UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

The Alternate-Block-Factorization Procedure for Systems of Partial Differential Equations

R.E. Bank
T.F. Chan
W.M. Coughran, Jr.
R.K. Smith

May 1989

CAM Report 89-13

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024-1555

The Alternate-Block-Factorization Procedure for Systems of Partial Differential Equations

R. E. Bank
*Univ. of Calif., San Diego**

T. F. Chan
Univ. of Calif., Los Angeles†

W. M. Coughran, Jr.
AT&T Bell Laboratories‡

R. K. Smith
AT&T Bell Laboratories‡

Abstract

The alternate-block-factorization (ABF) method is a procedure for partially decoupling systems of elliptic partial differential equations by means of a carefully chosen change of variables. By decoupling, we mean that the ABF strategy attempts to reduce intra-equation coupling in the system rather than intra-grid coupling for a single elliptic equation in the system. This has the effect of speeding convergence of commonly used iteration schemes, which use the solution of a sequence of linear elliptic PDEs as their main computational step. Algebraically, the change of variables is equivalent to a postconditioning of the original system. The results of using ABF postconditioning on some problems arising from semiconductor device simulation are discussed.

1 Introduction

In this paper, we are interested in approximately solving a system of elliptic partial differential equations (PDEs) on a domain $\Omega \in \mathbb{R}^d$ with appropriate boundary conditions. Let us write the system in terms of scalar PDEs as follows

$$L(z) = \begin{pmatrix} L_1(z_1, z_2, \ldots, z_m) \\ L_2(z_1, z_2, \ldots, z_m) \\ \vdots \\ L_m(z_1, z_2, \ldots, z_m) \end{pmatrix} = 0 \tag{1}$$

*Dept. of Mathematics, La Jolla, California 92093, USA.
†Dept. of Mathematics, Los Angeles, California 90024, USA.
‡Murray Hill, New Jersey 07974, USA.

where $z(x) \in \mathbb{R}^m$ is a vector function. Systems of this type arise from many applications in science and engineering.

In general, we are interested in a discretized form of (1). Let Ω^h be a triangulation of Ω. Then a finite-difference, finite-element, or finite-volume method can be applied to (1) resulting in

$$L^h(z^h) = \begin{pmatrix} L_1(z_1^h, z_2^h, \ldots, z_m^h) \\ L_2(z_1^h, z_2^h, \ldots, z_m^h) \\ \vdots \\ L_m(z_1^h, z_2^h, \ldots, z_m^h) \end{pmatrix} = 0. \tag{2}$$

If there are ν degrees of freedom associated with the underlying discrete approximation to each PDE, then (2) represents $m\nu$ nonlinear algebraic equations to be solved.

Our particular interest has been in the system of $m = 3$ drift-diffusion equations that occur in semiconductor device modeling [6, 8, 19]. A simplified form of this system is given by

$$L_1(u, n, p) = -\vec{\nabla}^2 u + n - p - N(x) = 0, \tag{3}$$

$$L_2(u, n, p) = \vec{\nabla} \cdot (n\vec{\nabla}u - \vec{\nabla}n) = 0, \tag{4}$$

$$L_3(u, n, p) = -\vec{\nabla} \cdot (p\vec{\nabla}u + \vec{\nabla}p) = 0. \tag{5}$$

(There are one carrier variants of this problem that amount to dropping (5) and p or, sometimes, (4) and n.) We assume that Ω is a simply connected polygonal region. $N(x)$ is the given doping profile. The unknown dependent variables are the electrostatic potential, u, and electron and hole densities, n and p, respectively; we refer to u, n, and p as the *primitive* variables. Dirichlet boundaries conditions are normally given on part of the boundary, $\partial\Omega_1 \subset \partial\Omega$, with homogeneous Neumann conditions elsewhere.

Equations (3)–(5) are sometimes written in terms of *quasi-Fermi* variables, u, v, and w, defined by

$$n = e^{u-v}, \tag{6}$$

$$p = e^{w-u}. \tag{7}$$

In these variables, the equations become

$$L_1(u, v, w) = -\vec{\nabla}^2 u + e^{u-v} - e^{w-u} - N(x) = 0, \tag{8}$$

$$L_2(u, v, w) = \vec{\nabla} \cdot \left(e^{u-v}\vec{\nabla}v\right) = 0, \tag{9}$$

$$L_3(u, v, w) = -\vec{\nabla} \cdot \left(e^{w-u}\vec{\nabla}w\right) = 0. \tag{10}$$

Equations (3)–(5), or (8)–(10), ignore complications like variable mobilities, oxide regions, and the generation and recombination of carriers. However,

(3)–(5) capture some of the difficulties that occur in practice and let us focus on our main interest — solving discrete analogs of (3)–(5). Equation (3) can be discretized by a number of finite-difference, finite-element, or finite-volume methods. Similar discretizations can be applied to (4) and (5), but specialized methods taking advantage of the the convective-diffusive nature of the PDEs are usually employed.

A variety of approaches exists for dealing with the nonlinear equations represented by (2), but we will concentrate on the two principle techniques used in device simulators. For notational simplicity, we will drop the use of the superscript h to denote the discrete forms since usually only the discretized systems will be considered in what follows. The first method for solving (2) is nonlinear Gauss-Seidel [17], known as Gummel's iteration in the device-simulation literature [9]. Suppose an initial guess $(z^0)^T = (z_1^0, z_2^0, \ldots, z_m^0)$ is given. To go from the kth iterate, z^k, to the $(k+1)$st iterate, z^{k+1}, simply solve, for $i = 1, 2, \ldots, m$, the ith PDE

$$L_i(z_1^{k+1}, z_2^{k+1}, \ldots, z_i^{k+1}, z_{i+1}^k, z_{i+2}^k, \ldots, z_m^k) = 0 \tag{11}$$

for z_i^{k+1}. We will refer to this nonlinear Gauss-Seidel approach as the *plug-in* method. One common scheme for solving each of the m scalar PDEs is Newton's method, as we will shortly discuss; we refer to the resulting overall scheme as the Gauss-Seidel-Newton method.

