Applied Bayesian Modeling module 6:
Sampling from posterior densities using Stan

Leontine Alkema, lalkema@umass.edu
Fall 2022

Motivating example

- Radon data: We want to estimate μ in $y_i|\mu, \sigma \sim N(\mu, \sigma^2)$ with σ UNKNOWN
- Module 5:
  - no closed-form expression for $p(\mu, \sigma|\boldsymbol{y})$
  - but we can work with samples $(\mu^{(s)}, \sigma^{(s)}) \sim p(\mu, \sigma|\boldsymbol{y})$ to summarize outcomes of interest (i.e., point estimate, CIs)
  - and we can use an MCMC algorithm to obtain those samples
- Let's use Stan to obtain some samples
  - Start at the end: fit a model using Stan, look at the outputs
  - Then zoom in on details

Stan

- Stan = programming language, commonly used as MCMC sampler for Bayesian analyses
- Stan website (`http://mc-stan.org/`) has lot of useful resources
- Various interfaces available, we will use the R packages `brms` and `rstan`
  - We start with `brms` (which takes in model formulas and has a lot of built-in defaults),
  - and consider `rstan` (and write our own model specs) later in the course.

Model fitting in R using `lm`

- We want to estimate μ in $y_i|\mu, \sigma \sim N(\mu, \sigma^2)$
- To start from familiar grounds (from fitting regression models): Let's first do maximum likelihood estimation in R with the `lm` function and using formula `y ~ 1`:
  - Left-hand side states that the response is data vector y, with default setting $y_i|\mu, \sigma^2 \sim N(\mu, \sigma^2)$
  - Right-hand side contains predictors for μ, here just an intercept
- Finding: $\hat{\mu}$ is 1.23 with 95% confidence interval $(1.17, 1.28)$; $\hat{\sigma} = 0.86$

R-code: `fit_lm <- lm(y ~ 1, data = dat)`

term	estimate	std.error	conf.low	conf.high
(Intercept)	1.23	0.03	1.17	1.28

```
Residual standard error: 0.8635 on 926 degrees of freedom
```

Model fitting in R using brm

- We want to estimate μ in $y_i|\mu, \sigma \sim N(\mu, \sigma^2)$ Bayesian-ly
- Fit using `brm` takes in the same formula `y ~ 1`
- If we don't specify priors, defaults are used (more on those later, as well as additional arguments)
- Finding: $\hat{\mu} = 1.23$ with 95% credible interval (1.17, 1.28); $\hat{\sigma} = 0.86$ with 95% CI (0.83, 0.90)

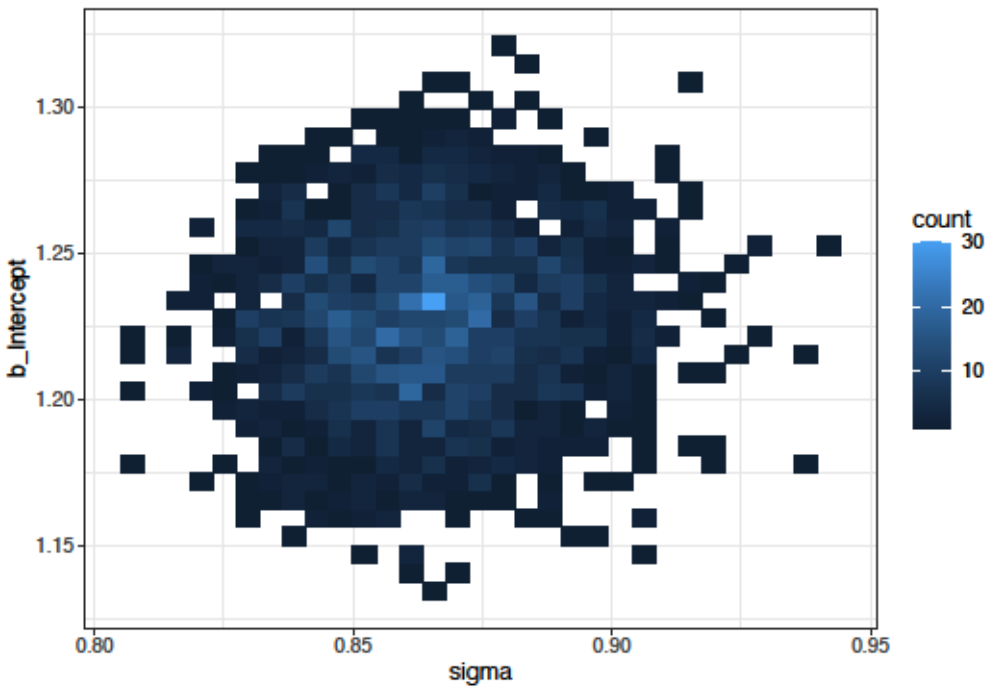
R-code: `fit <- brm(y ~ 1, data = dat, XXX)`

