PROCEEDINGS OF SCIENCE

Biased Metropolis-Heatbath Algorithms for Lattice Gauge Theory

Alexei Bazavov[*]

Department of Physics, Florida State University, Tallahassee, FL 32306-4350
School of Computational Science, Florida State University, Tallahassee, FL 32306-4120
E-mail: bazavov@csit.fsu.edu

Bernd A. Berg

Department of Physics, Florida State University, Tallahassee, FL 32306-4350
School of Computational Science, Florida State University, Tallahassee, FL 32306-4120
E-mail: berg@csit.fsu.edu

It is illustrated for 4D $SU(2)$ lattice gauge theory that sampling with a biased Metropolis scheme is essentially equivalent to using the heatbath algorithm. Only, the biased Metropolis method can also be applied when an efficient heatbath algorithm does not exist. Other cases for which the use of the biased Metropolis-heatbath algorithm is beneficial are briefly summarized.

XXIIIrd International Symposium on Lattice Field Theory
25-30 July 2005
Trinity College, Dublin, Ireland

[*] Speaker.

1. Introduction

In this talk we introduce a Metropolis-type updating procedure, which uses approximate heatbath probabilities for its proposals, so that an acceptance rate close to 1 is achieved without any undue shrinking of the proposal range. This introduces a *bias* in the probability ratio of the Metropolis accept/reject step, and we call the algorithm *Biased Metropolis-Heatbath Algorithm (BMHA)*, while we use the notation HBA for HeatBath Algorithm. Details of the application of the BMHA to $SU(2)$ and $U(1)$ gauge theories are already published in [1].

2. $SU(2)$ Pure Gauge Theory Example

Here we explain the $SU(2)$ example. The action is

$$S(\{U\}) = \frac{1}{N_c}\sum_{\Box} \mathrm{Re}\,\mathrm{Tr}\,(U_{\Box}) \tag{2.1}$$

with $U_{\Box} = U_{i_1 j_1} U_{j_1 i_2} U_{i_2 j_2} U_{j_2 i_1}$, where the sum is over all plaquettes of a 4D simple hypercubic lattice, i_1, j_1, i_2 and j_2 label the sites circulating about the plaquette and U_{ji} is a $SU(2)$ matrix ($N_c = 2$) associated with the link $\langle ij\rangle$. The reversed link is associated with the inverse matrix. While working at a particular link $\langle ij\rangle$, we need only to consider the contribution to S, which comes from the staples containing this link. We denote by $U_{\sqcup,k}$, $k = 1,\dots,6$, the products which interact with the link in question. Then the probability density of this link matrix is

$$dP(U) \sim dU\,\exp\left[\frac{\beta_g}{N_c}\,\mathrm{Re}\,\mathrm{Tr}\left(U\sum_{k=1}^{6} U_{\sqcup,k}\right)\right]\,. \tag{2.2}$$

Using the property of $SU(2)$ group elements that any sum of them is proportional to another $SU(2)$ element we define a $SU(2)$ matrix $U_{\sqcup}$ by

$$U_{\sqcup} = \frac{1}{s_{\sqcup}}\sum_{k=1}^{6} U_{\sqcup,k},\;\; s_{\sqcup} = \sqrt{\det\left(\sum_{k=1}^{6} U_{\sqcup,k}\right)}\,. \tag{2.3}$$

Let $V = U\,U_{\sqcup}$, using the invariance of the group measure, $dV = dU$, one finds

$$dP\,(V) \sim dV\,\exp\left[\frac{\beta_g}{2}\,s_{\sqcup}\,\mathrm{Re}\,\mathrm{Tr}\,(V)\right]\,. \tag{2.4}$$

In the basis of Pauli matrices, $\vec{\sigma}$, V is conveniently parametrized:

$$V = a_0\,I + i\,\vec{a}\cdot\vec{\sigma},\;\; a_0^2 + \vec{a}^{\,2} = 1 \;\;\Rightarrow\;\; \mathrm{Tr}\,(V) = 2a_0\,. \tag{2.5}$$

