AD

CERN-PRE 88-020
a

LA-UR-88-824

Comparison of Update Algorithms for Pure Gauge SU(3)

Rajan Gupta[§]

Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM 87545

Gregory W. Kilcup[†]

Physics Department, Brown University, Providence, RI 02912

Apoorva Patel

Theory Group, CERN, Genève 23

Stephen R. Sharpe[*]

Stanford Linear Accelerator Center, Stanford University, Stanford, CA 94305

Philippe de Forcrand

Physics Department, University of Minnesota, Minneapolis, MN 55455
and
Cray Research Inc., 1333 Northland Drive, Mendota Heights, MN 55120

ABSTRACT

We show that the overrelaxed algorithm of Creutz and of Brown and Woch is the optimal local update algorithm for simulation of pure gauge SU(3). Our comparison criterion includes computer efficiency and decorrelation times. We also investigate the rate of decorrelation for the Hybrid Monte Carlo algorithm.

3/88

[§] J. Robert Oppenheimer Fellow.

[†] Also Visiting Scientist at Boston University

[*] Work supported by the Department of Energy, contract DE-AC03-76SF00515

1. Introduction

In this paper we compare the following update algorithms for SU(3) pure gauge theory: (A) 20 hit Metropolis [1] (B) Overrelaxed [2-3] (C) Pseudo heat bath [4] and (D) Hybrid Monte Carlo [5-8]. The yardstick which we use is CPU time on a Cray X-MP per unit of decorrelation length, where the correlations are measured using blocked loops. We find that the overrelaxed algorithm (ORA) decorrelates considerably faster than the other algorithms. We also show how to generalize the overrelaxed algorithm to improved actions. The resulting algorithm again decorrelates considerably faster than a Metropolis algorithm.

We have undertaken this study for three main reasons. The first reason is to provide a detailed comparison of the Metropolis and pseudo heat bath algorithms. These are the algorithms which are used in practice on vectorized computers. To our knowledge such a comparison has not been made beyond the test in the original proposal [4]. We do not consider the full SU(3) heat bath algorithm [9] since its complexity makes it inefficient for vector machines.

Our second purpose is to evaluate the various types of ORA, and to compare them to the standard algorithms. The standard algorithms are local, and suffer from critical slowing down. A number of years ago Adler proposed that overrelaxed algorithms had the potential to alleviate this problem [10] . This has been demonstrated analytically for simple models [11] [12] . Recently Creutz [2] and Brown and Woch [3] have suggested overrelaxed algorithms which are simple to implement for SU(N) gauge groups. In studies on small lattices they found that the decorrelation times for the ORA were significantly shorter than for the standard algorithms. In this study we extend their work onto larger lattices (9^4) at higher β, and use improved observables to measure decorrelations.

Our final purpose is to evaluate the Hybrid Monte-Carlo Algorithm (HMC). This is an exact algorithm which shows promise for simulations of QCD including dynamical fermions. Since it is a small step size algorithm, we do not expect it to be competitive with the other algorithms for simulations of the pure gauge theory. However, when one includes dynamical fermions one is forced to use small step size algorithms, and thus it is interesting to know how quickly configurations decorrelate.

We study the SU(3) gauge theory for couplings in the range $5.8 \leq \beta \leq 7.5$ on 9^4 lattices. We measure small Wilson loops on the sequence of lattices produced by a scale factor $\sqrt{3}$ renormalization group transformation ($9^4 \rightarrow (3\sqrt{3})^4 \rightarrow 3^4 \rightarrow (\sqrt{3})^4 \rightarrow 1^4$). The loops we measure are the plaquette (P) and the three 6 link loops: the rectangle (R), the L-shaped (L) and the twisted loop (T). The auto-correlations of blocked loops are then calculated at each level and compared for the different algorithms. The auto-correlation coefficients for any loop expectation X using N measurements are defined as

$$C_n \;=\; \frac{\langle XX_n\rangle - \langle X\rangle\langle X_n\rangle}{\sqrt{\left(\langle XX\rangle - \langle X\rangle^2\right)\left(\langle X_nX_n\rangle - \langle X_n\rangle^2\right)}} \tag{1.1}$$

where for example $\langle XX_n\rangle \equiv \sum_{i=1}^{N-n} X_iX_{i+n}$, $\langle X\rangle \equiv \sum_{i=1}^{N-n} X_i$ and $\langle X_n\rangle \equiv \sum_{i=1}^{N-n} X_{i+n}$. We define the decorrelation time ξ to be the number of sweeps at which the auto-correlation coefficients fall to 0.2. Our conclusions are based on a comparative study, and do not depend on the precise definition of ξ.

An important feature of our calculation is the use of blocked loops. Ideally, one should measure the decorrelation times for physical observables such as large Wilson loops. Small loops on blocked lattices are, in fact, preferable to large Wilson loops on the original lattice in two ways. First, ultraviolet fluctuations are cancelled in the the sum of loops on the original lattice which makes up the single blocked loop. And, second, blocked loops are simpler to calculate. Of course, the advantages of using blocked loops compared to calculations of *small* loops on the original lattice are even greater. We have shown a dramatic example of these advantages of using blocked loops in Ref. [13] .

