JHEP HYPER VERSION

Received: *August 14, 2001*, Accepted: *September 5, 2001*

Locality and exponential error reduction in numerical lattice gauge theory

Martin Lüscher*

CERN, Theory Division
CH-1211 Geneva 23, Switzerland
E-mail: luscher@mail.cern.ch

Peter Weisz†

Center for Computational Physics, University of Tsukuba
Tsukuba, Ibaraki 305-8577, Japan
E-mail: weisz@draco.rccp.tsukuba.ac.jp

Abstract: In non-abelian gauge theories without matter fields, expectation values of large Wilson loops and loop correlation functions are difficult to compute through numerical simulation, because the signal-to-noise ratio is very rapidly decaying for increasing loop sizes. Using a multilevel scheme that exploits the locality of the theory, we show that the statistical errors in such calculations can be exponentially reduced. We explicitly demonstrate this in the SU(3) theory, for the case of the Polyakov loop correlation function, where the efficiency of the simulation is improved by many orders of magnitude when the area bounded by the loops exceeds $1\,\mathrm{fm}^2$.

Keywords: Lattice Gauge Field Theories, Confinement, Statistical Methods.

*On leave from Deutsches Elektronen-Synchrotron DESY, D-22603 Hamburg, Germany
†Permanent address: Max-Planck-Institut für Physik, D-80805 München, Germany

JHEP09(2001)010

Contents

JHEP09(2001)010

1. Introduction

The euclidean expectation values of Wilson loops and their products are probably the most natural quantities to consider in non-abelian gauge theories. They encode much of the physical information in these theories, such as the particle spectrum, for example, and the strength of the force between static colour sources. These properties can, however, only be extracted reliably if one is able to calculate the relevant expectation values accurately over a significant range of loop sizes and distances.

In lattice gauge theory the computation of loop expectation values through numerical simulation is in principle straightforward. The problem is that the signal-to-noise ratio is exponentially decreasing for large loops, roughly proportionally to $e^{-\sigma A}$ in the confinement phase, where σ denotes the string tension and A the minimal area spanned by the loop. According to this law (and if we set $\sigma \simeq 1\,\mathrm{GeV/fm}$), an increase in A by $1\,\mathrm{fm}^2$ at fixed relative errors requires the statistics to be multiplied by about 3×10^4. Computers are rapidly becoming faster, but it is clear that a brute-force approach will not pay under these conditions, i.e. progress in this field has to come mainly from better algorithms and computational strategies.

A significant improvement is achieved, for example, by the "multihit" method [1], where the gauge field variables in the Wilson loop are replaced by their average in the background of all other field variables. This has no effect on the loop expectation value, but the statistical errors are reduced by an exponential factor with exponent proportional to the length of the loop. Link-blocking techniques [2, 3] and variational methods [4, 5] are further improvements that are known to be effective in practice and are widely used.

In spite of all these advances, it remains difficult to reach loop areas A greater than about $1\,\mathrm{fm}^2$. Moreover, as is generally the case when applying variational techniques, the calculation is biased to some extent by the choice of basis operators. In the presence of matter fields, for example, string breaking is not observed unless basis elements for both the string and the two-meson states are included [6]–[8].

In this paper we describe a multilevel simulation scheme that leads to an exponential error reduction with exponent approximately proportional to the area A. The algorithmic idea is essentially the same as in the multihit method [1] but is applied to pairs of links instead of single links. It is then possible to iterate the procedure according to a hierarchical scheme that builds up averages over increasingly large sublattices. This is not as complicated as it sounds, and if use is made of recursive functions (programs that call themselves), the algorithm is in fact easy to program.

2. Preliminaries

For clarity we shall only consider the case of the pure SU(3) gauge theory in this paper, even though the techniques discussed later are expected to be more widely applicable. The theory is set up on a 4-dimensional hypercubic lattice with spacing a, time-like extent T and spatial size L in the usual way. In particular, the gauge field is represented by link variables $U(x,\mu)$ with values in SU(3). We impose periodic boundary conditions in all directions and take the standard expression [9]

$$S[U] = \frac{1}{g_0^2} \sum_{x,\mu,\nu} \mathrm{tr}\left\{1 - U(x,\mu)U(x+a\hat{\mu},\nu)U(x+a\hat{\nu},\mu)^{-1}U(x,\nu)^{-1}\right\} \tag{2.1}$$

for the lattice action, where g_0 denotes the bare coupling and $\hat{\mu}$ the unit vector in direction μ.

For any oriented closed curve $\mathcal{C}$ on the lattice, the associated Wilson loop

$$W(\mathcal{C}) = \mathrm{tr}\{U(\mathcal{C})\} \tag{2.2}$$

is the trace of the ordered product $U(\mathcal{C})$ of the link variables along the curve. In the special case of a straight line that passes through the point x and winds once around the lattice in the negative time direction, $W(\mathcal{C})$ is referred to as a Polyakov loop and is denoted by $P(x)$.

The expectation value of any product $\mathcal{O}$ of Wilson loops is defined by

$$\langle \mathcal{O} \rangle = \frac{1}{\mathcal{Z}} \int \mathrm{D}[U]\, \mathcal{O}\, \mathrm{e}^{-S[U]}, \qquad \mathrm{D}[U] = \prod_{x,\mu} \mathrm{d}U(x,\mu)\,, \tag{2.3}$$

where $\mathcal{Z}$ is a normalization factor such that $\langle 1 \rangle = 1$ and $\mathrm{d}U(x,\mu)$ the normalized invariant measure on SU(3). To keep the discussion as transparent as possible, our attention will be restricted, in the following, to the Polyakov loop correlation function $\langle P(x)^* P(y) \rangle$ and to the expectation value of plane rectangular Wilson loops at $x_2 = x_3 = 0$, with corners $(0,0), (t,0), (t,r), (0,r)$ in the (x_0, x_1) plane.

