Recent Advances in Krylov-Subspace Solvers for Linear Systems and Applications in Device Simulation

W. M. Coughran, Jr. and R. W. Freund
Bell Labs, Lucent Technologies
700 Mountain Avenue
Murray Hill, NJ 07974–0636

***Abstract*— The computational cost of many simulations is dominated by the solution of large, sparse systems of linear equations. Krylov-subspace methods, especially when combined with suitable preconditioning, are powerful algorithms for the iterative solution of such linear systems. One of the features of Krylov-subspace methods is that the matrix of the linear system is only used in the form of matrix-vector products, and thus sparsity is naturally exploited. In recent years, there have been many advances in Krylov-subspace methods for the solution of large, sparse, nonsymmetric linear systems. In this paper, we survey some of these recent advances, especially in the area of Lanczos-based methods. We also discuss the use of state-of-the-art Krylov-subspace methods in device simulation.**

I. Introduction

The computational cost of many simulations is dominated by the solution of systems of linear equations,

$$\mathbf{A}\,\mathbf{z} = \mathbf{b}, \tag{1}$$

where $\mathbf{A}$ is a large, but sparse, nonsingular matrix. Typically, these large, sparse linear systems arise from discretization and possibly linearization of partial differential equations (PDE's) that model the process to be simulated.

As an example, consider the classical drift-diffusion equations [19], [22] for the modeling of semiconductor structures. The drift-diffusion equations are a coupled system of PDE's for the electrostatic potential $\psi(\mathbf{x},t)$, the electron concentration $n(\mathbf{x},t)$, and the hole concentration $p(\mathbf{x},t)$. The equations can be written as follows:

$$\begin{aligned} -\nabla\cdot(\epsilon\nabla\psi) + q\,(n-p-C(\mathbf{x})) &= 0,\\ -q\,\frac{\partial n}{\partial t} + \nabla\cdot\mathbf{J}_n + q\,R_n(n,p) &= 0,\\ q\,\frac{\partial p}{\partial t} + \nabla\cdot\mathbf{J}_p - q\,R_p(n,p) &= 0, \end{aligned} \tag{2}$$

where ϵ, q, C, R_n, and R_p are the electron charge, the dielectric permittivity, the net impurity (doping) concentration, and the net electron and hole recombination rates, respectively. Furthermore, in (2), $\mathbf{J}_n$ and $\mathbf{J}_p$ are the electron and hole current densities. They are defined as

$$\begin{aligned} \mathbf{J}_n &= -q\,\mu_n\,n\,\nabla\psi + q\,D_n\nabla n,\\ \mathbf{J}_p &= -q\,\mu_p\,p\,\nabla\psi - q\,D_p\nabla p, \end{aligned} \tag{3}$$

where μ_n and μ_p are the electron and hole mobilities, and D_n and D_p are the electron and hole diffusion coefficients. Energy balance (EB) or transport (ET) modeling of semiconductor structures leads to PDE's that are similar in mathematical character to (2)–(3); see, e.g., [19].

We now focus on the static problem and assume $\partial n/\partial t = \partial p/\partial t = 0$. In this case, the PDE's (2) (with suitable boundary conditions) and (3) represent a coupled system of boundary-value problems for the functions $\psi(\mathbf{x})$, $n(\mathbf{x})$, and $p(\mathbf{x})$, where $\mathbf{x}\in\Omega$ and $\Omega\subset\mathbb{R}^d$, $d=2$ or $d=3$, is the two- or three-dimensional device structure. To solve (2)–(3) numerically, one first chooses an appropriate, in general irregular grid for Ω, and then discretizes the system of PDE's using a finite-element or finite-volume scheme. The result is a large, sparse system of nonlinear equations,

$$\mathbf{G}(\mathbf{z}) = \mathbf{0}, \quad \text{where} \quad \mathbf{G}:\ \mathbb{R}^{3\nu}\mapsto\mathbb{R}^{3\nu}, \quad \mathbf{z} = \begin{bmatrix}\boldsymbol{\psi}\\ \mathbf{n}\\ \mathbf{p}\end{bmatrix}. \tag{4}$$

Here, ν is the number of grid points, and $\boldsymbol{\psi}$, $\mathbf{n}$, $\mathbf{p}$ are vectors of length ν whose components are approximations of the function values $\psi(\mathbf{x})$, $n(\mathbf{x})$, $p(\mathbf{x})$, respectively, at the grid points. The nonlinear system (4) is solved by a Newton-type method. Computing the Newton search direction requires the solution of a large, sparse linear system (1), where $\mathbf{A}=\mathbf{G}'$ is the Jacobian of $\mathbf{G}$, at each Newton iteration. For the drift-diffusion equations, these Jacobians are nonsymmetric.

For linear systems of small size, the standard approach is to use direct methods, such as Gaussian elimination. These algorithms obtain the solution of (1) based on a factorization of the coefficient matrix $\mathbf{A}$. Direct methods have been adapted with great success to large, sparse linear systems; see, e.g., [8], [18] and the references therein. In fact, sparse direct methods are now widely used for the solution of large, sparse linear systems, especially for two-dimensional simulations. However, for the large, sparse linear systems arising from PDE's for three-dimensional simulations, direct solution methods have excessive storage requirements, and it becomes prohibitive to use them. For three-dimensional modeling, iterative methods are

0-7803-3775-1/97/$10.00 © 1997 IEEE.

usually far more efficient, and often the only way to tackle the large, sparse linear systems arising in such simulations.

Krylov-subspace methods, especially when combined with suitable preconditioning, are powerful algorithms for the iterative solution of large, sparse linear systems (1). One of the features of Krylov-subspace methods is that they use the matrix $\mathbf{A}$ of (1) only in the form of matrix-vector products, and thus they naturally exploit the sparsity of the linear system.

In recent years, there have been many advances in Krylov-subspace methods for the solution of large, sparse, nonsymmetric linear systems; see, e.g., [10], [20] and the references given therein. In this paper, we survey some of these recent advances, especially in the area of Lanczos-based Krylov-subspace methods. We discuss the design of robust and efficient iterations that remedy the erratic convergence behavior of earlier algorithms, show how possible breakdowns in the underlying Lanczos process can be avoided, and describe a block method for the solution of systems with multiple right-hand sides. We also discuss the use of state-of-the-art Lanczos-based Krylov-subspace methods in device simulation.

II. Krylov-Subspace Methods

We consider linear systems (1) where $\mathbf{A} \in \mathbb{C}^{N \times N}$ is a nonsingular, in general non-Hermitian matrix and $\mathbf{b} \in \mathbb{C}^N$. An iterative scheme for the solution of (1) is called a *Krylov-subspace method* if, for any initial guess $\mathbf{z}_0 \in \mathbb{C}^N$, it produces a sequence of approximations to $\mathbf{A}^{-1}\mathbf{b}$ of the form

$$\begin{aligned} &\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_\mu, \ldots, \\ &\text{where} \quad \mathbf{z}_\mu \in \mathbf{z}_0 + \mathcal{K}_\mu(\mathbf{A}, \mathbf{r}_0) \quad \text{for all} \quad \mu. \end{aligned} \tag{5}$$

Here, $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\,\mathbf{z}_0$ is the residual vector associated with the initial guess $\mathbf{z}_0$, and

$$\mathcal{K}_\mu(\mathbf{A}, \mathbf{r}_0) := \operatorname{span}\left\{\mathbf{r}_0, \mathbf{A}\,\mathbf{r}_0, \ldots, \mathbf{A}^{\mu-1}\,\mathbf{r}_0\right\} \tag{6}$$

is the μ-th Krylov subspace of $\mathbb{C}^N$ (induced by $\mathbf{A}$ and $\mathbf{r}_0$).

