Solving the Finite Difference Linearized Poisson-Boltzmann Equation: A Comparison of Relaxation and Conjugate Gradient Methods

M. E. Davis and J. A. McCammon

Department of Chemistry, University of Houston, Houston, Texas 77204-5641

Received 21 June, 1988; accepted 2 November 1988

Comparisons have been made between relaxation methods and certain preconditioned conjugate gradient techniques for solving the system of linear equations arising from the finite-difference form of the linearized Poisson-Boltzmann equation. The incomplete Cholesky conjugate gradient (ICCG) method of Meijerink and van der Vorst has been found to be superior to relaxation methods, with at least a factor of two improvement in speed, and only a 50% increase in storage.

I. INTRODUCTION

Since its application to the dielectric cavity model of proteins by Warwicker and Watson,[1] the finite-difference method of calculating electrostatic potentials has become an important tool for understanding the behavior of macromolecules. Warwicker and Watson's routine and variants of it have been applied to numerous problems of chemical and biological significance. Molecular stability,[1,2] molecular binding,[3] molecular attraction,[4] diffusional reaction rates,[5–9] and changes in pKa[10] have been studied.

The method is extremely powerful, but the price for that power must be paid in computer time. The amount of computer power required arises from the very nature of a finite-difference problem on a large three-dimensional grid. The linearized Poisson-Boltzmann equation,

$$-\nabla \cdot \varepsilon \nabla \phi + \varepsilon \kappa^2 \phi = \rho \,, \tag{1}$$

becomes a set of linear equations,

$$\begin{aligned} q_{i,j,k}/h = {} & \varepsilon_{xi-1,j,k}[\phi_{i,j,k} - \phi_{i-1,j,k}] \\ & + \varepsilon_{xi,j,k}[\phi_{i,j,k} - \phi_{i+1,j,k}] \\ & + \varepsilon_{yi,j-1,k}[\phi_{i,j,k} - \phi_{i,j-1,k}] \\ & + \varepsilon_{yi,j,k}[\phi_{i,j,k} - \phi_{i,j+1,k}] \\ & + \varepsilon_{zi,j,k-1}[\phi_{i,j,k} - \phi_{i,j,k-1}] \\ & + \varepsilon_{zi,j,k}[\phi_{i,j,k} - \phi_{i,j,k+1}] \\ & + h^2 \varepsilon_s \kappa^2_{i,j,k} \phi_{i,j,k} \end{aligned} \tag{2}$$

when finite differenced, where $\varepsilon_{xi,j,k}$, $\varepsilon_{yi,j,k}$, and $\varepsilon_{zi,j,k}$ are the permitivities along the grid lines between grid point (i,j,k) and the points $(i + 1,j,k)$, $(i,j + 1,k)$ and $(i,j,k + 1)$, respectively; $\kappa_{i,j,k}$ and $q_{i,j,k}$ are the Debye-Hückel parameter and charge at the grid point (i,j,k); and h is the grid spacing. The task is to solve this system of N^3 equations in N^3 unknowns: the potential at the grid points of the $N \times N \times N$ lattice. With 65 a typical value for N and with computing time a valuable resource, it is important that the most efficient methods be brought to bear.

Currently, successive over-relaxation (SOR) is the method most widely in use in this field, though some report using strictly Jacobi iteration.[11] Jacobi iteration is considered "not practical because it converges too slowly," by Press et al., in *Numerical Recipes*.[12] SOR is a significant improvement, but only for a small range of the relaxation parameter is its optimal performance attained.

In this work we compare the relaxation methods to some alternatives utilizing preconditioned conjugate techniques. As has been demonstrated previously in other fields, for example Kershaw's work in laser fusion code,[13] these alternatives run in significantly shorter times while requiring only a slightly larger amount of memory. A factor of at least two can be gained over optimal

CCC 0192-8651/89/030386-06$04.00

SOR and an order of magnitude over Gauss-Seidel can be achieved by one of these alternatives: the incomplete Cholesky conjugate gradient (ICCG) method. Memory requirements are only increased to $9N^3$ from $6N^3$. If memory limitations are stringent the code can be modified to take the vectors into core in blocks. The underlying preconditioned conjugate gradient methods are also used widely in finite-element packages. These methods code and vectorize easily, run fast, and require no parameter optimization to achieve their noteworthy results.

