MONTE CARLO SIMULATION OF PURE $U(N)$ AND $SU(N)$ GAUGE THEORIES
ON A SIMPLICIAL LATTICE

J.-M. Drouffe *)

CERN - Geneva

K.J.M. Moriarty and C.N. Mouhas

Department of Mathematics
Royal Holloway College
Englefield Green, Surrey, TW20 OEX, U.K.

A B S T R A C T

The method for implementing the Monte Carlo algorithm for pure $U(N)$ and $SU(N)$ gauge theories on the recently introduced simplicial lattice is presented. The calculational technique for calculating all Wilson loops containing up to 13 triangles or 9 squares is explained.

*) Permanent address : S.Ph.T., CEN Saclay, 91191 Gif-sur-Yvette Cedex, France.

PROGRAM SUMMARY

Title of program : LATTICE

Catalogue number :

Program available from : CPC Program Library, Queen's University of Belfast, N. Ireland (see application form in this issue)

Computer : DEC VAX 11/780; Installation: Royal Holloway College Computer Centre

Operating system : DEC VAX 11/780: VMS 3.2

Programming language used : FORTRAN-77

High speed storage required : 110 Kwords

Number of bits in a word : 32

Peripherals used : terminal, line printer

Number of cards in combined program and test deck : 798

Card punching code : ASCII

Keywords : Lattice gauge theory, Yang-Mills theory, $U(N)$ and $SU(N)$ gauge theories, simplicial lattice, Wilson loops, Monte Carlo methods

Nature of the physical problem :

Lattice gauge theory is usually formulated on a hypercubical lattice [1]. However, in order to check universality it is necessary to formulate the theory on another lattice, e.g. the simplicial lattice [2].

Method of solution

Pure U(N) and SU(N) gauge theories are simulated by Monte Carlo methods on a four-dimensional simplicial space-time lattice. The method of Metropolis et al. [3] is used to equilibrate our lattice.

Restrictions on the complexity of the program

The only restriction on the program is the size of the links array which is dimensioned to

$$10 \text{ (ISIZE)}^4 \times 2N^2,$$

where ISIZE is the number of lattice sites in any space-time dimension and N^2 is the number of elements in an U(N) or SU(N) matrix. This effectively sets the limit to the size of lattice which will reside in real memory. Of course, the VAX 11/780 is a virtual memory machine and we could use this facility to examine larger lattices.

Typical running time

The test run took 6 hrs. and 17 min. on the VAX 11/780 with ISIZE = 6 and N = 2 for 42 sweeps through the lattice.

References

- [1] K.G. Wilson, Phys. Rev. D10, 2455 (1974).
- [2] J.-M. Drouffe and K.J.M. Moriarty, Nucl. Phys. B220 (FS8), 253 (1983).
- [3] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller and E. Teller, J. Chem. Phys. 21, 1087 (1953).

LONG WRITE-UP

1. Introduction

The theory of strong interactions is believed to be Quantum Chromodynamics (QCD) which is a quantum field theory. Any quantum field theory is plagued by infinities which arise from two sources; infrared divergences, which arise from the infinite range of our space-time coordinates and, ultraviolet divergences, which arise from the continuity of our space-time coordinates. In addition, the running coupling constant of Quantum Chromodynamics increases with the separation of the interacting quarks. As a result, the Feynman weak-coupling expansion, which works so well in Quantum Electrodynamics, cannot be exploited in QCD. In order to overcome this problem Wilson [1], in 1974, suggested a non-perturbative formulation of QCD for regulating the divergences by formulating the quantum field theory on an Euclidean lattice and quantizing by the Feynman path integral approach.

This lattice formulation of Quantum Chromodynamics has several advantages:

- 1) the lattice gives a gauge invariant regularization of the divergences,
- 2) no gauge fixing is necessary because the gauge fields have compact support,
- 3) all the techniques of statistical mechanics can be applied, e.g. we can calculate high-temperature expansions, can carry out block spin renormalization group calculations, etc.,
- 4) because our space-time is discrete and of finite extent we have

a finite number of degrees of freedom; hence, numerical simulation can be carried out.

Until recently, all formulations of Quantum Chromodynamics have been on a hypercubical lattice [1]. Recently, however, a number of alternatives have been considered; the random lattice [2], the body-centered tesseract lattice [3] and the simplicial lattice [4]. Of course, our continuum physics results should be independent of the lattice on which the theory is formulated, a principle which is called universality. However, in order to test universality we must carry out calculations on a lattice other than the hypercubical lattice. In addition, because of the higher geometric symmetry of, e.g. the simplicial lattice, the rotational invariance of the continuum theory is more easily restored on the lattice.

