International Journal of Heat and Mass Transfer 52 (2009) 4031–4039

Contents lists available at ScienceDirect

International Journal of Heat and Mass Transfer

journal homepage: www.elsevier.com/locate/ijhmt

A multigrid method for the Poisson–Nernst–Planck equations

Sanjay R. Mathur *, Jayathi Y. Murthy

School of Mechanical Engineering, Purdue University, West Lafayette, IN 47907-2088, USA

ARTICLE INFO

Article history:
Received 12 December 2008
Accepted 23 March 2009
Available online 4 May 2009

Keywords:
Poisson–Nernst–Planck equations
Computational method
Multigrid
Finite volume
Ion channel

ABSTRACT

A computational technique for solving the Poisson–Nernst–Planck (PNP) equations is developed which overcomes the poor convergence rates of commonly used algorithms. The coupled Poisson and charge continuity equations are discretized using an unstructured cell-centered finite volume method. A Newton–Raphson linearization accounting for the coupling between the equations through boundary conditions, and the space charge and drift terms, is developed. The resulting linear system of equations is solved using an algebraic multigrid method, with coarse level systems being created by agglomerating finer-level equations based on the largest coefficients of the Poisson equation. A block Gauss–Seidel update is used as the relaxation method. The method is shown to perform well for the transport of K^+ and Cl^- in a synthetic ion channel for driving voltages, surface charges, ion concentrations and channel aspect ratios ranging over several orders of magnitude.

1. Introduction

Charge transport in the presence of an electric field occurs in a wide variety of modern microsystems, both natural and synthetic. Examples include electro-diffusion and electro-kinesis in biological systems [1,2], nanofluidic diodes [3], in a variety of electro-hydrodynamic (EHD) and ion-driven flows [4,5], and in microelectronics [6]. One area that has received particular attention in recent years is charge transport in ion channels [7–9], which occur in all biological cell membranes. Ion channels are formed by the folding of amino acids to form a water channel through the cell membrane [10]. The side chains of the amino acids can be ionized and can carry permanent charge, the nature and strength of which depends on the solute in which they are immersed. The permeability of the ion channel to specific ions, such as Na^+, K^+, and Cl^-, is controlled by this charge distribution. The regulation of the flow of ions in and out of the cell is critical to maintaining the necessary ion concentrations in the cell. A large community of researchers has performed both experimental and computational investigations of biological ion channels [7–16]. Charge transport also plays a critical role in other biological applications. For example, electro-diffusion is integral to the communication between neurons and muscle fibers, which is mediated by the diffusion of neurotransmitters at synapses and their consumption by hydrolyzation reactions in post-synaptic membranes [2].

With the advent of modern micro- and nano-fabrication processes, interest has focused on the similarity of biological ion channels to microelectronic devices [17–19]. Researchers have sought to mimic biological structures in micro- and nano-electro-mechanical systems (MEMS and NEMS) [19]. In the electronics cooling arena, researchers have exploited similar physics to develop EHD-driven micropumps [4]. Here, the primary pumping mechanism is the drag exerted by charged ions on a solvent fluid by an imposed traveling electric field; charge transport occurs primarily due to the electric field, but may also be assisted by fluid convection.

The Poisson–Nernst–Planck (PNP) equations have widely been used to simulate these classes of ion transport, and with careful modeling, good comparisons with experimental data have been obtained in many instances [12,10]. At extremely small length scales, the PNP approach may be erroneous. The errors stem from treating ions as a continuum fluid and ignoring the discrete interactions of individual ions with the domain boundaries. As the domain scale becomes smaller, the physical volume occupied by the ions and the solvent molecules must be accounted for; failure to do so results in an overestimate of ion density [15]. Researchers have sought to extend the applicability of the PNP approach through the use of corrective potentials [13,14]. The PNP approach is particularly useful in simulating synthetic ion channels where channel diameters, of the order of tens of nanometers [18], are large enough to mitigate the shortcomings of PNP theory. Methods designed to address truly nanoscale domains include Brownian dynamics [15,16], and more recently, molecular dynamics [8,9]. Though these techniques have been shown to yield more accurate results for small-length-scale domains, their cost, particularly for typical biological time scales, has thus far been too great to permit widespread use. As a consequence, PNP theory forms the mainstay of most MEMS and NEMS simulations today.

* Corresponding author.
E-mail addresses: smathur@purdue.edu (S.R. Mathur), jmurthy@ecn.purdue.edu (J.Y. Murthy).

0017-9310/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ijheatmasstransfer.2009.03.040

Nomenclature

$\mathbf{A}$	area vector, Jacobian matrix
e	electron charge
$\hat{\mathbf{e}}_\mathbf{n}$	outward unit normal vector to face
$\hat{\mathbf{e}}_\mathbf{s}$	unit vector joining cell centroids
$\mathbf{E}$	flux vector
$\mathbf{F}$	drift vector
$\mathbf{J}_p$, $\mathbf{J}_n$	total flux of p and n
k_B	Boltzmann constant
n	number density of negative charges
p	number density of positive charges
$\mathbf{Q}$	solution vector
$\mathbf{R}$	residual vector
$\mathbf{S}$	source vector
T	temperature
V	volume
x, y	Cartesian coordinates

Greek

$\Delta\mathscr{V}$	volume of control volume
ϵ	relative permittivity
ϵ_0	permittivity of free space
ϕ	potential
Γ	diffusion coefficient
ψ	scalar
σ	surface charge density

Subscripts and superscripts

f	face
0	cell C0
1	cell C1
n	iteration
nb	neighbor

A variety of techniques have been published in the literature to solve the PNP equations. Relatively simple simulation approaches have been taken in the ion channel literature [11,12]. Simple finite difference methods coupled to explicit successive over-relaxation (SOR) schemes have been utilized, with 3-D solutions being published only in the 1990s [12]. Both the Slotboom form [20], which is used to transform the charge transport equations into Laplacians, as well as the primitive Nernst–Planck form, have been used [12]. Reported computational times have been relatively long in these loosely coupled simple iterative techniques, ranging from several minutes to several hours per grid point [12]. More efficient schemes for solving the PNP equations are available in the semiconductor device literature [6,10,14,21–25], where they are referred to as the drift–diffusion equations. But these address much different length scales, and do not, in general, admit geometric complexity since this is not necessary for typical device simulations. Recently, a finite volume scheme coupled to Newton iteration of the underlying non-linear algebraic equations has been implemented in the device simulator PROPHET [10,14,26] and has been used to solve for ion transport in porin channels using a stair-step discretization of the complex pore and membrane geometry [10]. Suitable initial conditions were first generated by solving the Poisson–Boltzmann equation under zero concentration gradients, and continuation techniques were used to incrementally raise the applied bias in order to obtain solutions. Daiguji et al. [18] have reported PNP simulations of synthetic ion channels; however, no details of the underlying computational technique or its performance have been reported. They have also reported simulations coupling the PNP equations to Stokes flow simulations [17], but again, no details of the computational technique have been provided. A hybrid method combining a finite element discretization of the charge continuity equations with a boundary-element simulation of the potential field has been reported in [1,2]; a sequential update of the two equation sets has been reported.

