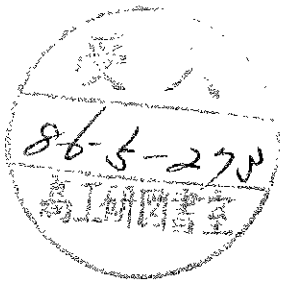
DEUTSCHES ELEKTRONEN-SYNCHROTRON DESY

DESY 85-141
December 1985




# UPDATING FERMIONS WITH THE LANCZOS METHOD

by

I.M. Barbour, N.-E. Behilil, P.E. Gibbs, M. Rafiq

*Dept. of Natural Philosophy, University of Glasgow*

K.J.M. Moriarty

*Institute for Computational Studies,*
*Dept. of Math., Stat. & Comp., Dalhousie Univ., Halifax*

G. Schierholz

*Deutsches Elektronen-Synchrotron DESY, Hamburg*
and
*Institut f. Theor. Physik, Universität Kiel*

ISSN 0418-9833

To be sure that your preprints are promptly included in the
HIGH ENERGY PHYSICS INDEX ,
send them to the following address ( if possible by air mail ) :

DESY
Bibliothek
Notkestrasse 85
2 Hamburg 52
Germany

DESY 85-141
December 1985

ISSN 0418-9833

# Updating Fermions with the Lanczos Method

I.M. Barbour, N.-E. Behilil, P.E. Gibbs, M. Rafiq
Department of Natural Philosophy, University of Glasgow

K.J.M. Moriarty
Institute for Computational Studies
Department of Mathematics, Statistics and Computing
Dalhousie University, Halifax

G. Schierholz
Institut für Theoretische Physik der Universität Kiel

and

Deutsches Elektronen-Synchrotron DESY, Hamburg

## Abstract

The Lanczos method is proposed for the Monte Carlo simulation of the QCD (lattice) vacuum including dynamical fermion loops. It appears that an exact fermion update is feasible on medium sized lattices with today's vector processors.



## 1. Introduction

A long-standing problem in lattice gauge theories is the inclusion of the effect of dynamic fermion fields into Monte Carlo calculations. The problem is essentially numerical, since it requires the calculation of the ratio of two very large determinants each time a gauge link variable is changed.

The pseudo-fermion method [1] has led to useful results on small lattices [2] but has poor convergence on reasonable sized lattices and at realistically small fermion masses. Furthermore, the method is not exact and amounts to a (small?) violation of the detailed balance. An alternative approach is the microcanonical technique [3]. This has some advantages but can only be used when the number of fermion species is a multiple of four. Moreover, it does not fulfil the ergodicity requirement, and the coupling constant has to be computed via the Monte Carlo simulation itself. A further alternative is the Langevin method [4], which is not exact either and may as well exhibit systematic errors once it has been tested sufficiently well.

Since we believe that topology and the accompanying fermionic zero modes play an important role in the dynamics of the QCD vacuum, we find an exact updating procedure indispensible. There is an exact method, which reduces the fermion matrix to a smaller, but denser one on spatial planes only [5]. This works well at small mass, since there are no convergence problems, but is only feasible on lattices with a very small volume,

although the time dimension may be large. In this paper we shall present a new method based on the Lanczos algorithm for inverting large, sparse matrices, which we have found to be feasible on medium sized lattices with present day computing power.

We have explained in a previous paper how the Lanczos algorithm can be used - in a similar manner to the conjugate gradient algorithm - for inverting matrices row by row [6]. If we apply this to fermionic updating, then we have a number of advantages. Firstly, the convergence of the Lanczos algorithm is superior to that of the conjugate gradient algorithm at realistically small mass. It will even converge at zero mass. This could mean a saving in time by a factor of 2 to 3. A further advantage is that, if we wish to include a number of different species of fermions at different masses, the Lanczos algorithm can simultaneously invert for all the masses at once with only a minor increase in computation.