The key feature of Krylov subspaces is that already for $\mu \ll N$, $\mathcal{K}_\mu(\mathbf{A}, \mathbf{r}_0)$ often yields very good approximations $\mathbf{z}_\mu \approx \mathbf{A}^{-1}\mathbf{b}$ even though the dimension μ of $\mathcal{K}_\mu(\mathbf{A}, \mathbf{r}_0)$ is much smaller than the order N of the matrix $\mathbf{A}$. This is especially so when $\mathbf{A}$ is a suitably preconditioned version of the coefficient matrix of the linear system to be solved.

While Krylov subspaces have good approximation properties, the power basis $\mathbf{A}^{j-1}\,\mathbf{r}_0$, $j = 1, 2, \ldots, \mu$, used in the definition (6) is numerically unstable. A second issue is the choice of the actual iterates $\mathbf{z}_\mu$. In fact, any viable Krylov-subspace method has two main ingredients:

(i) A procedure to generate suitable basis vectors for $\mathcal{K}_\mu(\mathbf{A}, \mathbf{r}_0)$;

(ii) A procedure to generate iterates $\mathbf{z}_\mu$.

In the next two subsections, we discuss the choice of (i) and (ii).

A. Basis Vectors

The goal is to compute a sequence of vectors $\{\mathbf{v}_\mu\}_{\mu \geq 1}$ such that

$$\operatorname{span}\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\mu\} = \mathcal{K}_\mu(\mathbf{A}, \mathbf{r}_0) \quad \text{for all} \quad \mu \geq 1. \tag{7}$$

There are two distinctive approaches to this task. The first one is the Arnoldi process [2]. It produces numerically "optimal" basis vectors in the sense that the $\mathbf{v}_\mu$'s are orthonormal:

$$\mathbf{v}_j^{\mathrm{H}}\,\mathbf{v}_\mu = \begin{cases} 1 & \text{if } j = \mu, \\ 0 & \text{if } j \neq \mu, \end{cases} \qquad \text{for all} \quad j, \mu.$$

However, for general non-Hermitian matrices $\mathbf{A}$, this optimality comes at a rather hefty price. Indeed, given the vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\mu$, the construction of the next vector $\mathbf{v}_{\mu+1}$ is only possible by means of a "long" recurrence that involves all previous vectors. More precisely, $\mathbf{v}_{\mu+1}$ is computed via a recurrence of the form

$$\mathbf{v}_{\mu+1} = \frac{1}{h_{\mu+1,\mu}} \left(\mathbf{A}\,\mathbf{v}_\mu - \sum_{j=1}^{\mu} \mathbf{v}_j\,h_{j,\mu} \right).$$

As a result, all basis vectors have to be stored, and the work per μ-th iteration grows linearly with μ. The long recurrences truncate to short recurrences only for Hermitian matrices $\mathbf{A}$ and for some very special classes of non-Hermitian matrices; see [10].

A second approach is the Lanczos process [16]. It uses "short", namely, in the *generic case*, three-term recurrences to construct the vectors $\{\mathbf{v}_\mu\}_{\mu \geq 1}$, but it gives up optimality of the basis. Moreover, in contrast to the Arnoldi process, which only involves matrix-vector products with $\mathbf{A}$, the Lanczos process involves multiplications with both $\mathbf{A}$ and its transpose $\mathbf{A}^{\mathrm{T}}$. In fact, the Lanczos process generates two sequences of vectors

$$\{\mathbf{v}_\mu\}_{\mu \geq 1} \quad \text{and} \quad \{\mathbf{w}_\mu\}_{\mu \geq 1}. \tag{8}$$

The first sequence, the *right Lanczos vectors*, again satisfy (7), while the *left Lanczos vectors* $\{\mathbf{w}_\mu\}_{\mu \geq 1}$ span a second sequence of Krylov subspaces:

$$\operatorname{span}\{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_\mu\} = \mathcal{K}_\mu(\mathbf{A}^{\mathrm{T}}, \mathbf{l}) \quad \text{for all} \quad \mu \geq 1.$$

Here, $\mathbf{l} \in \mathbb{C}^N$, $\mathbf{l} \neq \mathbf{0}$, is an arbitrary vector, usually chosen as a random vector. In the generic case, the sequences (8) are generated by means of three-term recurrences of the form

$$\begin{aligned} \mathbf{v}_{\mu+1} &= \frac{1}{\gamma_{\mu+1}} \left(\mathbf{A}\,\mathbf{v}_\mu - \mathbf{v}_\mu\,\alpha_\mu - \mathbf{v}_{\mu-1}\,\beta_\mu \right), \\ \mathbf{w}_{\mu+1} &= \frac{1}{\tilde{\gamma}_{\mu+1}} \left(\mathbf{A}^{\mathrm{T}}\,\mathbf{w}_\mu - \mathbf{w}_\mu\,\alpha_\mu - \mathbf{w}_{\mu-1}\,\tilde{\beta}_\mu \right), \\ \text{where} \quad \alpha_\mu &= \frac{\mathbf{w}_\mu^{\mathrm{T}}\,\mathbf{A}\,\mathbf{v}_\mu}{\mathbf{w}_\mu^{\mathrm{T}}\,\mathbf{v}_\mu}. \end{aligned} \tag{9}$$

Furthermore, the coefficients α_μ, β_μ, $\tilde{\beta}_\mu$, $\gamma_{\mu+1}$, and $\tilde{\gamma}_{\mu+1}$ in (9) are chosen such that, in the generic case, the right and left Lanczos vectors are *biorthogonal*:

$$\mathbf{w}_j^{\mathrm{T}}\, \mathbf{v}_{\mu+1} = \mathbf{w}_{\mu+1}^{\mathrm{T}}\, \mathbf{v}_j = 0 \quad \text{for all} \quad j = 1, 2, \ldots, \mu. \tag{10}$$

However, it turns out that enforcing the biorthogonality condition (10) may not be possible for all μ. Indeed, a pair of right and left Lanczos vectors $\mathbf{v}_{\mu+1}$ and $\mathbf{w}_{\mu+1}$ satisfying (10) exists if, and only if, $\delta_\mu := \mathbf{w}_\mu^{\mathrm{T}}\, \mathbf{v}_\mu \neq 0$. If $\delta_\mu = 0$, then the classical Lanczos process actually breaks down due to division by $\delta_\mu = 0$ when trying to compute the coefficient α_μ in (9). Moreover, $\delta_\mu \neq 0$, but $\delta_\mu \approx 0$, signals a *near-breakdown* in the classical Lanczos process due to division by a number close to zero. In recent years, remedies for the possible breakdowns and near-breakdowns in the classical Lanczos process have been developed. The basic idea is to relax the vector-wise biorthogonality (10) to a cluster-wise biorthognality whenever $\delta_\mu \approx 0$ and to modify the classical Lanczos process accordingly. The resulting computational procedure is called the *look-ahead Lanczos algorithm*; see [11] and the references given there. The look-ahead Lanczos algorithm is identical to the classical Lanczos process as long as only look-ahead steps of length 1 occur. A *true* true look-ahead step of length 1 occurs if, and only if, $\delta_\mu \approx 0$. For those Lanczos iterations corresponding to a true look-ahead step, the recursions (9) are replaced by suitable block three-term recurrences. For a detailed description of one particular implementation of the look-ahead Lanczos algorithm, we refer the reader to [11]. FORTRAN 77 codes for this algorithm are available as part of the software package QMRPACK [15].

