Reflections on the Large Scale Matrix Computations in Lattice QCD Conference

by Richard Brower, Jim Demmel, Alan Edelman, and George Hockney

This is not a conference summary of the meeting held from August 31, 1995 to September 2, 1995, but rather a short document whose goal is to extend the exciting interdisciplinary atmosphere that was generated at the conference. Here we 1) commit to paper a brief introduction to QCD, and 2) write down suggestions that were raised at the panel discussion. We consider the very existence of this note a sign of the success of this conference – our two cultures could really meet and talk.

1 QCD for the color-blind

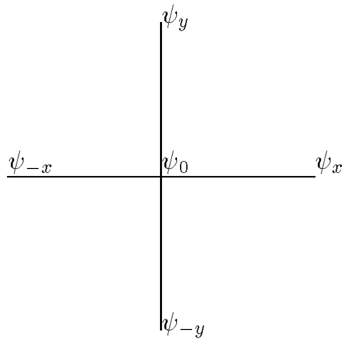
We recognize that decoupling the physics from the numerical problem is a long term mistake, but the only way to *start* a meaningful dialogue between the two groups is to begin with an oversimplified cocktail party version of the physical motivations without any equations, and then to present the numerical problem with very little of the physics.

Quantum Chromodynamics (QCD) is a theory meant to predict experimentally measurable physical quantities, for example the mass of the proton and neutron, from the underlying interaction of quarks and gluons through the so-called "strong" force." Particle physicists walk around with pocket tables of experimentally measured quantities whose values could, in principle, be computed from QCD theory. By contrast, a simpler theory exists for electrons, which are not subject to the strong force.

So much for the physics. Now for the numerical problems. An important computational bottleneck in QCD is the solution of a huge ($n \sim 2,000,000$) sparse system of linear equations. This system appears in the inner loop of a Monte Carlo simulation so an efficient solution is critical. The entire Monte Carlo simulation computes certain infinite dimensional integrals whose values can be extrapolated so as to obtain the physical quantities of interest.

The first question a numerical analyst wants to know is what is the matrix? The sparse linear operator is not so very different from the familiar finite difference discretization of the Laplacian operator on a 2d grid:

$$\nabla^2 \psi \approx \frac{1}{h^2}(-4\psi_0 + \{\psi_x + \psi_y + \psi_{-x} + \psi_{-y}\}), \qquad (1)$$



where the subscripts represent the grid value ψ_0, and its north, east, south, west neighbors. The diagram focuses on one lattice point and its neighbors, but of course it is understood that the operator applies to every lattice point and its neighbors.

In QCD, the so-called Wilson action $\mathcal{M}$ is always applied on a four dimensional space-time lattice, usually with periodic boundary conditions and lattice volumes (grid sizes) denoted $L^3 \times L_t$. For purposes of exposition, a 2-d version will be presented. On each site, the lattice variables ψ are not scalars, but rather two index tensors, $\psi_{a,i}$, (4x3 matrices) with 12 components: the color (row) index takes on 3 values ($a = 1, 2, 3$) and the spin (column) index takes on 4 values ($i = 1, 2, 3, 4$). (Physicists usually prefer to write ψ as a vector of length 12.) The operator is

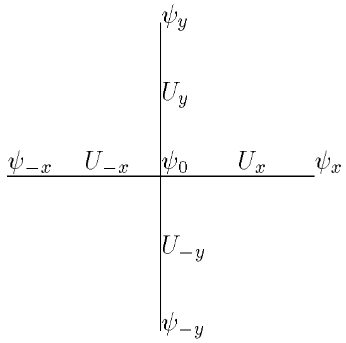
$$(\mathcal{M}\psi)_0 = \psi_0 - \kappa(C_x\psi_x U_x + C_y\psi_y U_y + C_{-x}\psi_{-x}U^H_{-x} + C_{-y}\psi_{-y}U_{-y}),$$

where

$$\begin{array}{ll} \psi_i \in \mathbb{C}^{4,3} & \text{the quark} \\ U_i \in \mathbb{C}^{3,3} & \text{the gluon}, U^H U = I, \det U = 1 \\ C_i \in \mathbb{C}^{4,4} & \text{transport matrices (built from Dirac matrices)} \end{array},$$

where $i = x, y, -x, -y$, i.e., east, north, west, and south. Here κ is a positive constant. The 3 by 3 (special) unitary matrices live on the links, and are randomly chosen from a probability distribution

$$P[U_i] = const \times \exp\{-\beta S_0[U] + Trace[Log(\mathcal{M}^H \mathcal{M})]\ \}.$$



Thus the sparse system may be thought of as a matrix problem. In the above expression U denotes the totality of all the U_i's at all of the sites. The action term $S_0[U]$ is the sum over all lattice squares ("plaquettes") of the corresponding product of U_i's.

Typical U matrices must be generated by a program or read in from a file. The C_i are very simple rank two matrices and are constant in the sense that over the whole lattice, C_x has the same value for every Eastward link, and C_y has the same value for every Northward link, etc. The C_i are rank two projections. The operator $\mathcal{M}$ would be Hermitian if reversing the ingoing and the outgoing links is equal to the complex conjugate. In this case, the operator is nearly Hermitian in that reversing the link orientation amounts to applying a permutation matrix (called γ_5) to each of the spin indices of the Hermitian conjugate. From the numerical point of view, this means that we can apply transpose free iterative methods towards the solution of the problem.

