Journal of Scientific Computing, Vol. 16, No. 4, December 2001 (© 2002)

Two- and Three-Dimensional Poisson–Nernst–Planck Simulations of Current Flow Through Gramicidin A

Uwe Hollerbach,[1,2] Duan-Pin Chen,[1] and Robert S. Eisenberg[1]

Received December 20, 2000; accepted (in revised form) January 26, 2001

We simulate sodium chloride currents through the gramicidin A channel using the spectral element method to solve the three-dimensional Poisson–Nernst–Planck (PNP) equations. Using the spectral element method, we are able to simulate the entire channel, plus large enough portions of the lipid bilayer and baths to ensure that all boundary conditions are realistic. In these simulations, we rely on the 3D charge distribution of the gramicidin molecule plus diffusion coefficients and dielectric coefficients. Our main results, which match the experimental data, are current-voltage (IV) curves for gramicidin at various concentrations of Na^+Cl^- in the surrounding baths. We give a detailed description of the numerical algorithms used to solve the PNP equations, and we present various sensitivity analyses which we have performed to determine which parameters of the model most affect the IV curves.

KEY WORDS: Gramicidin A; IV curves; simulation; spectral element method; finite element method.

1. INTRODUCTION

This paper describes the simulation of sodium chloride currents through the gramicidin A channel using the three-dimensional Poisson–Nernst–Planck (PNP) equations, a self-consistent mean-field approximate description of ionic motion. Two forces act on an ion in this theory: the electrostatic force produced by all other charges in the system, governed by the Poisson equation, and a diffusion "force" caused by variations in the concentration of ions, governed by the Nernst–Planck equations, which are an expression of the conservation of mass. The PNP equations are solved

[1] Department of Molecular Biophysics and Physiology, Rush Medical College, Chicago, Illinois 60612.
[2] PO Box 591, Franklin, Massachusetts 02038. E-mail: uh@alumni.caltech.edu

0885-7474/01/1200-0373/0 © 2002 Plenum Publishing Corporation

using the spectral element method, which is a particular kind of finite element method. This paper describes the spectral element method in some detail, using the PNP description of the gramicidin channel as a worked example. This problem has previously been studied by (Kurnikova *et al.*, 1999), who used a uniform grid to solve the PNP equations inside the channel, but could not reach far into the baths, and therefore used an artificial boundary condition at the channel entrances. In contrast, with the spectral element method it is possible to place the elements in such a way that there is high resolution in those places where it is needed, and low resolution elsewhere. As a consequence, it is thus possible to include realistic portions of the baths in our simulations and thereby use physically more-correct boundary conditions. In addition, (Kurnikova *et al.*, 1999) set $D_{Na^+} = D_{Cl^-}$ to an arbitrary value, then scaled their results to match one experimental point, and finally compared their scaled results against all other experimental points. In our work, there is no fitting of the 3D model to the experimental data; indeed, the model which we describe here has (almost) no internal non-physical parameters which might be adjusted to improve the fit (the placement of spectral element boundaries could in principle be used for this purpose). We rely solely on the 3D charge distribution of the gramicidin molecule, three dielectric coefficients, and four diffusion coefficients, to match the data.

Our main results are current-voltage (IV) curves for gramicidin at various concentrations of Na^+Cl^- in the surrounding baths. These currents are produced by the distribution of electrical and chemical potential in the channel, and those profiles are shown in a few cases to illustrate the importance of shielding. Various sensitivity analyses have also been performed to determine which parameters of the model most affect the IV curves.

2. GENERAL MATHEMATICAL METHODS AND EQUATIONS

2.1. Massaging of PNP Equations

We begin by showing all the steps that are required to convert the PNP equations from their original forms into the versions used in solving them numerically. The target form for the equations is the Helmholtz equation, $\nabla\cdot(P\nabla U) - QU = -S$, which is to be solved for U in some domain Ω. The details of the solution procedure, as well as what boundary conditions are allowed and how they are applied, are described further down. This so-called "self-adjoint" target form is desirable because it has been studied a great deal; indeed, much of the mathematical physics (Courant and Hilbert, 1989) and finite element (Strang and Fix, 1973;

Zienciewicz, 1971) literature deals with the analytical and numerical solution of such equations.

2.1.1. *Poisson Equation*

The starting point is the Poisson equation (Jackson, 1975), which specifies the electric potential given the charge density:

$$\nabla \cdot (\epsilon_0 \epsilon_{\text{rel}} \nabla \phi) = -e \sum_{\text{species}} z_i C_i \tag{1}$$

Here ϵ_0 is the permeability of free space, ϵ_{rel} is the relative dielectric coefficient and varies from about 1 to 80, ϕ is the potential in volts, z_i is the charge state of an ion (typically ± 1), e is the charge of the proton, and C_i is the concentration of an ion species, measured in meter^{-3}.

Potentials are made non-dimensional by the thermal voltage kT/e, lengths by $d = 1$ Å, and concentrations by $C_{\text{base}} \equiv \epsilon_0 kT/(e^2 d^2)$; then the nondimensional Poisson equation reads

$$\nabla \cdot (\epsilon_{\text{rel}} \nabla \phi^*) = -\sum_{\text{species}} z_i C_i^* \tag{2}$$

where now the derivative operators are non-dimensional, ϕ^* is the non-dimensional potential, and C_i^* is the non-dimensional concentration of species i. The equation is already in the desired Helmholtz target form, so no further modifications are necessary for the Poisson equation at this stage.

2.1.2. *Nernst–Planck Equations*

These equations, one per ionic species, relate the current density produced by each species to two driving forces: the electric field and the concentration gradient of that species. Mean-field equations of this sort are the starting point of much analysis in physics because they arise so directly from conservation laws, as described in many texts (Nicholson, 1983; Selberherr, 1984). Higher-resolution models are desirable, of course, particularly when dealing with systems as small as ionic channels, but ensuring that their results actually satisfy macroscopic conservation laws and constitutive relations (e.g., Ohm's law and Fick's law in appropriate domains, and the Poisson equation, in general) is a non-trivial task, necessary but not always performed.

$$J_{\text{current}} = -z^2 e \mu C \nabla \phi - zeD \nabla C \tag{3}$$

This is the fully-dimensional drift-diffusion equation, where μ is the mobility, D is the diffusion coefficient, and J_{current} is the electric current density. Note that we use the "conventional mobility," defined as charge times velocity divided by force (Selberherr, 1984) [a good discussion of various definitions of mobility can be found in chapter 4 of (Bockris and Reddy, 1998)]. The Einstein relation, which is $D = \mu kT/e$ using this definition of mobility, is assumed to apply. The equation is then partially non-dimensionalized by factoring the thermal voltage kT/e out of the potential. Then

$$\begin{aligned} J_{\text{current}} &= -z^2 eDC\nabla\phi^* - zeD\nabla C \\ &= -zeD(\nabla C + zC\nabla\phi^*) \end{aligned} \tag{4}$$

The above equation describes the electrical current density; by dividing out the charge of the species ($= ze$), one obtains the expression for the particle flux density J_{flux}, which is what is actually used in the computation. The non-dimensionalization of the equation is completed by defining D_{base} as the unit of the diffusion coefficient, and $J_{\text{base}} \equiv D_{\text{base}} C_{\text{base}}/d$ as the unit of particle flux density. Then the final, non-dimensional, form of the equation is

$$J^*_{\text{flux}} = -D^*(\nabla C^* + zC^*\nabla\phi^*) \tag{5}$$

Since all quantities are now nondimensional, and will be kept in that form, the superscript * will be dropped from here on.

This equation is used in the steady state, where there is no net inflow or outflow of ions at any point; equivalently, the divergence of J_{flux} is required to be zero everywhere:

$$\nabla \cdot J_{\text{flux}} = -\nabla \cdot (D(\nabla C + zC\nabla\phi)) = 0 \tag{6}$$

This is the equation which actually gets solved for C. However, it is not yet in the target form. This may be achieved by including the integrating factor $e^{z\phi}$; once this is done, the equation can be written as

$$\nabla \cdot (De^{-z\phi}\nabla(Ce^{z\phi})) = 0 \tag{7}$$

Using this integrating factor can lead to difficulties when potentials become large because of the large dynamic range of the unknown variable $Ce^{z\phi}$. Fortunately, potentials are usually relatively small in biological applications, so the use of the integrating factor does not usually cause problems.

2.1.3. Boundary Conditions

The gramicidin channel is effectively a very small hole in a membrane separating two baths. There is one electrode in each bath, and the potential on each electrode is maintained by external electronics. In the simulation, this setup is modeled by applying Dirichlet boundary conditions for the Poisson equation along "electrode" portions of the boundary, namely in the bath far from both the channel and the lipid bilayer. Along the remainder of the boundary, no boundary conditions are explicitly specified, which implicitly results in conditions corresponding to a zero normal electric field. Although the computational boundaries at which the boundary conditions are applied are far closer to the channel than the electrodes are in the physical system, this does not cause any significant difficulties, because the resistance of the bath is very small compared with the resistance of the channel; thus the voltage drop and concentration gradient in the bath are tiny compared with the voltage drop and concentration gradient in the channel, and the potential and concentration in the physical system at the location of each computational electrode are very close to the potential and concentration on the appropriate physical electrode.

For the Nernst–Planck equations, we take advantage of the relatively huge size of the baths on either side of the membrane compared to the current (both electrical and fluid) that flows through the channel. Because the baths are effectively infinite reservoirs, the concentrations on either side of the channel are unchanged by any flows through the channel; thus Dirichlet boundary conditions are also appropriate for the Nernst–Planck equations at the "electrode" portions of the computational boundary. Along the other boundaries, we again impose zero-flux boundary conditions. For the Nernst–Planck equations, the boundary also includes the surface of the lipid bilayer, as well as the wall of the channel, because ions do not penetrate these surfaces at appreciable rates. Thus zero-flux boundary conditions are also applied at these surfaces.

All of these boundary conditions correspond to quantities that are easily measured and that are physically meaningful; as will be shown, they are also easy to apply numerically.