The Monte Carlo simulation of pure SU(2) [5] gauge theory on a hypercubical lattice by the heat bath method and pure U(N) and SU(N) [6] gauge theory on a hypercubical lattice by the method of Metropolis et al. [7] has recently been discussed. In the present paper, we discuss the implementation of pure U(N) and SU(N) gauge theory on a simplicial lattice by the method of Metropolis et al.

2. Outline of the theory

We wish to study U(N) and SU(N) non-Abelian gauge theories on a four-dimensional simplicial lattice, which is a generalization of the two-dimensional triangular lattice and of the three-dimensional f.c.c. lattice. To form our simplicial lattice, we start with a five-dimensional hypercubical lattice and form the hyperplane $x_1 + x_2 + x_3 + x_4 + x_5 = 0$. This defines our simplicial lattice. Among the possible boundary conditions, we have imposed periodic ones by identifying the points $\{x_i\}$ and $\{y_i\}$ such that $x_i - y_i = 0$ modulo (ISIZE); this condition is compatible with the constraint $\sum x_i = \sum y_i = 0$ and appears to be the simplest for programming.

On each link b of our simplicial lattice, joining nearest neighbour lattice sites, sits on $N \times N$ matrix $U(b) \in U(N)$ or $U(b) \in SU(N)$. We define our partition function by

$$Z(\beta) = \int \left(\prod_b dU(b) \right) \exp(-\beta S[U]),$$

where β is the inverse coupling constant squared given by

$$\beta = \frac{4N}{\sqrt{5} g_0^2},$$

where g_0 is the bare coupling constant and $dU(b)$ is the normalized invariant Haar measure for $U(N)$ or $SU(N)$. Our action S is defined as the sum over all unoriented triangles Δ such that

$$S[U] = \sum_{\Delta} S_{\Delta} = \sum_{\Delta} \left(1 - \frac{1}{N} \text{Re Tr } U_{\Delta} \right),$$

where U_{Δ} is the parallel transporter around a triangle. A method based on a modified algorithm of Metropolis et al. [7] is used to equilibrate our lattice.

3. Code description

The program consists of five routines: the main driver routine called PROGRAM LATTICE, SUBROUTINE MONTE, SUBROUTINE RENORM, SUBROUTINE LOOPS and SUBROUTINE WILSON. SUBROUTINE MONTE implements the method of Metropolis et al. to equilibrate our simplicial lattice and hence generate an equilibrated gauge field configuration. SUBROUTINE RENORM carries out a Gram-Schmidt orthogonalization procedure on our $U(N)$ or $SU(N)$ matrices. SUBROUTINE LOOPS forms all the Wilson loops given in Fig. 3 of Ref. 4. SUBROUTINE WILSON carries out the actual matrix products needed in SUBROUTINE LOOPS.

The main routine PROGRAM LATTICE sets up the common blocks which are:

/VAR/B, ISIZE, IPOWER(5), APQ, NGROUP

where B = inverse coupling constant squared,

ISIZE = number of discrete sites in any space-time dimension,

IPOWER(N) = a pointer giving the starting point for a site in any space-time direction N,

APQ = average action per triangle,

NGROUP = degree of the group U(N) or SU(N).

/VAR1/ALAT (MF, NGROUP, NGROUP)

where ALAT = array of link matrices with MF the total number of links in the lattice.

/VAR3/IXX(10), JXX(10), KXX(5,5), LXX(5,5)

where IXX, JXX, KXX and LXX are pointers, in the I, J, K and L (x, y, z and t) directions.

In the main routine the variables ISIZE, NGROUP, B, IXX, JXX, KXX and LXX are set to the values they have throughout a given run. In addition, the links array ALAT is initialised so that all the link matrices are unit matrices. In order to effect a disordered starting lattice we set $B = 0.001$ for one call to MONTE and then set B to the value we wish to consider. We then initiate a number of calls to MONTE to equilibriate our lattice and then calls to LOOPS (which also calls MONTE to generate new equilibrated gauge field configurations) where the triangular and square Wilson loops are calculated. Periodically, in the main routine, calls are made to RENORM to renormalize our link matrices to the proper group symmetry.

SUBROUTINE MONTE is the subroutine which implements the algorithm of Metropolis et al. [7] to generate new gauge field configurations to replace ones already obtained. Initially, this subroutine is called to equilibriate the lattice from either an ordered or disordered start. On entering MONTE a table of 50 $U(N)$ or $SU(N)$ matrices is generated. This table is then sampled from when we apply our Metropolis et al. updating criterion. As we sweep systematically through the lattice, we try to update each link in turn. However, going from one link to another involves calculating pointers to link arrays and the gathering of these matrices from ALAT which involves a lot of overhead. One method of speeding up the algorithm is to sit on a given link and update it NTMAX times. This method realizes an interpolation between $NTMAX = 1$, which corresponds to the genuine Metropolis et al. algorithm and $NTMAX = \infty$, which is the heat-bath algorithm [8]. Empirically it is found that the convergence is optimized by taking $NTMAX = N^2$ for $U(N)$ or $SU(N)$.

