The Journal of Open Source Software

compareMCMCs: An R package for studying MCMC efficiency

Perry de Valpine[1], **Sally Paganin**[1], **and Daniel Turek**[2]

1 University of California, Berkeley **2** Williams College

DOI: 10.21105/joss.03844

Software

- Review
- Repository
- Archive

Editor: Fabian Scheipl

Reviewers:

- @rowlandseymour
- @tbrown122387

Submitted: 18 October 2021
Published: 14 January 2022

Summary

Markov chain Monte Carlo (MCMC) algorithms are used to simulate from complicated probability distributions. MCMC is very widely used to implement Bayesian statistical analysis, where the distribution of interest (the "target distribution") is a posterior distribution of parameters given data. In this context, the posterior is known only up to a constant, so only relative probabilities (or densities) can be easily calculated, which is sufficient for MCMC to work. In Bayesian statistical analysis, MCMC algorithms for large or complex statistical models and data sets are sometimes run for minutes, hours or days, making them an analysis bottleneck, so there is a premium on efficiency. MCMC efficiency includes both computational speed and algorithmic mixing, which refers to how well the algorithm explores the posterior distribution from one iteration to the next. Computational speed may comprise one or more steps such as algorithm setup, MCMC "burn-in" or "warm-up" phases, and MCMC execution or "sampling."

There are many MCMC algorithms (also called "samplers") and software packages implementing them. Because MCMC samplers can be validly combined (e.g., iterated in sequence), for example with different samplers for different dimensions of a target distribution, there is an enormous space of MCMC methods. Invention of new methods, comparisons among methods, and theoretical study of MCMC mixing are all important areas of active research. Various software packages provide samplers such as Gibbs, adaptive random-walk Metropolis-Hastings, slice, Hamiltonian, multivariate ("block") or other variants of these, and others. Different MCMC algorithms can yield efficiencies that differ by orders of magnitude for a particular problem, with these variations in efficiency being problem-dependent.

The R package `compareMCMCs` provides a highly modular system for managing performance comparisons among MCMC software packages for purposes of research on MCMC methods. MCMC runs can take a long time, so the output ("samples") and components of computation time from a run are stored regardless of whether performance metrics are computed immediately. Arbitrary MCMC packages ("MCMC engines") can be added to the system by writing a simple plug-in or wrapper to manage inputs and outputs in a unified way. Conversions among model parameter names and/or different parameterizations can be provided to standardize across packages. Performance metrics are organized by model parameter (one result per parameter per MCMC engine), by MCMC (one result per MCMC engine), or arbitrarily (a user-defined list of metric results per MCMC engine). Built-in metrics include two methods of estimating effective sample size (ESS), posterior summaries such as mean and common quantiles, efficiency defined as ESS per computation time, rate defined as computation time per ESS, and minimum efficiency per MCMC. New metrics can be provided by a plug-in system and applied programmatically to a set of MCMC samples without re-running the MCMC engines. Finally, standardized graphical comparison pages can be generated in html. Built-in graphical outputs include figures comparing MCMC efficiency and/or rate on a per-parameter or per-MCMC basis as well as comparing posterior distributions. New graphical

outputs can be provided by a plug-in system. In summary, compareMCMCs is modular and extensible for running new MCMC engines on comparable problems, for creating new metrics of interest (e.g., posterior summaries or effective sample size estimated in different ways), and for creating new graphical comparison outputs into a report.

Use of compareMCMCs supports but does not require a primary role for MCMCs created with the nimble (de Valpine et al., 2017, 2021) package for hierarchical statistical models. That is because nimble provides greater flexibility than other packages to customize its MCMC system, configuring which samplers will operate on which parts of a model and/or writing new samplers. Thus, it is of interest to compare multiple MCMC methods all implemented within nimble. Furthermore, nimble uses a model language that is a dialect of that used by WinBUGS, OpenBUGS, MultiBUGS, and JAGS (Goudie et al., 2020; D. Lunn et al., 2009; D. J. Lunn et al., 2000; Plummer & others, 2003). These packages are often called from R via packages such as R2WinBUGS (Sturtz et al., 2005), rjags (Plummer, 2019), and jagsUI (Kellner, 2019). Therefore, for fully compatible models, comparisons between nimble and JAGS can be run in compareMCMCs from the same model and data specifications. A plug-in is also provided for Stan via rstan (Stan Development Team, 2020), and the extension system to plug in new MCMC engines is clearly documented.

Statement of need

Many other packages run MCMC algorithms and/or post-process MCMC results, but compareMCMCs is distinct in its goal of supporting MCMC research by comparing MCMC methods. Packages that run MCMC from R are documented on the "Cran Task View" page for "Bayesian Inference" (Park, 2021) of the Comprehensive R Archive Network (CRAN). Some popular general packages include those listed above as well as others such as MCMCpack (Martin et al., 2011) and LaplacesDemon (Statisticat & LLC., 2021). Furthermore, there are MCMC engines based in Python, such as PyMC (Salvatier et al., 2016), and other languages. These may be called via appropriate interfaces from R to other languages.

