Multigrid Solution of the Poisson–Boltzmann Equation

Michael Holst* and Faisal Saied

Numerical Computing Group, Department of Computer Science, University of Illinois at Urbana-Champaign, 1304 W. Springfield Ave., Urbana, Illinois 61801

Received 22 April 1992; accepted 23 July 1992

A multigrid method is presented for the numerical solution of the linearized Poisson–Boltzmann equation arising in molecular biophysics. The equation is discretized with the finite volume method, and the numerical solution of the discrete equations is accomplished with multiple grid techniques originally developed for two-dimensional interface problems occurring in reactor physics. A detailed analysis of the resulting method is presented for several computer architectures, including comparisons to diagonally scaled CG, ICCG, vectorized ICCG and MICCG, and to SOR provided with an optimal relaxation parameter. Our results indicate that the multigrid method is superior to the preconditioned CG methods and SOR and that the advantage of multigrid grows with the problem size. © 1993 by John Wiley & Sons, Inc.

INTRODUCTION

Continuum models of molecules in ionic solutions, first proposed in 1923 by Debye and Hückel,[1] are increasingly important tools for studying electrostatic interactions and are now being incorporated into molecular dynamics simulators.[2–4] Because the electrostatic behavior contributes to the structure, binding properties, as well as the kinetics of complex molecules such as proteins, modeling these interactions accurately is an important problem in biophysics.

The fundamental equation arising in the Debye–Hückel theory is a three-dimensional second-order nonlinear elliptic partial differential equation describing the electrostatic potential $\Phi(\mathbf{r})$ at a field position $\mathbf{r}$. In the special case of a 1:1 electrolyte, this equation can be written as

$$-\nabla \cdot (\epsilon(\mathbf{r})\nabla\Phi(\mathbf{r})) + \bar{\kappa}^2 \left(\frac{k_B T}{e_c}\right) \sinh\left(\frac{e_c\Phi(\mathbf{r})}{k_B T}\right) = 4\pi \sum_{i=1}^{N_m} q_i \delta(\mathbf{r} - \mathbf{r}_i) \quad (1)$$

where the permitivity $\epsilon(\mathbf{r})$ takes the values of the appropriate dielectric constants in the different regions of the model (the value ϵ_m in the molecular region, and a second value ϵ_w in both the solution region and an ion-exclusion layer surrounding the molecule), and the *modified* Debye–Hückel parameter $\bar{\kappa} = \sqrt{\epsilon_w}\kappa$, where κ, the usual Debye–Hückel parameter, is proportional to the ionic strength of the solution (the modification makes $\bar{\kappa}$ *dielectric independent*). The molecule is represented by N_m point charges q_i at positions $\mathbf{r}_i$, yielding the delta functions in (1), and the constants e_c, k_B, and T represent the charge of an electron, Boltzmann's constant, and the absolute temperature. Equation (1) is referred to as the *nonlinear Poisson–Boltzmann equation* (NPBE), and its solution is usually approximated by solving the *linearized Poisson–Boltzmann equation* (LPBE):

$$-\nabla \cdot (\epsilon(\mathbf{r})\nabla\Phi(\mathbf{r})) + \bar{\kappa}^2\Phi(\mathbf{r}) = 4\pi \sum_{i=1}^{N_m} q_i \delta(\mathbf{r} - \mathbf{r}_i) \quad (2)$$

Analytic solutions to the LPBE and NPBE are quite complex, even in the few simple situations for which they exist.[5] Due to advances in computational algorithms and hardware in recent years, several investigations into the efficiency and accuracy of numerical methods for the LPBE have appeared,[3,6–10] while numerical solution of the NPBE remains largely unstudied (see Nicholls and Honig[8] for a discussion). These studies in general involve a finite volume discretization of (2) with approximated boundary conditions, followed by iterative solution of the discrete equations with a relaxation or preconditioned conjugate gradient method. (Exceptions are the studies of Yoon and Lenhoff[10] and Juffer et al.,[7] in which integral equation formulations of the LPBE are used.) In this article, the usual finite volume discretization approach is taken as in earlier studies[6,8] and a multigrid method is employed for solution of the discrete equations.

Outline

Below, a description of the multigrid approach is given and the difficulties presented by discontinui-

*Author to whom all correspondence should be addressed.

© 1993 by John Wiley & Sons, Inc.
CCC 0192-8651/93/010105-09

ties occurring in the NPBE and the LPBE are discussed. Methods currently in use for the LPBE are then reviewed, including conjugate gradient methods and successive overrelaxation. Numerical results and performance statistics comparing the multigrid method to methods currently in use (and some additional methods) are then presented for solution of the LPBE in the case of an acetemide molecule in water. The results are then summarized and some conclusions are drawn.

MULTIGRID METHODS

Multigrid methods are highly efficient numerical techniques for solving the algebraic equations resulting from the discretization of a partial differential equation on a fine mesh by using auxiliary problems on coarser meshes. They are provably optimal order for certain classes of problems[11-13] and are also extremely effective for many other problems. Detailed descriptions of these methods, for both linear and nonlinear elliptic problems, can be found in the early article by Brandt,[14] or in the extensive text on the subject by Hackbusch.[13]

The Two-Grid Algorithm

To explain the multigrid approach, consider a second-order linear elliptic equation $Lu = f$ on domain $\Omega \subset \mathbb{R}^d$. Given an appropriately accurate discretization $L_h u_h = f_h$ on a mesh Ω_h for some mesh parameter h, with the nonsingular matrix L_h defined by either finite differences, finite volumes, or finite elements, a sequence of problems $L_H u_H = f_H$ on coarser meshes Ω_H may be defined in a number of ways. One approach, termed *Galerkin coarsening* due to its connection to finite element theory, is defined as

$$L_H = I_h^H L_h I_H^h \tag{3}$$

where the linear operators I_h^H and I_H^h are *restriction* and *prolongation* operators, respectively, which map grid functions between the fine and coarse grid function spaces.

