Algebraic Multigrid for Industrial Semiconductor Device Simulation

Tanja Clees** and Klaus Stüben

Fraunhofer SCAI, Schloss Birlinghoven, 53754 St. Augustin, Germany
tanja.clees@scai.fhg.de, klaus.stueben@scai.fhg.de

Abstract. In this paper, strategies for solving systems of partial differential equations by algebraic multigrid are discussed. In particular, a general framework for so-called point-based approaches is outlined. Several applications from industrial semiconductor process and device simulation have been investigated, and detailed results for two industrially relevant devices are presented. It is shown that this framework allows to construct robust and fast algebraic multigrid approaches even for cases, where iterative one-level solvers of the type commonly used in such applications exhibit bad convergence or even fail.

1 Introduction

Classical algebraic multigrid (AMG) [1,2] is known to provide very efficient and robust solvers or preconditioners for large classes of matrix problems,

$$Au = f \ ,$$

an important one being the class of (sparse) linear systems with matrices A which are "close" to being M-matrices. Problems like this widely occur in connection with discretized *scalar* elliptic partial differential equations (PDEs). In such cases, classical AMG is very mature and can handle millions of variables much more efficiently than any one-level method. Since explicit information on the geometry (such as grid data) is not needed, AMG is especially suited for unstructured grids both in 2D and 3D. In fact, the coarsening process is directly based on the connectivity pattern reflected by the matrix A, and interpolation is constructed based on the matrix entries.

However, extensions of classical AMG are required to efficiently solve *systems* of PDEs involving two or more scalar *functions* (called *unknowns* in the following). This is because classical AMG realizes a *variable-based approach* which does not distinguish between different unknowns. Unless the coupling between different unknowns is very week, such an approach cannot work efficiently for systems of PDEs where, in general, the corresponding matrix A is far from being an M-matrix.

In the past, several ways to generalize AMG have been investigated, and there is still an ongoing rapid development of new AMG and AMG-like approaches. For a review, we refer to [3]. However, there is no unique and best

** formerly Tanja Füllenbach

approach yet. In fact, many problems cannot be tackled at all yet. All approaches seem to have their range of applicability but all of them may fail to be efficient in certain other applications. In this paper, the focus is on extensions of AMG which are direct generalizations of the classical approach.

We first want to recall a rather popular AMG approach to solve systems of PDEs, the so-called *unknown-based approach*, which is very similar to the variable-based approach except that all unknowns are treated separately. To be more specific, let us assume the variables to be ordered by unknowns, that is, $Au = f$ has the form

$$\begin{bmatrix} A_{[1,1]} & \cdots & A_{[1,n_u]} \\ \vdots & \ddots & \vdots \\ A_{[n_u,1]} & \cdots & A_{[n_u,n_u]} \end{bmatrix} \begin{bmatrix} u_{[1]} \\ \vdots \\ u_{[n_u]} \end{bmatrix} = \begin{bmatrix} f_{[1]} \\ \vdots \\ f_{[n_u]} \end{bmatrix} \tag{1}$$

where $n_u > 1$ denotes the number of unknowns of the given system of PDEs, $u_{[n]}$ denotes the vector of variables corresponding to the n-th unknown, and the matrices $A_{[n,m]}$ reflect the couplings between the n-th and the m-th unknown. Using this notation, coarsening the set of variables which correspond to the n-th unknown is strictly based on the connectivity structure reflected by the submatrix $A_{[n,n]}$, and interpolation is based on the corresponding matrix entries. In particular, interpolation to any variable i involves only coarse-level variables corresponding to the same unknown as i. The Galerkin matrices, however, are usually computed w.r.t. all unknowns.

This unknown-based approach has been proposed already in the very early papers on AMG (see [1]). It is certainly the simplest approach for solving PDE systems. By now a lot of experience has been gained with this approach which, in practice, works quite efficiently for many applications. Compared to the variable-based approach, the only additional information required is information about the correspondence between variables and unknowns. The unknown-based approach is mainly used for applications where the diagonal matrix blocks $A_{[n,n]}$ are close to being M-matrices. The essential additional condition for the approach to work is that smoothing the individual equations is sufficient to cause the resulting error to be smooth separately for each unknown. One advantage of this approach is that it can easily cope with anisotropies which are different between the different unknowns. Another advantage is that unknowns can virtually be distributed arbitrarily across mesh points. However, this approach will become inefficient, for instance, if the coupling between different unknowns is too strong.

In this paper, we focus on applications for which classical (variable-based) AMG [1,2] and also the unknown-based approach do not work. In particular, in Section 2, we outline a flexible framework for constructing new, "point-based" AMG approaches to solve various types of PDE systems. In contrast to the unknown-based approach, a point-based approach operates (i.e. coarsens and interpolates) on the level of points rather than variables. Recent results for industrial applications in semiconductor device simulation are presented in

Section 3, showing that suitable point-based AMG approaches yield efficient solution processes. These results are obtained by our code SAMG [4] which is a generalization - based on the framework mentioned before - of the (scalar) code RAMG described in detail in [2].

2 A General Framework for Point-Based Approaches

We talk about a *point-based approach* if, geometrically speaking, coarsening takes place on the level of grid points (rather than variables as before). The resulting hierarchy of grid points is then used for all unknowns, that is, each unknown is associated with the same hierarchy. (Note that this is different from the unknown-based approach where each unknown is associated with its own hierarchy.) Clearly, for any point-based approach, AMG needs additional information on the variable-to-point mapping. We assume corresponding information to be available to AMG.

Remark: Since we have the solution of PDEs in mind, we think of points as being real physical (grid) points in space. However, we want to point out that, from AMG's point of view, it is not important whether "points" really correspond to physical points. Instead, one may think of the nodes of a graph representing the connectivity structure of A. Regarding a point-based approach, it is only relevant for AMG to know whether there are "blocks" of variables (corresponding to different unknowns) which may be treated (coarsened and interpolated) simultaneously.

2.1 Primary Matrix

The framework which we propose allows for many different types of point-based approaches, and it depends on the concrete class of applications which of these possible approaches is meaningful. The idea which is common to all these approaches is that the (point) coarsening (i.e. splitting) process is performed based on some auxiliary (sparse) $(n_p \times n_p)$-matrix $P = (p_{kl})$, called the *primary matrix*, with n_p denoting the number of points. The resulting hierarchy of points is then assigned to all unknowns. For this process to make sense, the primary matrix employed should reflect the physical connectivity (the general structure as well as the strength of connections) of neighboring variables reasonably well, *simultaneously for all unknowns*.

