Moving mesh adaptation for Si and GaN-based power device simulation

Fawad Ismail · Palash Sarker · Mohamed Mohamed · Kyekyoon Kim · Umberto Ravaioli

Published online: 24 July 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract
In this paper, we describe the development of moving mesh adaptation framework and its application to charge transport simulation of semiconductor devices, with emphasis on its relevance to power semiconductor devices. Mesh adaptivity in the context of semiconductor device simulation is an important problem and can help deal with the convergence and numerical stability issues, as well as automate the meshing process. We demonstrate the efficacy of our proposed meshing scheme through the simulation of a GaN-based power diode, as well as a Si diode with a non-rectangular doping profile, by externally coupling our framework to Sentaurus Device TCAD. We perform error analysis and compare our results with simulations based on high-resolution uniform structured meshes as well as manually refined axis-aligned meshes. In addition to the benefits in terms of accuracy, automation, and generality, our method can be regarded as a stepping stone toward computationally scalable and adaptive semiconductor device simulations.

Keywords
Semiconductor device simulation · GaN · Gallium nitride · Power · Mesh adaptation · Moving mesh · Error estimation

1 Introduction
In the past few years, there has been a resurgence of research and innovation activity in the area of power devices. This is fueled in part by the ongoing innovation in electronic system miniaturization and enhancement, in the form of mobile devices, tablets, and internet-of-things on the one hand and solar, wind, and other renewable energy systems on the other. In addition, discovery of the suitability of materials like GaN for scaling of power device performance and size has played a large part in further increase in the possibilities of novel power devices and architectures.

Simulation of power devices based on GaN plays an important role in the characterization, analysis, and design of such devices and offers significant cost savings associated with substitution of part of the experimental work with modeling and simulation on computers. This entails a critical review of the some of the features and capabilities of state-of-the-art semiconductor device simulation systems, and an exploration of the opportunities for improvement. One such opportunity for improvement in semiconductor device simulation is the automated generation and adaptivity of the computational mesh. In addition, adaptive meshing is an enabling technology for simulating non-rectangular geometries and interfaces characteristic of real-world semiconductor device structures. As a result, owing to the automated nature of adaptive meshing, a successful mesh adaptation engine can help scale up the design and analysis phase of a semiconductor technology development program, in the form of high-throughput modeling, and the associated time and cost savings. In this paper, we tackle this problem in the context of a Si device and a GaN-based power device.

The relevance of mesh adaptation in regard to scientific simulations involving spatial discretization has been discussed by many researchers over the decades (see [1,2] and references therein). Here, we use the term “mesh adaptation” as an alternative to “adaptive mesh refinement,” since our adaptation consists of both refinement, as well as coarsening of the mesh. Generally, an adaptive scheme is either beneficial or necessary for problems involving sharp features in spatial fields, e.g., a 2-D potential function over...
the simulation domain. Such sharp features are characteristic of multiscale problems. In [3], Alauzet et al. assert that the use of a bad mesh for a computational problem implies an inaccurate solution. Semiconductor device modeling and simulation is multiscale in nature, owing to the doping concentrations spanning multiple orders of magnitude, as well as highly nonlinear character of the phenomena of charge transport in semiconductor devices. Therefore, semiconductor device simulation is a prime candidate for incorporation of mesh adaptation.

The need for mesh adaptation for semiconductor device simulation is evident by the efforts of many researchers that have published on this topic. In this regard, Carey et al. [4] use a quad-tree-based approach for successive refinement of rectangular meshes. In contrast, Son et al. [5] have adapted over an axis-aligned mesh. The dissertation of Schmithusen [6] is mainly focused on the adaptation of axis-aligned meshes for a finite volume solver. Both quad-tree and axis-aligned meshes are alternative meshing schemes compared with unstructured meshes, which is the main topic of this work. The main advantages of unstructured meshes are that it is direction agnostic in the isotropic form, whereas in the anisotropic form, the direction of the meshing triangles can be tailored to the solution fields. Hecht and Marrocco [7], with the first author being the creator of the bamg program, used bamg to explore fixed (i.e., non-moving) adaptive mesh coupled with a finite element solver for a given bias applied to a bipolar transistor.

