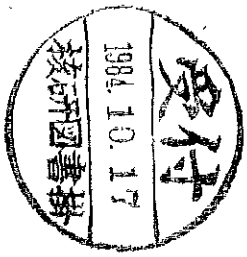
DEUTSCHES ELEKTRONEN-SYNCHROTRON DESY

DESY 84-087
LAPP-TH 118
September 1984

THE LANCZOS METHOD IN LATTICE GAUGE THEORIES



by

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

Dept. of Natural Philosophy, University of Glasgow

G. Schierholz

Deutsches Elektronen-Synchrotron DESY, Hamburg

M. Teper

L.A.P.P., Annecy

ISSN 0418-9833

NOTKESTRASSE 85 · 2 HAMBURG 52

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 84-087
LAPP-TH 118
September 1984

ISSN 0418-9833

The Lanczos Method in Lattice Gauge Theories

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

G. Schierholz
Deutsches Elektronen-Synchrotron DESY, Hamburg

M. Teper
L.A.P.P., Annecy

Abstract

We present a modified version of the Lanczos algorithm as a computational method for tridiagonalising large sparse matrices, which avoids the requirement for large amounts of storage space. It can be applied as a first step in calculating eigenvalues and eigenvectors or for obtaining the inverse of a matrix row by row. Here we describe the method and apply it to various problems in lattice gauge theories. We have found it to have excellent convergence properties. In particular it enables us to do lattice calculations at small and even zero quark mass.

To be published in Lecture Notes in Physics, " The Recursion Method and its Applications ", Springer-Verlag (Berlin, Heidelberg, New York, Tokyo, 1985)

1. Introduction

The lattice is a four dimensional cubic array of $n_x^3 n_t$ sites, which represents a discrete approximation to a volume of space-time. The gauge fields of QCD are represented by a configuration of 3 x 3 special unitary matrices (SU(3)), which are located on the links of the lattice joining neighbouring sites. The fermion field for the quarks is a colour triplet of anticommuting variables, which we place on the lattice sites.

We are supplied with configurations of the gauge fields generated by Monte Carlo methods. The dynamics of the quarks in these background fields is derived from the action

$$S_F = -\bar{\psi}(M+2ma)\psi, \tag{1}$$

where ψ is the fermion field, m is the quark mass, a is the physical distance between lattice sites and M is the fermion matrix - an operator depending on the gauge configuration, which can be represented by an NxN sparse matrix ($N = 3n_x^3 n_t$). There are various possible forms of M that can be used. We use Kogut-Susskind fermions, so that M takes the following form

$$\bar{\psi}M\psi = \sum_{n,\mu} \bar{\psi}_n U_{n,n+\hat{\mu}} f_{n,n+\hat{\mu}} \psi_{n+\hat{\mu}} \tag{2}$$

- hermitian conjugate,

where ψ_n is the single component colour triplet sited at $n = (x_1,x_2,x_3,t)$, μ indexes the 4 directions in space-time, $\hat{\mu}$ is a displacement vector of length a in direction μ, $U_{n,n+\hat{\mu}}$ is the 3x3 gauge matrix joining sites n to $n+\hat{\mu}$ and the sign factors are

$$f_{n,n+\hat{\mu}} = \begin{cases} 1 & , \mu = x_1 \\ (-1)^{x_1} & , \mu = x_2 \\ (-1)^{x_1+x_2} & , \mu = x_3 \\ (-1)^{x_1+x_2+x_3} & , \mu = t \end{cases} \qquad (3)$$

For the gauge field action we assume periodic boundary conditions, while for the fermionic action we shall use antiperiodic boundary conditions. This is accomplished by including an extra factor -1 in $f_{n,n+\hat{\mu}}$ between the first and last μ-plane.

Calculation of physical quantities such as the chiral condensate and the hadron spectrum require the calculation of the inverse of M + 2ma. Various methods have been used in the past to do this. Gaussian elimination cannot be used because of the problem of "fill in". I.e. during the calculation the zero elements in the sparse matrix M + 2ma are filled in, so that vast amounts of storage and computation are required. This can be kept to a minimum by choosing suitable pivots but could not be used on lattices any bigger than $4^3 8$. Hopping parameter expansions, Gauss-Seidel iteration and pseudo-boson methods have been tried with a certain amount of success. The drawback of these methods is that convergence is very slow at small quark masses. The conjugate gradient method for inverting matrices has been applied successfully [1,2] but again its convergence at low mass could be improved. The Lanczos method as we present it here has better convergence at small mass. It also has the advantage that for one run of the Lanczos algorithm calculations can be done at many different masses in contrast to the conjugate gradient algorithm, where each different mass requires a separate calculation.