The second approach is to linearize (2) and apply Newton's method. Each step of this method requires solving a system of linear equations and then updating

$$L'(z^k)x^k = -L(z^k), \tag{12}$$

$$z^{k+1} = z^k + d^k x^k \tag{13}$$

where L' is a (possibly approximate) Jacobian and d^k is a suitably chosen scalar damping factor [5]. L' can be viewed as a block matrix

$$L' = \left(\frac{\partial L}{\partial z}\right) = \begin{bmatrix} L_{11} & L_{12} & \cdots & L_{1m} \\ L_{21} & L_{22} & \cdots & L_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ L_{m1} & L_{m2} & \cdots & L_{mm} \end{bmatrix} \tag{14}$$

with $L_{ij} = \partial L_i / \partial z_j \in \mathbb{R}^{\nu \times \nu}$. We refer to this approach as the *coupled* method.

Both the plug-in and coupled methods are widely used in semiconductor device simulators (for example, see [6, 19, 18, 4, 12, 3]). The plug-in method results in smaller, mathematically more tractable systems of equations. If the PDEs in (2) are weakly coupled, the convergence can be quite rapid. When the PDEs are strongly coupled, the convergence of the outer plug-in iteration can be quite slow, or even diverge. The coupled method takes the interactions between

the equations into account (through the off-diagonal blocks of L'). However, dealing with (12) can be arduous because the entire linearized system must be computed, assembled, and solved.

Equation (12) can itself be solved by an inner iteration. An obvious approach is to employ a block Gauss-Seidel iteration, which would require solving systems involving the diagonal blocks, L_{ii}, of the Jacobian; we refer to this approach as the Newton-Gauss-Seidel method. These linear equations would be the same as those solved for the the plug-in method if Newton's method were used to deal with the scalar PDEs represented by (11). Hence, in the case of tightly coupled PDEs, the slow convergence of the outer iteration of the plug-in approach will correspond to slow convergence of the inner block iteration for the coupled method.

In both the plug-in and coupled Newton-Gauss-Seidel approaches, the idea is to reduce the solution of a large coupled system into a series of discrete scalar elliptic PDEs. Such PDEs are fairly well understood and a wide variety of algorithms exists to solve them. (Some of the possible algorithms include sparse direct methods, Krylov-subspace iterations, multigrid techniques, and fast Poisson solvers.)

The *alternate-block-factorization* (ABF) technique, which we are about to discuss in detail, can be applied in the context of either the plug-in or coupled iterations, assuming the coupled method makes use of an inner block iteration. (The ABF idea and some results were partially described in [3].) The ABF method was motivated by trying to find a 'better' sequence of scalar problems to solve in order to speed convergence of an outer iteration. This is done by a temporary local change of variables. Algebraically, this amounts to preconditioning (12) on the right (postconditioning it).

The ABF strategy is attractive because it is simple to describe and implement as well as having limited storage requirements. The computational work required, mainly the inversion of ν matrices of order m and the diagonal scaling of matrices corresponding to scalar problems, is generally a lower-order term in the overall work estimate. The ABF approach can substantially reduce the total effort to solve a tightly coupled system.

The remainder of this paper is organized as follows. In § 2, the ABF technique will be derived. Some special cases will be studied in § 3. The results of some computational experiments will be shown in § 4. Some conclusions will be drawn in § 5.

2 The ABF Method

The alternate-block-factorization (ABF) method can be motivated intuitively through the following formal line of reasoning. The plug-in iteration (11) converges slowly or diverges when z_i^k is strongly coupled to the other unknown functions through their PDEs, and thus not well determined by its own PDE.

This suggests that we have somehow made a 'bad' choice in assigning unknowns to equations; it is quite possible that none of the unknown functions is a 'good' choice. This leads us to (temporarily) associate a new unknown function $\zeta_i = \zeta_i(z_1, z_2, \ldots, z_m)$ with the ith PDE. Our goal is to choose the new unknowns such that they can be well determined by the solution of their associated PDEs; we do this by attempting to weaken the coupling between the PDEs. Formally linearizing (1) about the current iterate z^k, we have, for $i = 1, 2, \ldots, m$,

$$\sum_{j=1}^{m} \left\{ \left(\frac{\partial L_i}{\partial z_j} \right) (z^k) \right\} \frac{\partial z_j}{\partial \zeta_i} \delta\zeta_i = -L_i(z^k). \tag{15}$$

When discretized, each $\partial L_i / \partial z_j$ becomes a $\nu \times \nu$ matrix, the ij block of the the Jacobian, L_{ij}. The terms $\partial z_j / \partial \zeta_i$ generally must be considered dense $\nu \times \nu$ matrices. However, if we assume that the coupling between the PDEs is largely localized by grid point, then it is reasonable to approximate all the $\partial z_j / \partial \zeta_i$ terms by diagonal matrices; these matrices are estimated by solving local problems at each grid point, holding the solution at all other grid points fixed. We will now show how this procedure can be performed in practice using local matrix operations.

A description of the ABF approach in terms of matrices is rather straightforward. To simplify things, let us discuss the ABF method assuming that (2) is being solved by the coupled (Newton) approach with a block Gauss-Seidel iteration being used for (12). Hence, we are interested in solving a linear system of algebraic equations

$$Ax = b \in \mathbb{R}^{m\nu}. \tag{16}$$

The matrix, A, can be written in block form as

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1m} \\ A_{21} & A_{22} & \cdots & A_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mm} \end{bmatrix} \tag{17}$$

and $A_{ij} \in \mathbb{R}^{\nu \times \nu}$. Let

$$D = \begin{bmatrix} D_{11} & D_{12} & \cdots & D_{1m} \\ D_{21} & D_{22} & \cdots & D_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ D_{m1} & D_{m2} & \cdots & D_{mm} \end{bmatrix} \tag{18}$$

where $D_{ij} = \mathrm{diag}(A_{ij})$. If D^{-1} exists, we postcondition (16) by

$$(AD^{-1})(Dx) = b. \tag{19}$$

D^{-1} is the ABF postconditioner. We then solve (19) by block Gauss-Seidel (or block SSOR) iteration; the diagonal block systems arising in the Gauss-Seidel iteration can be solved via sparse direct or preconditioned iterative methods [3, 2].

