Iterative Methods in Semiconductor Device Simulation

CONOR S. RAFFERTY, MARK R. PINTO, STUDENT MEMBER, IEEE, AND ROBERT W. DUTTON, FELLOW, IEEE

Abstract—**This paper examines iterative methods for solving the semiconductor device equations. The emphasis is on fully coupled methods, because of the failure of decoupled methods for on-state devices. Using the PISCES-II device simulator as a vehicle, incomplete factorization and operator decomposition iterative methods are presented for solving the Newton equations. The dependencies of these methods on factors such as choice of variables, bias condition and initial guess are analyzed. The results are compared with sparse Gaussian elimination.**

I. Introduction

With the advent of virtual memory systems, the primary limitation on two-dimensional device simulation is the computer time required. Most of that time is spent solving large linear systems, typically containing several thousand equations. The most reliable solution method is Gaussian elimination, but the solution time increases rapidly ($\approx n^{1.5-1.75}$) with the number of grid points, and very rapidly with the number of independent variables ($\approx m^3$) per node.

Large linear systems are routinely solved by iteration in other fields, but iterative methods have not been widely adopted for solving the full set of device equations for a number of reasons. Many of the physical quantities vary over a huge range, leading to arithmetic difficulties. Device behavior is usually governed by a fine balance of drift and diffusion currents, which demands that the equations be solved to high precision. If several independent variables are solved simultaneously, the resulting matrix is not symmetric, nor is it diagonally dominant, nor positive definite. Nevertheless, by choosing a suitable preconditioner, it is possible to use iterative solvers.

To open the discussion and to allow comparison with other codes, the grid and discretization is presented in Sections II and III. The nonlinear behavior of the Newton iteration is studied in Section IV. In Section V the iterative methods used are presented and their dependence on bias condition and choice of independent variables is analyzed. Finally the methods are compared for a practical example and conclusions are drawn.

II. Grid

The PISCES-II [1] device simulator uses an adaptive triangular grid. Two facts motivated this choice. First, modern device structures are increasingly complex (Fig. 1). Fitting a rectangular grid to this structure is at least as difficult as generating triangles. This overrides a principal objection to irregular grids, the additional programming complexity, for the irregularity is inherent in the problem. Secondly, the economical allocation of grid is the most direct way to reduce simulation time. By using a triangular grid, the possibility of local refinement is allowed without adding unnecessary nodes in other parts of the device.

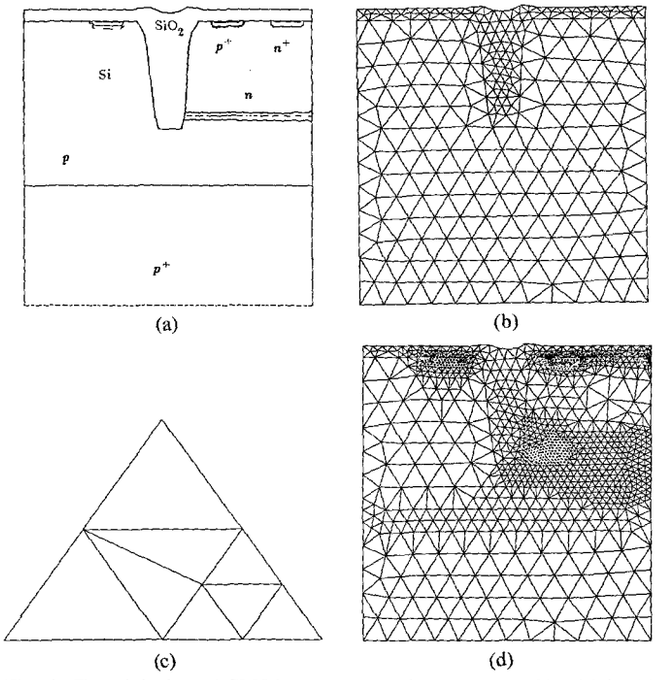


Fig. 1. Trench isolaated CMOS process. (a) Cross section. (b) Initial grid. (c) Refinement procedure. (d) Working grid.

The grid in PISCES-II is created in several steps. Starting with a list of boundary nodes specified by the user, a coarse grid is generated to cover the topology of the structure (Fig. 1(b)). This initial grid is then refined according to the doping profile before computing the equilibrium solution. The grid may be refined at any time after this, using the variation of any physical variable to determine where refinement is necessary (Fig. 1(d)).

Triangles which require refinement are subdivided into four congruent subtriangles by joining the midpoints of the sides. (Fig. 1(c)). These subtriangles may in turn be refined. The refinement strategy follows the proposals in [2] quite closely, with some modifications to control the number of obtuse triangles generated. However the decision as to where and when refinement is done is managed quite differently from a multigrid code such as [2]. Refinement is governed by the variation of physical variables

Manuscript received February 12, 1985; revised June 19, 1985.

C. S. Rafferty and R. W. Dutton are with Integrated Circuits Laboratory, Stanford University, Stanford, CA 94305.

M. R. Pinto was with Stanford University, Stanford, CA. He is now with AT&T Bell Laboratories, Murray Hill, NJ.

0018-9383/85/1000-2018$01.00 © 1985 IEEE

rather than the truncation error, because the latter is both expensive and difficult to estimate accurately on the coarse grids used in any practical problem. More significantly, the grid is updated only occasionally and at the user's request. This decision reflects the fact that the refinement itself is expensive, and would completely dominate the solution time if it were invoked several times during each iteration.

The most troublesome problem with triangular grids is to control the triangle shapes. Obtuse triangles can and do cause unphysical spikes in the solution, with disastrous effects on the convergence rate. It is very laborious to generate a grid entirely free of obtuse triangles, and we know of no method to refine it without introducing new obtuse angles. Fortunately, it is unnecessary to remove all obtuse angles; it turns out that by examining the diagonal separating each pair of triangles and redrawing it if the two angles it subtends add to more than π, one can avoid spikes [3]–[5]. The importance of doing this is borne out by Fig. 2, which illustrates a simulation that foundered on an uncorrected grid, but had no difficulty when the triangle edges were redrawn according to this principle. The ordinate is the electron quasi-Fermi potential in a MOSFET, early in solving a linear bias point. The maximum applied bias is 1 V, but the internal spike is a about 50 V in magnitude.

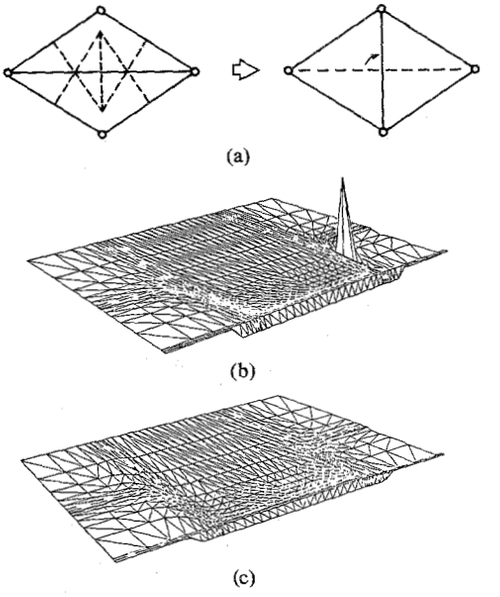


Fig. 2. Obtuse triangle problem. (a) Procedure for correcting obtuse neighbors. Electron quasi-Fermi potentials for a MOSFET during the solution procedure for (b) an uncorrected grid and (c) a corrected grid. The voltage spike is approximately 50 V.