Our work is an extension of [7] in the direction of moving adaptive meshes [8], as well as a method of coupling that is independent of the type of semiconductor device solver. Moving mesh adaptation is relevant, beyond a traditional (‘fixed’) mesh adaptation, in simulations where the features of physical fields of the solution at the end of a simulation run are different from those in the intermediate stages. In order to resolve those features, as well as reduce the error, the mesh needs to adapt at smaller increments in the intermediate stages and not just at the end of a full simulation. The contrast between a traditional mesh adaptation and a moving mesh adaptation is highlighted in Fig. 1. To the best of our knowledge, there has not been an investigation into the application of moving adaptive meshes, of the unstructured type, to semiconductor device transport problems in the literature.

This paper is organized as follows. We describe our methodology in Sect. 2 while introducing the adaptation formalism behind bamg. We explain the moving mesh adaptation (MMA) algorithm that utilizes bamg coupled with a device solver, 2-D Synopsys Sentaurus TCAD in our case. Section 3 is concerned with the results and related discussion. In Sect. 3.1, we present the application of MMA scheme to two semiconductor problems, a semicircular Si diode, and a GaN power diode. We illustrate the error reduction due to an adaptive mesh compared with a uniform axis-aligned mesh of similar complexity, i.e., same number of mesh points. After this, we apply the mesh adaptation to the quasistationary ramp of the applied bias of the diode, and we compare the errors resulting from a high-resolution uniform mesh, a manually refined mesh, and moving mesh from our method. This is followed by a discussion, in Sect. 3.2, of the relevance of moving mesh adaptation in the context of the simulation of a semiconductor device operating under a reverse bias condition, an important scenario from the point of view of power semiconductor devices. In addition, we comment on the comparison of axis-aligned meshes with unstructured meshes, the notion of gradation control, relevance of anisotropy, as well as extension of our approach to 3-D and higher-order transport models. Finally, we discuss the conclusions in Sect. 4.

2 Methodology

From a broader perspective, our method can be characterized as an indirect coupling scheme between a standalone semiconductor device equation solver, e.g., a drift–diffusion solver based on the finite volume method, and a standalone mesh generation and adaptation tool, or mesher, e.g., an isotropic or anisotropic unstructured mesh generator. For the purpose of presenting this work, we have made use of the 2-D version of Synopsys Sentaurus TCAD as the solver and the 2-D mesh generation program bamg as the mesher. We now describe features of the mesh generation system relevant to this work, followed by the description of our methodology for coupling it to a semiconductor charge transport system.

2.1 bamg Based mesh generation and adaptation

bamg, short for bidimensional anisotropic mesh generator, is a 2-D isotropic/anisotropic unstructured mesh generation and adaptation tool that first appeared in 1998 [9].

There are two major kinds of error estimation strategies used for adapting meshes, residual based and recovery
based. Residual-based error estimation involves computing the residual of the system of differential equations and refining the mesh at locations with higher residual while coarsening where the residual is closer to zero. Recovery-based error estimation, on the other hand, involves computing the Hessian of the relevant physical quantities, i.e., the solution fields of potential, electron concentration, and hole concentration. This step is often known as “gradient recovery” or “Hessian recovery.” The recovered Hessian is then used to compute the metric that is used as a criteria for mesh adaptation. This method is also known as “interpolation error”-based mesh adaptation and is the one used by bamg, the mesh generation and adaptation program used in this paper. The corresponding metric expression is as follows:

\[
M = \left( \frac{1}{\text{err} \cdot \text{coef}^2} \sup(\eta) - \inf(\eta) \right)^p
\]