2.2. Gummel Iteration

Before we present the detailed description of the solution procedure for an individual Helmholtz equation, it is worthwhile spending a moment on the overall solution scheme, because there is an apparent paradox which needs to be resolved: the Poisson equation solves for the electric potential ϕ if the concentrations C_i are known, and the Nernst–Planck equations solve

for the concentrations if the potential is known. This would appear to be a circular set of dependencies, making it difficult to solve for any of the quantities. This turns out not to be the case if the Poisson equation is rewritten slightly, by subtracting the term $(\sum_{\text{species}} z_i^2 C_i)\,\phi$ from both sides:

$$\nabla \cdot (\epsilon_{\text{rel}} \nabla \phi) - \left(\sum_{\text{species}} z_i^2 C_i \right) \phi = - \sum_{\text{species}} z_i C_i - \left(\sum_{\text{species}} z_i^2 C_i \right) \phi \tag{8}$$

This modification, which preserves the Helmholtz form of the equation, is a crucial step in what is known in the semiconductor literature as the Gummel iteration. The exact magnitude of the additional term is important for convergence; for a much more detailed treatment, see (Selberherr, 1984; Gummel, 1964; Jacoboni and Lugli, 1989; Jerome, 1996).

The actual work of the Gummel iteration then goes as follows: instead of treating all occurrences of ϕ equally, the ϕ on the right side of the Poisson equation is treated as a known quantity. The iteration begins by assuming an arbitrary value for ϕ—say, $\phi = 0$ everywhere. (Of course, the better the initial guess, the faster the iteration converges.) This initial value of ϕ is used in the Nernst–Planck equations to solve for approximate concentrations C_i; then the old value of ϕ and the just-computed approximate concentrations are used in the modified Poisson equation to solve for an improved value for ϕ. The loop is closed by using this improved value for ϕ in the Nernst–Planck equations again, and so on. This iteration scheme proves to be quite robust. In the authors' experience, it converges rather reliably to a set of self-consistent solutions of the individual equations, taking anywhere from 5 to 30 iterations to achieve a relative accuracy of 10^{-9}.

2.3. Solution of Helmholtz Equations

The spectral element method (Patera, 1984; Ghaddar, Karniadakis, and Patera, 1986) is a particular type of finite element method with two special characteristics. First, the basis functions are orthogonal polynomials; here, Legendre polynomials were used, but other families of orthogonal polynomials can also be utilized. Second, the philosophy for improving an under-resolved solution is different from that used in the standard finite element method. In the standard finite element method, if a solution is computed which has too large an error, some or all of the elements are subdivided and the solution is recomputed on the finer mesh. In the spectral element method the order of the polynomials used inside each element is increased instead, without changing the number or shape of elements. This also leads to a finer mesh, but the convergence behavior is different

for the two approaches: with the spectral element approach, it is possible to attain an exponential convergence to the true solution as the order of the polynomials is increased, provided that the true solution is smooth enough. If there are discontinuities in the derivatives, convergence to the true solution is significantly improved by placing element boundaries along the discontinuities. The standard finite element method, on the other hand, is better if the solution is not particularly smooth.

2.3.1. Conversion of a Helmholtz Equation into an Integral Form

Consider again the general equation

$$\nabla \cdot (P \nabla U) - QU = -S \qquad \text{in some domain } \Omega \text{ with boundary } \Gamma \tag{9}$$

Here P is required to be non-negative and real; Q, S, and U may in general be complex, although in the present work only real functions are used. The sign of Q affects what kinds of solution procedures can be used; this point will be discussed in more detail later. The following types of boundary conditions are allowed: $\partial U/\partial v + \alpha U = \beta$ (here $\partial/\partial v$ denotes the outward normal derivative) along Γ_1, and $U = U_0$ along Γ_2, where Γ_1 and Γ_2 are distinct parts of the boundary Γ, and α, β, and U_0 are known functions. An equivalent integral form of the problem is obtained by multiplying the differential equation by some suitable test function W and then integrating by parts over Ω:

$$\int_\Omega (P \nabla W \, \nabla U + QWU) \, dV - \int_\Gamma WP \nabla U \cdot dS = \int_\Omega WS \, dV \tag{10}$$

The natural boundary conditions $\partial U/\partial v + \alpha U = \beta$ are incorporated by multiplying them by another test function w, integrating over Γ_1:

$$\int_{\Gamma_1} \left(w \frac{\partial U}{\partial v} + w\alpha U - w\beta \right) |dS| = 0 \tag{11}$$

and adding the result to the first integral over Ω. The boundary test function w is set equal to $-WP$ on Γ_1, so that the terms $WP\nabla U$ and $w \, \partial U/\partial v$ cancel each other along Γ_1 ($\nabla U \cdot dS = \partial U/\partial v \, |dS|$). The second type of boundary condition, $U = U_0$, is imposed along Γ_2, by restricting the solution space from which the U are drawn, and the corresponding term in the integral equation is removed by requiring $W = 0$ along Γ_2. The final form of the equation is

$$\int_\Omega P \nabla W \, \nabla U + QWU \, dV + \int_{\Gamma_1} WP\alpha U \, dS = \int_\Omega WS \, dV + \int_{\Gamma_1} WP\beta \, dS \tag{12}$$

In this equation, only first derivatives of the unknown U appear, whereas in the original differential formulation second derivatives appear. This means that some solutions of the integral equation may not satisfy the differential equation: specifically, some solutions of the integral equation may be continuous but may not have a continuous derivative. These solutions cause no harm in the integral equation, but in the differential equation they would give rise to delta-functions that do not satisfy the equation in the classical sense. For this reason the integral formulation is also called the weak formulation of the problem.

2.3.2. *Discretization of the Integral Equation*

The integral equation is discretized by splitting the domain Ω into a number of distorted rectangular (in 2D) or brick-shaped (in 3D) elements. In each of these elements, the unknown U and the test function W are represented as tensor products of high-order polynomials: specifically, any function $F(\mathbf{x})$ is approximated as

$$F(r, s, t) = \sum_{i=0}^{N_r} \sum_{j=0}^{N_s} \sum_{k=0}^{N_t} F_{ijk} h_i(r)\, h_j(s)\, h_k(t) \tag{13}$$

where F_{ijk} is the value of $F(\mathbf{x})$ at the (i, j, k)-th node and h_l is the 1D Lagrangian interpolant polynomial associated with the lth node: it has the property that if r_i is the position of the ith node, then $h_l(r_i) = \delta_{il}$, the Kronecker delta. N_l is the order of the polynomials in the l-direction. In addition, it is necessary to specify a translation between the element-local coordinates (r, s, t) and the global coordinates $(\mathbf{x})$ in which the original problem is specified. This translation can be performed quite conveniently by expressing $(\mathbf{x})$ itself in the above functional form:

$$\mathbf{x}(r, s, t) = \sum_{i=0}^{N_r} \sum_{j=0}^{N_s} \sum_{k=0}^{N_t} \mathbf{x}_{ijk} h_i(r)\, h_j(s)\, h_k(t) \tag{14}$$

Derivatives are also easily evaluated:

$$\frac{\partial F(r, s, t)}{\partial r} = \sum_{i=0}^{N_r} \sum_{j=0}^{N_s} \sum_{k=0}^{N_t} F_{ijk} h_i'(r)\, h_j(s)\, h_k(t)$$

$$\equiv \sum_{i=0}^{N_r} \sum_{j=0}^{N_s} \sum_{k=0}^{N_t} F'_{ijk} h_i(r)\, h_j(s)\, h_k(t) \tag{15}$$

To find the values of the coefficients F'_{ijk}, it is simply necessary to evaluate the derivative at a particular node, say *lmn*:

$$\sum_{i=0}^{N_r} \sum_{j=0}^{N_s} \sum_{k=0}^{N_t} F'_{ijk} h_i(r_l)\, h_j(s_m)\, h_k(t_n) = \sum_{i=0}^{N_r} \sum_{j=0}^{N_s} \sum_{k=0}^{N_t} F_{ijk} h'_i(r_l)\, h_j(s_m)\, h_k(t_n) \tag{16}$$

Because the h_l are Kronecker deltas, the sums over j and k vanish, as does the sum over i on the left side:

$$F'_{lmn} = \sum_{i=0}^{N_r} F_{imn} h'_i(r_l) \equiv \sum_{i=0}^{N_r} F_{imn} D_{il} \tag{17}$$

This equation defines the 1D derivative matrix $D_{ij} = h'_i(r_j)$. (There is, of course, a separate derivative matrix for each polynomial order and thus for each direction.)

If the above functional form for U and for W is substituted into the integral equation, the result is a discrete set of linear equations for U_{ijk}, the unknown values of U at the nodal points, given a particular set of test functions W. One simple set of test functions is the set of Lagrangian interpolants themselves: $W_{ijk} = h_i(r)\, h_j(s)\, h_k(t)$. It is now only necessary to choose the set of basis functions and the integration method in order to be able to compute the integrals and so find the matrix equation which will determine the $U_{i,j,k}$. In the present work, Legendre polynomials have been used as the underlying basis functions, and Gauss–Lobatto integration to compute the integrals; this choice is particularly convenient because the integration points of the Gauss–Lobatto integration are the same as the element nodes. With this choice, this representation converges exponentially (faster than any power of $1/N_l$) as N_l increases, provided that the solution to the Helmholtz equation is continuous and the elements are chosen properly.

2.3.3. Computation of Matrix Elements and Source Terms

Once the basis functions and integration method have been chosen, the integrals must be evaluated in order to find explicit formulas for all matrix elements. These will first be computed for individual elements; putting the elements together into a global set of equations will be addressed later.