Another method of improving the convergence of the algorithm is achieved by means of a parameter called BEFF, i.e. B effective.

This parameter has the effect of weighting the random matrices towards the identity matrices. One has to empirically study BEFF as a function of the inverse coupling constant squared β in order to obtain the optimal value.

The link to be updated is called L15 with the sites ① and ⑤ at each end. In any two-plane the two triangles with L15 as base have apices ② and ③. Thus, the two triangles are bounded by the links L15 L52 L21 and L15 L53 L31. Taking the product of link matrices attached to these links gives the action used in the Metropolis et al. algorithm. Of course, we must remember that the reverse path gives the inverse matrix.

Because of the finite word length of a computer, the link matrices tend to drift from their group symmetry due to rounding errors. The group symmetry is restored by a Gram-Schmidt orthogonalization procedure which is effected by SUBROUTINE RENORM.

In SUBROUTINE LOOPS we calculate all the Wilson loops containing up to 13 triangles or 9 squares as shown in Fig. 3 of Ref. 4. In forming the average values for these Wilson loops, we average over all orientations in our lattice. For the calculation of these Wilson loops, we have to evaluate the product of $N \times N$ matrices. For example, to evaluate the Wilson loop involving 13 triangles (see Fig. 3 of Fig. 4) will involve the product of 9 link matrices. These products are carried out in SUBROUTINE WILSON.

The program is presently set up to calculate with the $SU(N)$ gauge group. It is simple, however, to change this to $U(N)$. All we have

to do is remove the two FORTRAN statements following the COMMENT statement REMOVE THE U(1) PART OF THE MATRIX in SUBROUTINE MONTE and SUBROUTINE RENORM.

The program also calculates the adjoint of U(N) or SU(N) which is $SU(N)/Z_N$. In order to do this we use the well-known group theoretical theorem

$$\text{Tr}_A U_\Delta = |\text{Tr} U_\Delta|^2 - 1 ,$$

where $\text{Tr} U_\Delta$ is the trace of the fundamental representation of our matrices.

4. Description of the TEST RUN

In the TEST RUN we consider the SU(2) gauge group so NGROUP = 2. We work on a 6^4 lattice so ISIZE = 6 and we take 5 Monte Carlo updates per link so NTMAX = 5. The inverse coupling constant squared is chosen so that $B = 2.2$ and a disordered starting lattice is used. The program initially carries out 16 sweeps through the lattice to bring it to equilibrium. We then average over the next 26 sweeps through the lattice taking every second sweep in order to eliminate correlations between events. The TEST RUN OUTPUT shows the results for every sweep through the lattice along with the final averages. The test run took 6 hrs and 17 mins on the VAX 11/780.

Acknowledgements

We would like to thank Dr. M. Creutz for his constant encouragement and advice and the University of London Computer Centre for the granting of time on their CDC 7600 where part of this calculation was undertaken and the Royal Holloway College Computer Centre for the granting of time on their VAX 11/780 where another part of this calculation was undertaken.

References

- [1] K.G. Wilson, Phys. Rev. D10, 2455 (1974).
- [2] N.H. Christ, R. Friedberg and T.D. Lee, Nucl. Phys. B202, 89 (1982).
- [3] W. Celmaster, Phys. Rev. D26, 2955 (1982).
- [4] J.-M. Drouffe and K.J.M. Moriarty, Nucl. Phys. B220 (FS8), 253 (1983).
- [5] R.W.B. Ardill and K.J.M. Moriarty, Comput. Phys. Commun. 24, 989 (1981).
- [6] R.W.B. Ardill, K.J.M. Moriarty and M. Creutz, Comput. Phys. Commun. 29, 97 (1983).
- [7] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller and E. Teller, J. Chem. Phys. 21, 1087 (1953).
- [8] C.-P. Yang, Proc. of Symp. in Appl. Math., Vol. XV (Amer. Math. Soc., Providence, Rhode Island, 1963), p351.