II. METHODS

Relaxation Methods

Relaxation methods appeal to chemists' and physicists' intuition. Analogous to a diffusion process, local disturbances propagate with each iteration from nearest neighbors to next nearest and so on to their distant effects. No doubt this feeling has contributed to their popularity. A complete description of these methods may be found in Young's *Iterative Solution of Large Linear Systems.*[14]

The basic concept of relaxation methods is straightforward. Move through the lattice and at each point assume the surrounding points to be correct and from the finite-difference equation calculate the change in the value at the point in question. In Jacobi relaxation, the changes are held until the end of a complete pass through the grid. In Gauss-Seidel (GS) relaxation, on the other hand, the updates are used as soon as they are available. A significant improvement on the convergence rates of these methods can be made by realizing that further iterations will shift the value again. So in anticipation of further corrections, the change is multiplied by a factor, ω, before it is applied. These successive overrelaxations, for which the SOR method is named, lead to much better convergence rates. The trick of course is picking the factor ω, for it is in the proper choice of ω that SOR gains its speed.

As we shall show, the convergence rate of SOR is strongly dependent upon the choice of ω, and the range of values for which the algorithm is optimal is small. In addition, the optimal value is only optimal in the limit as the process converges. A better selection allows ω to vary with the iteration. In our algorithm, we implement Chebyshev acceleration and odd-even ordering. Thus the algorithm we use is

```
$\omega = 1$
For $n = 1$ to convergence
  For $i = 2$ to $N - 1, j = 2$ to $N - 1, k = 2$ to $N - 1$
    If $(i + j + k \bmod 2) = (n \bmod 2)$ Then
      $\phi_{i,j,k} = \phi_{i,j,k} - \omega( \varepsilon_{\kappa i,j,k}\phi_{i,j,k} - q_{i,j,k}/h$
                          $- \varepsilon_{xi-1,j,k}\phi_{\mathrm{i}-1,j,k} - \varepsilon_{xi,j,k}\phi_{i+1,j,k}$
                          $- \varepsilon_{yi,j-1,k}\phi_{i,j-1,k} - \varepsilon_{yi,j,k}\phi_{i,j+1,k}$
                          $- \varepsilon_{zi,j,k-1}\phi_{i,j,k-1} - \varepsilon_{zi,j,k}\phi_{i,j,k+1})$
    End If
  End For
  If $n = 1$ Then
    $\omega = 1/(1 - \frac{1}{2}\rho^2)$
  Else
    $\omega = 1/(1 - \frac{1}{4}\rho^2\omega)$
  End If
End For
```

where ρ is ideally the spectral radius for the Jacobi iteration, and $\varepsilon_{\kappa i,j,k}$ is the sum of the adjacent ε_x, ε_y, and ε_z's plus $\varepsilon_s \kappa^2_{i,j,k}$. The code used is a modification of the routine given in *Numerical Recipes.*[12]

The optimal value for ρ can be determined analytically in certain cases. Alternately, it can be determined experimentally for a given equation, grid, and boundary condition. Typical studies of the sort done for electrostatics in molecules, however, vary the coefficients and therefore the change the equation. Only a small number of different sources might be considered, so the multiple run necessary to optimize ω would consume all of the time saved. As a result only approximations to the optimum are feasible in practice.

Conjugate Gradient Methods

Conjugate gradient techniques are most familiar to chemists and physicists for minimization. The large system of linear equations resulting from the finite-difference equations can also be expressed as a minimization problem. The solution of $\mathbf{Ax} = \mathbf{b}$ minimizes the expression $\frac{1}{2}\mathbf{x} \cdot A\mathbf{x} - \mathbf{x} \cdot \mathbf{b}$. In the case of the finite-difference scheme the elements of the vector, $\mathbf{x}$, are the values of ϕ at the N^3 grid points. The matrix $\mathbf{A}$ has the form shown in Figure 1.

Preconditioning is a refinement of the conjugate gradient technique, especially useful in cases like finite-difference equations on a regular lattice for which $\mathbf{A}$ is a very regular, very sparse matrix. The preconditioning consists of approximating the given matrix $\mathbf{A}$ by a matrix that is easily invertible.