A special point-based approach, sometimes called "block approach", has already been introduced in the very early paper [1] and has been further investigated, for instance, in [6]. To be more specific, we assume the variables to be ordered pointwise, that is, $Au = f$ has the form

$$\begin{bmatrix} A_{(1,1)} & \cdots & A_{(1,n_p)} \\ \vdots & \ddots & \vdots \\ A_{(n_p,1)} & \cdots & A_{(n_p,n_p)} \end{bmatrix} \begin{bmatrix} u_{(1)} \\ \vdots \\ u_{(n_p)} \end{bmatrix} = \begin{bmatrix} f_{(1)} \\ \vdots \\ f_{(n_p)} \end{bmatrix} , \tag{2}$$

where $u_{(k)}$ denotes the "block" of variables located at point k and the $A_{(k,l)}$ represents the "block coupling" between $u_{(k)}$ and $u_{(l)}$. If all unknowns are defined at all points, the $A_{(k,l)}$ are $(n_u \times n_u)$-matrices. Generally, however, they can be smaller and are not necessarily square.

Block coarsening corresponds to defining the primary matrix P by

$$p_{kl} = -||A_{(k,l)}|| \quad (k \neq l) \quad \text{and} \quad p_{kk} = -\sum_{l \neq k} p_{kl} \tag{3}$$

with $||.||$ denoting a suitable norm. Various different norms have been considered in practice, including the maximum, Schur and row sum norm. For the applications considered in Section 3.2, the performance of the employed point-based approach was not sensitively influenced by the concrete choice of the norm. However, in general, it depends on the application (whether block coarsening is meaningful and) which norm is most suitable.

Remark: In (3), p_{kk} has been selected to enforce P to become weakly diagonally dominant. Although this is very natural, it is not really necessary, and there are other choices of p_{kk} which may be advantageous. For instance, the setting $p_{kk} = ||A_{(k,k)}||$ takes also the "size" of the diagonal blocks $A_{(k,k)}$ into account. Sometimes this setting has advantages, for instance, if the matrix P is not only used for the coarsening but also for constructing the interpolation (see next section). However, as far as we have seen for applications of the type discussed in Section 3.2, the performance of the resulting AMG approach does not sensitively depend on the concrete choice of p_{kk}.

Depending on the type of application, there are many other possibilities for defining a primary matrix. Often, this can be done automatically as part of AMG's setup phase. In other cases, it may be better to let the user of AMG provide a reasonable matrix himself, based on his knowledge of the underlying physics of the given problem. In all cases, a primary matrix can usually be interpreted as describing the connectivity structure of some (auxiliary) scalar *primary unknown*. Clearly, the primary unknown should represent the connectivity structure of all "real" unknowns in the given system of PDEs reasonably well.

For instance, in simple cases, one may select $P = A_{[n,n]}$ with n being one of the unknowns of the given system of PDEs. Whether or not this makes sense, depends on the application, in particular, whether the connectivity structure of the n-th unknown is also representative for the other unknowns. Another situation arises if the given problem is isotropic by nature, and anisotropies of the corresponding discretized problem are only due to non-uniform mesh spacings. In such cases, a simple primary matrix might be given by a discretization of the Laplace operator. One can also imagine cases where it makes sense to provide a primary matrix based on some natural physical quantity for which there is no reasonable equation contained in the original system of PDEs, an example being the pressure in the context of the Navier-Stokes equations.

The original AMG did not exploit any geometric information on the given problem. In many PDE applications, this unnecessarily limits the possibilities for an efficient coarsening and interpolation. As a matter of fact, geometric information such as the coordinates of grid points, is usually available and can be exploited in AMG's setup phase. Note that this does not restrict the generality of the grid shape in any respect. If we assume coordinates to be known, P can easily (and automatically) be defined based on distances of points, leading to coarsening processes which are closely related to geometric coarsening. The most simple definition would be

$$p_{kl} = -1/\delta_{kl}^2 \quad (k \neq l) \quad \text{and} \quad p_{kk} = -\sum_{l \neq k} p_{kl} \tag{4}$$

where δ_{kl} denotes the distance between points k and l.

Remark: Since P has to be sparse, only points in small neighborhoods - corresponding to the non-zero pattern of A - are taken into account in (4).

Clearly, since (4) merely relies on distances, settings of this type are only meaningful if anisotropies of the discrete problem are due to non-uniform mesh spacings but not due to anisotropic properties of the continuous problem itself. A coordinates-based P "mimics" a discrete Laplacian here (see above).

Reaction-diffusion systems occurring in semiconductor process simulation have been investigated in [5] as an exemplary class of applications for which the previous, geometrically oriented coarsening process leads to very efficient point-based approaches. Such applications essentially correspond to a superposition of a diffusion system and strong chemical reaction terms. It turns out that the reaction terms are treated efficiently by the smoothing process, so that AMG's coarsening process should essentially be driven by the diffusion part. Since the given matrix A involves both reaction and diffusion terms, and since their influences cannot be separated on the matrix level any more, a coarsening process based on (4) can be regarded as a way to "filter out" the influence of the chemical reactions.

2.2 Interpolation

The main purpose of a primary matrix is the definition of an AMG hierarchy in terms of a nested sequence of points. In addition to the (recursive) coarsening process, also interpolation operators have to be constructed recursively. In practice, there are various possibilities to generalize the interpolation approaches used in classical AMG.

First, the use of *block interpolation* seems most natural, in particular, if P is defined according to (3). That is, formulas to interpolate the error $e_{(k)}$ at a point k are constructed by approximating the block equations

$$e_{(k)} = -A_{(k,k)}^{-1} \sum_{l \neq k} A_{(k,l)} e_{(l)} \tag{5}$$

in a way which is analogous to the classical, variable-wise approaches to define the interpolation. This idea has already been outlined in [1]. In [6,7], direct analoga of the classical *direct* and *standard* interpolation formulas (see [2]) are discussed.

Generally, *direct* analoga of the classical interpolation formulas (described in [1,2]) are obtained by replacing scalar matrix entries by block-coupling matrices $A_{(k,l)}$. All resulting formulas involve in particular inverses of (sums of) block-coupling matrices. Since, in general, it cannot be assumed that these inverses exist, such block interpolation formulas are not robust enough without modification. One possibility to overcome this disadvantage is the replacement of "critical inverses" by inverses of suitable regular diagonal matrices.

In any case, however, the construction of a block interpolation is computationally very expensive. In practice, much simpler types of interpolation often lead to more efficient AMG processes. Thus, besides block interpolations (with modification), we consider (variable-wise defined) interpolation formulas which are either

- separate for each unknown ("u-interpolation") or
- the same for each unknown ("s-interpolation").

Typical ways to define the interpolation *weights* are based on entries in the original matrix A, based on distances and/or positions of points, or based on entries in the primary matrix P. We cannot go into further details here but just want to mention that classical interpolation schemes, as described in [2], such as *direct*, *standard* or *multi-pass interpolation*, can be generalized to this setting in a straightforward way. Also the concept of *aggressive coarsening* carries over. For more details, see [4].