2. The Hermitian Lanczos Algorithm

First we will describe how to tridiagonalise a hermitian matrix H (e.g. H = iM) of size NxN. We seek a unitary transformation:

$$X^+ H X = T \quad , \quad X^+ X = 1 \quad , \qquad (4)$$

where T is tridiagonal, real and symmetric:

$$T = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \beta_2 & \alpha_3 & \ddots & \\ & & \ddots & & \beta_{N-1} \\ & & & \beta_{N-1} & \alpha_N \end{pmatrix} . \qquad (5)$$

Write X as a series of column vectors

$$X = (x_1, x_2, \ldots, x_N).$$

These are the Lanczos vectors and they are orthonormal

$$x_i^+ x_j = \delta_{ij} . \qquad (6)$$

$$HX = XT \iff \begin{aligned} Hx_1 &= \alpha_1 x_1 + \beta_1 x_2 \\ Hx_i &= \beta_{i-1} x_{i-1} + \alpha_i x_i + \beta_i x_{i+1} \quad , \quad 2 \le i \le N-1 \\ Hx_N &= \beta_{N-1} x_{N-1} + \alpha_N x_N \end{aligned} \qquad (7)$$

These are the Lanczos equations, which can be used recursively to calculate all the α_i , β_i and x_i. Chose x_1 to be any unit vector. Take the scalar product of x_1 with the first Lanczos equation and use the orthonormality of the Lanczos vectors to obtain α_1 :

$$\alpha_1 = x_1^+ H x_1 . \qquad (8)$$

α_1 is assured to be real because H is hermitian. Next calculate

$$\beta_1 x_2 = H x_1 - \alpha_1 x_1 \tag{9}$$

and use $x_2^+ x_2 = 1$ to obtain β_1 and x_2 (we can take either sign for β_1). Continue in a similar way with all the other equations in turn:

$$\begin{aligned} \alpha_i &= x_i^+ H x_i \\ \beta_i x_{i+1} &= H x_i - \beta_{i-1} x_{i-1} - \alpha_i x_i \end{aligned} \tag{10}$$

This defines the Lanczos algorithm for calculating all α_i , β_i and x_i. We can easily check that the hermitian nature of H ensures that all the Lanczos vectors are then orthogonal if the calculation can be done without rounding errors. E.g.

$$\begin{aligned} x_1^+ x_2 &= \frac{1}{\beta_1} x_1^+ (H x_1 - \alpha_1 x_1) \\ &= \frac{1}{\beta_1} (\alpha_1 - \alpha_1) = 0 \end{aligned} \tag{11}$$

Furthermore, once we have calculated

$$\alpha_N = x_N^+ H x_N \tag{12}$$

the calculation is then complete, and the last equation is automatically satisfied because we can show that

$$u = H x_N - \beta_{N-1} x_{N-1} - \alpha_N x_N \tag{13}$$

is orthogonal to all the Lanczos vectors and must therefore be zero. In fact a good check on the accuracy of the calculation is that

$$\beta_N = |u| = 0 . \tag{14}$$

Apart from rounding errors there is only one thing that can cause the algorithm to fail. That is if some β is zero, then we will have a division

by zero. This will happen if the first Lanczos vector x_1 was choosen to be orthogonal to some eigenvector of H, and it is inevitable if H has a degenerate eigenvalue. The solution is to chose the next x_i to be any unit vector orthogonal to all the previous ones and continue the calculation. This may be difficult to implement in practice, but since we have never encountered this situation we have ignored it.

The advantage of the Lanczos algorithm over other methods is that it does not require the matrix H to be stored in a large NxN array which is "filled in" by the calculation, even if H has a large number of zero elements. We only require storage space for about 3 Lanczos vectors and a subroutine to multiply a vector by H. If H is sparse or even a product of a few sparse matrices, then the multiplication can be done fast and with a minimum of storage space. All but the last two Lanczos vectors can be thrown away after each iteration. If they are ever needed again (e.g. if our aim is to calculate eigenvectors of H), then they can be recalculated as they are needed without resorting to large storage. In some cases it may be more effective to write them onto disk.

Before we can use the Lanczos algorithm, we must overcome the problem of rounding errors. If it is applied to large matrices, we find that $\beta_N \neq 0$ due to rounding errors. This is due to loss of orthogonality between the first few Lanczos vectors and the last ones. These errors tend to build up exponentially, so that no matter what precision is used in the calculation we soon find that after each iteration the last x_i is not orthogonal to x_1. The most straightforward way to overcome this is to reorthogonalise. When we have calculated a new Lanczos vector x_i , it can be made orthogonal to an earlier vector x_j by a projection

$$x_i \rightarrow x_i - x_j (x_j^+ x_i) . \tag{15}$$

x_i can be reorthogonalised against each previous vector in turn. Then, provided we have not lost too much orthogonality, the rounding errors will be reduced. Usually this does not need to be done after every iteration, unless there are many very close eigenvalues. Unfortunately, reorthogonalisation greatly slows down the calculation and means that all the Lanczos vectors must be stored. Usually it is necessary to use an external sequential disk file for this, and it is impractical to reorthogonalise for $N \gtrsim 1000$.