As multigrid methods for the solution of linear algebraic equations reached maturity [27–30], multigrid solutions for the Poisson equation began to appear [21,22]. These simulations provide a robust backbone not only for the PNP equation set, but also for Monte Carlo device simulation. Publications describing multigrid solutions of the entire non-linear PNP equation set (coupling the Poisson with charge continuity) are far less numerous, however. Most published methods only use the multigrid procedure as a linear solver for the individual governing equations. Meza and Tuminaro [23] used a multigrid preconditioner with a conjugate gradient method to solve the Slotboom form [20] of the drift–diffusion equations. They used a Gummel iteration procedure[31], i.e., a sequential solution of each of the PNP equations, for steady state device simulation in the DANCIR code [32]. The resulting solver was shown to be significantly faster and more parallel than that using incomplete lower–upper (ILU) preconditioning. Molenaar [24] employed a mixed finite element discretization of the 2D Poisson and drift–diffusion equations. He used Gummel iteration to resolve non-linearities, and a multigrid procedure for the linear solution of each of the separate PNP equations. The only truly coupled multigrid method we are aware of is the recent work of Clees [25], who developed an algebraic multigrid procedure for a coupled solution of the Poisson and drift–diffusion equations for semiconductor device applications. There has been extensive development of coupled multigrid methods for solving the Navier–Stokes equations in the computational fluid dynamics literature [33–37] which provide guidance on how similar procedures may be developed for the PNP equations.

The objective of this paper is to develop a general, robust, and efficient method for the Poisson–Nernst–Planck equations using an unstructured solution-adaptive finite volume formulation [38]. The method addresses complex geometries, and substantially improves robustness and convergence for strongly non-linear problems and high-aspect-ratio domains through the use of a coupled algebraic multigrid method. The method is verified against a known analytical solution and is found to yield accurate results. It is then applied to the problem of ion transport in a synthetic nanochannel [18] and is shown to perform well for a wide range of operating parameters.

2. Governing equations

The governing equations are the Poisson–Nernst–Planck equations written here for a system of two ion species.

$$\nabla\cdot\epsilon\nabla\phi+\frac{e}{\epsilon_0}(p-n)=0 \tag{1}$$

$$\nabla\cdot\left(\nabla p+\frac{e\nabla\phi}{k_BT}p\right)=0 \tag{2}$$

$$\nabla\cdot\left(\nabla n-\frac{e\nabla\phi}{k_BT}n\right)=0 \tag{3}$$

Here ϕ is the electrostatic potential and p and n are the concentrations of the positively and negatively charged ions, respectively.

Defining the solution vector $\mathbf{Q}\equiv[\phi\ p\ n]^T$, the governing equations can be written in vector form as

$$\mathbf{R}(\mathbf{Q}) \equiv \nabla \cdot \mathbf{E} + \mathbf{S} = 0 \tag{4}$$

E represents the fluxes,

$$\mathbf{E} = \begin{bmatrix} \epsilon\nabla\phi \\ \nabla p - \mathbf{F}p \\ \nabla n + \mathbf{F}n \end{bmatrix} \tag{5}$$

and **S** the source terms

$$\mathbf{S} = \begin{bmatrix} \frac{e}{\epsilon_0}(p-n) \\ 0 \\ 0 \end{bmatrix} \tag{6}$$

The term **F** is proportional to the electric field, $-\nabla\phi$, and is given by

$$\mathbf{F} = -\frac{e}{k_B T}\nabla\phi \tag{7}$$

R in Eq. (4) is the residual vector.

For future use, we also define the total n and p fluxes as:

$$\mathbf{J}_p = \nabla p - \mathbf{F}p$$
$$\mathbf{J}_n = \nabla n + \mathbf{F}n \tag{8}$$

The first term on the right-hand side of Eq. (8) represents the diffusive flux due to gradients in the charge concentration, while the second term on the right-hand side represents the drift flux resulting from the electric field.

3. Finite volume discretization

A finite volume discretization procedure, developed in [38] for simulating fluid flow, is employed. The computational domain is divided into arbitrarily shaped convex polyhedra or cells. A discrete value of the solution vector **Q** is associated with the centroid of each such cell. A typical cell associated with cell C0 is shown in Fig. 1. The formulation admits arbitrary mixes of polyhedral cells, and *hanging nodes*, such as node *b*, are admitted, facilitating solution adaptivity. The governing equations are integrated over each cell in the computational domain, resulting in a set of discrete equations. Thus, integrating Eq. (4) over the cell C0 shown in Fig. 1 yields:

$$\mathbf{R}_0 = \int\int\int_V (\nabla\cdot\mathbf{E})dV + \int\int\int_V \mathbf{S}dV = 0 \tag{9}$$

Here $\mathbf{R}_0$ represents the residual vector for cell C0. Using the divergence theorem, the first volume integral in the above equation may be converted to a surface integral, so that:

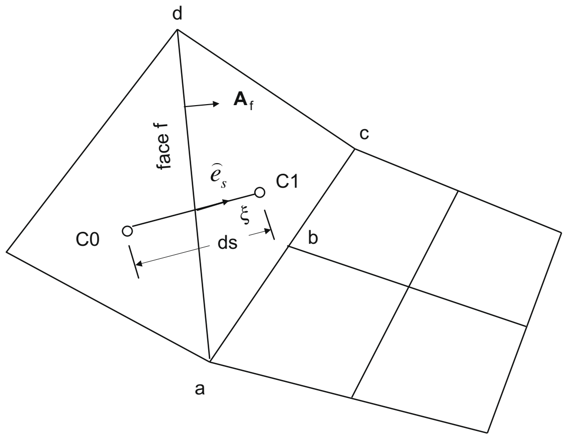

Fig. 1. Control volume.

$$\mathbf{R}_0 = \int\int_A \mathbf{E}\cdot d\mathbf{A} + \int\int\int_V \mathbf{S}dV = 0 \tag{10}$$

Applying the above equation to the control volume C0 in Fig. 1 yields:

$$\sum_f \mathbf{E}_f \cdot \mathbf{A}_f + \mathbf{S}(\mathbf{Q_0})\Delta\mathscr{V}_0 = 0 \tag{11}$$

where the summation is over the faces of the cell C0, $\mathbf{E}_f$ represents the discrete values of the flux vector at a face f and $\mathbf{A}_f$ is the outward-pointing area vector of the face. The procedure for the evaluation of the face flux vector is described next.

3.1. Flux discretization

The face flux $\mathbf{E}_f$ contains terms analogous to convection and diffusion terms that appear in the governing equations for heat or momentum transfer. Expressions for discretizing them on unstructured meshes have been described in detail elsewhere [38]. Only a brief summary is provided here.