where \(M\) is the metric used for the generation of the adapted mesh, err is the user-specified error tolerance (a smaller value would generate a mesh with higher point count), coef is a scaling coefficient, \(\eta\) is the solution to be used for metric, \(\nabla\) is the Hessian of the solution, and \(p\) is the power parameter, set to one for first-order methods. If multiple solutions are used, an intersection of all the metrics is used as the final metric. An intersection of two or more metrics, or metric tensor fields, at a given point consists of selection of the metric tensor components in each of the two directions which result in the strongest constraints, or smallest mesh features. In the case of isotropic mesh generations, this can be thought of as a selection of the metric at a point that results in the smallest triangle at that point. This combination of multiple metrics is called intersection because a given metric at a point is interpreted as a prescription of the upper bound on the mesh sizing, as opposed to a prescription of exact mesh size. Therefore, when such a metric is combined with a metric with a smaller upper bound, it is equivalent to a prescription, based on the smaller upper bound, that satisfies both metrics. When two metrics are derived from two different solution fields, e.g., a potential field and an electron density field, the resulting mesh would contain small triangles in regions that have sharp features in the potential field, even if the same region has coarse features in electron density field, and vice versa. The consequence of this intersection is that a mesh would have finer mesh sizing at locations of sharp features of both fields.

Some of the salient features of bamg relevant to our work include:

- Support for multiple solution fields as a source for metric computation.
- Fine control of mesh sizing in the form of a continuous (i.e., floating point) input parameter.
- Interpolation of non-metric solution quantities to the new mesh.
- Utilization of the old mesh during the adaptation process, resulting in minimal modification (i.e., insertion, removal, movement of mesh points, as well as retriangulation) to produce the new mesh.
- High performance, resulting in the ability to invoke the program at every iteration of the semiconductor device solver if necessary.

2.2 Moving mesh adaptation

“Moving” in MMA refers to the dynamic modification of the adaptive mesh as the simulation proceeds. An adaptive mesh that does not move simply consists of multiple full runs of a simulation for the same configuration, with the mesh adapting to the solution with each run. A moving adaptive mesh consists of multiple incremental runs, the increment defined by the nature of the system being solved, e.g., a time evolution, or a numerical continuation, while the mesh adapts to each of the increments.

According to Huang and Russell [8], moving mesh methods can be classified as (a) simultaneous solving of the governing equations and mesh described by a moving mesh PDE, (b) alternative solving of the governing equations and the mesh described by a moving mesh PDE, and (c) alternative solving of the governing equations and the mesh without a moving mesh PDE. Our method belongs to the last of the three classes. It can be seen as a front end to bamg for interfacing with a third-party semiconductor device solver. This separation enables interfacing with any 2-D simulation engine that can export the relevant simulation data to be used by the MMA framework. In this work, we have made use of Sentaurus TCAD by Synopsys [10].

Before we describe the MMA method, we would like to briefly discuss the concept of quasistationary ramp and its significance in the context of Synopsys Sentaurus TCAD. The nonlinear nature of semiconductor simulation is such that a method like Newton–Raphson is not sufficient for a successful simulation run without modifications. One of the well-known modifications to the method is numerical continuation, which also goes by other names, like pseudo-time stepping and virtual time stepping. When this technique is applied to the incremental increase in bias conditions in a device simulation, it often goes by the names quasistationary ramp, bias ramp, etc. Using this technique, we initially apply a fraction of the target applied bias, and we follow this by gradual bias increase with simultaneous monitoring of simulation convergence. For such a scheme, moving mesh adaptation can reduce the likelihood of convergence failure.