***** TEST RUN OUTPUT *****

```

ITERATION(S)      1
  ISIZE= 6      GROUP=SU(2)  B=  C.001  AV.PQ.=1.00C360
ITERATION(S)      4
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.749116
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.612710
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.544640
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=C.506382
ITERATION(S)      4
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=C.48C711
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.459128
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.444813
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.436835
RENORMALIZED GROUP ELEMENTS
ITERATION(S)      4
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.428361
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.426008
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.42C376
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.417004
ITERATION(S)      4
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=C.413368
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=C.408212
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.404315
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.402461
RENORMALIZED GROUP ELEMENTS
ITERATION(S)      2
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.401460
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.396554
TYPE      FUNDAMENTAL      ADJOINT      DETERMINANT
1      0.6040      0.2762      1.0000
2      0.1452      0.0058      1.0000
3      0.0109      0.0019      1.0000
4      0.0576      0.0016      1.0000
5      0.0011      -0.0002      1.0000
6      0.2487      0.0341      1.0000
7      0.0681      0.0044      1.0000
8      0.0180      0.0018      1.0000
9      0.0084      0.0015      1.0000
10     -0.0002      0.0016      1.0000
11     -0.0072      -0.0005      1.0000
12      0.3772      0.0860      1.0000
13     -0.0040      0.0019      1.0000
ITERATION(S)      2
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.394735
  ISIZE= 6      GROUP=SU(2)  B=  2.2C0  AV.PQ.=0.392177
TYPE      FUNDAMENTAL      ADJOINT      DETERMINANT
1      0.6108      0.2848      1.0000
2      0.1531      0.0114      1.0000
3      0.0121      0.0029      1.0000
4      0.0622      -0.0007      1.0000
5      -0.0037      0.0010      1.0000
6      0.2601      0.0354      1.0000
7      0.0816      0.0013      1.0000
8      0.0240      0.0003      1.0000
9      0.0077      0.0000      1.0000
10     0.0011      -0.0002      1.0000
11     -0.0008      -0.0029      1.0000
12      0.3869      0.0914      1.0000
13     -0.0033      0.0004      1.0000

```

```

ITERATION(S)      2
  ISIZE= 6      GROUP=SU(2)  E=  2.200  AV.PQ.=0.387610
  ISIZE= 6      GROUP=SU(2)  E=  2.200  AV.PQ.=0.387702
  TYPE      FUNDAMENTAL      ADJOINT      DETERMINANT
1      0.6133      0.2866      1.0000
2      0.1630      0.0115      1.0000
3      0.0176      -0.0019      1.0000
4      0.0666      -0.0011      1.0000
5      0.0053      0.0008      1.0000
6      0.2690      0.0404      1.0000
7      0.0854      0.0026      1.0000
8      0.0285      0.0040      1.0000
9      0.0040      -0.0017      1.0000
10     0.0045      -0.0009      1.0000
11     0.0016      -0.0012      1.0000
12     0.3929      0.0911      1.0000
13     -0.0009      -0.0006      1.0000

ITERATION(S)      2
  ISIZE= 6      GROUP=SU(2)  E=  2.200  AV.PQ.=0.388911
  ISIZE= 6      GROUP=SU(2)  E=  2.200  AV.PQ.=0.389265
  TYPE      FUNDAMENTAL      ADJOINT      DETERMINANT
1      0.6117      0.2827      1.0000
2      0.1664      0.0138      1.0000
3      0.0200      -0.0018      1.0000
4      0.0697      -0.0020      1.0000
5      0.0037      0.0001      1.0000
6      0.2669      0.0365      1.0000
7      0.0851      0.0005      1.0000
8      0.0246      0.0001      1.0000
9      0.0150      0.0049      1.0000
10     -0.0002      0.0001      1.0000
11     -0.0024      0.0033      1.0000
12     0.3915      0.0899      1.0000
13     0.0028      0.0016      1.0000

ITERATION(S)      2
  ISIZE= 6      GROUP=SU(2)  E=  2.200  AV.PQ.=0.391073
  ISIZE= 6      GROUP=SU(2)  E=  2.200  AV.PQ.=0.388933
  TYPE      FUNDAMENTAL      ADJOINT      DETERMINANT
1      0.6124      0.2838      1.0000
2      0.1623      0.0102      1.0000
3      0.0228      -0.0014      1.0000
4      0.0708      0.0024      1.0000
5      0.0024      -0.0001      1.0000
6      0.2649      0.0376      1.0000
7      0.0836      -0.0007      1.0000
8      0.0261      0.0013      1.0000
9      0.0041      -0.0004      1.0000
10     -0.0010      -0.0021      1.0000
11     -0.0015      0.0001      1.0000
12     0.3927      0.0926      1.0000
13     0.0002      -0.0004      1.0000

ITERATION(S)      2
  ISIZE= 6      GROUP=SU(2)  E=  2.200  AV.PQ.=0.387516
  ISIZE= 6      GROUP=SU(2)  E=  2.200  AV.PQ.=0.385572
  TYPE      FUNDAMENTAL      ADJOINT      DETERMINANT
1      0.6143      0.2870      1.0000
2      0.1698      0.0120      1.0000
3      0.0214      0.0008      1.0000
4      0.0756      -0.0031      1.0000
5      0.0031      0.0009      1.0000
6      0.2759      0.0367      1.0000

```