Each component of the face flux vector $\mathbf{E}_f$ may be written in the form:

$$(\Gamma\nabla\psi + \mathbf{V}\psi)_f \tag{12}$$

where f refers to a face of cell C0 in Fig. 1. Here, ψ represents ϕ, p or n. Γ takes the value ϵ for $\psi = \phi$ and a value of unity for $\psi = p$ or n. The vector **V** is zero for $\psi = \phi$. **V** is equal to $-\mathbf{F}$ $\left(= \frac{e}{k_B T}\nabla\phi\right)$ for $\psi = p$ and **F** $\left(= -\frac{e}{k_B T}\nabla\phi\right)$ for $\psi = n$.

3.2. Diffusion term

The procedure for discretizing the diffusion flux is described in detail in Ref. [38] and is summarized here. The gradient normal to the face f is decomposed into a component in the direction ξ joining the centroids of cells C0 and C1 (see Fig. 1), and a component normal to it. The unit vector in the ξ direction is given by $\hat{\mathbf{e}}_\mathbf{s}$. Thus, the diffusion component of the flux of an arbitrary scalar ψ with a diffusivity Γ may be written as

$$\begin{aligned}(\Gamma\nabla\psi\cdot\mathbf{A})_f &= \Gamma_f\left[\frac{\partial\psi}{\partial\xi}\frac{(\mathbf{A}_f\cdot\mathbf{A}_f)}{(\mathbf{A}_f\cdot\hat{\mathbf{e}}_\mathbf{s})} + \left(\overline{\nabla\psi}\cdot\mathbf{A}_f - (\overline{\nabla\psi}\cdot\hat{\mathbf{e}}_\mathbf{s})\frac{(\mathbf{A}_f\cdot\mathbf{A}_f)}{(\mathbf{A}_f\cdot\hat{\mathbf{e}}_\mathbf{s})}\right)\right] \\ &= \Gamma_f\left(\frac{\psi_1-\psi_0}{ds}\right)\frac{(\mathbf{A}_f\cdot\mathbf{A}_f)}{(\mathbf{A}_f\cdot\hat{\mathbf{e}}_\mathbf{s})} + \mathscr{S}_f\end{aligned} \tag{13}$$

The first term on the right-hand side of Eq. (13) is termed the *primary gradient* term and is a function of the unknowns on either side of the face. The $\mathscr{S}_f$ term is called the secondary gradient term and is zero for orthogonal quadrilateral/hexahedral meshes, and for equilateral triangular/tetrahedral meshes. The derivative $\overline{\nabla\psi}$, evaluated at the face, is taken to be the average of the gradients at the two adjacent cells.

Eq. (13) is equivalent to a second-order, three-point discretization of the Laplacian operator on a Cartesian grid. It is used to evaluate the first component of $\mathbf{E}_f$ (by setting $\psi \equiv \phi$ and $\Gamma_f \equiv \epsilon$) as well as the diffusion terms (∇p and ∇n) in the second and third components of $\mathbf{E}_f$ (with $\Gamma_f \equiv 1$ and $\psi \equiv p$ and $\psi \equiv n$, respectively).

3.3. Drift term

The remaining terms in the second and third component of $\mathbf{E}_f$ represent the drift (advection) of p and n with an advecting field of $-\mathbf{F}$ and **F**, respectively.

The drift term to be evaluated at the face takes form:

$-(\mathbf{F}p\cdot\mathbf{A})_f$ for the p equation
$(\mathbf{F}n\cdot\mathbf{A})_f$ for the n equation

To evaluate the face values of p and n we use an upwind-biased linear reconstruction [39]. Thus at face f, if $(\mathbf{F}\cdot\mathbf{A})_f > 0$, we have

$$p_f = p_0 + \Psi_p \nabla p_0 \cdot \mathbf{dr}_0 \tag{14}$$

$$n_f = n_1 + \Psi_n \nabla n_1 \cdot \mathbf{dr}_1 \tag{15}$$

where $\mathbf{dr}_0$ is the vector directed from the centroid of the cell C0 to the centroid of the face, $\mathbf{dr}_1$ is the vector directed from the centroid of the cell C1 to the centroid of the face (see Fig. 2). The quantities ∇p and ∇n are the reconstruction gradients [39] of p and n, respectively, Ψ_p and Ψ_n are limiting factors employed to avoid creating extrema as a part of the reconstruction.

It is also necessary to evaluate the quantity $(\mathbf{F}\cdot\mathbf{A})_f$ at the face to complete the discretization of the drift term. Following the procedure for diffusion terms, this quantity is evaluated as:

$$\begin{aligned}
-\left(\frac{e}{k_BT}\right)(\nabla\phi\cdot\mathbf{A})_f &= -\left(\frac{e}{k_BT}\right)\left[\left(\frac{\phi_1-\phi_0}{ds}\right)\frac{(\mathbf{A}_f\cdot\mathbf{A}_f)}{(\mathbf{A}_f\cdot\hat{\mathbf{e}}_\mathbf{s})}\right.\\
&\quad\left.+\left(\overline{\nabla\phi}\cdot\mathbf{A}_f-(\overline{\nabla\phi}\cdot\hat{\mathbf{e}}_\mathbf{s})\frac{(\mathbf{A}_f\cdot\mathbf{A}_f)}{(\mathbf{A}_f\cdot\hat{\mathbf{e}}_\mathbf{s})}\right)\right]\\
&= -\left(\frac{e}{k_BT}\right)\left(\frac{\phi_1-\phi_0}{ds}\right)\frac{(\mathbf{A}_f\cdot\mathbf{A}_f)}{(\mathbf{A}_f\cdot\hat{\mathbf{e}}_\mathbf{s})}+\mathscr{S}_{\phi f}
\end{aligned} \tag{16}$$

Thus, as with the diffusion fluxes, the quantity $(\mathbf{F}\cdot\mathbf{A})_f$ also consists of primary and secondary terms, the latter going to zero for orthogonal quadrilaterals and hexahedra and equilateral triangles and tetrahedra.

The discretization procedure described here results in a second-order discretization of the drift terms. In contrast, the so-called Scharfetter–Gummel discretization procedure [40] is widely used in the semiconductor device-simulation literature to avoid spatial wiggles and loss of diagonal dominance during iterative solution.

3.4. Source term discretization

Assuming a linear variation of the variables within each cell, the source term in Eq. (9) can be written simply as

$$\int\int\int_V \mathbf{S}\,dV = \mathbf{S}(\mathbf{Q}_0)\Delta\mathscr{V}_0 \tag{17}$$

where $\Delta\mathscr{V}_0$ is the volume of the cell C0 and $\mathbf{Q}_0$ is the solution vector in cell C0. This is consistent with the second-order discretization of the other terms.

3.5. Boundary conditions

In addition to storing $\mathbf{Q}$ at the cell centers, $\mathbf{Q}$ is also stored at all the boundary-face centroids. It is determined from the specified boundary conditions.

3.6. Poisson equation

For the Poisson equation, two types of boundary conditions are considered in this paper, (i) specified ϕ and, (ii) specified normal gradient of ϕ. We consider each in turn.