The confinement length scale (i.e. $1/T_c$) at $\beta = 5.9$ is ~ 6 and at $\beta = 6.5$ is ~ 16 [14] . Assuming asymptotic scaling, the length scales at $\beta = 7.0$ and 7.5 are then 27 and 48 respectively. Thus with a 9^4 lattice we are not probing the confinement length scale for $\beta > 6$. This means that we cannot test how the algorithms are critically slowed down as $\beta \to \infty$. This is unfortunate, since one of the merits of the ORA is that it might reduce the exponent governing critical slowing down, as well as the amplitude. However, since the reduction in amplitude is already enough to strongly favor the ORA, this shortcoming does not affect the ranking of the algorithms.

2. Details of algorithm implementation.

We give a brief description of the algorithms to make this paper self contained and to introduce the notation. All descriptions are for the Wilson action.

2.1 The Metropolis Algorithm

A given link is changed to $U' = hU$ and accepted with probability

$$\text{prob} = \min\left(1, \frac{P(U' \to U)}{P(U \to U')}\,\frac{e^{-H(U')}}{e^{-H(U)}}\right) \tag{2.1}$$

where the hit matrix h is an element of SU(3), and H is the gauge action. In practice one selects h from a distribution with equal probability for h and $h^\dagger$ so that the factor $P(U' \to U)/P(U \to U')$ is equal to unity . We use 20 hits at each link update. The hit matrices are selected using a trace biased algorithm [15] and the acceptance is adjusted to $\approx 30\%$. In the limit of infinite hits one gets the same distribution as from the heat bath algorithm [16] . Tests show that the algorithm's efficiency has a peak between 15 to 30 hits.

Using a checkerboard update, the time required for 20 hit update is 58 μs per link on a single processor Cray X-MP. Of this $22\mu s$ are spent in gathering and summing the staples and thereafter each hit takes $1.8\mu s$.

2.2 The Overrelaxation Algorithm

The variant of ORA we use is very similar to a Metropolis hit except that the new trial link is taken to be [2,3]

$$U' = \Sigma^\dagger U^{-1} \Sigma^\dagger \tag{2.2}$$

where U is the old link variable and Σ is the projection back onto the gauge group of the sum of staples adjoining the link (V). This choice of trial link satisfies detailed balance because of the symmetry $P(U \to U') = P(U' \to U)$. For abelian groups, and for SU(2), the projection $V \to \Sigma$ is simply a rescaling, and it is easy to see that the action is unchanged by the replacement $U \to U'$. Thus the acceptance is 100%, and the algorithm is microcanonical. For SU(3), the change in the action, and consequently the acceptance, depends on the method of projection. We use the same method as in the construction of block links, namely that Σ is the SU(3) matrix that maximizes $\mathrm{Tr}(\Sigma^\dagger V)$ [17] . We find an acceptance rate that varies between 99.5% to 99.7% as β is changed from 5.9 to 7.5. These acceptances are higher than those found in Refs. [2-3], partly because we work at larger β, and partly because we use a different projection. Clearly, our implementation of the ORA step is almost microcanonical. This is confirmed by the data: the plaquette on the 9^4 lattice is almost constant. However, we also observe that after a sufficient number of blockings the value of the plaquette with ORA is similar to that with the Metropolis algorithm. Thus, the long distance physics shows no detectable memory of the almost microcanonical nature of the update procedure.

The ORA hit can be supplemented by any number of standard Metropolis hits. We denote by **ORA*n*** the algorithm with n such hits. For $n \to \infty$, one again converges to the heat bath distribution. The goal is to adjust n to avoid the microcanonical, and possibly non-ergodic, nature of ORA and still have faster decorrelations than the heat bath algorithm.

An alternate way to implement this algorithm is to make a certain number of ORA sweeps and follow them with some Metropolis sweeps. We have not tried this approach to optimization.

The timing for ORA0 is 48 μs per link, with each subsequent Metropolis hit adding 1.8 μs.

2.3 Cabibbo-Marinari Pseudo Heat Bath (PHB)

In this algorithm, SU(2) subgroups of SU(3) link matrices are successively updated using the heat bath algorithm. We apply 3 successive heatbath hits with the first and third submatrices being identically constructed. This is the most common implementation, and for the SU(2) hits we use the prescription of Fabricius and Haan [18] .

The timing for a single link PHB update with the code used in this paper is $22\mu s$ for the gathers, and $6\mu s$ for each of the 3 hits. One of us (P. de F.) has a highly optimized version running at $\approx 12\mu s$ for gathers and $\approx 4\mu s$ per hit. A similar optimization for other algorithms would decrease their update time also.

Improvements to this algorithm might include taking 3 distinct subgroups for the 3 heatbath hits, and using more than 3 hits with a fully vectorized SU(2) version following the suggestion of Fredenhagen and Marcu [19] .