B. Choice of Iterates

Once a procedure for generating basis vectors $\{\mathbf{v}_\mu\}_{\mu \geq 1}$ has been specified, suitable Krylov-subspace iterates $\mathbf{z}_\mu$ need to be selected. Let

$$\mathbf{V}_\mu = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \ldots & \mathbf{v}_\mu \end{bmatrix} \tag{11}$$

be the matrix whose columns are just the first μ basis vectors. Then, by (5), (7), and (11), any possible μ-th iterate $\mathbf{z}_\mu$ can be parametrized as follows:

$$\mathbf{z}_\mu = \mathbf{z}_0 + \mathbf{V}_\mu\, \mathbf{y}_\mu, \quad \text{where} \quad \mathbf{y}_\mu \in \mathbb{C}^\mu. \tag{12}$$

Thus it only remains to select the free parameter vector $\mathbf{y}_\mu$ in (12).

Next, we note that the recurrences used to generate the first $\mu + 1$ basis vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{\mu+1}$ can always be summarized compactly as follows:

$$\mathbf{A}\, \mathbf{V}_\mu = \mathbf{V}_{\mu+1}\, \mathbf{T}_\mu, \quad \mu \geq 1. \tag{13}$$

Here, $\mathbf{T}_\mu$ is an $(\mu + 1) \times \mu$ matrix whose entries are the recurrence coefficients. In the case of the Arnoldi process, $\mathbf{T}_\mu$ is a full upper Hessenberg matrix, which just reflects the use of long recurrences. In the case of the classical Lanczos process, $\mathbf{T}_\mu$ is a tridiagonal matrix, reflecting the three-term recurrences. For the look-ahead Lanczos algorithm, $\mathbf{T}_\mu$ is tridiagonal plus a few "bulges".

Using (13) and the fact that $\mathbf{r}_0 = \rho\, \mathbf{v}_1$ for some scalar ρ, we obtain the following representation for the residual vector $\mathbf{r}_\mu$ associated with any potential μ-th iterate (12):

$$\begin{aligned} \mathbf{r}_\mu &= \mathbf{b} - \mathbf{A}\, \mathbf{z}_\mu \\ &= \mathbf{r}_0 - \mathbf{V}_{\mu+1}\, \mathbf{T}_\mu\, \mathbf{y}_\mu \\ &= \mathbf{V}_{\mu+1} \left(\rho\, \mathbf{e}_1^{(\mu+1)} - \mathbf{T}_\mu\, \mathbf{y}_\mu \right) \end{aligned} \tag{14}$$

Here, $\mathbf{e}_1^{(\mu+1)}$ is the first unit vector (in $\mathbb{C}^{\mu+1}$). Obviously, the goal for the choice of the free parameter vector $\mathbf{y}_\mu$ in (12) is to drive the residual vector $\mathbf{r}_\mu$ to $\mathbf{0}$ as fast as possible. The representation (14) of $\mathbf{r}_\mu$ suggests two strategies for the choice of $\mathbf{y}_\mu$. The first is a Galerkin-type approach. Here one deletes the last row of the $(\mu + 1) \times \mu$ matrix $\mathbf{T}_\mu$ to obtain a square $\mu \times \mu$ matrix $\mathbf{T}_\mu^{(s)}$, and then determines $\mathbf{y}_\mu$ as the solution of

$$\mathbf{T}_\mu^{(s)}\, \mathbf{y}_\mu = \rho\, \mathbf{e}_1^{(\mu)}. \tag{15}$$

Note that, in view of (14), the choice (15) implies

$$\mathbf{r}_\mu = -\left(\mathbf{e}_\mu^{\mathrm{T}}\, \mathbf{T}_\mu\, \mathbf{y}_\mu \right)\, \mathbf{v}_{\mu+1},$$

i.e., the μ-th residual vector is a scalar multiple of $\mathbf{v}_{\mu+1}$. The resulting Krylov-subspace methods based on the choice (15) are the biconjugate gradient (BCG) algorithm [17] if the Lanczos basis is used and the FOM algorithm [20] if the Arnoldi basis is used.

However, there is a problem with the choice (15) since, in general, the matrix $\mathbf{T}_\mu^{(s)}$ cannot be guaranteed to be nonsingular. In fact, if $\mathbf{T}_\mu^{(s)}$ happens to be singular, then both BCG and FOM break down since their respective iterates are not defined. Moreover, if $\mathbf{T}_\mu^{(s)}$ is close to singular, then the μ-th residual $\mathbf{r}_\mu$ may be large. Matrices $\mathbf{T}_\mu^{(s)}$ that are close to singular do indeed occur in practice, and are the reason for the typical erratic convergence behavior of BCG.

The second strategy is to choose $\mathbf{y}_\mu$ such that the Euclidean norm of $\mathbf{r}_\mu$ is minimized. In the case of the Arnoldi basis this is easy to do. Recall that the Arnoldi basis vectors are orthonormal, and that the Euclidean norm is invariant under multiplications with orthonormal matrices. Therefore, from (14), we obtain

$$\begin{aligned} \|\mathbf{r}_\mu\|_2 &= \left\| \mathbf{V}_{\mu+1} \left(\rho\, \mathbf{e}_1^{(\mu+1)} - \mathbf{T}_\mu\, \mathbf{y}_\mu \right) \right\|_2 \\ &= \left\| \rho\, \mathbf{e}_1^{(\mu+1)} - \mathbf{T}_\mu\, \mathbf{y}_\mu \right\|_2 \end{aligned} \tag{16}$$

By (16), we simply need to choose $\mathbf{y}_\mu$ as the solution of the least-squares problem

$$\left\| \rho \mathbf{e}_1^{(\mu+1)} - \mathbf{T}_\mu\, \mathbf{y}_\mu \right\|_2 = \min_{\mathbf{y} \in \mathbb{C}^\mu} \left\| \rho \mathbf{e}_1^{(\mu+1)} - \mathbf{T}_\mu\, \mathbf{y} \right\|_2 \tag{17}$$

in order to guarantee that $\|\mathbf{r}_\mu\|_2$ is minimal. The resulting algorithm is GMRES [21]. Note that the matrix $\mathbf{T}_\mu$ in (17) always has full column rank μ and thus (17) always has a unique solution $\mathbf{z}_\mu$.

