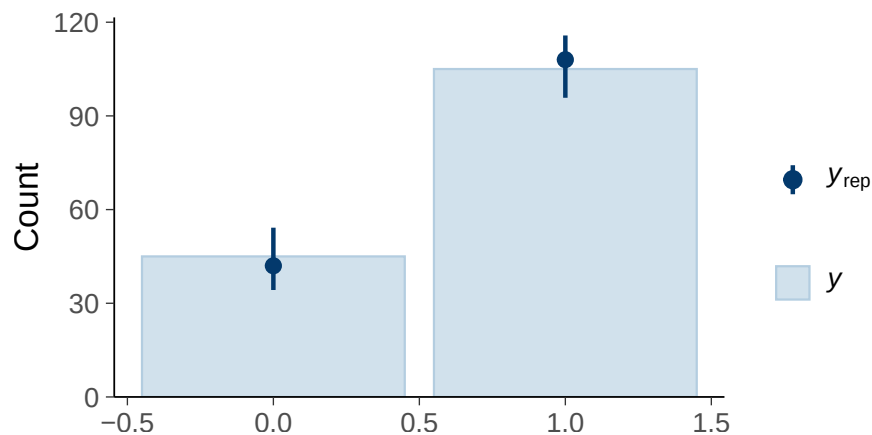
Latent hierarchical linear model + spline



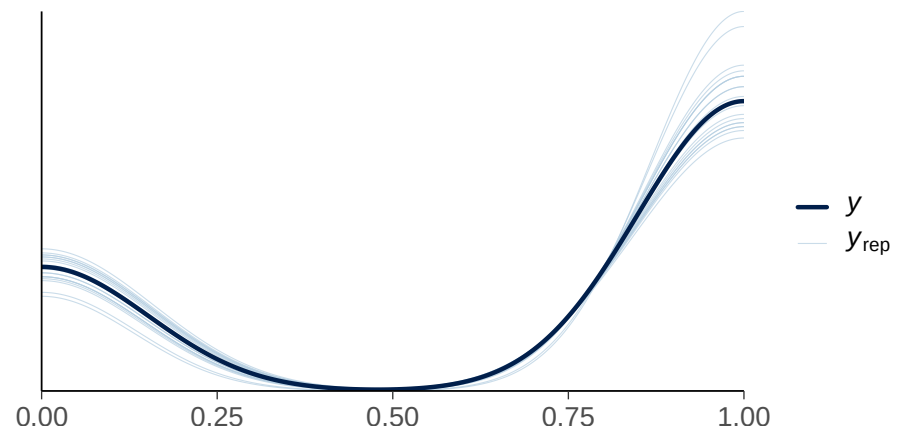
PPC for binary target - DON'T USE BAR OR KDE PLOTS

- Bar plots are not very informative for assessing model fit.
- Even a simple model can perfectly reproduce the overall proportion of positive outcomes, making these plots unable to distinguish good from poor models.
- KDE are even worse

Bad idea. Don't do this

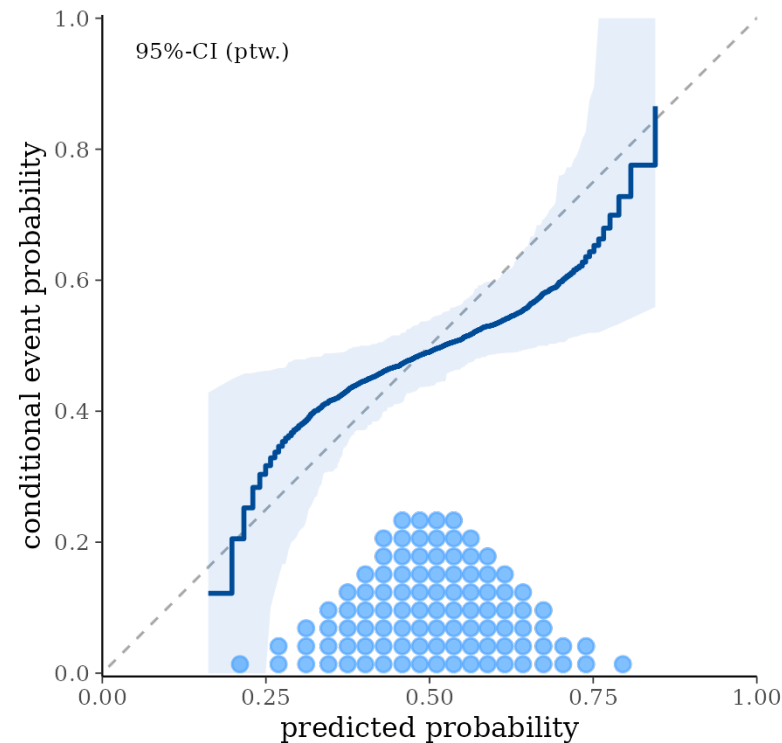


This is even worse!!!



Use Calibration plots for binary targets

- Compare predicted probabilities with the observed frequency of positive outcomes.
- Traditional calibration plots bin predictions (e.g., $[0-0.1]$, $[0.1-0.2]$, ... $[0.9-1.0]$), but results can depend strongly on the choice of bins.
- A more robust approach, PAV-adjusted calibration, avoids binning.



