PHYSICAL REVIEW D VOLUME 57, NUMBER 7 1 APRIL 1998

Multigrid implementation of the Fourier acceleration method for Landau gauge fixing

Attilio Cucchieri* and Tereza Mendes†
Gruppo APE, Dipartimento di Fisica, Università di Roma ''Tor Vergata'', and INFN, Sezione di Roma 2, Via della Ricerca Scientifica 1, 00133 Roma, Italy
(Received 17 December 1997; published 10 March 1998)

We present a new implementation of the Fourier acceleration method for Landau gauge fixing. By means of a multigrid inversion we are able to avoid the use of the fast Fourier transform. This makes the method more flexible, and well suited for vector and parallel machines. We study the performance of this algorithm on serial and on parallel (APE100) machines for the four-dimensional SU(2) case. We find that our method is equivalent to the standard implementation of Fourier acceleration already on a serial machine, and that it parallelizes very efficiently: its computational cost shows a linear speedup with the number of processors. We have also implemented, on the parallel machines, a version of the method using conjugate gradient instead of multigrid. This leads to an algorithm that is efficient at intermediate lattice volumes. [S0556-2821(98)50607-8]

PACS number(s): 11.15.Ha, 02.60.Pn

I. INTRODUCTION

Lattice gauge fixing is a necessary tool for understanding the relationship between continuum and lattice gauge theory. In fact, because of asymptotic freedom, the continuum limit of the lattice theory is the weak-coupling limit, and a weak-coupling expansion requires gauge fixing. Gauge fixing is also used in smearing techniques, and is necessary in order to evaluate quark or gluon matrix elements which can be used to extract nonperturbative results from Monte Carlo simulations [1]. It is therefore important to devise numerical algorithms to gauge fix efficiently a lattice configuration. An important issue regarding the efficiency of these algorithms is the problem of *critical slowing down* (CSD), which occurs when the relaxation time τ of an algorithm diverges as the lattice volume is increased [2]. Conventional local algorithms have a *dynamic critical exponent* $z \approx 2$, namely τ grows with the lattice side N roughly as N^2. Improved local methods show typically $z \approx 1$, while global methods may succeed in eliminating CSD completely, i.e., $z \approx 0$. Usually, global algorithms are more costly per iteration than local methods but, due to the elimination of CSD, their total computational cost becomes progressively lower than that of local methods at large lattice volumes. For this reason, efficient global algorithms are a highly desirable tool in large-volume applications. Another important issue is whether gauge-fixing algorithms can be implemented efficiently on parallel machines [3], since computers of this type are widely used in numerical studies of gauge theory on large lattices.

A well-known global approach for reducing CSD, applicable to gauge fixing as well as to other problems, is the method of *Fourier acceleration* (FA) [4]. The idea is to precondition a problem using a diagonal matrix in momentum space that is related to the solution of a simplified version of the problem [4,5]. For the SU(2) Landau gauge fixing case, it can be proved [6] that Fourier acceleration eliminates CSD completely at infinite β, namely the dynamic critical exponent z is equal to zero; we have also obtained [7], at finite β and in two dimensions, that z is equal to 0.036 ± 0.064. Moreover, the FA gauge-fixing method is very efficient in achieving a constant value for the longitudinal gluon propagator at zero three-momentum [6–8], which provides a very sensitive test of the goodness of the gauge fixing.

To fix Landau gauge on the lattice, one looks for a minimum of the functional [9]:

$$\mathcal{E}_U[g] \equiv \frac{1}{4V} \sum_{\mu,x} \mathrm{Tr}[\mathbb{1} - g(x) U_\mu(x) g^\dagger(x+e_\mu)] \,. \qquad (1)$$

(We refer to [7] for notation.) The FA update is given by $g^{(\mathrm{new})}(x) \equiv R(x) g^{(\mathrm{old})}(x)$ with

$$R(x) \propto \mathbb{1} - \hat{F}^{-1} \left(\frac{\alpha}{p^2(k)} \hat{F}(\nabla \cdot A^{(g)}) \right)(x). \qquad (2)$$