C. Preconditioning

Krylov-subspace methods are hardly ever applied directly to the original linear system (1). Instead, they are combined with *preconditioning.* A *preconditioner* for (1) is a nonsingular matrix

$$\mathbf{M} = \mathbf{M}_1 \cdot \mathbf{M}_2 \tag{18}$$

of the same size as $\mathbf{A}$ such that $\mathbf{M}$ in some sense "approximates" $\mathbf{A}$, while solving systems with $\mathbf{M}_1$ and $\mathbf{M}_2$ is much "cheaper" than solving systems with $\mathbf{A}$. In (18), it is allowed that one of the factors is trivial, i.e., $\mathbf{M}_1 = \mathbf{I}$ or $\mathbf{M}_2$. The Krylov-subspace method is then applied (at least implicitly) to the preconditioned system

$$\mathbf{A}'\,\mathbf{z}' = \mathbf{b}', \tag{19}$$

where $\mathbf{A}' = \mathbf{M}_1^{-1}\,\mathbf{A}\,\mathbf{M}_2^{-1}$, $\mathbf{z}' = \mathbf{M}_2\,\mathbf{z}$, and $\mathbf{b}' = \mathbf{M}_1^{-1}\,\mathbf{b}$.

Standard preconditioning techniques include incomplete factorization techniques, where the matrices $\mathbf{M}_1$ and $\mathbf{M}_2$ in (18) are lower, respectively upper, triangular and result from an incomplete factorization of $\mathbf{A}$, or SSOR, where $\mathbf{M}_1$ and $\mathbf{M}_2$ result from an additive triangular decomposition of $\mathbf{A}$. For a discussion of these standard preconditioning techniques, we refer the reader to [3], [4], [20].

Recall that the drift-diffusion equations for device simulation represent. a system of coupled PDE's. For the linear systems arising from systems of PDE's, preconditioners can also be motivated by reorderings of the equations. Write

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \cdots & \mathbf{A}_{1m} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \cdots & \mathbf{A}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{m1} & \mathbf{A}_{m2} & \cdots & \mathbf{A}_{mm}, \end{bmatrix}$$

where $\mathbf{A}_{ij} \in \mathbb{R}^{\nu \times \nu}$ for all i, j, and m and ν denote the number of PDE's and grid points, respectively. Let

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_{11} & \mathbf{D}_{12} & \cdots & \mathbf{D}_{1m} \\ \mathbf{D}_{21} & \mathbf{D}_{22} & \cdots & \mathbf{D}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{D}_{m1} & \mathbf{D}_{m2} & \cdots & \mathbf{D}_{mm} \end{bmatrix}$$

where $\mathbf{D}_{ij} = \operatorname{diag}(\mathbf{A}_{ij})$. There is a permutation matrix, $\mathbf{P}$, such that

$$\tilde{\mathbf{A}} = \mathbf{P}\mathbf{A}\mathbf{P}^{\mathrm{T}} = \begin{bmatrix} \tilde{\mathbf{A}}_{11} & \tilde{\mathbf{A}}_{12} & \cdots & \tilde{\mathbf{A}}_{1\nu} \\ \tilde{\mathbf{A}}_{21} & \tilde{\mathbf{A}}_{22} & \cdots & \tilde{\mathbf{A}}_{2\nu} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{\mathbf{A}}_{\nu 1} & \tilde{\mathbf{A}}_{\nu 2} & \cdots & \tilde{\mathbf{A}}_{\nu\nu} \end{bmatrix}$$

where

$$\tilde{\mathbf{A}}_{ij} = \begin{bmatrix} (\mathbf{A}_{11})_{ij} & (\mathbf{A}_{12})_{ij} & \cdots & (\mathbf{A}_{1m})_{ij} \\ (\mathbf{A}_{21})_{ij} & (\mathbf{A}_{22})_{ij} & \cdots & (\mathbf{A}_{2m})_{ij} \\ \vdots & \vdots & \ddots & \vdots \\ (\mathbf{A}_{m1})_{ij} & (\mathbf{A}_{m2})_{ij} & \cdots & (\mathbf{A}_{mm})_{ij} \end{bmatrix} \in \mathbb{R}^{m \times m}.$$

Note that $\mathbf{A}$ is the matrix blocked by PDE, while $\tilde{\mathbf{A}}$ is *alternately blocked* by grid point; that is, $\mathbf{A}_{ii}$ represents the diagonal operator of the i-th PDE with the discrete variables ordered in terms of the spatial grid, while $\tilde{\mathbf{A}}_{ij}$ represents the linearized coupling between the PDE's at the ij-th grid point. The matrix $(\mathbf{PDP}^{\mathrm{T}})^{-1}$ has $m \times m$ matrices on its diagonal. The ABF preconditioner [5] is then given by $\mathbf{M}_2 = \mathbf{D}$ and $\mathbf{M}_1 = \mathbf{I}$, and it is implemented in terms of the $\tilde{\mathbf{D}}$ analog. Incomplete factorizations based on either the "natural" or ABF ordering are alternatives [6].

In the following, we always assume that $\mathbf{A}\,\mathbf{z} = \mathbf{b}$ is already the preconditioned system.

III. Lanczos-Based Iterative Solvers

We now discuss Krylov-subspace methods based on the Lanczos process. We already described one such method, BCG, and mentioned its erratic convergence behavior. The question arises if BCG can be stabilized in a way similar to how GMRES avoids the potential breakdowns of FOM. This can indeed be done, even though the resulting method no longer minimizes the residual norm. Again, one uses the formula (14) for the residual $\mathbf{r}_\mu$ of any potential μ-th iterate $\mathbf{z}_\mu$. In (14), $\mathbf{V}_{\mu+1}$ is now the matrix of right Lanczos vectors, which are biorthogonal to the left Lanczos vectors but not orthonormal among themselves, and $\mathbf{T}_\mu$ is the tridiagonal matrix of Lanczos recurrence coefficients. Since $\mathbf{V}_{\mu+1}$ is no longer orthonormal, the Euclidean norm of $\mathbf{r}_\mu$ cannot be minimized cheaply, but it can be quasi-minimized. This means that we choose $\mathbf{y}_\mu$ such that the Euclidean norm of the bracketed term on the right-hand side of (14) is minimized. Thus, $\mathbf{y}_\mu$ is chosen as the solution of the least-squares problem (17), where $\mathbf{T}_\mu$ now is the Lanczos tridiagonal matrix. The resulting Krylov-subspace method is the quasi-minimal residual (QMR) algorithm [13], [14].

Due to the quasi-minimization of the residual norm, QMR effectively remedies the erratic convergence behavior of BCG. Moreover, the potential breakdowns and near-breakdowns in the underlying Lanczos process can be avoided by using look-ahead; we note that FORTRAN 77 codes for QMR with look-ahead are included in QMRPACK [15]. The following numerical examples illustrate these features of QMR.

Example 1. We consider a *pn* diode with a maximal p doping of 10^{19} with a Gaussian junction to an n doping of 10^{15}. The diode was in low injection (0.2V). We ran a simulation of the diode, using either QMR or BCG as the linear systems solver. In this case, BCG did not converge to sufficient accuracy for some of the linear systems arising in the outer Newton iteration, while QMR always converged. In Figure 1, we show the relative residual norms for QMR (solid line) and BCG (dotted line) for one of the linear systems where BCG failed.

