A simple method for automated equilibration detection in molecular simulations

John D. Chodera[1,*]

[1] *Computational Biology Program, Sloan Kettering Institute, Memorial Sloan Kettering Cancer Center, New York, NY 10065*

(Dated: February 7, 2015)

Molecular simulations intended to compute equilibrium properties (such as molecular dynamics and Metropolis Monte Carlo simulations) are often initiated from configurations that are highly dissimilar to equilibrium samples, a practice which typically generates a distinct initial transient in various mechanical observables computed over the timecourse of the simulation. Traditional practice in simulation data analysis recommends this initial transient portion be discarded to *equilibration*, but no simple, general, and automated procedure for this process exists. Here, we consider a conceptually simple, automated, easy-to-implement procedure that does not make strict assumptions about the distribution of the observable of interest, in which the equilibration region is chosen to maximize the number of effectively uncorrelated samples in the production portion used for computing equilibrium averages. We present a simple reference Python implementation of this procedure and illustrate its application to both synthetic and real simulation data.

Keywords: molecular dynamics (MD); Monte Carlo (MC); Markov chain Monte Carlo (MCMC); equilibration; timeseries analysis; statistical inefficiency; integrated autocorrelation time

INTRODUCTION

Molecular simulations use Markov chain Monte Carlo (MCMC) techniques [1] to sample configurations x from an equilibrium distribution $\pi(x)$, either exactly (using Monte Carlo methods) or approximately (using molecular dynamics simulations without Metropolization).

Due to the nature of the equilibrium distribution $\pi(x)$ and the difficulty in producing a sufficiently good guess at a typical equilibrium configuration, these molecular simulations are often started from highly atypical initial conditions. For example, simulations of biopolymers might be initiated from a fully extended conformation unrepresentative of behavior in solution, or a model derived from a fit to diffraction data of a cryocooled crystal; solvated systems may be prepared by periodically replicating a small solvent box that was equilibrated with a different forcefield under different conditions from the current simulation, thus yielding atypical densities; liquid mixtures or lipid bilayers may be constructed by using methods that fulfill spatial constraints but create locally aytpical geometries (e.g. PackMol [2]) that may require long simulation times to relax to typical configurations.

As a result, common practice in molecular simulations is to discard some initial portion of the trajectory to "equilibration" (also called *burn-in*[1] in MCMC literature [3]). While discarding an initial portion of the dataset is strictly unnecessary for the time-average of quantities of interest to converge to the desired expectations [3, 4], this process often allows the practitioner to avoid what would otherwise be extremely long run times to eliminate the bias in computed properties in finite-length simulations due to the initial atypical starting conditions.

As an illustrative example of this effect, consider the simulation shown in **Figure 1**, in which a simulation of liquid argon is started at an atypical density and allowed to equilibrate to its equilibrium density (see caption for detailed description of simulation methods). [JDC: Use TIP3P water instead?] The expectation of the running average of the density over many realizations of this procedure (**Figure 1b**) significantly deviates from the actual expectation, which would lead to biased estimates unless simulations were sufficiently long to eliminate this starting point dependent bias. Note that this significant bias is present because the *same* atypical starting condition is used for every realization of this simulation process.

For the purposes of this note, we presume that the goal is to compute some form of equilibrium expectation, <A> from a timeseries average:

$$\hat{A} \approx \int_0^T dt\, A(x(t)) \tag{1}$$

EFFECTIVE NUMBER OF UNCORRELATED SAMPLES

Consider a successively sampled configurations x_t, $t = 1, \ldots, T$ from a molecular simulation. We presume we are interested in the timeseries of a mechanical property $A(x)$, which we denote $A_1, \ldots, A_T$ with $A_t \equiv A(x_t)$. If we average this quantity

THE IDEA

Suppose we choose some arbitrary time t_0 and discard all samples $t \in [0, t_0)$ to equilibration, keeping

* Corresponding author; john.chodera@choderalab.org

[1] The term *burn-in* comes from the field of electronics, in which a short "burn-in" period is used to ensure that a device is free of faulty components—which often fail quickly—and is operating normally [3].

FIG. 1. **Illustration of the motivation for discarding data to equilibration in computing expectations from molecular simulations.** This is text.

$[t_0, T]$ as the dataset to analyze. How much data remains? We can determine this by computing the statistical inefficiency g_{t_0} for the interval $[t_0, T]$, and computing the effective number of uncorrelated samples $N_{\mathrm{eff}}(t_0) \equiv (T - t_0 + 1)/g_{t_0}$. If we start at $t_0 \equiv T$ and move t_0 to earlier and earlier points in time, we expect that the effective number of uncorrelated samples $N_{\mathrm{eff}}(t_0)$ will continue to grow until we start to include the highly atypical initial data. At that point, the integrated autocorrelation time τ (and hence the statistical inefficiency g) will greatly increase, and the effective number of samples N_{eff} will start to plummet.

ILLUSTRATION

All molecular simulations were performed with OpenMM 6.2 [**?**] using the Python API. All scripts used to run simulations, analyze data, and generate plots—along with the simulation data itself—are availabile on GitHub at `http://github.com/choderalab/automatic-equilibration-detection`.

ACKNOWLEDGMENTS

We are grateful to William C. Swope (IBM Almaden Research Center), Kyle A. Beauchamp (MSKCC), and Robert C. McGibbon (Stanford University) for valuable discussions on this topic, and Joshua L. Adelman (University of Pittsburgh) for helpful feedback and encouragement.

[1] J. S. Liu, *Monte Carlo strategies in scientific computing*, 2nd ed. ed. (Springer-Verlag, New York, 2002).

[2] L. Martínez, R. Andrade, E. G. Birgin, and J. M. Martínez, J. Chem. Theor. Comput. **30**, 2157 (2009).

[3] S. Brooks, A. Gelman, G. L. Jones, and X.-L. Meng, in *Handbook of Markov chain Monte Carlo*, *Chapman & Hall/CRC Handbooks of Modern Statistical Methods* (CRC Press, ADDRESS, 2011), Chap. Introduction to Markov chain Monte Carlo.

[4] C. Geyer, Burn-in is unnecessary., `http://users.stat.umn.edu/~geyer/mcmc/burn.html`.