Here $\hat{F}$ is a Fourier transform, α is a tuning parameter, $p^2(k)$ is the square of the lattice momentum, and $\nabla \cdot A$ is the lattice divergence of the gluon field A_μ. Thus, in this case, the preconditioning is obtained using in momentum space a diagonal matrix with elements given by $1/p^2(k)$ (see Ref. [4], Sec. II, for details). The FA method in its original form makes use of the *fast Fourier transform* (FFT) to evaluate $\hat{F}$ and $\hat{F}^{-1}$, which requires a work of only $V \log V$, where V is the lattice volume [4], making it very appealing from the computational point of view. On the other hand, in order to implement FFT efficiently, one is restricted to using lattice sides that are powers of 2 ([10], Chapter 12). Moreover, implementing FFT on parallel machines of the SIMD type, and especially in four dimensions, can be very cumbersome [3,11]. Here we present a new implementation of the FA method for Landau gauge fixing, which avoids completely the use of the fast Fourier transform, and we test its performance for the four-dimensional SU(2) case on serial and on parallel machines. Preliminary results have been reported in [12].

*Email address: cucchieri@roma2.infn.it
†Email address: mendes@roma2.infn.it

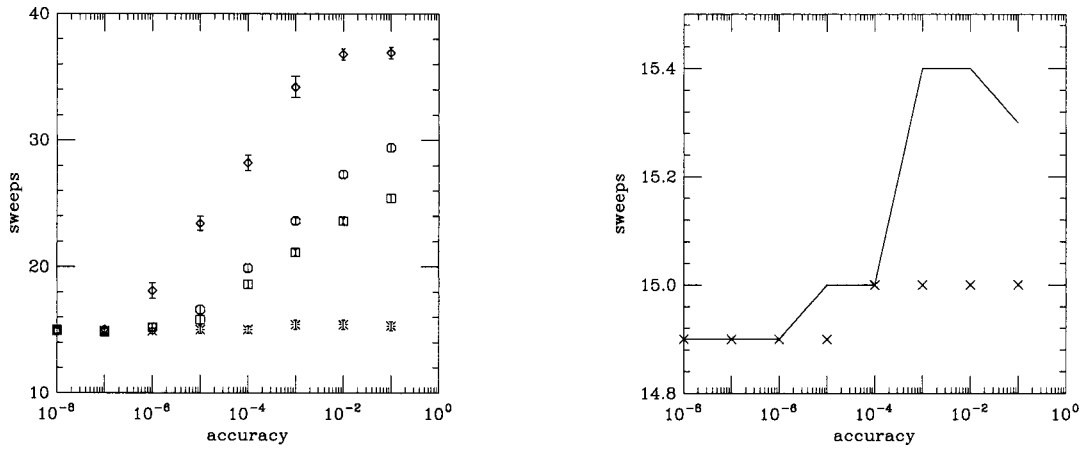

FIG. 1. Plot of the number of gauge-fixing sweeps for the MGFA algorithm as a function of the accuracy imposed on the inversion of the Laplacian. Simulations were carried out on a workstation, for lattice volume 8^4 and at $\beta=\infty$. The different types of multigrid cycle employed are: (left-hand figure) Gauss-Seidel update ($\diamond$), V-cycle with $N_r=1, N_{\min}=1$ ($\bigcirc$), W-cycle with $N_r=1, N_{\min}=1$ ($\square$), W-cycle with $N_r=2, N_{\min}=1$ (*); (right-hand figure) a closer view of the W-cycle with $N_r=2, N_{\min}=1$ case (solid line), together with the W-cycle with $N_r=2, N_{\min}=2$ case ($\times$). In the second plot errors are omitted for clarity. We note that for FFTFA the number of gauge-fixing sweeps is 14.9 ± 0.2. In all cases we set $\alpha=1.105$, and we stopped the gauge fixing when the condition $(\nabla\cdot A)^2\leqslant 10^{-10}$ was satisfied.