MMA can be described as the extension of a regular mesh adaptation scheme to simulations involving time evolution or numerical continuation, such that a new mesh is generated.
more than once during a single simulation run. A regular mesh adaptation scheme runs a complete simulation with a coarse mesh, after which, based on the solution error and metric, a new (adapted) mesh is generated, followed by another complete run. This adaptation is carried out one or more times, until some numerical criteria, such as error reduction below a certain tolerance, or convergence toward a certain number of mesh points, are met. In MMA, on the other hand, simulation does not run to completion using a given mesh. Instead, it only runs up to a given time step or numerical step, before the mesh adaptation is invoked. The mesh adaptation can then proceed one or more times, until some criteria are met, and then, simulation continues toward the next time step. This is illustrated in Fig. 1. Such a scheme can therefore be described by an algorithm as follows:

1. Adapt the mesh over the analytical doping profile.
2. Run solver for Poisson equation only, for the equilibrium case, and adapt the mesh over potential.
3. Run solver for all three semiconductor device equations, for the equilibrium case, and adapt the mesh over potential, electron concentration, and hole concentration.
4. For each stepping interval, going from zero applied bias up to the target bias: 
   (a) Run solver for all three semiconductor device equations.
   (b) Adapt the mesh over the three physical quantities.

The error quantification for the mesh in our method is based on the $L_2$-norm of the error between the reference solution and the solution computed for the adapted mesh. Our error computation scheme is based on the AIAA guidelines [11]. We conduct error analysis in two scenarios. In one scenario, we adapt the mesh to a given analytical function in 2-D, and we use the analytical function itself as the reference. In the second scenario, we use the solver to compute the solution based on the adapted mesh. In this case, we do not have access to the analytical solution. Instead, we use a cubic spline interpolant of a solution computed at relatively high resolution on a uniform mesh as a reference. The use of the interpolant helps further increase the closeness of the solution to a hypothetical accurate reference, as well as provide the ability to sample the solution at arbitrary locations within the simulation domain. The (relative) $L_2$ error norm is given by the following equation:

$$(\text{relative} \text{ error} = \frac{||\text{absolute error}||_{L_2}}{||\text{solution}||_{L_2}} = \sqrt{\frac{\Omega \int_{\Omega} |u - u_h|^2 da}{\Omega \int_{\Omega} |u|^2 da}},)$$

where $u$ is the reference solution and $u_h$ is the solution computed using a given mesh. The computation of the error involves 2-D integration, or cubature, for the triangular mesh.

We use quadpy [12] as the integration library. A triangular cubature scheme requires sampling of function values at arbitrary locations within the triangle. This is doable for the reference solution in both the analytical and the solver interpolant case. For the solution based on the adapted mesh, we compute a piecewise linear interpolant based on the concept of linear triangular irregular network.

We would like to put in perspective our method in relation to the adaptivity classification used in the finite element literature. Adaptive finite element methods belong to one or more of the three categories, $h$-adaptive, $p$-adaptive, and $r$-adaptive. $h$-adaptive refers to adaptivity based on mesh spacing and the resulting increase in number of mesh points. $p$-adaptive refers to the degree $p$ of the polynomials used as trial or test functions ($p = 1$ is known as a linear or first-order method, whereas $p > 1$ consists of higher-order methods). Finally, $r$-adaptivity refers to adaptivity based on movement of mesh points within the simulation domain, while keeping the total number of mesh points fixed. Usually $p$-adaptivity requires the solver to handle higher-order methods, whereas $h$- and $r$-adaptivity can be carried out without the need for any modification in the solver. Since our goal is to be able to couple our standalone mesh adaption engine with an external solver, we restrict our adaptivity to $h$ and $r$ only. In other words, our moving mesh adaptation features increase in, decrease in, as well as movement of mesh points within the simulation domain.

3 Results and discussion

3.1 Device details

The simulated devices in this work consist of an Si diode with a semicircular analytical doping profile, as well a GaN power diode. All simulations were done using Synopsys Sentaurus TCAD. The Si diode has dimensions of 10 µm by 5 µm. The whole device is doped with $1 \times 10^{16}$ cm$^{-3}$ As, whereas the $p$-type doping (B) follows an analytical doping profile (Fig. 2). The physical models used include doping-dependent mobility, SRH recombination, as well as impact ionization in the form of van Overstraeten model.