With two such discretizations, a two-grid *correction scheme*[14] or *linear multigrid method*[13] can be used to solve the underlying continuous problem. Given current approximation u_h^j to the discrete solution u_h at the jth iteration, the method computes a *correction* to u_h^j from the error equation

$$L_h e_h^j = L_h(u_h - u_h^j) = L_h u_h - L_h u_h^j = f_h - L_h u_h^j = r_h^j \tag{4}$$

where $e_h^j = u_h - u_h^j$ is the (unknown) *discrete error* and r_h^j is the (computable) *discrete residual*. If the error is smooth, it can be represented well on a coarse grid. Therefore, a relaxation or *smoothing* iteration

$$\mathcal{S}_h(u_h^j, f_h) = S_h u_h^j + T_h f_h$$

is first applied for ν iterations, denoted by $\mathcal{S}_h^\nu$, to smooth the error in the approximation. Equation (4) is then solved on the coarse grid, and the result is transferred back to the fine grid. The resulting two-grid iteration is

Let u_h^0 be an initial approximation on Ω_h.
Do $j = 0, 1, 2, \ldots$ until convergence:
1. Presmooth: $\overline{u}_h = \mathcal{S}_h^{\nu_1}(u_h^j, f_h)$
2. Restrict the residual: $r_H = I_h^H(f_h - L_h\overline{u}_h)$
3. Solve for the correlation: $e_H = L_H^{-1} r_H$
4. Prolongate and correct: $\overline{\overline{u}}_h = \overline{u}_h + I_H^h e_H$
5. Postsmooth: $u_h^{j+1} = \mathcal{S}_h^{\nu_2}(\overline{\overline{u}}_h, f_h)$

End Do. (5)

Multigrid and Nested Iteration

The *multigrid method*[13] or *V-cycle*[14] begins with the fine grid, performs the two-grid method successively on each level until a sufficiently coarse grid is reached (for which the cost of complete solution is negligible), and cycles back to the fine grid. The *full multigrid method*[14] or *nested iteration technique*[13] begins with the coarse grid, prolongates the solution to a finer grid, performs a V-cycle, and repeats the process until a V-cycle is performed on the finest grid. Another variation is the *W-cycle*, in which more than one V-cycle is performed on each grid level. Each algorithm has particular advantages in various situations[13] and are depicted in Figure 1.

Discontinuous Coefficients

In the case of elliptic problems with smooth coefficients, a red/black Gauss–Seidel smoothing method, linear grid transfer operators, finite volume discretization on all grids, and direct or iterative solution on the coarse grid combine to yield an efficient algorithm.[15,16]

However, in the case of interface problems occurring in reservoir simulation and reactor physics as well as in biophysics the convergence rates of multigrid methods degrade drastically, and the methods may even diverge. Numerous studies have appeared addressing this problem, most notably the studies by Alcouffe et al.,[17] Dendy,[18,19] and Behie and

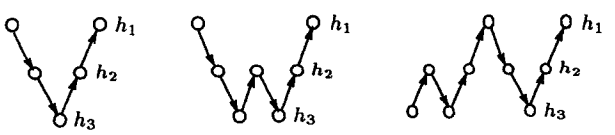

Figure 1. V-cycle, W-cycle, and nested iteration.

Forsythe.[20,21] Extensive numerical experiments indicate that forming the coarse grid equations by either the Galerkin coarsening procedure or a *harmonic averaging technique*, and coupling either of these with grid transfer operators that enforce continuity conditions across material interfaces (referred to as *operator-based prolongation*), leads to multigrid methods that regain their usual good convergence rates.

In the following three subsections, we give some details of how the continuity of $\epsilon(\mathbf{r})\nabla\Phi(\mathbf{r}) \cdot \mathbf{n}$ is enforced across interfaces and how harmonic averaging and operator-based prolongation are implemented.

Discretization of Interface Problems

Before discussing harmonic averaging and operator-based prolongation, we first review a standard discretization procedure for interface problems. Consider the following one-dimensional example, which will be used to explain all three of these procedures:

$$-\frac{\mathrm{d}}{\mathrm{d}x}\left(a(x)\frac{\mathrm{d}}{\mathrm{d}x}u(x)\right) + c(x)u(x) = f(x) \text{ in } (a, b),$$
$$u(a) = u(b) = 0 \quad (6)$$

The functions $a(x)$ and $c(x)$ are positive for all x in $[a, b]$, and $a(x)$, $c(x)$, and $f(x)$ are continuously differentiable everywhere, except that one or more of the three may be discontinuous at the *interface* point $x = \xi \in (a, b)$.