Prior predictive checking

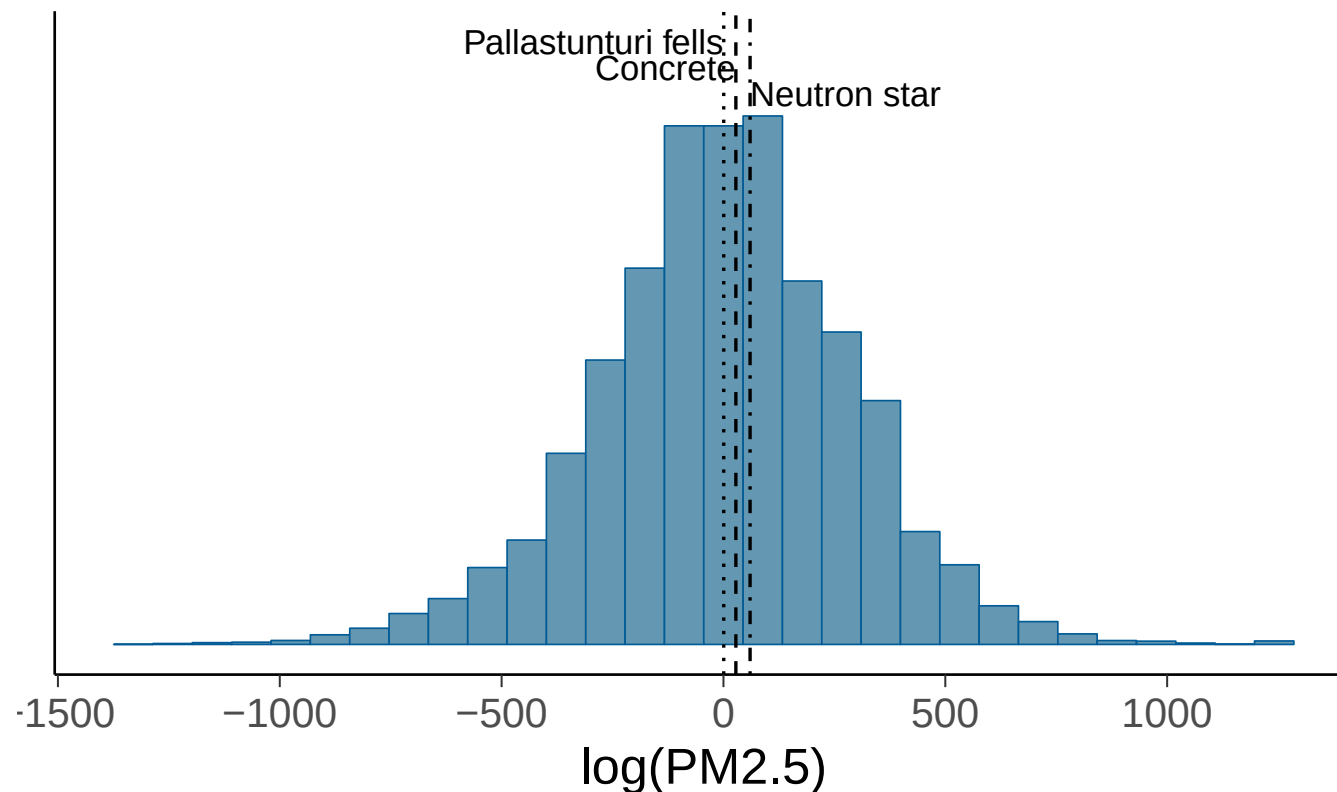
- Similar to posterior predictive checking, but done before observing the data
- Compute predictions (before seeing the data) and compare them to the domain knowledge (not the data)
- Assess if the model predictions are consistent with the domain knowledge
- If not, revise model or assumptions

https://arviz-devs.github.io/EABM/Chapters/Prior_posterior_predictive_checks.html

Exposure to air pollution

- Estimation of human exposure to air pollution from particulate matter measuring less than 2.5 microns in diameter ($\text{PM}_{2.5}$)

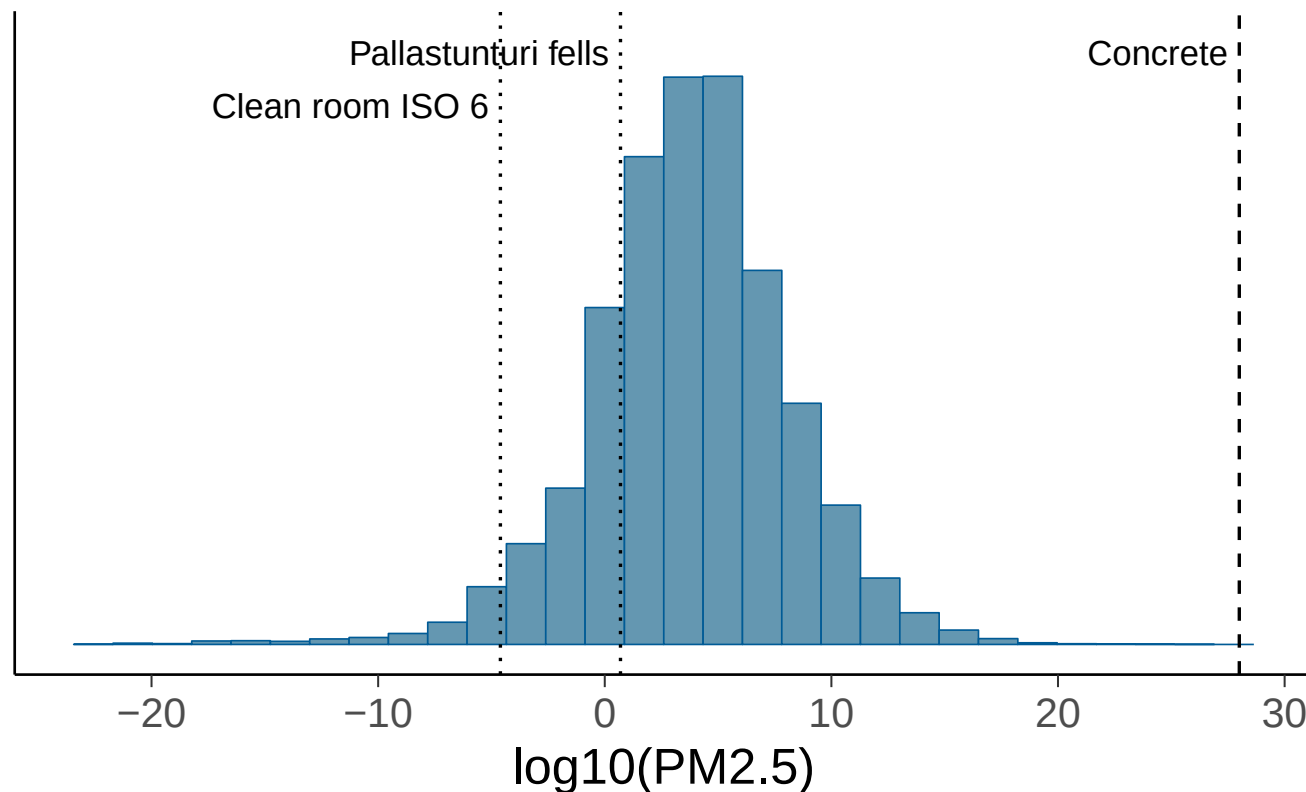
Prior predictive distribution with vague prior