The main advance in updating - which we will describe here - is, however, the use of rank annihilation to update a block of an universe exactly. For example, a hypercube contains 16 sites and 32 links, and for SU(3) a 48 x 48 block of the inverse is sufficient to update any link in a hypercube. Rank annihilation then allows us to update the block to give the inverse for the new configuration without any further inversion. This means that all the links in the hypercube can be updated, one at a time, as many times as is desired without much more than the computation of 48 rows of the inverse . The time for one sweep is then reduced by a factor of 4, and, in addition, the number of sweeps for thermalisation



may be reduced, since hypercubes (or larger objects) are brought close to equilibrium at each sweep. This method could be applied equally well to the conjugate gradient algorithm, but we can use block Lanczos to invert 24 rows (or more) simultaneously with litte increase in the amount of computation. We will describe this method here for the first time. Combining these ideas gives an overall time saving of a factor of 1-2 orders of magnitude depending on the size of the block, the quark mass and the coupling $\beta$ compared with single link updating.

## 2. Updating the fermion matrix

In all our computation we use Kogut-Susskind fermions. The fermion species doubling problem is concealed by taking the fourth root of the determinant.

The fermionic action is

$$S_F = \bar{\psi}\,(M+2m)\,\psi, \qquad (1)$$

where m is the fermion mass in lattice units and M is the anti-hermitean fermion matrix

$$\bar{\psi} M \psi = \sum_{n,\mu} \bar{\psi}_n U_{n,n+\hat{\mu}} (-1)^{n_1+\cdots+n_{\mu-1}} \psi_{n+\hat{\mu}} - h.c., \qquad (2)$$

where $\psi_n$ is a single component, colour triplet Grassmann variable sited at $n = (n_1,n_2,n_3,n_4)$ and $U_{n,u+\hat{\mu}}$ is the 3x3 SU(3) link matrix joining sites n to $n+\hat{\mu}$, $\hat{\mu}$ being a displacement vector of unit length (in lattice units) in direction $\mu$. Hence, on a lattice of size $L_s^3 \cdot L_t$, M is a large, sparse anti-hermitean matrix of size $3L_s^3 \cdot L_t$ square but with only 24

non-zero elements in each row.

In order to perform Metropolis updating of the gauge field including the effects of dynamic fermions, we need to calculate the ratio of determinants of M+2m, when a change is made to one link:

$$R = \frac{\det(H+\Delta H)}{\det(H)} = \det(1+H^{-1}\Delta H), \quad H=\gamma_5(M+2m), \tag{3}$$

where $\Delta H$ is the change in the fermion matrix, when one link matrix is updated. It is non-zero in the 6x6 block at the intersection of the 6 rows and 6 columns corresponding to the two end points of the link. Consequently, the only elements of $H^{-1}$ which contribute are those in the same 6x6 block as $\Delta H$. If we write $(\overline{\Delta H})$ and $(\overline{H^{-1}})$ for these blocks, then R is given by the 6x6 determinant

$$R = \det(1+(\overline{H^{-1}})(\overline{\Delta H})). \tag{4}$$

The Lanczos or the conjugate gradient algorithm can be used to calculate 6 columns of $H^{-1}$. This is sufficient to update the same link as many times as desired, since the ratio of determinants for two different changes is

$$R = \frac{\det(H+\Delta_1 H)}{\det(H+\Delta_2 H)} = \frac{\det(1+H^{-1}\Delta_1 H)}{\det(1+H^{-1}\Delta_2 H)} . \tag{5}$$