2.4 Hybrid Monte Carlo

Our implementation of the HMC uses the Φ-algorithm of Ref. [8]. This is based on the Hamiltonian

$$H = \frac{1}{2}\,\mathrm{Tr}\sum P^2_{i,\mu} + \frac{\beta}{N}\sum \mathrm{Re}\,\mathrm{Tr}(1-U_p) \tag{2.3}$$

where $P_{i,\mu}$ are the traceless antihermitian momenta conjugate to the link variables $U_{i,\mu}$, and U_p is the plaquette.

The U and P are evolved in molecular dynamics (MD) time using the following steps. At the initial time t the links have initial values $U = U(t)$.

1. Refresh the momenta $P(t)$ by drawing them from a distribution $\exp(-\,\mathrm{Tr}\,P^2/2)$. This is done by representing the momenta as

$$P_{i,\mu} = \sum_{n=1}^{8} r^n_{i,\mu}\lambda_n \quad . \tag{2.4}$$

Here λ_n are generators of $SU(3)$ normalized to $Tr(\lambda_a\lambda_b) = 2\delta_{ab}$ and $r^n_{i,\mu}$ are eight independent real gaussian random variables with variance $\frac{1}{2}$.

2. Evolve U forward a half step using

$$U(t+\frac{\epsilon}{2}) = e^{i\frac{\epsilon}{2}P}U(t) \ . \tag{2.5}$$

This form of the discretization of Hamilton's equations keeps U as an element of SU(3). We approximate the exponential by a fourth order polynomial and reunitarize the resulting matrix.

3. Calculate $\dot{P}$ at $t+\frac{\epsilon}{2}$ using

$$i\dot{P}_{i,\mu} = [-\ \frac{\beta}{N}U_{i,\mu}V_{i,\mu}\ ]_{TA}\ , \tag{2.6}$$

where TA stands for the traceless antihermitian part and $V_{i,\mu}$ are the staples adjoining $U_{i,\mu}$. This equation is determined by the requirement that H be a constant of motion.

4. "Leap frog" the momenta forward by a whole time step using the discrete approximation $P(t+\epsilon) \;=\; P(t) \;+\; \epsilon \times \dot{P}$.
5. Evolve U to $t + \frac{3\epsilon}{2}$ using $P(t+\epsilon)$ as in Eqn. (2.5). Steps 3 to 5 are repeated n_{md} times. At the end we have $P' \equiv P(t + n_{md}\epsilon)$ and $U(t + (n_{md} - \frac{1}{2})\epsilon)$. To complete the leap frog step we calculate $U' \equiv U(t + n_{md}\epsilon)$ as in (2.5).
6. The errors in this evolution arise in the discrete approximations used in Eqns.(2.5) and (2.6). These errors are removed by the global Metropolis step. The new configuration U' is accepted with probability

$$\min\bigl(1, e^{-\delta H}\bigr) \;=\; \min\bigl(1, \frac{e^{-H(U',P')}}{e^{-H(U,P)}}\bigr) \;. \tag{2.7}$$

At this point we return to step 1.

These equations yield an exact algorithm if the MD steps are reversible, i.e. only if the configuration $(U', -P')$ evolves back into $(U, -P)$. The only source of irreversibility is incomplete exponentiation in Eqn. (2.5). We have checked that this effect is undetectable by using sixth order expansion in test runs [20] . Nevertheless, we would recommend higher order or exact exponentiation for future tests, since it requires very little extra CPU time.

The algorithm remains exact even if the coupling in the MD steps, β_h, differs from that used in the accept/reject. This allows one to tune β_h, as well as ϵ and n_{md}, in order to maximize motion through configuration space. We have recently made a study of this tuning, and find that β_h slightly smaller than β is preferred in the sense of yielding higher acceptances [20]. However, the loss in acceptance using $\beta_h = \beta$ is reasonably small, and so in this paper we adopt this simple approach. This allows us to make preliminary estimates of the sensitivity to ϵ and n_{md} using correlation lengths rather than acceptances.

The luxury of being able to set $\beta_h = \beta$ at little cost is only true for the implementation just described in which the links are moved forward first. The alternative, i.e. first evolving the momenta [6][8], turns out to have its peak acceptance for β_h further from, and in fact greater than, β [20].

The timing for a single molecular dynamics step is 0.68 s, which corresponds to 25 μs per link.

3. Results

3.1 We begin with a comparison of the ORA algorithms. We have 10,000-20,000 sweeps at $\beta = 7.5$ for each of ORA0, ORA1 and ORA2. We measure the blocked loops every 10 sweeps, and plot the resulting correlation coefficients for the T loops on level 5 in Fig. 1.

Note that the horizontal axes are given in units of number of measurements. Thus C_{20} is the measurement of the correlation after 200 sweeps. The same is true for all figures in this paper. We also include in Fig. 1 the data for the Metropolis algorithm.

The figures show that there is little difference between ORA0 and ORA1, while ORA2 decorrelates more slowly. ORA2 also appears to have a long tail for $C_n < 0.2$. The Metropolis algorithm exhibits much slower decorrelation than any of the ORA, though we expect that the addition of further hits to the ORA would increase the correlation length towards that of the Metropolis algorithm. We conclude from this that ORA0 and ORA1 are the optimal overrelaxed algorithms. We prefer ORA1 because it is not microcanonical, and we use it as the representative of the ORA for the remaining tests.

