Parallelized Nested Sampling

R. Wesley Henderson and Paul M. Goggans

Department of Electrical Engineering, University of Mississippi

Abstract. One of the important advantages of nested sampling as an MCMC technique is its ability to draw representative samples from multimodal distributions and distributions with other degeneracies. This coverage is accomplished by maintaining a number of so-called live samples within a likelihood constraint. In usual practice, at each step, only the sample with the least likelihood is discarded from this set of live samples and replaced. In [1], Skilling shows that for a given number of live samples, discarding only one sample yields the highest precision in estimation of the log-evidence. However, if we increase the number of live samples, more samples can be discarded at once while still maintaining the same precision.

For computer code running only serially, this modification would considerably increase the wall clock time necessary to reach convergence. However, if we use a computer with parallel processing capabilities, and we write our code to take advantage of this parallelism to replace multiple samples concurrently, the performance penalty can be eliminated entirely and possibly reversed. In this case, we must use the more general equation in [1] for computing the expectation of the shrinkage distribution:

$$\mathrm{E}[-\log t] = (N_r - r + 1)^{-1} + (N_r - r + 2)^{-1} + \cdots + N_r^{-1},$$

for shrinkage t with N_r live samples and r samples discarded at each iteration. The equation for the variance

$$\mathrm{Var}(-\log t) = (N_r - r + 1)^{-2} + (N_r - r + 2)^{-2} + \cdots + N_r^{-2}$$

is used to find the appropriate number of live samples N_r to use with $r > 1$ to match the variance achieved with N_1 live samples and $r = 1$.

In this paper, we show that by replacing multiple discarded samples in parallel, we are able to achieve a more thorough sampling of the constrained prior distribution, reduce runtime, and increase precision.

Keywords: Nested Sampling, Parallel Computing, MCMC
PACS: 02.50.Tt

INTRODUCTION

The application of Bayesian inference to engineering problems often involves distributions that are multimodal, awkwardly shaped, or both. Many Markov chain Monte Carlo (MCMC) methods are ill-suited for dealing with these troublesome distributions; however, certain MCMC techniques perform acceptably under these conditions. One such technique is nested sampling [2].

Nested sampling is primarily a technique to estimate Bayesian evidence. For a set of hypotheses represented by parameters $\mathbf{B}$ and observed data $\mathbf{A}$, the evidence, Z, is the sum of the product of the prior distribution,

$$\pi(\mathbf{B}) = p(\mathbf{B}|I), \tag{1}$$

and likelihood function,

$$L(\mathbf{B}) \propto p(\mathbf{A}|\mathbf{B}, I), \tag{2}$$

for all possible hypotheses. If the parameters are continuous, this sum becomes an integral,

$$Z = \int \pi(\mathbf{B}) L(\mathbf{B})\, \mathrm{d}\mathbf{B}. \tag{3}$$

It is convenient to assume that the prior and likelihood have been reparameterized so that $\pi(\mathbf{B})$ is uniform on the unit hypercube.

This integral is usually impossible to evaluate analytically and difficult to evaluate numerically. Nested sampling addresses this issue by recasting the integral in (3) as a more tractable one-dimensional integral. Instead of considering the prior distribution directly, nested sampling instead deals with the prior mass, defined as

$$X(\lambda) = \int_{L(\mathbf{B})>\lambda} \pi(\mathbf{B})\, \mathrm{d}\mathbf{B}. \tag{4}$$

In (4), the λ parameter represents a likelihood threshold (or constraint), and $X(\lambda)$ is the proportion of the prior distribution that lies within this likelihood constraint. The likelihood can be written as a function of the prior mass, and simply returns the likelihood constraint:

$$L(X(\lambda)) = \lambda. \tag{5}$$

In this way, the evidence can be computed as

$$Z = \int_0^1 L(X)\,\mathrm{d}X. \tag{6}$$

The likelihood for a particular hypothesis can be determined exactly; however, the prior mass contained in $L(\mathbf{B}) > \lambda$ is impossible to compute analytically. To get around this limitation, the integral in (6) is computed numerically and becomes

$$Z = \sum_{i=1}^{m} w_i L_i, \tag{7}$$

in which the change in the abscissa is expressed as $w_i = X_i - X_{i-1}$. Nested sampling proceeds by computing each element of the sum in (7) for monotonically increasing likelihood constraints.

