Bayesian Analysis

Dr Niamh Cahill (she/her)

Markov Chain Monte Carlo (MCMC) Diagnostics

MCMC Reprentativeness, Accuracy and Efficiency

We have 3 main goals in generating an MCMC sample from the target (posterior) distribution.

1. The values must be representative of the posterior distribution. They shouldn't be influenced by the initial values of the chain. They should explore the full range of the parameter space.
2. The chain should be a sufficient size so that estimates are accurate and stable. Estimates and uncertainty intervals should not be much different if the MCMC is run again.
3. The chain should be generated as efficiently as possible.

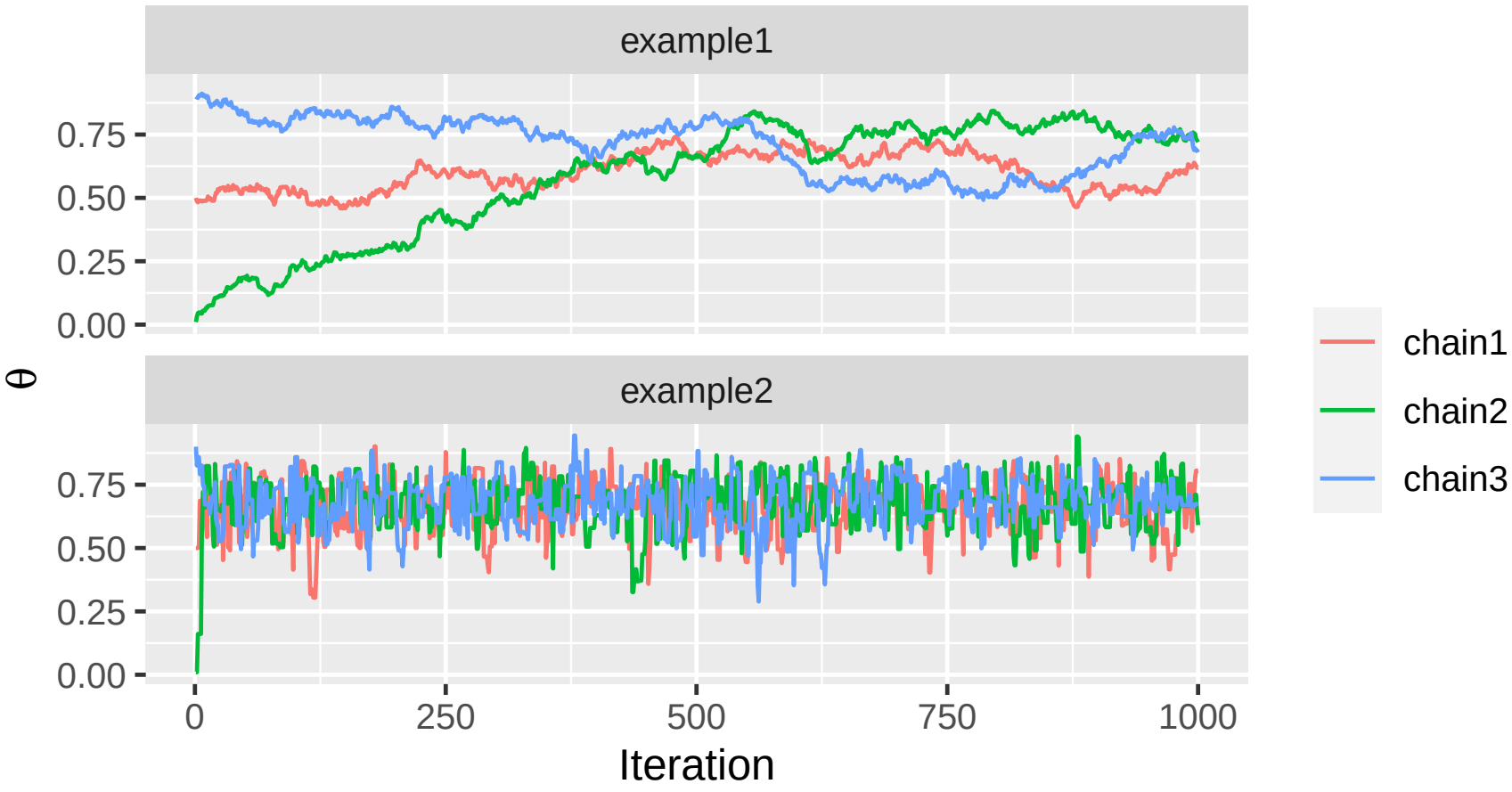
We cannot run chains for an infinitely long time so we must check the quality based on a set of finite samples from the chain.

