Wang-Landau Algorithm for Continuous Models and Joint Density of States

Chenggang Zhou,[1,2] T. C. Schulthess,[1] Stefan Torbrügge,[3] and D. P. Landau[2]

[1]*Center for Nanophase Materials Sciences and Computer Science and Mathematics Division, Oak Ridge National Laboratory, Post Office Box 2008, MS6493, Oak Ridge Tennessee, 37831-6493 USA*

[2]*Center for Simulational Physics, University of Georgia, Athens Georgia, 30602 USA*

[3]*Universität Osnabrück, Fachbereich Physik, D-49076 Osnabrück, Germany*

(Received 11 June 2005; published 28 March 2006)

We present a modified Wang-Landau algorithm for models with continuous degrees of freedom. We demonstrate this algorithm with the calculation of the joint density of states of ferromagnet Heisenberg models and a model polymer chain. The joint density of states contains more information than the density of states of a single variable-energy, but is also much more time consuming to calculate. We present strategies to significantly speed up this calculation for large systems over a large range of energy and order parameter.

DOI: 10.1103/PhysRevLett.96.120201 PACS numbers: 02.70.Tt, 02.50.Ey, 02.50.Fz, 02.70.Rr

The Wang-Landau (WL) algorithm [1] has been applied to a broad spectrum of interesting problems in statistical physics and biophysics [1–9]. These successful applications can be attributed to two features of this algorithm. First, the WL algorithm is not trapped by local energy minima. Secondly, by calculating the density of states, we can estimate thermodynamic observables including the free energy over a wide range of temperature with one single simulation. The efficiency and convergence of the WL algorithm has been quantitatively studied [10,11], and variations [6–8] and improvements have been proposed [4,5,12,13], among which the most interesting are those intended for systems with continuous degrees of freedom, e.g., protein [4] and liquids [5–7]. For such systems it is useful, but expensive, to calculate the joint density of states (JDOS) of two or more variables, e.g., $g(V, E)$ where V is the volume and E is the energy. JDOS is useful because quantities such as the free energy can be calculated as a function of temperature and pressure. Clearly, JDOS of continuous models provides information for many interesting problems, however the extra information does not come for free. Suppose we calculate the JDOS $g(M, E)$ of a 32×32 Ising model on a square lattice, a typical benchmark problem [1]. If every value of the magnetization M is counted, $g(M, E)$ costs about 10^3 times the CPU time of $g(E)$, because $g(M, E)$ contains about 10^3 times as many entries as $g(E)$. Therefore, such calculations have been restricted to small systems [5,14], or a carefully chosen small region in the parameter space [6,7]. Less costly alternative methods have been introduced, e.g., EXEDOS [4], which restricts the simulation at a fixed temperature instead of at a fixed external force field.

In this Letter, we discuss two strategies to efficiently calculate JDOS for generic models with a continuous degree of freedom, and give two examples. Our methods are equally applicable to systems with a phase transition, e.g., a classical Heisenberg ferromagnet, and biophysics systems, e.g., protein models. For the ferromagnet, we are interested in $g(M, E)$, where M is the magnetization; while for protein models, we are interested in $g(\xi, E)$, where ξ is a reaction coordinate [4].

We first show the result of our calculation for a three-dimensional Heisenberg ferromagnet in Fig. 1. The Hamiltonian of this model is $H = -\sum_{\langle i,j \rangle} \mathbf{S}_i \cdot \mathbf{S}_j$, where the summation goes over nearest neighbors on a cubic lattice of size L with periodic boundary conditions. This model has a ferromagnetic phase transition and displays global spin rotational symmetry. Here we define $E = H/L^3$ and $M = L^{-3}\sum_i S_i^z$. Figure 1(a) shows $\ln g(M, E)$ for negative E. A region with $\partial \ln g/\partial M = 0$ below $E = -1.2$ indicates a transition to the ferromagnetic phase with a global rotational symmetry. Near the ground state $\ln g(M, E)$ is logarithmically divergent, since the JDOS vanishes when E or M is maximized or minimized. In the following, we use the Heisenberg ferromagnet as a prototype model to identify several features and intrinsic difficulties of continuous models.