Define a discrete mesh $a = x_0 < x_1 < \ldots < x_{n+1} = b$, with $x_{i+1} = x_i + h_i$ for $h_i > 0$, such that the point of discontinuity coincides with some mesh point $x_i = \xi$. Then, the *integral method*[22] (the *box* or *finite volume method* in two or three dimensions) provides a rigorous technique for obtaining a discrete form of (6) at each mesh point x_i with valid error estimates despite the presence of the discontinuities. One considers the interval $[x_i - h_{i-1}/2, x_i + h_i/2]$ containing the point x_i, and integrates (6) over the interval. After performing the integration of the first term of (6) separately over the half-intervals $[x_i - h_{i-1}/2, x_i]$ and $[x_i, x_i + h_i/2]$, and enforcing the continuity condition at the interface point $x_i = \xi$

$$\lim_{x \to x_i -} a(x)\frac{\mathrm{d}}{\mathrm{d}x}u(x) = \lim_{x \to x_i +} a(x)\frac{\mathrm{d}}{\mathrm{d}x}u(x) \quad (7)$$

the following expression is obtained, which is exact for the solution $u(x)$ in the interval:

$$\left(a\left(x_i - \frac{h_{i-1}}{2}\right)\frac{\mathrm{d}}{\mathrm{d}x}u\left(x_i - \frac{h_{i-1}}{2}\right)\right) - \left(a\left(x_i + \frac{h_i}{2}\right)\frac{\mathrm{d}}{\mathrm{d}x}u\left(x_i + \frac{h_i}{2}\right)\right) + \int_{x_i - h_{i-1}/2}^{x_i + h_i/2} c(x)u(x)\mathrm{d}x = \int_{x_i - h_{i-1}/2}^{x_i + h_i/2} f(x)\,\mathrm{d}x$$

An algebraic expression is then obtained for an approximation to $u(x_i)$ by replacing the derivatives with differences and replacing the integrals with quadrature formulas separately over the half-intervals.

Denoting $u_i \equiv u(x_i)$, function values at half-grid points as $a_{i+1/2} \equiv a(x_i + h_i/2)$, and limiting function values at the left and right of the interface point x_i as $c_i^- \equiv c(x_i -)$ and $c_i^+ \equiv c(x_i +)$, we can, for example, write down an $O(h^2)$ (with $h = \max(h_{i-1}, h_i)$) approximation using centered differences and the rectangle rule:

$$a_{i-1/2}\left(\frac{u_i - u_{i-1}}{h_{i-1}}\right) - a_{i+1/2}\left(\frac{u_{i+1} - u_i}{h_i}\right) + u_i\left(\frac{h_{i-1}c_i^- + h_i c_i^+}{2}\right) = \left(\frac{h_{i-1}f_i^- + h_i f_i^+}{2}\right) \quad (8)$$

All approximations are performed over intervals where the functions are smooth; therefore, error estimates from the difference and quadrature formulas are valid. The extension to two or three dimensions is straightforward with use of the divergence theorem and by imposing continuity of the normal derivative $a(\mathbf{x})\nabla u(\mathbf{x}) \cdot \mathbf{n}$ at the interfaces.[22]

Harmonic Averaging

From the previous discussion, it should be clear that if discontinuities in the coefficients of (1) or (2) lie along grid lines and planes on all coarse grids then the standard finite volume discretization on all grids will produce accurate approximations. However, if the discontinuities are complex in shape, as in the case of nontrivial molecules, then the discontinuities may necessarily lie within individual elements or volumes on coarse grids, resulting in poor coarse grid approximations and poor multigrid convergence rates.

The Galerkin coarsening procedure, described by eq. (3), provides an algebraic mechanism in which the fine grid equation coefficients are averaged to produce the coarse grid equation coefficients. While this technique for improving the coarse grid approximation properties may be the preferred one,[17] it is difficult to implement and computationally costly in three dimensions, as seven-point difference stencils produced by the finite volume method on the fine grid expand to 27-point stencils on all coarser grids when standard grid transfer operators are used.[19] Convergence properties of methods employing this technique, as well as implementation issues, are discussed in detail for three-dimensional problems in the study by Dendy.[19]

An alternative is to explicitly average the coefficients in (1) or (2) to produce a new problem with smoother coefficients, essentially *smearing* the interfaces so that their effect may be captured by dis-

crete methods. The new problem is discretized on a coarser mesh, and the process is continued to produce discrete equations on a sequence of coarser meshes. These techniques are discussed in the studies of Alcouffe et al.[17] and Liu et al.[23] for two-dimensional problems.

For example, in our one-dimensional problem (6) the discrete eqs. (8) require that the function $a(x)$ be sampled at the *half*-grid points $x_{i-1/2}$ and $x_{i+1/2}$. In multigrid implementations, coarse grids are often constructed to be subsets of the next finer grid, referred to as *nested* grids. In this situation, assume that the fine grid points x_{i-1} and x_{i+1} correspond to *adjacent* coarse grid points. For discretization on the coarse grid, the function $a(x)$ must be sampled at the coarse grid half-grid point, which will correspond to the fine grid point x_i. Therefore, given the function values $a_{i-1/2}$ and $a_{i+1/2}$ we wish to produce a value a_i for use in the coarse grid discrete equations such that a_i in some sense represents the discontinuity in $a(x)$ at x_i.

Using results from homogenization theory and electrical network arguments, Alcouffe et al.[17] suggest the use of the *harmonic average*

$$a_i = \delta(a_{i-1}, a_{i+1}) \equiv \frac{2a_{i-1}a_{i+1}}{a_{i-1} + a_{i+1}}$$

to represent the coefficients across interfaces. In their article, this approach is discussed in detail for two-dimensional problems, where the harmonic average across interfaces is combined with arithmetic averages in the second grid direction to produce coefficients for coarser grids. Their extensive numerical experiments indicate that this technique is effective for regaining fast convergence rates for many types of interface problems when combined with the prolongation operators discussed below.

Note that this approach requires little extra computation over a standard discretization on coarse grids, and in the three-dimensional case results in seven-point stencils on all coarse grids.

Operator-Based Prolongation

These techniques can be explained by considering again our example (6). With two nested grids, assume we are given a coarse grid function at points that correspond to the fine grid points x_{i-1} and x_{i+1}, and we wish to *prolongate* (or *interpolate*) the coarse grid function to the fine grid points x_{i-1}, x_i, and x_{i+1}.