The group measure is $dV = \sqrt{1-a_0^2}\,da_0\,d\Omega$ with $d\Omega$ being solid angle of the three dimensional $\vec{a}$. Let us denote $\alpha = \beta_g\,s_{\sqcup}$, then

$$dP\,(V) \sim d\Omega\,da_0\,\sqrt{1-a_0^2}\,\exp\,(\alpha\,a_0)\,. \tag{2.6}$$

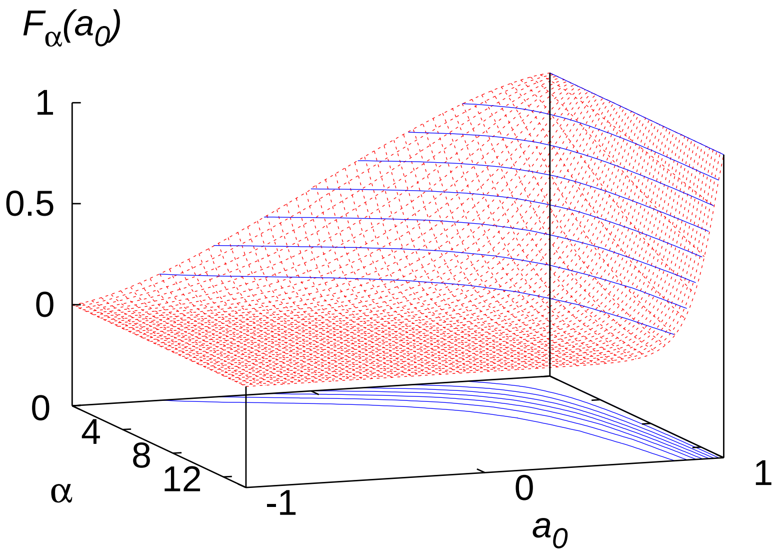

Figure 1: Cumulative distribution function $F_\alpha(a_0)$ with the level map in α–a_0 (x–y) plane.

As nothing depends on the solid angle, $\vec{a}$ is generated uniformly on a sphere of a radius $\sqrt{1-a_0^2}$ and the main step is to generate a_0 in the interval $-1 \le a_0 \le 1$ with probability density

$$P_\alpha(a_0) \sim \sqrt{1-a_0^2}\,\exp\left(\alpha\, a_0\right). \tag{2.7}$$

The cumulative distribution function (CDF) is

$$F_\alpha(a_0) = N_\alpha \int_{-1}^{a_0} da_0' \sqrt{1-a_0'^{\,2}}\,\exp\left(\alpha\, a_0'\right) \tag{2.8}$$

where N_α ensures the normalization $F_\alpha(1) = 1$ and, evidently, $P_\alpha(a_0) = dF_\alpha(a_0)/da_0$.

$F_\alpha(a_0)$ is a function of two variables: the parameter α, which incorporates the effect of interaction with the neighbors, and a_0, the variable being updated. In the following we will associate α to the x-, a_0 to the y-, and $F_\alpha(a_0)$ to the z-axis. Also, we will consider $SU(2)$ gauge theory at a coupling $\beta_g = 2.3$, close to the critical point. In this case $0 \le \alpha \le 6\beta_g = 13.8$. With these conventions $F_\alpha(a_0)$ is plotted on Fig. 1. Contour lines on the surface represent levels where $F_\alpha(a_0)$ increases from 0 to 1 by a chosen constant value (in this case $1/8$). Lines in α–a_0 (x–y) plane are projections of these contours and constitute a *level map* similar to those used to encode height on maps in geography. What we need to construct a BMHA is a discretized version of this level map.