This idea can be extended to a number of links at once. For example, consider all 32 links of one hypercube. To calculate the ratio of determinants for any change to these links, we need the 48 x 48 block



corresponding to the 16 sites of the hypercube. In order to avoid calculation of 48 x 48 determinants, we can update this 48 x 48 block by rank annihilation [7] as follows. Consider a change to one link of the hypercube. This makes a change $\Delta H$ in the fermion matrix with 18 non-zero elements, which we separate into 18 consecutive changes, each to just one element,

$$\Delta H = \Delta_1 H + \Delta_2 H + \cdots + \Delta_{18} H , \tag{6}$$

so that we can write

$$\Delta_i H = a\, u v^{+}, \tag{7}$$

where a is the change to the element and u, v are unit column vectors, which are zero in all elements but one. Then, if $H^{-1} = Z$,

$$\begin{aligned}(H+a\,uv^{+})^{-1} &= Z - Z a\,uv^{+}Z + Z a\,uv^{+}Z a\,uv^{+}Z - \cdots \\ &= Z - a(Zu)(v^{+}Z)(1-av^{+}Zu+a^{2}(v^{+}Zu)^{2}-\cdots) \\ &= Z - \frac{a(Zu)(v^{+}Z)}{1+av^{+}Zu} .\end{aligned} \tag{8}$$

The convergence of the series is not relevant, since the final result can be checked by back substitution. It can easily be seen that this formula can be applied to update the 48 x 48 block of Z without knowing the rest of its elements. Numerical tests on small to medium sized lattices have shown that the link matrices of the hypercube can practically be changed as many times as desired and in any order without any significant

rounding errors accumulating due to updating the block by rank annihilation.

To summarize, the Metropolis algorithm is carried out as follows. To cover all the links in one sweep, we need to consider 1/8 of all possible hypercubes which touch each other at corners only so that they have no links in common. We take each of these hypercubes in turn, either in sequence or at random, and calculate the appropriate 48 x 48 block of inverse required to update its links. This could be done by the conjugate gradient algorithm, but we shall see how block Lanczos can be used more efficiently with a substantial saving in computing time. We then take each of the 32 links in turn in any order, extract the appropriate 6 x 6 block from the 48 x 48 block and apply Metropolis updating to the link a large number of times, which requires the calculation of only 6 x 6 determinants and matrix multiplications each time. Before proceeding to the next link in the hypercube, we update the 48 x 48 block by rank annihilation for the overall change to the link. It proves worthwhile to go round the whole hypercube a few times until it is close to equilibrium within itself before proceeding to a new hypercube. This brings the configuration into equilibrium $\gtrsim$ 2-3 times faster.

## 3. The Lanczos algorithm

The Lanczos algorithm has already been used to calculate eigenvalues of the fermion matrix [8] and invert it row by row [9], and this has been applied to chiral condensate [8,10] and propagator calculations [9] as well as the investigation of the topological structure of SU(2) gauge theory [11]. We shall briefly review this before describing the block



Lanczos algorithm.

The hermitean Lanczos algorithm aims to tridiagonalize a hermitean matrix H by a unitary transformation X:

$$HX = XT \,, \quad T = \begin{pmatrix} \alpha_1 & \beta_1 & 0 & \cdots \\ \beta_1 & \alpha_2 & \beta_2 & 0 \cdots \\ 0 & \beta_2 & \alpha_3 & \beta_3 \; 0 \cdots \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} . \tag{9}$$

Let us denote the columns of X by $x_i$, i.e. $X = (x_1, x_2, \ldots, x_N)$. These are called the Lanczos vectors. Then

$$x_i^\dagger x_j = \delta_{ij} \tag{10}$$

and

$$\begin{aligned} Hx_1 &= \alpha_1 x_1 + \beta_1 x_2 \,, \\ Hx_i &= \beta_{i-1} x_{i-1} + \alpha_i x_i + \beta_i x_{i+1} \,, \quad i \geq 2 . \end{aligned} \tag{11}$$

Given an initial unit Lanczos vector $x_1$, we can proceed iteratively to calculate $x_i$:

$$\begin{aligned} \alpha_1 &= x_1^\dagger H x_1 \,, \\ \beta_1 &= | Hx_1 - \alpha_1 x_1 | \,, \\ x_2 &= \frac{1}{\beta_1} (Hx_1 - \alpha_1 x_1) \,, \\ \alpha_2 &= x_2^\dagger H x_2 \,, \\ &\text{etc.} \end{aligned} \tag{12}$$

In theory this guarantees the orthonormality of the Lanczos vectors,

and the algorithm should end with $\beta_N = 0$ after N ($= 3L_s^3 \cdot L_t$) iterations. However, this fails since rounding errors lead to loss of orthogonality between Lanczos vectors separated by a large number of iterations. Fortunately all is not lost, since the eigenvalues of T are found remarkably to converge to those of H together with ghosts and spurious eigenvalues, which can be removed by various means [6]. In this way it is possible to calculate all eigenvalues for a large, sparse matrix with great efficiency and accuracy, and this was the initial motivation for the Lanczos algorithm.