For the fine grid points x_{i+1} and x_{i+1} that correspond to coarse grid points, we can take the values of the new fine grid function to be equal to the coarse grid function, referred to as *injection*. To obtain the fine grid function value at the point x_i not coincident with a coarse grid point, a standard linear interpolation can be used:

$$u_i = \left(\frac{h_{i-1}}{h_{i-1} + h_i}\right) u_{i-1} + \left(\frac{h_i}{h_{i-1} + h_i}\right) u_{i+1} \quad (9)$$

On the other hand, in the case that the new point x_i is an interface point we would like to impose the continuity condition (7). We can approximate this by imposing

$$a_{i-1/2}\left(\frac{u_i - u_{i-1}}{h_{i-1}}\right) = a_{i+1/2}\left(\frac{u_{i+1} - u_i}{h_i}\right) \quad (10)$$

Solving for u_i gives the more general prolongation formula:

$$u_i = \left(\frac{h_i a_{i-1/2}}{h_{i-1}a_{i+1/2} + h_i a_{i-1/2}}\right) u_{i-1} + \left(\frac{h_{i-1}a_{i+1/2}}{h_{i-1}a_{i+1/2} + h_i a_{i-1/2}}\right) u_{i+1} \quad (11)$$

which reduces to (9) in the case that $a_{i-1/2} = a_{i+1/2}$.

This approach can be extended to two and three dimensions in a number of ways.[13,17,23] Our approach, as outlined by Hackbusch[13] for two dimensions, begins by noting that an alternative procedure for producing the more general prolongation formula (11) is by solving the ith equation of the system $L_h u_h = 0$ for u_i. The coefficients in the prolongation rule then come from the discrete stencil for the ith equation of L_h, which in the one-dimensional example (8) is

$$\left[-\frac{a_{i-1/2}}{h_{i-1}}\left(\frac{a_{i-1/2}}{h_{i-1}} + \frac{a_{i+1/2}}{h_i} + \frac{h_{i-1}c_i^- + h_i c_i^+}{2}\right) - \frac{a_{i+1/2}}{h_i}\right] \quad (12)$$

Ignoring the terms involving c_i gives the prolongation formula (11) above, while including the c_i terms provides a formula that uses additional information about the discrete differential operator. The difficulty with this approach in dimensions higher than one is that the resulting prolongation formula for u_i involves not only coarse grid points but as yet undefined fine grid points as well, unless the grids are defined in a nonstandard fashion.[13]

This difficulty can be avoided with standard nested grids in the following way. In three dimensions, one must consider four types of fine grid points in the prolongation procedure:

Type 1: Fine grid points coincident with coarse grid points.
Type 2: Fine grid points lying on a coarse grid line but not of Type 1.
Type 3: Fine grid points lying on a coarse grid plane not of Type 1 or Type 2.
Type 4: Fine grid points not on a coarse grid line or plane.

Injection is used for type 1 points. For type 2 points, dependencies in the discrete stencil corresponding to directions not on the coarse grid line are removed by compressing the three-dimensional stencil to a one-dimensional stencil (by simply summing the entries), producing a two-point prolongation formula, as in the one-dimensional case (11). A four-point prolongation formula for type 3 points results by summing away dependencies in the direction not coincident with a coarse grid plane. Type 4 points will require all six surrounding points in the prolongation formula. Note that if the prolongation is performed in the order type 1 $\rightarrow$ type 4 then all computations involve only fine grid quantities that have been previously computed by the preceding prolongation formulas.

It is common to take the restriction operator to be $I_h^H = (I_H^h)^*$, the *adjoint* of the prolongation operator with respect to the inner product

$$\langle u_h, v_h \rangle = h^d \sum_{x \in \Omega_h} u_h(x) v_h(x)$$

where d is the dimension of the problem and I_H^h is the d-dimensional version of either (9) or (11).

Multigrid Solver for the LPBE

The multigrid method developed for this study is based upon the above techniques. In particular, a three-dimensional version of the harmonic averaging procedure for coarse grid operator construction described for two-dimensional problems in Alcouffe et al.[17] was developed, and three-dimensional operator-based prolongation procedures were also implemented. These are combined with a vectorized point-wise red/black Gauss–Seidel smoother, nested iteration, and a *variable* V-cycle algorithm[12] (a V-cycle algorithm in which the number of smoothings employed is doubled each time a coarser grid is reached, yielding convergence properties similar to the W-cycle). A maximum number of grid levels are employed for each problem size (e.g., six levels in the 65 $\times$ 65 $\times$ 65 grid case, resulting in a coarse grid equation with one unknown). The resulting method is an extremely efficient numerical solution technique for the linearized Poisson–Boltzmann equation.

OTHER METHODS

Of recent investigations into numerical solution of the differential form of the LPBE, the two most efficient methods appear to be the adaptive SOR procedure described by Nicholls and Honig[8] and the incomplete Cholesky preconditioned conjugate gradient method of Davis and McCammon.[6] Consequently, it is these two methods that will be the focus of the comparison to follow.

Successive Overrelaxation

The successive overrelaxation procedure, an accelerated form of Gauss–Seidel iteration, is a product of the pioneering work of David Young to automate the process of "hand" relaxation. The classic reference is Young's book,[24] while its application to the Poisson–Boltzmann equation, along with an adaptive procedure for determining the optimal relaxation parameter, is described by Nicholls and Honig[8] and will not be discussed here.

In our comparisons with the multigrid method, we use an SOR method provided with the optimal relaxation parameter, implemented with a red/black ordering and array oriented data structures, yielding maximal vector lengths and, as will be apparent, high performance on both the Convex C240 and the CRAY Y-MP.