III. Discretization

PISCES-II is a finite difference code. The key to finite differences on an irregular grid is to use the generalized box discretization [6], [4]. The semiconductor device equations can be written

$$\nabla \cdot (\epsilon \nabla \psi) - \rho \equiv N_\psi = 0 \tag{1a}$$

$$\frac{1}{q}\nabla \cdot \boldsymbol{J}_n - U_n - \frac{\partial n}{\partial t} \equiv N_n = 0 \tag{1b}$$

$$\frac{1}{q}\nabla \cdot \boldsymbol{J}_p + U_p + \frac{\partial p}{\partial t} \equiv N_p = 0 \tag{1c}$$

where ψ is the electrostatic potential, ρ is the charge, ϵ is the dielectric permittivity, $\boldsymbol{J}_n$ and $\boldsymbol{J}_p$ are the electron and hole currents, U_n, U_p are the net electron and hole recombination rates, and n, p are the electron and hole densities. The current $\boldsymbol{J}_n$ can be written

$$\boldsymbol{J}_n = q\mu_n(-n\nabla\psi + kT/q\nabla n)$$

where the Einstein relation has been assumed for simplicity in the exposition and μ_n is the electron mobility. Substitutng from $n_i e^{q(\psi - \phi_n)/kT}$, where ϕ_n is the electron quasi-Fermi potential and n_i is the intrinsic concentration, the current can also be written

$$\begin{aligned}\boldsymbol{J}_n &= -q\mu_n n_i e^{q(\psi-\phi_n)/kT}\nabla\phi_n \\ &= n_i kT\mu_n e^{q\psi/kT}\nabla\Phi_n\end{aligned}$$

where $\Phi_n = e^{-q\phi_n/kT}$. Similar relations hold for the hole current.

In the box method, each partial differential equation is integrated over a small polygon surrounding each node, the polygon being defined by the perpendicular bisectors of the triangle sides (Fig. 3). The divergence operators are integrated using Green's formula, so that, for instance, the discretized electron continuity equation (for the node shown) is

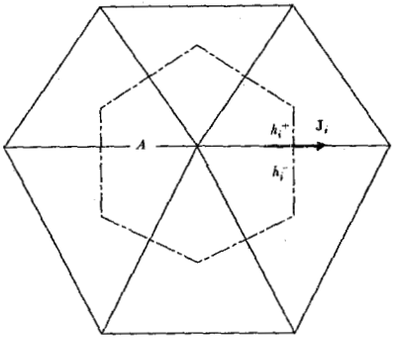


Fig. 3. Area associated with a node for generalized box discretization.

$$\frac{1}{q}\sum_i J_i(h_i^+ + h_i^-) - A\left(U_n + \frac{\partial n}{\partial t}\right) = 0$$

where J_i is the outward normal current on each face of the polygon, $h_i^+ + h_i^-$ is the length of that face, and A is the area of the polygon. The J_i are evaluated using the Scharfetter–Gummel formula [7]. For Poisson's equation, the outward normal displacements are computed using centered differences.

PISCES-II takes a finite element approach to the equation assembly. That is, the summations indicated above are computed on an element-by-element basis. This vastly simplifies the treatment of boundaries and corners in the simulation domain—in fact they are handled by the same algorithms used in the bulk. As a result, the assembly for

a general mesh is less complicated than that reported for a rectangular mesh in [8].

One problem arises for obtuse triangles, where the perpendicular heights h_i^+, h_i^- become negative. This is where the obtuse correction mentioned in the previous section comes into play. If the triangles have been arranged to satisfy the angle criterion, it can be shown [4], [9] that the *net* perpendicular height $h_i = h_i^+ + h_i^-$ is always positive. This ensures that current flow is always in the physical direction, and spurious spikes are avoided. Another benefit of this arrangement is that the integration polygon is actually the Vornoi polygon of the node it contains. That is, it is the area of the plane closer to that node than to any other.

The set of nonlinear equations which result from this discretization can be solved in terms of any three independent physical variables. Historically, the first set of variables used in device simulation was the potential and two carrier concentrations (ψ, n, p) [10]. Slotboom [11] introduced the exponentials of the quasi-Fermi potentials, Φ_n and Φ_p, because the continuity matrix takes on a symmetric positive definite form under this transformation. The quasi-Fermi potentials themselves (ψ, ϕ_n, ϕ_p) are frequently used, to reduce the numerical range of the variables. In the next section, it will be shown that the choice of variables has significant consequences for the solution process.

IV. Nonlinear Iteration

As the set of equations to be solved is nonlinear, there is no deterministic path to the solution, and iteration is necessary. Two iterative methods are widely used: the decoupled iteration originally used by Gummel [10] and the coupled Newton iteration, which seems to have been first used by [12]. In Gummel's method, the Poisson equation is solved with the quasi-Fermi potentials held fixed, then the continuity equations are solved using the new values of the potential. The cycle then begins again with the new quasi-Fermi potentials just computed. The method works well when the equations are decoupled, for instance in diodes with reverse or medium forward bias. However when the drift term tightly couples the Poisson and continuity equations, for instance in active transistors or even resistors, convergence becomes slow, as is well known in one- or two-dimensional work [3]. Curiously, this problem has not been reported in recent three dimensional papers [14]. We have also found the Gummel iteration extraordinarily slow for transient problems, at least when using a straightforward fully implicit time discretization. Other time discretization schemes are available, but they can impose severe restrictions on the timestep, or introduce other numerical problems [15], [16].

The Newton method includes the coupling between the different equations explicitly and is free of these difficulties. The price of that stability is a ninefold increase in matrix size. This leads to both memory and time restrictions on the size of problems which can be solved.

The primary motivation for this investigation was the steady state and transient analysis of parasitic SCR paths in CMOS structures [17], [18]. In view of the poor convergence of Gummel's method for transient high current simulations, we confine our attention to the Newton iteration (though the Gummel method is included in the PISCES-II simulator).