II. THE MULTIGRID FA METHOD

Let us start by noting that the Fourier-mode decomposition in Eq. (2) is equivalent to an inversion of the lattice Laplacian operator Δ:

$$\hat{F}^{-1}p^{-2}(k)\hat{F}=(-\Delta)^{-1}\,. \tag{3}$$

(Note that this inversion is carried out for each component of $\nabla\cdot A$.) Thus, the FFT can be avoided by inverting Δ using an alternative algorithm. Clearly, a good candidate should be suitable for use on parallel machines, and should require, ideally, the same computational work as FFT, i.e., $V\log V$. One such candidate is the *multigrid* (MG) algorithm with W-cycle and piecewise-constant interpolation. The idea of MG is to solve the lattice problem recursively, using local (Gauss-Seidel) updates on coarser versions of the original lattice in order to accelerate the convergence of slow modes of the solution. The MG is an efficient *iterative* routine for inverting the Laplacian Δ: with our choice of cycle and interpolation, each iteration of the method represents a work of order V, and the number of iterations required for convergence is proportional to $\log V$ at most [13,14]. Moreover, the MG routine can be successfully implemented on vector and parallel machines [15]. Thus this approach should preserve the property of eliminating CSD for Landau gauge fixing, while being applicable on parallel machines. At the same time, there is no restriction on the lattice size since, even for a fixed coarsening factor (e.g., 2), the size of the coarsest grid can be adjusted conveniently.

The overhead for the MG routine is likely to be larger than the one for FFT, but in our case it can be reduced by exploiting the fact that multigrid (as opposed to FFT) is an iterative method. For example, a significant computational gain can be obtained if one starts the inversion from a good initial guess for the solution. Also, by changing the stopping criterion for the inversion, the accuracy of the solution can be suitably varied, while for FFT the accuracy is fixed by the precision used in the numerical code. This is important since we will be tuning the parameter α in Eq. (2), and this tuning can be done only up to an accuracy of a few percent. Thus, the inversion of Δ most likely will not require the high accuracy obtained in the FFT case, making possible a substantial reduction of the computational cost.

In order to test the feasibility of this approach, we have started our simulations on a workstation, comparing the performance of the MG implementation of the Fourier acceleration method (MGFA) with the original implementation using FFT (which we call from now on FFTFA) [16]. As a first step, we tuned the parameter α for the FFTFA method at infinite β and $V=8^4$. We obtained $\alpha_{\rm opt}=1.105$ as optimal choice, and a number of gauge-fixing sweeps equal to $14.9(2)$. (Note that our data represent averages over a set of gauge configurations, and that our error bars are one standard deviation.) We then tested MGFA with $\alpha=\alpha_{\rm opt}$: in addition to the W-cycle (where each grid is visited twice before proceeding to the next finer grid), we also tested for comparison the V-cycle (where each grid is visited once before proceeding to the next finer grid) and the standard Gauss-Seidel update; for the W-cycle we also varied the number of Gauss-Seidel relaxation sweeps on each grid (N_r), and the minimum number of complete multigrid cycles ($N_{\min}$). Results for the number of gauge-fixing sweeps as a function of the accuracy [17], at infinite β and $V=8^4$, are reported in Fig. 1. By comparing these results with the result from the FFTFA method, it is clear that the number of gauge-fixing sweeps is independent of the method used for the inversion of Δ, provided that a high enough accuracy is required. Among the tested versions of the MGFA method, the best from the point of view of computational cost is the one with the following choice of parameters: $\gamma=2$, an accuracy of about 10^{-5}, two relaxation sweeps on each grid, and a minimum of two multigrid cycles for each inversion of Δ. (We note that the CPU cost (computer time) per iteration of this MG version is higher than for the other versions, but the fact that the inversion of the Laplacian can be stopped already at an accuracy of 10^{-5} makes it the fastest version.) *We then adopt this version of MG as the routine used for our MGFA method.* For the MG routine we have chosen to use a coarsest

TABLE I. Comparison between the FFTFA and MGFA algorithms at $\beta=\infty$ and at $\beta=2.2$ on a workstation. The optimal choice for the tuning parameter α for the two FA methods is reported [18]. We also quote here results for the overrelaxation (OVE) method.