3.2 We next compare the ORA with the Metropolis and PHB algorithms. We have done this at $\beta = 5.9$ and 6.5. These values bracket the range accessible for measurememt of static hadronic properties with present and next-generation computers. All runs use at least 10,000 sweeps, with measurements made every 10 sweeps. Some examples of the correlation lengths we deduce are given in Table 1. We show in Fig. 2 the correlation coefficients for the R loop on level 5 at $\beta = 5.9$. We note the following features of the results.

- For $\beta = 5.9$, the ORA1 appears marginally better than the Metropolis algorithm and $2 - 4$ times faster than the PHB. As the table shows, there are considerable uncertainties in these relative speeds because of the loop size dependence.
- When β increases to 6.5, the ORA1 stands out as a clear winner over the other algorithms by a factor of ~ 2. The Metropolis and PHB decorrelate at roughly the same rate.
- The correlation length deduced from the T loops is systematically higher than that from the rectangle R at $\beta > 6.0$. However, both loops give roughly the same relative merits to the algorithms.
- The Metropolis and PHB algorithms show a substantial tail in the correlation coefficients. Much longer runs would be needed to ascertain if this implies that there are even longer range correlations.

3.3 As part of a calculation of the β−function [21] we have extended the runs with ORA1 and the Metropolis algorithm to other values of β. The results are collected in Table 1. We find that the results at $\beta = 6.5$ are almost unchanged for values of β up to 7.5. Thus the ORA1 algorithm is preferred over this entire range. We stress that the constancy of ξ is not evidence for the absence of critical slowing down. The loops that we are measuring do not probe physical scales for $\beta > 6$, and decorrelation times for physical observables will be significantly larger.

The other noteworthy feature is that ξ appears to have a peak at $\beta \approx 5.9$. This is presumably related to the finite temperature transition and the nearby peak in the specific heat.

3.4 Finally, we present a comparison of the HMC with the other algorithms. We have made two runs at $\beta = 5.9$. Run (a) uses $\epsilon = 0.0375$ and $n_{md} = 20$, so that $\epsilon \times n_{md} = 0.75$. This run has an average acceptance of 70%. Run (b) has $\epsilon = 0.025$, $n_{md} = 40$, so that $\epsilon \times n_{md} = 1.0$, and has an acceptance of 84%. Measurements are made after every second accept/reject for run (a), and after every accept/reject in run (b). These two runs are compared in Fig. 3, where data for the level 5 R loop is shown. Since a measurement is made after 40 MD steps in both runs, the horizontal axis can be thought of as CPU time. To convert to the more conventional units of MD time ("trajectories"), the horizontal axis for run (a) must be multiplied by 1.5. The values for ξ from the various loops at levels 4 and 5 are in the range 40-50 for run (a), and 80-90 for run (b), in units of measurements (and CPU time). In units of MD time, the run (a) result is 60-75, while the run (b) numbers are unchanged.

β	Level	ORA1		20 hit Met		Pseudo Heat Bath	
		R	T	R	T	R	T
5.8	4			80	80		
5.8	5			40	40		
5.9	4	130	130	140	140	245	270
5.9	5	70	60	145	150	240	230
6.0	4			110	115		
6.0	5			105	90		
6.125	5			40	80		
6.25	5			40	80		
6.5	5	40	90	90	200	130	170
7.0	5	40	60	110	180		
7.25	5	40	60	100	170		
7.5	5	40	60	140	210		

Table 1. *The auto-correlation length ξ (in units of sweeps) for blocked Wilson Loops. We define ξ such that $C_\xi = 0.2$. The loops are calculated every 10 sweeps, so an estimate for the error is 20 sweeps. The data sample for each set is at least 1000 configurations.*

We draw three conclusions from these results.

- There is a 30% improvement in the decorrelation in MD time upon changing from $\epsilon \times n_{md} = 1.0$ to $\epsilon \times n_{md} = 0.75$ at fixed ϵ. To reach this conclusion we have assumed that the acceptance depends only on ϵ and not on n_{md}, as is observed in numerical tests [20]. We have also assumed that the product $\xi \times$ *acceptance*, i.e. the MD distance actually moved, depends only on $\epsilon \times n_{md}$. In particular we assume that had we made run (a) using $\epsilon = 0.025$ and $n_{md} = 30$, the acceptance would have increased to 84%, and the correlation length in units of MD time would have decreased from ~ 68 to ~ 56. Our result for the optimal $\epsilon \times n_{md}$ is similar to those found using hybrid algorithms alone (i.e. with no accept/reject) to study dynamical fermions on small lattices. Both Refs. [22] and [23] find that the optimal value of $\epsilon \times n_{md}$ is somewhat less than 1.