3. Factorization of the functional integral

In this section we rewrite the Wilson loop expectation values in a partly factorized form that is closely related to the transfer matrix representation. Tensor products of pairs of link variables, the *two-link operators*, play an important rôle in this transformation, and we thus introduce these first. The factorization reflects the local structure of the theory and will lead us (in section 4) to the multilevel simulation algorithm alluded to above.

3.1 Two-link operators

The expectation value of the rectangular loop defined at the end of the previous section may be interpreted as a transition matrix element between states that describe

JHEP09(2001)010

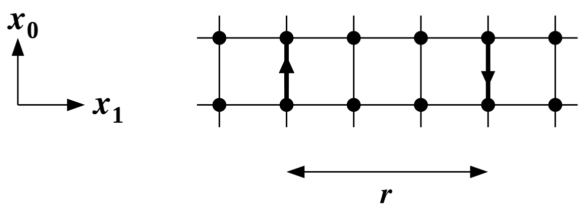

Figure 1: The two-link operator $\mathbb{T}(x_0)$ is equal to the tensor product of the time-like link variables at $x = (x_0, 0, 0, 0)$ and $x = (x_0, r, 0, 0)$.

a pair of static colour charges separated by a distance r [9]. The charges are created at time $x_0 = 0$ and annihilated at $x_0 = t$ through the line operator

$$\mathbb{L}(x_0)_{\alpha\beta} = \{U(x,1) \cdots U(x + (r-a)\hat{1}, 1)\}_{\alpha\beta}\,, \qquad x = (x_0, 0, 0, 0)\,, \tag{3.1}$$

and its complex conjugate respectively. Between these times the charge propagation is represented by the two-link operators

$$\mathbb{T}(x_0)_{\alpha\beta\gamma\delta} = U(x,0)^*_{\alpha\beta} U(x + r\hat{1}, 0)_{\gamma\delta} \tag{3.2}$$

(see figure 1). If we group the indices in pairs, (α, γ) and (β, δ), these operators assume the form of complex 9×9 matrices that act on colour tensors in the $\mathbf{3}^* \otimes \mathbf{3}$ representation of SU(3).

The multiplication of the time-like link variables in the Wilson loop corresponds to the multiplication law

$$\{\mathbb{T}(x_0)\mathbb{T}(x_0 + a)\}_{\alpha\beta\gamma\delta} = \mathbb{T}(x_0)_{\alpha\lambda\gamma\epsilon}\mathbb{T}(x_0 + a)_{\lambda\beta\epsilon\delta} \tag{3.3}$$

for the two-link operators. Using this rule the factorized expression

$$W(\mathcal{C}) = \mathbb{L}(0)_{\alpha\gamma} \{\mathbb{T}(0)\mathbb{T}(a) \cdots \mathbb{T}(t-a)\}_{\alpha\beta\gamma\delta}\, \mathbb{L}(t)^*_{\beta\delta} \tag{3.4}$$

is obtained, while in the case of a pair of Polyakov loops at distance r along the x_1 axis, there are no line operators and the corresponding formula,

$$P(x)^* P(x + r\hat{1}) = \{\mathbb{T}(0)\mathbb{T}(a) \cdots \mathbb{T}(T-a)\}_{\alpha\alpha\gamma\gamma}\,, \tag{3.5}$$

assumes an even simpler form.

3.2 Sublattice expectation values

Averages over sublattices is the next topic that we need to discuss. In the present context the relevant sublattices are time-slices of variable thickness. Such a time-slice consists of all lattice points with time coordinates in a given interval $[x_0, y_0]$ and its boundary are the equal-time hyperplanes at times x_0 and y_0.

Lattice gauge theory on a time-slice can be studied independently of the surrounding lattice if the spatial link variables at the boundaries are held fixed. This decoupling property is a consequence of the locality of the action (2.1) and more precisely of the fact that the action involves plaquette loops only. The link variables in the interior of the time-slice are then the dynamical degrees of freedom that are to be integrated over when calculating sublattice expectation values. We define the latter in the obvious way, with the action reduced to those terms that depend on the dynamical link variables. Sublattice expectation values are denoted by a square bracket $[\cdots]$ in order to distinguish them from the expectation values $\langle\cdots\rangle$ on the whole lattice.

Later we shall be dealing mostly with time-slice expectation values of products of two-link operators. These are explicitly given by

$$[\mathbb{T}(x_0)\cdots\mathbb{T}(y_0-a)] = \frac{1}{\mathcal{Z}_{\rm sub}}\int \mathrm{D}[U]_{\rm sub}\,\mathbb{T}(x_0)\cdots\mathbb{T}(y_0-a)\,\mathrm{e}^{-S[U]_{\rm sub}}\,, \tag{3.6}$$

where the index "sub" indicates that the sublattice expression is meant. In particular, the sublattice partition function $\mathcal{Z}_{\rm sub}$ is determined by the requirement that $[1]=1$. It will be important in the following to keep in mind that sublattice expectation values are well-defined functions of the link variables at the boundary of the sublattice, but do not depend on the gauge field elsewhere on the lattice.

3.3 Hierarchical integration formulae

We now insert the product representation (3.4) in the expectation value of the Wilson loop and reorganize the functional integral in a hierarchical way. The manipulations are essentially the same as in the case of the multihit method [1], but they are applied to the two-link operators and are carried to higher levels.