Algorithm	β	V	$\alpha_{\rm opt}$	GF sweeps	CPU time (s)
OVE	∞	4^4		18.1 ± 0.1	0.167 ± 0.003
FFTFA	∞	4^4	1.12	13.6 ± 0.3	0.34 ± 0.01
MGFA	∞	4^4	1.12	13.7 ± 0.3	0.55 ± 0.02
OVE	∞	8^4		37.5 ± 0.2	5.91 ± 0.03
FFTFA	∞	8^4	1.12	16.1 ± 0.3	9.60 ± 0.20
MGFA	∞	8^4	1.12	16.2 ± 0.3	11.4 ± 0.5
OVE	∞	16^4		76.6 ± 0.9	194 ± 2
FFTFA	∞	16^4	1.1	20.0 ± 0.4	246 ± 6
MGFA	∞	16^4	1.1	20.3 ± 0.4	242 ± 10
OVE	2.2	8^4		139 ± 8	22.5 ± 1.4
FFTFA	2.2	8^4	1.89	333 ± 27	200 ± 12
MGFA	2.2	8^4	1.89	312 ± 25	187 ± 15

grid of 2^4. Nevertheless we have checked that the performance of the chosen version does not change if a coarsest grid of 4^4 is used. This is an important result, as we will see, for the implementation of the algorithm on parallel machines. We have also tested different initial guesses for the MG inversion, finding that the use of the solution to the previous inversion of the Laplacian is preferable to a null or random initial guess.

III. RESULTS AND CONCLUSIONS

Initially, we have tested the performance of the MGFA method at $\beta=\infty$ and at $\beta=2.2$ on a workstation. The results, reported in Table I, are compared with those obtained for the FFTFA method and for the overrelaxation (OVE) algorithm [19], an improved local method which shows $z\approx1$ [6,7] and which is often used for production runs in lattice gauge theory. In all cases the stopping criterion for the gauge fixing was $(\nabla\cdot A)^2\leqslant10^{-12}$. From the data at $\beta=\infty$ we can study the volume dependence of the computational cost for the various algorithms. Clearly, FFTFA and MGFA have a similar performance [20], showing a number of gauge-fixing sweeps increasing less than logarithmically with the volume V, and the CPU time per sweep increasing roughly as $V\log V$. From our data we note that the number of MG cycles per inversion is essentially independent of the volume. As expected, for the overrelaxation algorithm the number of gauge-fixing sweeps is proportional to the lattice side N, and the CPU time per sweep grows as the volume N^4. The data for the total CPU time for the two implementations of the FA method and for the overrelaxation method are well fit by $V\log V$ and N^5, respectively. An extrapolation of our data using these fits predicts that either version of the FA method would be less costly than the overrelaxation method already at lattice side $N=24$. Actually, the CPU cost for MGFA scales slightly better with the volume than for FFTFA. Thus, on a serial machine, and at $\beta=\infty$, the fast Fourier transform can be successfully replaced by the MG routine, and MGFA should be the method of choice in the limit of large lattice volume. MGFA is equivalent to FFTFA also at $\beta=2.2$ and $V=8^4$, and therefore it appears that the use of FFT can be avoided also at finite β.

We remark that the FA method may have convergence problems at low values of β [6,21], probably related to the large number of local minima of the functional $\mathcal{E}_U[g]$ and/or to the existence of topological objects on the lattice. These problems are more likely to affect a global method than a local one (such as overrelaxation). A possible solution may be the smearing approach recently introduced in [22]: by smoothing out the lattice gauge configuration, one can perform gauge fixing at an effective β close to infinity; this result is then used as a preconditioning of the original gauge-fixing problem, i.e., the gauge-fixing iterations start already close to a minimum. This approach aims at eliminating the problem of Gribov copies on the lattice, and is very well suited for the FA method. In fact, the two gauge-fixing steps involved (the one for the preconditioning—at high β—and the one starting close to a minimum) are ideal applications of FA.