3. Constructing the Algorithm

3.1 Heatbath Algorithm

The HBA generates a_0 for a given α by converting a uniformly distributed random number $0 \le z < 1$ into $a_0 = F_\alpha^{-1}(z)$. For $SU(2)$ it was first introduced by Creutz [2] and improved in Ref. [3] and [4]. As $F_\alpha^{-1}(z)$ is approximated one needs a *repeat until accepted (RUA)* step to generate the correct distribution. The *acceptance rate (AR)* of the updates, defined as the ratio of updated links over the number of visited links, is always 1 for the HBA [5].

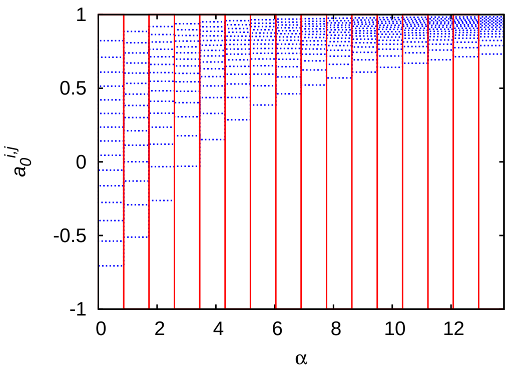

Figure 2: $m \times n$ partitioning of $\triangle a_0^{i,j}$ for $SU(2)$ at $\beta_g = 2.3$ for the table values discussed in the text.

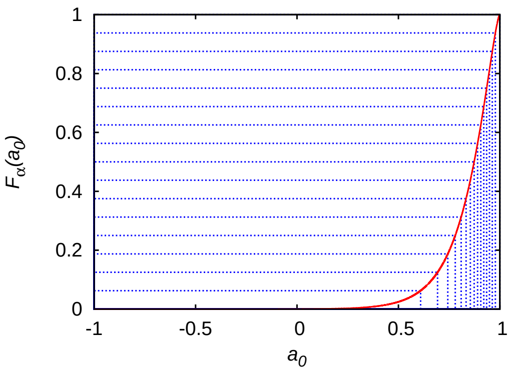

Figure 3: Discretization of the cumulative distribution function $F_{\alpha^{11}}(a_0)$ for $SU(2)$ at $\beta_g = 2.3$.

3.2 Metropolis Algorithm

In the conventional Metropolis scheme $a_{0,new}$ is generated uniformly in the range $[-1,1]$ and then accepted with probability

$$p_{Met} = \min\left\{1, \frac{P_\alpha(a_{0,new})}{P_\alpha(a_{0,old})}\right\} = \min\left\{1, \frac{\sqrt{1-a_{0,new}^2}\,\exp(\alpha\, a_{0,new})}{\sqrt{1-a_{0,old}^2}\,\exp(\alpha\, a_{0,old})}\right\}. \tag{3.1}$$

It may have a low AR in the region of interest. Possible cures are to decrease the proposal range, which makes the moves small, or *multi-hit* Metropolis, which needs a fixed number of hits (i.e. cannot be transformed into a RUA step).

3.3 Biased Metropolis-Heatbath Algorithm

Let us discretize the parameter α (x axis) into $m = 2^{n_1} = 16$ ($n_1 = 4$) bins. For simplicity we choose equidistant partitioning, although other discretizations are possible too. In each α^i bin we discretize $F_{\alpha^i}(a_0)$ (z axis) into $n = 2^{n_2} = 16$ ($n_2 = 4$) values and store $a_0^{i,j}$ which satisfy the condition: $j/n = F_{\alpha^i}(a_0^{i,j})$. In this way we achieve a discretized version of the level map at the bottom of Fig. 1, which is shown in Fig. 2.

Two two-dimensional arrays are needed: one for storing $a_0^{i,j}$ (levels themselves) and another for $\triangle a_0^{i,j} = a_0^{i,j} - a_0^{i,j-1}$ (distances between levels). Let us assume that for a link being updated α falls into the 11th bin, so $i = 11$. Finding i is achieved by an operation of the form: integer$[m\,\alpha/\alpha_{max}]$ with $\alpha_{max} = 6\beta_g = 13.8$ for $\beta_g = 2.3$. This is our first step.