Preconditioned Conjugate Gradient Methods

Krylov subspace methods, of which the conjugate gradient method of Hestenes and Stiefel[25] is the most important representative, are extensively used for the iterative solution of linear systems in numerical analysis and scientific computing. Their application to the Poisson–Boltzmann equation is discussed by Davis and McCammon,[6] including comparisons with some classical iterative methods such as SOR.

If the matrix is badly conditioned, then Krylov subspace methods may require excessive iterations to converge to the solution, as they are sensitive to both the condition number of the matrix and the clustering of its eigenvalues. By *preconditioning* the linear system, the number of iterations may be brought down to yield an efficient method. Among the most effective preconditioners for linear systems arising from the discretization of partial differential equations are the incomplete factorizations; unfortunately, the very implicitness that gives these preconditioners their effectiveness also makes them difficult to vectorize on vector computers. However, the incomplete Cholesky factorizations for symmetric problems on uniform grids developed by van der Vorst and others[26] employ special orderings to improve vectorization during the back substitutions.

We present experiments with a preconditioned conjugate gradient method (implemented so as to yield maximal vector lengths and high performance), provided with four different preconditioners: (1) diagonal scaling; (2) an incomplete Cholesky factorization (the method for which Davis and McCammon present results[6]); (3) the same factorization but with a *plane-diagonal-wise ordering*[26] allowing for some vectorization of the backsolves; and (4) a vectorized

modified incomplete Cholesky factorization[26] with modification parameter $\alpha = 0.95$, which has an improved convergence rate over standard ICCG.

Note that vectorized ICCG and MICCG results for the Poisson–Boltzmann equation do not appear to have been reported previously.

TEST PROBLEMS

The first test problem consists of an acetemide molecule CH_3CONH_2 lying in a water solution. The dielectric constant $\epsilon(\mathbf{r})$ in the LPBE is set to 80 in the ionic solution region and 2 in the molecular region. The infinite domain $\mathbb{R}^3$ is truncated to a finite box $\Omega \subset \mathbb{R}^3$ containing the molecule, with boundary $\partial\Omega$, and boundary conditions are provided by the analytic solution in the case of uniform dielectric $\epsilon(\mathbf{r}) = \epsilon_w = 80$, given as

$$\Phi(\mathbf{r}) = \sum_{i=1}^{N_m} \frac{e^{-\kappa|\mathbf{r}-\mathbf{r}_i}}{\epsilon_w|\mathbf{r} - \mathbf{r}_i|} \text{ on } \partial\Omega$$

A second test problem will also be used briefly to illustrate a point in the next section. This problem is quite similar to the first problem except the molecular region consists of a cube lying completely within Ω. This test problem allows us to look at the performance of each method for various problem sizes.

NUMERICAL RESULTS

Table I describes the methods investigated in this study and provides a key to the plots and tables to follow.

Unless otherwise indicated, all data in the plots and tables to follow *include* the preprocessing costs incurred by the various methods. In other words, the multigrid times include the additional time required to set up the problem on coarse grids and the times for the conjugate gradient methods employing incomplete factorizations include the initial costs of performing the factorizations. This gives a complete and fair assessment of the total time required to reach the solution.

Table I. Linearized Poisson–Boltzmann equation solvers.

Method	Description
MG	Multigrid method
OSOR	Successive overrelaxation method with optimal relaxation parameter ω
DSCG	Diagonally scaled conjugate gradient method
ICCG1	Incomplete Cholesky preconditioned conjugate gradient method
ICCG2	Same as ICCG1 but with plane-diagonal-wise ordering during backsolves
MICCG	Same as ICCG2 but with a modification parameter $\alpha = 0.95$

An initial approximation of zero was taken to start each method, and each method used a relative residual stopping criterion

$$\frac{\|r_h^j\|}{\|f_h\|} = \frac{\|f_h - L_h u_h^j\|}{\|f_h\|} < \text{TOL} = 1.0e - 6$$

where u_h^j represents the jth iterate. Normally, $\|r_h^j\|$ is not available in the preconditioned conjugate gradient iteration (the quantity $\langle Cr_h^j, r_h^j \rangle^{1/2}$ is available, where C is the preconditioner), and must be computed at extra cost; however, this additional cost was not included in the conjugate gradient timings to avoid unfairly penalizing the conjugate gradient methods.

Timings and Megaflop Rates

Timings, operation counts, and megaflops (one million floating point operations per second) figures on the CRAY Y-MP were obtained from the performance-monitoring hardware accessed through *perftrace* and *perfview*. Timing figures on the Convex C240 were obtained from the system timing routine `getrusage`, and megaflop rates were computed from the exact operation counts provided earlier by the CRAY.

Figure 2 gives the reduction in the relative residual per time work unit for each method on the Convex C240 (a work unit was chosen to be the time taken to perform a matrix-vector operation on the given architecture, equal to 0.246 CPU s on the Convex C240). Figure 3 gives the corresponding information on the CRAY Y-MP (with a time work unit of 0.0151 CPU s). In Table II, the information from Figures 2 and 3 is translated into a single number for each method, representing the *total time required to reach the solution* on a given architecture. Table III gives the *performance in megaflops* for each method, with and without preprocessing costs such as matrix construction and Cholesky factorizations.

These graphs and tables show that multigrid is nearly two times faster than the next best method, MICCG. It is interesting to note from Table II that optimal SOR is in fact equal or superior to all of the conjugate gradient methods for this problem except MICCG. Table III indicates that our implementation of the optimal SOR method is exceptionally efficient, operating at near the peak rate available from FORTRAN of matrix-vector operations on the CRAY Y-MP. In addition, the vectorized incomplete Cholesky preconditioned conjugate gradient methods execute with high rates, consistent with the earlier reports[26] for these methods on the CRAY X-MP.