Remark: To derive cheaper formulas, another possibility would be the replacement of *all* matrices $A_{(k,l)}$ by their diagonals $D_{(k,l)}$ in block interpolation schemes, as done in [6,7]. The emerging interpolations resemble variable-wise defined formulas again.

3 Industrial Semiconductor Device Simulation

The general framework outlined in Section 2 formally allows the definition of various concrete algorithms. It seems clear that there exists no unique AMG procedure which will work satisfactorily for all systems of PDEs. Instead, major work consists in developing concrete algorithms separately for certain classes of industrial applications. In this section, we consider the application of point-based AMG to semiconductor device simulation.

Semiconductor process and device simulation aim at the approximative computation of the final shape and doping profile of a semiconductor device and its electrodynamic behavior, respectively. Due to the complexity of the models and grids used, industrial semiconductor simulation is increasingly

recognized as an important and challenging area. Occurring PDE systems include *stress governing* and *reaction-diffusion equations* in process simulation and *drift-diffusion equations* in device simulation, all of which exhibit different numerical difficulties. That straightforward unknown-based AMG is suitable to speed up stress simulations has already been shown in [8]. For reaction-diffusion and drift-diffusion equations, the situation is considerably more complicated. For occurring matrices, classical iterative one-level solvers converge only slowly or even break down, classical (variable-based) AMG usually diverges, and also straightforward unknown-based AMG is not sufficient any more. In contrast to this, suitable point-based AMG approaches, accelerated by BiCGstab or GMRes, can cause remarkable speedups. For exemplary matrices arising from reaction-diffusion systems, this has already been discussed in [5]. Also some promising preliminary results for drift-diffusion systems have been presented there. In the following, we want to discuss the application of point-based AMG approaches to drift-diffusion systems in more detail.

3.1 Drift-Diffusion Systems

The most important results of a device simulation, at least from an engineer's point of view, are *current-voltage characteristics* (*IV-characteristics*) of the device considered. IV-characteristics reflect the dependence of (global) currents on voltages applied to contacts of the device. Currents can be computed from the electrostatic potential ψ and the electron and hole carrier concentrations n and p (which can be regarded as "intermediate" results of a device simulation).

There exists an extended hierarchy of semiconductor models (see [9]), ranging from quasi-hydrodynamic to kinetic and classical to quantum models. The simplest quasi-hydrodynamic model is the standard drift-diffusion system. Compared to more involved models, this simple model might provide less accurate local potentials and concentrations, but often predicts (current densities and) global quantities sufficiently accurately with much less computational effort. Because of this, the standard drift-diffusion system is of highest relevance for industrial simulation and will be considered in the following.

The Stationary Drift-Diffusion System. In an industrial environment, possible steady states of the devices sufficiently long after switching processes are typically investigated. For this purpose, a stationary drift-diffusion system consisting of so-called basic semiconductor equations has to be solved for the electrostatic potential ψ and the electron and hole carrier

concentrations n and p:

$$-\nabla \cdot (\epsilon_\Sigma \nabla\psi) + q(n - p - C) = 0 \qquad \text{Poisson(-type) equation,} \tag{6}$$
$$-\nabla \cdot J_n + qR(\psi, n, p) = 0 \qquad \text{electron continuity equation,} \tag{7}$$
$$\nabla \cdot J_p + qR(\psi, n, p) = 0 \qquad \text{hole continuity equation,} \tag{8}$$

where J_n and J_p are the electron and hole current densities, respectively,

$$J_n = -q(\mu_n n \nabla\psi - D_n \nabla n) \qquad \text{electron current relation,} \tag{9}$$
$$J_p = -q(\mu_p p \nabla\psi + D_p \nabla p) \qquad \text{hole current relation.} \tag{10}$$

ϵ_Σ denotes the permittivity, q the elementary charge, $C = C(x)$ the net impurity concentration (i.e. the doping profile), $R = R_p(\psi, n, p, ...)$ the recombination-generation term, $\mu_n = \mu_n(x, \nabla\psi, ...) > 0$ and $\mu_p = \mu_p(x, \nabla\psi, ...) > 0$ the electron and hole mobilities, respectively, $D_n > 0$ and $D_p > 0$ the electron and hole diffusivities, respectively (for which Einstein's relations $D_n = \frac{k_B T}{q}\mu_n$ and $D_p = \frac{k_B T}{q}\mu_p$ are usually assumed to hold), k_B the Boltzmann constant and T the device temperature. All of these quantities are assumed to be given, in general as functions.

By inserting (9) and (10) into equations (7) and (8), respectively, and multiplying the resulting equations by $1/q$, we obtain the following second-order (elliptic) partial differential equations for ψ, n and p:

$$-\nabla \cdot (\epsilon_\Sigma \nabla\psi) + q(n - p - C) = 0 \ , \tag{11}$$
$$-\nabla \cdot (D_n \nabla n - \mu_n n \nabla\psi) + R = 0 \ , \tag{12}$$
$$-\nabla \cdot (D_p \nabla p + \mu_p p \nabla\psi) + R = 0 \ . \tag{13}$$

Generally, the complete simulation domain $\Lambda \subset \mathbb{R}$ ($d = 2$ or 3) consists of two parts, $\Lambda = \Sigma \cup \Omega$. The first part, Σ, represents the union of all material regions where the above described coupled system of stationary semiconductor equations holds. This is typically the case in the semiconductor regions (often doped silicon, the wafer). The second part, Ω, is defined to be the union of regions for which it is assumed that (nearly) no charge carrier currents can occur as, for instance, in insulating regions. In Ω, the above PDE system (11)-(13) degenerates to the Laplace equation,

$$-\nabla \cdot (\epsilon_\Omega \nabla\psi) = 0 \quad \text{in } \Omega \ , \tag{14}$$

where ϵ_Ω represents the permittivity of the corresponding material layers. For example in case of a MOSFET (a metal oxide semiconductor field effect transistor), Ω represents the gate oxide. Here, the interface between Σ and Ω is the semiconductor/oxide-interface. Ω may also be empty, as in case of a diode.

An n-domain (or n-doped region) is defined to be a subdomain of Σ in which $C(x) > 0$ holds. Analogously, a p-domain (or p-doped region) is defined

to be a subdomain in which $C(x) < 0$ holds. The interface between an n- and an adjacent p-domain is called pn-junction. Such adjacent pn-domains, forming local diodes, determine the modes of operation of a device to a large extent.