There are two natural blockings (ordering of variables) for the system represented by (16). The first is the one given by (17) where the matrix blocks are associated with PDEs, that is, $A_{i\star}$ corresponds to the ith PDE. With this PDE blocking, the nonzero structure of the individual matrices, A_{ij}, captures the connectivity of the underlying spatial mesh, Ω^h. The nonzero structure of the block matrix, A, represents the coupling between the PDEs.

There is an *alternate blocking* that associates matrix blocks with individual points in the mesh (or degrees of freedom). In this case, the reordered matrix, $\tilde{A}$, consists of $\nu \times \nu$ blocks with each block being $m \times m$. The nonzero structure of the block matrix now represents the connectivity of the mesh while the individual matrices reflect the coupling between the PDEs.

Let us now consider how A and $\tilde{A}$ are related. There is a permutation matrix, P, such that

$$\tilde{A} = PAP^T = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} & \cdots & \tilde{A}_{1\nu} \\ \tilde{A}_{21} & \tilde{A}_{22} & \cdots & \tilde{A}_{2\nu} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{A}_{\nu 1} & \tilde{A}_{\nu 2} & \cdots & \tilde{A}_{\nu\nu} \end{bmatrix} \tag{20}$$

where

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

Once again, note that A is the matrix blocked by PDE while $\tilde{A}$ is alternately blocked by grid point. Let

$$\tilde{D} = PDP^T. \tag{22}$$

Obviously, $\tilde{D}^{-1}v$ can be computed using dense matrix techniques since it consists of $m \times m$ matrices on its diagonal (and we are assuming that $m \ll \nu$). Thus, the ABF-postconditioned matrix, AD^{-1}, can be formed by local operations; by construction, D^{-1} eliminates the diagonals of the off-diagonal blocks of A as we will now see with an example.

For specificity, consider a 2×2 block system arising from a system of two PDEs

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \tag{23}$$

where each $A_{ij} \in \mathbb{R}^{\nu \times \nu}$. Let D be given by

$$D = \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} = \begin{bmatrix} \operatorname{diag}(A_{11}) & \operatorname{diag}(A_{12}) \\ \operatorname{diag}(A_{21}) & \operatorname{diag}(A_{22}) \end{bmatrix} \in \mathbb{R}^{2\nu \times 2\nu}. \tag{24}$$

Assuming $\tilde{D}^{-1}$ exists, the ABF-postconditioned matrix is

$$\begin{aligned} AD^{-1} &= A(P^T \tilde{D}^{-1} P) \\ &= \begin{bmatrix} A_{11}D_{22} - A_{12}D_{21} & A_{12}D_{11} - A_{11}D_{12} \\ A_{21}D_{22} - A_{22}D_{21} & A_{22}D_{11} - A_{21}D_{12} \end{bmatrix} \begin{bmatrix} \delta & 0 \\ 0 & \delta \end{bmatrix} \end{aligned} \tag{25}$$

where

$$\delta = (D_{11}D_{22} - D_{21}D_{12})^{-1}. \tag{26}$$

The ABF-postconditioned matrix has zeroed out the diagonals of the off-diagonal blocks. The ABF method attempts to decouple a block system by reducing the effect of the off-diagonal blocks.

For the one-carrier drift-diffusion equations ((3) and (4) without p), the matrix in (23) becomes

$$A = \begin{bmatrix} -\Delta & I \\ -M & C \end{bmatrix}. \tag{27}$$

Here Δ is a discrete Laplacian so $-\Delta$ is symmetric and positive definite. C is a discretization of a convection-diffusion term so $C + C^T$ is positive definite. Physically, carrier densities should be positive from which it follows that M is symmetric and positive definite. Note that the identity matrix, I, in (27) could be replaced by a mass matrix if a finite-element method was used; the identity was used to simplify matters. Now AD^{-1} has the form

$$AD^{-1} = \begin{bmatrix} (-\Delta \operatorname{diag}(C) + \operatorname{diag}(M))\delta & (-\operatorname{diag}(\Delta) + \Delta)\delta \\ (-M \operatorname{diag}(C) + C \operatorname{diag}(M))\delta & (-C \operatorname{diag}(\Delta) + M)\delta \end{bmatrix}, \tag{28}$$

with

$$\delta = (-\operatorname{diag}(\Delta) \operatorname{diag}(C) + \operatorname{diag}(M))^{-1}. \tag{29}$$

Our goal is to solve (16) via a postconditioned block Gauss-Seidel iteration; with this in mind, let us carefully examine AD^{-1} in (28). We first consider the diagonal blocks, which correspond to scalar linear elliptic PDEs. In terms of solving equations involving these blocks as part of a block iteration, they seem superior to the original diagonal blocks of (27); in particular, the nonsymmetric parts of the diagonal block C has relatively less prominence and there is more positive weight on the diagonal of $-\Delta$. This will help iterative methods for solving linear systems involving these blocks converge more rapidly. As expected, there is a fair amount of cancellation in the off-diagonal blocks of A. This should have the effect of reducing the coupling between the discrete PDEs, thus speeding the convergence of the block iteration.

3 Model Problem Analysis

In this section, we will describe a few special cases. The first examples are idealized Fourier analyses of two one-dimensional convection-diffusion problems with equispaced meshes. Later, we will show how the ABF postconditioner can break down.