Fortunately it is possible to use the Lanczos method without reorthogonalisation, and this enables us to deal with much larger matrices. We allow the Lanczos algorithm to proceed beyond the N^{th} iteration calculating new Lanczos vectors and α's and β's until we have done $\tilde{N}$ iterations. Then

$$\begin{aligned} H x_1 &= \alpha_1 x_1 + \beta_1 x_2 \\ H x_i &= \beta_{i-1} x_{i-1} + \alpha_i x_i + \beta_i x_{i+1} \quad , \; 2 \le i \le \tilde{N} \end{aligned} \tag{16}$$

The α's and β's now form an $\tilde{N}\times\tilde{N}$ tridiagonal form $\tilde{T}$ with $\tilde{N}$ eigenvalues $\tilde{\lambda}_r$, from which we can sort out the eigenvalues λ_i of H. Write the Lanczos equations as

$$H\tilde{X} = \tilde{X}\tilde{T} + R \; . \tag{17}$$

$\tilde{X}$ is the $N\times\tilde{N}$ matrix formed with the Lanczos vectors as its columns, and R is the remainder due to rounding errors and stopping at $\tilde{N}$ iterations. The first $\tilde{N}-1$ columns of R will be small errors, but the last will be approximately $\beta_i x_{i+1}$. Suppose an eigenvalue $\tilde{\lambda}_r$ of $\tilde{T}$ has eigenvector $\tilde{e}_r$:

$$\tilde{T}\tilde{e}_r = \tilde{\lambda}_r \tilde{e}_r \tag{18}$$

$$H\tilde{X}\tilde{e}_r = \tilde{X}\tilde{T}\tilde{e}_r + R\tilde{e}_r \cong \tilde{\lambda}_r \tilde{X}\tilde{e}_r + \beta_{\tilde{N}} x_{\tilde{N}+1} \tilde{e}_{ri} \; . \tag{19}$$

Therefore $\tilde{\lambda}_r$ will be an eigenvalue of H with eigenvector $\tilde{X}\tilde{e}_r$ provided the last component of $\tilde{e}_r$ is very small. We have found empirically that if $\tilde{N}$ is sufficiently large, then all eigenvalues of H will converge as eigenvalues of $\tilde{T}$. But $\tilde{T}$ will also have spurious eigenvalues, which are not eigenvalues of H because $\tilde{e}_{ri}$ is large. Some eigenvalues of H may converge very fast and can be obtained from $\tilde{T}$ when $\tilde{N}$ is still much smaller than N. By the time $\tilde{N}$ is large enough for all the eigenvalues to have converged, the faster ones will appear many times as eigenvalues of $\tilde{T}$. These ghosts can be recognized because we assume H to be nondegenerate. The spurious eigenvalues of $\tilde{T}$ can also be recognized by comparing with the eigenvalue of the tridiagonal matrix $\hat{T}$ formed from the first $\tilde{N}-1$ iterations. The real eigenvalues of H will be eigenvalues of $\hat{T}$ as well as $\tilde{T}$, but $\hat{T}$ will have different spurious eigenvalues. This is because the last component of their eigenvectors are large and are therefore greatly effected by removing the last α and β.

The eigenvalues of $\tilde{T}$ can be found by the standard method of Sturm sequences. This is an algorithm which quickly tells us how many eigenvalues $\tilde{T}$ has less than a given value λ. It can be used to find the n^{th} eigenvalue of $\tilde{T}$ by a series of bisections starting from an interval known to contain all the eigenvalues. Sturm sequences can also be used to quickly determine which are the spurious eigenvalues. Once we have an eigenvalue $\tilde{\lambda}_r$ of $\tilde{T}$, we can find how many eigenvalues of $\hat{T}$ there are in a neighbourhood $[\lambda_r-\epsilon, \lambda_r+\epsilon]$ of λ_r. If there are none, then λ_r is spurious.

We have not made any general studies of how the eigenvalues converge for different matrices, but we note the following two points on convergence:

(i) In any region where the eigenvalues are relatively well separated, they will converge fastest and may appear with many ghosts before all other eigenvalues have converged. In some applications only these eigenvalues are needed, and it is only necessary to do a small number of iterations to obtain them.

(ii) If some eigenvalues are relatively small or close together, then a correspondingly high precision is needed for the α's, β's and Lanczos vectors, otherwise they will not all converge no matter how many iterations are done.

For further details we refer to ref. 3).