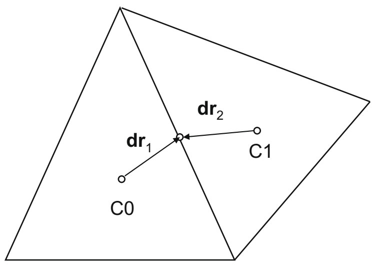

Fig. 2. Geometric details for upwind-biased second-order interpolation.

Given ϕ_b. When $\phi = \phi_b$ is given at a boundary, the diffusion flux in Eq. (13) is written in terms of ϕ_b as:

$$-\left(\frac{e}{k_BT}\right)(\nabla\phi\cdot\mathbf{A})_b = -\left(\frac{e}{k_BT}\right)\left(\frac{\phi_b-\phi_0}{ds}\right)\frac{(\mathbf{A}_b\cdot\mathbf{A}_b)}{(\mathbf{A}_b\cdot\hat{\mathbf{e}}_\mathbf{s})}+\mathscr{S}_{\phi f} \tag{18}$$

where $\mathbf{A}_b$ is the outward-pointing area vector at the boundary, ds is the distance from the centroid of the cell C0 to the boundary-face centroid, and $\hat{\mathbf{e}}_\mathbf{s}$ is the corresponding unit vector.

Given normal gradient. When the normal gradient $\nabla\phi\cdot\mathbf{A}_b$ is given as $q_{\phi,given}$, the given gradient is used directly in the cell balance. The boundary value of ϕ_b is then computed from:

$$q_{\phi,given} = -\left(\frac{e}{k_BT}\right)\left(\frac{\phi_b-\phi_0}{ds}\right)\frac{(\mathbf{A}_b\cdot\mathbf{A}_b)}{(\mathbf{A}_b\cdot\hat{\mathbf{e}}_\mathbf{s})}+\mathscr{S}_{\phi f} \tag{19}$$

3.7. Charge continuity equations

For the n and p equations, again two types of boundary conditions are admitted, (i) specified value of $n = n_b$ and $p = p_b$, or (ii) zero total flux, i.e., $\mathbf{J}_p\cdot\hat{\mathbf{e}}_\mathbf{n} = 0$ and $\mathbf{J}_n\cdot\hat{\mathbf{e}}_\mathbf{n} = 0$.

Given n_b and p_b. Both diffusive and drift (advective) fluxes must be computed at given-n and given-p boundaries. The diffusive flux evaluation is performed as in Eq. (18). For the drift fluxes, the upwinding principle is employed. Thus at a boundary-face b, if $(\mathbf{F}\cdot\mathbf{A})_b > 0$, we have

$$\begin{aligned} p_f &= p_0\\ n_f &= n_b \end{aligned} \tag{20}$$

The advecting field $\mathbf{F}$ is computed from the ϕ field, using Eq. (18) or (19).

Given zero total flux. At given zero-total-flux boundaries, we require

$$(\nabla\psi + \mathbf{V}\psi)_b\cdot\mathbf{A}_b = 0 \tag{21}$$

where $\mathbf{V} = -\mathbf{F}$ if $\psi = p$ and $\mathbf{V} = \mathbf{F}$ if $\psi = n$. The diffusive and drift components of Eq. (21) are discretized using the procedures in Eqs. (18) and (20). The advecting field is computed from the ϕ field, using Eq. (18) or (19). The result is a discrete equation for the boundary values p_b or n_b.

4. Solution method

The discretization procedure described above results in a set of mutually coupled and non-linear algebraic equations. The coupling results from the appearance of the n and p terms in Eq. (1) and the dependence of the drift flux $\mathbf{F}$ on ϕ in Eqs. (2) and (3). We use a Newton–Raphson linearization procedure to formulate an iterative scheme that drives the residual in each cell to zero. The Newton–Raphson procedure results in a set of nominally linear algebraic equations which are solved using an algebraic multigrid scheme. We describe the main components of the solution procedure below.

4.1. Newton–Raphson procedure

The Newton–Raphson iterative scheme may be described by:

$$\left(\frac{\partial\mathbf{R}_i}{\partial\mathbf{Q}}\right)^n(\mathbf{Q}^{n+1}-\mathbf{Q}^n) = \mathbf{R}_i^n \tag{22}$$

Here the superscript n denotes the iteration level. In general, the residual in each cell, $\mathbf{R}_i$, is a function of $\mathbf{Q}$ in the cell C0 and in neighboring cells. In the procedure adopted here in evaluating the Jacobian $\frac{\partial\mathbf{R}_i}{\partial\mathbf{Q}}$, only the dependence on neighboring cells that share a common face with cell C0 is considered. Thus Eq. (22) is written as

$$\left(\sum_f \frac{\partial \mathbf{E}_f}{\partial \mathbf{Q}_i} + \frac{\partial \mathbf{S}_i}{\partial \mathbf{Q}_i}\right)^n (\delta \mathbf{Q}_i) + \sum_f \left(\frac{\partial \mathbf{E}_f}{\partial \mathbf{Q}_{\text{nb}}}\right)^n (\delta \mathbf{Q}_{\text{nb}}) = \mathbf{R}_i^n \tag{23}$$

Here the summations are over the faces f, nb denotes the neighbor cell sharing the face f with cell C0 and $\delta\mathbf{Q} \equiv (\mathbf{Q}^{n+1} - \mathbf{Q}^n)$. Dropping the dependence on other cells implies that only the primary diffusion terms and the first-order contribution to the drift terms are considered implicitly; the secondary gradient term as well as the higher-order drift contributions are treated explicitly. Of course, this approximation in evaluating the Jacobian does not affect the accuracy of the final solution.

The nominally linear algebraic system resulting from the Newton–Raphson procedure is block-unstructured and sparse. The Jacobians $\frac{\partial \mathbf{R}_i}{\partial \mathbf{Q}_i}$ and $\frac{\partial \mathbf{R}_i}{\partial \mathbf{Q}_{\text{nb}}}$ are both 3×3 matrices. The linearization of the drift and diffusion terms contributes to the diagonal entries of the Jacobians $\frac{\partial \mathbf{R}_i}{\partial \mathbf{Q}_i}$ and $\frac{\partial \mathbf{R}_i}{\partial \mathbf{Q}_{\text{nb}}}$. Recalling that the solution vector is $\mathbf{Q} \equiv [\phi\ p\ n]^T$, linearization of the source term in the potential equation (Eq. (1)) results in contributions to the (1,2) and (1,3) entries of $\frac{\partial \mathbf{R}_i}{\partial \mathbf{Q}_i}$. In addition, accounting for the dependency of $\mathbf{F}$ on ϕ contributes to the (2,1) and (3,1) entries of both $\frac{\partial \mathbf{R}_i}{\partial \mathbf{Q}_i}$ and $\frac{\partial \mathbf{R}_i}{\partial \mathbf{Q}_{\text{nb}}}$. There is no direct coupling between the n and p equations in the linearization, hence the other entries in the Jacobians are identically zero. This accounting for the off-diagonal Jacobian entries is the essential ingredient in making the current procedure stable and efficient, especially at high biases. Equations for the boundary values are also included in the linear system with the dependencies of cell residuals on the boundary values being accounted for in the appropriate Jacobians. This aspect is also found to be essential for stability and efficiency.