Example 2. Here, we consider the *pn* diode again, but now in high injection (2V). In Figure 2, we show the re-

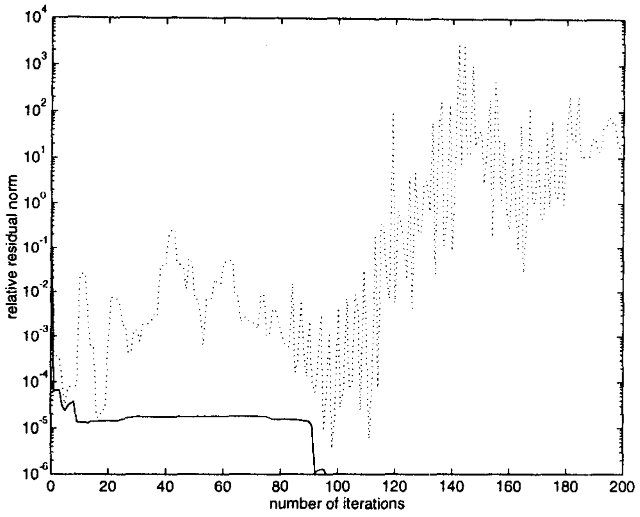

Fig. 1. QMR and BCG for simulation of *pn* diode; Example 1

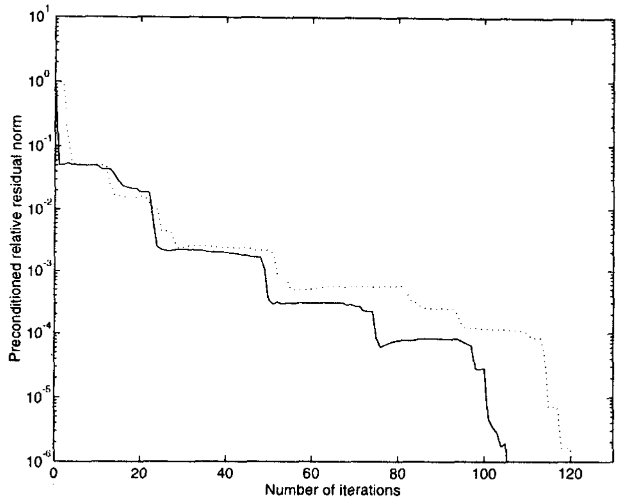

Fig. 2. QMR with and without look-ahead for simulation of *pn* diode; Example 2

sults of a QMR run with look-ahead (solid line) and without (dotted line). In this case, convergence is delayed when look-ahead is turned off, even though the underlying Lanczos algorithm encountered only a mild near-breakdown.

A. Transpose-Free Methods

Both the BCG and the QMR algorithm involve products with $\mathbf{A}$ and $\mathbf{A}^{\mathrm{T}}$. Sonneveld [23] was the first to observe that the products with $\mathbf{A}^{\mathrm{T}}$ in BCG can be eliminated, although this is done at the expense of computing different iterates. More precisely, his CGS algorithm [23] generates iterates $\mathbf{z}_{2\mu}^{\mathrm{CGS}}$ that are related to the BCG iterates $\mathbf{z}_{\mu}^{\mathrm{BCG}}$ as follows:

$$\begin{aligned} \mathbf{b} - \mathbf{A}\,\mathbf{z}_{2\mu}^{\mathrm{CGS}} &= (\phi_\mu(\mathbf{A}))^2\,(\mathbf{b} - \mathbf{A}\,\mathbf{z}_0)\,, \\ \mathbf{b} - \mathbf{A}\,\mathbf{z}_{\mu}^{\mathrm{BCG}} &= \phi_\mu(\mathbf{A})\;(\mathbf{b} - \mathbf{A}\,\mathbf{z}_0)\,. \end{aligned} \tag{20}$$

Here, ϕ_μ is the μ-th BCG polynomial. Unfortunately, the squaring of ϕ_μ in (20) implies that both the good and the bad properties of BCG are squared, and thus the convergence behavior of CGS can be even more erratic than that of BCG.

It is possible to derive QMR-type algorithms that—just like CGS—avoid the use of products with the transpose $\mathbf{A}^{\mathrm{T}}$, but whose convergence is less erratic. The most efficient of these algorithms is TFQMR [9].

Another approach to smoothing the convergence behavior of CGS is used in the Bi-CGSTAB algorithm [24]. Instead of squaring ϕ_μ, it is based on a product of ϕ_μ with a polynomial resulting from steepest descent steps.

IV. The Block-QMR Method for Multiple Right-Hand Sides

Some numerical simulations involve the solution of multiple linear systems

$$\mathbf{A}\,\mathbf{z}^{(j)} = \mathbf{b}^{(j)}, \quad j = 1, 2, \ldots, m_0, \tag{21}$$

with the same matrix $\mathbf{A}$, but different right-hand sides. For example, this situation arises when bordered systems are solved by means of repeated solves of systems with the unbordered subsystem. In (21), $\mathbf{A}$ is a fixed, nonsingular, in general complex non-Hermitian, $N \times N$ matrix, and $\mathbf{b}^{(j)} \in \mathbb{C}^N$. Assuming that all right-hand sides $\mathbf{b}^{(j)}$, $j = 1, 2, \ldots, m_0$, are available simultaneously—as it is the case for bordered systems—solving the m_0 systems (21) is equivalent to solving the block system of linear equations

$$\mathbf{A}\,\mathbf{Z} = \mathbf{B}, \quad \text{where} \quad \mathbf{B} = \begin{bmatrix} \mathbf{b}^{(1)} & \mathbf{b}^{(2)} & \cdots & \mathbf{b}^{(m_0)} \end{bmatrix}. \tag{22}$$

The solution vectors of (21) are then just the columns of the solution $\mathbf{Z} = \begin{bmatrix} \mathbf{z}^{(1)} & \mathbf{z}^{(2)} & \cdots & \mathbf{z}^{(m_0)} \end{bmatrix}$ of (22).

Instead of applying an iterative method, such as QMR, to each of the m_0 systems (21) individually, it is potentially more efficient to apply a suitable block version of the iterative method to the block system (22). For Krylov-subspace methods, the advantage of block iterations over individual runs can be formulated precisely; see, e.g., [12]. Essentially, the argument is that a block Krylov-subspace method selects its iterates from a sequence of subspaces that are higher dimensional than the subspaces from which the iterates of the individual runs are chosen, even though generating the "block" subspaces and all m_0 sequences of "individual" subspaces requires roughly the same computational work.

The block-QMR method [12] is an extension of QMR for systems with single right-hand sides to block systems (22). Let $\mathbf{Z}_0 \in \mathbb{C}^{N \times m_0}$ be an arbitrary initial guess for the exact solution $\mathbf{A}^{-1}\mathbf{B}$ of (22), and set $\mathbf{R}_0 := \mathbf{B} - \mathbf{A}\,\mathbf{Z}_0$.

The block-QMR method constructs approximate solutions of (22) that are of the form

$$\mathbf{Z}_\mu = \begin{bmatrix} \mathbf{z}_\mu^{(1)} & \mathbf{z}_\mu^{(2)} & \cdots & \mathbf{z}_\mu^{(m_0)} \end{bmatrix} \in \mathbb{C}^{N \times m_0}, \quad \mu \geq 1, \quad (23)$$

where $\mathbf{z}_\mu^{(j)} \in \mathbf{z}_0^{(j)} + \mathcal{K}_\mu(\mathbf{A}, \mathbf{R}_0)$ for each $j = 1, 2, \ldots, m_0$. Here, for each $\mu = 1, 2, \ldots,$

$$\mathcal{K}_\mu(\mathbf{A}, \mathbf{R}_0) := \operatorname{colspan}_\mu \begin{bmatrix} \mathbf{R}_0 & \mathbf{A}\mathbf{R}_0 & \cdots & \mathbf{A}^{N-1}\mathbf{R}_0 \end{bmatrix} \quad (24)$$

is the *μ-th block Krylov subspace* (generated by $\mathbf{A}$ and $\mathbf{R}_0$). In (24), the following notation is used. For any matrix $\mathbf{M}$ and any integer $0 < \mu \leq \operatorname{rank} \mathbf{M}$, we denote by $\operatorname{colspan}_\mu [\mathbf{M}]$ the μ-dimensional subspace spanned by the first μ linearly independent columns of $\mathbf{M}$ that one encounters when scanning the columns from left to right for linear independence.