In general, the models we study have many microscopic degrees of freedom $s = (s_1, \ldots, s_N)$, which labels the microscopic states in the phase space Ω. The Hamiltonian $H(s)$ is a real-valued function of s; so is the order parameter, e.g., magnetization $M(s)$ for the Heisenberg ferromagnet. The JDOS is defined as

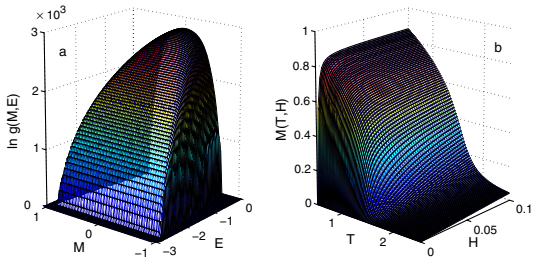

FIG. 1 (color online). (a) $\ln g(M, E)$ of a three-dimensional Heisenberg ferromagnet of size $L = 10$ with cutoff energy at -2.8, determined up to an additive constant. (b) The magnetization at different temperature and external field evaluated from $g(M, E)$.

$$g(M,E) = |\Omega|^{-1} \int \Pi_i ds_i \delta(E - H(s)) \delta(M - M(s)), \quad (1)$$

where $|\Omega|$ is the volume of the phase space, and the integral is replaced by a summation for discrete models. We refer to a pair (M, E) as a macroscopic state, and define the macroscopic phase space $\Lambda = \{(M, E) | \exists s, H(s) = E, M(s) = M\}$. Obviously $g(M, E) \geq 0$ is a probability measure of Λ induced by an *a priori* probability measure of Ω. In the following, we denote (M, E) collectively with x. The WL algorithm learns $g(x)$ in with a simple strategy. In a word, it substitutes $\exp(-E/T)$ in a Metropolis algorithm with $\exp[-w(x)]$, where $w(x)$ is an approximation to $\ln g(x)$, and updates $w(x)$ with $w(x) + \ln f$ when the random walker arrives at state x. Here f is called the modification factor. Previous studies [1,11] showed that for a discrete set of x, $w(x)$ quickly converges to $\ln g(x)$ within some statistical error proportional to $\sqrt{\gamma}$ for $\gamma \to 0$. For a continuous system, one cannot update $g(x)$ point by point but requires an algorithm that exploits the continuity of this function.

The simple binning scheme effectively approximates $g(x)$ with a piecewise constant function. By using sufficiently many bins, this method does work for $g(E)$ [15]; however, for JDOS this scheme results in an excessively large number of bins to sample, which will hamper the convergence. This can be avoided by using bilinear interpolation among neighboring bins [5]. Here we have adopted a kernel function update scheme similar to that of Ref. [16] for metadynamics, which seems more consistent with the continuous nature of $g(x)$. We select a localized positive continuous kernel function $k(x) \geq 0$, and scale it by two constants γ and δ: $k(x) \to \gamma k(x/\delta)$. If the random walker arrives in x_0, then the continuous histogram $w(x)$ is updated by

$$w(x) \to w(x) + \gamma k((x - x_0)/\delta), \quad (2)$$

and we express the acceptance rate as

$$A_{i \to f} = \min\{1, \exp[\ln\alpha [w(x_i) - w(x_f)]]\}, \quad (3)$$

where we have inserted a constant $\ln\alpha$ so that $w(x)$ converges to $\log_\alpha g(x)$. We have implemented a Gaussian kernel function $k(x) = \exp(-|x|^2)$, as well as an Epanechnikov kernel $k(x) = [1 - |x|^2]_+$, and found no visible difference. By slightly modifying the approach in Ref. [11], we can prove the convergence of this scheme.

Thus the single modification factor in the original WL algorithm is replaced by a triplet (α, γ, δ). In our simulations, we have used numbers between 0.0001 and 0.01 for γ, and δ corresponding to about $1/200$ of the width of the energy or magnetization windows. Unlike the original WL algorithm, we do not reduce γ to extremely small values in the simulation, nor do we change δ in the simulation. We have the freedom to change α provided that $w(x)$ is properly rescaled.