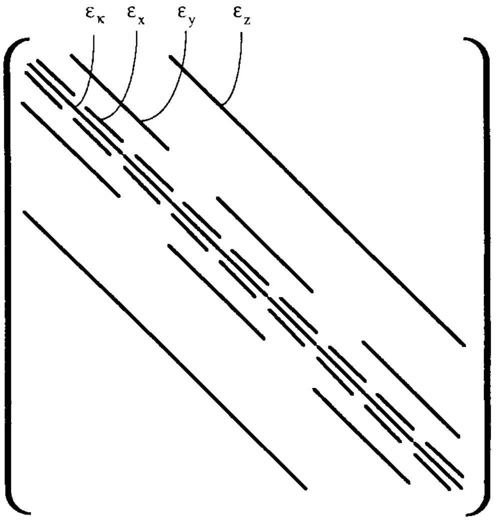

Figure 1. Sparsity pattern of **A**, the coefficient matrix arising from the finite-difference form of linearized Poisson-Boltzmann equation, for the case $N = 3$. ε_{xijk}, ε_{yijk}, and ε_{zijk} are the permitivities along the grid lines between the point (i, j, k) and the points $(i + 1, j, k)$, $(i, j + 1, k)$ and $(i, j, k + 1)$, respectively. The diagonal elements $\varepsilon_{\kappa ijk}$ are the sum of the adjoining ε_x, ε_y, and ε_z's plus $\varepsilon_s \kappa^2_{ijk}$, where κ is the Debye-Hückel parameter for the grid point (i, j, k).

The basic algorithm is

```
$\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}$
$\mathbf{p}_0 = \mathbf{M}^{-1}\mathbf{r}_0$

For $i = 0$ to convergence

    $\alpha = (\mathbf{r}_i \cdot \mathbf{M}^{-1}\mathbf{r}_i)/(\mathbf{p}_i \cdot \mathbf{Ap}_i)$
    $\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha\mathbf{p}_i$
    $\mathbf{r}_{i+1} = \mathbf{r}_i - \alpha\mathbf{Ap}_i$
    $\beta = (\mathbf{r}_{i+1} \cdot \mathbf{M}^{-1}\mathbf{r}_{i+1})/(\mathbf{r}_i \cdot \mathbf{M}^{-1}\mathbf{r}_i)$
    $\mathbf{p}_{i+1} = \mathbf{M}^{-1}\mathbf{r}_{i+1} + \beta\mathbf{p}_i$

End For
```

The optimal choice of $\mathbf{M}$ must strike a balance between approximating $\mathbf{A}$ and being easily invertible. The closer $\mathbf{M}$ resembles $\mathbf{A}$ the fewer iterations required, but the larger the problem of inverting $\mathbf{M}$. The simplest choice is to take $\mathbf{M}$ to be the diagonal of $\mathbf{A}$. As a further refinement we can take $\mathbf{M}$ to be the tridiagonal of $\mathbf{A}$ since a tridiagonal system is easily solved. The best method we have come upon is a very powerful technique developed by Meijerink and van der Vorst[15]: the incomplete Cholesky factorization.

"Complete" Cholesky factorization factors the matrix $\mathbf{A}$ into the form $\mathbf{LL}^T$. The inversion is then simply a matter of forward and backward substitution. In the process of factorization, however, the sparseness of $\mathbf{A}$ is lost. The "incomplete" factorization ignores these "fill-ins" and keeps the same sparseness pattern for $\mathbf{L}$ as was present in $\mathbf{A}$. For the particular sparseness of this problem, Meijerink and van der Vorst suggest a decomposition of the form $\mathbf{LDL}^T$ where $\mathbf{D}$ is a diagonal matrix whose elements are the reciprocals of the diagonal elements of $\mathbf{L}$. The off-diagonal elements of $\mathbf{L}$ are equal to the corresponding positions in $\mathbf{A}$ while the diagonal elements of $\mathbf{D}$, the reciprocal of the diagonal elements of $\mathbf{L}$, are given by

$$d_i^{-1} = \varepsilon_{\kappa i} - \varepsilon^2_{xi-1}d_{i-1} - \varepsilon^2_{yi-N}d_{i-N} - \varepsilon^2_{zi-N}d_{i-N^2}.$$

Meijerink and van der Vorst have a good review of incomplete decompositions.[16] For a thorough investigation of preconditioned conjugate gradient techniques see Chaps. 1 and 7 of Axelsson and Parker.[17]