The block-QMR method uses a recently developed Lanczos-type process [1] that produces two sequences of *biorthogonal* basis vectors for $\mathcal{K}_\mu(\mathbf{A}, \mathbf{R}_0)$ and the μ-th block Krylov subspace (generated by $\mathbf{A}^{\mathrm{T}}$ and $\mathbf{L}$) $\mathcal{K}_\mu(\mathbf{A}^{\mathrm{T}}, \mathbf{L})$; here, $\mathbf{L} \in \mathbb{C}^{N \times m_0}$ is an arbitrarily chosen matrix, different from the zero matrix. More precisely, after μ steps, the Lanczos-type process has produced *right* and *left Lanczos vectors*

$$\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\mu \quad \text{and} \quad \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_\mu, \quad (25)$$

respectively, such that

$$\begin{aligned} \operatorname{span}\{\, \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\mu \,\} &= \mathcal{K}_\mu(\mathbf{A}, \mathbf{R}_0), \\ \operatorname{span}\{\, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_\mu \,\} &= \mathcal{K}_\mu(\mathbf{A}^T, \mathbf{L}), \\ \mathbf{w}_i^{\mathrm{T}} \mathbf{v}_k &= 0 \quad \text{for all} \quad i \neq k = 1, 2, \ldots, \mu. \end{aligned}$$

An important feature of the Lanczos-type process is that the Lanczos vectors (25) are generated by means of recurrences of limited length, namely $(2m_0+1)$-term recurrences. For the right Lanczos vectors, these recurrences can be summarized in compact form as follows:

$$\mathbf{A}\,\mathbf{V}_\mu = \mathbf{V}_n\,\mathbf{T}_\mu + \widehat{\mathbf{V}}_\mu^{\mathrm{dl}}, \quad \mu \geq 1. \quad (26)$$

Here, $\mathbf{V}_\mu := \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_\mu \end{bmatrix}$, $\mathbf{T}_\mu$ is an $n \times \mu$ banded matrix with upper and lower bandwidth m_0 containing the recurrence coefficients, and $\widehat{\mathbf{V}}_\mu^{\mathrm{dl}}$ is a matrix with mostly zero or very small entries that accounts for "deflation" of linearly or almost linearly dependent columns in the block Krylov sequence $\mathbf{R}_0, \mathbf{A}\mathbf{R}_0, \ldots, \mathbf{A}^{N-1}\mathbf{R}_0$ in (24). The recurrences (26) are complemented by "initial" recurrences

$$\mathbf{V}_{m_1}\,\boldsymbol{\rho} + \widehat{\mathbf{V}}_0^{\mathrm{dl}} = \mathbf{R}_0 \quad (27)$$

that are used to biorthogonalize the columns of the "right" initial block $\mathbf{R}_0$ against the columns of the "left" initial block $\mathbf{L}$. In (27), $m_1 \leq \operatorname{rank} \mathbf{R}_0 \leq m_0$, and $\boldsymbol{\rho}$ is an upper triangular $m_1 \times m_0$ matrix. The left Lanczos vectors in (25) are generated by means of recurrences similar to (26) and (27).

In the block-QMR method, the right Lanczos vectors are used to parametrize any possible iterate (23) as follows:

$$\mathbf{Z}_\mu = \mathbf{Z}_0 + \mathbf{V}_\mu\,\mathbf{Y}_\mu, \quad \text{where} \quad \mathbf{Y}_\mu \in \mathbb{C}^{\mu \times m_0}. \quad (28)$$

By (26) and (27), the block residual associated with (28) is given by

$$\begin{aligned} \mathbf{R}_\mu &= \mathbf{R}_0 - \mathbf{V}_n\,\mathbf{T}_\mu\,\mathbf{Z} - \widehat{\mathbf{V}}_\mu^{\mathrm{dl}}\,\mathbf{Z} \\ &= \mathbf{V}_n \left(\begin{bmatrix} \rho \\ \mathbf{0} \end{bmatrix} - \mathbf{T}_\mu\,\mathbf{Z} \right) - \widehat{\mathbf{V}}_\mu^{\mathrm{dl}}\,\mathbf{Z}. \end{aligned} \quad (29)$$

Following the QMR philosophy and assuming that the columns $\mathbf{v}_j$ of the matrix $\mathbf{V}_n$ are all normalized to have Euclidean length 1, the free parameter matrix $\mathbf{Y}_\mu$ in (28) is determined so that the Euclidean norm of the bracketed term in (29) is minimized. Thus, in (28), one chooses $\mathbf{Y}_\mu$ as the solution of the matrix least-squares problem

$$\left\| \begin{bmatrix} \rho \\ \mathbf{0} \end{bmatrix} - \mathbf{T}_\mu\,\mathbf{Y}_\mu \right\|_2 = \min_{\mathbf{Y} \in \mathbb{C}^{\mu \times m_0}} \left\| \begin{bmatrix} \rho \\ \mathbf{0} \end{bmatrix} - \mathbf{T}_\mu\,\mathbf{Y} \right\|_2. \quad (30)$$

In the block-QMR method, the matrix least-squares problem (30) is solved by means of standard techniques based on a QR decomposition of $\mathbf{T}_\mu$. In particular, this allows to obtain the solution $\mathbf{Y}_\mu$ by updating the solution $\mathbf{Y}_{\mu-1}$ from the previous step. Moreover, the associate update $\mathbf{Z}_{\mu-1} \longrightarrow \mathbf{Z}_\mu$ for the actual iterates (28) turns out to be a simple rank-one update. For a description of this update procedure and for further implementation details, we refer the reader to [12].

V. (I, V) Continuation

We now return to the coupled systems of PDE's that arise in device simulation. In this section, we discuss continuation techniques for computing the (I, V) behavior of semiconductor devices [7].