The boundary of Λ can be split into two disjoint parts, $\partial\Lambda = \partial\Lambda_p \dot{\cup} \partial\Lambda_a$. $\partial\Lambda_p$ represents those parts of $\partial\Lambda$ which correspond to real "physical" boundary segments, i.e. interfaces to insulating material and contacts. $\partial\Lambda_a$ consists of artificial boundary segments, which are introduced, for example, to reduce the simulation domain and to obtain a "self-contained" device, i.e. to separate it from adjacent devices, if it is embedded in an integrated circuit. Different types of boundary and interface conditions are prescribed on different parts of $\partial\Lambda$ and the interfaces between Σ and Ω, depending on the physical properties of the neighboring material and the type of contacts (see [10], for instance).

The simplified sketch in Fig. 1 illustrates the simulation domain Λ in case of an n-MOSFET. In particular, Σ, Ω, the n-domains (source and drain), the p-domain, the physical boundaries (i.e. the contacts to source and drain and the gate contact) and interfaces to isolating material (other than Ω), the artificial boundaries and the interface between Σ and Ω are shown.

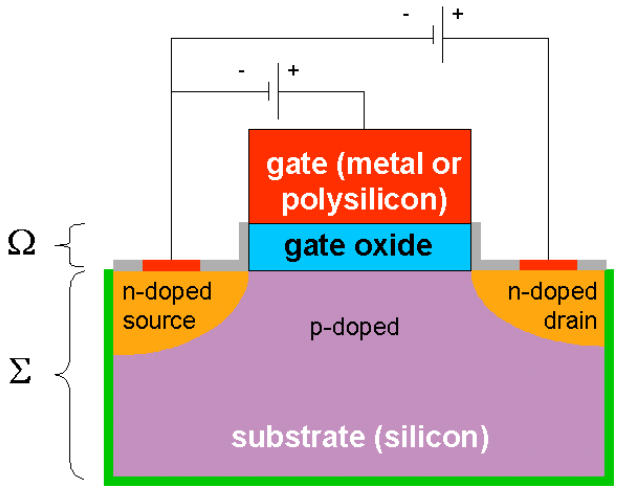

Fig. 1. Simplified sketch of the simulation domain Λ in case of an n-MOSFET. Depicted are Σ, Ω, the p-domain, the two n-domains (source and drain), isolating material (grey) (other than the gate oxide Ω), the contacts (red) and the artificial boundaries (green).

Layer Behavior and Conditioning. A detailed description of the basic semiconductor equations and their mathematical analysis can be found in [9–11], for instance. We just want to summarize some critical aspects.

It is well-known that ψ, n and p are very different in magnitude and show layer behavior. In order to analyze this behavior, one usually scales the system (11) - (13) by so-called "singular perturbation scaling" factors (see [11]). If we denote the scaled quantities by the same symbols as the original ones, equations (12) and (13) keep their form but (11) transforms into

$$\lambda^2 \Delta\psi - (n - p - C) = 0 \tag{15}$$

with λ^2 being a very small constant. As a consequence, this formulation reveals the singular perturbation character of the system. By a singular perturbation analysis, it can be shown that layers occur at pn-junctions, semiconductor/oxide interfaces and so-called Schottky contacts (in reconciliation with physics and numerical results). Outside of these regions, ψ, n and p are moderately varying functions.

A further analysis of (11) - (13) yields statements on the conditioning of these three equations. At least for moderate applied bias (e.g. a moderately large voltage applied to one of the contacts), the Poisson equation (11) is well-conditioned with respect to ψ. The electron continuity equation (12) (the hole continuity equation (13)) is well-conditioned with respect to n (p) only if every n- and p-domain has a contact. However, if an n- or p-domain has no contact (a so-called floating region), the continuity equation for the majority carrier concentration of this region is ill-conditioned. The errors might be amplified by a factor of $O(\lambda^{-4})$ and, therefore, floating regions can produce great numerical difficulties in computing carrier concentrations. An industrially very important example for a floating region is the channel domain in a device (e.g. a FinFET) fabricated by silicon-on-insulator (SOI) technology. Since SOI is one of the standard technologies nowadays, such problematic floating regions occur quite often in device simulation.

Discretization and Linearization. While the discretization of the Poisson(-type) equation (11) is straightforward, the discretization of the continuity equations, which can be characterized as special diffusion-convection-reaction equations, is crucial for an efficient solution of the drift-diffusion system. In practice, the system is discretized by a box method (BM) on (nearly perfect) Delaunay meshes. To gain stability, a special purpose discretization approach, the so-called Scharfetter-Gummel (SG) approach (see also [11]), is used to discretize the continuity equations.

The SG-BM scheme can be outlined as follows. We start with the assumption that, along mesh edges, the mobilities μ_n and μ_p are constant, and the electrostatic potential ψ behaves as a linear function. Approximations of J_n and J_p can then be obtained by solving a one-dimensional boundary value problem. This leads to an exponentially fitted scheme for the current

relations. The emerging approximations of the edge current densities are employed to obtain the final discretization of the continuity equations (for more details on the SG-BM, see [12], for instance).

In commercial device simulators, the resulting highly nonlinear discrete system is typically linearized by a (modified) Newton method yielding a "fully coupled approach" regarding ψ, n, and p. As an alternative, the so-called Gummel iteration decouples the treatment of the individual equations of the full drift-diffusion system: one iteration step consists of solving Poisson's equation (11) for ψ, then solving the electron continuity equation (12) for n, and finally the hole continuity equation (13) for p (see [10], for example).

Since the discretized and linearized systems inherit the layer behavior and conditioning of the original equations, we have to expect layer behavior near pn-junctions, Schottky contacts and semiconductor/oxide interfaces and ill-conditioned continuity equations in floating regions. In spite of the fact that the original system (11)-(13) is usually scaled by DeMari factors (see [10]), ψ, n, p and their discrete analoga are still very different in magnitude.

Due to all these reasons, the arising drift-diffusion matrices are ill-conditioned and often nearly singular even if the underlying problem is far away from a possibly existing bifurcation point. As a consequence, this leads to great difficulties in solving the sequence of matrix equations efficiently. Due to this, the linear, iterative one-level solvers usually employed in industry today, namely ILU- (or ILUT-)preconditioned CGS or BiCGstab, exhibit an unsatisfactory performance in general. A promising possibility to obtain more robust and more efficient methods is the investigation of *hierarchical* approaches such as multigrid.

By now, linear multigrid methods have been developed only for solving the three individual partial differential equations arising during a Gummel(-type) iteration. The most difficult part there is the solution of discretized and linearized continuity equations. For example, in [13], a geometric multigrid method for this type of equation was investigated and successfully applied to some examples on structured grids. In contrast to this, in this paper, we are interested in the solution of the matrix equations occurring in the fully coupled approach. This approach is typically used in modern (commercial) device simulators because it is often more favorable than a Gummel(-type) iteration. In summary, the matrices considered here arise from the fully coupled approach for drift-diffusion systems discretized by the SG-BM on unstructured grids.