Then $F_{\alpha^{11}}(a_0)$ is given by a cross section in the y–z plane shown on Fig. 3. The next step is to determine the bin label j_{old} which belongs to the (known) value $a_{0,old}$. This can be done in n_2 steps using the recursion $j \rightarrow j + 2^{i_2}\,\mathrm{sign}\,(a_0 - a_0^{i,j})$, $i_2 \rightarrow i_2 - 1$. Once j_{old} is known it gives a length of the bin: $\triangle a_0^{i,j_{old}}$.

The next step is to propose $a_{0,new}$. Two uniform random numbers (r_1 and r_2) are needed: one to pick up an integer j_{new} in the range $1,\ldots,n$ (as $j_{new} = $ integer$[n\,r_1]$), another to propose

Table 1: Efficiency of the $SU(2)$ algorithms on a 4×16^3 lattice at $\beta_g = 2.3$.

	HBA [3, 4]	Metropolis	BMHA
CPU time	194,873 [s]	181,321 [s]	199,244 [s]
Acceptance rate	1 (1.043 proposals)	0.111	0.975
$\langle \mathrm{Tr}(U_\Box)/2 \rangle$	0.603147 (17)	0.603066 (52)	0.603111 (21)
τ_{int}	49.8 (3.5)	409 (66)	48.2 (3.8)

$a_{0,new} = a_0^{i,j_{new}-1} + r_2 \triangle a_0^{i,j_{new}}$. Then the new value of a variable is accepted with probability:

$$p_{BMA} = \min\left\{1, \frac{P_\alpha(a_{0,new})}{P_\alpha(a_{0,old})} \cdot \frac{\triangle a_0^{i,j_{new}}}{\triangle a_0^{i,j_{old}}}\right\} = \min\left\{1, \frac{\sqrt{1-a_{0,new}^2}\,\exp(\alpha\, a_{0,new})}{\sqrt{1-a_{0,old}^2}\,\exp(\alpha\, a_{0,old})} \cdot \frac{\triangle a_0^{i,j_{new}}}{\triangle a_0^{i,j_{old}}}\right\}. \quad (3.2)$$

p_{BMA} in (3.2) differs from p_{Met} in (3.1) by a *bias* $\triangle a_0^{i,j_{new}} / \triangle a_0^{i,j_{old}}$.

We need to distinguish two issues here. First, the underlined updating procedure stays fully equivalent to the conventional Metropolis algorithm for *any* discretization $a_0^{i,j}$. Detailed balance for (3.2) can be proven the same way as for usual Metropolis without explicit knowledge of $\triangle a_0^{i,j}$. Choosing, for example, equidistant partitioning for a_0 ($\triangle a_0^{i,j} = \triangle a_0^{i,k}$ for any j, k) would turn the bias into 1 and simply get us back to original Metropolis algorithm. Hence, *the bias influences only the acceptance rate in* (3.2). This draws us to the second issue: when partitioning $a_0^{i,j}$ is achieved by discretizing the cumulative distribution function, then in the limit $m, n \to \infty$ the BMHA approaches the HBA and the acceptance rate approaches 1. The limit $m \to \infty$ simply means $\alpha^i \to \alpha$, the actual value. For the $n \to \infty$ limit let us turn to the Fig. 3. The discretization step on $z = F_\alpha(a_0)$ axis is $\triangle z = 1/n$ and $a_0 = F_\alpha^{-1}(z)$. Therefore, as $n \to \infty$, $\triangle z \to 0$:

$$\frac{\triangle a_0}{\triangle z} \to \frac{d}{dz} F_\alpha^{-1}(z) = \left(\frac{d}{da_0} F_\alpha(a_0)\right)^{-1} = \frac{1}{P_\alpha(a_0)}. \quad (3.3)$$

Then,

$$\frac{P_\alpha(a_{0,new})}{P_\alpha(a_{0,old})} \frac{\triangle a_0^{i,j_{new}}}{\triangle a_0^{i,j_{old}}} = \frac{P_\alpha(a_{0,new})}{P_\alpha(a_{0,old})} \frac{\triangle a_0^{i,j_{new}}/\triangle z}{\triangle a_0^{i,j_{old}}/\triangle z} \to \frac{P_\alpha(a_{0,new})}{P_\alpha(a_{0,old})} \frac{1/P_\alpha(a_{0,new})}{1/P_\alpha(a_{0,old})} = 1. \quad (3.4)$$

The table building process and the updating procedure are discussed in a more formal way in [1].