Recall that, once discretized, the DD or EB/ET equations become a nonlinear system

$$\mathbf{G}(\mathbf{z}) = \mathbf{0}. \quad (31)$$

Consider parametrizing the (I, V) curves by arc-length s so that $\mathbf{z} \rightarrow \mathbf{z}(s)$. Then differentiating (31) with respect to s we get

$$\begin{aligned} \mathbf{G}'\dot{\mathbf{z}}(s) + \frac{\partial \mathbf{G}}{\partial V}\,\dot{V}(s) &= \mathbf{0}, \\ |\dot{I}(s)|^2 + |\dot{V}(s)|^2 &= 1. \end{aligned}$$

Next, we replace the arc-length condition by one of the following pseudo-arclength conditions:

$$\begin{aligned} N_1 &= \theta \dot{I}_j (I - I_j) + (2 - \theta)\dot{V}_j (V - V_j) - \Delta\sigma_j, \\ N_2 &= (I - I_j)^2 + (V - V_j)^2 - (\Delta\sigma_j)^2. \end{aligned}$$

For either choice of N, the linearized matrices are of the form

$$\mathbf{A} = \begin{bmatrix} \mathbf{G}' & \partial_V \mathbf{G} \\ (\partial_{\mathbf{z}} N_\star)^{\mathrm{T}} & \partial_V N_\star \end{bmatrix},$$

where $\mathbf{G}' \in \mathbb{R}^{n \times n}$ is the usual DD or EB/ET Jacobian, and $\partial_V \mathbf{G}$, $\partial_{\mathbf{z}} N_\star \in \mathbb{R}^n$. Note that $\mathbf{A}$ is just the usual Jacobian, $\mathbf{G}'$, bordered by a single row and column. For $\Delta\sigma_j \ll 1$, the matrix $\mathbf{A}$ is nonsingular, even at simple limit points where $\mathbf{G}'$ itself is singular.

There are applications (such as computing CMOS latchup trigger and holding points) where limit points need to be found. In this case, bisection or more sophisticated algorithms are applied to find where the curve turns back on itself, necessitating numerous solves where $\mathbf{G}'$ is ill-conditioned or effectively singular. (I, V) continuation has proven to be a highly effective algorithm for such applications.

At bifurcations or for manifold exploration, $\mathbf{G}'$ must be normalized by bordering it with two rows and columns. There are applications of finding fold lines in a manifold for which multivariate continuation is a natural and efficacious approach.

VI. Incorporating Circuit Elements into Device Simulation Models

Bordered linear systems also arise when device simulation models are combined with circuit elements.

A. Circuit Elements

Lumped circuit elements can be treated via Kirchhoff's current and voltage laws

$$\mathcal{A}_a \, \mathbf{i} = \mathbf{0} \in \mathbb{R}^m,$$
$$\mathbf{v} = \mathcal{A}_a^{\mathrm{T}} \, \mathbf{u} \in \mathbb{R}^l,$$

(where $\mathcal{A}_a \in \mathbb{R}^{m \times l}$, $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{i}$ are the augmented incidence matrix, node voltages, branch voltages, and branch currents, respectively) and global constitutive relations

$$\mathbf{K}(\mathbf{i}, \mathbf{v}) \equiv \mathbf{i} - \left(\frac{d}{dt} \mathbf{q}(\mathbf{v}) + \mathbf{f}(\mathbf{v}) \right) = \mathbf{0} \in \mathbb{R}^l.$$

We refer the reader to [25] for a detailed discussion of these equations.

For voltage-controlled, lumped elements these reduce to the (linearized) nodal equations involving $\mathcal{A} \, \mathbf{K}_{\mathbf{v}} \, \mathcal{A}^{\mathrm{T}}$ where $\mathcal{A}$ is the incidence matrix.

B. Incorporating Circuit Elements

Consider a static device model with coupled lumped elements. The matrices from the linearizations are bordered matrices of the form

$$\mathbf{A} = \begin{bmatrix} \mathbf{G}' & \mathbf{c} \\ \mathbf{r}^{\mathrm{T}} & \mathcal{A} \, \mathbf{K}_{\mathbf{v}} \, \mathcal{A}^{\mathrm{T}} \end{bmatrix},$$

where $\mathbf{G}' \in \mathbb{R}^{N \times N}$ is the usual DD or EB/ET Jacobian, $\mathbf{c}$, $\mathbf{r} \in \mathbb{R}^{N \times m}$ represent the coupling between the device and circuit equations at contacts, and $\mathcal{A} \, \mathbf{K}_{\mathbf{v}} \, \mathcal{A}^{\mathrm{T}}$ are the nodal network equations. We assume $m \ll N$, and so $\mathbf{G}'$ is the dominating part of $\mathbf{A}$.

We remark that continuation can be applied to such problems as well, introducing $m+1$ (or even $m+2$) row and column bordering of $\mathbf{G}'$.

C. Block Elimination

Consider bordered systems

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{a}_{12} \\ \mathbf{a}_{21}^{\mathrm{T}} & \mathbf{a}_{22} \end{bmatrix} \mathbf{z} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}, \tag{32}$$

where $\mathbf{A}_{11} \in \mathbb{C}^{N \times N}$, $\mathbf{a}_{22} \in \mathbb{C}^{m \times m}$, and $m \ll N$. For the case that $\mathbf{A}_{11}$ is nonsingular, one popular approach to solving (32) is to first perform one step of block elimination with $\mathbf{A}_{11}$. This results in the equivalent system

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{a}_{12} \\ 0 & \mathbf{a}_{22} - \mathbf{a}_{21}^{\mathrm{T}} \, \mathbf{A}_{11}^{-1} \, \mathbf{a}_{12} \end{bmatrix} \mathbf{z} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 - \mathbf{a}_{21}^{\mathrm{T}} \, \mathbf{A}_{11}^{-1} \, \mathbf{b}_1 \end{bmatrix}.$$

Clearly, this approach requires the solution of $m+1$ linear systems with the same matrix $\mathbf{A}_{11}$ to obtain the m columns of $\mathbf{A}_{11}^{-1} \, \mathbf{a}_{12}$ and the vector $\mathbf{A}_{11}^{-1} \, \mathbf{b}_1$.

When direct sparse methods are used, one can solve these $m+1$ systems as follows:

(1) Factor $\mathbf{P} \, \mathbf{A}_{11} = \mathbf{L} \, \mathbf{U}$;

(2) Do $m+1$ backsolves to compute $\mathbf{A}_{11}^{-1} \, \mathbf{a}_{12}$ and $\mathbf{A}_{11}^{-1} \, \mathbf{b}_1$.

For limit-point computations with continuation, $\mathbf{A}_{11}$ is nearly singular, and then more complicated techniques like deflated elimination or working-precision iterative refinement may be required. In this case, more than $m+1$ backsolves are needed.

When iterative methods are used, the $m+1$ solves with $\mathbf{A}_{11}$ can be summarized as a block system

$$\mathbf{A}_{11} \, \mathbf{Z} = \begin{bmatrix} \mathbf{a}_{12} & \mathbf{b}_1 \end{bmatrix}$$

and the block-QMR method can be employed.

D. Bordering Preconditioners

Another approach to solving (32), which does not require $\mathbf{A}_{11}$ to be nonsingular, is to simply apply a Krylov-subspace methods to the bordered system (32). The bordered structure can still be used in the construction of suitable preconditioners for (32).

