ADVANCED BAYESIAN MODELING

USING MCMC IN PRACTICE:
OVERVIEW OF PRACTICAL MCMC

Review

Markov chain Monte Carlo (MCMC) is a simulation technique producing a sequence

$$\theta^0, \quad \theta^1, \quad \theta^2, \quad \ldots$$

of dependent variates (called *iterates*) drawn approximately from the posterior.

Basic algorithms include Gibbs sampling, Metropolis, and Metropolis-Hastings.

Using them properly requires choices, tuning, and monitoring.

Software

Instead of direct implementation, easier to use MCMC software:

- WinBUGS and OpenBUGS
- JAGS
- Stan
- Others (mcsim, PyMC, HBC, ...)

Ideally, all you should need to specify is a DAG model.

In practice, still need to make some decisions about the MCMC ...

- How many chains should I run?
- What starting values should I choose?
- What can I do about transient behavior?
- How do I know if/when the chains have converged?
- Which iterates should I use (for Bayesian inference)?
- How many iterates do I need?

Multiple Chains

Running several independent chains from different starting points is recommended:

- Allows better convergence diagnostics (later)
- Explores parameter space more effectively
- Aids discovery of problems with the model

Usually use 3 to 5 chains – using too many wastes burn-in iterations.

Usually run all chains in parallel, and for the same number of iterations.

Starting Points

Initialize the chains by specifying their starting values.

Ideally, starting values should be **overdispersed**: Much more extreme than values you expect from the posterior.

- Initialize some elements of parameter vector θ much larger than expected, others much smaller.
- Initialize them to different extremes for different chains.

Using overdispersed starting values enhances the benefits of using multiple chains:

- Convergence diagnostics are more likely to operate as intended (e.g., Gelman-Rubin – later).
- Multiple modes or other unexpected posterior features are more likely to be found.

In hierarchical models, usually only the top-level hyperparameters should be initialized.

Software usually offers auto-initialization for the other parameters (by sampling from the prior, or using a prior mean, or ...).

Warning: Starting values that are too extreme may cause software to crash.

Adaptation

Some simulation techniques (like Metropolis) must be tuned for good performance.

Most software packages provide (or offer) automatic tuning, called **adaptation**.

Requires an initial run, during which all sampling schemes are dynamically tuned based on the iterates generated.

Since tuning depends on the whole sequence generated, this sequence is not a Markov chain and may not have the posterior as a stationary distribution.

Use the tuning, but discard the initial run – it is not appropriate for Bayesian inference.

Convergence

For a good chain, the posterior is the limiting (long-run) distribution.

There could be transient (short-run) effects that do not reflect the posterior.

Suggests **burn-in**: Running the chain until transient effects are (almost) gone, then using only the iterations after this.

(Called **warm-up** in BDA3.)

Need methods to decide length of burn-in (later).

General MCMC process (after setting up DAG model):

1. Choose number of chains and overdispersed starting points.
2. Run iterations for adaptation (if needed).
3. Run more iterations, for initial burn-in and convergence monitoring.
4. Choose iterations after convergence to be used as the posterior sample.
5. Assess Monte Carlo error, and run more iterations until small enough.