Exposure to air pollution

- Estimation of human exposure to air pollution from particulate matter measuring less than 2.5 microns in diameter ($\text{PM}_{2.5}$)

Prior predictive distribution with weakly informative



Chapter 6

- 6.1 The place of model checking in applied Bayesian statistics
- 6.2 Do the inferences from the model make sense?
- 6.3 Posterior predictive checking
- 6.4 Graphical posterior predictive checks
 - better to read the paper Gabry et al. (2019). *Visualization in Bayesian workflow* <https://doi.org/10.1111/rssa.12378>
- 6.5 Model checking for the educational testing example

Model checking

- demo6_1: Posterior predictive checking - light speed
- demo6_2: Posterior predictive checking - sequential dependence
- demo6_3: Posterior predictive checking - poor test statistic
- https://avehtari.github.io/BDA_R_demos/demos_rstan/brms_demo.html

Model checking and model assessment

- Posterior predictive checks
 - is the model consistent with the observed data? where does it fail?
- Predictive performance
 - how well does the model predict new data?
 - which model predicts new data better? → model comparison

Predictive performance

- True predictive performance is measured by making predictions and comparing predictions to new observations
 - external validation
 - this can be costly or impractical
- Expected predictive performance
 - approximates the external validation using observed data

Predictive performance

- We need to choose the utility/cost function
- In principle these are application specific
 - e.g., money, life years, quality adjusted life years, etc.
- If we are interested in the overall goodness of the predictive distribution, then the log-score is a good theoretically justified choice.