3. Application to $\langle\bar{\psi}\psi\rangle$ Calculations in Lattice QCD

We have applied the hermitian Lanczos algorithm to study chiral symmetry breaking in lattice QCD. For a given configuration of gauge links on a lattice of N_L sites we require to calculate the trace of the inverse of an operator $M + 2ma$, where m is the quark mass. M is antihermitian and acts on a fermion field with 3 colour components at each lattice site.

$$\langle\bar{\psi}\psi\rangle = \frac{1}{N_L}\, tr\,(M+2ma)^{-1} \tag{20}$$

We are interested in the behaviour of $\langle\bar{\psi}\psi\rangle$ as ma becomes small. M can be represented by a large square matrix of size $N = 3N_L$ squared. But since M only connects field components which are joined by a lattice link, each row or column of M has only 24 components which are nonzero. Therefore it is very sparse, and the Lanczos algorithm can be used to obtain all eigenvalues of iM. Then

$$\langle\bar{\psi}\psi\rangle = \frac{3}{N}\sum_k \frac{1}{-i\lambda_k+2ma}\,. \tag{21}$$

The lattice has an even number of sites in each direction so that M has the following block structure between odd and even sites

$$iM = \begin{pmatrix} 0 & \hat{M} \\ \hat{M}^+ & 0 \end{pmatrix}. \tag{22}$$

This implies that the eigenvalues of M come in plus and minus pairs because

$$\begin{vmatrix} \lambda & \hat{M} \\ \hat{M}^+ & \lambda \end{vmatrix} = |\lambda^2 - \hat{M}\hat{M}^+| = 0\,,$$

$$\Rightarrow \langle\bar{\psi}\psi\rangle = \frac{3}{N}\,\frac{1}{2}\sum_k \left(\frac{1}{-i\lambda_k+2ma} + \frac{1}{i\lambda_k+2ma}\right) \tag{23}$$

$$= \frac{3}{N}\sum_k \frac{2ma}{\lambda_k^2+(2ma)^2}\,.$$

For large lattices

$$\langle\bar{\psi}\psi\rangle \cong 3\int_{-\infty}^{+\infty} d\lambda\, \frac{2ma\,\rho(\lambda)}{\lambda^2+(2ma)^2}\,, \tag{24}$$

where $\rho(\lambda)$ is the normalized spectral density. As $ma \to 0$ this gives

$$\langle\bar{\psi}\psi\rangle \cong 3\pi\,\rho(0). \tag{25}$$

Therefore we only need the eigenvalues close to zero.

The odd-even block structure of iM leads to a simplification in the Lanczos algorithm, if we choose the initial Lanczos vector to be nonzero only on

even sites

$$x_1 = \begin{pmatrix} \hat{x}_1 \\ 0 \end{pmatrix}. \tag{26}$$

Then

$$\alpha_1 = x_1^+ i\mathcal{M} x_1 = (\hat{x}_1^+, 0) \begin{pmatrix} 0 & \hat{M} \\ \hat{M}^+ & 0 \end{pmatrix} \begin{pmatrix} \hat{x}_1 \\ 0 \end{pmatrix} = 0,$$

$$\beta_1 x_2 = i\mathcal{M} x_1 - \alpha_1 x_1 = i\mathcal{M} x_1 = \begin{pmatrix} 0 \\ \hat{M}^+ x_1 \end{pmatrix}. \tag{27}$$

So x_2 will be nonzero only on odd sites. Continuing in a similar fashion we find that all the α's are zero, and the Lanczos vectors are all half zero:

$$x_{2k-1} = \begin{pmatrix} \hat{x}_{2k-1} \\ 0 \end{pmatrix}, \quad x_{2k} = \begin{pmatrix} 0 \\ \hat{x}_{2k} \end{pmatrix}, \quad k = 1, \ldots, \tfrac{1}{2}\tilde{N}. \tag{28}$$

The Lanczos equations take the form

$$\begin{aligned} \hat{M}^+ \hat{x}_1 &= \beta_1 \hat{x}_2 \\ \hat{M}\, \hat{x}_{2k} &= \beta_{2k-1} \hat{x}_{2k-1} + \beta_{2k} \hat{x}_{2k+1} \\ \hat{M}^+ \hat{x}_{2k+1} &= \beta_{2k} \hat{x}_{2k} + \beta_{2k+2} \hat{x}_{2k+2} \end{aligned} \tag{29}$$

We have applied this algorithm to the study of chiral symmetry breaking on lattice configurations of size 8^4 [4] and to the study of chiral symmetry

restoration at finite temperature on lattice configurations of sizes $12^3 \cdot 4$ and $12^3 \cdot 6$ [5]. We used $\tilde{N} = 2N$ to obtain all the eigenvalues. In the case of the $12^3 \cdot 4$ lattice there were 20756 of them. From this we were able to establish that the small eigenvalues were sufficient for our purposes. These could be obtained after $\tilde{N} = 1/4\ N$ iterations. In the Lanczos algorithm we used double precision(8 bytes) arithmetic, even though the elements of the matrix were only supplied to single precision, but for the Sturm sequences we were able to reduce this to 6 byte arithmetic. Eigenvalues of the tridiagonal form which differed by less than 1 part in 10^8 were taken to be ghosts, and those that moved by more than 1 part in 10^6 when the last α and β were removed were taken to be spurious. This left precisely the correct number of eigenvalues when $\tilde{N} = 2N$, and the sum of their squares checked with the original matrix to 8 significant figures.