2.3.4. *Matrix Elements due to* $\nabla \cdot (P \nabla U)$

Consider first the evaluation of the term $\int P \nabla W \, \nabla U \, dV$:

$$\nabla U = \begin{pmatrix} \partial U/\partial x \\ \partial U/\partial y \\ \partial U/\partial z \end{pmatrix} = \begin{pmatrix} \partial r/\partial x & \partial s/\partial x & \partial t/\partial x \\ \partial r/\partial y & \partial s/\partial y & \partial t/\partial y \\ \partial r/\partial z & \partial s/\partial z & \partial t/\partial z \end{pmatrix} \begin{pmatrix} \partial U/\partial r \\ \partial U/\partial s \\ \partial U/\partial t \end{pmatrix} \tag{18}$$

The evaluation of the derivatives $\partial U/\partial(r, s, t)$ was described above; the "inverse" derivatives $\partial(r, s, t)/\partial(x, y, z)$ may be computed in the following manner from the geometry transformation between local and global coordinates: (x, y, z) is known as a function of (r, s, t), so that $\partial(x, y, z)/\partial(r, s, t)$ is easily evaluated; then, because they are inverse transformations of each other:

$$\begin{pmatrix} \partial r/\partial x & \partial r/\partial y & \partial r/\partial z \\ \partial s/\partial x & \partial s/\partial y & \partial s/\partial z \\ \partial t/\partial x & \partial t/\partial y & \partial t/\partial z \end{pmatrix} = \begin{pmatrix} \partial x/\partial r & \partial x/\partial s & \partial x/\partial t \\ \partial y/\partial r & \partial y/\partial s & \partial y/\partial t \\ \partial z/\partial r & \partial z/\partial s & \partial z/\partial t \end{pmatrix}^{-1} \tag{19}$$

At this point, ∇W and ∇U can be evaluated using only quantities known at the element nodes, and it is now necessary to integrate over the volume of the element. Because the problem is specified in terms of the global coordinates (x, y, z), but the integral is most conveniently evaluated in terms of the local coordinates (r, s, t), the Jacobian of the transformation must be included in the integral. The Jacobian is just the determinant of the matrix $\partial(x, y, z)/\partial(r, s, t)$. By using Gaussian quadratures to evaluate integrals, the integral can be approximated as a weighted sum:

$$\begin{aligned} \int P \nabla W \, \nabla U \, dV = \sum_{i,l,o=0}^{N_r} \sum_{j,m,p=0}^{N_s} \sum_{k,n,q=0}^{N_t} W_{lmn}^T \begin{pmatrix} D_{lo}^T \delta_{mp} \delta_{nq} \\ \delta_{lo} D_{mp}^T \delta_{nq} \\ \delta_{lo} \delta_{mp} D_{nq}^T \end{pmatrix} \\ \cdot \begin{pmatrix} G_{opq}^{1,1} & G_{opq}^{1,2} & G_{opq}^{1,3} \\ G_{opq}^{2,1} & G_{opq}^{2,2} & G_{opq}^{2,3} \\ G_{opq}^{3,1} & G_{opq}^{3,2} & G_{opq}^{3,3} \end{pmatrix} \begin{pmatrix} D_{io} \delta_{jp} \delta_{kq} \\ \delta_{io} D_{jp} \delta_{kq} \\ \delta_{io} \delta_{jp} D_{kq} \end{pmatrix} U_{ijk} \end{aligned} \tag{20}$$

where

$$G_{opq}^{\mu\nu} = G_{opq}^{\nu\mu} = P_{opq} J_{opq} \rho_o \rho_p \rho_q \left(\frac{\partial r_\mu}{\partial x} \frac{\partial r_\nu}{\partial x} + \frac{\partial r_\mu}{\partial y} \frac{\partial r_\nu}{\partial y} + \frac{\partial r_\mu}{\partial z} \frac{\partial r_\nu}{\partial z} \right) \tag{21}$$

The derivative matrices on the left are transposed because they act on the W^T_{lmn}. J_{ijk} is the Jacobian evaluated at node ijk; ρ_l is the 1D weight for the Gaussian integration. The individual matrix elements are just the coefficients of W_{lmn} and of U_{ijk}:

$$A_{ijklmn} = \sum_{o=0}^{N_r} \sum_{p=0}^{N_s} \sum_{q=0}^{N_t} \begin{pmatrix} D^T_{lo}\delta_{mp}\delta_{nq} \\ \delta_{lo}D^T_{mp}\delta_{nq} \\ \delta_{lo}\delta_{mp}D^T_{nq} \end{pmatrix} \begin{pmatrix} G^{1,1}_{opq} & G^{1,2}_{opq} & G^{1,3}_{opq} \\ G^{2,1}_{opq} & G^{2,2}_{opq} & G^{2,3}_{opq} \\ G^{3,1}_{opq} & G^{3,2}_{opq} & G^{3,3}_{opq} \end{pmatrix} \begin{pmatrix} D_{io}\delta_{jp}\delta_{kq} \\ \delta_{io}D_{jp}\delta_{kq} \\ \delta_{io}\delta_{jp}D_{kq} \end{pmatrix} \tag{22}$$

Because the W have been chosen to be Lagrangian interpolants, it is not necessary to do any summations over the indices of W when writing the final matrix equation, as the coefficients W_{lmn} are all 0, except for one which is 1. Thus the indices of W serve only to specify which equation is under consideration. In addition, this choice for the W means that these matrix elements, and the others considered below, do not change when the indices ijk and lmn are exchanged, which means that the resulting matrix will be symmetric. This symmetry can be used to speed up the numerical solution of the matrix equation.

2.3.5. Matrix Elements due to QU

The next integral which must be evaluated is $\int QWU\, dV$. This is simpler than the previous term: it is just

$$\int QWU\, dV = \sum_{o=0}^{N_r} \sum_{p=0}^{N_s} \sum_{q=0}^{N_t} W_{opq}U_{opq}\, Q_{opq}J_{opq}\rho_o\rho_p\rho_q \tag{23}$$

The matrix elements which come from this integral are again just the coefficients of W_{lmn} and U_{ijk}; because in this integral the indices of W and of U are the same, this matrix is diagonal:

$$B_{ijklmn} = Q_{ijk}J_{ijk}\rho_i\rho_j\rho_k\, \delta_{il}\delta_{jm}\delta_{kn} \tag{24}$$

2.3.6. Source Terms due to S

The first integral on the right side of the equation, $\int WS\, dV$, has a very similar form as the previous term:

$$\int WS\, dV = \sum_{o=0}^{N_r} \sum_{p=0}^{N_s} \sum_{q=0}^{N_t} W_{opq}S_{opq}J_{opq}\rho_o\rho_p\rho_q \tag{25}$$

This integral determines the source term of the matrix equation: for equation lmn, the source term is

$$\sigma_{lmn} = S_{lmn}J_{lmn}\rho_l\rho_m\rho_n \tag{26}$$

2.3.7. *Terms due to Natural Boundary Conditions*

The last two integrals which must be considered are the two boundary integrals which determine the natural boundary condition $\partial U/\partial v + \alpha U = \beta$ along Γ_1: $\int_{\Gamma_1} WP\alpha U \, dS$ and $\int_{\Gamma_1} WP\beta \, dS$. These are surface integrals, which are again most conveniently evaluated in terms of the local coordinates (r, s); accordingly, if $\mathbf{x}(r, s)$ is the parametric representation of a surface S, the integral must include the Jacobian of the surface transformation, which is (Apostol, 1967)

$$J(r, s) = \left| \frac{\partial \mathbf{x}}{\partial r} \times \frac{\partial \mathbf{x}}{\partial s} \right| \tag{27}$$

so that the integral of a function F over the surface is

$$\int_S F(\mathbf{x}) \, dS = \iint F(\mathbf{x}(r, s)) \left| \frac{\partial \mathbf{x}}{\partial r} \times \frac{\partial \mathbf{x}}{\partial s} \right| dr \, ds \tag{28}$$

Thus the first boundary integral over, for example, the top surface (t index $= N_t$) of a spectral element is

$$\int_{\Gamma_1} WP\alpha U \, dS = \sum_{o=0}^{N_r} \sum_{p=0}^{N_s} W_{opN_t} P_{opN_t} \alpha_{opN_t} U_{opN_t} \left| \frac{\partial \mathbf{x}}{\partial r} \times \frac{\partial \mathbf{x}}{\partial s} \right|_{opN_t} \rho_o \rho_p \tag{29}$$

and the second boundary integral over the same surface is

$$\int_{\Gamma_1} WP\beta \, dS = \sum_{o=0}^{N_r} \sum_{p=0}^{N_s} W_{opN_t} P_{opN_t} \beta_{opN_t} \left| \frac{\partial \mathbf{x}}{\partial r} \times \frac{\partial \mathbf{x}}{\partial s} \right|_{opN_t} \rho_o \rho_p \tag{30}$$

In a similar manner as for the volume integrals, W_{lmn} determines which equation is under consideration. Thus, to specify natural boundary conditions at (for example) the top surface of an element, a term

$$a_{ijN_t lmN_t} = P_{ijN_t} \alpha_{ijN_t} \left| \frac{\partial \mathbf{x}}{\partial r} \times \frac{\partial \mathbf{x}}{\partial s} \right|_{ijN_t} \rho_i \rho_j \delta_{il} \delta_{jm} \tag{31}$$

is added to the matrix equation, and a term

$$b_{ijN_t} = P_{ijN_t} \beta_{ijN_t} \left| \frac{\partial \mathbf{x}}{\partial r} \times \frac{\partial \mathbf{x}}{\partial s} \right|_{ijN_t} \rho_i \rho_j \tag{32}$$

is added to the source vector. Similar terms are added for the other boundary surfaces of each element, with care being taken to keep the vector $\partial \mathbf{x}/\partial r \times \partial \mathbf{x}/\partial s$ always pointing out of the element.

2.3.8. Matrix Equation Including Essential Boundary Conditions

To find the complete matrix equation for U_{ijk}, all the pieces derived above must be assembled, just as the corresponding integrals are added together in the integral equation. The complete equation is

$$\sum_{i=0}^{N_r} \sum_{j=0}^{N_s} \sum_{k=0}^{N_t} (A_{ijklmn} + B_{ijklmn} + a_{ijklmn})\, U_{ijk} = \sigma_{lmn} + b_{lmn} \tag{33}$$

If essential boundary conditions ($U = U_0$) are specified along some portion of the boundary, the solution space from which the U are drawn is restricted. In writing down the matrix equation, this restriction is achieved by setting U_{ijk} to the known values U_0 along the given parts of the boundary and moving the appropriate columns of the matrix to the right side of the equation where they modify the source terms. In addition, the corresponding rows of the matrix are dropped from consideration, so that the number of equations remains equal to the number of unknowns.

2.3.9. Specification of Continuity Along Element Boundaries

The derivation of the matrix equation which was described above has ignored the question of what to do about continuity of the solution along the boundary between two elements. In the above presentation, there was no communication between corresponding nodes on two adjacent element faces; thus, a solution to the matrix equation could be discontinuous across the interface. Such a discontinuity is undesirable, because the solutions to the integral equation are expected to be continuous in general. This difficulty is resolved in the following way: at each interface node, only one degree of freedom is allowed. That means, in effect, that the basis functions of those nodes are modified. The basis functions are no longer those described above (the Lagrangian interpolants inside each element); instead, each basis function belonging to an interface node is the sum of the elemental basis functions of the two (or more in the case of a corner) adjoining elements, so that the interface basis function is nonzero in more than one element. All integrals are still performed over individual elements; for the interface nodes, the integrals are nonzero if the integration is over any element adjoining the interface. The effect of this is that all the integrations are done exactly as described above, but the matrix elements and source terms for a given interface node are the sums of the matrix elements and

source terms from the individual elemental equations. This procedure is called direct stiffness summation.

