# nimbleHMC: An R package for Hamiltonian Monte Carlo sampling in nimble

13 August 2022

## Summary

Markov chain Monte Carlo (MCMC) algorithms are widely used for fitting hierarchical (graphical) models to observed data, and more generally for simulating from high-dimensional probability distributions. MCMC is the predominant tool used in Bayesian analyses, where the distribution of interest (the "target distribution") is defined as the posterior distribution of the unknown model parameters conditional on the data. MCMC does not specify a single algorithm, but rather a family of algorithms admitting any assignment of valid sampling techniques ("samplers") to the unobserved model dimensions. There exist a vast and diverse landscape of valid samplers to draw upon, which differ significantly in their underlying approaches to the sampling problem, complexity, autocorrelation of the samples produced, and applicability.

Hamiltonian Monte Carlo (HMC; Brooks et al. 2011) is one such sampling technique which can be applied to any subset of continuous-valued model dimensions. HMC uses the gradient of the target distribution to generate large transitions (in parameter space) in the output sequence of samples. This results in low autocorrelation, and therefore high information content. That is, the samples generated using HMC are more likely to be highly informative about the target distribution of interest, relative for example to an equal-length sequence of highly autocorrelated samples. This rich information content does not come freely, however, as calculating gradients of the target distribution is computationally expensive.

There exist numerous software packages which provide implementations of MCMC for mainstream use such as `nimble` (Valpine et al. 2017), `jags` (Plummer and others 2003), `pyMC` (Fonnesbeck et al. 2015), and `Stan` (Carpenter et al. 2017). Each such package provides a language for specifying general higherarchical model structures, and supplying data. Following specification of the problem, each package generates an MCMC algorithm which specifically samples from the target posterior distribution of the specified model, and executes this algorithm to generate a sequence of samples from this distribution. These packages differ, however, in their approaches to sampler assignment for each unobserved model dimension. As sampling techniques vary in terms of computational demands and the quality of the samples produced, the effectiveness of the MCMC algorithms may vary depending on the software used, and the particular model at hand. Each software package provides a valid, but distinct approach for assigning samplers to define the MCMC algorithm.

Among general-purpose MCMC software packages, `nimble` uniquely provides the ability to specify which samplers are applied to each model dimension. Prior to generating an executable MCMC algorithm, `nimble` has the intermediate stage of MCMC configuration. At configuration time, users may select any valid assignment of samplers to each unosberved model dimension, mixing and matching between those samplers provided with `nimble`. The base `nimble` package provides a variety of non-derivative-based samplers, including random walk Metropolis-Hastings (Robert and Casella 1999), slice sampling (Neal 2003), conjugate samplers (George, Makov, and Smith 1993), and many others. After configruation is finished, an MCMC algorithm is generated according to the sampler assignments therein, and executed to generate a sequence of samples..

The `nimbleHMC` package provides an implementation of HMC sampling which is compatible for use within `nimble`. Specifically, `nimbleHMC` implements the No-U-Turn variety of HMC (HMC-NUTS; Hoffman, Gelman, and others 2014), which removes the necessity of hand-specifying tuning parameters of the HMC sampler. Using `nimbleHMC`, HMC samplers can be be assigned to any subset of continuous-valued model dimensions at the time of `nimble`'s MCMC configuration, which may be used in combination with any other samplers provided with the base `nimble` package.

# Statement of need

HMC is recognized as a state-of-the-art MCMC sampling strategy, capable of efficiently generating samples with strong inferential power. A testimony to this, packages such as `Stan` make use exclusively of HMC sampling. As a result, however, `Stan` is unable to operate on models with discrete (non-continuous) valued dimensions on account of the non-applicability of HMC. Models with discrete-valued dimensions arise in a broad range of statistical motifs including hidden Markov models, finite mixture models, and generally in the presence of unobserved categorical data, among others (Bartolucci, Pandolfi, and Pennoni 2022). In constrast, other mainstream MCMC packages such as `WinBUGS`, `OpenBUGS` and `jags` have the ability to sample discrete model dimensions, but do not implement HMC. This leaves a gap, as there is no support for applying HMC sampling to continuous-valued dimensions of hierchical models which also contain discrete dimensions.

It is an open question regarding what is an optimal MCMC algorithm, that assignment combination of samplers which maximizes the information content (in the sequence of samplers) generated per unit runtime of the MCMC. This metric can be quantified as MCMC efficiency (Turek et al. 2017), but what assignment of samplers maximizes this metric is a difficult and open question (Ponisio et al. 2020). For that reason, the ability to mix-and-match samplers from among as large a pool of candidates as possible is important from both practical and a theoretical standpoints. Indeed, there even exist packages such as `compareMCMCs` (Valpine, Paganin, and Turek 2022), the purpse of which is to compare the relative performance of distinct MCMC algorithms.