While the prior mass within a likelihood constraint cannot be exactly determined, the shrinkage t, i.e., the ratio of the prior mass within λ_i to the prior mass within λ_{i-1}, is known to be beta-distributed,

$$t \sim \mathrm{Beta}(N, r), \tag{8}$$

with parameters N and r. Nested sampling maintains a population of N so-called live samples uniformly distributed within the likelihood constraint, and at step i the r samples with the lowest likelihood in the live sample population are collected and discarded. The likelihood constraint λ at step i is set to the greatest likelihood in the set of r discarded samples. The geometric mean of the shrinkage distribution serves as an adequate estimate of the actual shrinkage. Since we are often more interested in the log-evidence than the evidence, we want to compute the log of the geometric mean:

$$\log G_t = \mathrm{E}[\log t], \tag{9}$$

$$\mathrm{E}[-\log t] = \sum_{n=0}^{r-1} \frac{1}{N_r - n}. \tag{10}$$

THEORY

If the nested sampling process proceeds with too few live samples and the likelihood function is multimodal, those live samples can get effectively "locked out" of portions of the likelihood-constrained prior that are important for accurate numerical estimation of (6). A large value of N, i.e., a large population of likelihood-constrained samples, provides robustness when dealing with multimodal likelihood functions. While a sufficiently large N ensures virtually complete coverage of all modes, it also increases the number of iterations necessary to reach convergence. Taking another look at (10), we see that as the number of live samples N is increased, the shrinkage of the prior mass between iterations becomes smaller, implying that more iterations will be required before the region of the prior mass containing most of the posterior distribution is reached.

This paper proposes to parallelize nested sampling in order to gain the benefits of a large population of live samples while minimizing the time necessary to complete the resultant nested sampling run. While the usual approach with nested sampling is to discard and replace only one sample at each iteration ($r = 1$), larger values of r can be used. This idea was proposed previously by Burkoff, et al. [3]. This paper's primary contribution is to examine the effect of concurrently replacing multiple samples on the variance of the log-evidence estimate and on the total execution time of nested sampling.

From (10), simply increasing r leads to a greater mean of the log of the shrinkage distribution, giving fewer steps to reach convergence. However, the variance of the log of the shrinkage distribution [1],

$$\mathrm{Var}(-\log t) = (N_r - r + 1)^{-2} + (N - r + 2)^{-2} + \cdots + N^{-2} = \sum_{n=0}^{r-1} \left(\frac{1}{N_r - n}\right)^2, \tag{11}$$

also increases with r. Greater variance in the shrinkage distribution leads to less precision in the evidence value estimate, so we need to find a way to tune the value of N such that the variance remains constant with increased r.

Scaling the Number of Live Samples

The precision in the estimation of the evidence is unchanged if the following relationship is satisfied:

$$\mathrm{Var}(-\log t_1) = \mathrm{Var}(-\log t_r). \tag{12}$$

In (12) and in the following equations, the subscripts on t and N correspond to the value of r. Substituting (11) into (12) and solving for N_1 yields

$$N_1 = \frac{N_r}{\sqrt{1+\sum_{n=1}^{r-1}\left(\frac{N_r}{N_r-n}\right)^2}}. \tag{13}$$

If we assert that the total number of live samples is always much greater than the number of samples discarded and replaced, i.e. $N_r >> r-1$, (13) reduces to

$$N_1 \approx \frac{N_r}{\sqrt{r}}. \tag{14}$$

Therefore, to maintain variance as seen in the $\{N, r=1\}$ case while increasing r, the number of live samples must be scaled as

$$N_r = \sqrt{r}N_1. \tag{15}$$

Returning to the mean, recall that

$$\mathrm{E}[-\log t_1] = \frac{1}{N_1} \approx \frac{\sqrt{r}}{N_r}, \tag{16}$$

and applying our assumption that $N_r >> r-1$ gives

$$\mathrm{E}[-\log t_r] = \frac{1}{N_r} + \sum_{n=1}^{r-1}\frac{1}{N_r-n} \approx \frac{r}{N_r}. \tag{17}$$

The ratio of the means is then

$$\frac{\mathrm{E}[-\log t_r]}{\mathrm{E}[-\log t_1]} = \sqrt{r}, \tag{18}$$

and the mean for the case with $r \neq 1$ is

$$\mathrm{E}[-\log t_r] = \sqrt{r}\,\mathrm{E}[-\log t_1]. \tag{19}$$

Ultimately, we find that if we increase the value of r and increase the value of N to maintain the same variance, we can achieve a greater shrinkage between iterations compared to the $r = 1$ case.