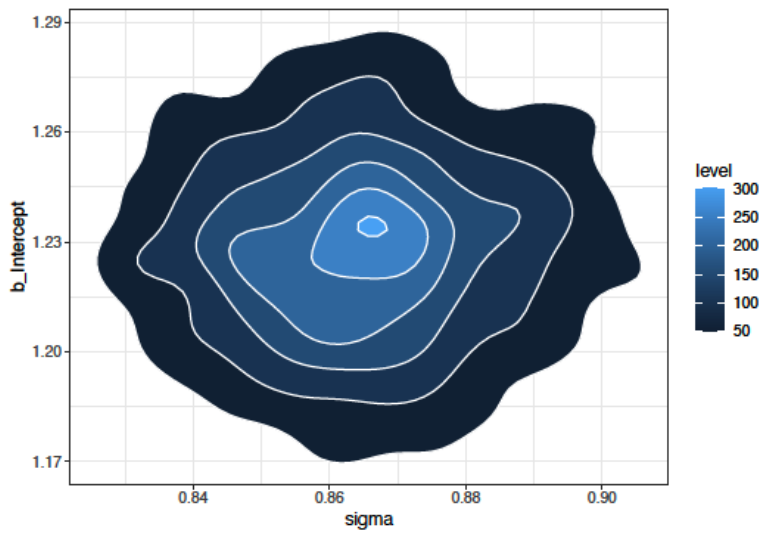
```
Population-Level Effects:
          Estimate Est.Error l-95% CI u-95%
Intercept     1.23      0.03     1.17     1.

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI R
sigma     0.86      0.02     0.83     0.90 1
```

brm-based outputs

- What are these summaries for μ and σ based on?
  Posterior samples $(\mu^{(s)}, \sigma^{(s)}) \sim p(\mu, \sigma | \boldsymbol{y})$