If we start from an unbiased initial $w_0(x) = 0$, we know $w_T(x)$ (where T labels the number of Monte Carlo steps) grows from the region of large $g(x)$, and extends to unexplored region of small $g(x)$ [11]. $w_T(x)$ can be written as: $w_T(x) = [C(T) + \log_\alpha g(x) + r_T(x)]_+$, where $C(T)$ is an increasing function of T only, $r_T(x)$ is a bounded error term controlled by the triplet (α, γ, δ). $w_T(x)$ increases monotonically in the simulation and remains zero in the unexplored region. One cannot expect it to be flat as in the original WL algorithm for discrete models. The simulation should be stopped by other criteria, e.g., the visited area reaches a low energy cutoff, after which we continue the simulation with reduced γ to improve the accuracy. If the result is accurate, $w_T(x)$ increases uniformly in the visited area. T is estimated by counting the number of kernel functions used to build up $w_T(x)$:

$$T = \left[\int \gamma k(x/\delta) dx \right]^{-1} \int_\Lambda w_T(x) dx. \quad (4)$$

However, we find that the initial accumulation in which $\Lambda_T = \text{supp}\{w_T(x)\}$ expands to Λ takes a very long time for JDOS. The expansion of Λ_T slows down as the area of Λ_T increases. The reasons are twofold. First, the simulation samples the macroscopic states within Λ_T uniformly, giving rise to a uniform growth there. This uniform growth takes a considerable CPU time, while only about a fraction $|\Lambda_T|^{-1/2}$ of Monte Carlo steps on the boundary of Λ_T happen to extend the simulation to the unexplored area. Secondly, close to the singular boundaries of Λ, $\nabla \log_\alpha g(x)$ becomes very large, requiring a very small δ (high resolution in the kernel function) to capture the large derivative.

To avoid repeated sampling of the visited region Λ_T, and to push the simulation to the unexplored region, we find it is most efficient to introduce a global update of $w_T(x)$: when $w_T(x)$ is a good estimate of $\log_\alpha g(x)$ inside Λ_T, we update $w_T(x)$ with the following formula:

$$w_T(x) \to w_T(x) + \kappa \exp\left[\frac{-\lambda}{w_T(x) - \omega}\right] \Theta(w_T(x) - \omega), \quad (5)$$

where Θ is the Heaviside step function. Basically, $w_T(x)$ is shifted up by an amount of κ where $w_T(x) > \omega$, and the exponential function removes the resultant discontinuity. Here ω is positive because the local update scheme requires a minimum number of visits to give a correct estimation of the density of states.

After this global update, the random walker is forced to sample the boundary of Λ_T; only when the accumulation on the boundary exceeds κ is the random walker likely to come back to the interior of Λ_T. Possible artifacts that result from the global update will then be quickly covered up by local updates. Figure 2 illustrates one cycle of calculation with the global update. The simulation is decomposed into a number of cycles. In each cycle, the simulation works on a subset Λ_n of Λ, $\Lambda_n = \{x | n\kappa > g_{\max} - \log_\alpha g(x) > (n-1)\kappa\}$, where $g_{\max}$ is the maximum of $g(x)$. In the nth cycle, the simulation roughly requires

$$T_n = \left[\int \gamma k(x/\delta) dx \right]^{-1} \int_{\Lambda_n} [\log_\alpha g(x) - g_{\max} + n\kappa] dx$$