The `nimble` package uniquely provides the ability to custom-specify the assignment of sampler algorithms, which allows the exploration and the study of efficiency approachs to MCMC. Here, the `nimbleHMC` package augments the `nimble` package by providing an HMC sampler suitable for use within `nimble`'s MCMC. This fills the gap, allowing HMC samplers to operate alongside the continuous and discrete sampling algorithms avaialble in `nimble`.

`Gala` is an Astropy-affiliated Python package for galactic dynamics. Python enables wrapping low-level languages (e.g., C) for speed without losing flexibility or ease-of-use in the user-interface. The API for `Gala` was designed to provide a class-based and user-friendly interface to fast (C or Cython-optimized) implementations of common operations such as gravitational potential and force evaluation, orbit integration, dynamical transformations, and chaos indicators for nonlinear dynamics. `Gala` also relies heavily on and interfaces well with the implementations of physical units and astronomical coordinate systems in the `Astropy` package (**???**) (`astropy.units` and `astropy.coordinates`).

`Gala` was designed to be used by both astronomical researchers and by students in courses on gravitational dynamics or astronomy. It has already been used in a number of scientific publications (**???**) and has also been used in graduate courses on Galactic dynamics to, e.g., provide interactive visualizations of textbook material (**???**). The combination of speed, design, and support for Astropy functionality in `Gala` will enable exciting scientific explorations of forthcoming data releases from the *Gaia* mission (**???**) by students and experts alike.

# Mathematics

Single dollars ($) are required for inline mathematics e.g. $f(x) = e^{\pi/x}$

Double dollars make self-standing equations:

$$\Theta(x) = \left\{\begin{array}{l} 0\textrm{ if } x < 0\cr 1\textrm{ else}\end{array}\right.$$

You can also use plain LaTeXfor equations

$$\hat f(\omega) = \int_{-\infty}^{\infty} f(x) e^{i\omega x} dx \tag{1}$$

and refer to Equation 1 from text.

# Citations

Citations to entries in paper.bib should be in rMarkdown format.

If you want to cite a software repository URL (e.g. something on GitHub without a preferred citation) then you can do it with the example BibTeX entry below for (**???**).

# Figures

# Acknowledgements

We acknowledge contributions from Brigitta Sipocz, Syrtis Major, and Semyeong Oh, and support from Kathryn Johnston during the genesis of this project.

# References

Bartolucci, Francesco, Silvia Pandolfi, and Fulvia Pennoni. 2022. "Discrete Latent Variable Models." *Annual Review of Statistics and Its Application* 9: 425–52.

Brooks, Steve, Andrew Gelman, Galin Jones, and Xiao-Li Meng. 2011. *Handbook of Markov Chain Monte Carlo*. CRC press.

Carpenter, Bob, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. 2017. "Stan: A Probabilistic Programming Language." *Journal of Statistical Software* 76 (1).

Fonnesbeck, Chris, Anand Patil, David Huard, and John Salvatier. 2015. "PyMC: Bayesian Stochastic Modelling in Python." *Astrophysics Source Code Library*, ascl–1506.

George, Edward I, UE Makov, and AFM Smith. 1993. "Conjugate Likelihood Distributions." *Scandinavian Journal of Statistics*, 147–56.

Hoffman, Matthew D, Andrew Gelman, and others. 2014. "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo." *J. Mach. Learn. Res.* 15 (1): 1593–1623.

Neal, Radford M. 2003. "Slice Sampling." *The Annals of Statistics* 31 (3): 705–67.

Plummer, Martyn, and others. 2003. "JAGS: A Program for Analysis of Bayesian Graphical Models Using Gibbs Sampling." In *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*, 124:1–10. 125.10. Vienna, Austria.

Ponisio, Lauren C, Perry de Valpine, Nicholas Michaud, and Daniel Turek. 2020. "One Size Does Not Fit All: Customizing Mcmc Methods for Hierarchical Models Using Nimble." *Ecology and Evolution* 10 (5): 2385–2416.

Robert, Christian P, and George Casella. 1999. "The Metropolis—Hastings Algorithm." In *Monte Carlo Statistical Methods*, 231–83. Springer.

Turek, Daniel, Perry de Valpine, Christopher J Paciorek, and Clifford Anderson-Bergman. 2017. "Automated Parameter Blocking for Efficient Markov Chain Monte Carlo Sampling." *Bayesian Analysis* 12 (2): 465–90.

Valpine, Perry de, Sally Paganin, and Daniel Turek. 2022. "CompareMCMCs: An R Package for Studying Mcmc Efficiency." *Journal of Open Source Software* 7 (69): 3844.

Valpine, Perry de, Daniel Turek, Christopher J Paciorek, Clifford Anderson-Bergman, Duncan Temple Lang, and Rastislav Bodik. 2017. "Programming with Models: Writing Statistical Algorithms for General Model Structures with Nimble." *Journal of Computational and Graphical Statistics* 26 (2): 403–13.