Of the packages listed on the "Bayesian Inference" Task View, only the SamplerCompare (Thompson, 2011) package appears to specifically support the goal of comparing MCMC performance. However, this package can only compare MCMC samplers that have exactly one scalar tuning parameter, target distributions that are continuous with constant dimension, and are implemented within the package.

Packages for post-processing of MCMC samples (e.g., coda (Plummer et al., 2006), BayesPostEst (Scogin et al., 2019), and MCMCvis (Youngflesh, 2018)) aim to provide features for scientific summary and presentation of results, whereas compareMCMCs provides features for comparisons of algorithm performance across packages. Assessing MCMC performance is not simply a matter of computational benchmarking. For example, effective sample size is itself a non-trivial property to estimate by statistical methods, different metrics may be of interest for different purposes, and consistency of algorithm results between different MCMC engines can only be determined statistically, i.e. within simulation error. Therefore, the features needed for comparing MCMC performance are distinct from those needed for presenting scientific results based on MCMC.

Acknowledgements

We thank other developers who have contributed to nimble, in which an early version of the features of compareMCMCs was first drafted and from which its design needs became clearer. Daniel Turek wrote the first version of what evolved into compareMCMCs, and Christopher Paciorek contributed ideas along the way.

References

de Valpine, P., Paciorek, C., Turek, D., Michaud, N., Anderson-Bergman, C., Obermeyer, F., Wehrhahn Cortes, C., Rodrìguez, A., Temple Lang, D., & Paganin, S. (2021). *NIMBLE: MCMC, particle filtering, and programmable hierarchical modeling* (Version 0.11.1) [Computer software]. https://doi.org/10.5281/zenodo.1211190

de Valpine, P., Turek, D., Paciorek, C., Anderson-Bergman, C., Temple Lang, D., & Bodik, R. (2017). Programming with models: Writing statistical algorithms for general model structures with NIMBLE. *Journal of Computational and Graphical Statistics*, *26*, 403–413. https://doi.org/10.1080/10618600.2016.1172487

Goudie, R. J. B., Turner, R. M., De Angelis, D., & Thomas, A. (2020). MultiBUGS: A parallel implementation of the BUGS modelling framework for faster bayesian inference. *Journal of Statistical Software*, *95*(7). https://doi.org/10.18637/jss.v095.i07

Kellner, K. (2019). *jagsUI: A wrapper around 'rjags' to streamline 'JAGS' analyses*. CRAN. https://cran.r-project.org/package=jagsUI

Lunn, D. J., Thomas, A., Best, N., & Spiegelhalter, D. (2000). WinBUGS-a bayesian modelling framework: Concepts, structure, and extensibility. *Statistics and Computing*, *10*(4), 325–337. https://doi.org/10.1023/A:1008929526011

Lunn, D., Spiegelhalter, D., Thomas, A., & Best, N. (2009). The BUGS project: Evolution, critique and future directions. *Statistics in Medicine*, *28*(25), 3049–3067. https://doi.org/10.1002/sim.3680

Martin, A. D., Quinn, K. M., & Park, J. H. (2011). MCMCpack: Markov chain monte carlo in R. *Journal of Statistical Software*, *42*(9), 22. https://doi.org/10.18637/jss.v042.i09

Park, J. H. (2021). *CRAN Task View: Bayesian Inference*. CRAN. https://CRAN.R-project.org/view=Bayesian

Plummer, M. (2019). *Rjags: Bayesian graphical models using MCMC*. CRAN. https://cran.r-project.org/package=rjags

Plummer, M., Best, N., Cowles, K., & Vines, K. (2006). CODA: Convergence diagnosis and output analysis for MCMC. *R News*, *6*(1), 7–11. https://journal.r-project.org/archive/

Plummer, M., & others. (2003). JAGS: A program for analysis of bayesian graphical models using gibbs sampling. *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*, *124*, 1–10.

Salvatier, J., Wiecki, T. V., & Fonnesbeck, C. (2016). Probabilistic programming in python using PyMC3. *PeerJ Computer Science*, *2*, e55. https://doi.org/10.7717/peerj-cs.55

Scogin, S., Karreth, J., Beger, A., & Williams, R. (2019). BayesPostEst: An r package to generate postestimation quantities for bayesian MCMC estimation. *Journal of Open Source Software*, *4*(42), 1722. https://doi.org/10.21105/joss.01722

Stan Development Team. (2020). *RStan: The R interface to Stan*. http://mc-stan.org/

Statisticat, & LLC. (2021). *LaplacesDemon: Complete environment for bayesian inference*. Bayesian-Inference.com. https://web.archive.org/web/20150206004624/http://www.bayesian-inference.com/software

Sturtz, S., Ligges, U., & Gelman, A. (2005). R2WinBUGS: A package for running WinBUGS from r. *Journal of Statistical Software*, *12*(3), 1–16. http://www.jstatsoft.org

Thompson, M. B. (2011). Introduction to SamplerCompare. *Journal of Statistical Software*, *43*(12), 1–10. http://www.jstatsoft.org/v43/i12/

Youngflesh, C. (2018). MCMCvis: Tools to visualize, manipulate, and summarize MCMC output. *Journal of Open Source Software*, *3*(24), 640. https://doi.org/10.21105/joss.00640