We use a set of convergence diagnostics to check the quality.

Checking Trace Plots

The first method to detect convergence (or a lack thereof) is a visual examination of the MCMC chains.

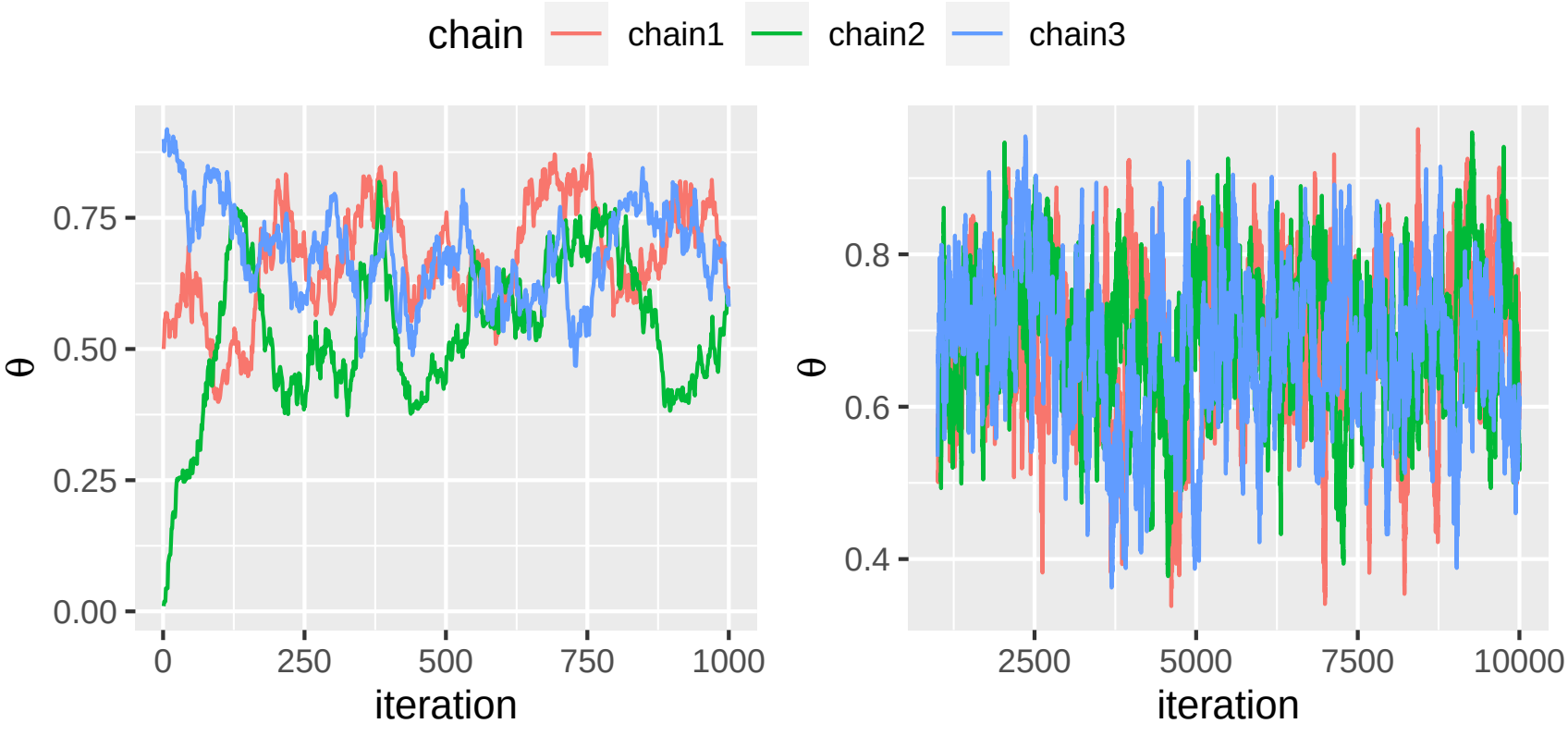
A graph of the sampled parameter values as a function of the MCMC step is called a *trace plot*. Here's some examples. Which one do you think is indicating convergence?