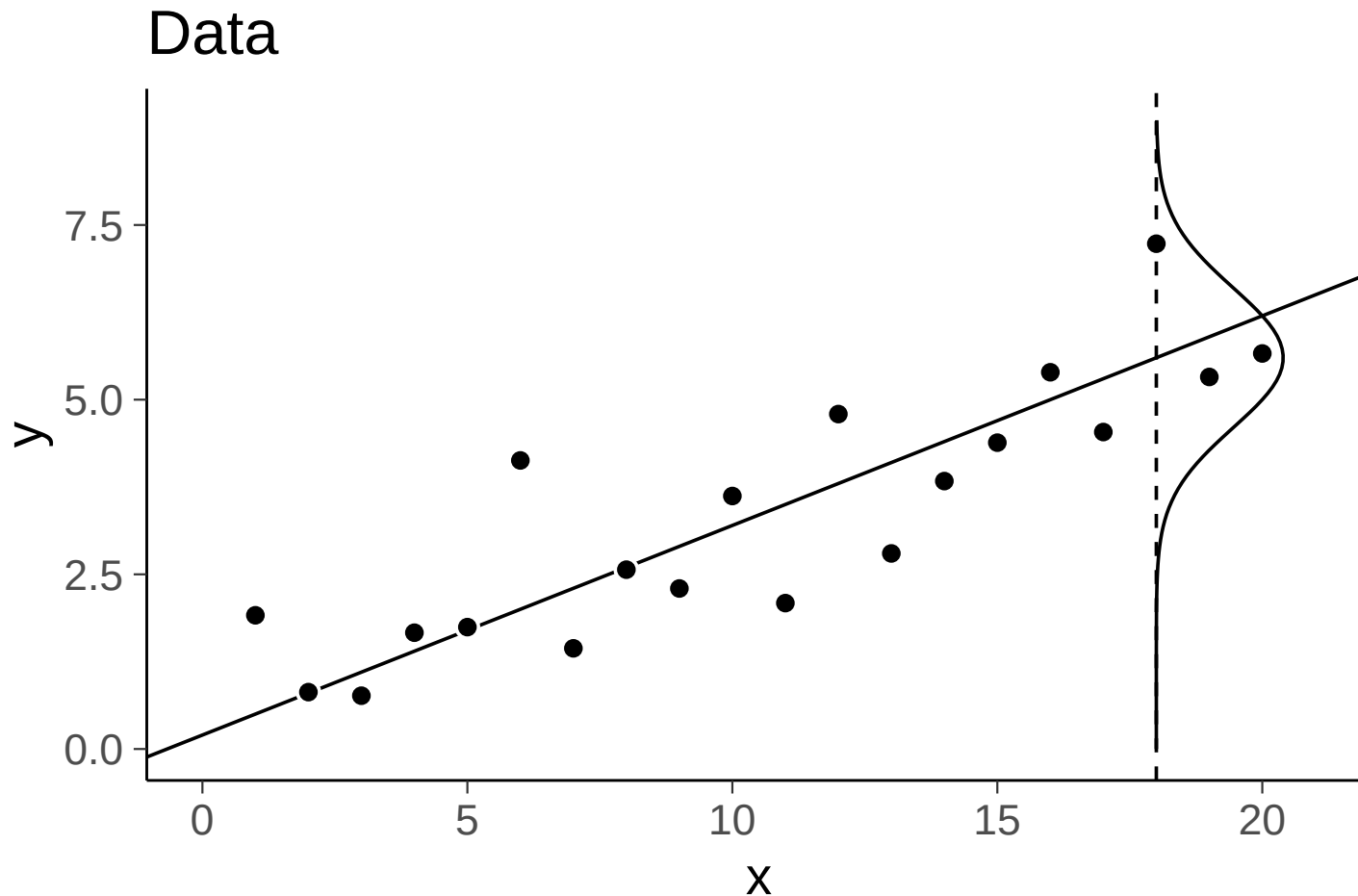
$$\log p(y^{\text{rep}} \mid y, M)$$

Cross-validation

- Cross-validation is a method to estimate predictive performance by splitting the data into training and test sets.
- There are many variants of cross-validation
- k-fold cross-validation.
 - split the data into k subsets and use each subset as a test set while training on the remaining $k-1$ subsets
- leave-one-out cross-validation (LOO-CV).
 - each observation is used once as a test set while the remaining observations form the training set

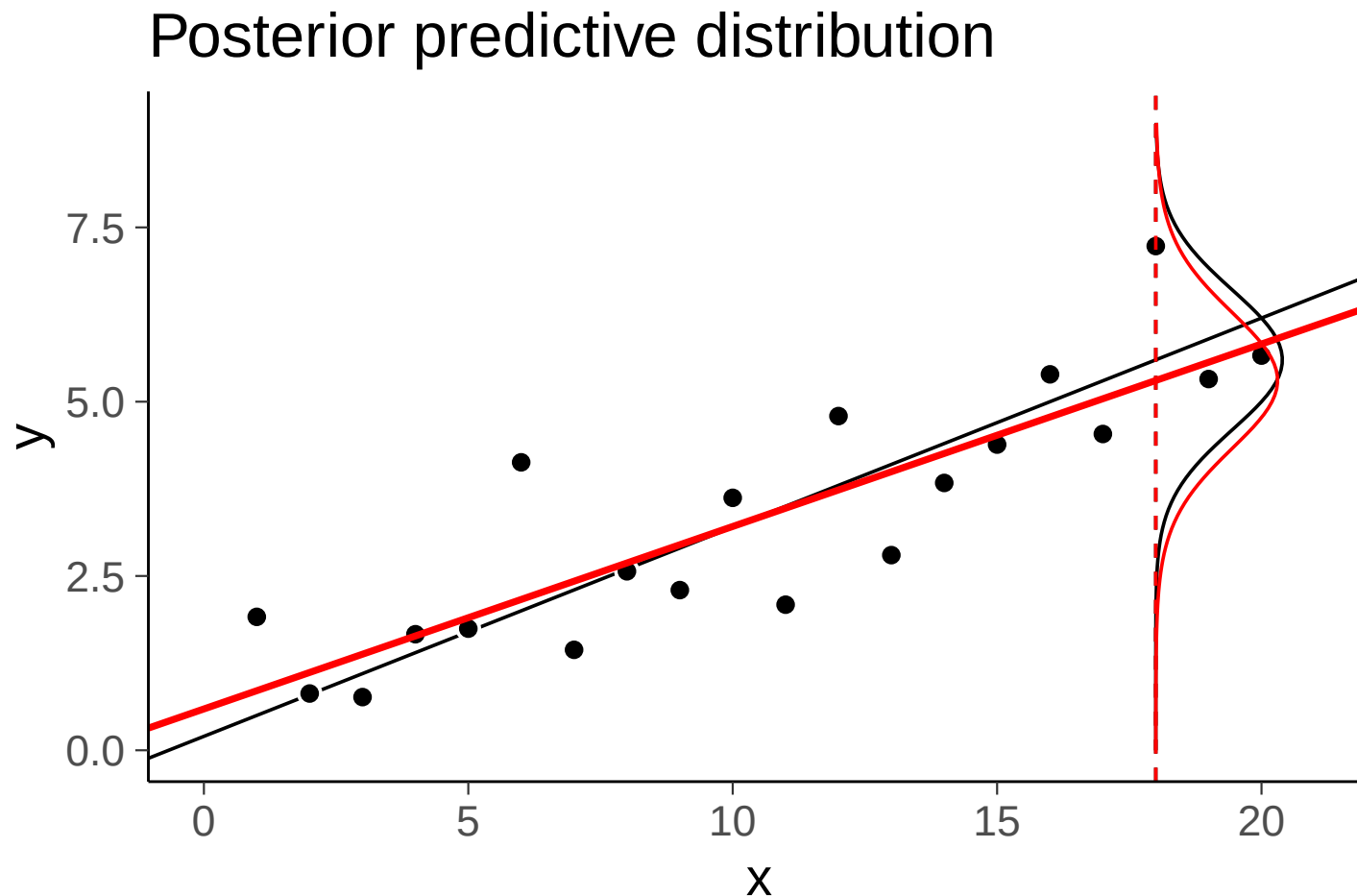
Toy linear model

- We know the true data-generating process for this toy linear model
- Normal with mean given by the linear model (known intercept and slope) and some variance