MCMC Calls

A common stopping criterion for nested sampling is when the number of iterations i becomes greater than the information H times the number of live samples N. A scaling constant can be used to ensure the desired compression is acheived. Adapting this criterion to the case in which $r \neq 1$, we have

$$ri > HN_r = \sqrt{r}HN_1. \tag{20}$$

From the inequality in (20), we can see that over the course of the nested sampling run, we will need to replace at least $\sqrt{r}HN_1$ samples using an MCMC method. Due to the nature of nested sampling, the vast majority of the total execution time of any nested sampling implementation is spent in MCMC calls; therefore, the amount of time necessary to complete a nested sampling run is strongly correlated with the number of MCMC calls required. The execution time t is then bounded as

$$\mathrm{Time}\left\{\frac{HN_1}{\sqrt{r}}\right\} \leq t \leq \mathrm{Time}\left\{\sqrt{r}HN_1\right\}. \tag{21}$$

IMPLEMENTATION

To test this idea, we use a C++ implementation of nested sampling using Galilean Monte Carlo (GMC). GCC 4.7.3 was used to compile the code, and the C++11 standard's [4] implementation of threading was used to run multiple GMC instances at once.

Various modifications must be made to the standard nested sampling algorithm in order to discard and replace multiple samples at once. Algorithm 1 details the modified routine used in our implementation.

Algorithm 1 Nested Sampling

Draw $\theta_1, \theta_2, \cdots, \theta_N$ from the prior
Calculate likelihood L_i for each sample.
$\log Z \leftarrow -\infty$
$\log X_0 \leftarrow \mathrm{E}[-\log t]$
for $i \leftarrow 1, M$ **do**
  `PartialSort`$(\boldsymbol{\theta}, r)$ ▷ Use the partial sort algorithm to sort at least the first r samples by log-likelihood.
  $\mathscr{L} \leftarrow L_r$ ▷ Set the likelihood constraint as the r-th log-likelihood value.
  $\log X_i \leftarrow \log X_i - \mathrm{E}[-\log t]$
  $w_i \leftarrow \log(\exp(X_{i-1}) - \exp(X_i))$ ▷ Increment the prior width.
  $\log Z \leftarrow \log Z + \mathscr{L} w_i$ ▷ Increment the log-evidence.
  $j_{(1,\cdots,r)} \sim U(r, N)$ ▷ Choose r random integer indexes j from a uniform distribution.
  Declare r threads.
  for $k \leftarrow 1, r$ **do**
    Start thread k.
    $\theta_k \leftarrow \mathtt{MCMC}(\theta_{j[k]}, \mathscr{L})$ ▷ Use MCMC to replace θ_k with new sample from $\mathscr{L}$-constrained prior.
  end for
  Block main thread until child threads are finished.
end for

Nested sampling requires the r live samples with the lowest likelihood to be discarded and replaced. When $r = 1$, this can be accomplished by simply finding the minimum of the likelihood values in the live sample population. However, when $r > 1$, a more sophisticated minimization algorithm is required. Our implementation uses a so-called "partial" quicksort [5] routine. This routine proceeds like normal quicksort until at least the first r values in the list are sorted, at which point the routine ends. This method of partially sorting the samples provides a performance increase over the brute-force method of finding multiple minimums or complete sorting.

The idea of Galilean Monte Carlo is explained in [1] and [6]. As a brief overview, GMC evolves a sample with M parameters by treating it as a particle in motion, where the sample's parameter values are analogous to the particle's Cartesian coordinates in $\mathscr{R}^M$. As the particle encounters boundaries (i.e., equal-likelihood contours), it specularly reflects off the boundaries. The ultimate goal of this motion is to explore the likelihood-constrained parameter space thoroughly enough to come to an end point that is independent of the starting position while also maintaining detailed balance.