We first note that the expectation value can be rewritten in the form

$$\langle W(\mathcal{C})\rangle = \Big\langle \mathbb{L}(0)_{\alpha\gamma}\,\{[\mathbb{T}(0)\cdots\mathbb{T}(t-a)]\}_{\alpha\beta\gamma\delta}\,\mathbb{L}(t)^*_{\beta\delta}\Big\rangle\,, \tag{3.7}$$

where the square bracket stands for the expectation value in the time-slice $[0,t]$. The Wilson loop expectation value may thus be computed in two steps, first calculating the time-slice average of the product of the two-link operators, for the current configuration of the gauge field at the boundary of the time-slice, and then the full lattice average of the product of the time-slice expectation value with the line operators.

We can in fact derive various expressions of this type, with many levels of nested averages. The key observation is that the time-slice expectation values are compatible with each other in the sense that they satisfy identities like

$$[\mathbb{T}(x_0)\mathbb{T}(x_0+a)] = [[\mathbb{T}(x_0)]\,[\mathbb{T}(x_0+a)]]\,. \tag{3.8}$$

The inner square brackets here refer to the time-slices $[x_0,x_0+a]$ and $[x_0+a,x_0+2a]$, respectively, while the time interval for the outer bracket is $[x_0,x_0+2a]$. A formula for the Wilson loop expectation value involving three levels of averaging, for example, is given by (for t/a even)

$$\langle W(\mathcal{C})\rangle = \left\langle \mathbb{L}(0)_{\alpha\gamma}\,\{[[\mathbb{T}(0)]\,[\mathbb{T}(a)]]\cdots[[\mathbb{T}(t-2a)]\,[\mathbb{T}(t-a)]]\}_{\alpha\beta\gamma\delta}\,\mathbb{L}(t)^{*}_{\beta\delta}\right\rangle . \qquad (3.9)$$

In general we may choose an arbitrary hierarchy of time-slices of increasing thickness that fit within one another and with the time extent of the loop. The pattern does not even need to be regular, although this will often be the case in practice.

3.4 Centre symmetry & quark confinement

On any time-slice $[x_0, y_0]$, the sublattice theory has a global $\mathbb{Z}_3$ symmetry that acts on the time-like link variables according to

$$U(x,0) \to U(x,0)z\,, \qquad z = \mathrm{e}^{i2\pi k/3}\,, \qquad k = 0,1,2\,. \qquad (3.10)$$

The symmetry situation is thus similar to the one in finite-temperature lattice gauge theory. In particular, for fixed time-slice thickness and small values of the gauge coupling, the $\mathbb{Z}_3$ symmetry is probably spontaneously broken. The time-slices should otherwise be in the confinement phase and we then expect that

$$\|[\mathbb{T}(x_0)\cdots\mathbb{T}(y_0-a)]\| \propto \mathrm{e}^{-m_0 r} \qquad (3.11)$$

at large distances r, where $\|\cdots\|$ denotes the usual operator norm for 9×9 matrices acting on complex vectors. This is surely so in the strong coupling regime, and the picture is also supported, at least qualitatively, by our present (limited) numerical experience.

Combining eq. (3.11) with the hierarchical integration formulae derived above, it follows that the Wilson loop expectation value satisfies an area law with string tension $\sigma \geq m_0/(y_0-x_0)$. The decay properties of the time-slice expectation value (3.6) are thus directly linked to the issue of quark confinement.

4. Multilevel simulation

In the context of numerical simulations, the hierarchical integration formulae derived above do not seem to be particularly useful, because the time-slice averages cannot in general be calculated exactly or so precisely that the errors are surely negligible. However, as in the case of the multihit method [1], the sublattice averages may be estimated stochastically without compromising the correctness of the simulation. Our aim in this section is to work this out in some detail and to explain why the resulting multilevel algorithm may be expected to be highly efficient for large loops.

JHEP09(2001)010

4.1 Update cycle

Let us consider, as a simple example, the Polyakov loop correlation function on a lattice with an even number of points in the time direction. A hierarchical integration formula that might be used in this case is given by

$$\langle P(x)^* P(x + r\hat{1})\rangle = \Big\langle \{[\mathbb{T}(0)\mathbb{T}(a)] \cdots [\mathbb{T}(T-2a)\mathbb{T}(T-a)]\}_{\alpha\alpha\gamma\gamma} \Big\rangle . \tag{4.1}$$

The associated multilevel simulation algorithm then proceeds along the following lines:

1. Generate a sequence of gauge field configurations using a mixture of heatbath and over-relaxation link updates as usual.
2. For a subsequence of configurations, estimate $[\mathbb{T}(x_0)\mathbb{T}(x_0+a)]$ by updating the gauge field in the interior of the time-slice $[x_0, x_0 + 2a]$ a number of times and by averaging $\mathbb{T}(x_0)\mathbb{T}(x_0 + a)$ over these configurations.
3. Compute the average of the trace of the product in eq. (4.1) using the stochastic estimates for $[\mathbb{T}(0)\mathbb{T}(a)], \ldots, [\mathbb{T}(T-2a)\mathbb{T}(T-a)]$ calculated in step 2.

In practice the simulation proceeds sequentially, and the second step may be integrated in the first by performing n_1 updates of the whole lattice, then n_2 updates of the time-slices, then n_1 full updates, and so on. The trace of the product of the time-slice expectation values $[\mathbb{T}(x_0)\mathbb{T}(x_0 + a)]$ is calculated in each of these cycles, and the Polyakov loop correlation function is finally obtained by computing the average of these values over a significant number n_0 of cycles.