III. COMPARISONS

For the comparison we have chosen a very simple system: a point charge near a low dielectric cavity. The potential is forced to vanish at the boundary. Comparisons were made for 20^3, 40^3, and 60^3 grids, with the radii and positions scaled to correspond to finer meshes for the same problem. In all cases the calculations were carried out on a VAX 8650 using single-precision-compiled Fortran. CPU times were determined by calls to the VMS functions LIB$INIT_TIMER and LIB$STAT_TIMER respectively before and after the subroutine which calculates the potential. For the preconditioned conjugate gradient methods, this subroutine included the decomposition step. On the other hand, the set up time for the dielectric and ion grids, which is independent of the method, has not been included in the convergence comparisons. These set up times ran about 0.1, 0.6, and 2.1 seconds for the 20^3, 40^3, and 60^3 grids. Memory usage was $6N^3$ for the relaxation methods and $10N^3$ for the preconditioned conjugate gradient methods. The preconditioned conjugate gradient methods would have needed only $9N^3$ storage, however, if the charge grid had not been saved. The routines all converge to the solution of the system of linear equations. So, convergence was measured as the fractional reduction in the norm of the residue, or $|\mathbf{Ax} - \mathbf{b}|$, from the guess of $\phi = 0$ to the final refinement. (For a discussion of deviations from the analytical results due to discretization error see Gilson et al., Ref. 11.) Finally, multiple runs were performed to measure the precision of the timings, and the deviations were found to be smaller than the size of the plotted data points.

Optimal values for ρ were obtained by running the program to a convergence of 10^{-2} for numerous runs to search for the shortest running time. The optimal values were found to be 0.9858, 0.9985 for the 20^3, 40^3, and 60^3 grids, respectively. The GS method was implemented by running the SOR code with ρ set to zero. Note that for the relaxation techniques the outermost grid points are the boundary, while for the conjugate gradient methods the boundary is one grid point beyond the actual stored grid.

IV. RESULTS

Figures 2, 3, and 4 show convergence versus time for 20^3, 40^3, and 60^3 grids. In all cases diagonal and tridiagonal preconditioned conjugate gradient techniques and optimal SOR run in comparable time, while ICCG converges twice as quickly. GS is significantly slower and worsens in comparison as the grid increases. At $N = 20$ GS takes at least five times longer than ICCG, and by the time $N = 60$ is reached it takes well over an order

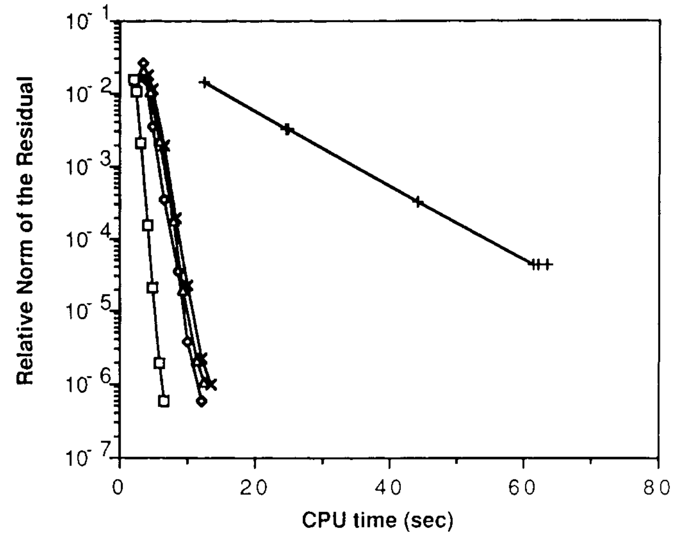

Figure 2. Convergence rates for the test problem on a 20^3 grid using GS, Gauss-Seidel (+); SOR, successive overrelaxation (◇); diagonal preconditioned conjugate gradient (×); tridiagonal preconditioned conjugate gradient (△); and ICCG, incomplete Cholesky conjugate gradient (□).