- The optimal ϵ is close to 0.0375. If we increase ϵ further, while holding $\epsilon \times n_{md}$ fixed, the acceptance will drop rapidly, and overwhelm the gain in CPU time. These issues are discussed in more detail in Ref. [20].
- Compared to the other algorithms one unit of MD time is roughly equivalent to a single ORA1 sweep, 2 Metropolis sweeps, or 2 PHB sweeps. Ref. [24] finds a similar ratio between a pure hybrid algorithm and a 15 hit Metropolis algorithm. Clearly for pure gauge simulations the HMC, or for that matter all small step algorithms, are much slower in CPU time than the other algorithms. Run (a) uses about 20 times as much CPU time as the ORA1.

4. An overrelaxed algorithm for Improved Gauge Actions

Improved actions that include Wilson loops in higher representations are not linear in the link being updated. We therefore need to modify the construction of Σ in (2.2). The necessary condition on Σ is that it be independent of the link being updated. To keep the acceptance rate high we need to construct Σ such that the change in the action is small. For an action consisting of the plaquette in the fundamental, 6 and 8 representations and the 1×2 loop, the ansatz we use is

$$\Sigma \;=\; \mathcal{P}\Big(\sum_{6} V \;+\; \frac{K_{1\times2}}{K^{\text{eff}}} \sum_{12} V_{1\times2}\Big) \tag{4.1}$$

where $\mathcal{P}$ is the projection back to SU(3), and V ($V_{1\times2}$) are the staples making up the plaquette (1×2 loop) with the link being updated, $K_{1\times2}$ is the coupling of the 1×2 loop and K^{eff} is the solution to the quadratic equation

$$K^{\text{eff}} \;=\; (K_f + 5/2K_6 + 9/4K_8) \;-\; \frac{(7.5K_6 + 5.625K_8)}{(K^{\text{eff}} + 8K_{1\times2})} \tag{4.2}$$

where K_f, K_6 and K_8 are the couplings in the fundamental, 6 and 8 representations respectively. The first term is the tree-level effective one parameter coupling which includes the effect of the plaquette in the fundamental and higher representations. The second term is the large-N Hartree-Fock correction [25] .

We have run tests with this ansatz at weak coupling (average plaquette ≈ 0.8) and find 96% acceptance. If, at stronger coupling, $K^{\rm eff}$ defined by (4.2) is not efficient, then we plan to use it as a starting guess and then optimize $K^{\rm eff}$ numerically to get the fastest decorrelation for a given CPU time.

We find an improvement in decorrelation time of $5-6$ compared to the 20 hit Metropolis algorithm. Also, the CPU time used per sweep for ORA1 is smaller by 25%.

5. Conclusions

We draw two main conclusions from this study. The first is that the overrelaxed algorithm of Refs. [2-3], with a single Metropolis hit, is at present the optimal algorithm for pure gauge simulations over the entire range of accessible β. This remains true for improved gauge actions. The differences between the Metropolis and pseudo heat bath algorithms are β dependent and small, so that the choice between them depends upon the efficiency of one's implementation.

The second conclusion is that the Hybrid Monte-Carlo algorithm decorrelates at about the same rate in molecular dynamics time as the overrelaxed algorithm does in terms of sweeps. Since the HMC requires many (in the present instance 30-40) steps to traverse a unit of molecular dynamics time, it is considerably slower than the other algorithms. Similar conclusions apply to all small step size algorithms. Of course this is no surprise, but it highlights the difficulties of simulating dynamical fermions. In such simulations, our results imply that it will be necessary to use very long runs.

Acknowledgements

This work was carried out on the Cray X-MP at the Pittsburgh Supercomputing Center. We thank R. Roskies for his support and M. Creutz for discussions. G.K., A.P. and S.S. thank Los Alamos for support while part of this work was done.

Figure Captions

1. Comparison of the T loop auto-correlations on the fifth blocked level for (a) the 20 hit Metropolis algorithm (b) ORA0 (c) ORA1 and (d) ORA2. The data is at $\beta = 7.5$. The correlation coefficients are defined in terms of measurements, which are separated by 10 sweeps through the lattice.
2. Comparison of the T loop auto-correlations on the fifth blocked level for (a) the 20 hit Metropolis algorithm (b) the 3 sub-group PHB and (c) ORA1. The data is at $\beta = 5.9$. Each measurement is separated by 10 sweeps through the lattice. The correlation coefficients are defined in terms of measurements.
3. Comparison of the R loop auto-correlations on the fifth blocked level for the HMC. The coupling is $\beta = 5.9$. Run (a) has $\epsilon = 0.0375$ and $n_{md} = 20$, with measurements every second accept/reject step. Run (b) has measurements after $n_{md} = 40$ steps of size $\epsilon = 0.025$. The correlation coefficients are defined in terms of measurements.

References

[1] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller and E. Teller, *J. Chem. Phys.* **21** (1953) 1087.

[2] M. Creutz, *Phys. Rev.* **D36** (1987) 515.

[3] F. R. Brown and T. J. Woch, *Phys. Rev. Lett.* **58** (1987) 2394.

[4] N. Cabibbo and E. Marinari, *Phys. Lett.* **119B** (1982) 387.

[5] R. T. Scalettar, D. J. Scalapino and R. L. Sugar, *Phys. Rev.* **B34** (1986) 7911.