Figure 3 shows the computed $L_2$ error norm for analytical doping profile for two meshes, a uniform mesh that does not have any refinement and a bamg mesh adapted to the doping profile. In the error-vs-N plot (where $N$ is the number of mesh points), we can see that bamg outperforms a uniform mesh for the same number of mesh points, i.e., the error is lower for bamg. This is due to the sharp fall in doping function value going from $p$-type material to $n$-type. Such features are characteristic of physical quantities involved in a semiconductor device, and our result shows that an adapted mesh is highly beneficial for such devices.
Doping profile of a semicircular \( p-n \)-diode. Semicircle (red) represents \( p \)-type with doping of \( 1 \times 10^{17} \) Boron. Rest of the region (blue) represents \( n \)-type with doping of \( 1 \times 10^{16} \) arsenic. Boron doping profile is represented by the analytical expression:

\[
\frac{1}{1 + \exp\left(50\sqrt{(y-2.5)^2 + x^2 - 2.0}\right) \times 10^{17}}
\]

Fig. 2

Fig. 3

Uniform mesh (top), adapted mesh for a semicircular doping profile (middle), and error versus number of mesh points for uniform (blue) and \bamg\ (orange) mesh (bottom) (Color figure online)

In case of Fig. 4, we analyze a slightly modified form of the device mentioned above, i.e., we reduce the length of the device from 10 \( \mu \)m down to 5 \( \mu \)m (from the right end). This does not affect the junction on the left of the device and is done for the purpose of lower computational work. In this computational experiment, we add a third mesh, which consists of fixed manual refinement around the junction. The refinement is in the shape of a semi-octagon and hence does not perfectly fit the semicircular junction. This can be ameliorated by increasing the width of the refinement region, at the cost of increase in number of mesh points.

The plot shows “moving error” for all three meshes for various applied reverse bias values. At zero applied bias, i.e., equilibrium solution, the error for the refined mesh is somewhat higher than the uniform mesh error, which is still higher than the \bamg\-mesh error. However, as the bias is increased and we leave the equilibrium regime, the error in manually refined mesh increases before settling at a higher level as we keep on increasing the applied bias. This can be attributed to the increase in the depletion region on the side of lower
doping, resulting in the depletion edge moving away from the junction and into a region where its accurate resolution could benefit from a refinement. However, since the manually refined mesh is a fixed mesh, it does not have a refined region away from the junction. This shows the importance of a moving mesh that adapts to the refinement as the depletion edges move in response to the applied bias conditions.

In addition, we simulate the forward bias characteristics of a GaN power diode. This is a $P$-$n$-$n$ diode with $n$ being an $n$-region with doping lower than the adjacent $n$-region. This is known as epitaxial diode, owing to the method of construction of the two $n$-regions using epitaxy, before doping the $n$-region with $p$-dopant. However, for this paper, both the $p$-$n$ and $n$-$n$ interfaces are defined using analytical formulas for the donor (Si) as well as acceptor (Mg).

The diode is 42 $\mu$m by 10 $\mu$m. MMA based mesh for this device is shown in Fig. 6. The physical models used in this device simulation are doping-dependent mobility and SRH recombination. Figure 7 shows the forward bias current–voltage (I–V) characteristics compared with the reference I–V characteristic [13], showing a good match.

### 3.2 Moving mesh and the depletion edge in reverse bias

We discuss the issues with a fixed meshing scheme in the context of a semiconductor device solver by considering the semicircular $p$–$n$–$n$-diode once again. A rule of thumb for the meshing of such a device is to create refined regions around the junction. However, we cannot prescribe how much extent such a refined region should have on both sides of the junction. One suggestion is to use the extent of the depletion region at equilibrium. Such a guideline works well for forward bias simulations of such a diode, when the depletion region only shrinks as the bias is increased, but not for a reverse bias simulation. In reverse bias, the depletion region expands beyond the equilibrium extent. It expands quicker on the side of lower doping. If the intent of the reverse bias simulation is breakdown analysis, the depletion region would have covered a significant fraction of the simulation domain by the time breakdown sets in.