4. The Non-Hermitian Lanczos Method

The Lanczos method can be generalized to include nonhermitian matrices. We require a similarity transform

$$X^{-1} H X = T. \tag{30}$$

T will not in general be hermitian, since H may have complex eigenvalues, but we can make it symmetric

$$T = \begin{pmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \alpha_2 & \beta_2 & \\ & \beta_2 & \ddots & \end{pmatrix} \tag{31}$$

with the α's and β's complex. We have the same Lanczos equations as before

$$H x_1 = \alpha_1 x_1 + \beta_1 x_2$$
$$H x_i = \beta_{i-1} x_{i-1} + \alpha_i x_i + \beta_i x_{i+1} ,$$

but X may be nonunitary, so we must also generate its inverse

$$Y = X^{-1\dagger} , \quad H^\dagger Y = Y T^\dagger , \quad Y^\dagger X = 1 . \tag{32}$$

The columns of Y can be calculated with the additional Lanczos equations

$$H^\dagger y_1 = \alpha_1^* y_1 + \beta_1^* y_2 , \quad Y = (y_1, \ldots, y_N)$$
$$H^\dagger y_i = \beta_{i-1}^* y_{i-1} + \alpha_i^* y_i + \beta_i^* y_{i+1} \tag{33}$$

$$y_i^\dagger x_j = \delta_{ij} \tag{34}$$

The α's are calculated from

$$\alpha_i = y_i^\dagger H x_i . \tag{35}$$

The β's and the Lanczos vectors x_{i+1}, y_{i+1} come from

$$\beta_i x_{i+1} = H x_i - \alpha_i x_i - \beta_{i-1} x_{i-1}$$
$$\beta_i^* y_{i+1} = H^\dagger y_i - \alpha_i^* y_i - \beta_{i-1}^* y_{i-1} \tag{36}$$

$$\beta_i^2 = (\beta_i^* y_{i+1})^\dagger (\beta_i x_{i+1}) . \tag{37}$$

We need to take a complex square root to calculate β_i.

As with the hermitian case the orthogonality between the Lanczos vectors is rapidly lost due to rounding errors. Again we can continue the algorithm past N iterations and hope that the eigenvalues will still converge. Unfortunately, there is no easy way of generalizing the method of Sturm sequences to a complex symmetric tridiagonal form.[6] The only alternative is to construct the coefficients of the characteristic polynomial and find its roots, but this does not work well for $N \gtrsim 100$. Therefore we are forced to use reorthogonalization. Each x_i calculated must be reorthogonalized against all previous y_i, and similarly each y_i against the previous x_i. Once we have the tridiagonal form, the trace of the inverse can be calculated relatively quickly and can be repeated with different constants added to the diagonal.

5. Matrix Inversion

The Lanczos algorithm can be applied in a different way to invert a matrix column by column. This applies to both hermitian and nonhermitian matrices, though we have only tried it out for hermitian matrices. Generate the Lanczos equations as before without reorthogonalization:

$$H x_1 = \alpha_1 x_1 + \beta_1 x_2 , \quad H x_i = \beta_{i-1} x_{i-1} + \alpha_i x_i + \beta_i x_{i+1} . \tag{38}$$

We shall aim to calculate $H^{-1} x_1$ as an infinite series in the Lanczos vectors,

$$H^{-1} x_1 = c_1 x_1 + c_2 x_2 + \cdots , \tag{39}$$

by using the Lanczos equations iteratively. After k iterations of the Lanczos algorithm we will have constructed k terms in the series with a remainder term

involving $H^{-1}x_k$ and $H^{-1}x_{k+1}$:

$$H^{-1}x_1 = v_k + a_k H^{-1}x_k + b_k H^{-1}x_{k+1}$$

$$v_k = \sum_{i=1}^{k} c_i x_i \quad . \tag{40}$$

The next Lanczos equation can be used to eliminate $H^{-1}x_k$:

$$H^{-1}x_k = \frac{1}{\beta_k} x_{k+1} - \frac{\alpha_{k+1}}{\beta_k} H^{-1}x_{k+1} - \frac{\beta_{k+1}}{\beta_k} H^{-1}x_{k+2} \tag{41}$$

$$\Rightarrow \quad H^{-1}x_1 = v_k + \frac{a_k}{\beta_k} x_{k+1} + \left(b_k - \frac{\alpha_{k+1}}{\beta_k} a_k\right) H^{-1}x_{k+1} - \frac{\beta_{k+1}}{\beta_k} a_k H^{-1}x_{k+2} \tag{42}$$

giving the following recurrence relations

$$v_{k+1} = v_k + \frac{a_k}{\beta_k} x_{k+1}$$

$$\begin{pmatrix} a_{k+1} \\ b_{k+1} \end{pmatrix} = \begin{pmatrix} -\frac{\alpha_{k+1}}{\beta_k} & 1 \\ -\frac{\beta_{k+1}}{\beta_k} & 0 \end{pmatrix} \begin{pmatrix} a_k \\ b_k \end{pmatrix} . \tag{43}$$