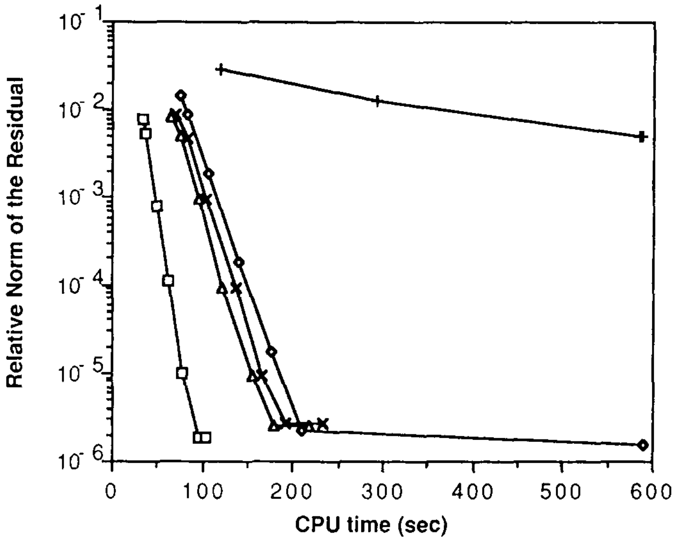

Figure 3. Convergence rates for the test problem on a 40^3 grid using GS, Gauss-Seidel (+); SOR, successive overrelaxation (◇); diagonal preconditioned conjugate gradient (×); tridiagonal preconditioned conjugate gradient (△); and ICCG, incomplete Cholesky conjugate gradient (□).

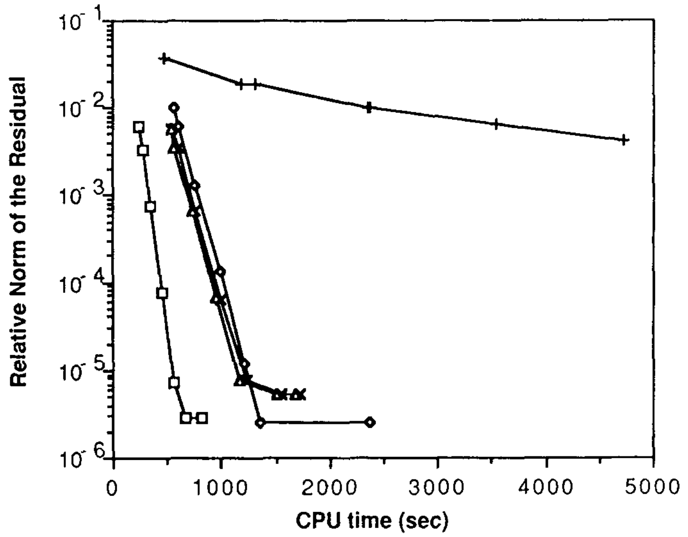

Figure 4. Convergence rates for the test problem on a 60^3 grid using GS, Gauss-Seidel (+); SOR, successive overrelaxation (◇); diagonal preconditioned conjugate gradient (×); tridiagonal preconditioned conjugate gradient (△); and ICCG, incomplete Cholesky conjugate gradient (□).

of magnitude longer to converge. The flattening of the convergence curves arises from reaching the rounding limits in the regions of large electric field.

Examination of the potential grids, using HYDRA, has shown that the convergence proceeds most quickly at the charge. With SOR, the convergence propagates radially as the disturbance produced by the introduction of the point charge diffuses across the grid. With ICCG, the propagation's approximate spherical symmetry is broken by the incompleteness of the decomposition, which introduces extra diagonals into the matrix **M**. These additional couplings help the convergence to spread along various diagonals radiating from the charge. By the time the rounding limits are reached, however, the contribution to the residual at each point is essentially random for both methods.

In Figure 5, we see an example of the dependence of the convergence of the SOR method upon ρ. Note that small changes can result in much longer convergence times,

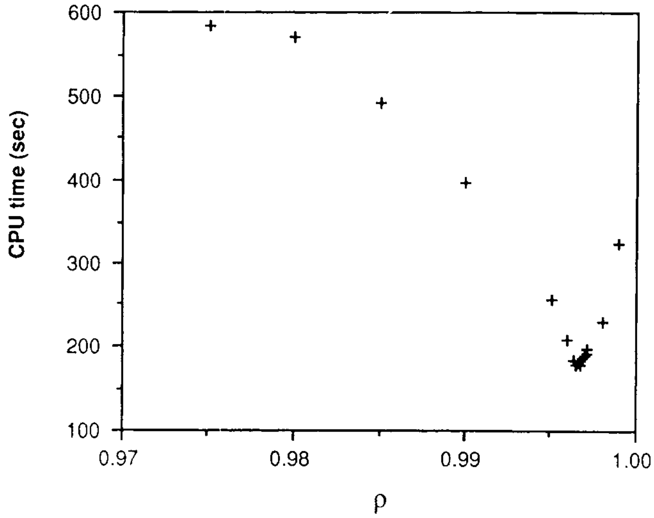

Figure 5. Dependence of SOR convergence upon ρ for the 40^3 grid.

especially to the low side of the optimal value. The optimal value for ρ increases with grid size and Debye length, $1/\kappa$. The window for ρ, on the other hand, narrows with increasing grid size.