2.3.10. Solution of the Matrix Equation

Once the matrix equation has been set up according to the above procedure, it must be solved. In the most general case, LU-decomposition of the matrix, followed by back-substitution, will work; however, because the matrix quickly gets very large as the N_l are increased (especially in 3D), it is desirable to take advantage of any special properties of the matrix. One rather simple property is the fact that the matrix is symmetric; this allows the use of Cholesky decomposition rather than general LU-decomposition (Golub and Van Loan, 1989).

2.3.11. Static Condensation

Another, somewhat more specific, property of the matrix is the organization of the nonzero entries that is imposed by the spectral element scheme: because the basis functions of the internal nodes are nonzero over only a single element, the internal nodes of any element are coupled only to the boundary nodes of that element. The boundary nodes of all the elements are all more or less coupled to each other. (The boundary coupling of course depends on the detailed structure of the mesh.) Because of this structure of the matrix, it is possible to usefully perform static condensation on it. This is essentially block Gaussian elimination (Golub and Van Loan, 1989), but is described in some detail here because it may be somewhat less familiar than the standard version of Gaussian elimination.

Consider the general (symmetric) matrix equation

$$\begin{pmatrix} [a] & [b]^T \\ [b] & [c] \end{pmatrix} \begin{pmatrix} [\phi_b] \\ [\phi_i] \end{pmatrix} = \begin{pmatrix} [f_b] \\ [f_i] \end{pmatrix} \tag{34}$$

where $[a]$, $[b]$, and $[c]$ are matrices. The $[a]$ matrix represents the coupling among boundary nodes, the $[b]$ matrix represents the coupling between internal and boundary nodes, and the $[c]$ matrix represents the coupling among internal nodes; subscript b indicates a boundary term and i denotes an interior term. The second set of equations in the above matrix equation can formally be solved for the values of the interior nodes:

$$[\phi_i] = [c]^{-1} ([f_i] - [b] [\phi_b]) \tag{35}$$

This formal solution for the interior nodes can then be substituted back into the first set of equations for the boundary nodes:

$$([a] - [b]^T [c]^{-1} [b]) [\phi_b] = [f_b] - [b]^T [c]^{-1} [f_i] \tag{36}$$

This equation now contains only the ϕ_b. Once it is solved, the previous equation may be used to find the ϕ_i. The advantage of this procedure for the particular matrices encountered with spectral elements comes from the organization of the matrices $[b]$ and $[c]$: because of the structure of the coupling described above, $[b]$ and $[c]$ consist of many small blocks which may be manipulated independently of each other. The inversion of $[c]$ thus consists of the independent inversion of many small blocks, and the multiplication $[c]^{-1}[b]$ consists of the independent multiplication of these small blocks with other small blocks. This matrix structure is clearly illustrated in Fig. 1: (a) shows a small three-element mesh, and (b) shows the corresponding structure of the matrix. By applying static condensation, it is thus possible to replace the inversion of one large matrix by the inversion of several smaller matrices, plus additional operations; this change speeds up the overall solution process considerably.

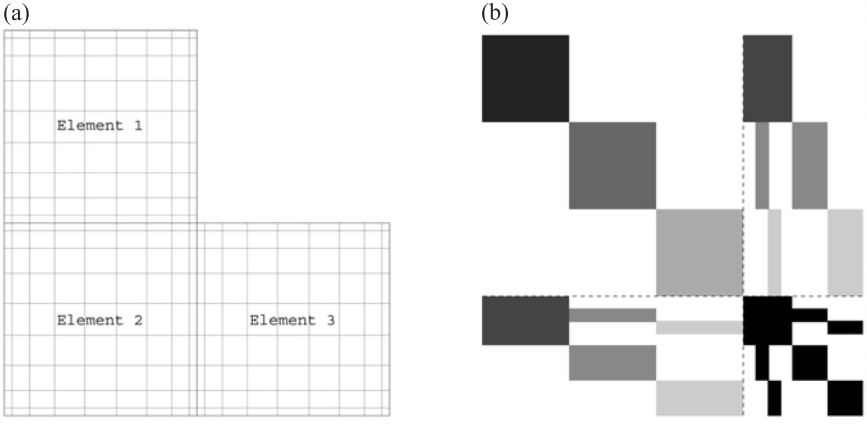


Fig. 1. (a) A small L-shaped sample mesh showing all nodes. Each element has $8 \times 8 = 64$ internal nodes. Elements 1 and 3 each have 26 boundary nodes which are not shared with any other element, while element 2 has 17 unshared boundary nodes. Elements 1 and 2 share 10 boundary nodes, as do elements 2 and 3; one boundary node is common to all three elements. There are thus 280 degrees of freedom. (b) A schematic representation of the full matrix of the sample mesh. The dotted lines show the main separation into internal and boundary nodes. The square blocks in the upper left correspond to the coupling between internal nodes of the same element: each gray level is one element. The rectangular blocks on the upper right and the lower left correspond to the coupling between internal nodes of a given element and the corresponding boundary nodes of that same element. Again, each distinct gray level represents one element. Note that some boundary nodes (i.e., rows or columns of the matrix) are shared by more than one element. Finally, the lower right corner corresponds to the coupling between different boundary nodes. (There is still structure in this portion of the matrix, but our present code does not take advantage of that remaining structure.)

In addition, it is desirable to number the elements in such a way that the bandwidth of the global boundary matrix is as small as possible. This numbering can also provide dramatic speedups, which depend on the extent to which the bandwidth is reduced. (As is evident in Fig. 1b, the boundary matrix also has a special structure like that of the overall matrix; at the present time our code does not take advantage of that additional structure.)

2.3.12. Iterative Solution of Linear Equations: Conjugate-Gradient Method

In the previous sections, we described various ways to speed up the solution of the matrix equation by direct inversion of the matrix. Another way to solve the matrix equation is by using an iterative method. Various iterative methods (Barrett *et al.*, 1994) can be used, but the matrices encountered in the present work share two properties that make the conjugate-gradient method in particular quite suitable. The first, which has already been described, is that the matrices are symmetric. The second is that they are positive-definite. In terms of the general Helmholtz equation shown in Sec. 2.3.1, this property of the matrices is implied by the sign of the coefficient Q: it is not negative anywhere in any of the PNP equations.

Detailed descriptions of the conjugate-gradient method can be found in many places (Barrett *et al.*, 1994; Hestenes and Stiefel, 1952; Press *et al.*, 1986). Generally, the most computationally-intensive step in the conjugate-gradient method is the calculation of a matrix-vector product $\mathbf{A}x$, where $\mathbf{A}$ is the matrix to be inverted, and x is a vector generated by other parts of the conjugate-gradient method. When using the spectral-element method, the best way to compute this product is to apply the operators described above to each spectral element; then, after all the element-local operations are completed, the necessary summations are performed to account for nodes on the boundary between multiple elements.

The A_{ijklmn} matrix elements described in Eq. (22) in Sec. 2.3.4 (recall that inside each spectral element, each node is addressed locally by three indices, one for each dimension) possess the most complex operator structure. They are processed by applying sub-operators, from right to left: first we apply the local derivative operators D in order to compute the matrix products $D_{io}\delta_{jp}\delta_{kq}x_{ijk}$, $\delta_{io}D_{jp}\delta_{kq}x_{ijk}$, and $\delta_{io}\delta_{jp}D_{kq}x_{ijk}$; next we multiply these intermediate results by the appropriate $G^{a,b}_{opq}$; and finally we apply the transposed derivative operators D^T in a similar fashion as for the first step. Because of the presence of the Kronecker deltas in the first and third stages, the operations count for these stages is $O((N_r+N_s+N_t)\,N_rN_sN_t)$. The operations count for the middle stage is $O(N_rN_sN_t)$. The auxiliary storage required is relatively modest: the derivative operators D occupy

$N_r^2 + N_s^2 + N_t^2$ storage locations, and the $G_{opq}^{a,b}$ each occupy $N_r N_s N_t$ storage locations.

In contrast, if the full A_{ijklmn} matrix were to be precomputed, even for each element individually, the operations count and storage requirements would both be $O(N_r^2 N_s^2 N_t^2)$, significantly worse than the above method. (Storage of, and multiplication by, the full global matrix are so horrific that we will not even consider them.)

The B_{ijklmn} matrix elements are simpler, since that portion of the matrix is diagonal. Thus the operations count and storage requirements for this portion of the operation are both $O(N_r N_s N_t)$.

Remaining are the operations required to include the boundary conditions, as well as the summations necessary to enforce continuity across element boundaries. Both the operations count and the storage requirements depend more specifically on the exact geometry being used, so it is more difficult to give expressions for these; but in general both the operations counts and storage requirements are small compared with the previous element-wise operations.

It is also worth pointing out that (most of) the above operations can be performed in parallel: it is quite feasible to perform all of the element-wise operations on different processors. However, we have not yet implemented such a parallelization, so we cannot comment on the degree of speedup that might be achieved.

The other issue which must be considered in the solution of the PNP equations is the choice of a suitable preconditioner. Because the diffusion coefficients for each ion vary quite widely between different elements, the "plain" un-preconditioned conjugate-gradient method does not converge very well, if at all. In effect, the matrix is too nearly singular. In the present work, we have used the simplest non-trivial preconditioner: the main diagonal of the global matrix, but without application of the pieces due to the boundary conditions. This preconditioner is easily computed from the expressions given above for the various matrix elements, and it is very easily applied. Use of this preconditioner makes the conjugate-gradient method converge rather reliably when applied to the PNP equations.

2.4. Numerical Checks; Conservation Laws

There are several levels at which the numerical methods can and must be checked for accuracy. First, the basic Helmholtz solver must be checked. This is done by the standard method of substituting known functions P, Q, and U into the left side of Eq. (9) and thus computing the function S. P, Q, and S are then specified as inputs to the Helmholtz solver, along with the appropriate boundary conditions. The result should then be (very close to)

the function U. For the spectral element method, it should be possible to get exponential convergence to the true solution, provided it is analytic, as the polynomial order inside each element is increased. In our experience, it is easily possible to reach an accuracy of a few hundred units of least precision; for standard IEEE 64-bit floating-point arithmetic, this corresponds to a relative error of no more than a few times 10^{-14}. The details of such a test are shown in an appendix.