4. Performance

In our simulations we used a finer discretization, $m = 32$ and $n = 128$. Table 1 illustrates the performance of the $SU(2)$ BMHA for a long run on a 4×16^3 lattice at $\beta_g = 2.3$. At this coupling the system exhibits critical slowing down, because of its neighborhood to the deconfining phase transition (see for instance [6] and references therein). We used 16,384 sweeps for reaching equilibrium and, subsequently, $32 \times 20,480$ sweeps for measurements. Simulations were performed on 2 GHz Athlon PCs with the -O2 option of the (freely available) g77 Fortran compiler.

Our comparison is with the Fabricius-Haan-Kennedy-Pendleton HBA [3, 4], which at this coupling is more efficient than Creutz's HBA [2], and with the conventional Metropolis. A direct measure for the performance of an algorithm is the integrated autocorrelation time τ_{int}. It is given in the Table 1 for the Wilson plaquette together with the expectation value of this operator. Error bars are given in parenthesis and apply to the last digits. They are calculated with respect to 32 bins (jackknife bins in case of τ_{int}), relying on the data analysis software of [7].

5. Summary and Conclusions

Using BMHA may be beneficial in several lattice gauge theory situations:

- When a HBA implementation exists, but the inversion of the cumulative distribution function in question is relatively inefficient. An example is $U(1)$ lattice gauge theory explored in [1].
- When a HBA implementation does not exist. An example is $SU(2)$ lattice gauge theory in the mixed fundamental-adjoint representation (work in progress [9]).
- In multi-variable case to which the BMHA generalizes for the price of increasing dimensions of the tables.
- In (checkerboard) parallelization the updating speed of the BMHA is uniform over all nodes. This is not the case for a HBA if a RUA step is involved (as for $SU(2)$).

When the canonical CDF similar to (2.8) is *a priori* unknown a HBA becomes impossible, but there are still good chances for a Biased Metropolis Algorithm (BMA). In this case an *estimator* of the CDF may be constructed *empirically* as it was explored for biophysical applications in [8], where a BMA of the type discussed here was first proposed within a *Rugged Monte Carlo* scheme.

Acknowledgments

The authors would like to thank Wolfhard Janke and his group at Leipzig University for their kind hospitality during the summer 2005. This work was in part supported by the US Department of Energy under contract DE-FG02-97ER41022.

References

[1] A. Bazavov and B.A. Berg, Phys. Rev. **D** 71, 114506 (2005)

[2] M. Creutz, Phys. Rev. D **21**, 2308 (1980).

[3] K. Fabricius and O. Haan, Phys. Lett. B **143**, 459 (1984).

[4] A.D. Kennedy and B.J. Pendleton, Phys. Lett. B **156**, 393 (1985).

[5] In some of the literature the quantity 1/(average number of RUA heat bath iterations per update) is also called acceptance rate. It should not be confused with the acceptance rate defined here.

[6] J. Fingberg, U. Heller, and F. Karsch, Nucl. Phys. B **392**, 493 (1993).

[7] B.A. Berg, *Markov Chain Monte Carlo Simulations and Their Statistical Analysis*, World Scientific, 2004.

[8] B.A. Berg, Phys. Rev. Lett **90**, 180601 (2003); B.A. Berg and H.-X. Zhou, Phys. Rev. E **72**, 016712 (2005).

[9] A. Bazavov, B.A. Berg and U. Heller, in preparation.