Initial values for v_1, a_1 and b_1 can be obtained from the first equation if $\alpha_1 \neq 0$:

$$H^{-1}x_1 = \frac{1}{\alpha_1} x_1 - \frac{\beta_1}{\alpha_1} H^{-1}x_2 \, , \tag{44}$$

but we can also start from the identity

$$H^{-1}x_1 = H^{-1}x_1 \, . \tag{45}$$

As we shall see, this ambiguity in initial conditions is important for convergence of the series. Combine (44) and (45) to give the most general starting conditions

$$\left(r - \frac{\alpha_1}{\beta_1} s\right) H^{-1}x_1 = -\frac{s}{\beta_1} x_1 + r H^{-1}x_1 + s H^{-1}x_2 \tag{46}$$

$$\Rightarrow \quad v_1 = -\frac{s}{r - \frac{\alpha_1}{\beta_1} s} \frac{x_1}{\beta_1} \tag{47}$$

$$\begin{pmatrix} a_1 \\ b_1 \end{pmatrix} = \frac{1}{r - \frac{\alpha_1}{\beta_1} s} \begin{pmatrix} r \\ s \end{pmatrix} . \tag{48}$$

The parameters r and s must be left undetermined until the end of the calculation, when they can be choosen in such a way that the remainder term is small

$$\begin{pmatrix} a_k \\ b_k \end{pmatrix} = \frac{1}{r - \frac{\alpha_1}{\beta_1} s} \Pi_k \begin{pmatrix} r \\ s \end{pmatrix} , \tag{49}$$

where the 2x2 matrix Π_k can be calculated from the recurrence relations

$$\Pi_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{50}$$

$$\Pi_{k+1} = \begin{pmatrix} -\frac{\alpha_{k+1}}{\beta_k} & 1 \\ -\frac{\beta_{k+1}}{\beta_k} & 0 \end{pmatrix} \Pi_k \,. \tag{51}$$

For convergence of the series we require

$$\begin{pmatrix} a_k \\ b_k \end{pmatrix} \to \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{52}$$

This would happen for any choice of initial conditions if

$$\Pi_k \to 0 \,, \tag{53}$$

but

$$\det \Pi_k = \prod_{i=1}^{k-1} \det \begin{pmatrix} -\frac{\alpha_{i+1}}{\beta_i} & 1 \\ -\frac{\beta_{i+1}}{\beta_i} & 0 \end{pmatrix}$$

$$= \prod_{i=1}^{k-1} \frac{\beta_{i+1}}{\beta_i} = \frac{\beta_k}{\beta_1} \,. \tag{54}$$

Therefore, unless we have a β equal to zero, we cannot have $\Pi_k \to 0$. However, if one eigenvalue of Π_k tends to zero, we can take $\begin{pmatrix} r \\ s \end{pmatrix}$ to be the corresponding eigenvector, and this will suffice to make the remainder term small. Since we will not know r and s until the end, we must compute v_k as a linear combination

$$v_k = \frac{1}{r - \frac{\alpha_1}{\beta_1} s} \; \sigma_k \begin{pmatrix} r \\ s \end{pmatrix}. \tag{55}$$

σ_k has two component vectors and is generated from the following relations

$$\sigma_1 = \left(0, -\frac{x_1}{\beta_1}\right) \tag{56}$$

$$\sigma_{k+1} = \sigma_k + \left(\frac{x_{k+1}}{\beta_k}, 0\right) \Pi_k \,. \tag{57}$$

If we now proceed to calculate Π_k and σ_k from the relations (50), (51), (56) and (57) naively, then we would run into a problem with rounding errors. As one eigenvalue of Π_k converges to zero, the other tends to diverge because the determinant fluctuates about a constant value. This means that the components of Π_k and σ_k will grow large and the convergent part will be lost in rounding errors, since it is a difference of large values. These errors can be avoided if we choose an unconventional representation of Π_k and σ_k, which separates the convergent and divergent parts:

$$\Pi_k = \begin{pmatrix} A_k & \gamma_k A_k \\ B_k & \gamma_k B_k + t_k \end{pmatrix} \tag{58}$$

$$\sigma_k = (0, 1)\, V_k + (1, \gamma_k)\, U_k \,. \tag{59}$$

As one eigenvalue of Π_k converges to zero, we will have $t_k \to 0$ while

$$A_k, B_k, U_k \to \infty \,. \tag{60}$$

If we then choose

$$\begin{pmatrix} r \\ s \end{pmatrix} = \begin{pmatrix} \gamma_k \\ -1 \end{pmatrix}, \tag{61}$$

we will have

$$\begin{pmatrix} a_k \\ b_k \end{pmatrix} = \frac{1}{\gamma_k + \frac{\alpha_1}{\beta_1}} \begin{pmatrix} 0 \\ -t_k \end{pmatrix} \to 0 \tag{62}$$

and

$$v_k = -\frac{1}{\gamma_k + \frac{\alpha_1}{\beta_1}} V_k \to H^{-1} x_1 . \tag{63}$$