It is also possible to solve various electrostatics problems for which analytic solutions are known: for example, the problem of a point charge located some distance from a plane at which a jump in dielectric coefficient occurs (Jackson, 1975). Here, too, it is possible to reproduce essentially exactly the known solution, including the surface charge induced at the plane of discontinuity.

The next level which needs to be checked is the overall PNP solver, including the Gummel iteration. It is possible to "bootstrap" the multi-dimensional solver by comparing it against a 1D PNP solver. We have implemented two independent 1D solvers, which agree quite well with each other. In the 1D case, it is also possible to do much more analytical integration of the differential equations than in the multi-dimensional case, so that a 1D solver is somewhat less dependent on complex numerics. Our 2D solver agrees very well with the 1D solvers when it is used to solve a 1D problem, and our 3D solver agrees with both the 1D and the 2D solvers when it is used to solve 1D and 2D problems, respectively. (Of course, when solving a 1D problem, the efficiency of the multi-dimensional solvers is quite abysmal compared with the efficiency of the 1D solvers, but as these are all quite small test problems, this is not a concern.)

The last verification which needs to be done is to check that the underlying physical laws (conservation of mass, divergence of electric field) are properly satisfied by the final solutions. In one sense, this is trivial, because the differential equations, which are being solved correctly, are a direct expression of the physical laws; however, it is always reassuring to check that the actual solutions in each simulation are correct. In the 2D and 3D PNP solvers, the divergence of the electric field is almost exactly equal to the local charge density: their difference (i.e., the error in the Poisson equation) is about 10^{-12} in nondimensional units almost everywhere. The only points at which the error is large are the corners where regions with different dielectric coefficients meet. The potential is continuous, but the electric field does become singular at such points, so it is not surprising that the error becomes large there. Although this does prevent the uniform exponential convergence described above, it is not a problem in these simulations because the error is so localized. Thus it is still quite possible to observe exponential convergence in integrated

quantities such as the current. The situation for the Nernst–Planck equation is quite similar.

3. SPECIFIC MODELS FOR GRAMICIDIN A

In the previous sections, we presented the derivation and solution of a set of linear equations arising from the spectral element method, but without reference to any particular spectral element mesh or geometry. Because proper specification of the computational domain, and in particular the placement of individual spectral elements, is quite important for the success of these simulations, a detailed description of the construction of the gramicidin mesh is presented here.

The basic guiding principle is that spectral elements perform well if the boundaries of the individual elements are placed to coincide with any discontinuities in material properties that may be present. In the case of the PNP channel simulations described here, there are several discontinuities. First, there are jumps in the dielectric coefficient between water-filled regions (where a dielectric coefficient in the neighborhood of 80 is used) and protein- and lipid-filled regions (where the dielectric coefficient is nearer 2). There may also be jumps between different water-filled regions: for example, it is quite possible that the effective dielectric coefficient that best describes the channel region is different from the coefficient that applies in the bulk of the baths. Second, there are jumps in the diffusion coefficients of the various ionic species: most importantly, the diffusion coefficient inside the protein and lipid regions is zero, since the ions do not penetrate into these regions at appreciable rates.

In the construction of a mesh that would honor these constraints, our starting point was a version of the gramicidin molecule that was provided by Ron Elber (private communication, 1998). This is a slight modification of the published structure of gramicidin (1MAG in the Brookhaven Protein Data Bank), refined with contributions from the CHARMM force-field, as well as charge assignments on the individual atoms as computed by AMBER; as a result of the CHARMM refinement, the dimer is not entirely symmetric. The molecule was sliced into 2-Å pieces along the channel axis, which had been rotated so that it coincided with the Z axis. In each of those slices, all the atoms in that slice were then plotted in their proper X-Y position. Each of these plots served to locate the opening of the channel in that slice, and so they locate the boundaries of some of the spectral elements in that slice (the most important boundaries, since these are the main discontinuities in the simulation). In placing the spectral element boundaries, our main concern was not with making them conform *exactly* to every single atom of the molecule. Instead, the opening of the

channel was approximated as a somewhat irregular hexagonal tube, with some atoms actually inside the nominal channel, and with some open areas outside the nominal channel. The hexagonal cross-sections were constructed by hand and eye, and there is thus some arbitrariness in their placement—arbitrariness which could, in principle, be used to improve the fit between simulations and experiments. In fact, however, no such fits were performed: all the results which are reported below were done with the same mesh, which was completely constructed before any simulations were run. To the extent possible within the framework of PNP theory, these simulations attempt to predict the IV curves of gramicidin, rather than to fit the theory to experiments. At the same time that the channel cross-sections were being built, the outer boundaries of the gramicidin molecule, as well as further internal boundaries inside the lipid layer, were placed. These boundaries are much less critical than the channel boundaries, since only the channel boundaries mark a discontinuity in the dielectric coefficient as well as in the various ionic diffusion coefficients. Once the 2D grid for each slice was constructed, all the cross-sections were assembled into 3D spectral elements.

The upper and lower interfaces between the lipid bilayer and the bath, as well as the mouths of the channel, were treated simply as flat planes, again with an abrupt jump in both the dielectric coefficient and the diffusion coefficients. Taller spectral element layers were then added to include parts of the bath regions. A partial view of the final spectral element mesh is shown in Fig. 2. In total, there are 336 3D elements.

Next, the charge distribution of the gramicidin molecule was computed. Each atom was assumed to have a Gaussian charge distribution centered on the known position of the atom and with the proper amplitude such that the integrated charge equaled the known charge of the atom; a nominal "radius" of each Gaussian cloud of 1.5 Å was also assumed (but see also the discussion in Sec. 4.1.2). All the Gaussian clouds were then evaluated and summed at each point of the mesh, thus producing the overall continuous fixed charge distribution. In this procedure, any fixed charge in or on the lipid bilayer was ignored, and the bilayer was treated simply as an uncharged barrier to the ions. Such a treatment is clearly incomplete, and the effects of lipid charge and bilayer thickness on the IV curves of gramicidin will be reported elsewhere.

Finally, it was also necessary to have good values for the diffusion coefficients of the various ions inside the channel. The diffusion coefficient of Na^+ was found by a least-squares fit of 1D PNP to the experimental gramicidin data. The details of this fitting procedure have been reported elsewhere (Chen, Lear, and Eisenberg, 1997), so it will not be described here. The experimental IV curves were measured in five symmetrical solutions of

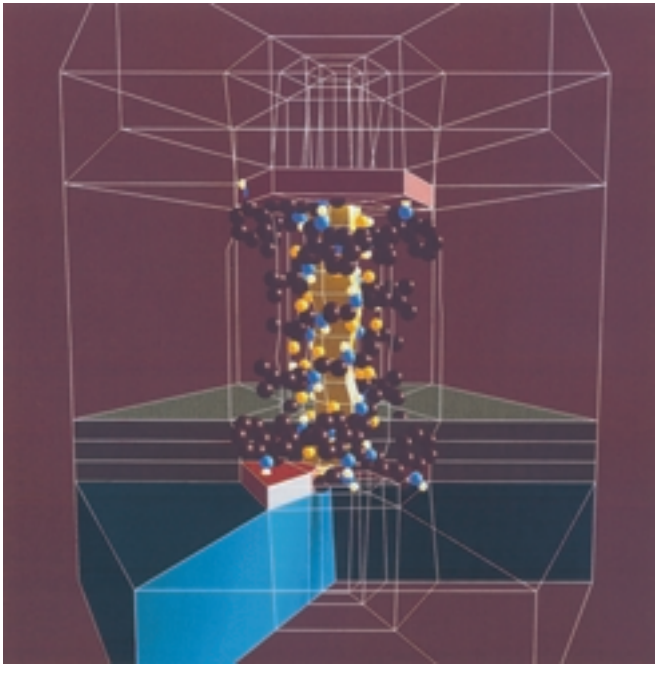
Na^+Cl^- and K^+Cl^- with salt concentrations ranging from 0.1 up to 2.0 M. Measurements were conducted with both wild-type gramicidin and a specifically modified gramicidin. The least-squares-fitted 1D PNP gave a good fit to the twenty measured IV curves (five each of Na^+Cl^- and K^+Cl^- in two kinds of gramicidin) simultaneously, with a single diffusion coefficient for Na^+ ions and one for Cl^- ions. Only these Na^+ and Cl^- diffusion coefficients are considered in the 3D calculations presented here; the 1D fitting procedure also produces a 1D effective charge distribution which is not used in the present paper. The baseline diffusion coefficients used in this work are, in cm^2/sec:

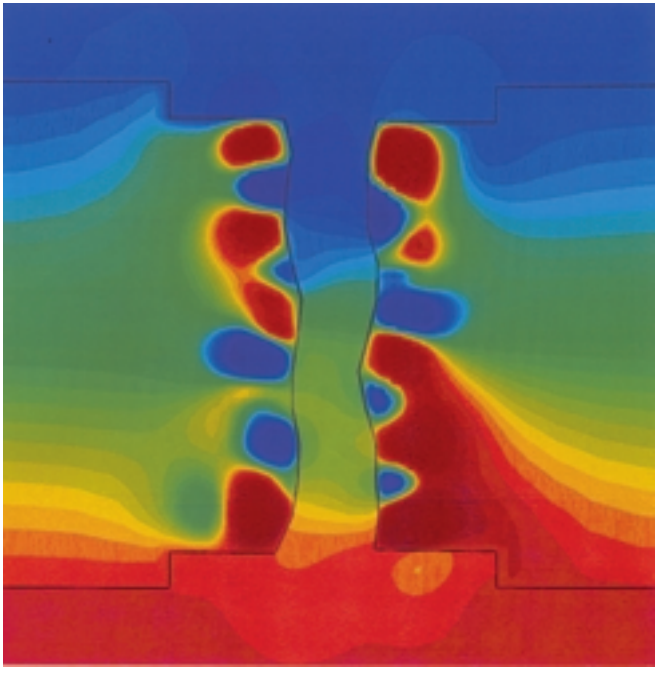
	channel	bath
Na^+	4.67×10^{-7}	2.0×10^{-5}
Cl^-	1.0×10^{-10}	1.9×10^{-5}

The diffusion coefficients in the baths are taken from standard tables (Robinson and Stokes, 1959). The channel value of D_{Na^+} is that determined from the 1D fit, and is likely to be close to the true value. However, the 1D fit yielded a value for D_{Cl^-} of 1.82×10^{-99}. This exceedingly small number is most likely unphysical, and would in any case have caused significant numerical difficulties; therefore, the value shown in the table was chosen somewhat arbitrarily. The estimate of D_{Cl^-} from the 1D fit is unreliable because Cl^- carries so little current in the gramicidin channel. Indeed, virtually no Cl^- can be found inside the gramicidin channel because it has such a negative electrical potential under nearly all conditions. The partial charges of the gramicidin molecule are so arranged that the potential inside the pore is negative, even though the molecule is overall neutral. This is not the first case where the shape of the electric field is more important than its average value. The Cl^- current is thus very small, and so the total current is very insensitive to D_{Cl^-}. Therefore it is very difficult to estimate D_{Cl^-} from measurements of total current alone. In effect, the only conclusion about D_{Cl^-} that can be drawn from the 1D fit is that it can be any value less than some upper bound, which is approximately D_{Na^+}. The effects of the value chosen for D_{Cl^-} are discussed below.