We consider the application of ABF to the block 2×2 matrix

$$A = \begin{bmatrix} T & I \\ -\eta T & T + \epsilon S \end{bmatrix}. \tag{30}$$

Here η and ϵ are nonnegative scalars, T is the tridiagonal matrix

$$T = h^{-2}[-1 \;\; 2 \;\; -1], \tag{31}$$

and S is the tridiagonal matrix

$$S = h^{-1}[-1 \;\; 1 \;\; 0]. \tag{32}$$

This problem captures some of the character of the drift-diffusion equations written in terms of the primitive variables, that is, (3)–(5). In particular, η corresponds to a carrier density, say n, while $-\epsilon$ corresponds to the electric field, $E = -\vec{\nabla} u$.

The ABF postconditioner for this example is

$$\begin{aligned} D^{-1} &= \begin{bmatrix} 2h^{-2}I & I \\ -2\eta h^{-2} I & (2h^{-2} + \epsilon h^{-1})I \end{bmatrix}^{-1} \\ &= \sigma^{-1} \begin{bmatrix} (1 + \epsilon h/2)I & -h^2 I/2 \\ \eta I & I \end{bmatrix} \end{aligned} \tag{33}$$

with

$$\sigma = \left(\frac{2}{h^2}\right)\left(1 + \frac{\epsilon h}{2} + \frac{\eta h^2}{2}\right). \tag{34}$$

The postconditioned matrix AD^{-1} is given by

$$AD^{-1} = \sigma^{-1} \begin{bmatrix} (1 + \epsilon h/2)T + \eta I & I - h^2 T/2 \\ \epsilon\eta(S - hT/2) & (1 + \eta h^2/2)T + \epsilon S \end{bmatrix}. \tag{35}$$

Notice that the off-diagonal blocks have zero diagonals, as a result of the postconditioning. To compute the spectral radius of the ABF iteration matrix, we must consider the eigenvalues of the matrix

$$G = \begin{bmatrix} (1 + \epsilon h/2)T + \eta I & 0 \\ \epsilon\eta(S - hT/2) & (1 + \eta h^2/2)T + \epsilon S \end{bmatrix}^{-1} \begin{bmatrix} 0 & h^2 T/2 - I \\ 0 & 0 \end{bmatrix}. \tag{36}$$

We would like to reduce this problem to that of computing the spectral radii of (scalar) 2×2 iteration matrices. This cannot be done in a rigorous fashion because $ST \neq TS$. However, we can get an idea of the behavior of the method through the (formal) application of the Fourier transform [7].

First consider the 2×2 matrix

$$\begin{bmatrix} a & b \\ e & d \end{bmatrix} \tag{37}$$

and the corresponding iteration matrix

$$g = \begin{bmatrix} a & 0 \\ e & d \end{bmatrix}^{-1} \begin{bmatrix} 0 & -b \\ 0 & 0 \end{bmatrix}. \tag{38}$$

A straightforward calculation shows that the spectral radius, $\rho(g)$, is given by

$$\rho(g) = \left| \frac{be}{ad} \right|. \tag{39}$$

To apply the Fourier transform, we ignore the effect of the boundary conditions and consider a discrete form of the function $\exp(ik\pi x)$, where $i = \sqrt{-1}$. Setting $x_j = jh$, we define the (complex-valued) vector v_k with components $\exp(ik\pi x_j)$. Excluding the first and last rows of the matrices, we have

$$Tv_k \approx \frac{2(1-c)}{h^2} v_k, \tag{40}$$

$$Sv_k \approx \frac{(1-c)+is}{h} v_k \tag{41}$$

where $c = \cos(k\pi h)$ and $s = \sin(k\pi h)$.

Using the Fourier transform, it suffices to consider the 2×2 iteration matrix, g in (38), with

$$a = \frac{2(1-c)}{h^2} \left(1 + \frac{\epsilon h}{2}\right) + \eta, \tag{42}$$

$$b = c, \tag{43}$$

$$e = \frac{\eta \epsilon i s}{h}, \tag{44}$$

$$d = \frac{2(1-c)}{h^2} \left(1 + \frac{\eta h^2}{2}\right) + \left(\frac{(1-c)+is}{h}\right) \epsilon. \tag{45}$$

From (39) and some algebraic manipulation, we arrive at an estimate for the spectral radius of the ABF-postconditioned system

$$\rho_{\text{ABF}} = c \left(\frac{h^2 \eta}{(1-c)(2+h\epsilon)+h^2\eta}\right) \left| \frac{h\epsilon i s}{(1-c)(2+h^2\eta+h\epsilon)+h\epsilon i s} \right|. \tag{46}$$

It is easy to see that $\rho_{\mathrm{ABF}} < 1$ for all choices of η, ϵ, and h.

The largest value of ρ_{ABF} occurs for the lowest frequency, where

$$1 - c \approx \frac{\pi^2 h^2}{2}, \tag{47}$$

$$s \approx \pi h, \tag{48}$$

so, in the limit as $h \to 0$,

$$\rho_{\mathrm{ABF}} \approx \left(\frac{\eta}{\pi^2 + \eta}\right)\left(\frac{\epsilon}{\sqrt{\pi^2 + \epsilon^2}}\right) < 1. \tag{49}$$

In the extreme case when $h \to 0$, $\eta \to \infty$, and $\epsilon \to \infty$, we find that $\rho_{\mathrm{ABF}} = 1 - O(h)$.

This, of course, is the behavior in the asymptotic limit as $h \to 0$; it is also possible to consider the behavior with respect to other parameters. For example, holding all parameters except η fixed, it is clear from (46) that $\rho_{\mathrm{ABF}} \to 0$ if $\eta \to 0$ or $\eta \to \infty$. This behavior with respect to η is consistent with empirical results from the semiconductor device model. For small η (weak coupling) or large η (strong coupling), convergence should be quite rapid; there is clearly a maximum of ρ_{ABF} ($\partial \rho_{\mathrm{ABF}}/\partial \eta = 0$) for some $\eta > 0$, corresponding to a case of moderate coupling. Figure 1 shows the unimodal character of the ABF postconditioner. Finally, if all parameters except ϵ are fixed, then $\rho_{\mathrm{ABF}} \to 0$ as $\epsilon \to 0$ or $\epsilon \to \infty$ follows from (46).