Our results are consistent with an earlier study by Dendy and Hyman,[27] who compared multigrid to nonvectorized forms of ICCG and MICCG for two-

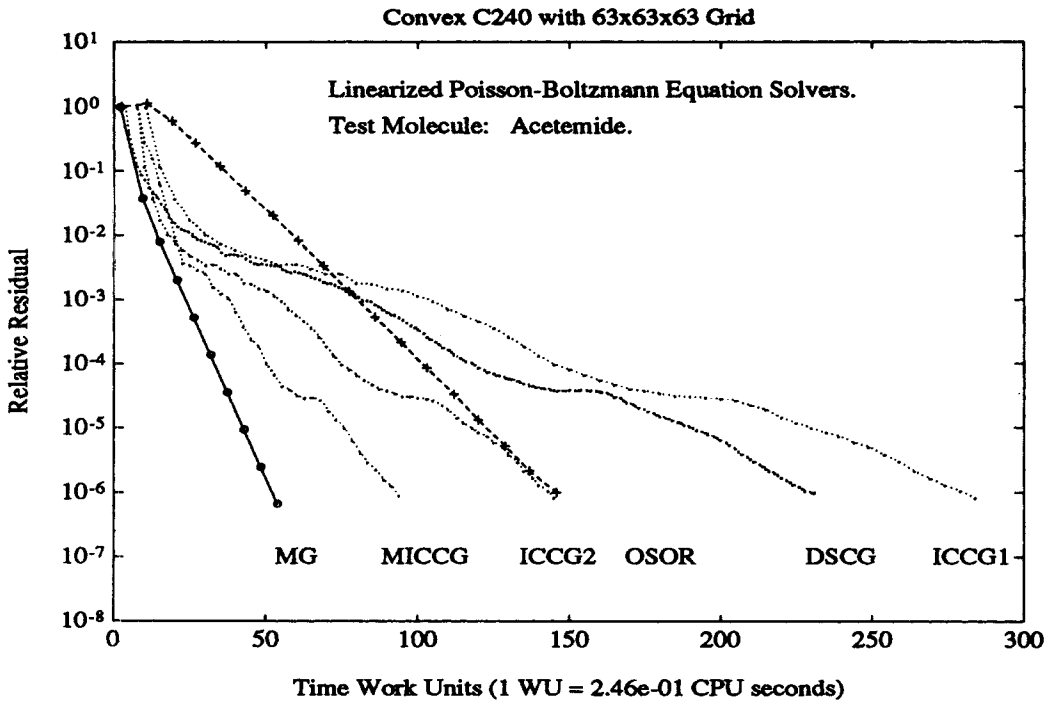

Figure 2. Relative residual reduction per time work unit on the Convex C240.

dimensional interface problems, including the multigroup neutron diffusion problem. Their conclusions for the two-dimensional case were that the multigrid method developed by Alcouffe et al.[17] was superior to both ICCG and MICCG. One observation they made, which we did not take advantage of in this study, was the following: *Multigrid reaches discretization error accuracy rapidly*—with far fewer iterations than that required to reach a small residual tolerance. In fact, one can prove for model problems that a small fixed number of V-cycle iterations, coupled with an initial guess provided by nested iteration, reaches an approximation to the continuous problem with accuracy on the order of discretization error.[13]

While the implementations presented here may also be used for general second-order problems in three dimensions, in the case of the Poisson–Boltz-

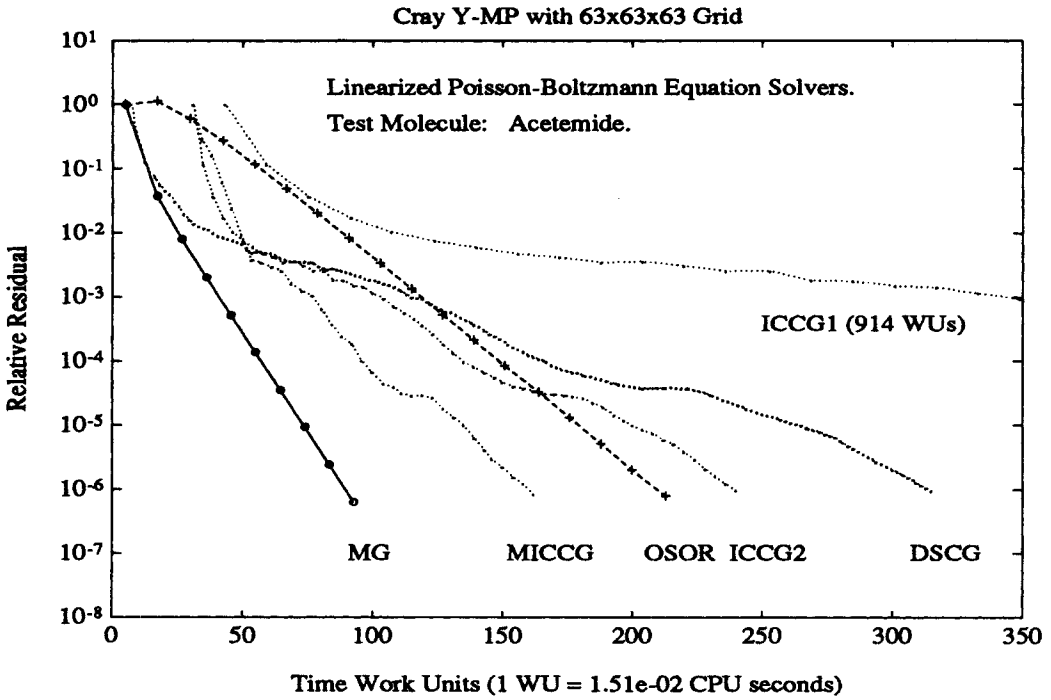

Figure 3. Relative residual reduction per time work unit on the CRAY Y-MP.