4. RESULTS AND DISCUSSION

The first thing that must be checked is the accuracy of the solution as a function of the polynomial order: since we do not have an analytic solution against which to check the simulation results, we must rely on the





solver to eventually yield an answer which has enough significant figures to be useful. We therefore chose symmetric 1 M solutions, with 200 mV applied across the channel, as the nominal test case, and varied the polynomial order from 5 to 12. The results are shown in the table below. The last column shows the absolute value of the difference between the current at that order and the current at the previous order.

Order	Na^+ flux atoms/sec	Cl^- flux atoms/sec	current pA	Δ current pA
5	$+1.646765 \times 10^7$	-3.829617×10^2	+2.638470	
6	$+1.602774 \times 10^7$	-3.857658×10^2	+2.567991	0.070479
7	$+1.630153 \times 10^7$	-3.843817×10^2	+2.611856	0.043865
8	$+1.622969 \times 10^7$	-3.844139×10^2	+2.600346	0.011510
9	$+1.625230 \times 10^7$	-3.841401×10^2	+2.603967	0.003622
10	$+1.623766 \times 10^7$	-3.843262×10^2	+2.601622	0.002345
11	$+1.624218 \times 10^7$	-3.842449×10^2	+2.602347	0.000725
12	$+1.624309 \times 10^7$	-3.842266×10^2	+2.602493	0.000146

Fig. 2. (Opposite top) The geometry used to solve the Poisson–Nernst–Planck equations for the Gramicidin A channel. The turquoise regions at the bottom of the picture represent the bath; some elements have been cut away in order to show the geometry more clearly. There is a corresponding bath region at the top, shown here only as a wire mesh. The beige regions at the top and bottom mouths of the channel represent "atrium" regions, which in this computation have the same properties as the bath proper. Again, some elements at the bottom mouth of the channel have been cut away. The central yellow regions represent the channel proper. The entire channel is shown. The lighter green regions toward the rear of the computational domain represent a portion of the lipid and protein. Most of the elements representing the bilayer have been cut away in order to make the channel visible; in the computation, the entire region between the baths is filled by such elements. The backbone and charged atoms of the Gramicidin A dimer are shown as well: carbon atoms are black, nitrogen atoms are blue, oxygen atoms are yellow, and hydrogen atoms are off-white. The electrodes are located at the top and the bottom of the mesh. In total, there are 336 elements, arranged in 16 layers of 21 elements each.

Fig. 4. (Opposite bottom) A cross-section of the electrical potential of the 3D simulation, showing the strong variation both along and across the axis. This is not very axisymmetric on an atomic scale, and in fact internal potential differences exceed the voltage applied at the electrodes by an order of magnitude: the internal potentials range from -870 to $+930$ mV, whereas the applied voltage is only 200 mV. The black line shows the boundary between the protein + lipid region and the bath + channel region.

It is clear that the simulation converges quite well; however, the runtime per IV point increases quickly as the polynomial order is increased. For polynomial order 8, there are about 180000 total degrees of freedom, and the above simulation takes about 5 hours of CPU time to compute on a 450 MHz Pentium-III. In contrast, for order 12, there are about 600000 total degrees of freedom, and the simulation takes about 26 hours of CPU time on the same machine. For a full IV curve, the calculations are slighly more efficient, since each successive point can use the previous solution as a starting point, so that somewhat fewer Gummel iterations are required, but total runtimes are still of the order of days for order 8, and (would be) weeks for order 12. Since the order-8 simulation appears to have a relative error of about 0.1%, we decided to select polynomial order 8 as the standard setting for these simulations, in order to keep this study computationally feasible. Thus all of the cases reported below use 8th-order spectral elements, except in Sec. 4.1.2, where 9th-order elements were used.

4.1. Experimental Data versus Simulations

In Fig. 3, we show the first major result of this work: a comparison between the experimentally measured IV curves of gramicidin at several symmetric concentrations and the curves predicted by the 3D PNP simulation. (A short note describing this result, but without any details of the solution procedure or any of the other results described below, has been published elsewhere (Hollerbach *et al.*, 2000).) In generating these IV curves, we scaled the concentrations in each bath by the appropriate activity coefficients. As is evident in the figure, the agreement between experiment and simulation is quite good, especially given that the 3D runs were made without fits of any kind.

In order to test the quality of the agreement between these simulations and the experimental data, a series of sensitivity analyses of the simulations were done: different parameters in the input specification were varied and the changes in the IV and flux curves were observed. The results for the different parameters are discussed below.

4.1.1. Diffusion Coefficient of Cl^-

The first parameter that we examined was D_{Cl^-}. As was previously described, the 1D fitting procedure yielded a value of 1.8×10^{-99} cm^2/sec, which would have made the Nernst–Planck equation for Cl^- numerically singular. Therefore, an arbitrarily chosen value of 10^{-10} cm^2/sec was used instead. In order to assess the effect of this choice, a series of runs was

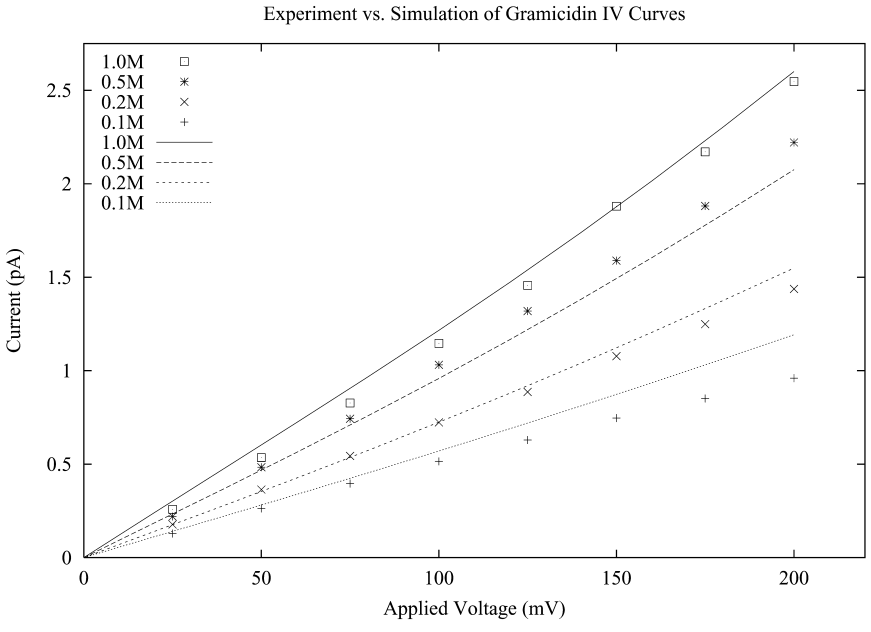


Fig. 3. A comparison between the simulated IV curves and the corresponding experimentally measured IV curves. The points represent the experimental data, and the lines represent the simulated IV curves. Note that the concentrations were scaled by the activity coefficients inside the simulations. The data shown here has been published elsewhere (Hollerbach *et al.*, 2000).

made with D_{Cl^-} ranging from 10^{-11} to 10^{-6} cm^2/sec. 10^{-6} cm^2/sec is larger than D_{Na^+}, and it is unlikely that the true D_{Cl^-} will be as high as that. All other parameters were kept fixed at settings duplicating those of the nominal symmetric 1 M case shown in Fig. 3, with an applied voltage of 200 mV. The results are shown in the following table:

D_{Cl^-} cm^2/sec	Na^+ flux atoms/sec	Cl^- flux atoms/sec	current pA
10^{-11}	$+1.623\times10^{7}$	-3.882×10^{1}	$+2.600$
10^{-10}	$+1.623\times10^{7}$	-3.844×10^{2}	$+2.600$
10^{-9}	$+1.623\times10^{7}$	-3.844×10^{3}	$+2.601$
10^{-8}	$+1.623\times10^{7}$	-3.844×10^{4}	$+2.606$
10^{-7}	$+1.623\times10^{7}$	-3.839×10^{5}	$+2.662$
10^{-6}	$+1.622\times10^{7}$	-3.793×10^{6}	$+3.206$

It is clear that the Na^+ flux carries the majority of the current. Even at the largest value of D_{Cl^-}, the Cl^- flux carries only about a quarter of the

current. If the true value of D_{Cl^-} is smaller than 10^{-7}, any error incurred will be less than about 2%, which starts to become difficult to distinguish from both numerical errors of other kinds and experimental errors. Thus it seems safe to conclude that the arbitrarily selected value of 10^{-10} for D_{Cl^-} will not affect the IV curves presented here by very much.

