Directed geometrical worm algorithm applied to the quantum rotor model

Fabien Alet[1,2,*] and Erik S. Sørensen[3]

[1]*Computational Laboratory, ETH Zürich, CH-8092 Zürich, Switzerland*
[2]*Theoretische Physik, ETH Zürich, CH-8093 Zürich, Switzerland*
[3]*Department of Physics and Astronomy, McMaster University, Hamilton, Ontario, Canada L8S 4M1*

We discuss the implementation of a directed geometrical worm algorithm for the study of quantum link-current models. In this algorithm the Monte Carlo updates are made through the biased reptation of a worm through the lattice. A directed algorithm is an algorithm where, during the construction of the worm, the probability for erasing the immediately preceding part of the worm, when adding a new part, is minimal. We introduce a simple numerical procedure for minimizing this probability. The procedure only depends on appropriately defined local probabilities and should be generally applicable. Furthermore, we show how correlation functions $C(\mathbf{r}, \tau)$ can be straightforwardly obtained from the probability of a worm to reach a site $(\mathbf{r}, \tau)$ away from its starting point independent of whether or not a directed version of the algorithm is used. Detailed analytical proofs of the validity of the Monte Carlo algorithms are presented for both the directed and undirected geometrical worm algorithms. Results for autocorrelation times and Green's functions are presented for the quantum rotor model.

I. INTRODUCTION

Improving and developing new numerical algorithms lies at the heart of computational physics. Among others, the Monte Carlo (MC) methods are often seen as the best choice for the study of phase transitions taking place in classical or quantum models. For the study of spin models, for example, cluster algorithms, either in the classical [1,2] or quantum [3–5] case, perform nonlocal moves in phase space, allowing for the treatment of systems much larger than with traditional local update methods (single spin-flip algorithms). These types of algorithms have almost completely solved the problem of critical slowing down arising near phase transitions.

The class of systems for which cluster methods are known to exist is limited, and it is, therefore, of great interest to search for new algorithms possessing the same efficient features for other models. In this context, we have proposed recently a nonlocal "worm" algorithm for the study of quantum link-current models [6]. These models arise from a phase approximation of bosonic Hubbard models, but are also relevant in the context of quantum electrodynamics [7]. Previous MC simulations of the quantum link-current (quantum rotor) model used a local algorithm suffering from critical slowing down. In the new algorithm [6] updates are made by reptating a "worm" through the lattice [5,8]. Since the movement of the worm only depends on a few probabilities determined locally with respect to the current position of the "head" of the worm, we call this type of algorithm a geometrical worm algorithm as opposed to other recently developed worm algorithms based on high-temperature series expansions [8]. The geometrical worm algorithm gives rise to very small autocorrelation times and by directing the algorithm these autocorrelation times can be even further reduced.

In this paper, we briefly recall the principles of the geometrical worm algorithm [6]. During the construction of a worm a new part is added to the worm by moving the worm through one of the σ nearest-neighbor links. Usually the associated σ probabilities p_σ are chosen in an unbiased geometrical way and there is, therefore, a significant probability that the new part of the worm will backtrack in its own path, thereby erasing the immediately preceding part. In many cases this backtracking (or bounce) probability is the dominant probability among the σ probabilities and using these unbiased probabilities is, therefore, clearly rather wasteful. Here we describe an improvement of this geometric worm algorithm, which we call the directed worm algorithm, as a reference to recently developed directed loop methods for the quantum Monte Carlo (QMC) simulations of spin systems [9]. This directed geometrical worm algorithm is identical to its undirected counterpart except for the fact that the probabilities p_σ are now chosen in a biased way, using knowledge of the immediately preceding step in the construction of the worm. These biased probabilities can all be tabulated at the start of the simulation and the additional computational effort stems solely from the significantly wider distribution of the directed worms. The directed algorithm gives rise to even better results, as will be shown in the following part of this paper, where we present results on autocorrelation times for both directed and "undirected" worm algorithms. The procedure for choosing the "biased" probabilities leading to the directed algorithm is quite general and should be applicable to other algorithms that depend on local probabilities. Furthermore, we show how Green's functions $C(\mathbf{r}, \tau)$ of the original quantum model can be measured efficiently during the construction of the worm by calculating the probability that the worm reaches a given site $(\mathbf{r}, \tau)$ away from its starting point, independent of whether a directed or undirected algorithm is used. For both the derivation of the directed algorithm and the measurements of correlation functions,

*Electronic address: alet@phys.ethz.ch

analytical proofs of the validity of the algorithms are presented.

The outline of the paper is as follows. In the following section, we present the quantum rotor model and introduce some useful notation. Then, a brief description of the "undirected" worm algorithm is given in Sec. III A, before proceeding to the main contents of this paper, a description of the directed geometrical worm algorithm (Sec. III B). A simple procedure for numerically determining the biased probabilities for the worm moves is presented. In addition, we derive a proof of detailed balance for the directed worm algorithm. In order to compare our algorithms to related ones, we present in Sec. III C another recent approach due to Prokof'ev and Svistunov [8], originally based on high-temperature series expansion for classical statistical models, which we therefore will refer to as "classical worms" throughout this paper. In Sec. IV, we estimate the efficiency of the three algorithms by calculating autocorrelation times. We then discuss the measurements of correlation functions within the worm algorithm in Sec. V and show some results at a specific point of the phase diagram. We conclude with a discussion of the features of the directed algorithm.

II. THE MODEL

Many magnetic systems, Josephson Junction arrays and several other systems can be described by a quantum rotor model [10]:

$$H_{qr} = \frac{U}{2} \sum_{\mathbf{r}} \left(\frac{1}{i} \frac{\partial}{\partial \theta_{\mathbf{r}}} \right)^2 + i \sum_{\mathbf{r}} \mu_{\mathbf{r}} \frac{\partial}{\partial \theta_{\mathbf{r}}} - t \sum_{\langle \mathbf{r}, \mathbf{r}' \rangle} \cos(\theta_{\mathbf{r}} - \theta_{\mathbf{r}'}). \tag{1}$$

Here, $\theta_{\mathbf{r}}$ is the phase of the quantum rotor, t is the renormalized coupling strength, and $\mu_{\mathbf{r}}$ is an effective chemical potential. If $\mu \equiv 0$, it can be shown that this model displays the same critical behavior as the $(D+1)$-dimensional XY model. However, when $\mu_{\mathbf{r}} \neq 0$, this model is not amenable to direct numerical treatment in this representation due to the resulting imaginary term. It is, therefore, very noteworthy that an equivalent *completely real* representation in terms of link-currents exists even for nonzero $\mu_{\mathbf{r}}$. This link-current (Villain) representation is a classical $(2+1)$D equivalent Hamiltonian, which is usually written in the following manner [11]:

$$H = \frac{1}{K} \sum_{(\mathbf{r}, \tau)} \left[\frac{1}{2} \mathbf{J}_{(\mathbf{r}, \tau)}^2 - \mu_{\mathbf{r}} J_{(\mathbf{r}, \tau)}^{\tau} \right]. \tag{2}$$

The sum is taken over all divergenceless current configurations $\nabla \cdot \mathbf{J} = 0$. The degrees of freedom are "currents" $\mathbf{J} = (J^x, J^y, J^{\tau})$ living on the links of the lattice. These link variables $J^x, J^y, J^{\tau} = 0, \pm 1, \pm 2, \pm 3 \ldots$ are integers. K is the effective temperature, varying like t/U in the quantum rotor model. We refer to Ref. [11] for a precise derivation of this model and for a description of its physical implications.