Table II. Total time to reach solution with a 65 × 65 × 65 grid (CPU s, with relative residual tolerance of 1.0*e*-6).

Machine (1 processor)	Method					
	MG	MICCG	OSOR	ICCG2	DSCG	ICCG1
Convex C240	13.2	23.1	35.9	35.7	56.8	69.9
CRAY Y-MP	1.40	2.45	3.22	3.62	4.76	13.8

Table III. Megaflop rates with [and without] matrix construction and Cholesky factorization preprocessing.

Machine (1 processor)	Method					
	MG	MICCG	OSOR	ICCG2	DSCG	ICCG1
Convex C240	12.3 [12.7]	13.1 [13.6]	18.9 [20.4]	13.6 [14.1]	18.1 [18.5]	7.24 [7.05]
CRAY Y-MP	118 [120]	135 [158]	215 [220]	142 [158]	215 [218]	36.8 [35.5]

mann equation special techniques may be used to increase solver efficiency. In particular, the highly optimized SOR method of Nicholls and Honig,[8] using a technique referred to as *stripping*, along with a novel procedure for determining the optimal relaxation parameter adaptively, achieves a factor of two improvement over our "unstripped" optimal SOR method (35.9 CPU s, from Table II). Their code solves the same problem on the Convex C240 in 17.3 CPU s, compared to 13.2 CPU s for our multigrid implementation.

It should be stressed that their optimization techniques may be used to equal advantage with the multigrid method presented here, as it is based upon a red/black Gauss–Seidel smoothing iteration; therefore, we would expect a similar (factor of two) improvement in multigrid speed. However, it is unclear how to take advantage of their *stripping* technique in the preconditioning phases of the incomplete Cholesky conjugate gradients methods, which in our experiments made up more than 60% of the total execution times of these methods (more than 85% in the nonvectorized ICCG case).

Computational Complexity

Multigrid methods are provably optimal order for a broad class of problems, meaning that the cost to solve a problem with N unknowns is proportional to N. Unfortunately, the discontinuities of $\epsilon(\mathbf{r})$ in the LPBE preclude the use of the existing theory, which requires strong smoothness assumptions on the problem coefficients.[11–13] However, the optimal order behavior may be investigated empirically, which is what the final set of experiments is designed to do.

Figure 4 gives the cost of each method to solve the second test problem, in time work units on the Convex C240, as the problem size is increased by a factor of two beginning with a 17 × 17 × 17 grid and ending with a 129 × 129 × 129 grid. Note that

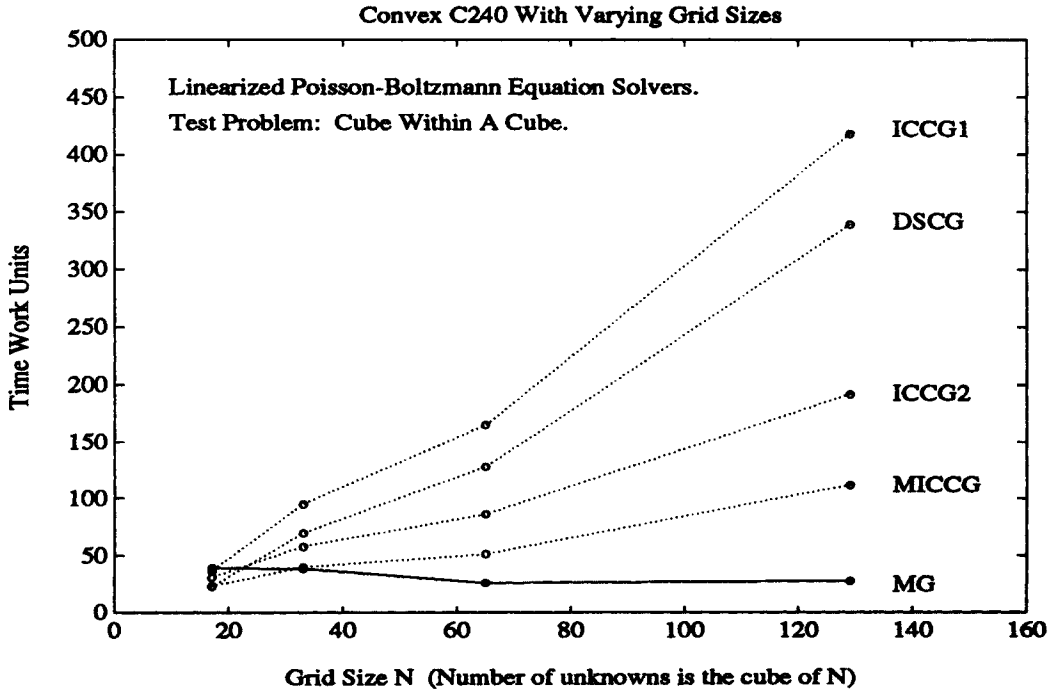

Figure 4. Cost per unknown as a function of grid size.

in this figure the *time per unknown* is being plotted as a function of the problem size. The fact that the multigrid curve is virtually horizontal reflects the optimal order behavior of multigrid. In particular, we can see that the superiority of multigrid increases as we move to larger grids. This behavior can often be demonstrated for the multigrid method even when the existing theory is no longer applicable.

CONCLUSIONS

The first conclusion to be drawn from the numerical evidence presented earlier is that the multigrid method is the most efficient method presented for the two test problems with a grid size of 65 × 65 × 65. Second, the advantage of multigrid *grows with the problem size*, as it demonstrates optimal order behavior for our test problems.