Furthermore, for a $p$–$n$–$n$-diode, and as it is illustrated in Fig. 5, the need for refined meshing can largely be categorized into three, a region around the junction, a region around the depletion edge of the side of higher doping, and a region around the depletion edge of the side of lower doping. Everywhere else within the depletion region, the solution variables are flat. Therefore, refining the whole depletion region is wasteful in the first place. On the other hand, even this scheme does not solve the issues associated with a fixed mesh, since a fixed mesh would assume a certain extent of the depletion region while refining the junction and the two edges. This scheme is likely to fail in forward bias, in addition to reverse bias simulation.

### 3.3 Comparison with axis-aligned mesh

An axis-aligned mesh is constructed by having each rectangle consisting of two right triangles sharing a common hypotenuse. In case of gradation due to mesh size variation, a rectangle consisting of three triangles, two of which are right, can be incorporated. Axis-aligned mesh is a well-known kind of mesh used to simulate semiconductor devices using the finite volume method. It consists of two or more triangles forming a rectangle. A solver based on the naive finite volume method puts severe restrictions on the kind of mesh that can be used with it. Specifically such a mesh has to result in a matrix that satisfies the M-matrix property. Such a property is fairly easy to satisfy using a rectangular box mesh. However, in case of a simplex mesh, i.e., a triangular mesh in 2-D or a tetrahedral mesh in 3-D, the centroid has to lie inside the simplex. This is true of an axis-aligned mesh, as well as an isotropic unstructured mesh, one that is used in this paper. This is also true of anisotropic axis-aligned meshes, i.e., mesh in which the height of the rectangles is different from the width. However, this fails to be a fully general solution, e.g., in cases where anisotropy of the system is not aligned to the two axes. An unstructured mesh can achieve such a general anisotropy, but at the cost of the centroid going outside the triangle. A discretization scheme that works with general meshes, paired with an anisotropic mesh adaptation engine, would be a comprehensive solution to this problem, and we will discuss this issue later in the paper.

MMA is relevant to semiconductor device simulation even when numerical continuation or time evolution is not involved, and that is in the context of I–V curve sweep. In order to generate an I–V curve of an electronic device, semiconductor device equations are solved over multiple applied biases. This typically involves starting from an equilibrium solve (i.e., zero applied bias), followed by the smallest applied bias corresponding to first data point on the curve, which is gradually increased as the simulation proceeds. If this gradual increase in applied bias is coupled with a moving adaptive mesh, it ensures that convergence is achieved at every bias. Often times, the purpose of a quasistationary ramp is twofold, the convergence of the nonlinear system through numerical continuation and generation of I–V curve data points along the way. Adaptive meshing becomes crucial when breakdown analysis of a power device is involved, since close to breakdown, physical instability of the system manifests in numerical sensitivity and accurate placement of mesh points serves to reduce numerical artifacts and inaccuracies in the numerical solution.
Fig. 5 Moving mesh adaptation at three stages of a semicircular $p$–$n$-diode simulation. Top snapshot of moving mesh at the given value of applied bias. Bottom cross sections of electron density, hole density, and potential, along the $x$-axis, in the middle of the device’s span along $y$-axis. Going from left to right, we can see a high-resolution region break off from the area around the junction. This region represents the refinement in response to the edge of the depletion region on the side of low doping ($n$-type) moving away from the junction at a rate faster than that on the side of high doping ($p$-type).
3.4 Gradation control in unstructured meshes