The relations (50), (51), (56) and (57) translate into the following relations for y_k, t_k, A_k, B_k, V_k and U_k:

$$\begin{aligned} y_1 &= 0 \\ t_1 &= 1 \\ A_1 &= 1 \\ B_1 &= 0 \\ V_1 &= -\frac{x_1}{\beta_1} \\ U_1 &= 0 \end{aligned} \tag{64}$$

$$\begin{pmatrix} A_{k+1} \\ B_{k+1} \end{pmatrix} = \begin{pmatrix} -\frac{\alpha_{k+1}}{\beta_k} & 1 \\ -\frac{\beta_{k+1}}{\beta_k} & 0 \end{pmatrix} \begin{pmatrix} A_k \\ B_k \end{pmatrix} \tag{65}$$

$$\gamma_{k+1} = \gamma_k + \frac{t_k}{A_{k+1}} \tag{66}$$

$$t_{k+1} = -\frac{B_{k+1}}{A_{k+1}} t_k \tag{67}$$

$$U_{k+1} = U_k + \frac{A_k}{\beta_k} x_{k+1} \tag{68}$$

$$V_{k+1} = V_k - \frac{t_k}{A_{k+1}} U_{k+1} \tag{69}$$

6. Application to Propagator Calculations in Lattice QCD

Quark propagators in lattice QCD can be obtained from the inverse of the fermion matrix M + 2ma. As with the $\langle \bar{\psi}\psi \rangle$ calculations we can use the odd-even block structure of M to improve the calculation. We apply the Lanczos inversion algorithm to the matrix

$$H = i(M + 2ma) = \begin{pmatrix} 2mai & \hat{M} \\ \hat{M}^+ & 2mai \end{pmatrix} \tag{70}$$

$$H^{-1} = \begin{pmatrix} -2ma\,i\,[\hat{M}\hat{M}^{+}+(2ma)^2]^{-1} & \hat{M}\,[\hat{M}\hat{M}^{+}+(2ma)^2]^{-1} \\ \hat{M}^{+}\,[\hat{M}\hat{M}^{+}+(2ma)^2]^{-1} & -2ma\,i\,[\hat{M}\hat{M}^{+}+(2ma)^2]^{-1} \end{pmatrix} . \tag{71}$$

The Lanczos equations are

$$\begin{aligned} H x_1 &= 2ma\,i\,x_1 + \beta_1 x_2 \\ H x_i &= \beta_{i-1} x_{i-1} + 2ma\,i\,x_i + \beta_i x_{i+1} \; . \end{aligned} \tag{72}$$

The Lanczos vectors and the β's do not depend on 2ma, so it is possible to obtain the inverse of a column at many different masses at the same time. This may be limited by storage space, in which case it is possible to calculate just the half of the column of the inverse which is in the diagonal block in equation (71). The other half can be reconstructed by multiplying by $-\frac{1}{2ma\,i}\,\hat{M}^{+}$ or $-\frac{1}{2ma\,i}\,\hat{M}$ as appropriate.

In order to avoid a division by zero in the case 2ma = 0, we can use a slightly different representation of Π_k and σ_k than (58) and (59)

$$\begin{aligned} \Pi_{2k-1} &= \begin{pmatrix} t_{2k-1} - (2ma)^2 y_{2k-1} z_{2k-1} & 2ma\,i\,z_{2k-1} \\ 2ma\,i\,y_{2k-1} A_{2k-1} & A_{2k-1} \end{pmatrix} \\ \Pi_{2k} &= \begin{pmatrix} 2ma\,i\,y_{2k}\,A_{2k} & A_{2k} \\ t_{2k} - (2ma)^2 y_{2k} z_{2k} & 2ma\,i\,z_{2k} \end{pmatrix} \\ \sigma_k &= (2ma\,i\,V_k\,,0) + (2ma\,i\,Y_k\,,1)\,U_k \; . \end{aligned} \tag{73}$$

Then the factor 2mai divides out explicitly. This representation also has the advantage that the coefficients y_k, t_k, A_k and B_k are real.