To summarize the problem space, this matrix problem for QCD has 4 parameters:

- L and L_t which determine the overall size of the huge matrix problem that needs to be solved, $n = 12L^3 \times L_t$.
- κ (a constant) is related to the condition number (which numerical analysts also often call κ, but we will not do that here). The condition number diverges as κ is increased from 0 to $\kappa_{critical}$
- β the coefficient for a local smoothing term $S_0[U]$ — large β implies smooth physics or large lattice spacing.
- κ_{sea} the kappa for the Trace-Log term in the probability distribution for the U's. (The κ for the sparse system need not be the same as the one for the probability distribution.)

The first two parameters enter directly into defining the matrix $\mathcal{M}$ being inverted (actually of course no matrix is being inverted, a system is being solved) and the second two indirectly influencing the typical values of the random link matrices U. In a truly self-consistent (ie correct!) simulation, κ_{sea} and κ should be the same, but in practice one often uses $\kappa_{sea} \leq \kappa$. The example code will have $\kappa_{sea} = 0$, thus removing the Trace-Log piece all together. (Physicists call this the quenched approximation. It always good to have a nice name for committing an error!)

In solving the linear system there are two types of right hand sides (RHS). One class typically represents twelve (or more) linear systems whose solutions we want simultaneously

$$\mathcal{M}\psi = \left\{ \begin{array}{cc} E_{i,a} & \text{center node} \\ 0 & \text{elsewhere} \end{array} \right. , i = 1, 2, 3, 4\ a = 1, 2, 3,$$

where $E_{i,a}$ is the 4×3 tensor of the same character as ψ on one lattice site. (In other words, $E_{i,a}$ is also a 4×3 matrix.) The twelve cases correspond to taking $E_{i,a}$ to be the matrix who (i, a)th entry is 1 and the other entries 0. The second class of problems has for its RHS a dense random vector (precisely the RHS is $\phi = M^H b$ where all n components of b are independent random Gaussian numbers.) For a fixed value of the RHS, one typically considers 100 inversion problems (normal equations),

$$\mathcal{M}^H(t_i)\mathcal{M}(t_i)\psi = \phi$$

generated in succession by slowly and continuously varying the U matrices at the links in $\mathcal{M}$.

The first question numerical analysts want to know is what is the distribution of the singular values or the eigenvalues of these operators. This information is known and may be found in a number of papers. (Current Best way to look is the Physics Papers web server found on `http://xxx.lanl.gov/archive/hep-lat`.) An important point is that the condition numbers are quite reasonable, perhaps only in the thousands in many cases. What is needed is to take an already well-conditioned problem, and still find ways to make it converge faster somehow.

The solution of this system enters into the Monte Carlo calculations, but we will not specify how in this note.

2 Suggestions for further research

It was generally agreed that successful technology transfer of recent improvements in itervate methods for non-symmetric linear systems has already occurred prior to the meeting.

Fast methods for obtaining the best answers in the usual Krylov spaces may well have reached the limits of what is possible. The current best algorithm for the solution of the linear system taking into account the storage requirements seems to be BiCGSTAB with the ILU Red-Black preconditioner, although the look-ahead Lanczos method is supposedly stabler and allows multiple κ values to be computed simultaneously from the same source. However, although the non-look-ahead Lanczos can take advantage of the γ_5 symmetry to require the same storage as BiCGSTAB it is not stable, especially in the presence of the so-called clover term.

Until recently the most commonly used algorithms were GMRES(1) (called minimum residual by the lattice community) and Conjugate Gradient using $\mathcal{M}^H\mathcal{M}$ in the light-quark-mass region where GMRES(1) failed to converge.

These best algorithms all use the ILU Red-Black preconditioning, because it not only provides a factor of about 1.5 as a preconditining step but also reduces the arithmetic by a factor of 2. It is known that the ILU hyperplane preconditioner is better (with a preconditioning factor of about 4) but it does not reduce the storage and is difficult to program efficiently on existing computers.

Many other preconditioners have been tried, notably FFT and multi-grid techniques. Except on non-physical (for example, cold) lattices, the FFT does not precondition as well as ILU Red-Black per unit arithmetic.

Further improvements will have to be obtained by wandering out of the Krylov space and/or more effectively applying information from previous iterations. One can imagine using information from the past to pick an improved starting vector, or to create more effective preconditioners. The former is already working in some codes, while nobody knows how to employ the latter.

Typically the solution is required for multiple right hand sides. One approach towards multiple right hand sides is to use block methods with deflation. Are there other methods that may be of use.

Since these operators so closely resemble the Laplacian, the multigrid idea comes to mind. Multigrid has not been very successful in this field, in part because, unlike the Laplacian, the operators are not discretization of smooth operators. The random U create a graininess that seems to interfere with multigrid. An important numerical problem, then, is to find the right way to use multigrid in this context, possibly taking into account the theory of renormalization groups.

There are also important parallel computing challenges that could be of interest to the numerical community. Hyperplane preconditioners, etc.

3 Conclusion

These notes are just a start. For other articles for the starting reader we wish to mention

- Monte Carlo simulations of lattice gauge theories by Philippe de Forcrand in *Aspects of Computing on Asynchronous Parallel Processors*, M.H. Wright, ed., Elsevier.
- Chapter 4.3 of *Parallel Computing Works* by G. Fox, R. Williams, and P. Messina.

We all agreed that the physicists and the numerical analysts were able to communicate and that the physicists truly had challenging problems to solve.