4.1.2. Size of Gaussian Atom "Clouds"

The next parameter that was examined was the size of the Gaussian clouds that represent the charge of the fixed atoms. In the main series of runs, the Gaussians were given a "radius" σ of 1.5 Å. This is slightly larger than the usual radius of an atom, which is somewhere between 0.5 and 1 Å. On the other hand, it probably does capture at least some of the effects of thermal motions of the gramicidin molecule. In order to assess the importance of this parameter, a series of runs with varying σ ranging from 0.5 to 1.5 was made. Again, all other parameters were kept fixed at settings duplicating those of the nominal symmetric 1 M case shown in Fig. 3, with an applied voltage of 200 mV, with one exception: the polynomial orders of the spectral elements were increased to 9 for this series of runs, because at the smaller values of σ the higher polynomial orders were needed to properly resolve the more point-like charges. The results are as follows:

σ Å	Na^+ flux atoms/sec	Cl^- flux atoms/sec	current pA
0.5	$+1.453 \times 10^7$	-3.369×10^2	+2.328
0.7	$+1.515 \times 10^7$	-3.353×10^2	+2.427
0.9	$+1.553 \times 10^7$	-3.380×10^2	+2.488
1.1	$+1.584 \times 10^7$	-3.470×10^2	+2.537
1.3	$+1.608 \times 10^7$	-3.628×10^2	+2.576
1.5	$+1.623 \times 10^7$	-3.844×10^2	+2.600

It is clear that the radius of the ions does affect the fluxes, but the effect is not so dramatic that it would make comparisons between simulation and experiment impossible: our choice of σ introduces an error that is less than about 12% in the worst case. If the true value of σ is greater than 1 Å, the error incurred will be less than about 4%, which again is difficult to distinguish from both numerical errors of other kinds and experimental errors. Thus, here too it seems safe to conclude that the value of 1.5 Å for σ will not affect the IV curves by very much.

4.1.3. Dielectric Coefficients of Bath, Channel, and Lipid

Next, we examined the effects of the dielectric coefficients of the bath, the channel, and the lipid, respectively, on the channel current. The bath

dielectric coefficient was set to either 60 or 80; the channel coefficient was set to one of 40, 60, or 80; and the lipid coefficient was set to one of 2, 4, or 6. All other parameters were kept fixed at settings duplicating those of the nominal symmetric 1 M case, with an applied voltage of either 25 or 150 mV. Below, we show the matrix of currents (in pA) for the 150-mV case; the trends for the 25-mV case are essentially the same. In each submatrix, the column specifies the lipid dielectric coefficient (shown at the top), and the row specifies the channel dielectric coefficient (shown at the far left).

	bath dielectric = 60			bath dielectric = 80		
	2	4	6	2	4	6
40	1.469	1.524	1.562	1.459	1.513	1.549
60	1.741	1.739	1.734	1.720	1.718	1.713
80	1.908	1.874	1.845	1.876	1.842	1.814

It is clear that the bath dielectric coefficient does not significantly affect the currents: in going from 60 to 80, there is a consistent decrease in current of about 1%; as stated above, such a small variation is hard to distinguish from other sources of error. Because the dielectric coefficient of water is well known, it is safe to assume that 80 is the correct value to use. The lipid dielectric coefficient affects the currents slightly more: in going from 2 to 6, the current varies between -3 and $+6\%$, depending on the channel dielectric coefficient. Finally, the channel dielectric coefficient affects the currents most: in going from 40 to 80, the current increases by between 17 and 30%. It is not surprising that this parameter has the most effect, because the channel is the most critical region in determining the current. Overall, the best fit to the experimental values is found when the bath and channel dielectric coefficients are both 80, and the lipid dielectric coefficient is 2. These values are reasonable.

4.1.4. Position of Atoms

Another parameter class that may be varied is the position of the individual atoms. Multidimensional PNP simulations are not yet advanced enough to extract structural information such as number and position of charges from functional information such as IV curves. Nonetheless, some simple structural information can be inferred—i.e., the given structure does or does not produce the given IV curve. To make such judgments, it is useful to know how sensitive the IV curves are to the placement of the individual charges, and so we made a series of runs with the positions of all

atoms perturbed by a random amount. Each cartesian coordinate was perturbed by a uniform random amount between -0.25 and $+0.25$ Å. The fractional charge on each atom was kept constant, as was the shape and location of the spectral elements making up the permeable channel. While such perturbations are certainly unphysical in the sense that they do not preserve chemical bond lengths or angles, they do offer a starting point for a rough check of the sensitivity of the IV curves with respect to charge distribution; it is in this spirit *only* that they are presented here. 26 perturbed sets of atom positions were generated, and all were run with the same boundary conditions of the nominal symmetric 1 M case with 200 mV applied. The current for the unperturbed case is 2.602 pA. The currents for the perturbed cases ranged from 1.113 to 4.042 pA. The average was 2.321 pA, and the median was 2.227 pA. These results do indicate that the current is fairly sensitive to the placement of the charges, and that therefore it may be possible to make at least simple judgments about whether or not a given structure can produce a particular IV curve.

4.2. Variation of Internal Potentials with External Conditions

After performing the various sensitivity analyses described above, we examined in greater detail the potentials, both inside the channel and throughout the gramicidin molecule. Figure 4 shows a color contour map of the potential on a plane that passes through the central axis of the channel. It is clear that the potential distribution inside the molecule very much follows the spiral structure of the molecule itself, and it is important to note that the potential differences inside the molecule reach nearly 2 volts. In contrast, the voltage applied across the electrodes is never greater than about 200 mV, an order of magnitude lower. Thus the potential variations in the vicinity of the channel are dominated by the internal potential of the molecule itself, and the applied electrode voltage adds only a relatively small perturbation. Inside the channel proper, the potential is closer to the electrode potentials, because the internal electric fields of the molecule are damped by the transition to the higher dielectric coefficient of the channel, but there are still areas where the gradient is across the channel rather than along it, and near the top of the channel there is a region where the potential is more negative than in the bath above the channel. In Fig. 5, we plot the average potential as a function of position along the channel for two applied voltages, 0 and 100 mV, and two external bath concentrations, 0.1 and 1.0 M. The averaging was done by multiplying the three-dimensional potential field by a three-dimensional gaussian of half-width 1 Å, then integrating the result over the entire computational

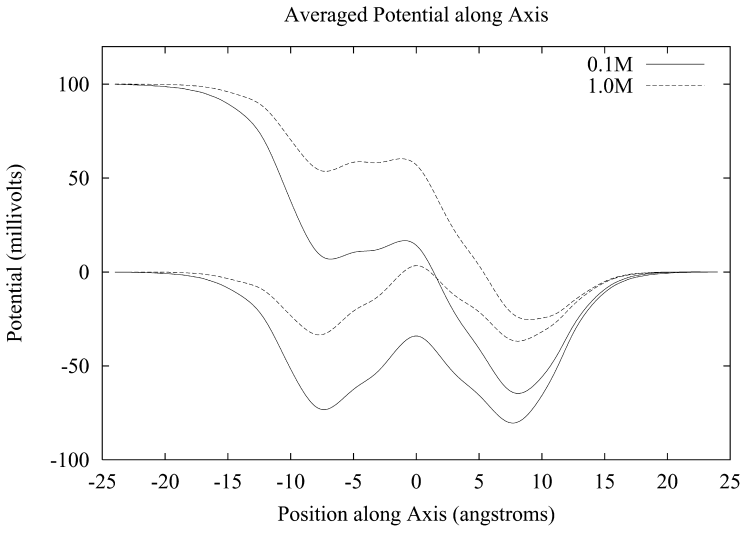


Fig. 5. The averaged potential along the channel axis as a function of the external concentration and the externally-applied voltage. The solid lines show the potential with symmetric bath concentrations of 0.1 M, and the dotted lines show the potential with bath concentrations of 1.0 M. The applied voltage may be read off the left edge of the graph.

domain. The gaussian was moved along the central axis of the channel in steps of 0.1 Å. Thus each point on the curve represents the average potential in a small sphere centered on that point along the channel axis. From the figure, it is clear that the mobile ions significantly affect the internal potential variation: as the concentration of ions increases, they shield and attenuate the fixed charge of the gramicidin molecule. This effect has not to our knowledge been taken into account in previous work. The figure also shows that, at least for fixed bath concentration, the overall potential may be approximately decomposed into an internally-generated component and an externally-applied component (Roux, 1997; Roux, 1999). This decomposition will be further described in a future publication.

4.3. Construction of and Results for a Reduced (2D Circularly-Symmetric) Model

Another question of interest is finding the extent to which the 3D model of gramicidin presented in the previous sections can be simplified. The 3D model is rather complicated even though gramicidin is much

simpler than other channels. A significant amount of work was required both to construct the model (human work) and to compute each point on the IV curves (computer work). Since the gramicidin channel is a roughly axisymmetric structure (leaving aside details of atomic position and of the spiral structure of the backbone of the molecule), it seemed not entirely unreasonable to ask what the effect might be of creating an axisymmetric model of gramicidin, in which each charge would be smeared out in a ring around the axis of the channel. If this could be done, the amount of computer time at least could be very significantly reduced, because the model would then be two-dimensional: r and z. With our present codes, it would be impossible to do any fitting of 3D simulations to experimental data in order to do any kind of parameter extractions: because of the number of runs required, it would take too long. However, such a reduced 2D model is simple enough to allow extraction of some parameters, and a reduced 2D model is far more realistic than a 1D model.

A priori it is not clear that such an axisymmetric model of gramicidin should work very well. The spiral structure shown in Fig. 4 would not be preserved in such a reduced 2D model. It was possible, however, to assess the likely success of such an axisymmetric model. We performed a "pseudo"-2D simulation using the 3D model by making 360 copies of each atom, each copy displaced from the previous one by a 1° rotation about the channel axis, and then dividing the resulting charge distribution by 360. The 1° spacing is small enough that the charge distribution is essentially constant as a function of the angle. The results of this "pseudo"-2D calculation were almost indistinguishable from the real 3D calculation, and showed that proper construction of the reduced 2D model would certainly be worthwhile. In Fig. 6, the IV curves computed from the full 3D model are compared with the IV curves from the reduced 2D model. It is clear that the two models are in excellent agreement, even though there is a slight systematic variation which depends on the concentrations of the baths. Certainly it will be possible to use the reduced model for parameter fitting. The reason for the agreement is not clear: it may be a consequence of the particular design of the gramicidin molecule (which after all has been shaped by evolution to function as an ion pipe) and not a consequence of a more general physical or mathematical property.