MCMC diagnostics: burn-in

The initial phase of an MCMC chain is called the burn-in phase, during which the chain converges towards the target distribution.

- Samples from the burn-in period should be discarded.
- The trace plots can be used to detect burn-in.



MCMC diagnostics: Potential Scale Reduction Factor ($\hat{R}$)

A popular numerical check for convergence is a measure of how much variance there is between chains relative to how much variance there is within chains.

- The idea is, that if all chains have settled into a converged state with representative sampling from the posterior, then the average difference between the chains should be the same as the average difference (across steps) within the chain.
- This is called the Brooks-Gelman-Rubin statistic or the "potential scale reduction factor" or the $\hat{R}$.
- The optimal value is 1. Usually 1.1 is used as a cutoff for flagging convergence issues.

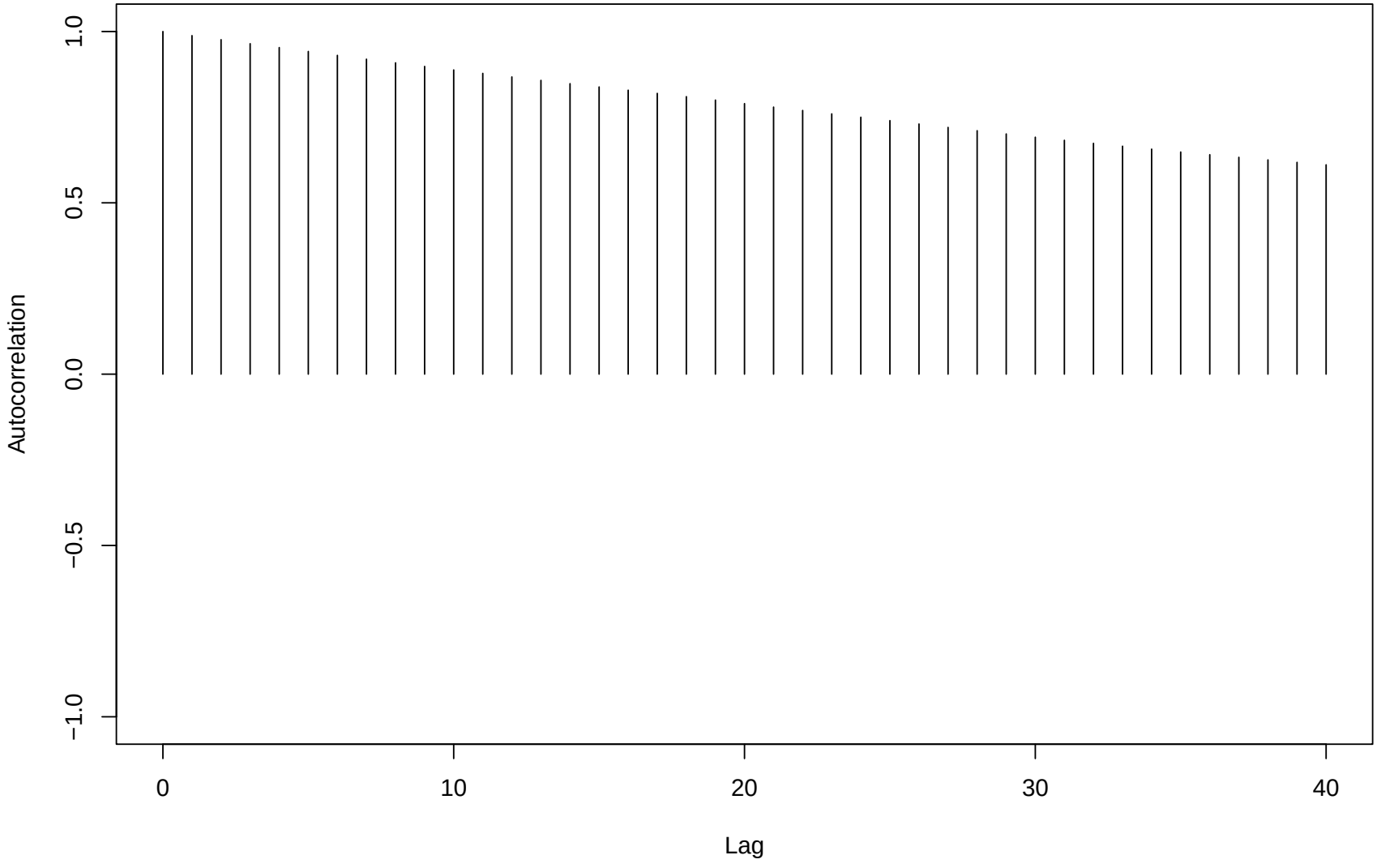
MCMC diagnostics: Autocorrelation

- Mont Carlo samples are direct/random/independent draws from a target distribution.
- MCMC samples are NOT independent draws from a target distribution, because:
  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 - samples are autocorrelated

However, we would like some measure of how much independent information there is in the autocorrelated chains.