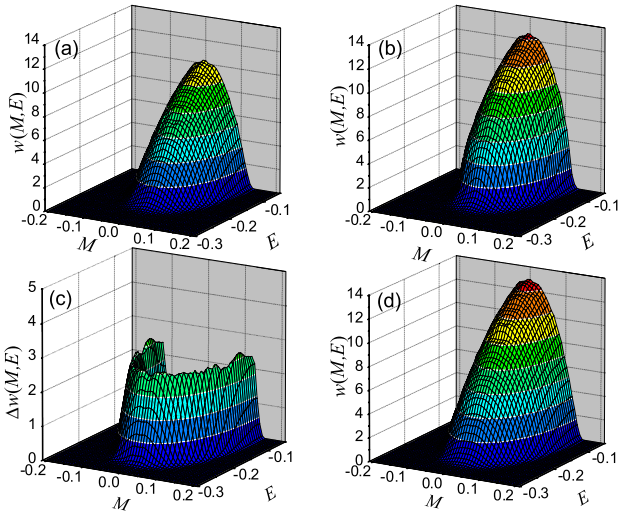

FIG. 2 (color online). A cycle of the simulation with global update. (a) original $w_T(x)$, (b) $w_T(x)$ after the global update, (c) the increment accumulated beginning with the $w_T(x)$ in (b), (d) sum of (b) and (c) gives a new $w_T(x)$ in the end of this cycle.

Monte Carlo steps. By comparison with Eq. (4), we see that instead of filling up the bulk of $w_T(x)$, the simulation only fills up a thin surface layer of $w_T(x)$ of thickness roughly given by κ. Consequently, the algorithm with the global update saves a huge number of Monte Carlo steps. Compared with distributing the random walkers to a number of "windows" [1,5,12], the global update has the advantage of automatically selecting the windows on the frontier of the simulation and avoiding the boundary errors [12]. Figure 3 shows that the global update saves 90% of the CPU time in a typical simulation, where we plot t as a function of maximum histogram $W \approx w(0, 0)$. Without the global update, since the increment in W is due to the uniform accumulation of samples inside the visited area Λ_T, therefore $dt/dW \propto |\Lambda_T|$. From Fig. 1, we notice that Λ_T mainly grows towards lower energy. Hence we expect approximately $|\Lambda_T| \approx aW + b$, which results in a leading quadratic dependence $t \propto W^2$ in Fig. 3. With the global update, both the prefactor and the exponent of this relation are reduced. The short global samplings at the end of each cycle result in $T \propto W^\lambda$, with $\lambda \approx 1.55$ in Fig. 3.

The algorithm can be summarized as the following: A calculation from scratch is divided into an initial accumulation stage and a refining stage. In the initial accumulation stage, (1) we start the calculation with $w(x) = 0$, using local updates Eq. (2). (2) As soon as $w(x) > \omega$ for some x, we apply the global update Eq. (5), and continue the accumulation with local updates. $w_T(x)$ initially increases on the boundary of Λ_T. (3) After it resumes a uniform growth over Λ_T, we start the next cycle with another global update. The refining stage starts when $w_T(x)$ expands to the entire area of interest. Then we continue the simulation with only local updates until a uniform growth of $w_T(x)$ in this area is observed. The JDOS can be refined with reduced γ or taking average of multiple uncorrelated results of $w_T(x)$.

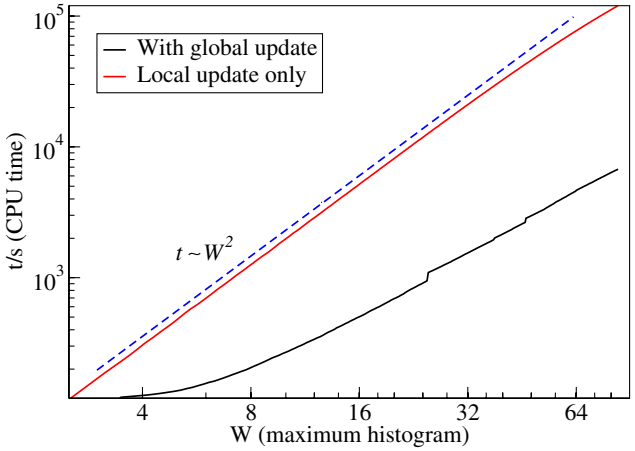

FIG. 3 (color online). Comparison of CPU time used by the calculations of $g(M, E)$ with and without the global update for an $L = 5$ Heisenberg model of ferromagnet; $\omega = 0.5$ is used. The dashed line is a guide for the eye.