We then applied MGFA on two parallel (APE-Quadrics) machines of the APE100 series: the Q1 (2^3 processors) and the QH4 (8^3 processors) [23]. In order to implement the method on a parallel machine, the idea is to use as the coarsest grid for the MG routine a grid with volume equal to or larger than the number of processors. This avoids the problem of idle processors discussed in Ref. [15]. For example, for the Q1 we implemented the MG routine with coarsest grid 4^4 (respectively, 6^4) for the lattice volume $V=8^4$ (respectively, $V=12^4$), while for the QH4 we used 8^4 as the coarsest grid [24]. We have checked that the number of gauge-fixing sweeps does not change, if we use for the MG routine an accuracy in the range 10^{-4}–10^{-5}. This confirms the result obtained for a serial machine (see Fig. 1). In all our runs on APE machines, we set the accuracy for the inversion to 5×10^{-5}. Since these machines work in single precision, we have also decreased the stopping criterion for the gauge fixing to $(\nabla\cdot A)^2\leqslant10^{-7}$.

As an alternative to the MG inversion, we have also implemented a version of the Fourier acceleration method in which the Laplacian Δ is inverted using a standard (iterative) conjugate-gradient (CG) method. We call this method CGFA. The CG algorithm is simpler to program, and has been widely used on parallel machines. Its CPU time per cycle should be smaller than the one for MG, but the number of iterations required in order to achieve a fixed accuracy should increase faster than for the MG routine. In fact, we have checked that the number of multigrid iterations is essentially independent of the lattice volume, while the number of CG iterations grows roughly as $N^{0.37}$.

In Table II we report the number of gauge-fixing sweeps obtained, at $\beta=\infty$ and for several lattice sizes, for the MGFA and the CGFA methods on APE machines. Their performances are compared with that of a standard overrelaxation (OVE) and with that of an unaccelerated local algorithm (the so-called Los Alamos algorithm, LOS) [6,7]. These runs on parallel machines confirm that local algorithms are usually not able to achieve a constant value for the longitudinal gluon propagator at zero three-momentum. This can be checked by changing the stopping criterion for the gauge fixing: instead of considering $(\nabla\cdot A)^2$ we can consider

TABLE II. Results for the so-called Los Alamos (LOS) method, the overrelaxation (OVE) method, the CGFA and MGFA algorithms at $\beta=\infty$. Runs were carried out on the Q1 and QH4 machines, for the lattice volumes $V=8^4,12^4$ and $V=16^4,32^4$, respectively. The optimal choice for the tuning parameter α for the two FA methods is also reported. In all cases we stopped the gauge fixing when the condition $(\nabla\cdot A)^2\leqslant 10^{-7}$ was satisfied. For the second set of data at $V=32^4$ the condition $e_6\leqslant 5\times 10^{-6}$ has been used.

Algorithm	V	$\alpha_{\rm opt}$	GF sweeps	CPU time (s)
LOS	8^4		57.3 ± 0.6	≈ 1
OVE	8^4		23.0 ± 0.3	<1
CGFA	8^4	1.14	13.9 ± 0.1	4.56 ± 0.07
MGFA	8^4	1.10	13.8 ± 0.1	13.4 ± 0.2
LOS	12^4		117 ± 2	18.4 ± 0.4
OVE	12^4		33.6 ± 0.5	5.9 ± 0.1
CGFA	12^4	1.22	16.4 ± 0.3	34.1 ± 0.7
MGFA	12^4	1.24	16.2 ± 1.0	92 ± 2
LOS	16^4		198 ± 2	≈ 1
OVE	16^4		46.3 ± 0.6	<1
CGFA	16^4	1.33	17.6 ± 0.2	2.12 ± 0.04
MGFA	16^4	1.26	17.3 ± 0.1	5.95 ± 0.08
LOS	32^4		640 ± 20	83 ± 2
OVE	32^4		84 ± 2	12.0 ± 0.3
CGFA	32^4	1.38	21.5 ± 0.7	49 ± 1
MGFA	32^4	1.35	20.7 ± 2.0	98 ± 3
LOS	32^4		1970 ± 90	553 ± 30
OVE	32^4		252 ± 70	75 ± 20
CGFA	32^4	1.34	23.4 ± 0.8	57 ± 2
MGFA	32^4	1.33	22 ± 3	123 ± 3

the quantity e_6 defined in Refs. [6,7], which monitors the fluctuations of this gluon propagator. Results are reported in Table II for the lattice volume $V=32^4$. (Note that for LOS and OVE the results, in this case, have a large statistical error, due to the fact that for some configurations the gauge fixing did not converge within 3000 sweeps.)