When the Lanczos algorithm is applied to the fermion matrix for Kogut-Susskind fermions, there is a useful simplification due to the even-odd block structure of M. An even site of the lattice is one, whose component indices add up to an even number. The matrix M connects even sites to odd sites only and vice versa. In matrix notation this means that H has the following block structure:

$$H = \begin{pmatrix} 2im & \hat{M} \\ \hat{M}^+ & 2im \end{pmatrix} . \tag{13}$$

If we put m = 0 and apply the Lanczos algorithm to H taking the first Lanczos vector to be zero on all odd sites,

$$x_1 = \begin{pmatrix} \hat{x}_1 \\ 0 \end{pmatrix} \tag{14}$$

then we find that all $\alpha_i = 0$ and each odd Lanczos vector takes the form



$$x_{2i+1} = \begin{pmatrix} \hat{x}_{2i+1} \\ 0 \end{pmatrix} , \tag{15}$$

and for the rest

$$x_{2i} = \begin{pmatrix} 0 \\ \hat{x}_{2i} \end{pmatrix} . \tag{16}$$

The Lanczos equations then reduce to

$$\begin{aligned} \hat{M}^+ \hat{x}_1 &= \beta_1 \hat{x}_2 , \\ \hat{M} \hat{x}_{2i} &= \beta_{2i-1} \hat{x}_{2i-1} + \beta_{2i+1} \hat{x}_{2i+1} , \quad i \geq 1 , \\ \hat{M}^+ \hat{x}_{2i+1} &= \beta_{2i} \hat{x}_{2i} + \beta_{2i+1} \hat{x}_{2i+1} , \quad i \geq 1 , \end{aligned} \tag{17}$$

with the even vectors being mutually orthonormal and similarly for the odd. The immediate advantage of this is that we have halved the amount of computation, since there is no need to compute $\alpha_i$, and each Lanczos vector is half-zero. There are also savings in space, and in fact we need only store two of these half vectors between iterations.

## 4. Inversion by the Lanczos algorithm

Let us consider now how we may use the Lanczos algorithm to invert the fermion matrix. We have

$$H x_1 = \beta_1 x_2 \,,$$
$$H x_i = \beta_{i-1} x_{i-1} + 2im\, x_i + \beta_i x_{i+1} \,, \quad i \geq 2. \tag{18}$$

The betas and Lanczos vectors are independent of the mass m. That is why we can simultaneously invert the matrix at a number of different masses without increased computation.

We shall use these Lanczos equations iteratively to calculate $H^{-1}x_1$ as a series

$$H^{-1}x_1 = c_1 x_1 + c_2 x_2 + \ldots \tag{19}$$

The details of this were given in ref. 6), and we do not repeat it here, since it is complicated algebraically. However, as an illustration we can do the much simpler case m = 0. We need only every alternate Lanczos equation starting with the second:

$$H^{-1}x_1 = \frac{1}{\beta_1} x_2 - \frac{\beta_2}{\beta_1} H^{-1} x_3 \,. \tag{20}$$

We use the other Lanczos equations in sequence to eliminate the remainder term. This gives

$$H^{-1}x_1 = \frac{1}{\beta_1} x_2 - \frac{\beta_2}{\beta_3\beta_5} x_4 + \frac{\beta_2\beta_4}{\beta_3\beta_5\beta_7} x_6 - \ldots \tag{21}$$

At first sight it seems highly unlikely that this will converge, since the betas typically fluctuate randomly about some constant value. However, if we are brave enough to persist, we find that although the series proceeds for many iterations without any sign of convergence, we eventually reach a point where there is a rapid convergence of the series down



to about machine precision. This point can be identified with the point, where the smallest eigenvalues of the tridiagonal form are converging to the true eigenvalues of H. It is remarkable that such good convergence is possible for such a highly singular matrix, and there is certainly no similar convergence for the conjugate gradient algorithm at zero mass.