Another incentive for studying the critical behavior of the quantum rotor model comes from the close relation between
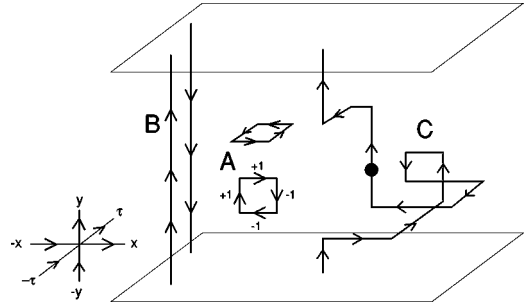

FIG. 1. The Monte Carlo moves in the model. To ensure the divergenceless condition, only closed moves can be performed. In the left part of the figures, previous Monte Carlo updates with the local algorithm are depicted. In the right part is given an example of a move with the worm algorithm starting from an initial random site (black dot).

this model and bosonic systems. Bosonic systems with strong correlations are often described in terms of the (disordered) boson Hubbard model: $H_{bH} = \sum_{\mathbf{r}} [(U/2) \hat{n}_{\mathbf{r}}^2 - \mu_{\mathbf{r}} \hat{n}_{\mathbf{r}}] - t_0 \sum_{\langle \mathbf{r}, \mathbf{r}' \rangle} (\hat{\Phi}_{\mathbf{r}}^{\dagger} \hat{\Phi}_{\mathbf{r}'} + c.c)$. The correlations are here described by U, the on-site repulsion. The hopping strength is given by t_0 and $\mu_{\mathbf{r}}$ the chemical potential varying uniformly in space between $\mu \pm \Delta$. $\hat{n}_{\mathbf{r}} = \hat{\Phi}_{\mathbf{r}}^{\dagger} \hat{\Phi}_{\mathbf{r}}$ is the number operator. If we set $\hat{\Phi}_{\mathbf{r}} \equiv |\hat{\Phi}_{\mathbf{r}}| e^{i \hat{\theta}_{\mathbf{r}}}$ and integrate out amplitude fluctuations, it can be shown that H_{bH} is equivalent to the quantum rotor model [11]. For systems where amplitude fluctuations can be neglected at the critical point, such as granular superconductors and Josephson junctions arrays, the quantum rotor model should, therefore, correctly describe the underlying quantum critical phenomena.

In the following we only discuss the quantum rotor model in $d=2$ dimensions corresponding to the $(d+1)$-dimensional link-current model. In general, a nonzero μ will introduce separate dynamics for the time and space directions from which a dynamical critical exponent z can be defined. If the divergence of the spatial correlation length close to criticality is characterized by the exponent ν, z is defined by requiring that the correlation length in the time direction diverges with the exponent $z\nu$. For what will be discussed here $\mu = 0$ and $z = 1$.

III. ALGORITHMS

A. The geometrical (undirected) worm algorithm

The quantum rotor model has been extensively studied in the link-current representation using the conventional Monte Carlo technique using local updates [11–15].

The conventional Monte Carlo updates on model (2) consists of updating simultaneously four link variables, as shown in the left part of Fig. 1 (A). To ensure ergodicity, one also has to use global moves, updating a whole line of link variables (B in Fig. 1). The acceptance ratio for these global moves becomes exponentially small with the system size for large systems. Many interesting quantities such as the stiff-

ness, necessary for the determination of the critical point, and current-current correlations, necessary for the calculation of transport properties such as the resistivity and the compressibility, are *only* nonzero when these global moves are successful. An effective Monte Carlo sampling of these global moves is, therefore, imperative and it is easy to understand that the performance of the local algorithm is rather poor, especially near a phase transition: critical slowing down in the Monte Carlo simulations prohibits the study of large system sizes.

In order to be able to correctly describe the directed geometrical worm algorithm, we have to review the geometrical worm-cluster algorithm introduced in Ref. [6] in some detail. This algorithm allows for nonlocal moves as those depicted on the right part of Fig. 1 (*C*). The performances of this algorithm have been reported in the previous work [6]. This algorithm is closely related to other cluster algorithms [1–3] and especially to "worm" algorithms [5,4,8], from which we have borrowed the name. We stress that this algorithm is different from the "classical" worm algorithm presented in Ref. [8] in the sense that it is geometrical: link variables are not "flipped" with a thermodynamic probability, instead, a new part of the worm is added by selecting a direction according to σ locally determined probabilities p_σ. Since these probabilities only depend on the local environment, we call them "*geometrical*" probabilities. Secondly, even though the local probabilities p_σ do depend on the effective temperature K, we always have $\Sigma_\sigma p_\sigma = 1$ (per definition) and the only effect of the temperature is, therefore, to preferentially move the worm in one direction as opposed to another one. In some sense this is very similar to the N-fold way [16] as well as to the Rosenbluth scheme [16] of performing the Monte Carlo simulations.

We now describe the contents of the geometrical algorithm: we update the configurations by moving a "worm" through the lattice of links—the links through which the worm pass are updated *during* its construction. The configurations generated during the construction ("reptation") of the worm are not valid (the vanishing of the divergence of **J** is not fulfilled) but at the end of its path, when the worm forms a closed loop, this condition is verified and the final configuration is valid.

We first define a convention for the orientation of the lattice. Around each site with coordinates $[\mathbf{r}=(x,y),\tau]$, there are six links on which the integer currents $J_{(\mathbf{r},\tau)}^\sigma$ are defined with $\sigma=x,y,\tau,-x,-y,-\tau$. The change in $J_{(\mathbf{r},\tau)}^\sigma$ that the worm will perform during its course depends on whether σ is an incoming or outgoing link: here our convention is to consider positive x,y,τ as outgoing directions and $-x,-y,-\tau$ as incoming (see left lower part of Fig. 1).

If the worm is leaving the site $(\mathbf{r},\tau)$ passing through an outgoing link $\sigma=x,y,\tau$, then

$$J_{(\mathbf{r},\tau)}^\sigma \rightarrow J_{(\mathbf{r},\tau)}^\sigma + 1. \tag{3}$$

If it is leaving through an incoming link $\sigma=-x,-y,-\tau$, we have

$$J_{(\mathbf{r},\tau)}^\sigma \rightarrow J_{(\mathbf{r},\tau)}^\sigma - 1. \tag{4}$$

Graphically, the convention means that the update in $J_{(\mathbf{r},\tau)}^\sigma$ is $+1$ (-1) if the worm goes in the same (opposite) direction as the arrows denoted in the left lower part of Fig. 1.

The construction ("reptation") of the worm can be described by the following way: First, we start the worm at a random site $N_1=(\mathbf{r}_1,\tau_1)$ of the lattice (black dot in Fig. 1). From this site, the worm has the possibility to go to one of its six neighboring sites. To choose which direction to take, a weight A^σ is calculated for the six directions $\sigma=\pm x,\pm y, \pm \tau$. For A^σ, we use a Metropolis-like weight:

$$A^\sigma = \min\{1,\exp[-(E'_\sigma-E_\sigma)/K]\}, \tag{5}$$

where $E_\sigma=\frac{1}{2}(J^\sigma)^2-\mu J^\sigma\delta_{\sigma,\tau}$ is the local energy carried by the link σ and $E'_\sigma=\frac{1}{2}(J^\sigma\pm 1)^2-\mu(J^\sigma\pm 1)\delta_{\sigma,\tau}$ is the local energy on the link σ *if the worm passes through this link*. The plus or minus sign depends on the incoming or outcoming nature of the link (see above). Please note that there are other possible choices for A_σ [17].

Once the A_σ's are calculated, one computes the probabilities p_σ by normalizing the weights A_σ:

$$p_\sigma=\frac{A_\sigma}{N}, \tag{6}$$

where $N=\Sigma_\sigma A_\sigma$ is the normalization. A random number uniformly distributed in $[0,1]$ is generated and a direction σ chosen according to Eq. (6). Once a direction is chosen, the corresponding link variable J^σ is updated by ± 1 and the worm moved to the next lattice site in this direction.

From there, we apply the same procedure to choose another site, modify the link variable, and move the worm until the worm eventually reaches its starting point and forms a closed loop. This is then the end of this nonlocal move.

To satisfy the detailed balance condition, this worm move must either be accepted or rejected. To check this, one has to store the initial and final normalizations $N_{s_1}^i$ and $N_{s_1}^f$ [calculated as in Eq. (6)] of the weights at the site $s_1=(\mathbf{r}_1,\tau_1)$. $N_{s_1}^i$ is the initial normalization *before* the worm is inserted and $N_{s_1}^f$ is the final normalization *after* the worm reaches the initial point. The worm move is then accepted with probability $N_{s_1}^i/N_{s_1}^f$. If the move is rejected, we have to cancel all changes of the link-currents made during the construction of the worm. During a typical simulation the rejection probability is usually very small.