In the next section it is demonstrated that a robust and rapidly converging point-based AMG method for the drift-diffusion matrices can be obtained using the framework described in Section 2. It should be noted that, due to our experience, classical (variable-based) [1,2] and unknown-based AMG methods are not robust enough for this kind of matrices. They often diverge and are therefore not discussed in the remainder of this paper.

Remark: In particular until the early nineties, the application of nonlinear geometric multigrid methods (full approximation schemes (FAS)) was

investigated as a further approach to solve drift-diffusion systems (see the references given in [13]). These approaches do not carry over in a straightforward way to the more sophisticated models and unstructured grids used in modern commercial device simulators.

3.2 Numerical Results

According to the different modelling situations in the two parts Σ and Ω of Λ, the finally resulting matrices, A, consist of two parts which are very different by nature. On the one hand, the part A_Ω of A which corresponds to Laplace's equation (on Ω) does not pose problems. Since the submatrix describing the ψ-to-ψ couplings is an M-matrix, and there are only a few couplings to n and p (namely across the interface to Σ), A_Ω is especially suited for AMG.

On the other hand, the system of PDEs given on Σ is tightly coupled. In particular, the corresponding part A_Σ of A is dominated by the couplings *to* the potential ψ. To be more specific[1], either the submatrix $A_{[2,1]}$ or $A_{[3,1]}$ (of A_Σ), depending on the majority carrier concentration, contains a significant part of the largest couplings (measured by absolute value).

Because of this tight coupling between the different PDEs in Σ, unknown-based AMG fails for such applications - although all diagonal block matrices $A_{[n,n]}$ $(n = 1, \ldots, 3)$ are essentially M-matrices. Instead, it has turned out that point-based AMG (used as a preconditioner for BiCGstab) leads to a very robust approach if, for example, the following components are chosen:

- ILU(0) as the smoother,
- a primary matrix based on norms[2] (3),
- an s-interpolation with weights being based on the entries of P.

Generally, to demonstrate the performance of AMG for a given device, it is not sufficient to look at its performance in solving just a few selected linear systems arising as part of a whole simulation. In fact, hundreds of linear systems have to be solved during a full simulation series, most of which have different properties. Consequently, to obtain a clear picture of the performance of AMG, one has to consider full simulation series.

In device simulators such a TAURUS [14], a simulation series for a given device consists of many individual simulations of drift-diffusion systems which differ, for example, in their respective boundary conditions. More precisely, each simulation run starts with the zero bias step, a step in which all voltages are set to zero. Afterwards, several *bias ramps* are applied to the device. For instance, Fig. 7 shows the sequence of bias ramps applied in case of a

[1] assuming the unknowns ψ, n and p to be numbered 1,2 and 3, respectively.

[2] For all drift-diffusion systems tested by now, the performance of the resulting approach was neither sensitively influenced by the concrete choice of the norm nor the concrete choice of p_{kk} (see Section 2.1).

particular FinFET device: In the first 21 simulation steps (the first ramp), the gate voltage is gradually increased from 0 to 1V, keeping the drain voltage fixed at 0.05V. During the next 10 bias steps (the second ramp), the gate voltage is fixed at 1V, and the drain voltage is increased step by step to a value of 1V. For each individual simulation step (i.e. for each applied bias) of such a series, a Newton process is employed to solve the discretized problem, and, within each Newton step, a few matrix solves are necessary. Therefore, during a whole simulation run, several hundred linear systems have to be solved. In (commercial) simulators such as TAURUS, sophisticated control mechanisms are integrated to detect and "repair" possibly occurring difficulties during the linear and nonlinear iterations. In the worst case, this means a bias step rejection and step size reduction.

In order to demonstrate the performance of AMG, we have replaced the iterative linear TAURUS solver (an ILU-CGS method) by SAMG. To demonstrate the effects of coarse-level corrections, we also compare the performance of AMG with that of the corresponding one-level method, i.e. the smoother ILU (both accelerated by BiCGstab).

Up to now, we have tested examples from several classes of industrially relevant semiconductor devices, including a diode, "conventional" MOSFETs[3], STIs[4], an EEPROM[5], several FinFETs[6] and a power bipolar transistor (for general information about these devices, see [15,16], for instance).

In all these cases, the outlined AMG approach was able to solve the arising matrix equations efficiently (see also [5]). In the following, we will present detailed results for two exemplary cases, an EEPROM and a FinFET. Table 1 shows details on the concrete test cases and dimensions of the arising matrix problems. The layout and doping profile of the FinFET example is depicted in Fig. 2. The bias ramps for the EEPROM are depicted in Fig. 3, the ramps for the FinFET in Fig. 7. Each of the remaining graphs shows results (discussed below) for a simulation series. Note that the matrices arising during a full simulation series are always numbered consecutively in these graphs.

Table 1. Details on the device examples (dim: spatial dimension, n_Σ: number of regions in Σ, n_Ω: number of regions in Ω, n_p: number of points, n_v: number of variables, n_A: number of stored matrix elements ($n_A \geq$ number of non-zeroes)).

Example	dim	n_Σ	n_Ω	n_p	n_v	n_A
EEPROM	3D	1	9	10,493	15,415	310,361
FinFET	3D	2	5	69,092	97,530	1,443,940

[3] metal oxide semiconductor field effect transistors.
[4] shallow trench isolated transistors.
[5] electrically erasable programmable read-only memory cell.
[6] double-gate MOSFET structures in which a thin, fin-shaped body is straddled by the gate forming two self-aligned channels that run along the sides of the fin.

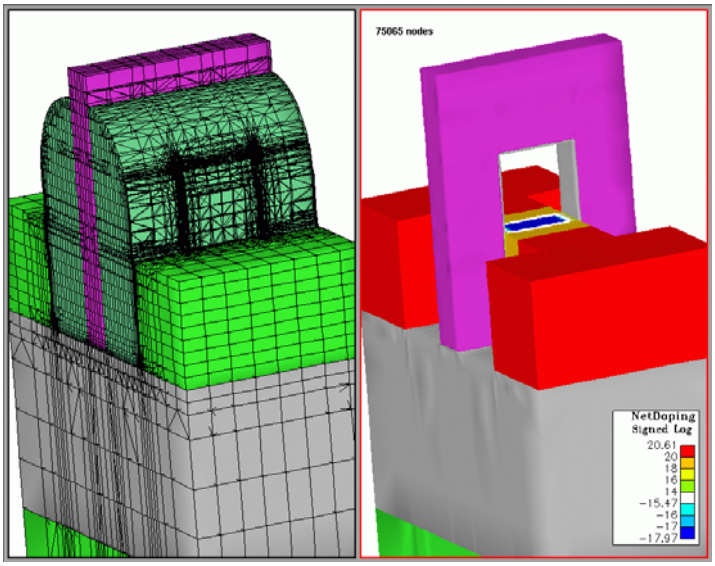

Fig. 2. FinFET example: layout and grid (left), doping profile of the wafer (right). Courtesy of Synopsys Inc.