Toy linear model

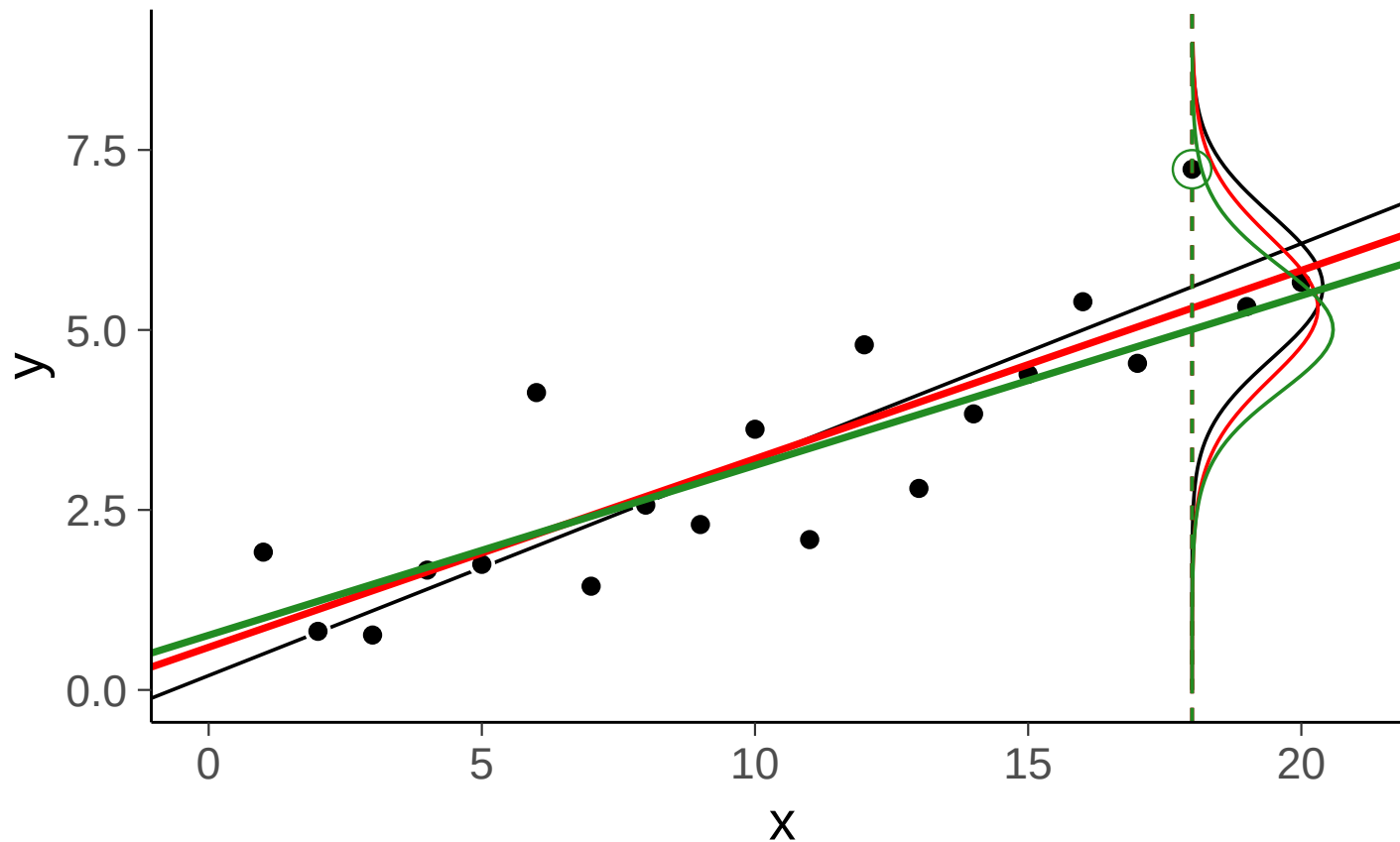
- We compute the posterior and then the posterior predictive distribution
- For posterior predictive checking we compare the observed data to the posterior predictive distribution



Toy linear model

- We can remove one observation ($x = 18$) and compute the leave-one-out posterior mean and posterior predictive distribution (at $x = 18$)

Leave-one-out predictive distribution



Toy linear model

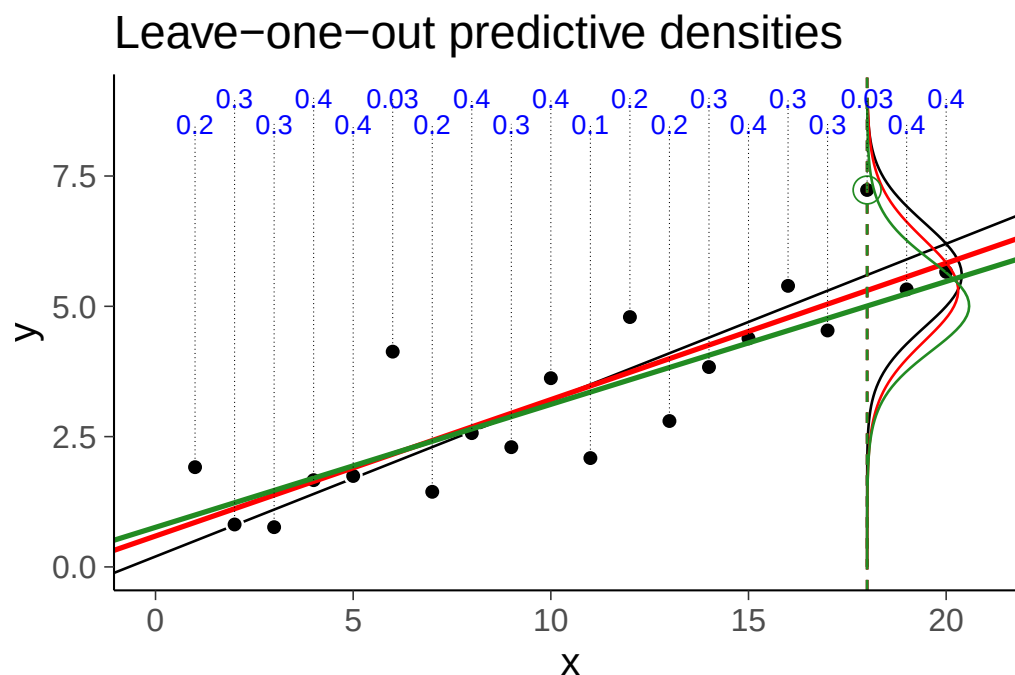
- We can evaluate the density including or excluding the observation at $x = 18$
- It's expected to be higher when including the observation and lower when excluding it