It should be emphasized that this algorithm produces exact results, up to statistical errors of order $(n_0)^{-1/2}$, for any choice of $n_1 \geq 1$ and $n_2 \geq 1$. In this respect the quality of the stochastic estimation of the time-slice averages is therefore irrelevant, but as we shall see shortly the efficiency of the simulation strongly depends on it. A formal proof of the exactness of the algorithm is given in appendix A.

4.2 Exponential error reduction

We now show that the two-level algorithm described above leads to an exponential reduction of the statistical errors if the time-slices of thickness $2a$ are in the confinement phase, where the expectation values $[\mathbb{T}(x_0)\mathbb{T}(x_0 + a)]$ decay exponentially at large r.

The stochastic estimates of the latter, which are obtained in each cycle of the algorithm, are accurate to within a statistical error proportional to $(n_2)^{-1/2}$. We may, for example, fix n_2 so that the signal-to-noise ratio is approximately equal to unity at the specified value of r, which requires n_2 to be scaled according to

$$n_2 \propto \mathrm{e}^{2m_0 r} \tag{4.2}$$

(cf. eq. (3.11)). The factors in the product (4.1) are then of order $e^{-m_0 r}$, so that the magnitude of the trace of the product calculated in each cycle is roughly proportional to $e^{-m_0 r T/2a}$. In particular, the statistical fluctuations are reduced to this level.

As a result we expect that the algorithm achieves an exponential error reduction, with exponent proportional to the area $A = rT$, for a computational effort growing as suggested by eq. (4.2). It goes without saying that this analysis ignores many details and can only give a first indication of what the true behaviour of the algorithm is going to be.

4.3 Higher-order schemes

It should be quite clear at this point that any given hierarchical integration formula, with possibly many levels of nested time-slice expectation values, corresponds to a multilevel simulation algorithm. At each level the associated time-slice expectation values are estimated stochastically, and these estimates are then used in the averages taken at the next higher level. The algorithm thus follows a cycle during which the thin time-slices are updated more often than the thicker ones.

Further details on the algorithm and how to program it can be found in appendix B. Here we only note that at the lowest level, where the products of the basic link variables are averaged, the multihit method [1] (perhaps also in its analytic version [10]) may be used to achieve a further reduction of the statistical errors.

5. Test of the method

5.1 Lattice size and choice of parameters

For this first test of the multilevel simulation algorithm, we decided to calculate the Polyakov loop correlation function at gauge coupling $\beta \equiv 6/g_0^2 = 5.7$, where the Sommer reference scale r_0 is about $2.92a$ [11, 12]. This implies $a \simeq 0.17\,\mathrm{fm}$, and a lattice of spatial size $L = 12a$ is thus approximately $2\,\mathrm{fm}$ wide.

Some experimenting reveals that the time-slices of thickness $2a$ appear to be in the confinement phase, for this gauge coupling and lattice size, and if $T \geq 6a$. The two-level simulation algorithm discussed in the previous section is hence expected to perform well. As it turns out, for the larger values of r and T, an even greater algorithmic efficiency is achieved with a further level of averaging. The corresponding hierarchical integration formula reads

$$\langle P(x)^* P(x + r\hat{1})\rangle = \Big\langle \{[[\mathbb{T}(0)\mathbb{T}(a)]\,[\mathbb{T}(2a)\mathbb{T}(3a)]]\cdots[\cdots[\mathbb{T}(T-2a)\mathbb{T}(T-a)]]\}_{\alpha\alpha\gamma\gamma} \Big\rangle, \tag{5.1}$$

where T/a is assumed to be a multiple of 4.

Multilevel algorithms have many parameters and it is not easy to find their optimal values. The most important parameters are the numbers of time-slice updates

T/a	r/a	$\langle P^* P \rangle$	T/a	r/a	$\langle P^* P \rangle$
12	2	$6.46(2) \times 10^{-5}$	6	6	$2.48(2) \times 10^{-4}$
12	3	$4.93(3) \times 10^{-6}$	8	6	$9.53(13) \times 10^{-6}$
12	4	$5.42(5) \times 10^{-7}$	12	6	$1.90(4) \times 10^{-8}$
12	5	$7.06(9) \times 10^{-8}$	16	6	$3.91(7) \times 10^{-11}$
			20	6	$8.23(16) \times 10^{-14}$

Table 1: Results for the Polyakov loop correlation function at $\beta = 5.7$, $L = 12a$.

that are to be performed at each level. Using an optimization strategy discussed in appendix B, we found that a good efficiency at distance $r = 6a$ is achieved with 96 update sweeps at the lowest level (time-slices of thickness $2a$) and 10 sweeps at the next-to-lowest level (thickness $4a$). In addition, the application of the multihit method at the lowest level proved to be beneficial. The ratio of heatbath to over-relaxation link updates is not critical and was set to $1:5$. At smaller distances r, it is generally better to reduce the numbers of time-slice updates, which reflects the fact that it does not pay to determine the time-slice averages accurately.

5.2 Simulation results

The results of our calculations (table 1 and figures 2 and 3) show rather strikingly that the Polyakov loop correlation function is obtained for a large range of loop sizes and distances with statistical errors that decrease exponentially. With these algorithms the *relative* errors can effectively be kept fixed even if the correlation function is very rapidly decaying.

In total the simulations that we have done required the equivalent of about 1000 hours of processor time on a PC with 1.4 GHz Pentium 4 processor. With a program that makes use of the multihit method only, and with any currently available super-computer, it would be quite impossible to reproduce the data listed in table 1. At $r = 6a$ and $T = 12a$, for example, the multilevel algorithm achieves an efficiency (in terms of the computer time required for a specified error on a given machine) that is better by a factor of 3×10^5 or so, and this factor rises to astronomical values at the larger values of T.