As already mentioned, the link configurations generated during the worm move do not satisfy the divergenceless constraint, but it is easy to see that the final configuration does. It is important to note that the worm may pass many times though the same link and that at each step, it can bounce back (back track) to the previous lattice site in its path.

A proof of detailed balance for this algorithm is obtained by considering the moves of the worm and of an antiworm, going exactly in the opposite direction [6]. This worm algorithm satisfies ergodicity since the worm can make local loops and line moves as in the local algorithm, which is ergodic.

All in all, the geometrical undirected worm algorithm can be summarized by using the following pseudoalgorithm.

(1) Choose a random initial site $s_1 = (\mathbf{r}_1, \tau_1)$ in the space-time lattice.

(2) For each of the directions $\sigma = \pm x, \pm y, \pm \tau$, calculate the weights $A_{s_i}^{\sigma}$ with $A_{s_i}^{\sigma} = \min[1, \exp(-\Delta E_{s_i}^{\sigma}/K)]$, $\Delta E_{s_i}^{\sigma} = E_{s_i}'^{\sigma} - E_{s_i}^{\sigma}$.

(3) Calculate the normalization $N_{s_i} = \Sigma_{\sigma} A_{s_i}^{\sigma}$ and the associated probabilities $p_{s_i}^{\sigma} = A_{s_i}^{\sigma}/N_{s_i}$.

(4) According to the probabilities $p_{s_i}^{\sigma}$, choose a direction σ.

(5) Update $J_{s_i}^{\sigma}$ for the direction chosen and move the worm to the new lattice site s_{i+1}.

(6) If $s_i \neq s_1$, go to (2).

(7) Calculate the normalizations $\bar{N}_{s_1}$ and N_{s_1} of the initial site s_1, with and without the worm present. Erase the worm with probability $P^e = 1 - \min(1, N_{s_1}/\bar{N}_{s_1})$.

B. The directed geometrical worm algorithm

The above algorithm for geometrical worms is not optimal since the worm quite often will choose to erase itself by returning to the previous site. While it is, in general, not possible to always set this backtracking (or bounce) probability to 0, it is quite straightforward to choose the probabilities $p_{s_i}^{\sigma}$ such that the bounce or backtracking probability will be eliminated in almost all cases and, in general, will be as small as possible. The procedure for doing this amounts to solving a simple linear programming optimizing problem. If we consider models with disorder, this has to be done at each site, but the correctly optimized (biased) probabilities $p_{s_i}^{\sigma}$ can still be tabulated at the start of the calculation.

In order to see how we can minimize the backtracking probability, let us define the 6×6 matrix P_{s_i} of probabilities where the element $P_{s_i}^{kl}$ of the matrix P_{s_i} is given by the conditional probability $p_{s_i}(\sigma_k|\sigma_l)$ for going in the direction σ_k at the site s_i if the worm is coming from the direction σ_l. The backtracking probabilities at the site s_i now correspond to the *diagonal* elements of the matrix P_{s_i}. For the algorithm described in the preceding section, $p_{s_i}(\sigma_k|\sigma_l)$ was simply chosen as $A_{s_i}^{\sigma_k}/N_{s_i}$ independent of σ_l. Thus, all the columns of P_{s_i} were the same and P_{s_i} had, in general, rather large diagonal elements. However, as we shall see below, the matrix P_{s_i} only needs to satisfy the following two conditions in order to define a working geometrical worm algorithm. These conditions are

$$\sum_k p_{s_i}(\sigma_k|\sigma_l) = 1 \tag{7}$$

(probability),

$$\frac{P_{s_i}^{kl}}{P_{s_i}^{lk}} \equiv \frac{p_{s_i}(\sigma_k|\sigma_l)}{p_{s_i}(\sigma_l|\sigma_k)} = \frac{A_{s_i}^{\sigma_k}}{A_{s_i}^{\sigma_l}} \tag{8}$$

(detailed balance).

These conditions are not very restrictive and will in most cases allow us to define a matrix P_{s_i} with all the diagonal elements (backtracking probabilities) equal to 0. The conventional geometrical worm algorithm, discussed in the preceding section, corresponds to $P_{s_i}^{kl} = A_{s_i}^{\sigma_k}/N_{s_i}$.

If we define a function f as the sum of the diagonal elements of P_{s_i}, $f = \Sigma_k P_{s_i}^{kk}$, we can reformulate the search for a matrix P_{s_i} with minimal diagonal elements as a standard linear programming problem. Writing $P_{s_i}^{kk} = 1 - \Sigma_{l \neq k} P_{s_i}^{kl}$ we should minimize f subject to the constraints $\Sigma_{l \neq k} P_{s_i}^{kl} \leq 1 \ \forall k$. The minimum can be found using standard techniques of linear programming [18] and corresponds in almost all cases to $f = 0$. The matrix P_{s_i} depends on the value of all the six link currents $J_{s_i}^{\sigma}$. During the construction of the worm only sites where $J_{s_i}^{-x} + J_{s_i}^{-y} + J_{s_i}^{-\tau} - J_{s_i}^{x} - J_{s_i}^{y} - J_{s_i}^{\tau} = 1$, $s_i \neq s_1$ occur. Since, in general, $|J_{s_i}^{\sigma}|$ will almost never exceed a certain value $J_{\max}$, it is easy to construct a lookup table for the matrices P_{s_i} at the beginning of the simulation and only calculate $P_{s_i}(\{J_{s_i}^{\sigma}\})$ during the simulation if for some $\sigma |J_{s_i}^{\sigma}| > J_{\max}$.

This idea of minimizing the bounce processes is also at the heart of the quantum Monte Carlo directed loop methods [9]. The previous restrictions on the matrix P and the way to solve them numerically are indeed very general, and constitute a simple framework for how one can construct a directed algorithm out of a "standard" nonlocal loop, worm or cluster algorithm.

We can now define a *directed* geometrical worm algorithm with minimal backtracking probability. Using a pseudocode notation, we have the following.

(1) Choose a random initial site $s_1 = (\mathbf{r}_1, \tau_1)$ in the space-time lattice.

(2) If $i = 1$, then for each of the directions $\sigma = \pm x, \pm y, \pm \tau$, calculate the weights $A_{s_i}^{\sigma}$ with $A_{s_i}^{\sigma} = \min[1, \exp(-\Delta E_{s_i}^{\sigma}/K)]$, $\Delta E_{s_i}^{\sigma} = E_{s_i}'^{\sigma} - E_{s_i}^{\sigma}$. Calculate the normalization $N_{s_i} = \Sigma_{\sigma} A_{s_i}^{\sigma}$ and the associated probabilities $p_{s_i}^{\sigma} = A_{s_i}^{\sigma}/N_{s_i}$. Else: According to the incoming direction σ_l, set $p_{s_i}^{\sigma}$ equal to the lth column of P_{s_i}.

(3) According to the probabilities $p_{s_i}^{\sigma}$, choose a direction σ.

(4) Update $J_{s_i}^{\sigma}$ for the direction chosen and move the worm to the new lattice site s_{i+1}.

(5) If $s_i \neq s_1$, go to (2).

(6) Calculate the normalizations $\bar{N}_{s_1}$, of site s_1 with the worm present, and N_{s_1}, without the worm. Erase the worm with probability $P^e = 1 - \min(1, N_{s_1}/\bar{N}_{s_1})$.