$$p(\tilde{y} = y_{18} \mid \tilde{x} = 18, x, y) \approx 0.07$$

$$p(\tilde{y} = y_{18} \mid \tilde{x} = 18, x_{-18}, y_{-18}) \approx 0.03$$

Toy linear model

- We remove each observation one at a time and compute the corresponding leave-one-out log density
- The sum of these leave-one-out log densities gives an estimate of the leave-one-out log predictive density for the whole dataset



$$\text{elpd_loo} = \sum_{i=1}^{20} \log p(y_i \mid x_i, x_{-i}, y_{-i}) \approx -29.5$$

Toy linear model

$$\text{elpd_loo} = \sum_{i=1}^{20} \log p(y_i \mid x_i, x_{-i}, y_{-i}) \approx -29.5$$

$$\text{lpd} = \sum_{i=1}^{20} \log p(y_i \mid x_i, x, y) \approx -26.8$$

- The lpd is optimistic compared to the elpd_loo (higher values indicate better fit)

$$\text{p_loo} = \text{lpd} - \text{elpd_loo} \approx 2.7$$

- For correctly specified regular model p_loo asymptotically approaches the number of parameters p in a model

Toy linear model

- elpd_loo is an estimate of the expected log predictive density for new data
- the uncertainty associated with this estimate is quantified using the standard error (SE)

$$\text{elpd_loo} = \sum_{i=1}^{20} \log p(y_i \mid x_i, x_{-i}, y_{-i}) \approx -29.5$$

$$\text{SE} = \text{sd}(\log p(y_i \mid x_i, x_{-i}, y_{-i})) \cdot \sqrt{20} \approx 3.3$$

Brute-force LOO

- Re-run MCMC n times to sample from $p(\theta \mid x_{-i}, y_{-i})$
 - can take a lot of time
 - can be parallelized easily as each leave-one-out fit is independent

Fast cross-validation

- K -fold cross-validation
- Pareto smoothed importance sampling leave-one-out cross-validation (PSIS-LOO-CV)

see Vehtari, Gelman & Gabry (2017a) and mc-stan.org/loo/



Importance sampling LOO-CV

- We want to compute $p(y_i \mid x_i, x_{-i}, y_{-i}) = \int p(y_i \mid x_i, \theta) p(\theta \mid x_{-i}, y_{-i}) d\theta$
- We can use importance sampling and a single fit (one MCMC run)
 - Proposal distribution is full posterior $\theta^{(s)} \sim p(\theta \mid x, y)$
 - Target distribution is LOO-posterior $p(\theta \mid x_{-i}, y_{-i})$
- Importance ratio, using that observations are conditionally independent given θ :

$$w_i^{(s)} \propto \frac{p(\theta^{(s)} \mid x_{-i}, y_{-i})}{p(\theta^{(s)} \mid x, y)} = \frac{p(y_{-i} \mid x_{-i}, \theta^{(s)}) p(\theta^{(s)})}{p(y_i \mid x_i, \theta^{(s)}) p(y_{-i} \mid x_{-i}, \theta^{(s)}) p(\theta^{(s)})}$$

$$w_i^{(s)} \propto \frac{1}{p(y_i \mid x_i, \theta^{(s)})}$$

- Normalize the weights:

$$\tilde{w}_i^{(s)} = \frac{w_i^{(s)}}{\sum_{s'=1}^S w_i^{(s')}}}$$

Importance sampling LOO-CV

- Posterior draws $\theta^{(s)} \sim p(\theta \mid x, y)$ with weights $\tilde{w}_i^{(s)}$ approximate the LOO posterior:

$$E_{\theta \mid x_{-i}, y_{-i}}[h(\theta)] \approx \sum_{s=1}^S \tilde{w}_i^{(s)} h(\theta^{(s)})$$

- For example, the LOO predictive density of the left-out observation ($h(\theta) = p(y_i \mid x_i, \theta)$):

$$p(y_i \mid x_i, x_{-i}, y_{-i}) \approx \sum_{s=1}^S \tilde{w}_i^{(s)} p(y_i \mid x_i, \theta^{(s)})$$

Importance sampling LOO-CV

- LOO predictive density of the left-out observation: how well do we predict y_i when it is not used for fitting?

$$p(y_i \mid x_i, x_{-i}, y_{-i}) = \int p(y_i \mid x_i, \theta) p(\theta \mid x_{-i}, y_{-i}) d\theta$$

- Importance sampling with draws $\theta^{(s)} \sim p(\theta \mid x, y)$ and normalized weights $\tilde{w}_i^{(s)}$

$$p(y_i \mid x_i, x_{-i}, y_{-i}) \approx \sum_{s=1}^S \tilde{w}_i^{(s)} p(y_i \mid x_i, \theta^{(s)}) = \frac{\sum_{s=1}^S w_i^{(s)} p(y_i \mid x_i, \theta^{(s)})}{\sum_{s'=1}^S w_i^{(s')}}$$

- With $w_i^{(s)} \propto 1/p(y_i \mid x_i, \theta^{(s)})$ the weight cancels the likelihood in the numerator