V. CONCLUSIONS

For relaxation methods, overrelaxation provides a significant improvement in convergence time, especially with increasing grid size. The choice of overrelaxation parameter, however, is critical. It is important to remember that the times given in Figures 2–4 are for *optimal* SOR. As shown in Figure 5 these speeds can quickly slip by significant factors with small changes in ρ_J. The optimal value depends upon boundary conditions, grid size, and equation. The dependence upon the equation requires a new optimization for changes in the salt concentration, through changes in κ, and in the molecule's position, through changes in ε. Optimization for a certain problem would only be an approximation to the optimum for a different molecular geometry, dielectric constants, or salt concentration when the Debye–Hückel term is included. The time for determining the optimal ω would be wasted unless different source terms were to be studied.

Conjugate gradient methods, on the other hand, are faster than optimized SOR and do not require any parameter optimization. Vectorization of the preconditioned conjugate gradient routine is straightforward since the algorithm deals in vector quantities, while Ashcraft and Grimes[18] describe a fully vectorizable implementation of the forward and backward substitutions involved in evaluating $\mathbf{M}^{-1}\mathbf{r}$. In addition, boundary conditions more complicated than $\phi = 0$, including periodic,[16] are easily handled. So, we recommend ICCG as the method of choice for linearized Poisson-Boltzmann problems. As presented, however, ICCG deals only with linear equations and so is not directly applicable to the full Poisson-Boltzmann equation. But, since the conjugate gradient method works for nonlinear systems, a variant may prove useful.

We are grateful to Dr. Giles Auchmuty for helpful discussions. This work has been supported in part by the NIH, the Robert A. Welch Foundation, and the Texas Advanced Research Program. J.A.M. is the recipient of the G.H. Hitchings Award from the Burroughs Wellcome Fund.

References

1. J. Warwicker and H.C. Watson, *J. Mol. Biol.*, **157,** 671–679 (1982).
2. N.K. Rogers and M.J.E. Sternberg, *J. Mol. Biol.*, **174,** 527–542 (1984).
3. J. Warwicker, D. Ollis, F.M. Richards, and T.A. Steitz, *J. Mol. Biol.*, **186,** 645–649 (1985).
4. I. Klapper, R. Hagstrom, R. Fine, K. Sharp, and B. Honig, *Proteins,* **1,** 47–59 (1986).
5. S.A. Allison, R.J. Bacquet, and J.A. McCammon, *Biopolymers,* **27,** 251–269 (1988).
6. S.H. Northrup, J.O. Boles, and J.C.L. Reynolds, *J. Phys. Chem.*, **91,** 5991–5998 (1987).
7. S.H. Northrup, J.A. Luton, J.O. Boles, and J.C.L. Reynolds, *Comp. Aid. Mol. Des.*, **1,** 291–311 (1988).
8. K.A. Sharp, R. Fine, and B. Honig, *Science,* **236,** 1460–1462 (1987).
9. J.A. McCammon, *Science,* **238,** 486–491 (1987).
10. M.J.E. Sternberg, F.R.F. Hayes, A.J. Russel, P.G. Thomas, and A.R. Fersht, *Nature,* **330,** 86–87 (1988).
11. M.K. Gilson, K.A. Sharp, and B.H. Honig, *J. Comp. Chem.*, **9,** 327–335 (1988).
12. W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vettering, *Numerical Recipes,* Cambridge University Press, Cambridge: 1986.
13. D.S. Kershaw, *J. Comp. Phys.*, **26,** 43–65 (1978).
14. D.M. Young, *Iterative Solution of Large Linear Systems,* Academic Press, New York; 1971.
15. J.A. Meijerink and H.A. van der Vorst, *Math. Comp.*, **31,** 148–162 (1977).
16. J.A. Meijerink and H.A. van der Vorst, *J. Comp. Phys.*, **44,** 134–155 (1981).
17. O. Axelsson and V.A. Barker, *Finite Element Solution of Boundary Value Problems,* Academic Press, Orlando, 1984.
18. C.C. Ashcraft and R.G. Grimes, *SIAMI J. Sci. Stat. Comput.*, **9,** 122–151 (1988).