[6] S. Duane, A. D. Kennedy, B. J. Pendleton and D. Roweth, *Phys. Lett.* **195B** (1987) 216.

[7] S. Duane, *Nucl. Phys.* **B257** (1985) 652 ;
S. Duane and J. Kogut, *Phys. Rev. Lett.* **55** (1985) 2774.

[8] S. A. Gottlieb, W. Liu, D. Toussaint,R. L. Renken and R. L. Sugar, *Phys. Rev.* **D39** (1987) 2531.

[9] E. Pietarinen, *Nucl. Phys.* **B190** (1981) 349.

[10] S. Adler, *Phys. Rev.* **D23** (1981) 2901.

[11] S. Adler, *Phys. Rev.* **D37** (1988) 458.

[12] H. Neuberger, *Phys. Rev. Lett.* **59** (1987) 1877.

[13] R. Gupta, in *Lattice Gauge Theory Using Parallel Processors*, Gordon and Breach, 1987 ;
R. Gupta, G. Guralnik, G. W. Kilcup, A. Patel and S. R. Sharpe, unpublished.

[14] N. Christ *et al.* , Proc. Int. Symp. *"Field theory on the Lattice"*, Seillac, France (9/87);
S. A. Gottlieb, W. Liu, D. Toussaint,R. L. Renken and R. L. Sugar, *Phys. Rev. Lett.* **59** (1987) 1513.

[15] G. Guralnik, T. Warnock and C. Zemach, *J. Comput. Phys.* **61** (1985) 105 .

[16] M. Creutz, *Phys. Rev.* **D21** (1980) 2308.

[17] R. Gupta, G. Guralnik, A. Patel, T. Warnock, and C. Zemach, *Phys. Rev. Lett.* **53** (1984) 1721.

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

[19] K. Fredenhagen and M. Marcu, CERN preprint TH-4669 (1987).

[20] R. Gupta, G. Kilcup, S. Sharpe, SLAC-4587, LAUR-88-886 (3/88) .

[21] R. Gupta, G. Kilcup, A. Patel and S. Sharpe, LAUR-88-971 (3/88).

[22] S. Duane and J. B. Kogut, *Nucl. Phys.* **B275** (1986) 398.

[23] S. A. Gottlieb, W. Liu, D. Toussaint,R. L. Renken and R. L. Sugar, *Phys. Rev.* **D35** (1987) 2531.

[24] S. A. Gottlieb, W. Liu, D. Toussaint,R. L. Renken and R. L. Sugar, *Phys. Rev.* **D36** (1987) 3797.

[25] B. Grossmann and S. Samuel, *Phys. Lett.* **120B** (1983) 383.

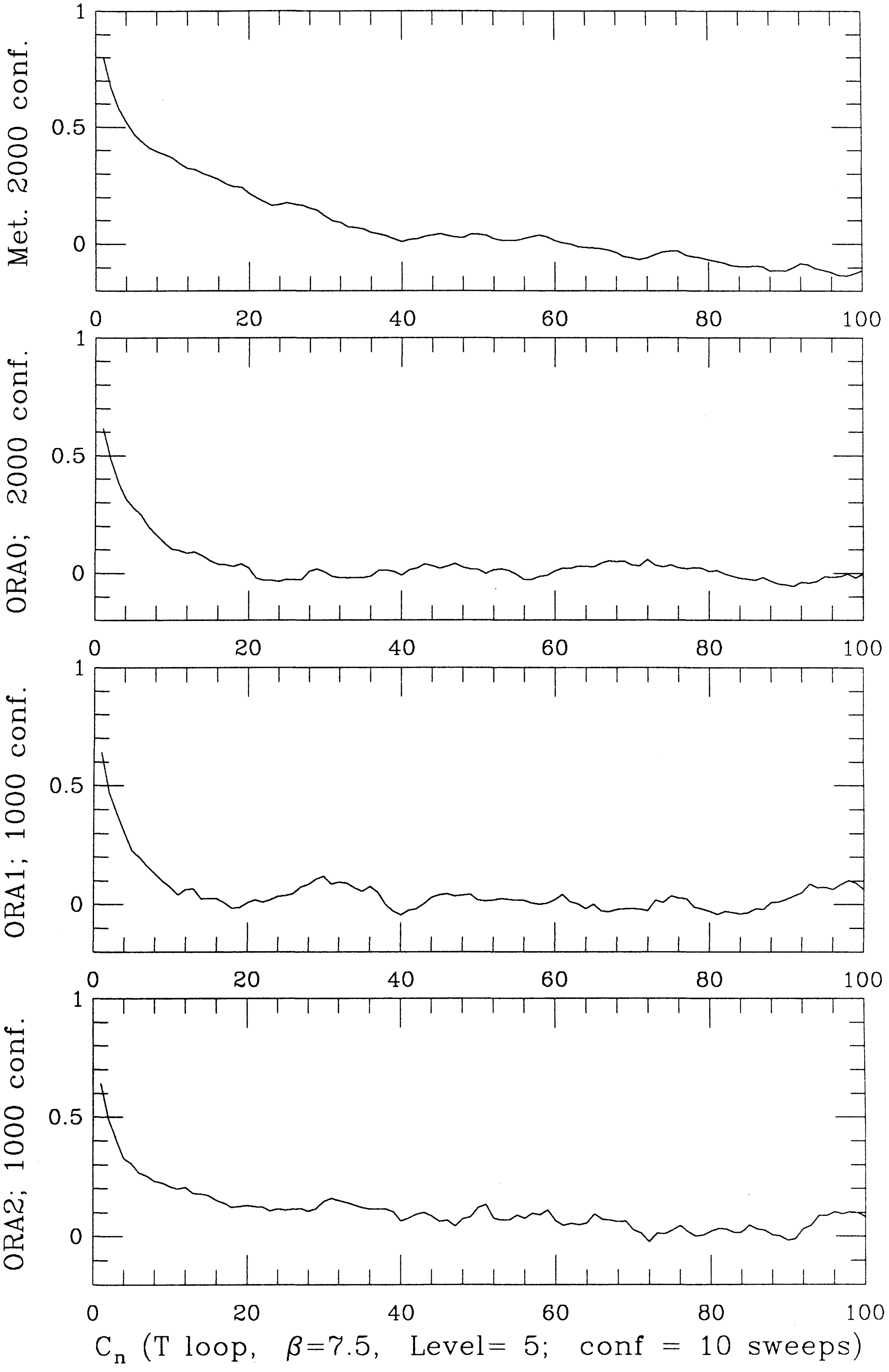
Met. 2000 conf.
ORA0; 2000 conf.
ORA1; 1000 conf.
ORA2; 1000 conf.
1
0.5
0
0
20
40
60
80
100
C_n (T loop, β=7.5, Level= 5; conf = 10 sweeps)