Eq. (23) represents a linear system of equations in the form

$$\mathbf{Ax} = \mathbf{b} \tag{24}$$

The matrix $\mathbf{A}$ is sparse, with the same fill pattern as that of a matrix resulting from discretization of a scalar equation. However, the solution vector $\mathbf{x}$ and the right-hand side $\mathbf{b}$ consist of a vector of length 3 at each cell instead of a scalar, and the diagonal and off-diagonal entries of $\mathbf{A}$ are 3×3 matrices instead of scalars.

A Gauss–Seidel iterative scheme is used to solve this block-sparse equation set, and is used as the relaxation sweep in a multigrid procedure described in the next section. The Gauss–Seidel scheme operates not on individual scalars, but on the (3×3) blocks that constitute the matrix $\mathbf{A}$. This is achieved by replacing scalar multiplication, division and addition operations used in the conventional Gauss–Seidel procedure with the equivalent matrix and vector operations. Note that in this process, the coupling between the variables in the same cell is resolved directly. This is in contrast to treating Eq. (23) as a scalar system of $3N$ equations. Accounting for this coupling results in significant improvement in the linear solver performance.

4.2. Multigrid method

It is well known that as the size of the linear system increases the convergence of simple relaxation methods such as the Gauss–Seidel method stalls. The use of multigrid methods is very effective in accelerating convergence in such cases. The general procedure has been described in [27–30]. We use an algebraic multigrid technique wherein coarse linear systems are constructed algebraically from the fine-level linear system, as described in [41]. Starting with the finest linear system, n_1 number of relaxation sweeps are performed at each level before the residual is transferred to the next coarse level where it forms the source. This is repeated recursively till the coarsest level, which typically has only a few equations and on which the solution can be obtained easily. Corrections from the coarse level are then propagated to the solution at the next fine level which is then further improved by n_2 relaxation sweeps. This constitutes one V cycle, with each level being visited twice, once during the down leg and once during the up leg. For stiff systems, it is sometimes useful to perform a W cycle which recursively repeats the entire cycle at each level and thus does increasingly more work at coarser levels. In this paper, the so called F cycle is used. Here, a recursive V cycle is performed at each coarse level. The cycle is repeated till the desired level of residual reduction for the finest level equations is obtained. For the results reported in this paper $n_1 = 0$ and $n_2 = 1$ are found to be optimal.

Various methods can be used to create the coarse level linear system. For discretizations of the Laplacian operator, a simple agglomerative procedure is usually sufficient and we have found it satisfactory for the present problem as well. In this method each fine level equation is visited in turn and is assigned to a new group if it has not already been assigned to a coarse group. q of its ungrouped neighbors with the largest coefficient are also assigned to the same group. Each such coarse group constitutes an unknown at the coarse level and the coefficients for the equations for these unknowns are computed by agglomerating the coefficients for the corresponding fine level equations. The source for each coarse level equation is the sum of the residuals from the fine level equations that make up this equation and likewise the solution from this coarse level equation forms the correction for each of the fine level equations. This process is repeated at each stage a coarse level with roughly $1/q$ number of equations is formed. Best performance is usually observed when $q = 2$ and this is used for all results reported in this paper. Unlike geometric multigrid methods, in an algebraic multigrid method, the coarsening is based on the current values of the coefficients and as the non-linear solution evolves, this coarsening adapts to provide the most effective performance at each iteration.

For our vector system of equations, the multigrid procedure is identical to that used for a scalar system. The only additional issue relates to the measure used for determining the neighbors with the largest coefficient because we have a 3×3 matrix instead of a scalar. Various choices are possible, such as using the trace or the highest eigenvalue. In this paper, agglomeration is based on the largest coefficients of the ϕ equation.

At the finest level, for quadrilateral meshes, approximately 45 words per cell are required. This accounts for the storage of a 3×3 array for each cell for the coupling of ϕ, n, and p, with four face neighbors in the Jacobian matrix. For coarsening by a factor $q = 2$, total storage requirements are 90 words per cell.

4.3. Overall solution procedure

To summarize, the overall solution procedure consists of the following steps:

1. Start with guesses for $\mathbf{Q}$ at all cell centers and boundary faces.
2. Using the current values, calculate the residual $\mathbf{R}$ as well as the Jacobian matrix $\mathbf{A}$. If the ratio of the L_2 norm of the current residual and that of the residual at the first iteration is less than the specified tolerance, the solution is converged. For the results reported in this paper, we use a tolerance value of 10^{-6}.
3. If the solution is not converged, solve the linear system $\mathbf{A}\delta\mathbf{Q} = \mathbf{R}$ using the algebraic multigrid method, starting with $\delta\mathbf{Q}^0 = 0$. Because the overall problem is non-linear, we do not need to solve this linear system exactly. For the results presented here, we solve this system only till the norm of the residual at iteration n, i.e., $\mathbf{A}\delta\mathbf{Q}^n - \mathbf{R}$, is less than 0.1 times the L_2 norm of $\mathbf{R}$.
4. Update the current values of $\mathbf{Q}$ with the $\delta\mathbf{Q}$ obtained from the linear solver and repeat from Step 2.

5. Results

In this section, we first present a verification problem to establish the accuracy of the computational procedure. We then apply the multigrid method described above to compute ion transport in a synthetic ion channel [18]. All computations are performed on a DELL Inspiron 9300 laptop computer with a 2 GHz Intel Pentium M processor and 1 GB RAM.

5.1. Verification problem

The problem considered is shown in Fig. 3 and has also been considered in [18]. A rectangular domain of height $H = 0.5$ µm and width $L = 0.05$µm is bounded by symmetry boundaries at $x = 0.0$ and $x = L$, on which the boundary conditions are given by:

$$-\nabla\phi\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$
$$\mathbf{J}_p\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$
$$\mathbf{J}_n\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$

Here $\hat{\mathbf{e}}_{\mathbf{n}}$ is the outward-pointing normal to the boundary. On $y = 0$, the following boundary conditions are assumed:

$$\phi = 0$$
$$n = n_{\mathrm{Cl}}$$
$$p = p_{\mathrm{K}}$$

Here, n_{Cl} and p_{K} are the number densities (number per m^3) of the Cl^- and K^+ ions corresponding to a bulk aqueous solution of potassium chloride (KCl) with a number density n_{KCl}. On $y = H$, a surface charge σ (C/m^2) is applied, and the boundary is assumed impermeable to both K^+ or Cl^-. The corresponding boundary conditions are:

$$\nabla\phi\cdot\hat{\mathbf{e}}_{\mathbf{n}} = -\frac{\sigma}{\epsilon\epsilon_0}$$
$$\mathbf{J}_p\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$
$$\mathbf{J}_n\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$

For a charged surface in a semi-infinite medium containing a solution of KCl, the surface potential developed is given by the Grahame equation [42]:

$$\phi_i = \frac{2k_BT}{e}sinh^{-1}\left(\frac{\sigma}{(8\epsilon\epsilon_0 k_BTn_{KCl})^{\frac{1}{2}}}\right) \tag{25}$$

Three different bulk concentrations, $n_{\mathrm{KCl}} = 10^{-4},\ 10^{-3}$, and 10^{-2} M, corresponding to $p_{\mathrm{K}},\ n_{\mathrm{Cl}} = 6.023\times(10^{22}, 10^{23}, 10^{24})$ m^{-3} are used for a single value of surface charge density $\sigma = -10^{-3}$ C/m^2. A non-uniform structured 3×50 mesh strongly packed towards $y = H$ is used. Computations are performed with an initial guess of $\phi = 0,\ n = n_{\mathrm{Cl}},\ p = p_{\mathrm{K}}$ is used in all cases, with the system being held at $T = 300$ K. The relative permittivity of the medium, ϵ, is 80. Table 1 compares the values of the predicted and exact values of ϕ_i. Reasonable agreement with the exact solution is obtained. For higher values of the surface charge, extremely packed meshes are necessary to capture the steep concentration gradient. Convergence is obtained easily for all cases considered without need to manipulate initial guesses or multigrid parameters.