The FinFET case represents the simulation of a modern device on a moderately large grid. Since a FinFET is fabricated on an SOI wafer, numerical difficulties arise due to the occurring floating region. The EEPROM example, which is rather small in terms of variables, was chosen because it exhibits an additional difficulty: for bias ramp (6), the system is extended by one equation (an algebraic condition) leading to a row with a zero diagonal. Such exceptional rows have to be treated separately during the AMG setup phase. Here, the corresponding row was simply excluded from the coarsening process.

For the examples considered, it can be observed that the linear TAURUS solver (ILU-CGS) does often not fulfill the prescribed convergence criterion (i.e. a residual reduction of at least 10^{-3} within a maximum number of iterations). This is depicted in Figs. 4 and 8 (a value above the yellow line means a violation of the criterion, a value above the red line means divergence of the linear solver for the current matrix). Especially in the EEPROM case, the L2-norm of the last residual, denoted by $||r_e||_2$, is often more than 10^5 times larger than the first residual, $||r_0||_2$.

In contrast to this, the AMG approach shows a stable and fast convergence behavior for both examples. The convergence criterion is fulfilled in all cases, and hence, instead of $||r_e||_2/||r_0||_2$, average residual reduction factors, ρ, are depicted in Figs. 6 and 10. The average residual reduction factors are usually lower than 0.5 and often much better, and less matrix solves were necessary during the Newton steps (see Tables 2 and 3), especially in case of the larger example, i.e. the FinFET.

A comparison of AMG-BiCGstab with the corresponding one-level solver, ILU-BiCGstab, demonstrates that this drastic improvement of robustness and convergence speed is caused to a large extent by employing a hierarchy: In contrast to AMG-BiCGstab, ILU-BiCGstab exhibits average residual reduction factors which are close to or sometimes even larger than 1 (divergence). Additionally, ILU-BiCGstab needs (much) more matrix solves, even more than ILU-CGS (see Tables 2 and 3).

Tables 2 and 3 also show timings for full simulation runs, including meshing and assembling of the matrices (the test character of the TAURUS interface to SAMG leads to extra overhead for transfer of the matrix data to SAMG). Whereas for the smaller EEPROM example the TAURUS solver is considerably faster than AMG-BiCGstab, the efforts for employing the more robust but a bit more expensive AMG approach are paid off for the larger FinFET example.

One should point out that both test cases are still rather small - and too small to demonstrate "real" advantages of AMG over one-level solvers in terms of computational speed. However, since AMG clearly shows a very robust behavior and fast convergence, it can be expected that for increasingly larger problem sizes AMG will be increasingly more efficient than one-level solvers.

Table 2. EEPROM example. Timings and number of necessary matrix solves. "total" is the total wall-clock time in hours needed for the whole simulation run. "SAMG" is the part of "total" which SAMG needed to solve the matrices.

approach	♯ matrices	total	SAMG
TAURUS solver	538	2.38	
AMG-BiCGstab	520	3.81	2.19
ILU-BiCGstab	560	4.31	2.56

Table 3. FinFET example. Timings and number of necessary matrix solves. Description as for Table 2.

approach	♯ matrices	total	SAMG
TAURUS solver	157	4.46	
AMG-BiCGstab	100	4.27	3.25
ILU-BiCGstab	216	11.49	9.15

4 Conclusions

AMG approaches for solving systems of PDEs were presented and discussed. Especially a general framework for point-based approaches was described, which employs a primary matrix to construct a point-based coarsening. Several possibilities for selecting a primary matrix and for the computation of the final interpolation weights were outlined. Recent results for applications in semiconductor device simulation were presented, which demonstrate that robust and rapidly converging point-based AMG methods can be obtained using this framework. The tested point-based methods have the potential to replace one-level preconditioners of the type commonly used in industrial device simulators today.

Acknowledgement. The authors would like to thank Synopsys Inc. for providing a test license of TAURUS with an interface to SAMG and the examples mentioned.

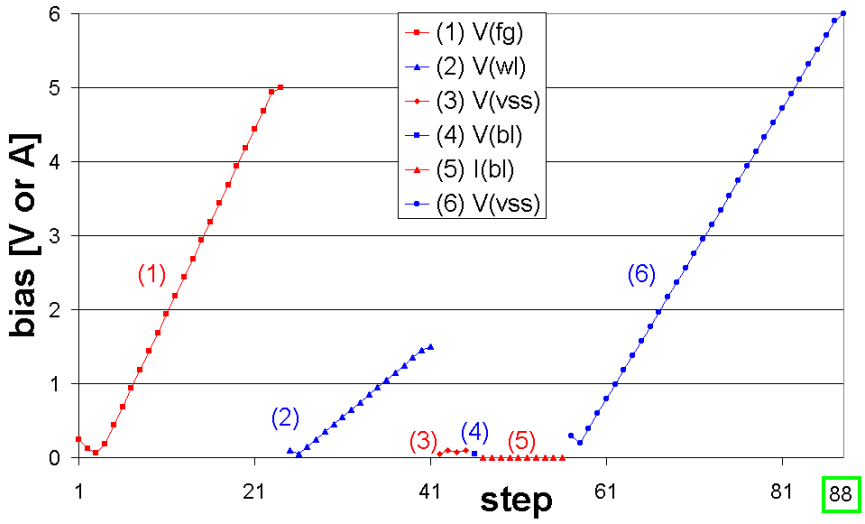

Fig. 3. EEPROM example. Bias steps.

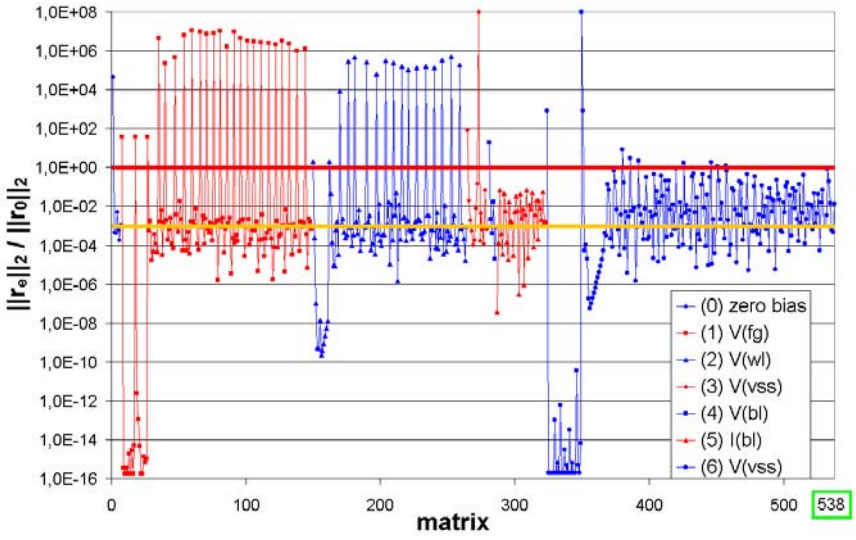

Fig. 4. EEPROM example. Convergence results for TAURUS solver: Last residual, $||r_e||_2 = ||Au_e - f||_2$, divided by first residual, $||r_0||_2 = ||Au_0 - f||_2$.