Newton's Iteration

The Newton iteration proceeds by stacking the equations (1a)–(1c) into one vector known as the Newton residual ($\boldsymbol{N}$). The Jacobian $\boldsymbol{J}$ of this vector with respect to the independent variables is computed, and the equation

$$\boldsymbol{J}\Delta x = -\boldsymbol{N} \tag{2}$$

is solved for the update vector Δx. The linear system (2) can be solved either directly by (sparse) Gaussian elimination, or by linear iterative methods. The independent variables x are then updated

$$x \leftarrow x + \Delta x \tag{3}$$

and a new Newton iteration begins. Note that an independent set of nested inner iterations may be required for each outer Newton loop if an iterative method is used to solve (2). In one dimension the most time-consuming part of this procedure is the assembly of the Jacobian; in two or three dimensions the solution of the linear system (2) dominates, typically by at least an order of magnitude. In PISCES-II, the Jacobian is always evaluated by computing the derivatives with respect to ψ, n, p; the other sets of variables are implemented by subsequently applying the chain rule. When using the Slotboom variables, the matrix must be scaled as the entries are of order $e^{qV_{\text{applied}}/kT}$ and would cause overflow. Several scalings are possible; we use

$$(\boldsymbol{J}\boldsymbol{D}^{-1})(\boldsymbol{D}\Delta x) = -\boldsymbol{N}$$

where $\boldsymbol{D}$ is a diagonal matrix, containing 1 for Poisson rows, $e^{q\psi/kT}$ for electron rows, and $e^{-q\psi/kT}$ for hole rows. (The scaling is done implicitly, since the intermediate terms cannot be evaluated.) Scaling is also necessary using the quasi-Fermi potentials, because the continuity rows are proportional to the local carrier concentrations. To avoid ill-conditioning each continuity row is scaled by its carrier concentrations.

In the following subsection, the convergence behavior of the nonlinear Newton iteration procedure will be analyzed. Since Gaussian elimination gives an "exact" solution of the linear system (2) (subject to roundoff error), it will be used exclusively.

Convergence

For the purpose of illustration, an idealized problem is considered throughout this section, a one-dimensional diode with a p^+ Gaussian implant into an n-type substrate. The grid is rectangular and aligned with the axis of the device. The convergence criterion is $10^{-5}kT/q$ on both the electrostatic and quasi-Fermi potential updates. Table I shows the number of Newton iterations required to solve

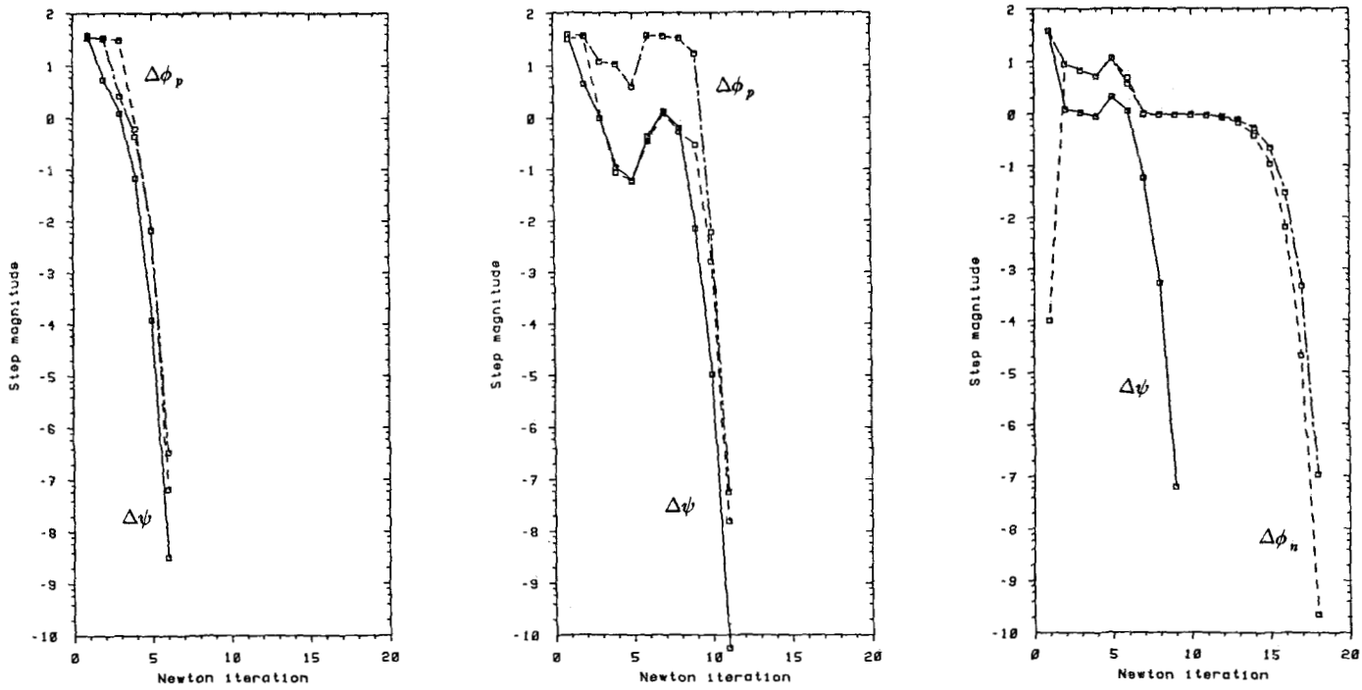


Fig. 4. Size of Newton updates as a function of iteration number for (a) carrier concentrations, (b) Slotboom variables, (c) quasi-Fermi potentials.

TABLE I
NONLINEAR CONVERGENCE VERSUS CHOICE OF VARIABLE

	Forward bias						
	0→0.1	0.2	0.3	0.4	0.5	0.6	0.7
(ψ,n,p)	5	5	5	6	6	8	10
(ψ,ϕ_n,ϕ_p)	8	9	9	9	9	10	10
(ψ,Φ_n,Φ_p)	6	7	7	8	8	8	9

	Reverse bias							
	0→-0.1	-0.2	-0.5	-1	-2	-5	-10	-20
(ψ,n,p)	5	5	6	7	8	10	11	14
(ψ,ϕ_n,ϕ_p)	9	12	15	18	–	–	–	–
(ψ,Φ_n,Φ_p)	5	6	7	9	10	12	14	20

the ideal problem. Results for different bias steps and the different choices of variables are shown. Dashes indicate that a solution was not obtained, in this case due to an artifact of the assembly routine related to carrier concentrations underflowing.

One immediately observes that the iteration depends quite strongly on the choice of variables. The least number of iterations is required using the carrier concentrations, and the Slotboom variables are comparable. The quasi-Fermi potentials usually require significantly more work. Part, but not all, of the low convergence rate is due to damping (see below). The other reason is shown in the next figure (Fig. 4). The size of the Newton updates as a function of iteration count is plotted for each set of variables. The updates are measured in kT/q units of change in potential or quasi-Fermi potential. All the methods eventually converge quadratically, strongly suggesting that the Jacobian is correct and accurate. However, the quasi-Fermi updates are almost exactly a single kT/q for many iterations; it was also observed that the Newton residual decreased linearly rather than quadratically during this phase, though none of the damping mechanisms are active. The cause of this "self-limiting" phenomenon is not known.