At larger masses the convergence of the Lanczos algorithm is more or less identical to that of the conjugate gradient algorithm, and a similar amount of calculation is required. As the mass becomes smaller, the convergence rate decreases in both cases, so that the number of iterations required is inversely proportional to m. When the mass becomes very (i.e. realistically) small, so many iterations are required that we reach the point of rapid convergence for the Lanczos algorithm, while the conjugate gradient algorithm continues to require more and more iterations.

## 5. Block Lanczos

The Lanczos algorithm can be generalized so that the alphas and betas become small LxL matrices. The alphas are hermitean, and the betas can be chosen to be triangular [12], so that H is transformed into a band matrix of width 2L+1. The Lanczos vectors are NxL arrays

$$H x_1 = x_1 \alpha_1 + x_2 \beta_1 \,,$$
$$H x_i = x_{i-1} \beta_{i-1} + x_i \alpha_i + x_{i+1} \beta_i \,, \quad i \geq 2. \tag{22}$$

The algorithm proceeds in a way analogous to the L=1 case. For the fermion matrix we can again have $\alpha_i = 0$ if the initial Lanczos vector is chosen

to be zero on odd sites. The algorithm is then:

$$\beta_1^+ \beta_1 = (Hx_1)^+ (Hx_1) , \tag{23}$$

which we solve for $\beta_1$, as a lower triangular matrix to compute

$$x_2 = H x_1 \beta_1^{-1} \tag{24}$$

and so on:

$$\begin{aligned} \beta_i^+ \beta_i &= U^+ U , \\ x_{i+1} &= U \beta_i^{-1} , \end{aligned} \tag{25}$$

where

$$U = H x_i - x_{i-1} \beta_{i-1}^+ , \quad i \geq 2 . \tag{26}$$

We can now apply block Lanczos to inversion to calculate L rows of the inverse at one time. The reason that block Lanczos is more efficient lies in the fact that one is not transforming to a tridiagonal form but rather only to a block tridiagonal form, which is less constraining. The optimum block size is a function of the machine architecture, since the algorithm involves the inversion of a non-hermitean matrix.

We shall not describe in detail the derivation of the complete algorithm, since it is merely a case of generalising the L=1 case [6] replacing all variables by LxL matrices. The resulting recurrence relations are



$$\begin{aligned}
A_1 &= 1 , \\
B_1 &= 0 , \\
\gamma_1 &= 0 , \\
t_1 &= 1 , \\
V_1 &= 0 , \\
U_1 &= -x_1 \beta_1^{-1} , \\
A_{2k} &= A_{2k-1} + m^2 (\beta_{2k-1}^{-1})^+ B_{2k-1} , \\
B_{2k} &= -\beta_{2k} (\beta_{2k-1}^{-1})^+ B_{2k-1} , \\
\gamma_{2k} &= \gamma_{2k-1} - A_{2k}^{-1} \beta_{2k-1}^{-1} t_{2k-1} , \\
t_{2k} &= -\beta_{2k} A_{2k-1} A_{2k}^{-1} \beta_{2k-1}^{-1} t_{2k-1} , \\
U_{2k} &= U_{2k-1} + im\, x_{2k} \beta_{2k-1}^{-1} B_{2k-1} , \\
V_{2k} &= V_{2k-1} + x_{2k} \beta_{2k-1}^{-1} t_{2k-1} + im\, U_{2k} A_{2k}^{-1} \beta_{2k-1}^{-1} t_{2k-1} , \\
A_{2k+1} &= \beta_{2k+1} (\beta_{2k}^{-1})^+ A_{2k} , \\
B_{2k+1} &= B_{2k} - (\beta_{2k}^{-1})^+ A_{2k} , \\
\gamma_{2k+1} &= \gamma_{2k} , \\
t_{2k+1} &= t_{2k} , \\
U_{2k+1} &= U_{2k} + x_{2k+1} (\beta_{2k}^{-1})^+ A_{2k} , \\
V_{2k+1} &= V_{2k} , \\
(1 - (\beta_1^{-1}) m^2 \gamma_{2k+1})^{-1} V_{2k+1} &\rightarrow H^{-1} x_1 .
\end{aligned} \tag{27}$$