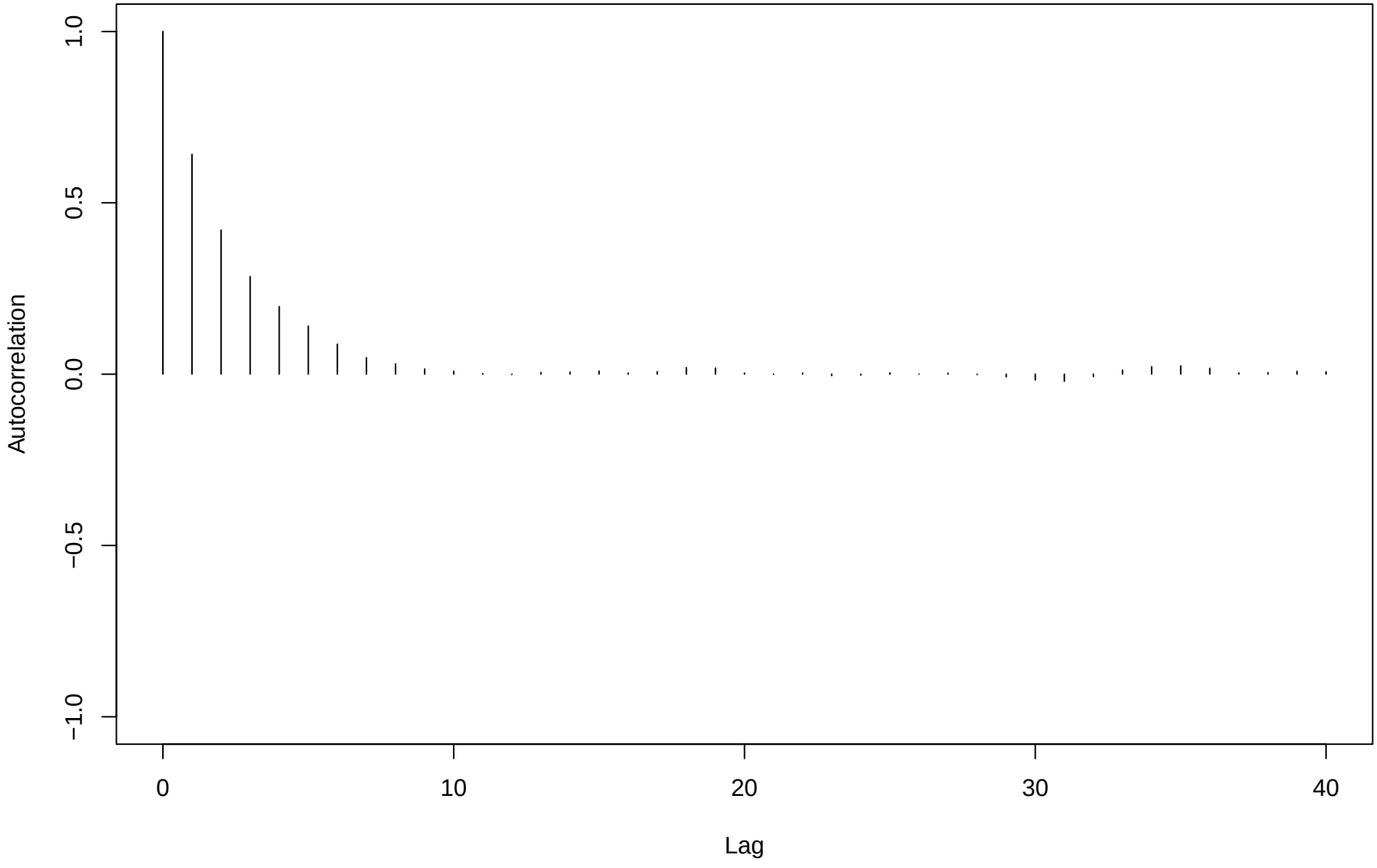
MCMC diagnostics: Autocorrelation

We can use autocorrelation function (ACF) plots to help us diagnose autocorrelation issues.



MCMC diagnostics: Autocorrelation

We can use autocorrelation function (ACF) plots to help us diagnose autocorrelation issues.



MCMC diagnostics: effective sample size (ESS)

We want to know what the sample size of a non-autocorrelated chain, that yields the same information, would be. An answer to this question can be provided with a measure called *effective sample size (ESS)*

The effective sample size (ESS) divides the actual sample size by the amount of autocorrelation.

$$ESS = \frac{N}{1 + 2\sum_{k=1}^{\infty} \rho_k}$$

where ρ_k is the autocorrelation of the chain at lag k. A good rule of thumb for the ESS is for it to be 10% of the total number of samples.

MCMC diagnostics: Monte Carlo Standard Error (MCSE)

Another useful measure for the effective accuracy of the chain is the Monte Carlo standard error (MCSE).

- The standard deviation (SD) of the sample mean accross many replications is called the standard error and is estimated as $\text{SE} = \text{SD}/\sqrt{N}$
- So as the sample size N increases the SE decreases. In other words the bigger the sample, the more precise the estimate.

Extending this to MCMC chains, we substitute the sample size N with the ESS to get

$$\text{MCSE} = \text{SD}/\sqrt{\text{ESS}}$$

where SD is the standard deviation of the chain.

MCMC efficiency

There are a number of ways to attempt to improve efficiency in the MCMC process

1. Run chains in parallel
2. Adjust the sampling algorithm e.g., use Gibbs instead of Metropolis
3. Change model parameterisation (e.g., mean center the data for regression analysis)

Let's look at a demo of some of the diagnostics we've been discussing.

```
shinystan::launch_shinystan_demo()
```