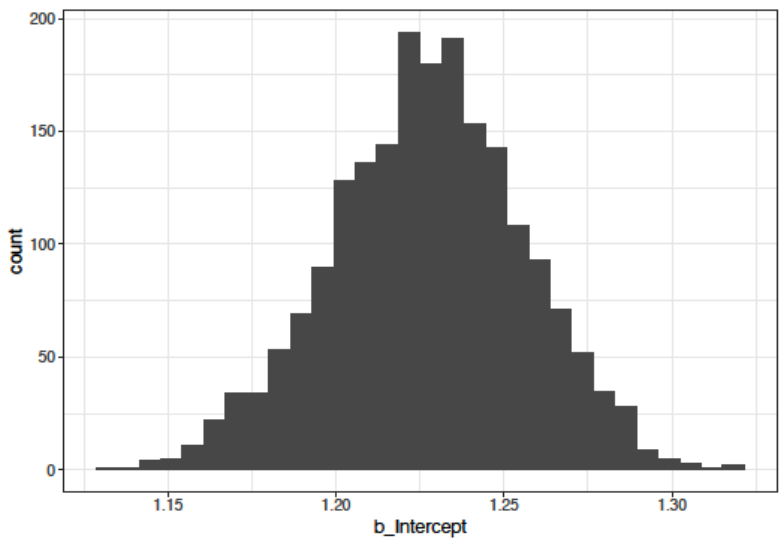




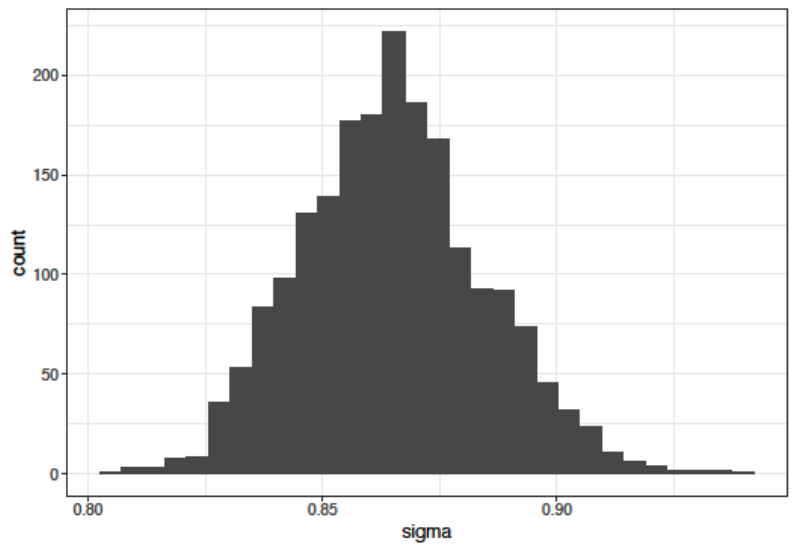
brm-based outputs: posterior samples

- These are histograms with corresponding marginal densities
- These samples were used to produce point estimates and CIs

$$\mu^{(s)} \sim p(\mu|\boldsymbol{y})$$



$$\sigma^{(s)} \sim p(\sigma|\boldsymbol{y})$$



brm-based outputs: what else?