Our tests have also shown that the multilevel algorithm is well behaved when the time extent of the lattice increases. For fixed r and a given number of "measurements" of the loop correlation function, the relative errors appear to be growing roughly linearly with T. In other words, the required computer time scales approximately like T^3 if the relative errors are to be kept fixed. Eventually this becomes a big factor, but the improvement with respect to the exponential decrease of the signal-to-noise ratio that is characteristic of the simulation algorithms used to date is obvious.

JHEP09(2001)010

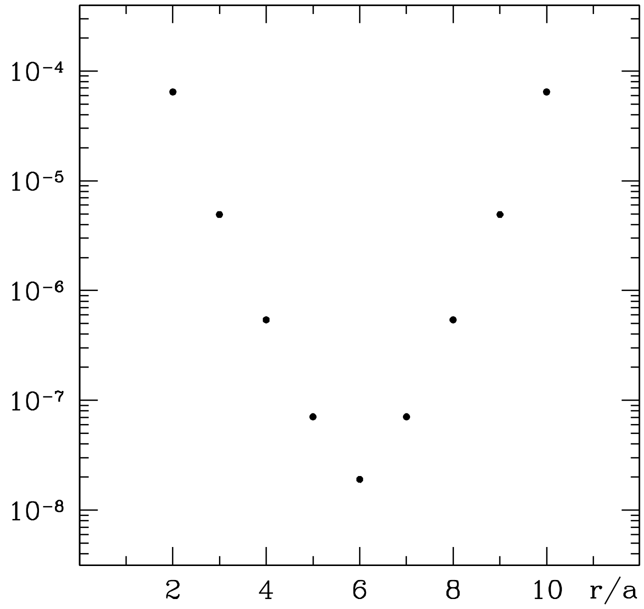

Figure 2: Polyakov loop correlation function at distance r on a 12^4 lattice at $\beta = 5.7$. The loops on this lattice are about 2 fm long and enclose an area $A = rT$ ranging from 0.7 to 2.1 fm^2. Statistical errors are smaller than the dots.

6. Conclusions

At the gauge coupling that we have considered, the multilevel algorithm proposed in this paper performs exceedingly well, and there is little doubt that a similarly impressive error reduction will be achieved also at other couplings. Using this technique, it is possible to compute loop correlation functions and Wilson loop expectation values in a range of loop sizes and distances that had remained inaccessible so far. In particular, one of the physics issues that might be reconsidered at this point is the question of whether long colour flux tubes are described by an effective bosonic string theory [13, 14] or perhaps a different kind of string theory (or none at all).

In some cases it may be useful to combine the multilevel algorithm with link-blocking techniques [2, 3] and the variational method [4, 5]. The calculation of the energy spectrum of excited colour flux tubes is an example for this. String breaking, on the other hand, is perhaps better approached by considering correlation functions of long Polyakov loops, as we did in this paper, since their exponential decay at large time extents T is guaranteed to yield the true ground-state energy in the chosen charge sector (i.e. this is an unbiased method).

We finally mention that the basic ideas underlying the algorithm (sublattice averages and hierarchical averaging) are likely to be more widely applicable. An important case to consider are the correlation functions of two or more local operators, which are another instance where the signal-to-noise ratio is exponentially decreasing if the established simulation techniques are employed.

JHEP09(2001)010

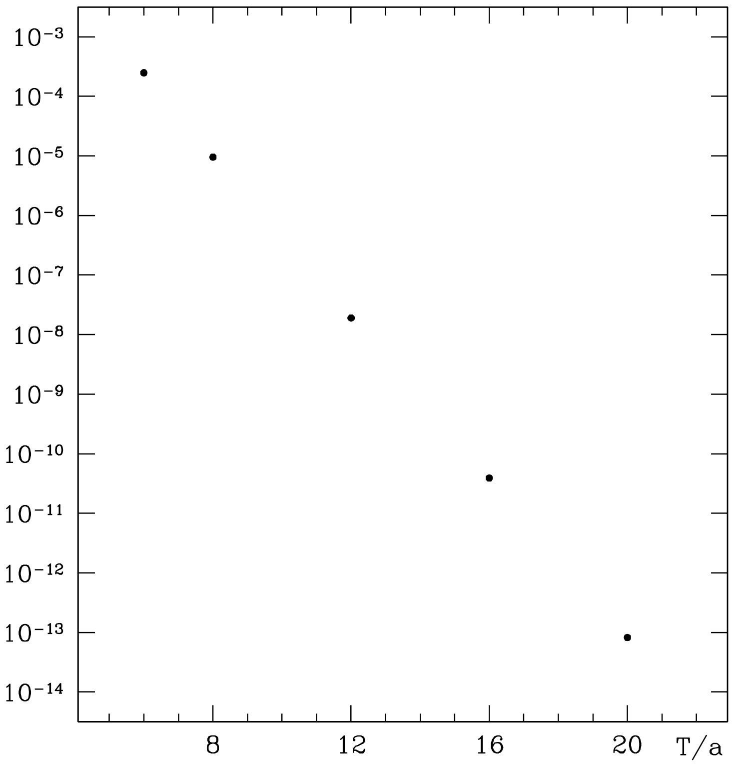

Figure 3: Polyakov loop correlation function at $\beta = 5.7$ and distance $r = 6a \simeq 1.0\,\mathrm{fm}$, on a lattice of spatial size $L = 12a$ and variable time extent T. Statistical errors are not visible on this scale. On the largest lattice, $T = 20a$, the loops are $3.4\,\mathrm{fm}$ long.