We have investigated the convergence of t_k on 8^4 lattices using double precision arithmetic. We find that for $ma \geq 0.05$ at $\beta = 5.7$ convergence is similar to that of the conjugate gradient method. However, for zero mass or very small mass we find the convergence of the Lanczos algorithm is much better. For nonzero mass t_k converges steadily from the start of the calculation at a rate proportional to m. At zero mass the situation is very different. Then t_k shows no sign of convergence until we reach the point at which the first eigenvalues start to converge. From then on t_k drops rapidly and stops when convergence of the inverse is complete to within machine precision.

We are currently applying the algorithm to 16^4 lattices using single precision arithmetic. In order to check the accuracy we calculate:

$$r^2 = |Hz - x|^2 \; , \tag{74}$$

where z is the calculated value for $H^{-1}x$. We find

$$r^2 \lesssim 10^{-8} .$$

7. Application to Fermion Updating

Probably the most difficult problem in lattice gauge theories is to include the effect of dynamical fermions on the gauge field configurations. This requires us to calculate the ratio of determinants of M + 2ma between gauge configurations which differ only at one link. This must be done many times when the background gauge configuration is being generated. Because of the vast amount of computing that this takes, the quenched approximation has been used in the past in which the ratio of determinants has been taken as one.

This gives quite reasonable results but it is not ultimately satisfactory. There have been some useful attempts to overcome this using pseudo fermions, microcanonical methods, etc. but once again the convergence at physically relevant masses is poor.

It is possible to use our inversion method in the updating problem as follows.

$$H = M + 2am \tag{75}$$

Changing one link changes H by ΔH :

$$R = \frac{\det (H+\Delta H)}{\det H} = \det (1 + H^{-1}\Delta H) \tag{76}$$

ΔH has only 18 nonzero elements, which lie within a 6 x 6 "block" at the intersection of the 6 rows and columns, corresponding to the 2 sites joined by the link which has been changed. It can be readily seen from this that only the corresponding 6 x 6 block of the inverse H^{-1} contributes in (76). Therefore if we calculate 6 rows of columns of H^{-1}, we can update the link. Moreover the same link can be updated a number of times without recalculating the inverse. There is some waste here in calculating a whole row when we only need 6 elements, but there is a way we can make use of it to improve the rate of convergence. Suppose for the moment that we have an approximation to the whole inverse

$$Z = H^{-1} + \epsilon \ . \tag{77}$$

We can try to get an estimate of the error ϵ. Calculate

$$HZ = 1 + H\epsilon \ . \tag{78}$$

Then use the estimate of H^{-1} to calculate ϵ approximately

$$Z(HZ-1) = (H^{-1}+\epsilon) H\epsilon = \epsilon + \epsilon H\epsilon . \tag{79}$$

Then provided Z was a good approximation, $\epsilon H\epsilon$ will be much smaller than ϵ and

$$Z - Z(HZ-1) = H^{-1} - \epsilon H\epsilon , \tag{80}$$

so we have a better approximation to the inverse from a relatively small amount of extra computation. If, now, we only have 6 columns of Z, it is still possible to do part of the calculation. First note that the rows of Z can be obtained trivially from the corresponding columns because of the hermitian - antihermitian structure in (71). Then it is possible to calculate the 6x6 elements of (80) that are needed for the updating.

We intend to apply this method using Lanczos to update on modest sized lattices.

Acknowledgement

We would like to thank Roger Haydock and Ian Duff for useful discussions. IMB and PEG would like to thank the DESY theory group for their hospitality during parts of the present work. We would also like to thank the SERC for use of the DAP facility at Queen Mary College and Prof. D. Parkinson and Kevin Smith for help in using it. PEG would like to thank the SERC for financial support and NB would like to thank the MESRS of Algeria for financial support. We would also like to thank J.P. Gilchrist and H. Schneider, our collaborators, for the lattice calculations.

References

1) J. Gilchrist, G. Schierholz, H. Schneider, M. Teper: DESY Preprint 84-021 (1984), to be published in Nucl. Phys. B

2) K. Bowler, D. Chalmers, A. Kenway, R. Kenway, G. Pawley, D. Wallace: Edinburgh Preprint 84/295 (1984)

3) J. Cullum, R.A. Willoughby in Sparse Matrix Proceedings 1978, Eds. I. Duff and G. Stewart (Siam Press, 1979). For a more recent variation of the method see: B.N. Parlett, J.K. Reed: IMA Journal of Numerical Analysis 1, 135 (1981).
See also I.S. Duff, A survey of sparse matrix software, Harwell report AERE-R 10512 (1982). B.N. Parlett: The software scene in the extraction of eigenvalues from sparse matrices, Berkeley Report PAM-132 (1983).
R. Haydock: Consequences of rounding error in the recursion and Lanczos methods, Cavendish preprint. B.N. Parlett: The Symmetric Eigenvalue Problem (Prentice-Hall, 1980)

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

5) In preparation

6) This method has now been generalized and the eigenvalues of the non-symmetric case found by a method analogous to the symmetric case. We thank I. Duff for providing a method which diagonalises the complex symmetric tridiagonal form.