Let us contrast these results for the ABF method with the block Gauss-Seidel iteration applied directly to A of (30). Using the Fourier transform technique and (39), we arrive at an estimate for the spectral radius

$$\rho_{\mathrm{PV}} = \left|\frac{h^2 \eta}{(1-c)(2+h\epsilon) + h\epsilon i s}\right|, \tag{50}$$

which is monotonically increasing in η. In order for $\rho_{\mathrm{PV}} < 1$, it is necessary that $\epsilon = \delta\eta$ for δ sufficiently large or, in other words, A must be diagonally dominant. These arguments suggest that Newton-Gauss-Seidel applied to (3)–(5) can diverge, which is in agreement with the results of [18].

Now A in (30) roughly corresponds to the drift-diffusion equations written in terms of the primitive variables, (3)–(5). As noted in § 1, the drift-diffusion equations can also be written in terms of quasi-Fermi variables, u, v, and w; that system is represented by (8)–(10). The analogous matrix corresponding to A for the quasi-Fermi variables is simply

$$B = \begin{bmatrix} T & I \\ -\eta T & T + \epsilon S \end{bmatrix}\begin{bmatrix} I & 0 \\ \eta I & -\eta I \end{bmatrix} = \begin{bmatrix} T + \eta I & -\eta I \\ \eta\epsilon S & -\eta(T + \epsilon S) \end{bmatrix}, \tag{51}$$

where T and S are defined in (31) and (32), respectively. We can consider how B is affected by the application of the ABF postconditioner as well as the unconditioned block Gauss-Seidel iteration.

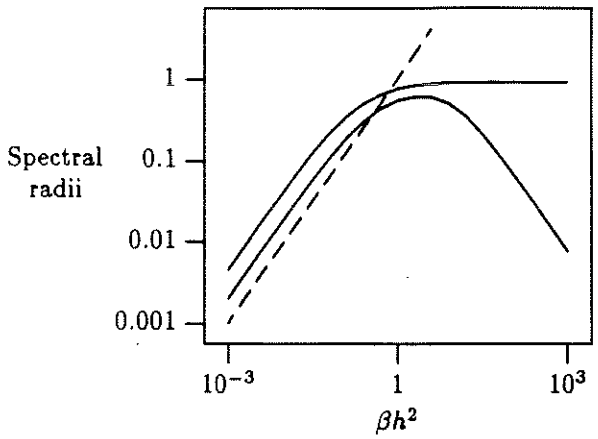

Figure 1: This figure shows: (1) the ABF spectral radius, $\rho_{\mathrm{ABF}}(\eta h^2)$ from (46), as the lower solid line; (2) the block Gauss-Seidel spectral radius, $\rho_{\mathrm{PV}}(\eta h^2)$ from (50), as the dashed line; and (3) the block Gauss-Seidel spectral radius for the equations in the quasi-Fermi variables, $\rho_{\mathrm{QF}}(\eta h^2)$ from (52), as the upper solid line. Here $c = 0.9$ and $\epsilon h = 2$.

If we postcondition B by ABF, then we can estimate the spectral radius by studying the eigenvalues of the matrix, G, appearing in (36) again; this leads to the same approximation, ρ_{ABF} in (46), for the spectral radius. This follows from ABF's insensitivity to the scaling in (51). (Nevertheless, the ABF technique can be effective when applied to the quasi-Fermi variant of the drift-diffusion equations, as we will see in § 4.) Obviously, we can apply block Gauss-Seidel directly to B. After using the local Fourier trick again and some algebraic manipulation, we find that an estimate for the spectral radius is

$$\rho_{\mathrm{QF}} = \left(\frac{\eta h^2}{2(1-c)+\eta h^2}\right)\left|\frac{(1-c)\epsilon h + \epsilon h i s}{(1-c)(2+\epsilon h)+\epsilon h i s}\right|. \tag{52}$$

It is easy to see that $\rho_{\mathrm{QF}} < 1$ and that $\rho_{\mathrm{ABF}} < \rho_{\mathrm{QF}}$ where ρ_{ABF} is given by (46). Moreover, the limit of ρ_{QF} as $h \to 0$ is the expression as shown in (49). Hence, the block Gauss-Seidel procedure applied to (51) is equivalent to applying block Gauss-Seidel to the ABF-postconditioned system, (35), for sufficiently fine meshes.

In practical problems, it is more interesting to compare the various methods when h is a fixed positive quantity. As we just noted, the ABF postconditioner is equally effective when applied to the matrix in (30) or (51). Recall that (30) was motivated by the drift-diffusion equations, (3)–(5), written in terms of u,

n, and p while (51) is motivated by the quasi-Fermi variant, (8)–(10). We have characterized the spectral radii of the various block Gauss-Seidel iterations as summarized below:

- ρ_{ABF} in (46) for the ABF-postconditioned matrix, AD^{-1}, in (35);
- ρ_{PV} in (50) for the unconditioned matrix, A, in (30);
- ρ_{QF} in (52) for the unconditioned quasi-Fermi matrix, B, in (51).

Let us emphasize again that $\rho_{\text{ABF}} < \rho_{\text{QF}}$ which suggests that the ABF approach applied to (3)–(5) or (8)–(10) will converge more rapidly than a Newton-Gauss-Seidel iteration applied to (8)–(10). Figure 1 shows the values of these spectral radii as functions of ηh^2.