Acknowledgments

P.W. sincerely thanks the Center for Computational Physics, Tsukuba, for hospitality during the time that this work was completed. All computations reported in this paper have been done at the computer centres of DESY at Hamburg and Zeuthen. We would like to thank the staff of these centres for their support.

A. Theoretical discussion of the multilevel algorithm

In this appendix we establish the correctness of the two-level simulation algorithm defined in section 4. The principal question is whether the replacement of the two-link operators by their averages over certain subsets of configurations is permissible.

A.1 Abstract model

To be able to bring out the essence of the argument more clearly, an abstract system will be considered with a finite number of states s. Each state is characterized by a vector $(s_0, s_1, \ldots, s_n)$ of discrete variables, whose joint probability distribution is of

JHEP09(2001)010

the factorized form

$$p(s) = p_0(s_0) \prod_{k=1}^{n} p_k(s_0, s_k)\,, \tag{A.1}$$

$$\sum_{s_0} p_0(s_0) = \sum_{s_k} p_k(s_0, s_k) = 1\,. \tag{A.2}$$

We are then interested in calculating expectation values of factorized observables

$$\mathcal{O}(s) = \mathcal{O}_0(s_0) \prod_{k=1}^{n} \mathcal{O}_k(s_0, s_k)\,. \tag{A.3}$$

Evidently this model fits the case of interest if we identify s_0 with the space-like link variables at all even times, s_1 with the link variables in the interior of the time-slice $[0, 2a]$, s_2 with those in $[2a, 4a]$, and so on.

A.2 Two-level simulation

We now suppose that the system is simulated by updating the variables $s_0, s_1, \ldots, s_n$ one by one in an arbitrary order. The update algorithm is assumed to be such that the transition probability for changing s_k at fixed s_0 is independent of the current values of all other variables. This implies, in particular, that the algorithm simulates, and thus preserves, the conditional probability distribution $p_k(s_0, s_k)$.

If we perform the simulation in cycles, where, in each cycle, n_1 updates of all variables are followed by n_2 updates of $s_1, \ldots, s_n$ only, it is clear that the subsequence of configurations that are obtained during the n_1 full updates simulates the probability distribution $p(s)$. The complete sequence of configurations does that too, but it is useful to look at the n_2 intermediate configurations in a different way. Since s_0 is fixed in this part of each cycle, the algorithm effectively generates n_2 configurations of each s_k separately, so that in total there are $(n_2)^n$ configurations of the vector $(s_1, \ldots, s_n)$. In practice these are not stored in the memory of the computer at any time, but for the theoretical discussion we may assume this to be the case.

A.3 Factorization lemma

As we just remarked, the simulation generates $(n_2)^n$ configurations of the vector $(s_1, \ldots, s_n)$ in each cycle, and we now focus on the sequence of these sets of states.

Lemma. *The vectors $(s_1, \ldots, s_n)$ in the sets with the same value of s_0 occur with conditional probability $p_1(s_0, s_1) \ldots p_n(s_0, s_n)$.*

To prove this, we first note that the states s at the end of the n_1 full updates in each cycle are distributed with probability $p(s)$. In particular, the states in this sequence with the same value of s_0 are distributed with probability $p_1(s_0, s_1) \ldots p_n(s_0, s_n)$.

JHEP09(2001)010

When the algorithm arrives at any of these points, it continues to generate the next set of $(n_2)^n$ vectors with a transition probability that preserves the conditional probabilities of the variables s_k. Since the initial values of these variables are already properly distributed, and since they are statistically independent, the sequence of the sets of $(n_2)^n$ additional states with the specified value of s_0 must be distributed in the same way.

A.4 Expectation values

The expectation value of the factorized observable (A.3) can be written in the form

$$\langle \mathcal{O} \rangle = \sum_{s_0} p_0(s_0) \mathcal{O}_0(s_0) \prod_{k=1}^{n} [\mathcal{O}_k](s_0) \,, \tag{A.4}$$

$$[\mathcal{O}_k](s_0) = \sum_{s_k} p_k(s_0, s_k) \mathcal{O}_k(s_0, s_k) \,. \tag{A.5}$$

From the lemma it now follows that the product

$$\prod_{k=1}^{n} [\mathcal{O}_k](s_0) \tag{A.6}$$

is equal to the average over all sets of $(n_2)^n$ configurations with the specified value of s_0 that occur in the course of the simulation. Moreover the average over the $(n_2)^n$ configurations in any one of these sets is trivially given by the product of the averages of the factors $\mathcal{O}_k(s_0, s_k)$. In particular, there is no need to store any configurations other than the current one, since the averages of the factors can be computed sequentially.

We have thus shown that the expectation value $\langle \mathcal{O} \rangle$ may be obtained by substituting stochastic estimates for the factors $[\mathcal{O}_k](s_0)$ on the right-hand side of eq. (A.4) and by averaging the so calculated product over the sequence of values of s_0. For each k the estimate of $[\mathcal{O}_k](s_0)$ is the average over the n_2 configurations of s_k that are generated in the second part of each update cycle.

B. Program description

The implementation of the multilevel algorithm requires some care in order to avoid excessive memory usage and arithmetic inefficiencies. In the following paragraphs we discuss the structure of our program in outline and address some of the key points that should be taken into account.

B.1 Basic update algorithm

We use the now standard "hybrid over-relaxation" simulation algorithm that combines heatbath with over-relaxation link updates in an adjustable proportion (for a review, see ref. [17], for example). In both cases a link update involves three Cabibbo-Marinari rotations [18] in the obvious SU(2) subgroups of SU(3). Depending on the driving force exerted by the surrounding link variables, the heatbath algorithm of Creutz [19] or the one of Fabricius-Haan [20] and Kennedy-Pendleton [21] is applied. In this way a high efficiency is achieved in all situations.