Table 1
Comparison of computed and exact solutions and multigrid performance for verification problem.

n_{KCl} (M)	ϕ (exact) (V)	ϕ (computed) (V)	ϕ (Daiguji et al. [18]) (V)
10^{-4}	−39.5	−39.58	−39.5
10^{-3}	−13.5	−13.63	−13.7
10^{-2}	−4.97	−4.42	−4.56

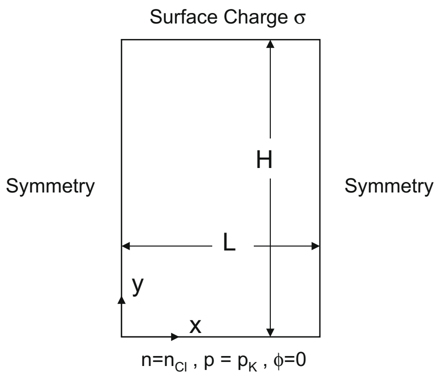

Fig. 3. Computational domain for verification problem.

5.2. Ion transport in nanochannel

Ion transport in a nanochannel connecting two reservoirs is considered next, similar to that considered in [18]. The computational domain is shown in Fig. 4. The reservoirs contain an aqueous solution of KCl with a number density n_{KCl}, corresponding to number densities n_{Cl} and p_{K} for the Cl^- and K^+ ions, respectively. A voltage bias ϕ_b volts is applied between the two lateral boundaries. The walls of the nanochannel carry a charge σ C/m^2. The objective is to compute the $\phi,\ n$ and p distributions in the channel, and to evaluate the performance of the Newton–Raphson procedure and algebraic multigrid scheme for a variety of domain aspect ratios and operating conditions.

The boundary conditions on the different boundaries in Fig. 4 are given below.

Left and right reservoir boundaries

$$\phi = 0 \quad \text{at } x = 0$$
$$\phi = \phi_b \quad \text{at } x = (2L_r + L_c)$$
$$n = n_{\mathrm{Cl}} \quad \text{at } x = 0 \quad \text{and} \quad x = (2L_r + L_c)$$
$$p = p_{\mathrm{K}} \quad \text{at } x = 0 \quad \text{and} \quad x = (2L_r + L_c)$$

Other reservoir boundaries

$$\nabla\phi\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$
$$n = n_{\mathrm{Cl}}$$
$$p = p_{\mathrm{K}}$$

Nanochannel wall

$$\nabla\phi\cdot\hat{\mathbf{e}}_{\mathbf{n}} = -\frac{\sigma}{\epsilon\epsilon_0}$$
$$\mathbf{J}_p\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$
$$\mathbf{J}_n\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$

Symmetry line

$$\nabla\phi\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$
$$\mathbf{J}_p\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$
$$\mathbf{J}_n\cdot\hat{\mathbf{e}}_{\mathbf{n}} = 0$$

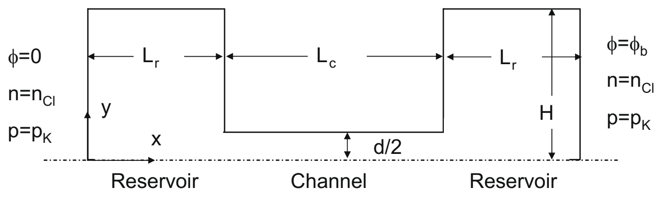

Fig. 4. Nanochannel computational domain.

Table 2
Properties and operating conditions for nanochannel simulation.

Variable	Value
T	300 K
ϵ	80
n_{KCl}, n_{Cl}, p_K	10^{-4} M (6.023×10^{22} m^{-3})
ϕ_b	1–5 V
d	30 nm
L_R	1 μm
L_C	0.5, 5, 50 μm
H	0.5 μm

Here, $\hat{\mathbf{e}}_{\mathbf{n}}$ is the outward-pointing normal to the boundary. Table 2 shows the geometry and operating conditions considered in the simulations.

A non-uniform structured rectangular mesh is used. The mesh size in the two reservoirs is 19×50, while that in the channel is 202×30. This level of mesh resolution is found to yield $\phi(x)$ distributions along the axis of the channel accurate to within 1% compared to finer meshes.

A number of trial computations were also performed using a sequential solution procedure, solving the ϕ, p, and n equations sequentially and iteratively. These computations failed to converge to a solution for any case considered here.

Solution characteristics. Fig. 5 shows the variation of ϕ along the axis of the channel with the applied bias ϕ_b as parameter. Fig. 6 shows the variation of n and p in the reservoirs and channel along the channel axis for different values of the applied bias ϕ_b, which range from 1 to 5 V. It is seen that the values of n and p in the channel satisfy:

$$p - n = -\frac{\sigma}{ed}$$

consistent with [18]; for $\sigma = -10^{-3}$ C/m^2, a value of 4.17×10^{23} m^{-3} (0.69 mM), equal to the theoretical value, is recovered exactly by the finite volume scheme, as expected.

Aspect ratio variation. The performance of the multigrid scheme is tested for three different channel aspect ratios L_C/d, keeping reservoir and mesh size constant. The applied bias is 3 V, $\sigma = -10^{-3}$ C/m^2 and $n_{KCl} = 10^{-4}$ M. The initial condition is $\phi = 0$, $n = n_{Cl}$ and $p = p_K$ in the reservoirs and the channel. The entire bias is applied at once, instead of stepping up gradually as in typical Gummel iteration procedures. Table 3 shows the number of iterations to convergence, and the required CPU time in seconds. We see from Table 3 that the algebraic multigrid method performs well for all aspect ratios considered. The number of iterations to convergence is approximately the same for all aspect ratios, and the CPU times also do not show a strong dependence on aspect ratio.