Also in the parallel case, the FA method eliminates CSD at $\beta=\infty$, and therefore should be the method of choice on large lattices. With respect to the overrelaxation method, the two implementations of FA are competitive already at volume 32^4, if one requires a sensitive stopping criterion for the gauge fixing. We observe that, at our lattice sizes, the CG implementation is about two times faster than MGFA, but at large volumes we expect MGFA to win out.

We recall that local methods are more efficiently implemented on parallel machines than global ones, since they require smaller communication between processors. Global methods need implementations specifically designed for parallel machines in order to achieve a significant reduction of their computational overhead. For example, in a parallel implementation, the update for MG is not exactly of the Gauss-Seidel type, and in fact we observe that a higher (fixed) number of MG iterations is needed [25]. We think that our code for the MG routine can be optimized, since we have not explored more advanced features of MG that can play a role on parallel machines, such as asynchronous multigrid and the use of accommodative cycles instead of a fixed cycling strategy.

We have checked the dependence of the performance of the algorithms on the number of processors using the data from the Q1 and from the QH4, respectively, for lattice volume 12^4 and 32^4. The CPU time per gauge-fixing iteration per site scales down by a factor of approximately 62 for all the four algorithms, showing that the two FA methods parallelize as efficiently as the local ones. (Note that the number of processors increases by a factor 64 going from the Q1 to the QH4.)

We have shown that the fast Fourier transform in the Landau gauge-fixing Fourier acceleration method can be successfully substituted—on serial as well as on parallel machines—by an alternative inversion routine. This idea can in principle be extended to other applications of FA such as the case of quark propagators, and the Monte Carlo method for thermalization of lattice configurations.

ACKNOWLEDGMENTS

Simulations were done on an IBM RS-6000/340 workstation at New York University, and on two APE-Quadrics machines at the University of Rome Tor Vergata. We wish to thank Philippe de Forcrand and Alan Sokal for helpful discussions.

[1] For a review, see, G.C. Rossi, in *Lattice '96*, Proceedings of the International Symposium, St. Louis, Missouri, edited by C. Barnard *et al.* [Nucl. Phys. B (Proc. Suppl.) **53**, 3 (1997)].

[2] See, for example, U. Wolff, in *Lattice '89*, Proceedings of the International Symposium, Capri, Italy, edited by R. Petronzio *et al.* [Nucl. Phys. B (Proc. Suppl.) **17**, 93 (1990)]; A.D. Sokal, in *Lattice'90*, Proceedings of International Symposium, Tallahasse, Florida, edited by U.M. Heller *et al.* [*ibid.* **20**, 55 (1990)].

[3] H. Suman and K. Schilling, Parallel Computing **20**, 975 (1994).

[4] C.T.H. Davies *et al.*, Phys. Rev. D **37**, 1581 (1988).

[5] G.G. Batrouni *et al.*, Phys. Rev. D **32**, 2736 (1985); G. Katz, *et al.*, *ibid.* **37**, 1589 (1988); E. Dagotto and J.B. Kogut, Phys. Rev. Lett. **58**, 299 (1987); E. Dagotto and J.B. Kogut, Nucl. Phys. **B290**, 451 (1987).

[6] A. Cucchieri and T. Mendes, in *Lattice'96* [1], p. 811; A. Cucchieri and T. Mendes (in preparation).

[7] A. Cucchieri and T. Mendes, Nucl. Phys. **B471**, 263 (1996).

[8] P. Marenzoni, G. Martinelli, and N. Stella, Nucl. Phys. **B455**, 339 (1995).

[9] K.G. Wilson, *Recent Developments in Gauge Theories*, Proceedings of the NATO Advanced Study Institute, Cargèse, 1979, edited by G. 't Hooft *et al.* (Plenum, New York, 1980); J.E Mandula and M. Ogilvie, Phys. Lett. B **185**, 127 (1987).