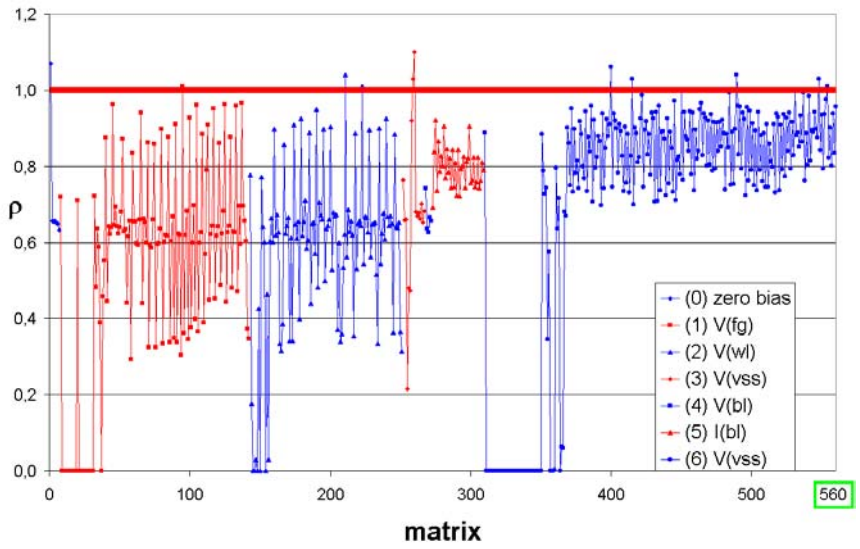

Fig. 5. EEPROM example. Results for ILU-BiCGstab: Average residual reduction factors, ρ.

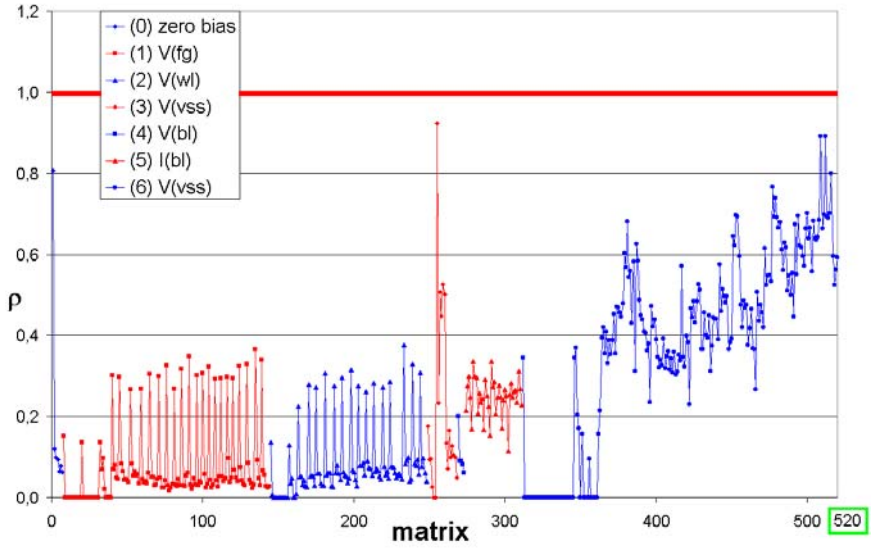

Fig. 6. EEPROM example. Results for AMG-BiCGstab: Average residual reduction factors, ρ.

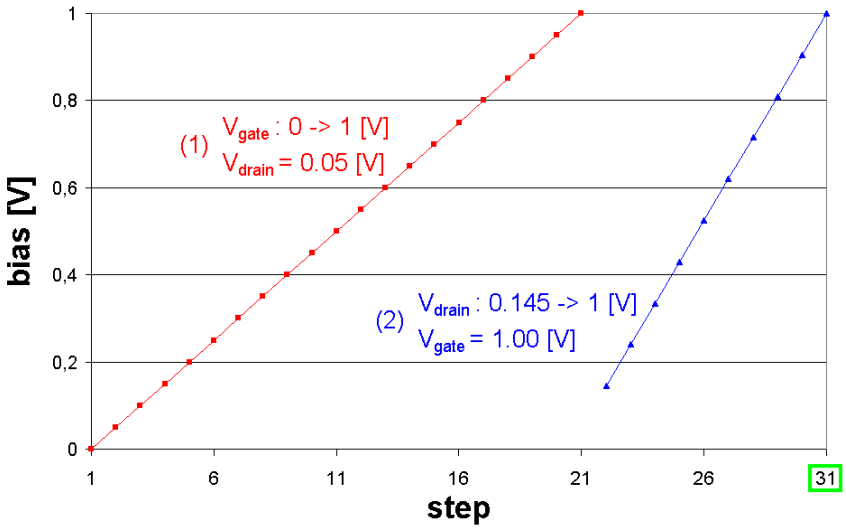

Fig. 7. FinFET example. Bias steps.

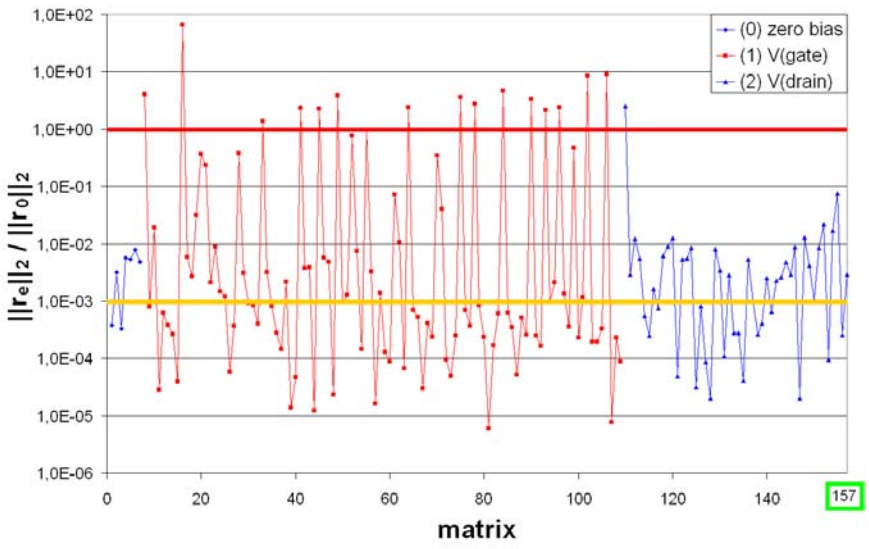

Fig. 8. FinFET example. Convergence results for TAURUS solver: Last residual, $||r_e||_2 = ||Au_e - f||_2$, divided by first residual, $||r_0||_2 = ||Au_0 - f||_2$.