For all but the smallest bias steps, convergence requires significant damping of the Newton update Δx, particularly in reverse bias. In the case of the carrier concentrations, it is enough to set negative concentrations, which occur during the iteration, to zero. These values become positive as the solution is approached. With the Slotboom variables, the treatment of negative concentrations is different; instead of setting them to zero considerably faster convergence is observed if they take on the same values as at the previous iteration. In addition, Bank–Rose damping [19] is required to ensure convergence. Damping is most critical using the quasi-Fermi potentials; in the initial stages of the iteration, minority quasi-Fermi potential updates of 10^7 V occur in the neighborhood of contacts. In these cases pure Bank–Rose damping is insufficient; the damping step necessary to reduce the contact spike is so small that all internal updates vanish, and no progress is made towards the solution. Instead it was found that truncating the quasi-Fermi potentials to the minimum and maximum applied bias, and *then* applying damping was preferable.

Condition

Although a direct solution method (sparse Gaussian elimination) was chosen for this study, it could be that the properties of the linear solver are intruding on the nonlinear behavior, through ill-conditioning. To examine this possibility the condition estimator [20] was used. Before applying the estimator, the Jacobian was row-scaled by its

TABLE II
JACOBIAN CONDITION NUMBER

Newton loop	0	1	2	3	4
(ψ,n,p)	2.7×10^{15} (*)	5.6×10^{15} (*)	8.2×10^{14} (4.4)	3.8×10^{15} (10.5)	3.0×10^{15} (0.56)
(ψ,ϕ_n,ϕ_p)	3.5×10^{3} (19.7)	1.4×10^{15} (1.1×10^{3})	4.5×10^{6} (3.4×10^{3})	5.3×10^{5} (1.0)	8.1×10^{4} (1.0)
(ψ,Φ_n,Φ_p)	8.4×10^{19} (*)	9.8×10^{16} (*)	1.3×10^{16} (*)	4.2×10^{13} (5.35)	6.2×10^{8} (0.64)

diagonal elements[1] to ensure more reliable estimates. For the bias step $0 \rightarrow -0.5$ the following conditions were reported for the first five Newton iterations (Table II). In each column, the first number is the estimated condition of the Jacobian. The figures in parentheses denote the maximum quasi-Fermi update for that loop before damping, in kT/q units; an asterisk represents the occurrence of negative concentrations. Although the condition numbers are high, for the most part they will not cause breakdown when working with double precision (17 digits). In particular, a severe overshoot using the quasi-Fermi potentials is observed with a condition as low as 4.5×10^6, implying that the problem is *not* in the linear solver but in the Newton iteration itself.

Initial Guess

In the above discussion, the initial guess was generated by modifying the previous solution (equilibrium there) only at the contacts. Another frequently used initial guess is to fix the majority quasi-Fermi potential at the applied bias throughout heavily doped regions which adjoin a contact, and to set the potential at the corresponding charge-neutral position. The minority quasi-Fermi potential is unaffected. While usually a better approximation to the true solution, this embellishment has little effect on the Newton iteration. A related idea is to perform one or more Gummel loops prior to entering the Newton iteration; but we do not know of any satisfactory criterion for switching from the Gummel to the Newton. Most of the heuristics we have tried, such as waiting for the residual to decrease by a certain factor or taking a fixed number of loops, either do not work for all devices or prove more expensive than Newton's method alone. The most effective initial guess we have found is linear extrapolation from a pair of previous solutions, which reduces the work in generating I–V curves by a factor of 2–3.

V. LINEAR ITERATION

The direct Newton algorithm described above is satisfyingly stable with respect to device operating condition. However, solving a large linear system directly at each iteration (even using a sparse package) is quite expensive, and therefore iterative (as opposed to direct) methods for solving (2) are now considered. In general iterative methods are not applied directly to the system of interest, but to a related system which is significantly easier to solve. The related system is obtained by splitting the Jacobian into a part that is "easy" to invert, $\boldsymbol{A}$, and a remainder $\boldsymbol{B}$.

[1]Actually by the nearest power of two to avoid introducing additional roundoff.

$$\boldsymbol{J} = \boldsymbol{A} + \boldsymbol{B} \tag{4}$$

The iterative method is then applied to the system

$$(\boldsymbol{J}\boldsymbol{A}^{-1})(\boldsymbol{A}\Delta x) = -\boldsymbol{N} \tag{5}$$

instead, solving for $y \equiv \boldsymbol{A}\Delta x$. If the invertible part $\boldsymbol{A}$ is sufficiently close to $\boldsymbol{J}$, then $\boldsymbol{J}\boldsymbol{A}^{-1}$ is close to the identity and this system should be much easier to solve than the original (2).

In the following, the splittings and iterative methods used will be surveyed. The results are then presented and characterized in terms of the spectrum of the matrix $\boldsymbol{J}\boldsymbol{A}^{-1}$, and directly in terms of the convergence rate.

Splittings

The splittings considered were

a) None.
This corresponds to applying iteration directly to the original system. Such "point" iterative schemes are not usually a practical strategy but are of interest in connection with multigrid methods.

b) Operator (Block) Splitting.
This splitting takes advantage of the structure of the Jacobian to obtain an invertible part which is sometimes close to $\boldsymbol{J}$. If the Jacobian (in terms of the carrier concentrations for instance) is written as

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial N_\psi}{\partial \psi} & \frac{\partial N_\psi}{\partial n} & \frac{\partial N_\psi}{\partial p} \\ \frac{\partial N_n}{\partial \psi} & \frac{\partial N_n}{\partial n} & \frac{\partial N_n}{\partial p} \\ \frac{\partial N_p}{\partial \psi} & \frac{\partial N_p}{\partial n} & \frac{\partial N_p}{\partial_p} \end{bmatrix}$$

then the operator splitting sets $\boldsymbol{A}$ equal to the three diagonal blocks of this matrix, that is,

$$\boldsymbol{A} = \begin{bmatrix} \frac{\partial N_\psi}{\partial \psi} & 0 & 0 \\ 0 & \frac{\partial N_n}{\partial n} & 0 \\ 0 & 0 & \frac{\partial N_p}{\partial p} \end{bmatrix}.$$

This splitting, which we call the block splitting, was proposed in device simulation by [21] and [22]; similar splittings have been used in mechanical engineering [23]. The matrix $\boldsymbol{A}$ is "easy to invert" relative to the Jacobian because each block is only of order $N \times N$ so the cost of factorizing A is $\approx \frac{1}{9}$ that of factorizing $\boldsymbol{J}$. The technique works well when the off-

diagonal blocks are small relative to the diagonal, thus it can be expected to work well under the same conditions as Gummel iteration. In fact by completely ignoring the off-diagonal blocks one would obtain a method closely related to the "single-Poisson" variant of Gummel's method proposed in [4].

c) Incomplete Factorization/(Knot) Splitting.

A very different splitting is obtained if, in the course of Gaussian elimination of the Jacobian, fill-in is ignored. An approximate LU factorization of $\boldsymbol{J}$ is generated, which implicitly defines the splitting matrix $\boldsymbol{A}$. Such methods were first proposed, (again in mechanical engineering), by [24], [25], but did not become widely popular until the mid-seventies. In these applications the system to be solved usually had "nice" properties, for instance the matrix was Stieljes or at least symmetric positive definite.