[10] W.H. Press *et al.*, *Numerical Recipes in Fortran*, 2nd ed.

(Cambridge University Press, Cambridge, England, 1992).

[11] Very recently, a new implementation of FFT for SIMD systems has been presented in Th. Lippert *et al.*, hep-lat/9710060. The method is applied to a two-dimensional case and could in principle be extended to four dimensions.

[12] A. Cucchieri and T. Mendes, hep-lat/9710040 (to be published in the Proceedings of the Lattice 97 Conference).

[13] For details of the implementation and for an introduction to the recursive (deterministic) MG method, see, for example, Sec. II of J. Goodman and A.D. Sokal, Phys. Rev. D **40**, 2035 (1989), and references therein.

[14] We note that we use here the MG routine inside the FA method, as an alternative way of dealing with the Fourier transforms in Eq. (2). This should not be confused with the multigrid gauge-fixing method presented by Ph. de Forcrand and R. Gupta, in *Lattice '88*, Proceedings of the International Symposium, Batavia, Illinois, edited by A. Kronfeld and P. MacKenzie [Nucl. Phys. B (Proc. Suppl.) **9**, 516 (1989)]; A. Hulsebos, M.L. Laursen, and J. Smit, Phys. Lett. B **291**, 431 (1992). In that case, multigrid is used directly in order to minimize $\mathcal{E}_U[g]$. The method seems to reduce CSD significantly only if multigrid is combined with overrelaxation.

[15] L. Stals, *Parallel Implementation of Multigrid Methods*, Proceedings of the Sixth Biennial Conference on Computational Techniques and Applications, Canberra, 1993, edited by D. Stewart *et al.* (World Scientific, Singapore, 1993).

[16] We have used single precision for the FFT and for the MG inversion routines, while for the gauge fixing we have used double precision.

[17] Our definition of accuracy is $|r(t)|/|r(0)|$, where $r(t)$ is the residual at the tth iteration.

[18] The tuning of α was done independently for the MGFA and the FFTFA method. In all cases the optimal choice $\alpha_{\rm opt}$ was the same. This result confirms that the use of a different routine for inverting the Laplacian Δ does not modify the performance of the gauge-fixing algorithm, if a high enough accuracy is required for this inversion.

[19] J.E. Mandula and M. Ogilvie, Phys. Lett. B **248**, 156 (1990).

[20] We have *chosen* the version of the MG routine that provides us with the same number of gauge-fixing sweeps as for the FFTFA. In this way the critical behavior of the MGFA method is that of the FFTFA method (CSD almost completely eliminated). Alternatively, one could choose to require even less accuracy in the inversion of Δ, which would imply a higher number of gauge-fixing sweeps (see Fig. 1), but possibly with a lower total CPU cost. This would represent a modified MGFA method, with a different critical behavior, and it is not clear whether this method would show no CSD.

[21] F. Gutbrod, ''A Study of the Gluon Propagator in SU(2) Lattice Gauge Theory,'' DESY-96-252 report.

[22] J.E. Hetrick and Ph. de Forcrand, hep-lat/9710003 (to be published in the Proceedings of the Lattice 97 Conference).

[23] The Q1 has a peak performance of about 400 Mflops and a total memory of 32 Mbytes, while the QH4 has a peak performance of about 25.6 Gflops and a total memory of 2048 Mbytes. Both machines work in standard single precision, and have a 3D torus topology.

[24] On the coarsest grid, we have used Gauss-Seidel relaxation. In vector and parallel machine applications of multigrid one often uses a conjugate gradient algorithm to relax the solution on the coarsest grid [see, for example, S. Solomon and P.G. Lauwers, in *Proceedings of the Workshop on Fermion Algorithms*, Jülich, 1991, edited by H.J. Herrmann and F. Karsch (World Scientific, Singapore, 1991), p. 149]. In our case, even for our largest coarsest grid (8^4), we do not see an effective gain with respect to the simple Gauss-Seidel update (see Ref. [12]).

[25] We have checked that increasing the number of relaxation sweeps per grid causes a reduction of the total MG cycles needed per inversion, but does not lead to a computational gain.