Figure 1 illustrates several important points. First, the ABF approach can slow down for intermediate coupling between the equations, but does well when the equations are weakly or strongly coupled. Our estimates suggest that ABF-postconditioned matrix, (35), is uniformly superior for a block Gauss-Seidel iteration than the unconditioned system based on quasi-Fermi variables, (51); this superiority is especially dramatic for strongly coupled systems. From this, we surmise that a coupled Newton-ABF-Gauss-Seidel method should converge faster than a simple Gauss-Seidel-Newton method. Nevertheless, we would expect a Newton-Gauss-Seidel approach based on quasi-Fermi variables to converge based on our heuristic analysis, which concurs with the results reported in [13]. Finally, we can see that a Newton-Gauss-Seidel method for the usual system will diverge when the coupling becomes large [18].

Let us turn to a simple example that shows how the ABF technique can break down. Consider the block matrix

$$W = \begin{bmatrix} I + M & \alpha I \\ \alpha I & I + M \end{bmatrix} \in \mathbb{R}^{2\nu \times 2\nu}. \tag{53}$$

Here, M is symmetric with zeros on its diagonal and having eigenvalues, $\{ \mu_j \}$, in $(-1, 1)$. One possibility for $I + M$ is the block tridiagonal matrix that is obtained from discretizing $-\Delta$ by scaled finite differences on a uniform two-dimensional grid; in this case, M would have 4 nonzero entries of $-1/4$ in most of its rows. The ABF postconditioner would then be

$$D^{-1} = \left(\frac{1}{1 - \alpha^2} \right) \begin{bmatrix} I & -\alpha I \\ -\alpha I & I \end{bmatrix} \tag{54}$$

and so

$$WD^{-1} = \left(\frac{1}{1 - \alpha^2} \right) \begin{bmatrix} (1 - \alpha^2)I + M & -\alpha M \\ -\alpha M & (1 - \alpha^2)I + M \end{bmatrix}. \tag{55}$$

The eigenvalues of the block Gauss-Seidel iteration matrix associated with WD^{-1} in (55) are given by the solutions of the scalar problem

$$\det \begin{bmatrix} (1-\alpha^2+\mu)\lambda & \alpha\mu \\ -\alpha\mu\lambda & (1-\alpha^2+\mu)\lambda \end{bmatrix} = 0. \tag{56}$$

Now, (56) holds when $\lambda = 0$ and

$$\lambda = \frac{\alpha^2\mu^2}{(1-\alpha^2+\mu)^2}. \tag{57}$$

Let μ be a fixed positive eigenvalue of M. Then, from (57), $|\lambda(\alpha)| < 1$ if $0 < \alpha < 1$ or $1+\mu < \alpha < \infty$. However, things can break down for $1 \leq \alpha \leq 1+\mu$ and there is an infinite eigenvalue associated with $\alpha = \sqrt{1+\mu}$. This is not at all surprising since W is also singular for $\alpha = \sqrt{1+\mu}$; such behavior is not characteristic of matrices obtained by discretizing elliptic PDEs. If you change the sign in one of the off-diagonal blocks to $-\alpha I$ in (53), then no singularities occur and the results then correspond to figure 1. Similar results hold for negative μ, but things are a bit worse in that case. Hence, like other preconditioners, ABF is not a panacea.

We can also compute the eigenvalues of the iteration matrix for block Gauss-Seidel applied to W in (53) by solving problems of the form

$$\det \begin{bmatrix} (1+\mu)\lambda & \alpha \\ \alpha\lambda & (1+\mu)\lambda \end{bmatrix} = 0, \tag{58}$$

where μ is an eigenvalue of M. Equation (58) holds when $\lambda = 0$ or $\lambda = \alpha^2/(1+\mu)^2$. For small α, everything is fine; for large α, things can go awry since μ may be quite small.

4 Numerical Experiments

Let us now consider the results of computational experiments on some 'realistic' semiconductor modeling problems. Excluding oxide regions, the PDEs actually used in our computer experiments are the drift-diffusion equations based on either the primitive or quasi-Fermi variables

$$-\vec{\nabla}^2 u + n - p - N = -\vec{\nabla}^2 u + e^{u-v} - e^{w-u} - N = 0, \tag{59}$$

$$\vec{\nabla} \cdot J_n = 0, \tag{60}$$

$$\vec{\nabla} \cdot J_p = 0 \tag{61}$$

where the current densities are now

$$J_n = \mu_n(n\vec{\nabla}u - \vec{\nabla}n) = \mu_n e^{u-v}\vec{\nabla}v, \tag{62}$$

$$J_p = -\mu_p(p\vec{\nabla}u + \vec{\nabla}p) = -\mu_p e^{w-u}\vec{\nabla}w. \tag{63}$$

Here $N(x)$ represents the (net) impurity concentration while μ_n and μ_p are carrier mobility functions. Recall from § 1 that the primitive variables are u, n, and p; the quasi-Fermi variables are u, v, and w, defined by (6) and (7).

We have performed experiments with several semiconductor structures using both coupled and plug-in methods. The plug-in implementation uses Newton's method, (12) and (13), to solve the nonlinear Poisson equation, based on the Slotboom variables [20], for u. The linear electron continuity equation, (60) and (62), is then solved for n; p is then determined from the corresponding linear continuity equation. The coupled algorithm is based on applying Newton's method to (59)–(63) in order to obtain either the primitive variables or the quasi-Fermi variables. The coupled-ABF procedure uses block Gauss-Seidel to solve the Newton correction equations, (12). In our experiments, sparse direct methods were always used for the innermost linear equations. We have presented our most recent experience in more detail elsewhere [3, 2].