The coefficients A,B,y and t are all LxL matrices, and U and V are NxL arrays. However, if only a small part of the inverse is required, as is the case for fermion updating, it is not necessary to compute the whole of U and V but only some KxL block of them.

If we are updating hypercube by hypercube, this algorithm can be applied to calculate the 48x48 block of $H^{-1}$ required as follows. We take L=24 and calculate the block in two 48x24 pieces in two separate inversions, one to cover the odd sites and another for the even sites of the hypercube.

## 6. Outlook

The block Lanczos method has been successfully applied to fermion updating on small lattices ($\leq 8^4$) for gauge group SU(2) [13], and we were able to obtain a time saving of a factor of 10 over single row inversion for a block of one hypercube. We believe, however, that this is not optimal yet, but that it may be more efficient to take blocks of 2 or 3 hypercubes at a time.

In any case, it appears that the block Lanczos method is capable of simulating the vacuum of gauge theories including the effect of fermion loops on medium sized lattices with today's vector processors. We hope to be able to report on the outcome of such a calculation in the near future.

## Acknowledgement

We are grateful to K. Göke, F. Hossfeld and J. Speth for granting us time on the Cray X-MP at the KFA in Jülich, on which the Lanczos method has been tested.



## References

1) F. Fucito, E. Marinari, G. Parisi, C. Rebbi: Nucl. Phys. B180, 360 (1981);
D.J. Scalapino, R.L. Sugar: Phys. Rev. Lett. 46, 519 (1981)

2) F. Fucito, S. Solomon: Caltech preprint CALT-68-1259 (1985)

3) J. Polonyi, H.W. Wyld: Phys. Rev. Lett. 51, 2257 (1983)

4) G.G. Batrouni, G.R. Katz, A.S. Kronfeld, G.P. Lepage, B. Svetitsky, K.G. Wilson: Cornell preprint CNLS-85/651 (1985);
A. Ukawa, M. Fukugita: Phys. Rev. Lett. 55, 1854 (1985)

5) U. Wolff: Phys. Rev. D30, 2236 (1984)

6) I.M. Barbour, N.-E. Behilil, P.E. Gibbs, G. Schierholz, M. Teper: DESY preprint 84-087 (1984), in Lecture Notes in Physics, "The Recursion Method and its Applications" (Springer, Berlin, Heidelberg, New York, Tokyo, 1985)

7) A. Ralston, H.S. Wilf: Mathematical Methods for Digital Computers (Wiley, New York, 1960-67)

8) I.M. Barbour, P.E. Gibbs, J. Gilchrist, G. Schierholz, H. Schneider, M. Teper: Phys. Lett. 136B, 80 (1984)

9) I.M. Barbour, P.E. Gibbs, G. Schierholz: unpublished

10) I.M. Barbour, K. Bowler, P.E. Gibbs, D. Roweth: Phys. Lett. 158B, 61 (1985)

11) E.M. Ilgenfritz, M.L. Laursen, M. Müller-Preussker, G. Schierholz, H. Schiller: DESY preprint 85-108 (1985), to be published in Nucl. Phys. B

12) D.S. Scott: in Sparse Matrices and Their Uses, ed. I.S. Duff (Academic Press, London, New York, Toronto, Sydney, San Francisco, 1981)

13) I.M. Barbour, P.E. Gibbs, G. Schierholz: to be published