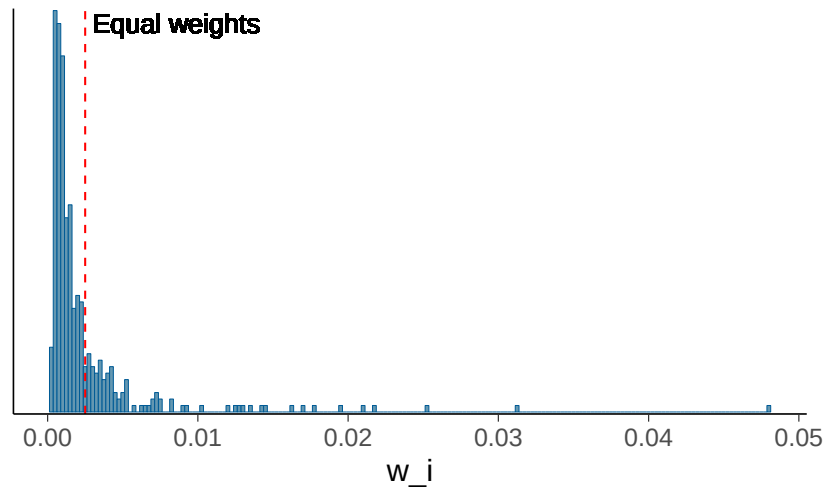
$$= \frac{\sum_{s=1}^S \frac{1}{p(y_i \mid x_i, \theta^{(s)})} p(y_i \mid x_i, \theta^{(s)})}{\sum_{s'=1}^S \frac{1}{p(y_i \mid x_i, \theta^{(s')})}} = \frac{1}{\frac{1}{S} \sum_{s=1}^S \frac{1}{p(y_i \mid x_i, \theta^{(s)})}}$$

Pareto smoothed importance sampling LOO

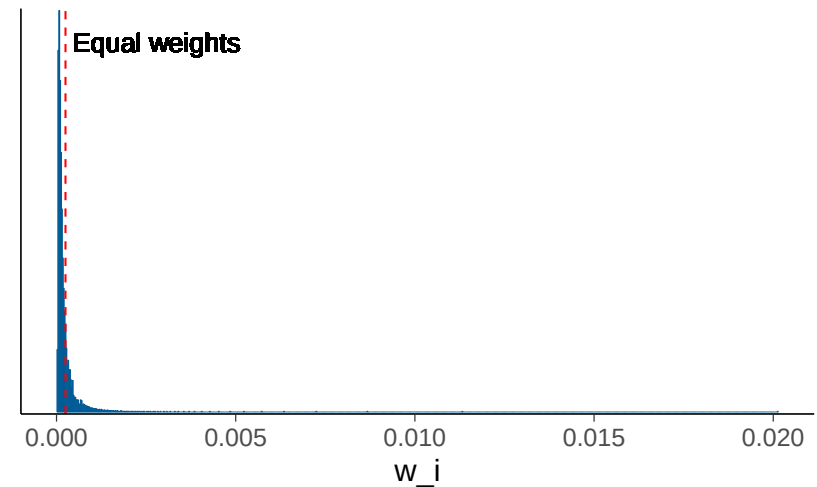
- The estimate is dominated by draws where $p(y_i | x_i, \theta^{(s)})$ is small, so the weights $1/p(y_i | x_i, \theta^{(s)})$ can have very high or infinite variance
 - this happens when observation i is highly influential, so the full and LOO posteriors differ a lot
- Pareto- $\hat{k}$ diagnostic: fit a generalized Pareto distribution to the largest weights
 - $\hat{k}$ tells how heavy the tail is, and whether the estimate can be trusted
- Pareto smoothed importance sampling LOO (PSIS-LOO)
 - replace the largest weights with the expected order statistics of the fitted Pareto
 - smoothed weights have lower variance, at the cost of a small bias

Pareto smoothed importance sampling LOO

400 importance weights for leave-18th-out



4000 importance weights for leave-18th-out



$$\text{ESS} \approx \frac{1}{\sum_{s=1}^S (\tilde{w}^{(s)})^2} \approx 459$$

Pareto $\hat{k} \approx 0.52$

- Pareto $\hat{k}$ estimates the tail shape which determines the convergence rate of PSIS. Less than 0.7 is ok.

Software in R and Python

- R: loo package
- Python: arviz package

Computed from 4000 by 20 log-likelihood matrix

	Estimate	SE
elpd_loo	-29.5	3.3
p_loo	2.7	1.0

MCSE of elpd_loo is 0.1.

MCSE and ESS estimates assume MCMC draws (r_eff in [0.6, 1.0]).

All Pareto k estimates are good (k < 0.7).

See `help('pareto-k-diagnostic')` for details.

Computing the log-likelihood

- For Stan we need to compute the log-likelihood in the generated quantities block

```
model {  
  alpha ~ normal(pmualpha, psalpha);  
  beta ~ normal(pmubeta, psbeta);  
  y ~ normal(mu, sigma);  
}  
generated quantities {  
  vector[N] log_lik;  
  for (i in 1:N)  
    log_lik[i] = normal_lpdf(y[i] | mu[i], sigma);  
}
```

- stanarm and brms compute `log_lik` by default
- PyMC and bambi used to compute it by default, but now it needs to be explicitly requested using the `compute_log_likelihood` function.