Although the system under consideration is far from these ideals, we have implemented an incomplete factorization method by regrouping the Jacobian in 3×3 blocks, each corresponding to the three equations and variables at a single node (knot). The factorization is carried out by considering the Jacobian as an $N \times N$ matrix of 3×3 blocks and applying Gaussian elimination to this blocked matrix. Each arithmetic operation in the usual algorithm is replaced by the corresponding 3×3 matrix operation. The resulting decomposition differs from the usual ILU decomposition in that several extra inter-equation couplings are retained, resulting in direct coupling at each node. It is therefore convenient to think of this procedure as "knot" splitting in contrast to the block (operator) splitting described above.

Iterative Methods

The two iterative schemes considered are Jacobi and conjugate residual (GCR) iteration. Jacobi iteration [6] has a slight advantage in simplicity, while conjugate residual [26] iteration is much more stable. In the examples which follow, the grid is rectangular with the natural ordering, so the behavior of SOR or Gauss–Seidel is similar to Jacobi iteration. The stopping criterion on the inner loop is chosen as suggested in [19] to maintain quadratic convergence in the Newton loop.

Results

A graphic way to analyze the behavior of iterative methods is to examine the eigenvalue spectrum of the matrix $\boldsymbol{JA}^{-1}$. The closer the eigenvalues are to unity, the easier it is to solve (5). If any eigenvalue is more than one unit distant from unity, the Jacobi iteration will diverge. If there are few distinct eigenvalues, the conjugate residual method will converge rapidly.

No preconditioning: Fig. 5 shows the eigenvalue spectrum when no preconditioning is used for each choice of variables. That is, the graphs show the eigenvalues of the matrix $\boldsymbol{JD}^{-1}$, where $\boldsymbol{D}$ is the diagonal of $\boldsymbol{J}$. The eigenvalues are complex because the Jacobian is asymmetric. The picture is taken at the first Newton iteration with 0.1 V forward bias on the diode.

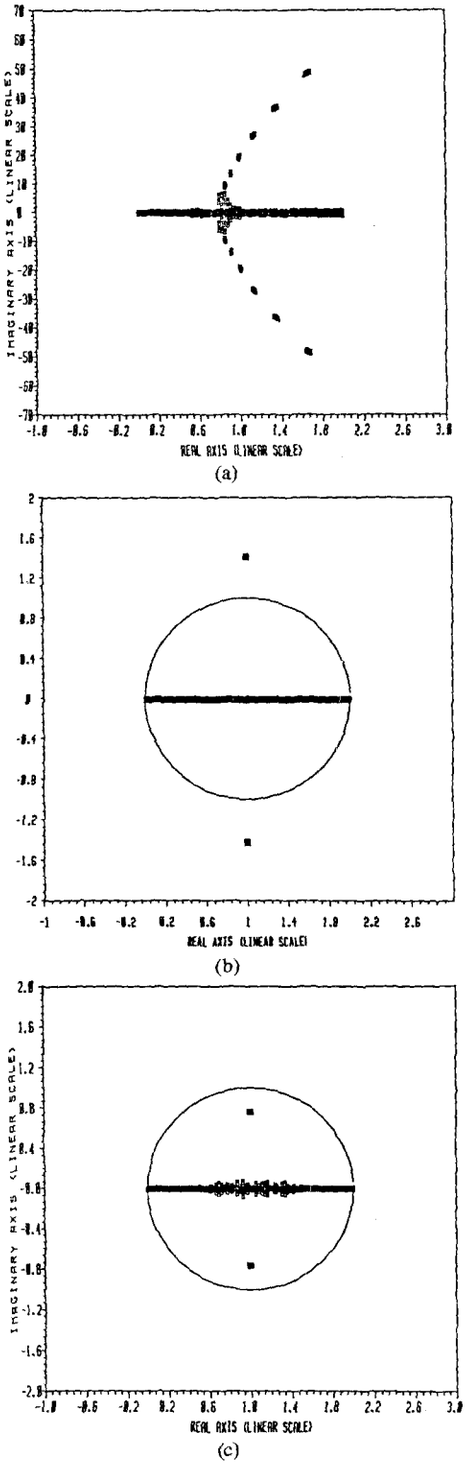


Fig. 5. Eigenvalue spectra for point-Jacobi iteration (no preconditioning) applied to a p-n diode at 0.1 V for (a) carrier concentrations, (b) Slotboom variables, (c) quasi-Fermi potentials.

For the carrier concentrations and Slotboom variables, some eigenvalues lie outside the circle $1 + e^{i\theta}$. In the case of the carrier concentrations some of the eigenvalues are as large as $\pm 70i$ and the unit circle is invisible at that scale.

As a consequence point Jacobi iteration for solving the Newton equations will diverge. For the quasi-Fermi potentials all values lie inside the circle, and in fact this is the case for every bias condition examined, suggesting that this matrix may have some interesting theoretical properties. Regardless of the range of the eigenvalues, the richness of each spectrum ensures that the conjugate residual iteration will also have little success.

Block Splitting: Fig. 6 shows similar results for the block (operator) splitting. Even at this low bias, eigenvalues outside the unit circle are observed for any choice of variables, indicating that Jacobi diverges. Again this is most pronounced using the carrier concentrations. The spectrum does improve as the Newton iteration converges, so that by carrying out some initial smoothing, it might be possible to induce the Jacobi iteration to converge, as reported in [27]. The conjugate residual method converges rapidly with either the Slotboom or quasi-Fermi variables. As one might expect from the small number of eigenvalues, less than ten iterations are required for forward bias less than 0.6 V, giving a substantial speed advantage over Gaussian elimination. However, the Gummel method also behaves well in this range, and in fact it is found that the Gummel and block methods are very similar in execution time. For higher applied bias, both methods suffer from a rapid increase in the number of iterations required. (The spectrum then becomes very rich, although the size of the eigenvalues does not increase.) The block method also suffers from the same difficulties for transient problems as does Gummel's method.

ILU/Knot Splitting: Finally the incomplete LU factorization using knot splitting is considered. The Jacobi iteration with this splitting was not competitive with the conjugate residual method in any of its forms. Therefore, in the analysis that follows, the conjugate residual method was used exclusively.

Similar to [28] a very compressed spectrum is found, with no eigenvalues outside the unit circle. There is less difference between the different variables than for the previous examples, but the Slotboom and quasi-Fermi variables retain some advantage over the concentration variables. Fig. 7 shows the spectrum for the (ψ, n, p) variables; the other spectra are similar. This splitting is less susceptible to bias variation than the block method because the equation-to-equation coupling is partly retained in the incomplete factorization. Fig. 8 shows the 1-norm of the residual as a function of the inner loop count; each break in the curve signifies the start of a new Newton iteration. The first two panels display convergence using the Slotboom variables at 0.4- and 0.9-V forward bias, and show that the linear convergence rate at 0.9 V is nearly half that at 0.4 V. (With the block splitting, the ratio is 1:16). The third panel shows 0.9 V using the concentration variables; the convergence is about 50 percent slower than using either of the other variables. One significant disadvantage of the GCR iteration is the need to maintain orthogonality between successive directions. This requires storing a number of the previous directions, ideally all. In practice not more than a dozen or so can be kept, which slows convergence. The last panel shows the same test problem as the third but with 18 rather than 6 previous directions retained; the cyclic pattern and slow convergence rate are less pronounced.