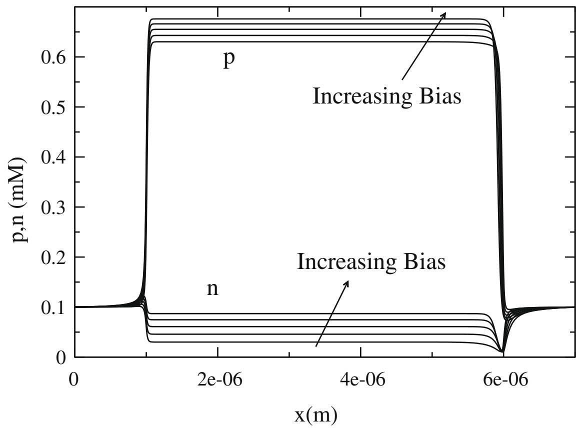

Fig. 6. Variation of p and n along centerline with ϕ_b as parameter.

Table 3
Multigrid performance for different channel aspect ratios.

Aspect ratio L_C/d	Iterations	CPU (s)
16.67	7	3.35
166.67	7	3.21
1666.67	8	3.24

Variation in applied bias. Next, the influence of applied bias on multigrid performance is tested for applied biases in the range 1–5 V for $\sigma = -10^{-3}$ C/m^2, $n_{KCl} = 10^{-4}$ M and aspect ratio of 166.67. Again, the same initial guess is used as before, and the entire bias is applied at once, instead of stepping up gradually. The results are shown in Table 4. We see that the performance of the method is good across the range of applied voltage. The number of iterations to convergence and the CPU time are both relatively insensitive to the applied voltage for the range of parameters considered.

Variation in surface charge density. The performance of the numerical procedure for different surface charge densities is shown in Table 5. The applied bias is held at 3 V with $n_{KCl} = 10^{-4}$ M and an aspect ratio of 166.67. The starting guess is the same as before, and the desired value of σ is applied at once without stepping up. As surface charge density increases, the problem becomes increasingly non-linear, with positive ions being drawn strongly into the channel and negative ions being repelled

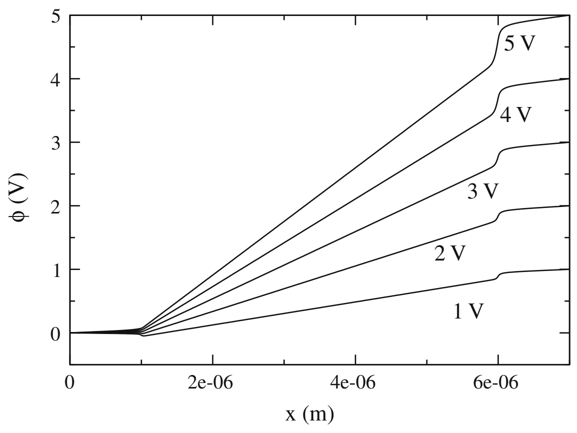

Fig. 5. Variation of ϕ along centerline with ϕ_b as parameter.

Table 4
Multigrid performance for different applied biases.

Applied bias (V)	Iterations	CPU (s)
1	7	3.1
2	7	3.08
3	7	3.21
4	7	3.27
5	7	3.53

Table 5
Multigrid performance for different σ.

σ C/m^2	Iterations	CPU (s)
-10^{-3}	7	3.21
-10^{-2}	9	5.39
-10^{-1}	21	9.11

Table 6
Multigrid performance for different n_{KCl}.

n_{KCl} (M)	Iterations	CPU (s)
10^{-5}	9	7.63
10^{-4}	7	3.21
10^{-3}	6	2.28
10^{-2}	6	2.08

Table 7
Multigrid performance for different mesh sizes.

Mesh size (cells)	Iterations	CPU (s)
8000	7	3.21
32,000	7	25.34
128,000	7	201.83

from the channel in order to achieve neutrality. Therefore, as expected, the number of iterations to convergence increases with increasing σ, with a commensurate increase in CPU time. No convergence difficulties were encountered for the range of σ values tested in the table.

Variation in KCl concentration. Table 6 shows how the numerical method performs with variation in KCl concentration in the reservoirs. The applied bias is held at 3 V and $\sigma = 10^{-3}$ C/m^2 is used, with an aspect ratio of 166.67. The same starting guess is used as before, and the desired value of n_{KCl} is used directly. Convergence is most difficult for the lowest values of n_{KCl} as the high value of the surface charge draws positive ions into the channel and strongly depletes the reservoir. Thus, the number of iterations to convergence, and commensurately, the CPU time, increase as n_{KCl} decreases. Nevertheless, convergence is obtained easily for the range of n_{KCl} investigated here.

Dependence on mesh size. The dependence of multigrid performance on mesh size is evaluated next. An applied bias of 3 V, $\sigma = -10^{-3}$ C/m^2 and $n_{KCl} = 10^{-4}$ M are used, with the same initial guess as before. Table 7 shows the number of iterations to convergence and the CPU seconds required. The number of iterations are nearly constant with grid size. The CPU time is seen to scale superlinearly with grid size, approximately as $N\log(2N)$, because the computational effort required to solve the linear system for each outer iteration increases superlinearly. This is primarily due to the increased number of multigrid levels for finer meshes, and the attendant extra cost in performing an F cycle.

6. Conclusions

A Newton–Raphson iteration procedure coupled to an algebraic multigrid method has been developed for the solution of the Poisson–Nernst–Planck equations. The underlying discretization is based on a cell-centered finite volume method admitting unstructured solution-adaptive polyhedra. The solution technique accounts for the non-linear coupling between the Poisson and charge continuity equations through the space charge and drift terms, and also through the boundary conditions. A block Gauss–Seidel scheme is used to solve the resulting block-unstructured sparse equation set, with an algebraic multigrid scheme being used for solution acceleration. The method is applied to ion transport in a nanochannel geometry for operating parameters ranging over several orders of magnitude and shown to perform well. Convergence is robust for all cases considered here and memory and CPU requirements are relatively modest. In contrast, tests performed using a sequential solution of the Poisson and charge continuity equations failed to converge even on far more limited parameter spaces. Therefore, the overall accuracy and performance of the method make it a suitable vehicle for the simulation of a variety of transport problems in emerging microsystems.

Acknowledgments

J. Murthy wishes to acknowledge support from the NSF-funded Network for Computational Nanotechnology (NCN) and nanoHub (http://www.nanohub.org) at Purdue University.