Figure 1

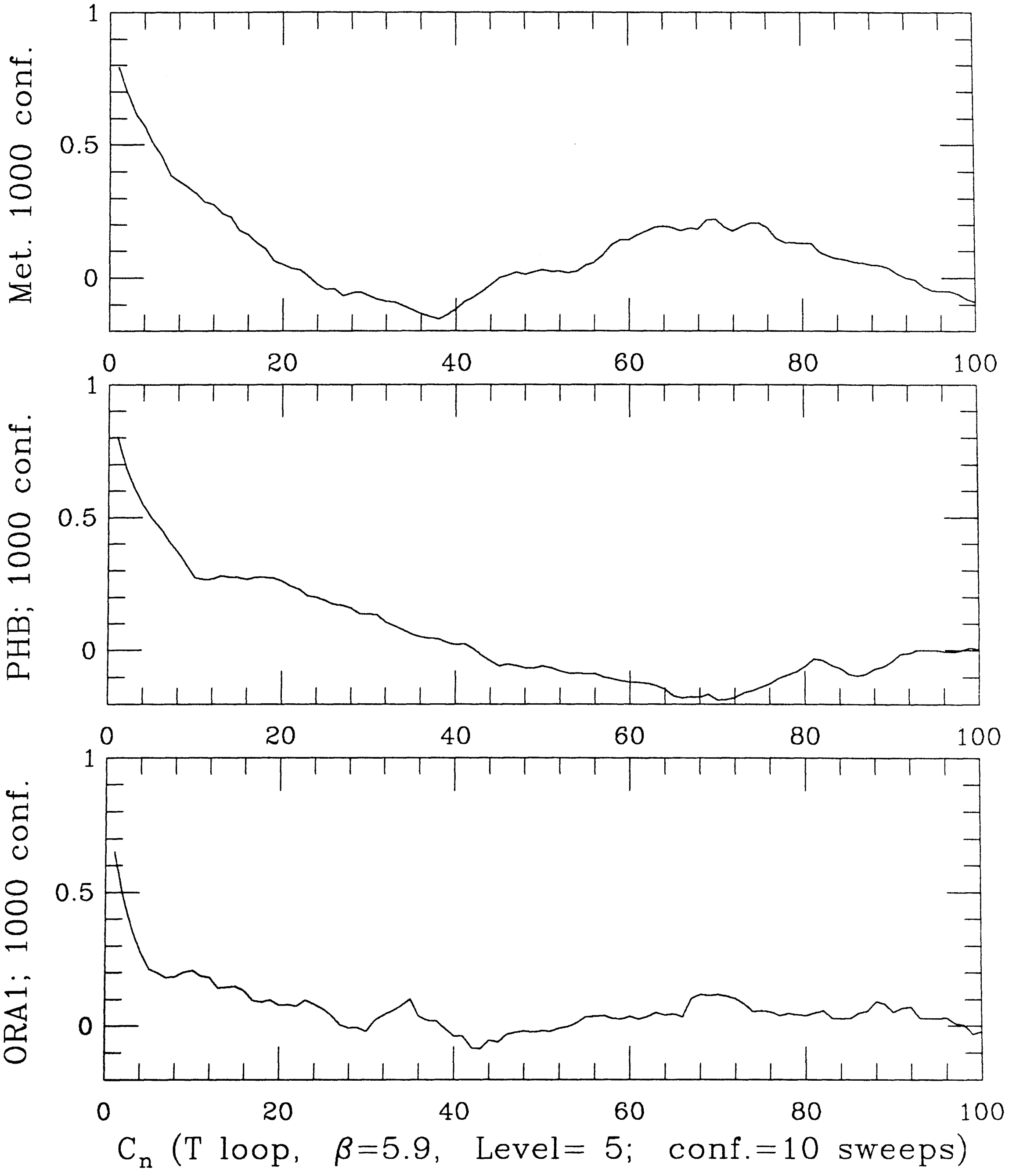
1
0.5
0
Met. 1000 conf.
0
20
40
60
80
100
1
0.5
0
PHB; 1000 conf.
0
20
40
60
80
100
1
0.5
0
ORA1; 1000 conf.
0
20
40
60
80
100
C_n (T loop, β=5.9, Level= 5; conf.=10 sweeps)