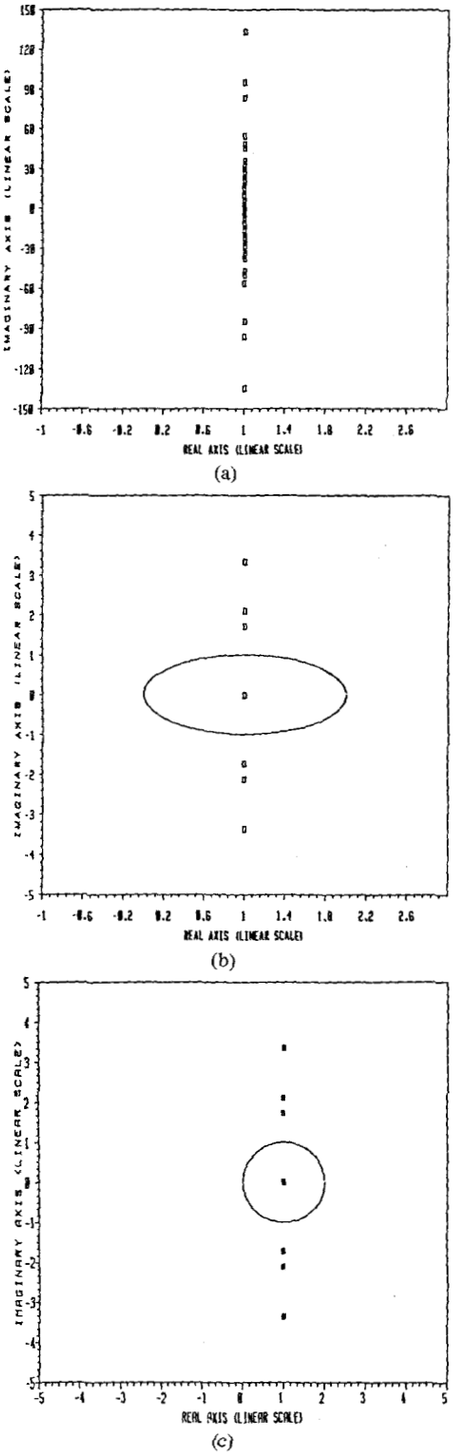


Fig. 6. Eigenvalue spectra for block (operator) splitting applied to p-n diode at 0.1 V for (a) carrier concentrations, (b) Slotboom variables, (c) quasi-Fermi potentials.

Finally, it is noted that the orthomin variant of the con-

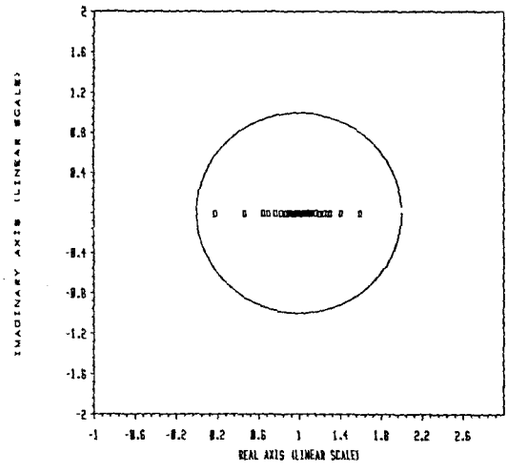


Fig. 7. Eigenvalue spectrum for ILU using knot splitting applied to p-n diode at 0.1 V for carrier concentrations.

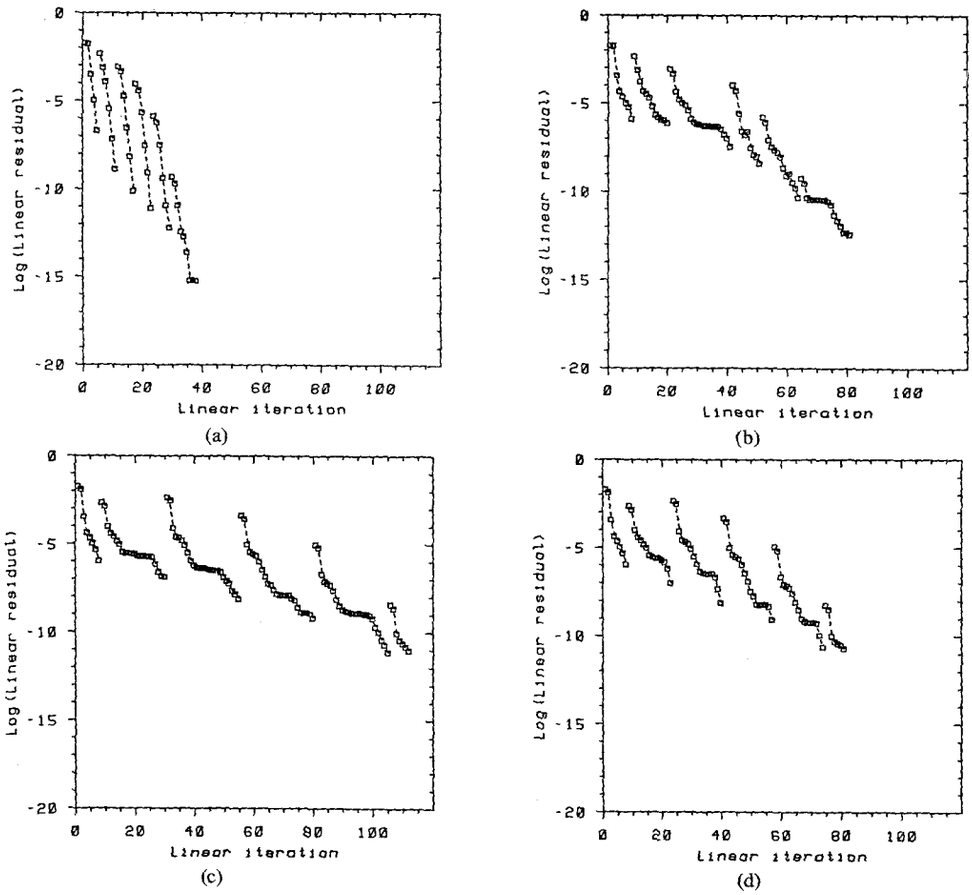


Fig. 8. Linear convergence for ILU/knot method applied to p-n diode (a) at 0.4 V using Slotboom variables, (b) at 0.9 V using Slotboom variables, (c) at 0.9 V using carrier concentrations and 6 back vectors, (d) at 0.9 V using carrier concentrations and 18 back vectors.

jugate residual iteration [26] gave essentially the same results as the restarted GCR variant used here. Adding more matrix elements to the incomplete factorization, as suggested in [29], [30], made surprisingly little difference to the number of iterations required, until 85–90 percent of the complete factorization had been attained. Experiments with conjugate gradient squared as implemented in [31] were even less successful.