Now we turn to the proof of detailed balance for the di-

rected algorithm. Let us consider the case where the worm w visits the sites $\{s_1, \ldots, s_N\}$ where s_1 is the initial site. The worm then goes through the corresponding link variables $\{l_1, \ldots, l_N\}$, with l_i connecting s_i and s_{i+1}. Note that s_N is the last site visited before the worm reaches s_1. Hence, s_N and s_1 are connected by the link l_N. The total probability for constructing the worm w is then given by

$$P_w = P_{s_1}(1 - P_w^e)\frac{A_{s_1}^\sigma}{N_{s_1}}\prod_{i=2}^{N} p_{s_i}(s_{i+1}|s_{i-1}). \qquad (9)$$

The index σ denotes the direction needed to go from s_1 to s_2, P_{s_1} is the probability for choosing site s_1 as the starting point, and P_w^e is the probability for erasing the worm w after construction. $p_{s_i}(s_{i+1}|s_{i-1})$ is the conditional probability for continuing to site s_{i+1}, at site s_i, given that the worm is coming from s_{i-1}. If the worm w has been accepted, we have to consider the probability for reversing the move. That is, we consider the probability for constructing an antiworm $\bar{w}$ annihilating the worm w. We have

$$P_{\bar{w}} = P_{\bar{s}_1}(1 - P_{\bar{w}}^e)\frac{\bar{A}_{\bar{s}_1}^{\bar\sigma}}{\bar{N}_{\bar{s}_1}}\prod_{i=N}^{2} p_{s_i}(s_{i-1}|s_{i+1}). \qquad (10)$$

Here, the index σ denotes the direction needed to go from s_1 to s_N. Note that, in this case, the sites are visited in the opposite order, $s_1, s_N, \ldots, s_2$. From Eq. (8) we have that $p_{s_i}(s_{i+1}|s_{i-1})/p_{s_i}(s_{i-1}|s_{i+1}) = A_{s_i}^{\sigma_k}/\bar{A}_{s_i}^{\sigma_l}$. Since

$$A_{s_i}^\sigma/\bar{A}_{s_i}^\sigma = \exp(-\Delta E_{s_i}^\sigma/K), \quad i=1, \ldots, N, \qquad (11)$$

and since $P_{s_1} = P_{\bar{s}_1}$, we find

$$\frac{P_w}{P_{\bar{w}}} = \frac{1 - P_w^e}{1 - P_{\bar{w}}^e}\frac{\bar{N}_{\bar{s}_1}}{N_{s_1}}\exp(-\Delta E_{\text{Tot}}/K), \qquad (12)$$

where ΔE_{Tot} is the total energy difference between a configuration with and without the worm w present. With our definition of P^e we see that $[1 - P^e(w)]/[1 - P^e(\bar{w})] = N_{s_1}/\bar{N}_{s_1}$, and it follows that

$$\frac{P_w}{P_{\bar{w}}} = \exp(-\Delta E_{\text{Tot}}/K). \qquad (13)$$

For a worm of length N there are N starting sites that will yield the same final configuration. The above proof shows that for each of the starting sites there exists an antiworm $\bar{w}$, such that $P_w = \exp(-\Delta E_{\text{Tot}}/K)P_{\bar{w}}$. Hence, if we denote by μ the configuration without the worm and ν the configuration with the worm and furthermore let $P_w(s_i)$ denote the probability of building the worm w starting from site s_i, we see that

$$\frac{P(\mu\to\nu)}{P(\nu\to\mu)} = \frac{\sum_i^N P_w(s_i)}{\sum_i^N P_{\bar{w}}(s_i)} = \frac{\sum_i^N P_{\bar{w}}(s_i)}{\sum_i^N P_{\bar{w}}(s_i)}\exp(-\Delta E_{\text{Tot}}/K)$$

$$= \exp(-\Delta E_{\text{Tot}}/K). \qquad (14)$$

Ergodicity is proved the same way as for the undirected algorithm as the worm can perform local loops and wind around the lattice in any direction, as in the conventional algorithm.

C. The classical worm algorithm

Prokof'ev and Svistunov [8] have proposed a very elegant way of performing the Monte Carlo simulations on the high-temperature expansion of classical statistical mechanical models using worm algorithms. In order to distinguish between the algorithms we call this algorithm the classical worm algorithm. In a recent study [19] these authors have performed simulations on the quantum rotor model in the link-current representation, Eq. (2). Due to the divergence-less constraint, the classical worm algorithm is in this case quite close to the geometrical worm algorithm proposed previously in Ref. [6] and not directly related to the high-temperature expansion of this model. Recasting their algorithm in the same framework used above we outline our understanding of their algorithm below for comparison.

(1) Choose a random initial site $s_1 = (\mathbf{r}_1, \tau_1)$ in the space-time lattice.

(2) For each of the directions $\sigma = \pm x, \pm y, \pm \tau$, calculate the probabilities $A_{s_i}^\sigma$ with $A_{s_i}^\sigma = \min[1, \exp(-\Delta E_{s_i}^\sigma/K)]$, $\Delta E_{s_i}^\sigma = E'^\sigma_{s_i} - E_{s_i}^\sigma$.

(3) With uniform probability choose a direction σ.

(4) With probability $A_{s_i}^\sigma$ accept to go in the direction σ, and with probability $1 - A_{s_i}^\sigma$ go to (3).

(5) Update $J_{s_i}^\sigma$ for the direction chosen and move the worm to the new lattice site s_{i+1}.

(6) If $s_i \neq s_1$, go to (2).

(7) If $s_i = s_1$, go to (1) with probability p_0 and to (3) with probability $1 - p_0$ [$p_0 \in (0,1)$ and usually $p_0 = 1/2$]. We use $p_0 = 1/2$ in the following.

One advantage of this algorithm is its simplicity and the fact that a constructed worm is always accepted; on the other hand, this algorithm is not directed and steps (3) and (4) above are quite wasteful since in many cases the worm is not moved. This is avoided in the geometrical worm algorithm at the price of occasionally having to reject a complete worm. The geometrical worm algorithm, as described in the preceding sections, should be straightforwardly applicable to the high-temperature expansion, as it was done in Ref. [8], using the classical worm algorithm. We expect that this would enhance the efficiency of the Monte Carlo sampling.

IV. PERFORMANCE OF THE ALGORITHMS

Here we present results on autocorrelation times obtained with both directed and undirected algorithms. For the sake of
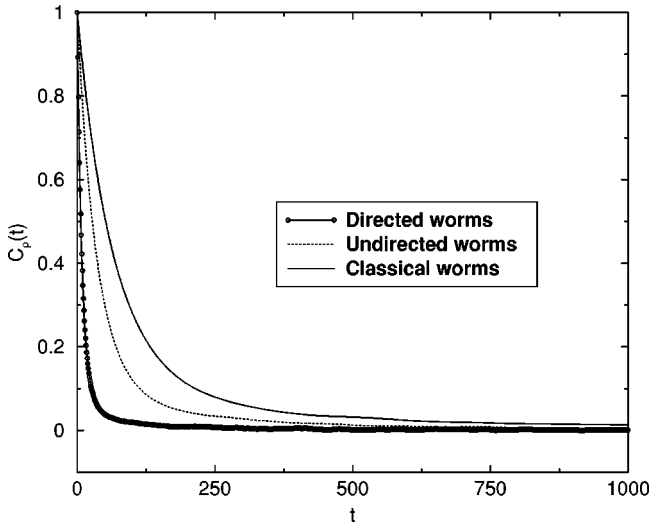
FIG. 2. Autocorrelation function of stiffness vs the Monte Carlo time (defined by one worm construction, see text) for $L = 56$ at $K = 0.333$ for directed (circle), undirected (dotted line), and classical (solid line) algorithms.

FIG. 3. Autocorrelation times of the stiffness ρ for directed, undirected, and classical algorithms vs lattice size L. Shown are the raw autocorrelation times, *before* rescaling to take into account the computational effort expended.

brevity, we restrict ourselves to the case $\mu = 0$, where the critical point is known with high precision, and where results on autocorrelations for the undirected worm algorithm have already been published [6]. All the results presented in this section correspond to run on cubic lattices of 10^7–10^8 Monte Carlo worms for a value of $K = 0.333$, extremely close to the critical point [estimated as $K_c = 0.33305(5)$ in Ref. [6]]. In principle, simulations should be performed on lattices of size $L \times L \times L_\tau$, but here since the dynamical exponent $z = 1$ at $\mu = 0$, we can set $L_\tau = L$. We focus here on calculations of the energy $E = \langle H \rangle$ and the stiffness ρ defined as

$$\rho = \frac{1}{L^3} \left\langle \left(\sum_{\mathbf{r}, \tau} J^x_{\mathbf{r}, \tau} \right)^2 \right\rangle, \tag{15}$$

where L is the linear size of the lattice.