Two of the challenges typically encountered when implementing GMC are finding a useful time step value and dealing with significantly non-spherical likelihood contours. Our implementation deals with the time step issue by tuning the time step using the initial particle velocity such that the particle can make about five moves from boundary to boundary. We address the issue of encountering a potentially non-spherical distribution by scaling the initial velocity with a so-called "Galilean semi-metric" [1].

Computational Overhead

In this implementation, there are several potential sources of overhead. These include

- spawning and killing threads,
- the partial sort routine,
- and the possibility that r is greater than the number of CPU cores available.

TABLE 1. Example results

N	r	Mean $\log Z$	Stdev $\log Z$	Mean time (s)
20	1	-46.3	1.447	1.88
28	2	-42.6	1.385	1.80
40	4	-43.5	1.312	2.02
57	8	-43.6	3.367	2.32
200	1	-42.0	1.179	16.7
280	2	-42.5	0.892	17.8
400	4	-42.5	0.813	19.6

The last item in that list can be mitigated by ensuring that r is not set to be greater than the number of CPU cores and that the nested sampling program is the only thing running on the computer.

EXAMPLE

We test our new method using a simple example inspired by an example employed by Skilling [2]. This example uses a 10-dimensional Gaussian distribution for the likelihood function, and a uniform prior distribution on $(-1,1)$. The log-likelihood is defined as

$$\log L = -\frac{\sum_{i=1}^{10}(\theta_i - \mu_i)^2}{2\sigma^2}, \tag{22}$$

with $\mu_i = 0$ and $\sigma = 0.01$.

We explore several combinations of N and r values, and run the nested sampling routine 10 times with that configuration. Table 1 shows the sample mean and standard deviation for the log-evidence over these runs, as well as the mean execution time for the runs.

DISCUSSION AND CONCLUSIONS

These tests are run in two sets, the first starting with $N = 20$ and the second starting with $N = 200$. As r increases, N is scaled according to (15). Table 1 shows that for all but one case the standard deviation in the log-evidence slightly decreases as r and N are increased. This trend supports our contention that scaling N according to (15) allows for parallel computation without sacrificing precision. In fact, the decreasing trend in standard deviation coupled with the negligible change in run time suggest that this technique can be used to achieve more precision in the evidence estimate without increasing run time. Also, by increasing the number of live samples, the nested sampling routine should be able to more completely sample from multimodal distributions with no additional run time.

In the case in which $N = 57$ and $r = 8$, our assertion that $N_r >> r - 1$ is no longer very accurate; thus, scaling N using (15) is possibly no longer sufficient to maintain the same variance in shrinkage distribution.

The time per run stays about the same with each comparable configuration, and as r and N increase, we see that the run time is far below the worst case as described in (21). While these results do not confirm our contention that this method would reduce the overall run-time for nested sampling, we think that future implementations with the goal of further reducing overhead might see significant reductions in run-time. The test case explored here involves a very simple likelihood function, and as a result, it is possible that the time required to compute the likelihood is eclipsed by the time required to spawn and kill computational threads. This being the case, a more computationally complex likelihood function may see a greater reduction in run time using this method.

ACKNOWLEDGMENTS

This work was supported by a grant from the U.S. Department of Transportation, Research and Innovative Technology Administration, administered by the National Center for Intermodal Transportation for Economic Competitiveness (NCITEC) at Mississippi State University.

REFERENCES

1. J. Skilling, *AIP Conference Proceedings* **1443**, 145 – 156 (2012), ISSN 0094243X.
2. J. Skilling, *Bayesian Analysis* **1**, 833–860 (2006).
3. N. S. Burkoff, C. Várnai, S. A. Wells, and D. L. Wild, *Biophysical journal* **102**, 878–886 (2012).
4. ISO, *ISO/IEC 14882:2011 Information technology — Programming languages — C++*, International Organization for Standardization, Geneva, Switzerland, 2012.
5. C. A. Hoare, *The Computer Journal* **5**, 10–16 (1962).
6. P. M. Goggans, R. W. Henderson, and N. Xiang, "Using nested sampling with Galilean Monte Carlo for model comparison problems in acoustics," in *Proceedings of Meetings on Acoustics*, 2013, vol. 19, pp. 38–45.