VI. Comparison of Methods

To draw conclusions, one must consider application of these methods to a more practical device. Fig. 9 shows a

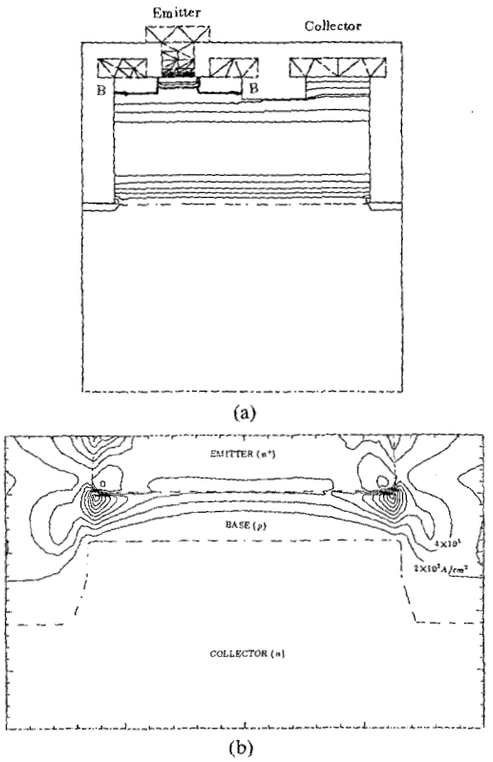


Fig. 9. Sub-micron bipolar transistor (a) cross section, (b) current contours at base-emitter voltage of 0.9 V. Significant current crowding at the corners of the emitter is evident.

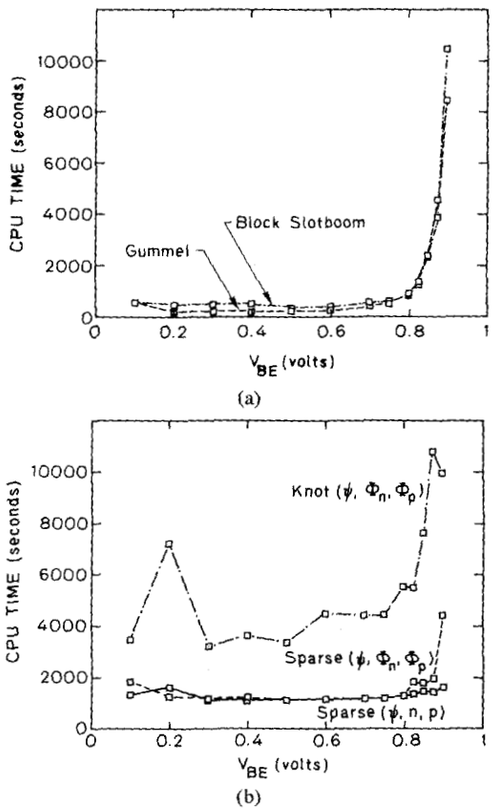


Fig. 10. CPU time versus base-emitter voltage for the submicron bipolar transistor in Fig. 9 for (a) Gummel's method and block splitting using Slotboom variables, (b) ILU/knot method using Slotboom variables and direct Newton (sparse Gaussian elimination) using Slotboom variables and carrier concentrations.

submicron bipolar device being studied at Stanford [32]. The lateral dimensions have been shrunk to the point where they are comparable with the junction depths. The device shows strong two-dimensional effects, particularly for collector resistance and current crowding. Fig. 9(b) shows a close-up of the intrinsic region of the device, with a base–emitter bias of 0.9 V. Contours of hole current density are plotted at intervals of 2×10^3 A/cm^2. The base current is dominated by injection through the corners of the emitter, degrading the gain. Such effects can be reduced by accurate control of the emitter and extrinsic base profiles.

This device was simulated in steady state on a grid of around 1300 nodes, stepping the base–emitter voltage in 0.1 V increments initially and 0.025 V increments at higher bias. Identical simulations were performed using Gummel's method, the block method with Slotboom and quasi-Fermi potentials, the ILU/knot method with the same choices of variables, and a sparse direct method with all choices of variables. For Gummel's method, ICCG was used to solve the Poisson equation, while a sparse direct method was used for the continuity equations. The sparse code is from [33] with the minimum degree ordering from [34]. The direct method was considerably accelerated by a Newton–Richardson technique (see [1] for details).

Fig. 10 illustrates the performance of each method; it is split into two panels for clarity, the first giving Gummel's method and the block iteration, the second showing the sparse direct and ILU/knot iterative methods. CPU times are for a VAX-780[2] with floating point accelerator, running UNIX[3] 4.2BSD.

In low level injection, the block splitting performs very well compared to the sparse direct code; however Gummel's method is still better in this bias range. As the built-in voltage of the base-emitter junction (0.9 V) is approached, both Gummel and block iterative methods become rapidly more expensive; it is concluded that the block method is in effect a decoupled method albeit derived from the Newton formalism.

It is found that with (ψ, n, p) or (ψ, Φ_n, Φ_p) variables, the sparse direct method is insensitive to bias condition, and ultimately is 4–5 times cheaper than the decoupled (Gummel and block) methods. On this more difficult structure, it was impossible to achieve convergence with the quasi-Fermi potentials using either an iterative or direct solver. For any significant bias increment using these variables, the Newton iteration made no progress, neither diverging nor converging.

Because the ILU/knot method retains the equation to equation coupling at each node, it was found to be quite stable with regard to operating point unlike the other iterative Newton methods. However on the VAX, the ILU/knot method was consistently more expensive than the

[2] VAX is a trademark of Digital Equipment Corporation.
[3] UNIX is a trademark of AT&T Bell Laboratories.

sparse direct method. It also broke down unless at least a dozen previous GCR directions were retained, bringing its storage costs in line with sparse elimination.

VII. Summary

A variety of iterative methods have been tested for solving the Newton equations arising from device simulation. The results lead us to conclude that for on-state two-dimensional devices, the system is sufficiently refractory to favor sparse direct methods over any iterative technique at our disposal. The block iteration was found to be overrated, behaving as a decoupled method (i.e., Gummel), and the ILU/knot iteration was consistently slower than sparse Gaussian elimination. It is however a possible alternative for very large structures or three-dimensional problems, where sparse methods are not practical.

The outer Newton iteration was observed to be most stable using the carrier concentrations as the independent variables. Reasonable stability was seen with the Slotboom variables, while the quasi-Fermi potentials were quite unsatisfactory. Ironically, the inner iteration had the reverse dependence on the choice of variables; the linear system was usually much better conditioned using the quasi-Fermi potentials than with the alternatives.

It is therefore concluded that for two-dimensional device simulation, Gummel's iteration should be used for steady-state operating points through medium-level injection. For transient simulation and high-level injection, the sparse direct Newton method is preferred.

References

[1] M. R. Pinto, C. S. Rafferty, and R. W. Dutton, "PISCES-II: Poisson and continuity equation solver," Stanford Electronics Lab. Tech. Rep., Sept. 1984.