Isotropic unstructured mesh adaptation is related to the gradation control problem [14]. If we have a region in the middle of a domain with vanishingly small thickness, this will result in gradation of the mesh that negates the effects of mesh adaptation. This will force gradation to spread to farther and farther regions on both sides until, for a given thickness $\varepsilon$, the gradation-based mesh sizing will overwhelm the domain, resulting in a high mesh point count, and essential loss of adaptation savings. This is one reason such an adaptation is unsuitable for geometries that have thin regions like GaN-caps or AlGaN-spacers in a GaN HEMT. Anisotropic mesh adaptation can help with this problem. However, it requires the solver to be capable of supporting arbitrary unstructured meshes, something that is a known weakness of the finite volume method. In this regard, Droniou et al. [15], in their review of finite volume schemes, have discussed some of the extensions of the naive finite volume method that claim improved results, one of which shows that a nonlinear flux approximation scheme can be regarded as general purpose. Finally, Bochev et al. [16] have shown that a CVFEM (control volume finite element method) solver based on Scharfetter Gummel upwinding supports general unstructured meshes for the purpose of semiconductor device simulation.

3.5 Higher-order transport models

We have utilized mesh adaptation with a device solver based on drift–diffusion model and finite volume discretization. However, our approach is agnostic to modeling physics and can be applied to higher-order transport models, including those based on hydrodynamic transport, as well as the Boltzmann transport equation. This is especially useful, because even in higher-order models, the sharp features of solutions fields do not vanish, and therefore, such simulations could benefit from mesh adaptation. For deterministic and continuum solvers, as in our case, the approach is applicable without any modifications. However, transport simulations based on stochastic or atomistic approaches typically have high-frequency noise components in the field quantities, either due to the stochastic nature of the simulation, or due to the presence of particles, exhibiting point-like features in the solution fields. For such solvers, it is recommended to perform a smoothing step before using the relevant fields as input for metric computation and mesh adaptation.

3.6 3-D mesh adaptation

This paper is concerned with moving mesh adaptation in 2-D, based on metric tensor fields derived from one or more solution fields of interest. The metric-based approach is general in principle and is applicable to 3-D as well. Loseille [2] have remarked on the availability of multiple 3-D anisotropic mesh generation programs, including one of their own. They have also listed a variety of 2-D mesh programs, including bamg. However, in our experience, the feature sets, robustness, and quality of the generated mesh vary from program to program. We expect this variation to be more pronounced in the case of 3-D, with the appropriate choice of a mesh generation program resulting only after some analysis and benchmarking of the available mesh generators. The two most widely known 3-D unstructured mesh generation programs, Gmsh [17], and TetGen [18], do not provide comprehensive multifield metric-based mesh adaptation similar to bamg. One of the reasons for 3-D programs lagging behind in progress compared to their 2-D equivalents is the practical complexity in implementing metric-based adaptation. As an
example, in 2-D anisotropic meshes, the triangles are either isotropic, or elongated with a given magnitude and orientation. This could be thought of as a “line-like anisotropy.” In 3-D, however, the tetrahedrons have a line-like anisotropy, in which the base triangle of the tetrahedron is small compared with the distance between the out-of-plane vertex and the in-plane base triangle, but they might also exhibit “sheet-like anisotropy,” in which the base triangle is much larger than the out-of-plane vertex of the tetrahedron. In addition, these anisotropies will have orientations in 3-D space, e.g., represented by two angles, as opposed to a 2-D orientation, which only requires one angle to describe the orientation. As part of the future work, we intend to explore 3-D mesh adaptation techniques as applied to charge transport problems in 3-D semiconductor device simulation.

4 Conclusion

To summarize, we have shown an adaptive method based on moving meshes for semiconductor device equations. Pseudo-time stepping solution strategy, as employed in semiconductor simulations, results in movement of the sharper features of various solution fields through the simulation domain. Such movement merits the use of a meshing strategy in which the mesh moves with such features in tandem. We have shown that once such a scheme is employed, the result-ing error reduces, leading to more accurate solution of the semiconductor devices. With mesh adaptation, automation comes free. This relieves the engineer from the manual step of mesh refinement. It also eliminates the likelihood of the creation of a bad mesh as a result of human error. Finally, our results based on non-rectangular doping profiles show that a moving mesh adaptation scheme works well with arbitrary doping and solution features.