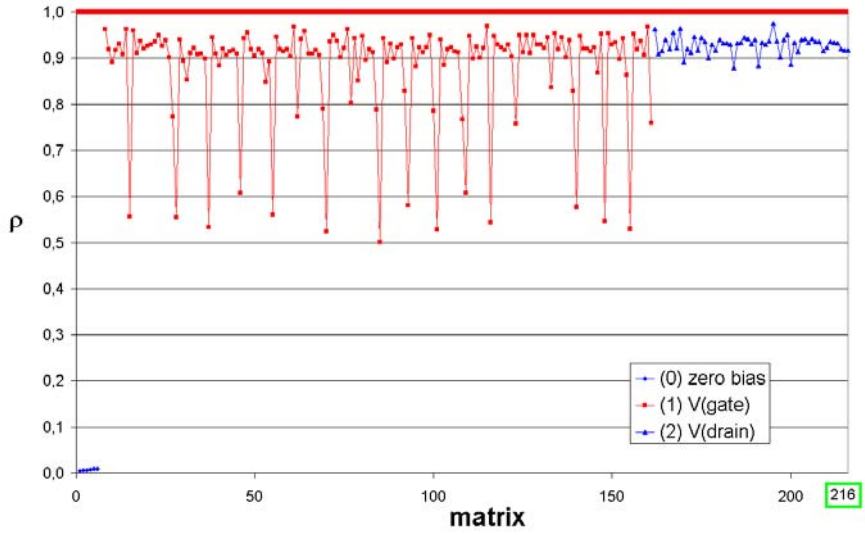

Fig. 9. FinFET example. Results for ILU-BiCGstab: Average residual reduction factors, ρ.

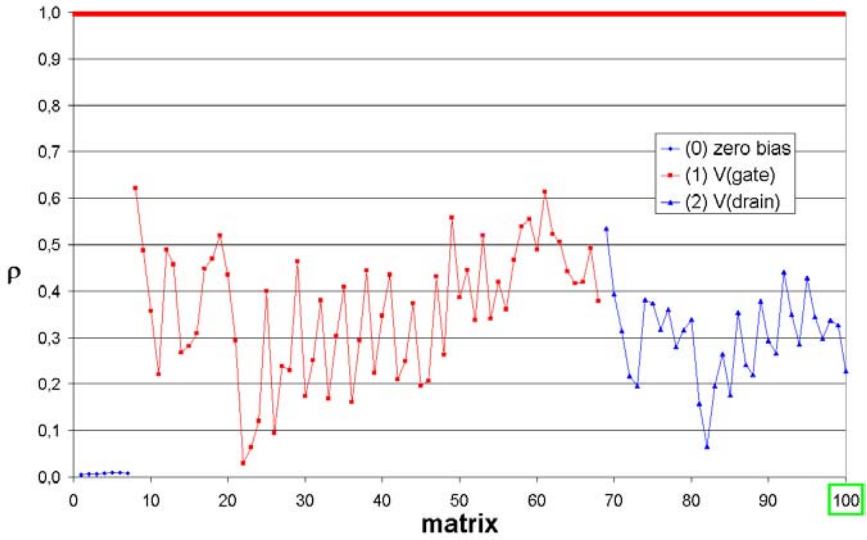

Fig. 10. FinFET example. Results for AMG-BiCGstab: Average residual reduction factors, ρ.

References

1. Ruge, J.W., Stüben, K.: Algebraic Multigrid (AMG). In: McCormick, S.F. (ed.): Multigrid Methods. Frontiers in Applied Mathematics, Vol. 3. SIAM, Philadelphia (1987) 73–130
2. Stüben, K.: An Introduction to Algebraic Multigrid. In: Trottenberg, U., Oosterlee, C.W., Schüller, A.: Multigrid. Academic Press, London San Diego (2001) 413–532
3. Stüben, K.: A review of algebraic multigrid. Journal of Computational and Applied Mathematics **128** (2001) 281–309
4. Stüben, K., Clees, T.: SAMG Release 21b, User's Manual. Fraunhofer SCAI, St. Augustin, Germany (2002)
5. Füllenbach, T., Stüben, K.: Algebraic Multigrid for Selected PDE Systems. In: Elliptic and Parabolic Problems, Rolduc and Gaeta 2001. Proceedings of the 4th European Conference. World Scientific, New Jersey London (2002) 399–410
6. Oeltz, D.: Algebraische Mehrgittermethoden für Systeme partieller Differentialgleichungen. Diplomarbeit, Rheinische Friedrich-Wilhelms-Universität, Bonn (2001)
7. Griebel, M., Oeltz, D., Schweitzer, M.A.: An algebraic multigrid method for linear elasticity. SIAM J. Sci. Comp., submitted
8. Füllenbach, T., Stüben, K., Mijalković, S.: Application of an algebraic multigrid solver to process simulation problems. In: Proceedings SISPAD 2000, International Conference on Simulation of Semiconductor Processes and Devices. IEEE, Piscataway, NJ (2000) 225–228
9. Markowich, P.A., Ringhofer, C.A., Schmeiser, C.: Semiconductor Equations. Springer, Wien New York (1990)
10. Selberherr, S.: Analysis and Simulation of Semiconductor Devices. Springer, Wien New York (1984)
11. Markowich, P.A.: The Stationary Semiconductor Device Equations. Springer, Wien New York (1986)
12. Kerkhoven, T.: On the Scharfetter-Gummel box method. In: Selberherr, S., Stippel, H., Strasser, E. (eds.): Simulation of Semiconductor Devices and Processes, Vol. 5. Springer, Wien New York (1993) 237–240
13. Fuhrmann, J., Gärtner, K.: Incomplete factorization and linear multigrid algorithms for the semiconductor device equations. In: Beauwens, R., de Groen, P. (eds.): Proceedings of the IMACS International Symposium on Iterative Methods in Linear Algebra. Elsevier, Amsterdam (1992) 493–503
14. Taurus, Release 2001.4. Synopsys Inc., Mountainview, CA (2002)
15. Sze, S.M. (ed.): Modern Semiconductor Device Physics. Wiley, New York Chichester (1998)
16. Huang, X., et al.: Sub-50 nm p-channel FinFET. IEEE Transactions on Electron Devices **48** (2001) 880–886