For the simulations with directed worms, we restrict ourselves to $|J| \leq 3$ for the tabulation of probabilities. Probabilities involving higher values of $|J|$ were calculated during the construction of the worm. Such configurations were found to be exceedingly rare.

For the case at hand, only 1% of the "scattering" matrices P_{s_i} contained diagonal elements corresponding to a nonzero back-tracking probability. Moreover, these back-tracking (bounce) processes were found to occur for very unlikely configurations. The acceptance rate $1 - P^e$ is very high for both algorithms at K_c (around 98% for undirected worms and 97% for directed worms for all lattice sizes). For the classical worms, all worms are accepted due to the nature of the algorithm. However, we found that many proposed attempts at changing one link were refused (more than 60% in our simulations).

In Fig. 2, we present the autocorrelation function of stiffness for a lattice size $L = 56$, for directed, undirected, and classical worms. The autocorrelation function $C_\mathcal{O}(t)$ of an observable $\mathcal{O}$ is defined in the standard way:
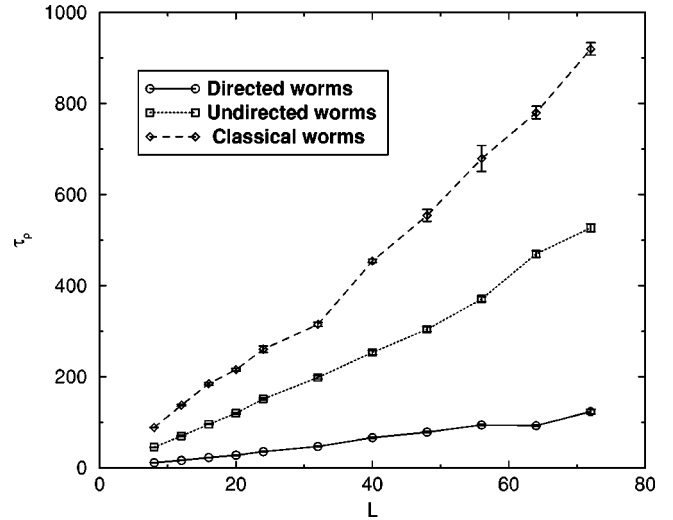
$$C_\mathcal{O}(t) = \frac{\langle \mathcal{O}(t) \mathcal{O}(0) \rangle - \langle \mathcal{O} \rangle^2}{\langle \mathcal{O}^2 \rangle - \langle \mathcal{O} \rangle^2}, \tag{16}$$

where $\langle \cdots \rangle$ denotes statistical average and t is the MC time, measured in the number of constructed worms (accepted or not). We clearly see in Fig. 2 that the directed worm algorithm is more efficient at decorrelating the data than undirected and classical worms, the latter having the longest autocorrelation times.

Now we define the autocorrelation time $\tau_\mathcal{O}$ of an observable $\mathcal{O}$. In Ref. [6], $\tau_\mathcal{O}$ was defined as the greater time of a double-exponential fit of the autocorrelation function. Here we use a much simpler definition, independent of any fitting procedure: $\tau_\mathcal{O}$ is defined as the time where the normalized autocorrelation function decrease below a threshold $t_\mathcal{O}$. We can use different thresholds for different observables $\mathcal{O}$. Since for small lattices and especially for directed worms, autocorrelation times are small, and since $C_\mathcal{O}(t)$ is known only for discrete values of t, $\tau_\mathcal{O}$ is determined by a simple linear interpolation between the two times surrounding the threshold. It is important to note that the values of the autocorrelation time depends on the threshold $t_\mathcal{O}$, but the dependence on lattice size of these autocorrelation times should not change as long as $t_\mathcal{O}$ is small enough. Error bars on $\tau_\mathcal{O}$ have been estimated by slightly changing the threshold, by an amount in between 2% and 5% in this work.

Using the above-mentioned determination of autocorrelation times, we extract autocorrelation times of the stiffness ρ and the energy E for directed, undirected, and classical worms. The threshold was set the same for all algorithms when comparing the same quantity: we used through this work $t_\rho = 0.02$ for the stiffness and $t_E = 0.05$ for the energy. Scaling of these times with the lattice size is shown in Fig. 3 for stiffness and in Fig. 4 for energy. It can be seen that although autocorrelation times grow approximately linearly
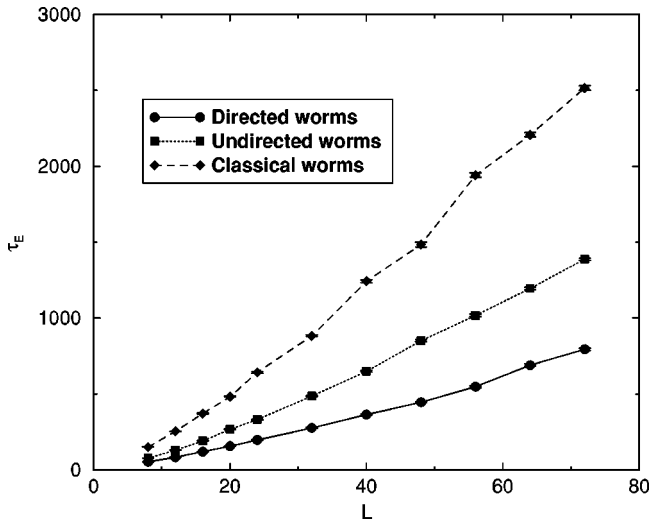
FIG. 4. Autocorrelation times of energy E for the three algorithms vs lattice size L. Shown are the raw autocorrelation times, *before* rescaling to take into account the corresponding computational effort expended.

FIG. 5. Mean size $\langle w \rangle$ divided by $3L^3$ vs lattice size L for directed, undirected, and classical worms.

with lattice size for all algorithms, the slope is significantly smaller for the directed worm algorithm.

It is clear from these results that the directed algorithm significantly reduces the autocorrelation times. However, the average size of the directed worms could be larger, and hence on average consume more computational time. For all algorithms the computational effort is linearly proportional to the length of the worm. To make an honest comparison, we therefore have to multiply the autocorrelation times by the number of attempted changes per link, which we define as $\langle w \rangle /(3L^3)$, where $\langle w \rangle$ is the mean worm size (the mean number of links the worm has attempted to visit), L is the lattice size preceded by an irrelevant factor indicating that there are three links per site. For the classical worms, the mean worm size $\langle w \rangle$ is defined as the *total* number of proposed attempts (step 4 in the pseudocode presentation in Sec. III C). In order to make an unbiased comparison of the three algorithms, it is here necessary to include the updates refused during the construction of the classical worms in the definition of $\langle w \rangle$.

As mentioned, the computational effort (the CPU time) is linear in $\langle w \rangle$ for all algorithms. An equivalent rescaling was used in Ref. [6] in order to make a fair comparison with the local algorithm. Figure 5 shows the mean worm size $\langle w \rangle$ (divided by $3L^3$) for the three algorithms versus lattice size, corresponding to the average fraction of the total number of links occupied by the worm. In both cases, we see that this fraction decreases with L. We also note that the classical worms are longer than that in the other proposed algorithms, which will result in larger autocorrelation times. Directed and undirected worms are almost of the same size, with very slightly larger directed worms. The corresponding effect on the value of rescaled effort (presented in the following paragraph) will be small when comparing autocorrelation times for those algorithms; however, we wish to keep it present for a more fair analysis.