- Other outputs are related to MCMC diagnostics

```
summary(fit)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: y ~ 1
##    Data: dat (Number of observations: 927)
##   Draws: 4 chains, each with iter = 1000; warmup = 500; thin = 1;
##          total post-warmup draws = 2000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     1.23      0.03     1.17     1.28 1.00     1603     1352
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.86      0.02     0.83     0.91 1.00     1800     1349
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

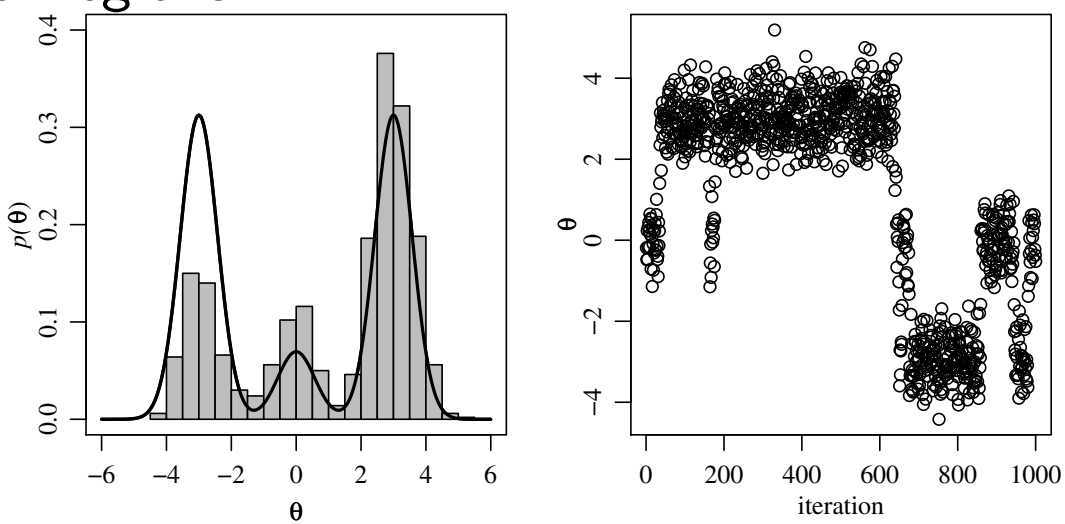
To discuss: diagnostics and tuning when using MCMC

- Recap of a Markov Chain Monte Carlo (MCMC) algorithm:
  - Let $\boldsymbol{\phi}$ = parameter vector of interest, i.e. $\boldsymbol{\phi} = (\mu, \sigma)$.
  - Goal: obtain samples $\boldsymbol{\phi}^{(s)}$ from the target distribution, here $p(\boldsymbol{\phi}|\boldsymbol{y})$
  - MCMC approach:
    - get some initial value $\boldsymbol{\phi}^{(1)}$ and create a sequence $\boldsymbol{\phi}^{(1)}, \boldsymbol{\phi}^{(2)}, \ldots$
    - such that for some large s, $\boldsymbol{\phi}^{(s)}$ is a draw from the target distribution
  - In MCMC, $\boldsymbol{\phi}^{(s)}$ depends on $\boldsymbol{\phi}^{(s-1)}, \boldsymbol{\phi}^{(s-2)}, \ldots, \boldsymbol{\phi}^{(1)}$ only through $\boldsymbol{\phi}^{(s-1)}$. This is called the Markov property, and so the sequence is called a **Markov chain**.
  - We approximate quantities of interest, e.g. $E(\mu|\boldsymbol{y})$, using resulting samples, which adds in the **Monte Carlo** part.
- This module: MCMC-related terminology, how to work with MCMC samples, MCMC diagnostics
- A later module: more details on sampling in Stan, tuning

Working with MCMC samples: what's the (potential) problem?

Illustrative example (Hoff 6.6)

- Suppose that we want to obtain a sample from density $p(\theta)$, which is shown with the solid black line (and given by a mixture of normals).
- Histogram: first 1,000 samples from an MCMC algorithm (sampled sequence shown on the right).
- The problem: MCMC does NOT give a representative sample yet: the chain 'spent too much time around '$\mu = 3$ and 0' as compared to the other regions.



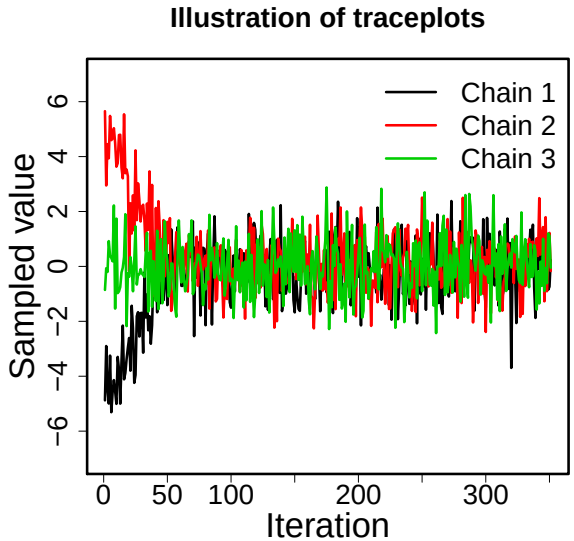
Working with MCMC samples: summary of what to look out for

- MCMC samples are NOT independent draws from a target distribution:
  - The first draw is not a random draw from the target distribution.
  - Tuning of MCMC parameters may be needed at the start of the chain
  - Subsequently, draw $s+1$ depends on draw s:
    the samples are autocorrelated.
- We can use samples from an MCMC algorithm to do inference but ONLY IF we have "waited long enough" for those samples to be representative of the distribution of interest.
- To work with MCMC samples, we need to
  - exclude samples from the initial period
  - (try to) check that the chain has generated a representative sample.

MCMC diagnostics

- General approach: run several chains, with different initial values
- Visual checks: Trace plot for each parameter
- Calculate measures

MCMC diagnostics: trace plots & burn-in/warm-up phase



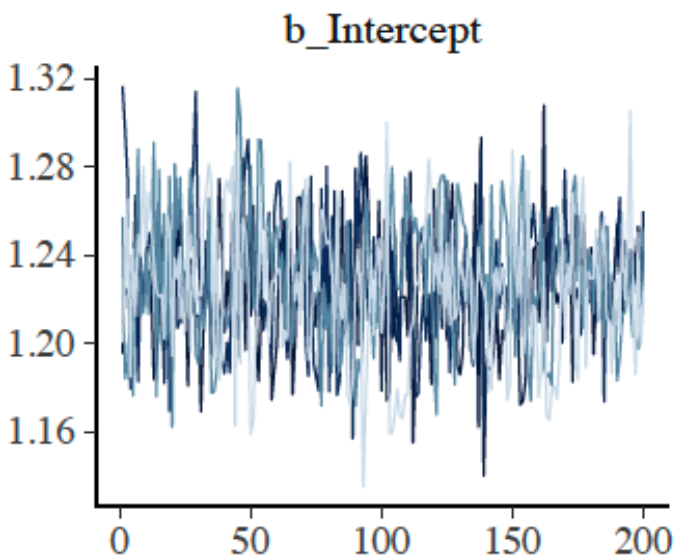
- Trace plot: sampled parameter against iteration number
- Trace plots can be used to detect issues with the MCMC chains

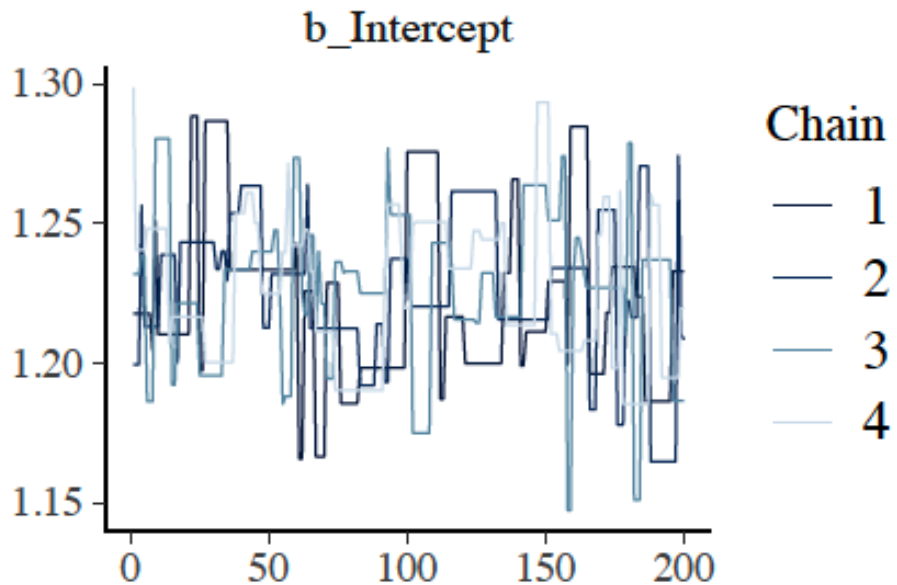
- **Burn-in phase** = Initial phase of an MCMC chain, when the chain converges away from initial values towards the target distribution.
  - Samples from the burn-up period should be discarded.
  - Note that we can never be sure that a chain has converged but at least we can detect lack of convergence, i.e. if chains don't overlap.
- When working with Stan/brms:
  - A user-specified number of initial iterations is referred to as **warm-up phase**, during which adaption of MCMC tuning parameters takes place (in addition to convergence away from initial values). This phase is excluded by default.

MCMC diagnostics: mixing

- Mixing refers to how the chains are exploring the parameter space
- Chains should be mixing well
  - chains are moving around quickly; autocorrelation in sampled values is low

Which chains are mixing faster in trace plots below (excluding warmup)?





MCMC diagnostics measures

- After excluding warmup, you need to check if you've generated "enough samples".
- How many samples are needed depends on:
  - Required precision of your estimates (MC error, more in a bit),
  - How fast the chain moves around the sample space (autocorrelation in the sampled values).
    - This is called the speed of mixing:
      Fast mixing means you'll get a representative sample from the target distribution faster.
- Diagnostic criteria: $\hat{R}$ and (various forms of) effective sample size

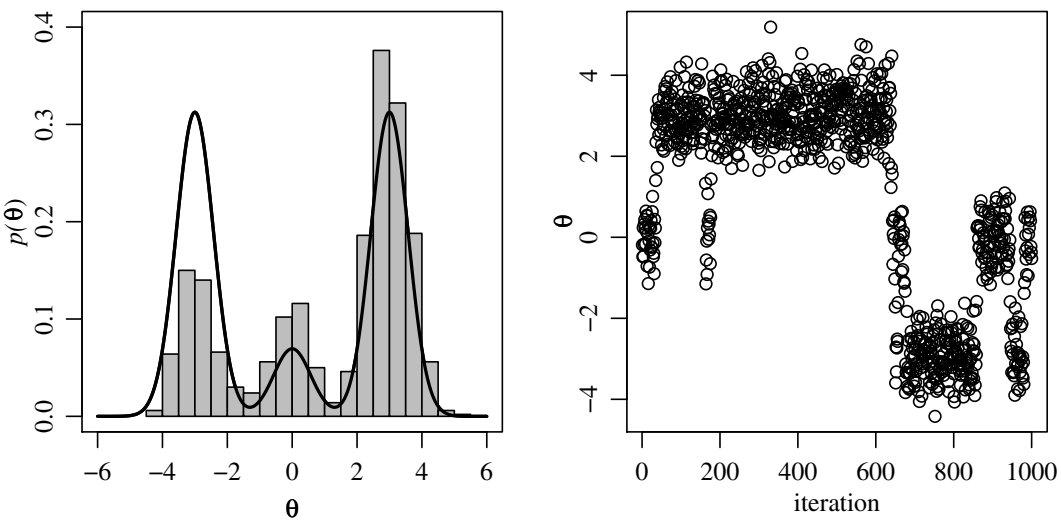
```
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     1.23      0.03     1.17     1.28 1.01      480      369
##
```

$\hat{R}$

- The (Gelman-Rubin) convergence diagnostic statistic $\hat{R}$, or potential scale reduction factor/shrink factor, is based on a comparison between the average variance of samples within each chain to the variance of the pooled samples across chains.
  - First proposed by Gelman and Rubin
  - Recently updated and improved by Vehtari et al, see `https://mc-stan.org/rstan/reference/Rhat.html`
- If chains have mixed well, then $\hat{R}$ is close to 1.
- Some rules of thumb:
  - Run at least 4 chains (more chains is better) and use disperse starting points
  - Aim for $\hat{R} < 1.05$

Effective sample size S_{eff}

- Main idea informally: calculate how many independent samples the set of correlated MCMC samples corresponds to
- Original definition of S_{eff} for a given MCMC sample of size S: S_{eff} = the number of independent MC samples that would give the same precision for estimating the mean, as obtained with the MCMC sample of size S.
- Example from Hoff 6.6: $S = 1000$, $S_{eff} \approx 19$.



Effective sample size S_{eff}: supplementary details

- S_{eff} = the number of independent MC samples that would give the same precision for estimating the mean, as obtained with the MCMC sample of size S.
- Details:
  - Set $\hat{\theta} = \bar{\theta} = 1/S \sum_s \theta^{(s)}$ when estimating $E(\theta)$ using $\theta^{(1)}, \ldots, \theta^{(S)}$
  - If the θ's are a random sample (independent draws) then $Var_{MC}(\hat{\theta}) = 1/S^2 Var(\sum_s \theta^{(s)}) = 1/S^2 \sum_s Var(\theta^{(s)}) = Var(\theta)/S$
  - If you use an MCMC algorithm, and the samples are positively correlated, then $Var_{MCMC}(\hat{\theta}) = 1/S^2 Var(\sum_s \theta^{(s)}) > 1/S^2 \sum_s Var(\theta^{(s)}) = Var_{MC}(\hat{\theta})$
  - $S_{eff} = Var(\theta)/Var_{MCMC}(\hat{\theta})$ such that S_{eff} MC samples gives the same variance as S MCMC samples: $Var_{MCMC}(\hat{\theta}) = Var(\theta)/S_{eff}$.
  - S_{eff} is estimated using an estimate of the autocorrelation in each chain

Effective sample size S_{eff} (ctd)

- Original definition of S_{eff} for a given MCMC sample of size S: S_{eff} = the number of independent MC samples that would give the same precision for estimating the mean, as obtained with the MCMC sample of size S.
- More recently developed standard MCMC diagnostics include:
  - Bulk Effective Sample Size (bulk-ESS): measure for effective sample sizes for mean and median estimates
  - Tail Effective Sample Size (tail-ESS): minimum of effective sample sizes for 5% and 95% quantiles
- Recommendation: Both bulk-ESS and tail-ESS should be at least 100 (approximately) per Markov Chain

MCMC-related arguments and diagnostics for example

Bayesian regression