We calculate the thermodynamic quantities from $g(M, E)$ with numerical integral. Figure 1(b) shows the magnetization of a Heisenberg ferromagnet as a function of external field and temperature. The specific heat in Fig. 4 shows a typical peak at T_c of Heisenberg models. We also compare our results with that calculated with the original WL algorithm in Fig. 4, which evaluates $g(E)$ on a grid of 3000 bins. They only differ slightly at low temperatures where both results show small errors. This error comes from the binning or interpolation scheme used to represent the continuous $g(E)$ or $g(M, E)$ (not from the numerical integration).

Actually, given that the standard deviation of the canonical distribution of the energy is $\sigma_E = \sqrt{T^2 C_v / L^3} \approx 0.036$ for $L = 10$, $T = 1$, and $\sigma_E \approx 0.1$ for $L = 5$, an accurate numerical integration requires enough data points within

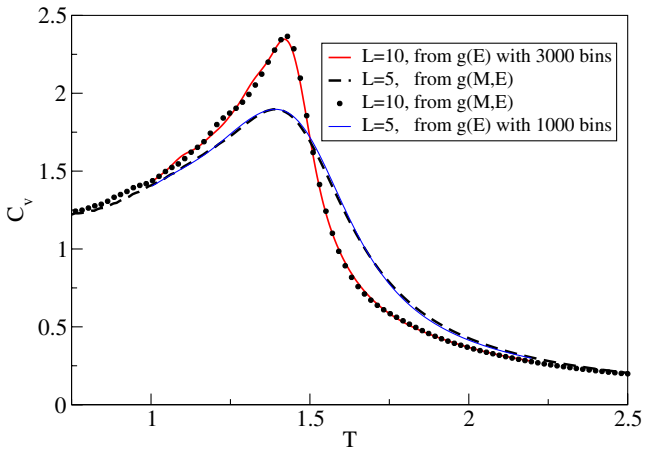

FIG. 4 (color online). Specific heat of the Heisenberg ferromagnet of size $L = 10$ and 5, with comparison to the results from the original WL algorithm performed with a large number of bins.

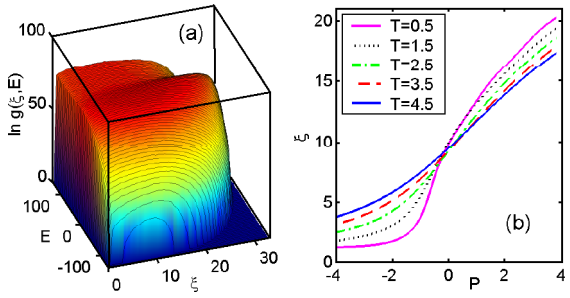

FIG. 5 (color online). (a) JDOS $g(\xi, E)$ of a chain with 30 atoms. (b) The end-to-end length ξ as a function of pulling force P at different temperatures.

σ_E. This criterion applies to both our kernel function updates and the bilinear interpolation scheme [5]. A larger σ_E explains why the error is smaller for $L = 5$ in Fig. 4 than that for $L = 10$. In case of $L = 10$, the internal array we used to store $g(M, E)$ has an energy resolution of 0.0012, which is comparable to the bin size (0.001) of the original WL algorithm we used for $g(E)$. Consequently, they show errors of comparable sizes. The conclusion is that the resolution in each macroscopic quantity must increase as $\sqrt{N}$ to maintain the accuracy in the numerical integral, where N is the number of degrees of freedom.

Our second example is a simple protein model whose ground state is a perfect helix. The model is a polymer chain of fixed bond length and bond angle; its dihedral angles are the only degrees of freedom. We use our algorithm to calculate $g(\xi, E)$, where ξ is the end-to-end distance (referred to as a reaction coordinate in Ref. [4]) and we use the same parameters as in Ref. [17] where it was originally studied. Figure 5 shows the JDOS and ξ as a function of the pulling force P at different temperatures for a chain of 30 atoms. A high energy cutoff is needed because the Lennard-Jones (LJ) repulsion has no upper bound. Because of the geometric constraints, the smallest polygon that the chain can form is a hexagon, in order to have a large LJ term. In this case, six bonds are replaced by the radius of the repulsion potential. Stretching the rest of the chain as much as possible, one can estimate that the threshold length is about $\xi = 27$. This threshold gives rise to a clear shoulder structure in Fig. 5(a). $\xi(P, T)$ is calculated in the same way as $M(h, T)$ for the Heisenberg ferromagnet. At low T and $P = 0$, ξ is the length of the helix, which behaves as a Hook spring for small pulling forces. The free energy for each ξ (calculated with the EXEDOS algorithm in Ref. [4]) can be obtained by integrating $g(\xi, E)e^{-E/T}$ over E directly. This example only took about half an hour on a single CPU.