[2] R. E. Bank and A. H. Sherman, "An adaptive, multi-level method for elliptic boundary value problems," *Computing*, vol. 26, pp. 91–105, 1981.

[3] C. L. Lawson, "Software for C^1 surface interpolation," in *Mathematical Software III*, J. Rice, Ed., New York: Academic, 1977.

[4] C. H. Price, "Two-dimensional numerical simulation of semiconductor devices," Ph.D. dissertation, Dep. of Electrical Eng., Stanford Univ., CA, May 1982.

[5] M. J. Fritts, "Numerical approximations on distorted lagrangian grids," *Advances in Computer Methods for Partial Differential Equations*, vol. III, pp. 137–142, IMACS, 1979.

[6] R. S. Varga, *Matriz Iterative Analysis*. Englewood Cliffs, NJ: Prentice-Hall, 1962.

[7] D. Scharfetter and H. K. Gummel, "Large-signal analysis of a silicon Read diode oscillator," *IEEE Trans. Electron Devices*, vol. ED-16, pp. 64–77, 1969.

[8] J. A. Greenfield and R. W. Dutton, "Nonplanar VLSI device analysis using the solution of Poisson's equation," *IEEE Trans. Electron Devices*, vol. ED-27, pp. 1520–1532, 1980.

[9] R. H. Macneal, "An asymmetrical finite difference network," *Quart. Appl. Math.*, vol. 11, pp. 295–310, 1953.

[10] H. K. Gummel, "A self-consistent iterative scheme for one-dimensional steady state transistor calculations," *IEEE Trans. Electron Devices*, vol. ED-11, pp. 455–465, 1964.

[11] J. W. Slotboom, "Iterative scheme for 1- and 2-dimensional DC transistor simulation," *Electron. Lett.*, vol. 5, p. 677–678, 1969.

[12] B. V. Ghokale, "Numerical solutions for a one-dimensional silicon n-p-n transistor," *IEEE Trans. Electron Devices*, vol. ED-17, pp. 594–602, Aug. 1970.

[13] E. M. Buturla and P. E. Cottrell, "Simulation of semiconductor transport using coupled and decoupled solution techniques," *Solid State Electronics*, vol. 23, pp. 331–334, 1980.

[14] A. Yoshii, H. Kitazawa, M. Tomizawa, S. Horiguchi, and T. Sudo, "A three-dimensional analysis of semiconductor devices," *IEEE Trans. Electron Devices*, vol. ED-29, pp. 184–189, 1982.

[15] M. S. Mock, "The charge-neutral approximation and time-dependent simulation," in *NASECODE-I*, pp. 120–135, Boole Press, Dublin, June 1979.

[16] M. S. Mock, "Time discretization of a nonlinear initial value problem," *Journal of Computational Physics*, vol. 21, pp. 20–37, 1976.

[17] E. C. Sangiorgi, M. R. Pinto, S. E. Swirhun, and R. W. Dutton, "Two-dimensional numerical analysis of latchup in a VLSI CMOS technology," pp. 2117–2130, this issue.

[18] M. R. Pinto and R. W. Dutton, "Accurate trigger condition analysis for CMOS latch-up," *IEEE Electron Device Lett.*, vol. EDL-6, Feb. 1985.

[19] R. E. Bank and D. J. Rose, "Global approximate Newton methods," *Numer. Math.*, vol. 37, pp. 279–295, 1981.

[20] A. K. Cline, C. B. Moler, G. W. Stewart, and J. H. Wilkinson, "An estimate for the condition number of a matrix," *SIAM J. Numer. Anal.*, vol. 16, pp. 368–75, 1979.

[21] R. E. Bank, D. J. Rose, and W. Fichtner, "Numerical methods for semiconductor device simulation," *IEEE Trans. Electron Devices*, vol. ED-30, pp. 1031–1041, Sept. 1983.

[22] A. F. Franz, G. A. Franz, S. Selberherr, C. Ringhofer, and P. Markowich, "Finite boxes—A generalization of the finite-difference method suitable for semiconductor device simulation," *IEEE Trans. Electron Devices*, vol. ED-30, pp. 1070–1082, Sept. 1983.

[23] O. Axelsson and I. Gustafsson, "Iterative methods for the solution of the Navier equations of elasticity," *Computer Methods in Applied Mechanics and Engineering*, vol. 15, pp. 241–258, 1978.

[24] A. D. Tuff and A. Jennings, *Int. J. Num. Methods Engng.*, vol. 7, pp. 175–183, 1973.

[25] A. Jennings and G. M. Malik, "Partial elimination," *J. Inst. Math. Appl.*, vol. 20, pp. 307–316, 1977.

[26] H. C. Elman, "Preconditioned Conjugate Gradient Methods for Nonsymmetric Systems of Linear Equations," Yale University, Dep. of Computer Science, Res. Rep. 203, Apr. 1983.

[27] W. Fichtner and D. J. Rose, "On the numerical solution of nonlinear elliptic PDEs arising from semiconductor device modeling," in *Elliptic Problem Solvers*, New York: Academic, 1981, pp. 277–284.

[28] D. S. Kershaw, "The incomplete cholesky-conjugate gradient method for the iterative solution of systems of linear equations," *J. Comput. Physics*, vol. 26, pp. 43–65, 1978.

[29] J. A. Meijerink and H. A. van der Vorst, "An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix," *Math. Computation*, vol. 31, pp. 148–162, 1977.

[30] N. Munksgaard, "Solving sparse symmetric sets of linear equations by preconditioned conjugate gradients," *ACM Trans. Math. Software*, vol. 6, pp. 206–219, 1980.

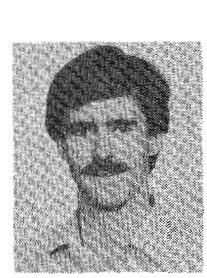
[31] C. den Heijer, "Preconditioned iterative methods for nonsymmetric linear systems," in *Proc. Int. Conf. on Simulation of Semiconductor Devices and Processes*, Pineridge Press, Swansea, UK, 1984.

[32] E. Crabbe, private communication.

[33] D. A. Calahan and P. G. Buning, "Vectorized General Sparsity Algorithms with Backing Store," SEL Rep. 96, Systems Engineering Lab., Ann Arbor, MI, 1977.

[34] S. C. Eisenstat, M. C. Gursky, M. H. Schultz, and A. H. Sherman, "Yale Sparse Matrix Package," Yale University, Dep. of Computer Science, Res. Rep. 114, 1977.

*



Conor S. Rafferty was born in Zurich, Switzerland, on March 22, 1960. He received the B.S. degree in physics and the B.A. degree in mathematics from Trinity College, Dublin, Ireland in 1981 and the M.S. degree in engineering in 1982. He is currently working towards the Ph.D. degree at Stanford University, Stanford, CA.

His research interests are in numerical methods for modeling physical processes.

*

Mark R. Pinto, for a photograph and biography please see page 2130 of this TRANSACTIONS.

*

Robert W. Dutton (S'67–M'70–SM'80–F'84), for a photograph and biography please see page 2130 of this TRANSACTIONS.