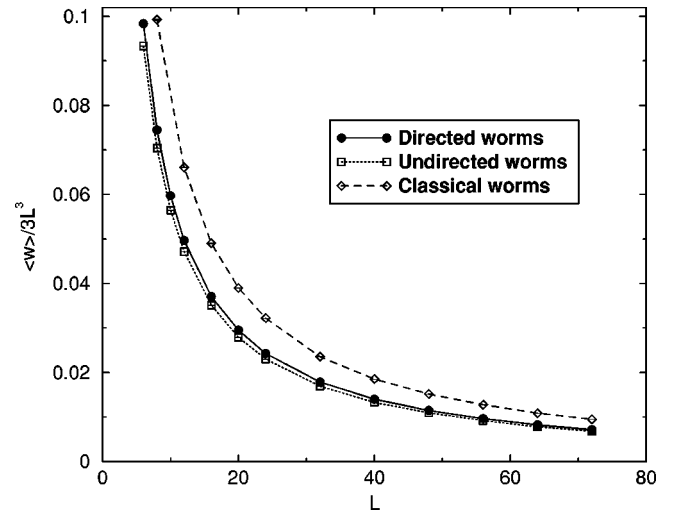Having discussed the behavior of $\langle w \rangle$ we now take into

account the computational effort used to construct the worm by rescaling the autocorrelation times by $\langle w \rangle /(3L^3)$. We show in Fig. 6 the rescaled autocorrelation times for the three algorithms. We find that the autocorrelation times per link stay reasonably small for all algorithms, but the directed algorithm clearly gives better results, with autocorrelation times smaller by a factor around 4 (1.5–1.7) for the stiffness (energy) with respect to the undirected algorithm, and a factor around 10 (4) with respect to the classical worm for the largest sizes. The fact that both algorithms are more efficient at decorrelating the stiffness than the energy seems to indicate that the worms couple more effectively to "winding modes," from which the stiffness is uniquely determined, than to simple local modes that determine the energy. With the same argument, we can see that directed worms are more efficient at updating winding modes than undirected or classical worms.
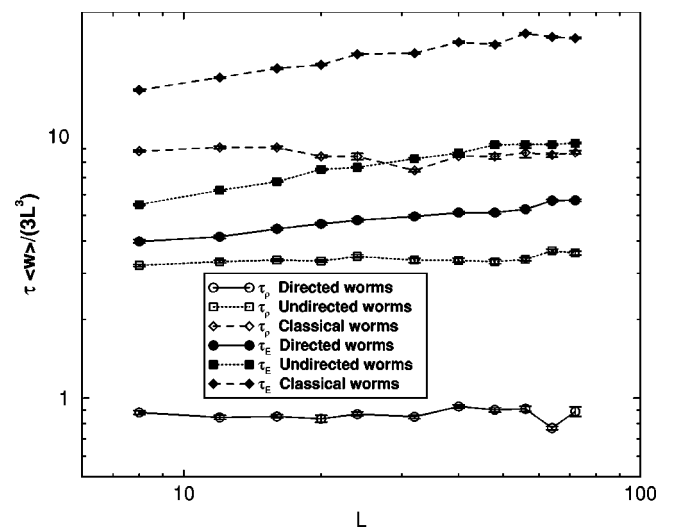

FIG. 6. Autocorrelation times of stiffness ρ and energy E for the three presented algorithms vs lattice size L. These autocorrelation times are *rescaled* autocorrelation times where the computational effort is taken into account.
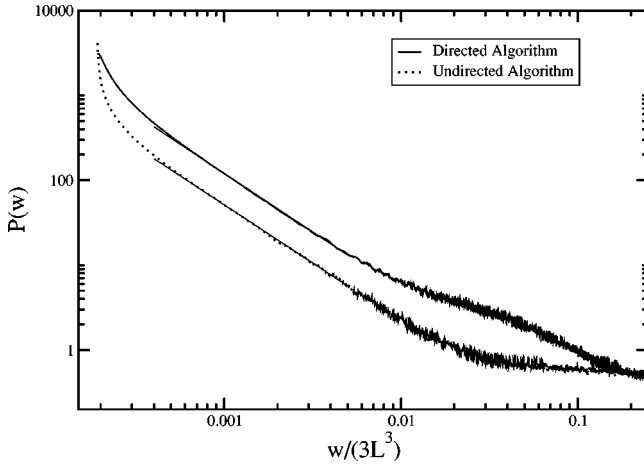
FIG. 7. The probability density $P(w)$ for generating a worm occupying a fraction of $w/3L^3$ of the lattice, as a function of $w/3L^3$ for the directed and undirected algorithms. Shown are the results for a lattice of linear size $L=56$ at $K=K_c$. The solid lines indicate power-law fits to the data.

The actual distribution of the size of the worms generated, $P(w)$, is also of interest. In Fig. 7 we show results for the probability density, $P(w)$ for generating a worm occupying a fraction of $w/3L^3$ of the lattice, as a function of $w/3L^3$ for the directed and undirected algorithms. The classical worm algorithm has a distribution identical to that shown for the undirected algorithm. Clearly, the directed worms have a somewhat broader distribution but for both algorithms the distribution follows a power-law form $P(w)\sim w^{-\alpha}$ with $\alpha \sim 1.37$. The power-law behavior is to be expected since the simulations were performed at the critical point. Away from the critical point we have verified that the initial power-law form crosses over to an exponential behavior at large arguments.

To summarize, we find that in all cases, rescaled autocorrelation times stay almost constant with the lattice size, but could also be fitted with a very small power-law or logarithm, showing an almost complete elimination of critical slowing down. All in all, the main result of this section is that directed worms produce less correlated data (smaller autocorrelation times), even if the scaling is good for all the three (directed, undirected, and classical) algorithms. We also note that the geometrical worm algorithms perform better than the classical worm algorithm.

The fact that both directed and undirected algorithms have almost the same scaling of the rescaled autocorrelation time with L seems also to be observed for the directed loop quantum Monte Carlo cluster algorithms [9].

V. THE CORRELATION FUNCTIONS

A. Measurements of correlation functions with worm algorithms

For the quantum rotor model, the correlation functions of interest have the following form [11]:

$$C(\mathbf{r},\mathbf{r}',\tau,\tau')=\langle e^{i[\hat{\theta}_{\mathbf{r}}(\tau)-\hat{\theta}_{\mathbf{r}'}(\tau')]}\rangle, \qquad (17)$$

where $\hat{\theta}$'s are the operators for the phase of the bosons, and $e^{i\hat{\theta}_{\mathbf{r}}(\tau)}=e^{\tau H}e^{i\hat{\theta}_{\mathbf{r}}}e^{-\tau H}$. Due to translational invariance, $C(\mathbf{r},\mathbf{r}',\tau,\tau')=C(\mathbf{r}-\mathbf{r}',\tau-\tau')$. Physically this corresponds to creating a particle at $(\mathbf{r},\tau)$ and destroying it at $(\mathbf{r}',\tau')$. When this correlation function is mapped onto the link-current representation, the creation and destruction of the particle is interpreted as a particle current going from $(\mathbf{r},\tau)$ to $(\mathbf{r}',\tau')$. As is evident from the definition of the correlation function in Eq. (17) the value of the correlation function cannot depend on the specific path taken from $(\mathbf{r},\tau)$ to $(\mathbf{r}',\tau')$, as long as we take into account the fact that going in the x,y,τ directions increases the local current, whereas going in the $-x,-y,-\tau$ directions decreases the local current. In the link-current representation this correlation function can be written [11] in the following way:

$$C(\mathbf{r},\tau)=\left\langle \prod_{(\mathbf{r}_i,\tau_i)\in\text{ path}} \exp\left\{ -\frac{1}{K}\left(\text{sgn}(\sigma_i)(J^\nu_{(\mathbf{r}_i,\tau_i)} \right.\right.\right.$$
$$\left.\left.\left. -\delta_{\sigma_i,\pm\tau}\tilde{\mu}_{\mathbf{r_i}})+\frac{1}{2}\right)\right\}\right\rangle, \qquad (18)$$