Figure 2

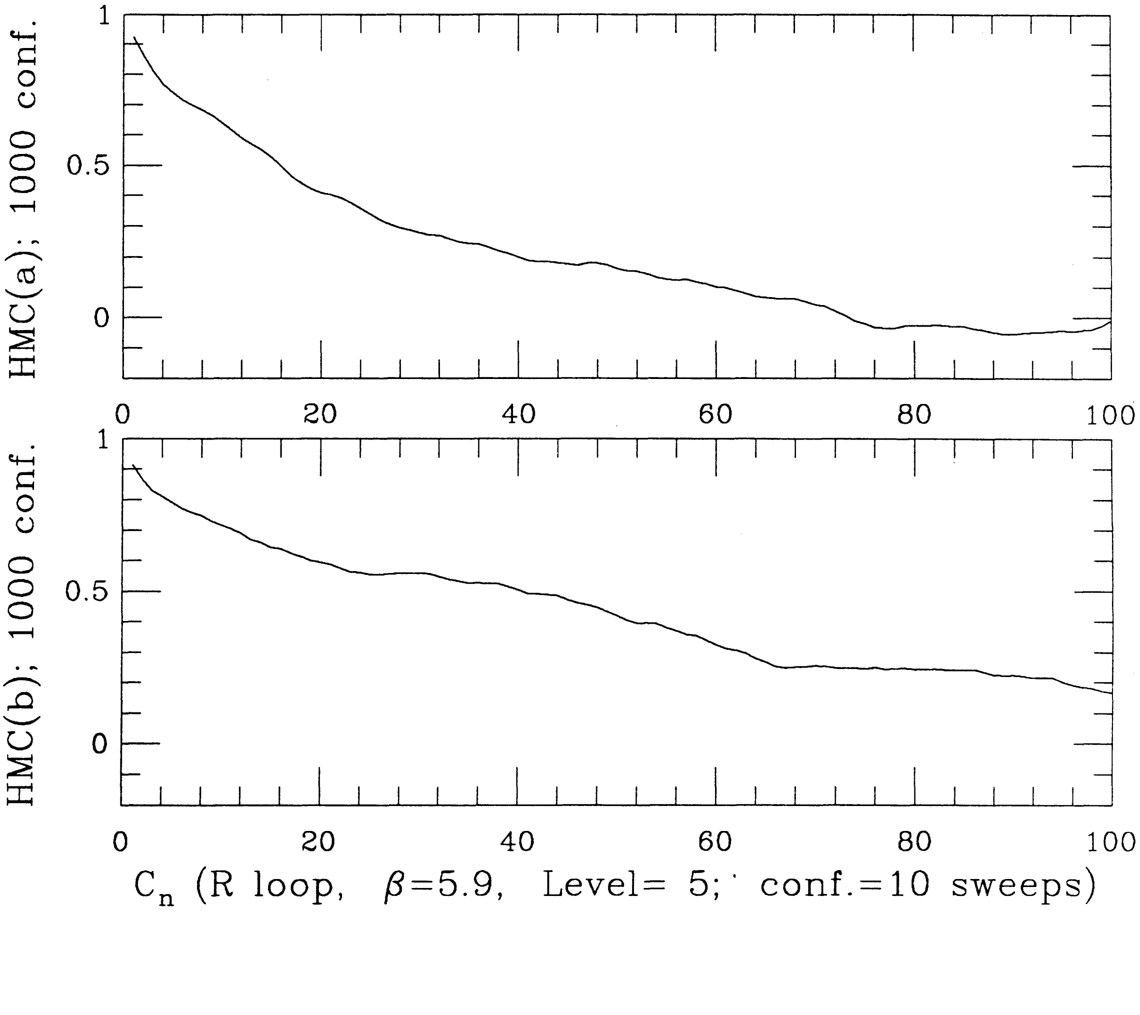
HMC(a); 1000 conf.
1
0.5
0
0
20
40
60
80
100
HMC(b); 1000 conf.
1
0.5
0
0
20
40
60
80
100
C_n (R loop, β=5.9, Level= 5; conf.=10 sweeps)

Figure 3