It is well-known that the choice of the random number generator can introduce a systematic bias in numerical simulations. To be on the safe side we use the `ranlux` generator [15, 16] even though it consumes a significant fraction of the update time.

B.2 Program structure

The multilevel simulation algorithm cycles through several levels that correspond to time-slices of increasing thickness. At the lowest level the product $\mathbb{T}(x_0)\cdots\mathbb{T}(y_0-a)$ is averaged over a set of configurations on the time-slice $[x_0, y_0]$. The result of this calculation is then passed to the next level, where the contributions from two or more lowest-level time-slices are multiplied and averaged. From here on the procedure repeats itself until the top level is reached, at which point the product of the nested averages of the two-link operators is calculated and its trace is taken.

This algorithm thus has a tree-like structure, where at each level the following parameters need to be specified:

$d_{\rm ts}$: thickness of the associated time-slice

$n_{\rm ms}$: number of "measurements" to be made for the time-slice average

$n_{\rm up}$: number of time-slice updates between "measurements"

$n_{\rm hb}$, $n_{\rm or}$: numbers of heatbath and over-relaxation sweeps per time-slice update

$p_{\rm ws}$: pointer to a memory area that may be used as workspace at this level

In the program the full set of parameters is globally visible so that the calculated time-slice expectation values (which reside in the workspace of the associated level) can easily be accessed when the algorithm has moved to the next higher level.

The level structure can be elegantly programmed by defining a recursive function that takes the level number and the time-slice initial time x_0 as arguments and calculates the corresponding (nested) time-slice average. Exactly what is to be done can be inferred from the globally visible parameters that describe the hierarchy of the time-slices. Internally the program calls itself until the lowest level is reached, and the tree-structure of the algorithm is thus mapped to the call sequence generated by the program.

JHEP09(2001)010

B.3 Rounding errors

In the range of loop sizes and distances that we have considered, the Polyakov loop correlation function decays over many orders of magnitude. One might suspect that significance losses become an issue when the calculated values approach the machine precision. This is not the case, however, because the correlation function is obtained by averaging a product of factors of about equal magnitude (that are themselves averages of products of still larger factors, etc.). In other words, the small numbers do not result from an enormous cancellation but by multiplication of many factors.

Some significance losses may still occur when the time-slice averages are computed. This problem (if present) can be avoided by performing all operations involving two-link operators, their products and averages in 64-bit floating-point arithmetic. On the other hand, there is no reason not to use single-precision data and arithmetic for the link variables and the basic update algorithm.

B.4 Memory requirements

Since the update cycles of the multilevel algorithm are time-consuming, it is economical to calculate the Polyakov loop correlation function simultaneously at all points $x = (0, x_1, x_2, x_3)$ and displacement vectors $r\hat{k}$, $k = 1, 2, 3$. At each level the workspace must then be sufficiently large to contain this many two-link operators. One actually needs twice this space for the calculation of the averages at all x.

Two-link operators have 162 real components and thus occupy 1296 bytes of storage if double-precision arithmetic is used. The total memory space required per level is hence $7.6\,\mathrm{KB} \times (L/a)^3$ times the number of distances r at which the correlation function is to be calculated. This quickly adds up to a large number, but it should be taken into account that there is not much to be gained by processing many values of r at the same time, because the optimal choice of the level parameters depends on r. Note that the same memory area may be used for all time-slices at a given level since these are visited sequentially.

B.5 Operation count and timing

The tensor product (3.2) and the product (3.3) of two two-link operators require 486 and 5670 floating-point operations respectively. These numbers are large but not out of proportion, considering the fact that 1926 operations are required for the calculation of the "staples" in the link update programs. In our test runs, for example, the total time spent to manipulate the two-link operators was comparable to the time needed to update the gauge field.

At the lowest level, multiplications of two-link operators should be avoided by first calculating the products $U(x_0, 0) \cdots U(y_0 - a, 0)$ and then the tensor products of these. If the multihit method is used, the link variables $U(z_0, 0)$ are averaged over a number of heatbath link updates before they are inserted in the products.

JHEP09(2001)010

The simulations reported in section 5 have been performed on an 8-node cluster with 550 MHz Pentium III processors and on a stand-alone PC with 1.4 GHz Pentium 4 processor. Using vector arithmetic (SSE instructions), the processor time required for a heatbath (over-relaxation) link update on this PC is 3.4 μs (2.0 μs). The timing of the multilevel algorithm is more difficult, but a rough estimate of the execution time per cycle may be obtained by adding the link update times and multiplying this number by 2. On the 16×12^3 lattice, for example, 100 "measurements" of the loop correlation function at distance $r = 6a$, with cycle parameters as quoted below, require about 50 hours of PC processor time.

B.6 Parameter tuning

It is our experience that the parameters of the multilevel algorithm are best determined by fixing them at the lowest level, then at the next-to-lowest level, and so on. Since the multihit method leads to a further reduction of the statistical errors, the first step is to optimize this part of the algorithm. The thickness $d_{\rm ts}$ of the time-slice at the lowest level must then be determined. As discussed in subsections 3.4 and 4.2, $d_{\rm ts}$ should be as small as possible, but sufficiently large that the time-slice is in the confinement phase.