```
fit <- brm(y ~ 1, data = dat,
           chains = 4, iter = 1000, warmup = 500, cores = getOption("mc.cores", 4))
```

```
summary(fit)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: y ~ 1
##    Data: dat (Number of observations: 927)
##   Draws: 4 chains, each with iter = 1000; warmup = 500; thin = 1;
##          total post-warmup draws = 2000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     1.23      0.03     1.17     1.28 1.00     1603     1352
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.86      0.02     0.83     0.91 1.00     1800     1349
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

What to do if there are issues?

- Do not use the samples!
- Minor issues for standard models, i.e. effective sample sizes little low, no other warnings:
  - Increase warm-up and number of iterations as needed
- Bigger issues, i.e. Rhat exploding, lots of warnings:
  - Are you sure about your model specs and data inputs?
  - More details on warnings, reparametrization, and tuning of MCMC parameters to follow later

Summary

- MCMC algorithm produces chain $\boldsymbol{\phi}^{(1)}, \boldsymbol{\phi}^{(2)}, \ldots, \boldsymbol{\phi}^{(S)}$;
  1. The first draw is set by the user and thus not a random draw from the target distribution.
  2. Subsequently, draw $s+1$ depends on draw s:
     We say that the samples are autocorrelated.
- For most problems, we can construct an MCMC algorithm that converges to the posterior distribution of interest such that, *eventually*, an MCMC sample is a draw from the target posterior distribution
- But... we canNOT just use samples to do inference as we would with random MC samples:
  - Remove warmup/burn-in and check mixing using trace plots and diagnostic measures
- If there are issues: do not use the samples yes, fix them!