It is difficult to relate the general nonlinear semiconductor problem to linear systems of equations involving the matrix, A, in (30). The mobilities, $\mu_\star$, typically are functions of the spatial variable, x, or are nonlinear. The semiconductor equations are normally discretized by an exponentially upwinded scheme (for example, see [6, 19]) that effectively reduces the convective terms in the carrier equations, (60)–(63). Ignoring the variable mobilities and meshes, the semiconductor problem discretized with such a scheme gives rise to matrices analogous to A appearing in (30) or B appearing in (51). In the notation of § 3, ηh^2 can be as large as 10^3 or more while $\epsilon h = O(1)$ for realistic semiconductor simulations. These problems are characterized by the extreme variations in the dependent variables and the nonlinearity of the PDEs.

Figure 2 shows the results of using plug-in and coupled-ABF for a one-dimensional, one-carrier, resistive-bar problem with a constant mobility. For this problem, the coupled algorithm is based on solving for u and n with the Newton-ABF-Gauss-Seidel approach. The plug-in and coupled methods were iterated until approximately six-digit accuracy (in the 2-norm) was obtained; for each outer coupled Newton iteration, the ABF-postconditioned block Gauss-Seidel method was iterated until six-digit accuracy was obtained. (In practice, an adaptive Newton-Richardson strategy should be used to terminate the inner Gauss-Seidel iteration [5, 3].) As the answer is approached, the matrices that arise are exactly of the form found in (30). Note that plug-in slows down with increasing coupling and the ABF approach slows for intermediate coupling, which supports the results of § 3 and mimics figure 1.

We repeated the computations for a two-dimensional two-carrier resistive slab with a modest voltage applied across it. In this case, we solved for u, v, and w in (59)–(63) when applying the coupled Newton-ABF method, but retained the same strategy as before for the plug-in algorithm. The nonlinear iterations were terminated when approximately four-digit accuracy was obtained; the inner block Gauss-Seidel iteration was terminated based on an adaptive Newton-Richardson strategy [5, 3]. For successive constant dopings ranging

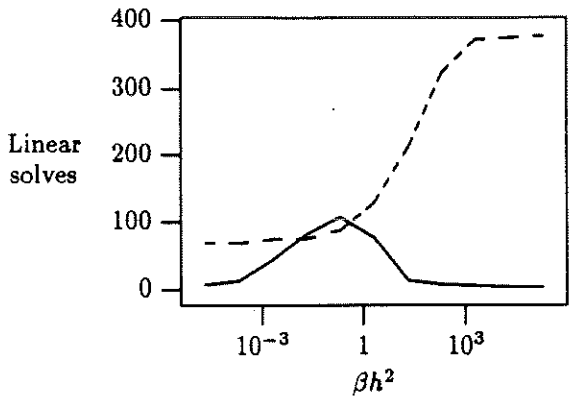

Figure 2: This plot contrasts the number of linear solves for the plug-in algorithm (dashed line) with the number of linear solves for the Newton-ABF-Gauss-Seidel algorithm (solid line) for a one-dimensional resistive bar. For this idealized problem, ηh^2 is a measure of the off-diagonal coupling (see (30)).

from approximately 10^4 up to 10^8 (in the scaled units), the coupled-ABF algorithm used 6 iterations; moreover, only one or two block Gauss-Seidel iterations were usually required per outer Newton iteration. The plug-in iteration used roughly 50 iterations to obtain the same accuracy for that range of doping values. In [3], we reported results obtained by simulating a large essentially uniformly doped structure in a magnetic field; the results there showed that the ABF approach is quite competitive for these drift-dominated devices.

In table 1, some results from the simulation of a small two-dimensional MOS transistor in a high-current state are summarized. We made use of the same solution strategies as just described for the two-dimensional resistive slab. For the coarser grid consisting of 1163 points, the coupled-ABF method reduced the number of linear solves as compared with the plug-in method. The coupled approach used fewer iterations, but required the solution of a matrix of order 3 times larger than the plug-in and coupled-ABF methods. For the finer grid, the ABF scheme was able to substantially reduce both the nonlinear and linear work as compared to the plug-in method. Once again, the ABF preconditioner was remarkably effective for this drift-dominated problem, and only one or two block Gauss-Seidel iterations were usually required per outer Newton iteration.

We also simulated a forward-biased *pn* junction with low doping, where diffusion effects are more prominent. The coupled-ABF approach has more difficulty with this problem. It is sometimes necessary to do 12 or 15 block Gauss-Seidel iterations for each Newton iteration. Further experiments on *pn* junctions and bipolar transistors are needed.

Table 1: Results from various algorithms for a small MOS transistor in saturation.

Nonlinear algorithm	Vertices	Nonlinear iterations	Linear iterations
coupled	1163	14	–
	2765	12	–
plug-in	1163	42	217
	2765	41	200
coupled-ABF	1163	41	123
	2765	26	78

5 Conclusions

In summary, the ABF method automatically provides an alternative set of dimensionless, scaled grid-function unknowns for use in either a plug-in iteration or in the inner iteration of a coupled Newton scheme. These new grid functions are found by examining the coupling between the original unknown grid functions locally at each grid point.

For either approach (plug-in or coupled), the selection of new variables (that is, the computation of the postconditioner, D^{-1}) can be carried out at every iteration step, or less frequently, to the extreme of being computed once at the beginning and then held fixed throughout the calculation. Indeed, as plug-in and coupled are generic terms covering a wide classes of potential algorithms, the number of possible ABF schemes is really unbounded. In fact, the obvious nonlinear variant may prove to be attractive.

The heuristic analysis of some model problems in § 3 (born out by the results in § 4) suggests that the ABF postconditioner avoids some of the convergence difficulties associated with plug-in algorithms for tightly coupled PDEs. With some exceptions, our results also suggest that a Newton-Gauss-Seidel iteration applied to the drift-diffusion equations written in terms of the primitive variables should converge faster if the matrices are postconditioned by ABF; the exceptions occur for cases of moderate coupling when diffusion plays a more significant role. We also considered the drift-diffusion equations written in quasi-Fermi variables where it appears that ABF postconditioning should always improve the convergence rate of a Newton-Gauss-Seidel iteration. For drift-diffusion modeling, our analyses further validate the usual engineering practice of performing plug-in on a nonlinear version of the drift-diffusion equations.