The other parameters listed in subsection B.2 are fixed essentially by minimizing the average over all points x and directions k of the *absolute value* $|P^*P|$ of the trace of the product of the time-slice expectation values calculated at the lowest level, for a single thermalized gauge field configuration. More precisely the average $\langle|P^*P|\rangle$ should be balanced against the processor time τ required to compute it so that $\tau \times \langle|P^*P|\rangle^2$ is minimized (which yields the maximal error reduction for a given amount of computer time).

At $\beta = 5.7$, $r = 6a$, $L = 12a$ and all $T \geq 8a$, the lowest-level parameter list that we obtained in this way is $(d_{\rm ts}, n_{\rm ms}, n_{\rm up}, n_{\rm hb}, n_{\rm or}) = (2a, 96, 1, 1, 5)$. The optimum is rather flat and variations of $n_{\rm ms}$ by 10% or so make practically no difference. At the next level the same optimization procedure suggests to take $(d_{\rm ts}, n_{\rm ms}, n_{\rm up}, n_{\rm hb}, n_{\rm or}) = (4a, 10, 16, 1, 5)$. The additional error reduction that is achieved at this level is very substantial, but having a third level with $d_{\rm ts} = 8a$ seems to have no positive effect.

We finally note that the auto-correlations between successive "measurements" of the loop correlation function can be practically reduced to zero by updating the full lattice a significant number of times before every "measurement". This adds an only small overhead to the total execution time, which is dominated by the time required for the computation of the time-slice averages.

References

[1] G. Parisi, R. Petronzio and F. Rapuano, *A measurement of the string tension near the continuum limit*, *Phys. Lett.* **B 128** (1983) 418.

[2] M. Teper, *An improved method for lattice glueball calculations*, *Phys. Lett.* **B 183** (1987) 345.

[3] APE collaboration, M. Albanese et al., *Glueball masses and string tension in lattice QCD*, *Phys. Lett.* **B 192** (1987) 163.

[4] N.A. Campbell, A. Huntley and C. Michael, *Heavy quark potentials and hybrid mesons from* SU(3) *lattice gauge theory*, *Nucl. Phys.* **B 306** (1988) 51.

[5] M. Lüscher and U. Wolff, *How to calculate the elastic scattering matrix in two-dimensional quantum field theories by numerical simulation*, *Nucl. Phys.* **B 339** (1990) 222.

[6] O. Philipsen and H. Wittig, *String breaking in non-abelian gauge theories with fundamental matter fields*, *Phys. Rev. Lett.* **81** (1998) 4056 [hep-lat/9807020], erratum *ibid.* **83** (1999) 2684; *String breaking in* SU(2) *Yang-Mills theory with adjoint sources*, *Phys. Lett.* **B 451** (1999) 146 [hep-lat/9902003].

[7] ALPHA collaboration, F. Knechtli and R. Sommer, *String breaking in* SU(2) *gauge theory with scalar matter fields*, *Phys. Lett.* **B 440** (1998) 345 [hep-lat/9807022], erratum *ibid.* **B454** (1999) 399; *String breaking as a mixing phenomenon in the* SU(2) *Higgs model*, *Nucl. Phys.* **B 590** (2000) 309 [hep-lat/0005021].

[8] Ph. de Forcrand and O. Philipsen, *Adjoint string breaking in 4d* SU(2) *Yang-Mills theory*, *Phys. Lett.* **B 475** (2000) 280 [hep-lat/9912050].

[9] K.G. Wilson, *Confinement of quarks*, *Phys. Rev.* **D 10** (1974) 2445.

[10] Ph. de Forcrand and C. Roiesnel, *Refined methods for measuring large distance correlations*, *Phys. Lett.* **B 151** (1985) 77.

[11] R. Sommer, *A new way to set the energy scale in lattice gauge theories and its applications to the static force and α_s in* SU(2) *Yang-Mills theory*, *Nucl. Phys.* **B 411** (1994) 839 [hep-lat/9310022].

[12] ALPHA collaboration, M. Guagnelli, R. Sommer and H. Wittig, *Precision computation of a low-energy reference scale in quenched lattice QCD*, *Nucl. Phys.* **B 535** (1998) 389 [hep-lat/9806005].

[13] M. Lüscher, K. Symanzik and P. Weisz, *Anomalies of the free loop wave equation in the WKB approximation*, *Nucl. Phys.* **B 173** (1980) 365.

[14] M. Lüscher, *Symmetry breaking aspects of the roughening transition in gauge theories*, *Nucl. Phys.* **B 180** (1981) 317.

[15] M. Lüscher, *A portable high-quality random number generator for lattice field theory simulations*, *Comput. Phys. Commun.* **79** (1994) 100 [hep-lat/9309020].

[16] F. James, *RANLUX: a Fortran implementation of the high-quality pseudorandom number generator of Lüscher*, *Comput. Phys. Commun.* **79** (1994) 111, erratum *ibid.* **97** (1996) 357.

[17] A.D. Kennedy, *Progress in lattice field theory algorithms*, *Nucl. Phys.* **30** *(Proc. Suppl.)* (1993) 96 [hep-lat/9212017].

[18] N. Cabibbo and E. Marinari, *A new method for updating* SU(N) *matrices in computer simulations of gauge theories*, *Phys. Lett.* **B 119** (1982) 387.

[19] M. Creutz, *Monte Carlo study of quantized* SU(2) *gauge theory*, *Phys. Rev.* **D 21** (1980) 2308.

[20] K. Fabricius and O. Haan, *Heat bath method for the twisted Eguchi-Kawai model*, *Phys. Lett.* **B 143** (1984) 459.

[21] A.D. Kennedy and B.J. Pendleton, *Improved heat bath method for Monte Carlo calculations in lattice gauge theories*, *Phys. Lett.* **B 156** (1985) 393.