Given any preconditioner $\mathbf{M}_{11}$ for $\mathbf{A}_{11}$, we want to construct a preconditioner $\mathbf{M}$ for the bordered matrix $\mathbf{A}$ in (32), Since $\mathbf{M}_{11}$ is a preconditioner, $\mathbf{M}_{11} = \mathbf{L} \cdot \mathbf{R}$, where systems with $\mathbf{L}$ and $\mathbf{R}$ can be solved easily. We then border $\mathbf{L}$ and $\mathbf{R}$ by setting

$$\mathbf{M}_1 = \begin{bmatrix} \mathbf{L} & \mathbf{0} \\ \mathbf{l}^{\mathrm{T}} & \mathbf{c} \end{bmatrix} \quad \text{and} \quad \mathbf{M}_2 = \begin{bmatrix} \mathbf{R} & \mathbf{r} \\ \mathbf{0} & \mathbf{d} \end{bmatrix}, \tag{33}$$

and use $\mathbf{M} = \mathbf{M}_1 \cdot \mathbf{M}_2$ as a preconditioner for $\mathbf{A}$. In (33), we choose $\mathbf{l}$, $\mathbf{r}$, $\mathbf{c}$, and $\mathbf{d}$ such that $\mathbf{R}^T \mathbf{l} = \mathbf{a}_{21}$, $\mathbf{L}\,\mathbf{r} = \mathbf{a}_{12}$, and $\mathbf{c}\,\mathbf{d} = \mathbf{a}_{22} - \mathbf{l}^T \mathbf{r}$. Then the preconditioner

$$\mathbf{M} = \mathbf{M}_1 \cdot \mathbf{M}_2 = \begin{bmatrix} \mathbf{M}_{11} & \mathbf{a}_{12} \\ \mathbf{a}_{21}^T & \mathbf{a}_{22} \end{bmatrix},$$

matches $\mathbf{A}$ in its bordered part.

VII. Concluding Remarks

Semiconductor device modeling remains an important tool for exploring novel structures and to estimate the impact of technology (i.e., process) changes. As line widths shrink, interactions between devices or interconnects become relevant; simulation of multiple devices coupled by circuit elements is important, and it often requires three-dimensional modeling. Moreover, device models including energy transport are needed for small device geometries or more exotic structures; these models involve more nonlinear, coupled PDEs than the traditional drift-diffusion equations. For two-dimensional drift-diffusion simulators of individual devices, sparse, direct factorization algorithms were adequate to deal with the linear systems arising from the solution of the nonlinear systems. However, in the more complicated simulation environment described here, iterative methods are essential. Our challenge has been to supply iterative algorithms that approach the same level of robustness as the direct schemes.

An important component of semiconductor process simulation are models for impurity diffusion, which involve multiple species and thereby many coupled advection-diffusion equations. The same iterative methods are applicable to this set of problems as well.

In recent years, there have been many advances in the area of Krylov-subspace methods, and these algorithms have become standard tools for the iterative solution of large, sparse linear systems. Research activities are now beginning to shift away from basic Krylov-subspace methods to the development of better and more robust preconditioners for nonsymmetric systems. For symmetric positive definite systems resulting from scalar self-adjoint PDE's, multi-level preconditioners, such as hierarchical bases, have proven to be very efficient; see, e.g., [27], [26]. The development of similarly efficient multi-level preconditioners for the nonsymmetric linear systems arising from non-selfadjoint PDE's or even coupled systems of PDE's, such as drift-diffusion equations, has hardly begun.

References

[1] J.I. Aliaga, D.L. Boley, R.W. Freund, and V. Hernández, "A Lanczos-type algorithm for multiple starting vectors," Numerical Analysis Manuscript No. 96–18, Bell Laboratories, Murray Hill, NJ, Sep. 1996. (Available on WWW at `http://cm.bell-labs.com/cs/doc/96`)

[2] W.E. Arnoldi, "The principle of minimized iterations in the solution of the matrix eigenvalue problem," *Quart. Appl. Math.*, vol. 9, pp. 17–29, 1951,

[3] O. Axelsson, "A survey of preconditioned iterative methods for linear systems of algebraic equations," *BIT*, vol. 25, pp. 166–187, 1985.

[4] O. Axelsson, *Iterative Solution Methods*, Cambridge: Cambridge University Press, 1994.

[5] R.E. Bank, T.F. Chan, W.M. Coughran, Jr., and R.K. Smith, "The alternate-block-factorization procedure for systems of partial differential equations," *BIT*, vol. 29, pp. 938–954, 1989.

[6] R.E. Bank, W.M. Coughran, M.A. Driscoll, R.K Smith, and W. Fichtner, "Iterative methods in semiconductor device simulation," *Comput. Phys. Comm.*, vol. 53, pp. 201–212, 1989.

[7] W.M. Coughran, Jr., M.R. Pinto, and R.K. Smith. "Continuation methods in semiconductor device simulation," *J. Comp. Appl. Math.*, vol. 26, pp. 47–65, 1989.

[8] I.S. Duff, A.M. Erisman, and J.K. Reid, *Direct Methods for Sparse Matrices*, Oxford: Oxford University Press, 1986.

[9] R.W. Freund. "A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems," *SIAM J. Sci. Comput.*, vol. 14, pp. 470–482, 1993.

[10] R.W. Freund, G.H. Golub, and N.M. Nachtigal, "Iterative solution of linear systems," *Acta Numerica*, vol. 1, pp. 57–100, 1992.

[11] R.W. Freund, M.H. Gutknecht, and N.M. Nachtigal, "An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices," *SIAM J. Sci. Comput.*, vol. 14, pp. 137–158, 1993.

[12] R.W. Freund and M. Malhotra, "A block QMR algorithm for non-Hermitian linear systems with multiple right-hand sides," *Lin. Alg. Appl.*, vol. 254, pp. 119–157, 1997.

[13] R.W. Freund and N.M. Nachtigal, "QMR: a quasi-minimal residual method for non-Hermitian linear systems," *Numer. Math.*, vol. 60, pp. 315–339, 1991.

[14] R.W. Freund and N.M. Nachtigal, "An implementation of the QMR method based on coupled two-term recurrences," *SIAM J. Sci. Comput.*, vol. 15, pp. 313–337, 1994.

[15] R.W. Freund and N.M. Nachtigal, "QMRPACK: a package of QMR algorithms," *ACM Trans. Math. Software*, vol. 22, pp. 46–77, 1996.

[16] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *J. Res. Nat. Bur. Standards*, vol. 45, pp. 255–282, 1950.

[17] C. Lanczos, "Solution of systems of linear equations by minimized iterations," *J. Res. Natl. Bur. Stand.*, vol. 49, pp. 33–53, 1952.

[18] J.W.H. Liu, "The multifrontal method for sparse matrix solution: theory and practice," *SIAM Rev.*, vol. 34, pp. 82–109, 1992.

[19] M. Lundstrom, *Fundamentals of Carrier Transport*, Reading, MA: Addison-Wesley Publishing Company, 1990.

[20] Y. Saad, *Iterative Methods for Sparse Linear Systems*, Boston: PWS Publishing Company, 1996.

[21] Y. Saad and M.H. Schultz, "GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 7, pp. 856–869, 1986.

[22] S. Selberherr, *Analysis and Simulation of Semiconductor Devices*, Wien: Springer-Verlag, 1984.

[23] P. Sonneveld, "CGS, a fast Lanczos-type solver for nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 10, pp. 36–52, 1989.

[24] H.A. van der Vorst, "BI-CGSTAB: A fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 13, pp. 631–644, 1992.

[25] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design*, 2nd ed. New York, N.Y.: Van Nostrand Reinhold, 1993.

[26] J. Xu, "Iterative methods by space decomposition and subspace correction," *SIAM Rev.*, vol. 34, pp. 581–613, 1992.

[27] H. Yserentant, "On the multi-level splitting of finite element spaces," *Numer. Math.*, vol. 49, pp. 379–412, 1986.