A point that should be stressed is that the SOR and conjugate gradient performance results reported here are based upon highly optimized codes, and these codes ran close to their respective maximal rates on both architectures. Based upon earlier results,[15,16] we expect a fully optimized multigrid method to run at rates comparable to the smoothing iteration alone on architectures such as those considered here. Thus, we expect that the multigrid method can achieve nearly the peak rate obtained by the SOR iteration, shown in Table III. As a consequence of these points, it should be clear that the results presented here for the multigrid method are conservative and give only an indication of its potential for the LPBE.

Finally, while we considered only the LPBE in this article the multigrid method can be extended to nonlinear problems through either a combination of Newton's method and the linear multigrid algorithm presented here or a nonlinear multigrid algorithm.[13] These methods have been used successfully for nonlinear problems in computational fluid mechanics[28] and semiconductor device simulation,[29] and their application to the NPBE will be investigated in a future article.

Dr. Anthony Nicholls of Columbia University made much effort to supply the authors with the molecule data for testing, and they also benefited greatly from his experience with this problem and expertise with the electrostatics program Delphi, developed in the laboratory of Dr. Barry Honig. In addition, the authors would like to thank Dr. Shankar Subramaniam and Dr. Richard Kozack of the Beckmann Institute at the University of Illinois and Dr. Robert Skeel in the Computer Science Department at the University of Illinois for several helpful discussions on computational biophysics and the Poisson–Boltzmann equation. The vectorizable plane-diagonal-wise ordered incomplete Cholesky factorization preconditioners were obtained from the NETLIB numerical software library with the help of Dr. Henk van der Vorst (the author of these preconditioning codes). The National Center for Supercomputer Applications, located at the University of Illinois at Urbana-Champaign, provided access to the CRAY Y-MP through support by the National Science Foundation. Access to the Convex C240 was provided by the Computing Services Organization, also at the University of Illinois at Urbana-Champaign. This work was supported in part by Department of Energy Grant DOE DE-FG02-91ER25099. The authors thank the reviewers for several helpful comments that led to improvements of this article.

References

1. P. Debye and E. Hückel, *Physik. Z.*, **24,** 185 (1923).
2. M.E. Davis, J.D. Madura, B.A. Luty, and J.A. McCammon, *Comp. Phys. Comm.*, **62,** 187 (1990).
3. C. Niedermeier and K. Schulten, technical report, Department of Physics and Beckman Institute, University of Illinois at Urbana-Champaign, 1990.
4. K.A. Sharp, *J. Comp. Chem.*, **12,** 454 (1991).
5. C. Tanford, *Physical Chemistry of Macromolecules*, John Wiley & Sons, New York, 1961.
6. M.E. Davis and J.A. McCammon, *J. Comp. Chem.*, **10,** 386 (1989).
7. A.H. Juffer, E.F.F. Botta, B.A.M. van Keulen, A. van der Ploeg, and H.J.C. Berendsen, *J. Comp. Phys.*, **97,** 144 (1991).
8. A. Nicholls and B. Honig, *J. Comp. Chem.*, **12,** 435 (1991).
9. K.A. Sharp and B. Honig, *Annu. Rev. Biophys. Biophys. Chem.*, **19,** 301 (1990).
10. B.J. Yoon and A.M. Lenhoff, *J. Comp. Chem.*, **11,** 1080 (1990).
11. R.E. Bank and T.F. Dupont, *Math. Comp.*, **36**(153), 35 (1981).
12. J.H. Bramble and J.E. Pasciak, *Math. Comp.*, **49**(180), 311 (1987).
13. W. Hackbusch, *Multi-Grid Methods and Applications*, Springer-Verlag, Berlin, 1985.
14. A. Brandt, *Math. Comp.*, **31,** 333 (1977).
15. M. Holst and F. Saied, in *Proceedings of the Third IMACS Symposium on Computational Acoustics*, North Holland, Amsterdam, 1991.
16. M. Holst and F. Saied, Technical Report UIUCDCS-R-91-1697, University of Illinois at Urbana-Champaign, 1991.
17. R.E. Alcouffe, A. Brandt, J.E. Dendy, Jr., and J.W. Painter, *SIAM J. Sci. Stat. Comp.*, **2**(4), 430 (1981).
18. J.E. Dendy, Jr., *J. Comp. Phys.*, **48,** 366 (1982).
19. J.E. Dendy, Jr., *SIAM J. Sci. Stat. Comp.*, **8**(2), 673 (1987).
20. A. Behie and P. Forsyth, Jr., *IMA J. Numer. Anal.*, **3,** 41 (1983).
21. A. Behie and P. Forsyth, Jr., *Soc. Petr. Eng. J.*, **23,** 623 (1983).
22. R.S. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1962.
23. C. Liu, Z. Liu, and S. McCormick, technical report, Computational Mathematics Group, University of Colorado at Denver, 1991.
24. D.M. Young, *Iterative Solution of Large Linear Systems*, Academic Press, New York, 1971.
25. M.R. Hestenes and E. Stiefel, *J. Res. NBS*, **49,** 409 (1952).
26. H.A. van der Vorst, *SIAM J. Sci. Stat. Comp.*, **10**(6), 1174 (1989).
27. J.E. Dendy, Jr. and J.M. Hyman, in *Elliptic Problem Solvers*, M. Schultz, Ed., Academic Press, New York, 1981, p. 247–253.
28. A. Brandt, Technical Report GMD-Studien 85, GMD, Bonn, 1984.
29. R.E. Bank and D.J. Rose, *Math. Comp.*, **39**(160), 453 (1982).