References

[1] B. Lou, Y. Zhou, G. Huber, S. Bond, M. Holst, J. McCammon, Electro-diffusion: a continuum modeling framework for bio-molecular systems with realistic spatio-temporal resolution, J. Chem. Phys. 127 (2007) 135102.
[2] Y. Zhou, B. Lou, G. Huber, M. Holst, J. McCammon, Continuum simulations of acetylcholine consumption by acetylcholisnesterase: a Poisson–Nernst–Planck approach, J. Phys. Chem. 112 (2008) 270–275.
[3] C. Dragos, Z. Siwy, Poisson–Nernst–Planck model of ion current rectification through a nanofluidic diode, Phys. Rev. E 76 (2007) 041202.1.
[4] B. Iverson, L. Cremaschi, S. Garimella, Effect of discrete-electrode configuration on traveling wave electro-hydrodynamic pumping, Microfluid. Nanofluid. (2008), doi: 10.1007/s10404-008-0317-1.
[5] M. Rickard, D. Dunn-Rankin, Numerical simulation of a tubular ion-driven wind generator, J. Electrostat. 65 (2007) 646–654.
[6] S. Selberherr, Analysis and Simulation of Semiconductor Devices, Springer-Verlag, New York, 1984.
[7] R. Eisenberg, Ionic channels in biological membranes: natural nanotubes, Acc. Chem. Res. 31 (1998) 117–123.
[8] B. Roux, M. Karplus, Ion transport in gramicidin channel: free energy of the solvated ion in a model membrane, J. Am. Chem. Soc. 115 (1993) 3250–3260.
[9] R. Elber, D. Chen, D. Rojewska, R. Eisenberg, Sodium in gramicidin:an example of permion, Biophys. J. 68 (1993) 906–924.
[10] T.V.D. Straaten, J. Tang, U. Ravaioli, R. Eisenberg, N. Aluru, Simulating ion permeation through ompF porin ion channel using three-dimensional drift–diffusion theory, J. Comput. Electron. 2 (2003) 29–47.
[11] V. Barcilon, D. Chen, R. Eisenberg, Ion flow through narrow membrane channels. II, SIAM J. Appl. Math. 52 (1992) 1405–1425.
[12] M. Kurnikova, R. Coalson, P. Graf, A. Nitzan, A lattice relaxation algorithm for three-dimensional Poisson–Nernst–Planck theory for application to ion transport through gramicidin A channel, Biophys. J. 76 (1999) 642–656.
[13] D. Gillespie, W. Nonner, R. Eisenberg, Coupling Poisson–Nernst–Planck and density functional theory to calculate ion flux, J. Phys.: Condens. Mat. 14 (2002) 12129–12145.
[14] Z. Yang, T.V.D. Straaten, U. Ravaioli, L. Yan, A coupled 3D PNP/ECP model for ion transport in biological ion channels, J. Comput. Electron. 4 (2005) 167–170.
[15] B. Corry, S. Kuyucak, S. Chung, Test of Poisson–Nernst–Planck theory in ion channels, J. Gen. Phys. 114 (1999) 597–599.
[16] S. Chung, M. Hoyles, T. Allen, S. Kuyacak, Study of ion currents across a model membrane channel using Brownian dynamics, Biophys. J. 75 (1998) 793–809.
[17] H. Daiguji, P. Yang, A. Szeri, A. Majumdar, Electrochemomechanical energy conversion in nanofluidic channels, Nanoletters 4 (12) (2004) 2315–2321.
[18] H. Daiguji, P. Yang, A. Majumdar, Ion transport in nanofluidic channels, Nanoletters 4 (1) (2004) 2137–2142.
[19] R. Karnik, R. Fan, M. Yue, D. Li, P. Yang, A. Majumdar, Electrostatic control of ions and molecules in nanofluidic transistors, Nanoletters 5 (5) (2005) 943–948.
[20] J. Slotboom, Iterative scheme for 1- and 2-dimensional DC transistor simulation, Electron. Lett. 5 (1969) 677–678.
[21] M. Saraniti, A. Rein, G. Zandler, P. Vogl, P. Lugli, An efficient multigrid Poisson solver for device simulations, IEEE Trans. Comput. Aid. Des. Integr. Circ. Syst. 15 (2) (2007) 141–150.
[22] C. Millar, A. Asenov, J. Watling, Excessive over-relaxation method for multigrid Poisson solvers, J. Comput. Electron. 1 (3) (2002) 341–345.
[23] J. Meza, R. Tuminaro, A multigrid preconditioner for the semiconductor equations, SIAM J. Sci. Comput. 17 (1) (1996) 118–132.
[24] J. Molenaar, Multigrid methods for semiconductor device simulation, Technical Report, CWI Tract 100, Center for Math. Comput. Sci., Amsterdam, The Netherlands, November 1993.
[25] T. Clees, AMG strategies for PDE systems with applications to industrial semiconductor simulation, Ph.D. thesis, University of Cologne, Cologne, Germany, 2005.
[26] Available from: <http://wwwtcad.stanford.edu/prophet>.
[27] W. Hackbush, U. Trottenberg, Multigrid Methods, Springer-Verlag, Berlin, 1982.
[28] K. Steuben, U. Trottenberg, Multigrid methods: fundamental algorithms model, problem analysis and applications, in: W. Hackbush, U. Trottenberg (Eds.), Multigrid Methods, Lecture Notes in Mathematics, Springer-Verlag, Berlin, 1982, pp. 1–176.
[29] A. Brandt, 1984 guide with applications to fluid dynamics, Technical Report, The Weizmann Institute, Rehovat, Israel, 1984.
[30] U. Trottenberg, A. Schuller, Multigrid, Academic Press, Orlando, Florida, 2000.
[31] H. Gummel, A self-consistent iterative scheme for one-dimensional steady state transistor calculations, IEEE Trans. Electr. Devices 11 (1964) 455–465.

[32] J.C. Meza, J. Grcar, DANCIR: a three-dimensional semiconductor device simulator, Technical Report SAND89-8266, Sandia National Laboratories, 1990.
[33] W. Shyy, M.-H. Chen, C.-S. Sun, Pressure-based multigrid algorithm for flow at all speeds, AIAA J. 30 (11) (1992) 2660–2669.
[34] S. Vanka, Performance of a multigrid calculation procedure in three-dimensional sudden expansion flows, Int. J. Numer. Meth. Fluids 6 (1986) 459–477.
[35] D. Mavriplis, V. Venkatakrishnan, A 3D agglomeration multigrid solver for the Reynolds-averaged Navier–Stokes equations on unstructured meshes, in: 33rd Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 1995.
[36] N. Lambropoulos, D. Koubogiannis, K. Giannakoglou, Acceleration of a Navier–Stokes equation solver for unstructured grids using agglomeration multigrid and parallel processing, Comput. Meth. Appl. Mech. Eng. 193 (2004) 781–803.
[37] P. Sathyamurthy, S. Patankar, Block-correction-based multigrid method for fluid flow problems, Numer. Heat Transfer B: Fundam. 25 (4) (1994) 375–394.
[38] S. Mathur, J. Murthy, A pressure-based method for unstructured meshes, Numer. Heat Transfer B: Fundam. 32 (2) (1997) 195–216.
[39] T.J. Barth, D.C. Jespersen, The design and application of upwind schemes on unstructured meshes, AIAA 89-0366, 1989.
[40] D. Scharfetter, H. Gummel, Large-signal analysis of a silicon read diode oscillator, IEEE Trans. Electr. Devices ED16 (1969) 64–77.
[41] S. Mathur, J. Murthy, Unstructured finite volume methods for multi-mode heat transfer, in: W. Minkowyz, E.M. Sparrow (Eds.), Advances in Numerical Heat Transfer, vol. 2, Taylor and Francis, 2001, pp. 37–67.
[42] D. Grahame, Diffuse double layer theory for electrolytes of unsymmetrical valence types, J. Chem. Phys. 21 (6) (1953) 1054–1060.