The ABF postconditioner shares some similarities with 'element-by-element' preconditioners [10, 11, 21] and the recently introduced 'transforming smoothers' [22]. The ABF technique is not identical to either of these approaches, as far as we can determine. We have essentially ignored incomplete preconditioners

[16, 1, 15] since we are interested in computations where it is difficult to store just the Jacobian and a few solution vectors in main memory. It has also been suggested that a nonlinear GMRES algorithm can be used to accelerate the plug-in algorithm [14]. Further experimentation and comparisons are needed.

The decisions as to what overall approach to follow, what particular inner and outer iterations to use, and how to incorporate the ABF technique all seem to be highly problem dependent, and the usual array of empirical trade-offs must be taken into account. What we want to emphasize here is that ABF is a simple and easily implemented postconditioner that can have a dramatic effect on the convergence rate of commonly used plug-in and coupled algorithms for systems of nonlinear elliptic PDEs.

Acknowledgements

We gratefully acknowledge the efforts of J. Bürgler and W. Fichtner of the Integrated Systems Laboratory of the Swiss Federal Institute of Technology in studying the behavior of these methods in practice. The work of R. Bank was supported in part by the Office of Naval Research under contract N00014-82K-0197. The work of T. Chan was supported in part by the National Science Foundation under grant NSF-DMS87-14612 and by the Army Research Office under contract DAAL03-88-K-0085.

References

[1] O. Axelsson and N. Munksgaard. A class of preconditioned conjugate gradient methods for the solution of a mixed finite-element discretization of the biharmonic operator. *Int. J. Numer. Math. Eng.*, 14:1001–1019, 1978.

[2] R. E. Bank, J. Bürgler, W. M. Coughran, Jr., W. Fichtner, and R. K. Smith. Recent progress in algorithms for semiconductor device simulation. In R. Burlisch, editor, *Proceedings of the 1988 Oberwolfach Conference on VLSI Modeling*. Birkhäuser Verlag, Basel, 1989. to appear.

[3] R. E. Bank, W. M. Coughran, Jr., M. A. Driscoll, W. Fichtner, and R. K. Smith. Iterative methods in semiconductor device simulation. *Computer Phys. Comm.*, 53, 1989. in press.

[4] R. E. Bank, W. M. Coughran, Jr., W. Fichtner, D. J. Rose, and R. K. Smith. Computational aspects of transient device simulation. In W. L. Engl, editor, *Process and Device Simulation*, pages 229–264. North-Holland, Amsterdam, 1986.

[5] R. E. Bank and D. J. Rose. Global approximate Newton methods. *Numer. Math.*, 37:279–295, 1981.

[6] R. E. Bank, D. J. Rose, and W. Fichtner. Numerical methods for semiconductor device simulation. *IEEE Trans. Electr. Dev.*, ED-30:1031–1041, 1983.

[7] T. F. Chan and H. C. Elman. Fourier analysis of iterative methods for elliptic boundary value problems. *SIAM Review*, 31:20–49, 1989.

[8] W. Fichtner, D. J. Rose, and R. E. Bank. Semiconductor device simulation. *IEEE Trans. Electr. Dev.*, ED-30:1018–1040, 1983.

[9] H. K. Gummel. A self-consistent iterative scheme for one-dimensional steady-state transistor calculations. *IEEE Trans. Electr. Dev.*, ED-11:455–465, 1964.

[10] T. J. Hughes, I. Levit, and J. Winget. An element-by-element solution algorithm for problems of structural and solid mechanics. *Comp. Meth. Appl. Mech. Eng.*, 36:241–254, 1983.

[11] T. J. Hughes, J. Winget, I. Levit, and T. E. Tezduyar. New alternating direction procedures in finite element analysis based upon EBE approximate factorization. In S. Atluri and N. Perrone, editors, *Recent Developments in Computer Methods for Nonlinear Solid and Structural Mechanics*, pages 75–109. ASME, New York, 1983.

[12] T. Kerkhoven. *Coupled and Decoupled Algorithms for Semiconductor Simulation.* PhD thesis, Dept. of Computer Science, Yale Univ., New Haven, 1985.

[13] T. Kerkhoven. A spectral analysis of the decoupling algorithm for semiconductor simulation. *SIAM J. Numer. Anal.*, 25:1299–1312, 1989.

[14] T. Kerkhoven and Y. Saad. Acceleration methods for systems of coupled nonlinear partial differential equations. Technical report, Dept. of Computer Science, Univ. of Illinois at Urbana-Champaign, 1989.

[15] D. S. Kershaw. The incomplete Choleski-conjugate gradient method for the iterative solution of systems of linear equations. *J. Comp. Phys.*, 26:43–65, 1978.

[16] J. A. Meijerink and H. A. van der Vorst. An iterative method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Math. Comp.*, 31:148–162, 1977.

[17] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables.* Academic Press, New York, 1970.

[18] C. S. Rafferty, M. R. Pinto, and R. W. Dutton. Iterative methods in semiconductor device simulation. *IEEE Trans. Electr. Dev.*, ED-32:2018–2027, 1985.

[19] S. Selberherr. *Analysis and Simulation of Semiconductor Devices.* Springer-Verlag, Vienna, 1984.

[20] J. W. Slotboom. Computer aided analysis of bipolar transistors. *IEEE Trans. Electr. Dev.*, 20:669–679, 1973.

[21] J. Winget and T. J. Hughes. Solution algorithms for nonlinear transient heat conduction analysis employing element-by-element iterative strategies. *Comp. Meth. Appl. Mech. Eng.*, 52:711–815, 1985.

[22] G. Wittum. Multi-grid methods for Stokes and Navier-Stokes equations. Transforming smoothers: Algorithms and numerical results. *Numer. Math.*, 54:543–563, 1989.