In summary, we use kernel function local updates and a global update to extend the WL algorithm to efficiently treat continuous systems and their JDOS. Our new strategies have potential applications to many complex systems with thousands of degrees of freedom. In particular, the kernel function update benefits from the continuity of the model; and the global update effectively drives the random walker to unexplored areas, so that extreme values of macroscopic variables can be searched. Compared to the original WL algorithm, the global update saves about 90% CPU time in our calculations. Recently, we have also studied magnetic nanoparticles made of $NiFe_2O_4$ [18] with our method presented in this Letter.

We thank D. Sanders for his helpful comments. This research is supported by the Department of Energy through the Laboratory Technology Research Program and the Computational Materials Science Network, as well as the Division for Scientific User Facilities, and by NSF DMR-0341874.

[1] F. Wang and D. P. Landau, Phys. Rev. Lett. **86**, 2050 (2001); Phys. Rev. E **64**, 056101 (2001); Comput. Phys. Commun. **147**, 674 (2002).
[2] C. Yamaguchi, and Y. Okabe, J. Phys. A **34**, 8781 (2001).
[3] Y. Okabe, Y. Tomita, and C. Yamaguchi, Comput. Phys. Commun. **146**, 63 (2002).
[4] N. Rathore and J. J. de Pablo, J. Chem. Phys. **116**, 7225 (2002); N. Rathore, T. A. Knotts IV, and J. J. de Pablo, *ibid.* **118**, 4285 (2003); N. Rathore, Q. Yan, and J. J. de Pablo, *ibid.* **120**, 5781 (2004).
[5] M. S. Shell, P. G. Debenedetti, and A. Z. Panagiotopoulos, Phys. Rev. E **66**, 056703 (2002).
[6] E. A. Mastny and J. J. de Pablo, J. Chem. Phys. **122**, 124109 (2005).
[7] Q. Yan, R. Faller, and J. J. de Pablo, J. Chem. Phys. **116**, 8745 (2002); Q. Yan, and J. J. de Pablo, Phys. Rev. Lett. **90**, 035701 (2003).
[8] S. Trebst, S. Trebst, D. A. Huse, and M. Troyer, Phys. Rev. E **70**, 046701 (2004).
[9] M. Troyer, S. Wessel, and F. Alet, Phys. Rev. Lett. **90**, 120201 (2003).
[10] P. Dayal, S. Trebst, S. Wessel, D. Würtz, M. Troyer, S. Sabhapandit, and S. N. Coppersmith, Phys. Rev. Lett. **92**, 097201 (2004).
[11] C. Zhou and R. N. Bhatt, Phys. Rev. E **72**, 025701(R) (2005).
[12] B. J. Schulz, K. Binder, M. Müller, and D. P. Landau, Phys. Rev. E **67**, 067102 (2003).
[13] A. Malakis, A. Peratzakis, and N. G. Fytas, Phys. Rev. E **70**, 066128 (2004).
[14] D. P. Landau, Shan-Ho Tsai, and M. Exler, Am. J. Phys. **72**, 1294 (2004).
[15] G. Brown, and T. C. Schulthess, J. Appl. Phys. **97**, 10E303 (2005).
[16] C. Micheletti, A. Laio, and M. Parrinello, Phys. Rev. Lett. **92**, 170601 (2004); Y. D. Wu, J. D. Schmitt, and R. Car, J. Chem. Phys. **121**, 1193 (2004).
[17] D. C. Rapaport, Phys. Rev. E **66**, 011906 (2002).
[18] C. Zhou, T. C. Schulthess, and D. P. Landau, cond-mat/0511410 [J. Appl. Phys. (to be published)].