where "path" is any path on the space-lattice connecting two points a distance $(\mathbf{r},\tau)$ apart and σ_i is the direction needed to go from $(\mathbf{r_i},\tau_i)$ to $(\mathbf{r}_{i+1},\tau_{i+1})$, $\sigma_i=\pm x,\pm y,\pm\tau$. When going in the direction $\sigma_i=x,y,\tau$, we propagate a *particle* and the correlation function corresponds to *incrementing* the corresponding link variable by 1. When going in the direction $\sigma_i=-x,-y,-\tau$, we propagate a *hole* in the x,y,z directions and the correlation function corresponds to *decrementing* the corresponding link variable by 1. This is indicated in Eq. (18) by $\text{sgn}(\sigma_i)$. Furthermore, we only get a contribution from $\mu_{\mathbf{r_i}}$ whenever we go in the τ direction and we take this into account by $\delta_{\sigma_i,\pm\tau}$. If we define $J^{-x}_{(x,y,\tau)}=-J^x_{(x-1,y,\tau)}$ with analogous definitions for the other directions we see that by incrementing and decrementing the link-current variables in the above manner $\Sigma_\sigma J^\sigma_{(\mathbf{r},\tau)}=0$ at all the sites between $(\mathbf{r_i},\tau_i)$ and $(\mathbf{r}_{i+1},\tau_{i+1})$. The current is divergenceless at all the intermediary sites. The sites $(\mathbf{r_i},\tau_i)$ and $(\mathbf{r}_{i+1},\tau_{i+1})$ will have nonzero divergence with $\Sigma_\sigma J^\sigma_{(\mathbf{r},\tau)}=1$ corresponding to a site where a particle is created (or a hole destroyed). A site with $\Sigma_\sigma J^\sigma_{(\mathbf{r},\tau)}=-1$ is a site where a hole is created (or a particle destroyed). In Fig. 8 we show two possible paths $\mathcal{P}_a$ and $\mathcal{P}_b$ for the evaluation of the correlation function $C(\mathbf{r},\tau)$. As usual, $C(\mathbf{r},\tau)=C(\mathbf{r}+L,\tau+L_\tau)$ but $C(\mathbf{r},\tau)$ is, in general, not equal to $C(\mathbf{r},-\tau)$.

Previous work [11,14] have attempted to calculate the correlation function by evaluating the thermal expectation value in Eq. (18) along a straight path from $(\mathbf{r},\tau)$ to $(\mathbf{r}',\tau')$. Although formally correct, this method fails for large arguments of the correlation function due to the fact that for a given configuration of the link variables roughly only *one* specific path between $(\mathbf{r},\tau)$ and $(\mathbf{r}',\tau')$ will yield a contribution of order 1.

The geometrical worm algorithm allows for a much more efficient way of evaluating the correlation functions. In essence, before the worm returns to the starting site, the path of
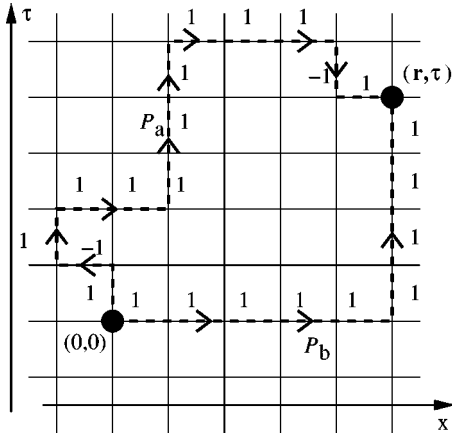
FIG. 8. Two possible paths $\mathcal{P}_a$ and $\mathcal{P}_b$ for the evaluation of $C(\mathbf{r},\tau)$. When the path is going in the x, y, τ directions a *particle* is propagated in the forward direction corresponding to an *increment* in the current. When the path is going in the $-x, -y, -\tau$ directions, we propagate a *hole* in the forward direction corresponding to a *decrement* in the current. The solid circles correspond to sites where a *single* particle is created or destroyed.

the worm corresponds precisely to the creation of a particle at site s_1 and the destruction at the current site s_i with a current going between the two sites. This is precisely the correlation function that we want to calculate. More precisely we extend Eq. (18) to include a summation over all possible paths:

$$
C(\mathbf{r},\tau) = \frac{1}{N_{\mathcal{P}}} \sum_{\mathcal{P}} \left\langle \prod_{(\mathbf{r}_i,\tau_i) \in \mathcal{P}} \exp\left\{ -\frac{1}{K}\left(\mathrm{sgn}(\sigma_i)(J^{\nu}_{(\mathbf{r}_i,\tau_i)}\right.\right.\right.
$$
$$
\left.\left.\left. - \delta_{\sigma_i, \pm \tau \tilde{\mu}_{\mathbf{r}_i}} + \frac{1}{2}\right) \right\} \right\rangle . \tag{19}
$$

Here $\mathcal{P}$ is a path for the correlation function and $N_{\mathcal{P}}$ is the number of paths included in the sum. Since the geometrical worm algorithm generates paths between $(\mathbf{r},\tau)$ and $(\mathbf{r}_n,\tau_n)$ with the correct exponential factor (except for a multiplicative constant), it is now easy to calculate the correlation functions.

Suppose that, by using either the directed or undirected worm algorithm, we have reached the equilibrium configuration μ. The probability for, during the construction of a worm starting at site $s_1 = (\mathbf{r}_1, \tau_1)$, creating a current j that reaches $s_n = (\mathbf{r}_n, \tau_n) \neq s_1$ is given by

$$
P(j; \mu \to \mu') = P(s_1) \prod_{i=1}^{n-1} \frac{A^{\sigma}_{s_i}}{N_{s_i}} \tag{20}
$$

for the undirected algorithm. For the directed algorithm we have

$$
P(j; \mu \to \mu') = P(s_1)\frac{A^{\sigma}_{s_1}}{N_{s_1}} \prod_{i=2}^{n-1} p_{s_i}(s_{i+1}|s_{i-1}). \tag{21}
$$

If we call the resulting state μ' we can calculate the probability for, starting from μ', creating an anticurrent $\bar{j}$, going from s_n to s_1. We find, for the undirected algorithm,

$$
P(\bar{j}; \mu' \to \mu) = P(s_n)\prod_{i=n}^{2} \frac{\bar{A}^{\sigma}_{s_i}}{N_{s_i}}, \tag{22}
$$

and for the directed algorithm,

$$
P(\bar{j}; \mu' \to \mu) = P(s_N)\frac{\bar{A}^{\sigma}_{s_n}}{\bar{N}_{s_n}} \prod_{i=n-1}^{2} p_{s_i}(s_{i-1}|s_{i+1}). \tag{23}
$$

In both cases we see that

$$
\frac{P(j; \mu \to \mu')}{P(\bar{j}; \mu' \to \mu)} = \frac{\bar{N}_{s_n}}{N_{s_1}} \prod_{(\mathbf{r}_i,\tau_i) \in \mathcal{P}} \exp\left\{ -\frac{1}{K}\left(\mathrm{sgn}(\sigma_i)(J^{\nu}_{(\mathbf{r}_i,\tau_i)}\right.\right.
$$
$$
\left.\left. - \delta_{\sigma_i, \pm \tau \tilde{\mu}_{\mathbf{r}_i}} + \frac{1}{2}\right) \right\}. \tag{24}
$$

Hence, we see that for both algorithms the intermediate states generated during the construction of the worm follows precisely the distribution needed apart from the factor $\bar{N}_{s_n}/N_{s_1}$. It follows that whenever a worm reaches a point, a distance $(\mathbf{r},\tau)$ away from the initial point, it contributes a factor of $N_{s_1}/\bar{N}_{s_n}$ to the correlation function of argument $(\mathbf{r},\tau)$. Note that it follows from the above proof that *all* worms, even those that are finally rejected, have to be included in the calculation of Green's functions. As per definition $C(0,0,0) \equiv C(L,L,L_{\tau}) \equiv 1$.