References

1. Carey, G.F.: Computational Grids: Generations, Adaptation & Solution Strategies. CRC Press, Baca Raton (1997)
2. Loseille, A.: Unstructured mesh generation and adaptation. Handb. Numer. Anal. 18, 263 (2017). https://doi.org/10.1016/s.hna.2016.10.004
3. Alauzet, F., Loseille, A.: A decade of progress on anisotropic mesh adaptation for computational fluid dynamics. Compu. Aided Des. 72, 13 (2016). https://doi.org/10.1016/j.cad.2015.09.005
4. Carey, G.F., Schmidt, J., Sharma, M.: Computational electronics. In: Hess, K., Leburton, J.P., Ravaïoli, U. (eds.) Springer International Series in Engineering and Computer Science, vol. 113, pp. 37–41. Springer, Berlin (1991). https://doi.org/10.1007/978-1-4757-2124-9_6
5. Son, I., Tang, T.W., Eydeland, A.: Computational electronics. In: Hess, K., Leburton, J.P., Ravaïoli, U. (eds.) The Springer International Series in Engineering and Computer Science, pp. 33–36. Springer, Boston, MA (1991). https://doi.org/10.1007/978-1-4757-2124-9_5
6. Schmittsen, B.: Grid adaptation for the stationary two-dimensional drift-diffusion model in semiconductor device simulation. Ph.D. thesis (2002)
7. Hecht, F., Marrocco, A.: Mesh Adaption and Numerical Simulation of Semiconductor Devices, ECCOMAS-2000. Barcelona, Spain (2000)
8. Huang, W., Russell, R.D.: Adaptive Moving Mesh Methods. Springer, Berlin (2010)
9. Hecht, F.: BAMG: bidimensional anisotropic mesh generator, User Guide. INRIA, Rocquencourt (1998)
10. Sentaurus Device User’s Guide. Synopsys (2016)
11. Guide for the Verification and Validation of Computational Fluid Dynamics Simulations (AIAA G-077-1998(2002)). AIAA Standards, American Institute of Aeronautics and Astronautics, Inc. (1998). https://doi.org/10.2514/4.472855
12. Schlömer, N.: quadpy: Numerical integration (quadrature, cubature) in Python (2018). https://github.com/nschloe/quadpy
13. Sabui, G., Parbrook, P.J., Arredondo-Arechavala, M., Shen, Z.J.: Modeling and simulation of bulk gallium nitride power semiconductor devices. AIP Adv. 6(5), 055006 (2016). https://doi.org/10.1063/1.4948794
14. Alauzet, F.: Size gradation control of anisotropic meshes. Finite Elements Anal. Des. 46(1–2), 181 (2010). https://doi.org/10.1016/j.finel.2009.06.028
15. Droniou, J.: Finite volume schemes for diffusion equations: introduction to and review of modern methods. Math. Models Methods Appl. Sci. 24(08), 1575 (2014). https://doi.org/10.1142/S0218202514400041
16. Bochev, P., Peterson, K., Gao, X.: A new control volume finite element method for the stable and accurate solution of the drift-diffusion equations on general unstructured grids. Comput. Methods Appl. Mech. Eng. 254, 126 (2013). https://doi.org/10.1016/j.cma.2012.10.009
17. Geuzaine, C., Remacle, J.F.: Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. Int. J. Numer. Methods Eng. 79(11), 1309 (2009). https://doi.org/10.1002/nme.2579
18. Si, H.: TetGen, a delaunay-based quality tetrahedral mesh generator. ACM Trans. Math. Softw. 41(2), 11:1 (2015). https://doi.org/10.1145/2629697