High Pareto- $\hat{k}$ values

- Pareto- $\hat{k}$ estimates how far the LOO posterior is from the full posterior
 - high $\hat{k}$ means observation i is highly influential, so importance sampling is unreliable
 - very high values often indicate misspecification, outliers or data errors
- If $\hat{k} > 0.7$, p_{loo} can be used as a diagnostic relative to the number of parameters, see more in [CV-FAQ](#)
- Not every high $\hat{k}$ needs fixing, e.g. discard a model that is clearly bad by PPC or much worse elpd

What if many high Pareto- $\hat{k}$'s

- `rstan::loo(..., moment_match = TRUE)`
`brms::loo(..., moment_match = TRUE)`
support implicitly adaptive importance sampling with moment matching algorithm by Paananen et al. (2021). See <http://mc-stan.org/loo/articles/loo2-moment-matching.html>
- `rstanarm::loo(..., k_threshold = 0.7)`
`brms::loo(..., reloo = TRUE)`
runs MCMC for the folds with $\hat{k}$ above the threshold
- Integrated LOO (for some models)
See <https://users.aalto.fi/~ave/modelselection/roaches.html>
- Use K -fold-CV (more about this later)
`rstanarm::kfold(..., K=10)`
`brms::kfold(..., K=10)`
RStan/CmdStanR vignette <http://mc-stan.org/loo/articles/loo2-elpd.html>

Student retention – loo computation

PSIS-LOO

```
> fit4 <- add_criterion(fit4, 'loo')
```

Pareto k diagnostic values:

		Count	Pct.	Min. ESS
(-Inf, 0.7]	(good)	32	80.0%	114
(0.7, 1]	(bad)	7	17.5%	<NA>
(1, Inf)	(very bad)	1	2.5%	<NA>

PSIS-LOO + moment matching + reloo

```
> ...(fit4, 'loo', moment_match=TRUE, reloo=TRUE, overwrite=TRUE)
```

All Pareto k estimates are good ($k < 0.7$).

Cross-validation for model assessment

- CV gives a predictive distribution for each left-out observation y_i
 - so we can compute any LOO quantity, not only the log score (elpd)
 - choose the one that matters for the application
- Examples of LOO quantities
 - predictive means, quantiles and event probabilities
 - errors and accuracy (MAE, RMSE, classification accuracy)
 - continuous ranked probability score (CRPS)
 - R^2
 - available in R (`loo`, `rstanarm`, `brms`) and Python (ArviZ)
- Also useful for diagnosing misspecification and for model checking
 - Pareto- $\hat{k}$, `p_loo`, LOO-PIT

LOO-PIT predictive checking

- Probability integral transform (PIT):
 - Where does the observation fall in its own predictive distribution?
 - if the model is well calibrated we should see a uniform distribution
- Posterior predictive PIT

$$p_i = p(y_i^{\text{rep}} \leq y_i \mid x_i, x, y)$$

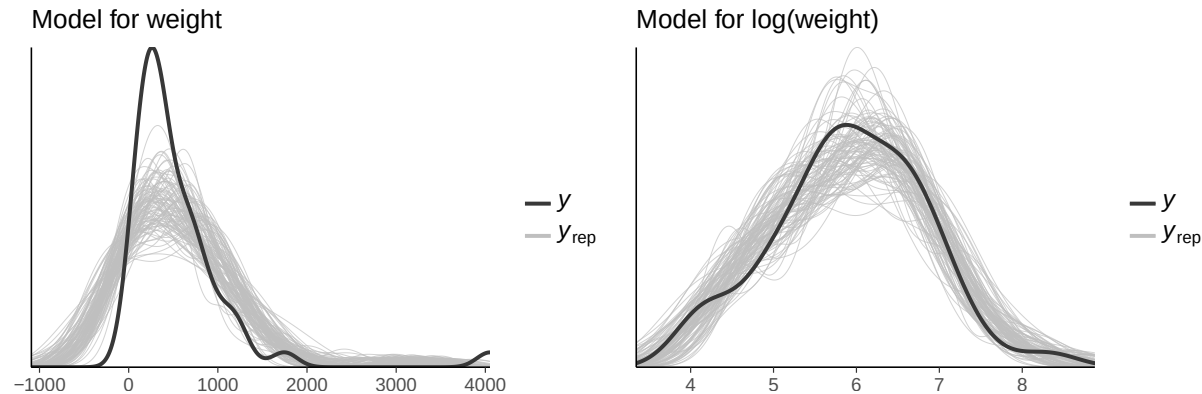
- double dipping $\rightarrow$ PIT values tend to concentrate around 0.5
- LOO-PPC-PIT

$$p_i = p(y_i^{\text{rep}} \leq y_i \mid x_i, x_{-i}, y_{-i})$$

- y_i is not used for fitting, so the check is closer to evaluating on new data
 - we can use PSIS-LOO-CV to approximate the LOO-PIT values efficiently

Sometimes cross-validation is not needed

- Posterior predictive checking is often sufficient



Predicting the yields of mesquite bushes.

Gelman, Hill & Vehtari (2020): Regression and Other Stories, Chapter 11.

- BDA3, Chapter 6
- Gabry, Simpson, Vehtari, Betancourt, Gelman (2019). Visualization in Bayesian workflow. JRSS A, <https://doi.org/10.1111/rssa.12378>
- mc-stan.org/bayesplot/articles/graphical-ppcs.html

Chapter 7

- 7.1 Measures of predictive accuracy
- 7.2 Information criteria and cross-validation
 - Instead of 7.2, read:
Vehtari, A., Gelman, A., Gabry, J. (2017). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*. 27(5):1413–1432. [preprint at arxiv.org/abs/1507.04544](https://arxiv.org/abs/1507.04544).
 - See also <https://users.aalto.fi/~ave/CV-FAQ.html>

Next week

- 7.3 Model comparison based on predictive performance
- 7.4 Model comparison using Bayes factors
- 7.5 Continuous model expansion / sensitivity analysis
- 7.5 Example (may be skipped)