B. Results

The above procedure is straightforward to implement. Suppose we want to calculate Green's functions for a $(d+1)$-dimensional system with $d=2$. Since the two space directions are equivalent by symmetry, it is only necessary to calculate $C(x,\tau)$. This is easily done by keeping track of the position of the worm during construction. If the relative position of the worm with respect to its starting point s_1 is denoted by (x_r, y_r, τ_r), when the worm has reached site s_n, we add $N_{s_1}/\bar{N}_{s_n}$ to $C(x_r, y_r, \tau_r)$. This can be done with very little computational effort and since an enormous amount of worms are generated during the simulation extremely good statistics can be obtained for $C(x_r, y_r, \tau_r)$ by averaging over the worms (which cannot be achieved with the local algorithm). As mentioned, in order not to bias the calculation, even worms that are eventually rejected should be included for a correct calculation of Green's functions. In Fig. 9 we show results for Green's function as a function of x for a system of size $L^3, L=64$. For this simulation the directed algorithm was used with a total number of worms equal to 1.5×10^8. It is easy to obtain extremely small error bars on Green's functions even for very large system sizes. For the results shown in Fig. 9, $\mu=0$ and by symmetry $C(\tau)$ is identical to $C(x)$. From scaling relations [21] $C(r)$ is ex-
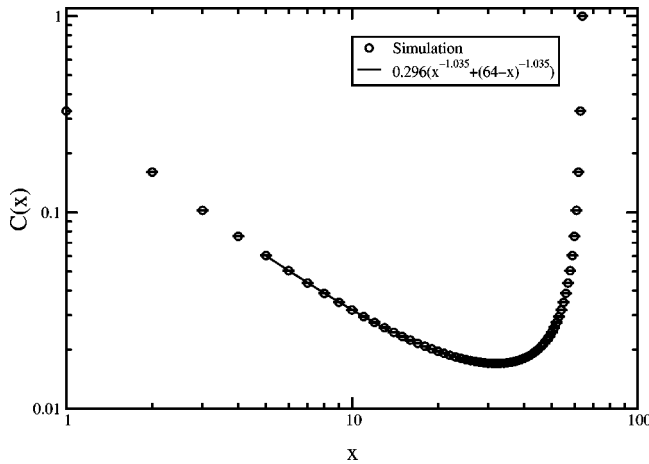
FIG. 9. Green's function $C(x)$ for a system of size $L^3, L = 64$ at $K = K_c = 0.33305$, $\mu = 0$, as a function of x. The solid line indicates a power-law fit of the form $0.296[x^{-1.035} + (64 - x)^{-1.035}]$.

pected to decay as $r^{-(d-2+z+\eta)}$, where z is the dynamical critical exponent. With $\mu = 0$, $z = 1$ we find $C(r) \sim r^{-(1+\eta)}$. Fitting to this form we find $\eta = 0.035(5)$. The obtained critical exponents are in excellent agreement with previous work [11] and more recent high-precision estimates for the critical exponents of the 3D XY model [20].

It would be of much interest to calculate $C(\mathbf{r}, \tau)$ for $\mu \neq 0$, using this method. Such calculations are currently in progress [22].

VI. SUMMARY AND DISCUSSION

We have proposed a directed worm algorithm for the quantum rotor model. This algorithm is an improvement of the "undirected" algorithm presented in Ref. [6]. It has been shown that by adjusting the degrees of freedom left in the detailed balance condition, one can construct a more efficient algorithm by minimizing the backtracking (bounce) probability for the worm to erase itself. The minimal probabilities can be found by solving a linear programming problem subject to a few well-defined constraints. A proof of detailed balance for the directed case has also been presented. The directed and undirected algorithms are identical except for the fact that appropriately defined local probabilities p_σ for moving the worm through the lattice are chosen in an optimal manner for the directed algorithm. Hence, only a very

limited amount of additional programming has to be done to implement the directed algorithm.

These central ideas for this directed algorithm can be straightforwardly applied to directed quantum Monte Carlo (QMC) loop algorithms [9] and one can avoid an analytical calculation for each new model where one wants to implement a directed algorithm. More generally speaking, we believe that the framework presented here could be useful for constructing new algorithms for other models, for example, classical spin models [23].

We have shown the superiority of the directed algorithm as compared to the undirected one and to the approach ("classical worms") proposed in Ref. [8] by calculating autocorrelation times of different observables near a critical point. Although the computational gain is not as drastic as when passing from a local update algorithm to a worm algorithm [6,17], we showed that one gains a factor ranging from 1.5 to 10 (depending on the quantity and on the comparison) for the simulations considered here. We did not try to estimate autocorrelation exponent z for the algorithms, because in all cases, it is small (as can be seen in Fig. 6) and it would be hard to determine with high precision. Looking at the data, it is likely that values of z for all algorithms are the same or quite close. A logarithmic dependence of τ on L, indicating $z = 0$, cannot also be excluded.

In this paper, we have also derived an efficient way of measuring correlation functions during the worm constructions. This feature is similar to other worm algorithms [4,5], but here we show, including analytical arguments, that it also works for directed worms. The situation for directed QMC loop algorithms [9] is less certain, even if some results were recently presented in Ref. [24].

The directed worm algorithm could be specially useful to study the transition for a noncommensurate value of the chemical potential in the pure quantum rotor model or for the disordered case, where very strong finite size effects have been identified [17,19,22].

[1] R.H. Swendsen and J.S. Wang, Phys. Rev. Lett. **58**, 86 (1987).

[2] U. Wolff, Phys. Rev. Lett. **62**, 361 (1989).

[3] H.G. Evertz, G. Lana, and M. Marcu, Phys. Rev. Lett. **70**, 875 (1993); H.G. Evertz, Adv. Phys. **52**, 1 (2003).

[4] A.W. Sandvik, Phys. Rev. B **59**, R14157 (1999); A. Dorneich and M. Troyer, Phys. Rev. E **64**, 066701 (2001).

[5] N.V. Prokof'ev, B.V. Svistunov, and I.S. Tupitsyn, Phys. Lett. A **238**, 253 (1998).

[6] F. Alet and E.S. Sørensen, Phys. Rev. E **67**, 015701(R) (2003).

[7] T. Banks, R. Myerson, and J. Kogut, Nucl. Phys. B **129**, 493

(1977); P.R. Thomas and M. Stone, *ibid.* **114**, 513 (1978); M.E. Peskin, Ann. Phys. (N.Y.) **113**, 122 (1978); R. Savit, Rev. Mod. Phys. **52**, 453 (1980).

[8] N. Prokof'ev and B. Svistunov, Phys. Rev. Lett. **87**, 160601 (2001).

[9] O.F. Syljuåsen and A.W. Sandvik, Phys. Rev. E **66**, 046701 (2002); K. Harada and N. Kawashima, *ibid.* **66**, 056705 (2002); O.F. Syljuåsen, *ibid.* **67**, 046701 (2003); J. Smakov, K. Harada, and N. Kawashima, e-print cond-mat/0301416; F. Alet, S. Wessel, and M. Troyer (unpublished).

[10] S. Sachdev, *Quantum Phase Transitions* (Cambridge University Press, Cambridge, 1999).

[11] E.S. Sørensen *et al.*, Phys. Rev. Lett. **69**, 828 (1992); M. Wallin *et al.*, Phys. Rev. B **49**, 12 115 (1994).

[12] M.-C. Cha *et al.*, Phys. Rev. B **44**, 6883 (1991).

[13] A. vanOtterlo and K.-H. Wagenblast, Phys. Rev. Lett. **72**, 3598 (1994); A. vanOtterlo *et al.*, Phys. Rev. B **52**, 16 176 (1995).

[14] J. Kisker and H. Rieger, Phys. Rev. B **55**, R11981 (1997); Physica A **246**, 348 (1997).

[15] S.Y. Park *et al.*, Phys. Rev. B **59**, 8420 (1999); J.-W. Lee, M.-C. Cha, and D. Kim, Phys. Rev. Lett. **87**, 247006 (2001).

[16] A.B. Bortz, M.H. Kalos, and J.L. Lebowitz, J. Comput. Phys. **17**, 10 (1975); M.N. Rosenbluth and A.W. Rosenbluth, J.

[17] F. Alet, Ph.D. thesis, Université Paul Sabatier, Toulouse, 2002 (unpublished).

[18] *Numerical Recipes in C++*, edited by W. Press *et al.* (Cambridge University Press, Cambridge, 2002).

[19] N.V. Prokof'ev and B.V. Svistunov, e-print cond-mat/0301205.

[20] See I. Dukovski, J. Machta, and L.V. Chayes, Phys. Rev. E **65**, 026702 (2002), and references therein.

[21] M.P.A. Fisher, P.B. Weichman, G. Grinstein, and D.S. Fisher, Phys. Rev. B **40**, 546 (1989).

[22] F. Alet and E. S. Sørensen (unpublished).

[23] P. Hitchcock, F. Alet, and E. Sørensen (unpublished).

[24] A. Cuccoli *et al.*, Phys. Rev. B **67**, 104414 (2003).

Chem. Phys. **23**, 356 (1955).