The closeness of the fit is perhaps somewhat counter-intuitive, and so we present here a detailed description of how the 2D model was constructed, and how the 2D charge distribution was computed. In building the 2D model, the starting point was the 3D geometry whose construction was described in Sec. 3. The cross-sectional area of the hexagonal channel was calculated at each point along the channel axis, and from that area an

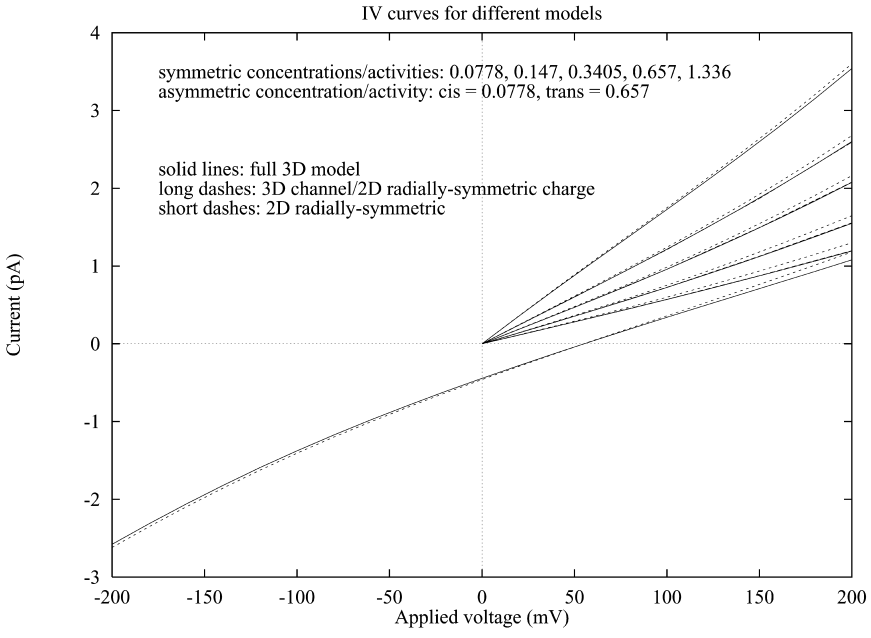


Fig. 6. A comparison between the IV curves of the 2D axisymmetric model and of the full 3D model. Note the rather close agreement, for both symmetric and asymmetric bath solutions, with larger relative errors at lower concentrations.

equivalent radius was computed. (The 2D model, which has radial symmetry, of course imposes a circular cross-section.) The result was a somewhat complicated dependence of radius as a function of position along the axis, approximated fairly well by five parabolic arcs chosen such that the tangents were continuous at the connection points. As in the 3D model, this curve is the most important part of the 2D model, because it marks the boundary between high-dielectric permeable channel and low-dielectric impermeable protein. Five spectral elements representing the channel were therefore placed such that the outer boundary of each element coincided with one of the five parabolic arcs. The remaining elements were then placed to appropriately match up with the channel elements. The central portion of the resulting mesh is shown in Fig. 7; the total number of elements is 55. The total number of degrees of freedom is far smaller than for the full 3D model, ranging from about 3650 for 8th-order spectral elements to about 11000 for 14th-order elements. For the runs shown here, we used 8th-order elements, which had about the same accuracy as for the 3D runs, but ran in around 15 seconds of CPU time per Gummel iteration.

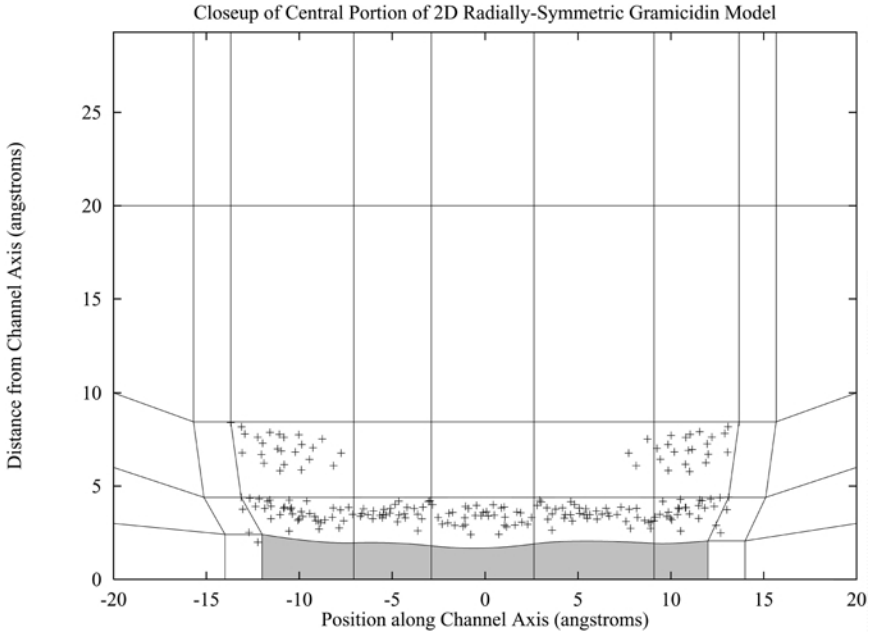


Fig. 7. The central portion of the 2D radially-symmetric model for the gramicidin A channel. The five elements that are shaded represent the channel proper. The crosses show the positions of the atoms in (r, z) space. The total number of elements is 55.

After building the mesh, it was necessary to compute the appropriate fixed charge distribution. It is not sufficient to simply compute (r, z) for each atom, then place a 2D Gaussian charge cloud at that location in (r, z) space. Because each atom of a given type has a fixed size regardless of its position, an atom that is farther from the axis occupies a smaller arc than a similar atom closer to the axis. In order to properly capture this effect, it is necessary to place each atom in 3D (r, θ, z) space, then integrate over θ to get the proper effective charge in 2D (r, z) space. Fortunately, if the atoms are treated as 3D Gaussian clouds, it is possible to do this integration analytically, with a reasonably simple result. If the original 3D charge distribution is written as

$$Q(x, y, z; x_0, y_0, z_0) = \frac{Q_{\text{total}}}{\sigma^3 \pi^{3/2}} e^{-((x-x_0)^2+(y-y_0)^2+(z-z_0)^2)/\sigma^2} \tag{37}$$

then the effective charge in 2D (r, z) space becomes, after some manipulations,

$$Q_{\text{eff}}(r, z; r_0, z_0) = \frac{Q_{\text{total}}}{\sigma^3 \pi^{3/2}} e^{-((r-r_0)^2+(z-z_0)^2)/\sigma^2} 2\pi M(2rr_0/\sigma^2) \tag{38}$$

where $M(z)$ is related to the modified Bessel function (Abramowitz and Stegun, 1972) of the first kind, of order 0, by $M(z) = I_0(z)\, e^{-z}$. Here x, y, z, r denotes the position at which the amplitude of the charge cloud is being evaluated, while x_0, y_0, z_0, r_0 denotes the position of the atom. Thus 2D Gaussian clouds do effectively reappear, but with an additional factor which takes into account the distance of the atom from the axis.

5. CONCLUSIONS

In summary, we find that the PNP model of gramicidin seems to fit the available data quite well. The model is not very sensitive to the size of the atomic charge clouds; but it is quite sensitive to the positions of those charge clouds. This sensitivity confirms the importance of correctly computing electrostatic interactions in performing studies of current flow through channels. Interestingly, the model is quite *in*sensitive to rotational averaging of the atomic charge clouds around the axis of the channel; thus it would appear that only the distance from the axis (r) and the position along the axis (z) are important, at least for gramicidin. This insensitivity allows construction of a satisfactory 2D reduced model of gramicidin. The model is also quite insensitive to variations in the diffusion coefficient of Cl^-, which is just as well since that coefficient is not well determined experimentally.

The 2D reduced model of gramicidin is quite helpful, because it enables parameter extraction from experimental data with much more realism than a 1D model, yet much less computational effort than a full 3D model. Indeed, parameter extraction with a 3D model is likely to be difficult even after several more iterations of Moore's law. Finally, we find that when varying the dielectric coefficients, the channel coefficient has the largest effect on the current, followed by the lipid/protein coefficient, and the bath coefficient has the least effect. This ordering is not surprising. The best fit is found when the channel has the same dielectric coefficient as the bulk of the baths; however, the PNP equations do not contain enough atomic detail to be able to infer that the interior of the channel is therefore "watery" in the same way that the baths are "watery."

APPENDIX: DETAILED TEST OF CONVERGENCE OF SOLVER

In this Appendix we describe the details of one two-dimensional synthetic test case which we use to check our spectral-element Helmholtz

solver. We begin with the specified functions $U = e^{-xy}$, $P = 1 + 0.1\sin(2x) + 0.15\cos(y)$, and $Q = e^{-2\cdot(x-y)} - 1$. These are substituted into the differential equation $\nabla\cdot(P\nabla U) - QU = -S$ to determine S. However, we do not do this substitution entirely by hand, and then evaluate the resulting mathematical expression; rather, we write code fragments that evaluate the required (analytical) derivatives of P and U, then combine these according to the (analytical) expression for the PDE. The following is a fragment of C code from the inner portion of one loop in the setup phase of the program:

```
ufunc(x, y, &u, &ux, &uy, &uxx, &uyy);
qfld[k]=exp(-2.0*(x-y))-1.0;
pfld[k]=1.0+0.1*sin(2.0*x)+0.15*cos(y);
px=0.2*cos(2.0*x);
py=-0.15*sin(y);
sfld[k]=-(pfld[k]*(uxx+uyy)+px*ux+py*uy-qfld[k]*u);
```

Here `ufunc()` is a routine which evaluates the specified function U and its first and second partial derivatives. Note that the value `u` is not saved in any array, whereas P, Q, and S are saved. The reason for this arrangement is that it has too often been our experience that, as the test program gets changed, the computation of S gets out of sync with the computations of P and Q, so that the test program converges exceedingly accurately to the wrong solution. This makes debugging difficult.

Figure 8 shows the domain on which the differential equation is solved. X varies from 0 to 2, and Y varies from 0 to 1.4. Note that both the orientation and the ordering of the elements are randomized in this mesh; such randomization is useful because it tests more parts of the code. Essential boundary conditions are applied along the left and right sides of the domain; i.e., the function $U = e^{-xy}$ is evaluated at the locations of the nodes along the left and right vertical edges, and those values are applied at the nodal locations. Along the top and bottom edges, natural boundary conditions $\partial U/\partial v + \alpha U = \beta$ are applied, with $\alpha = \sin(xy) + \cos(x)$, and β calculated from the values of U, α, and the outward normal vector $\hat{n}$.

The results are shown in the following table: the first column is the polynomial order (the same in both directions inside each element), the second column is the total number of degrees of freedom, and the third and fourth columns show the maximum pointwise error, i.e., the maximum value of $|U_{\text{computed}} - e^{-xy}|$, for the direct and the conjugate-gradient solvers, respectively.

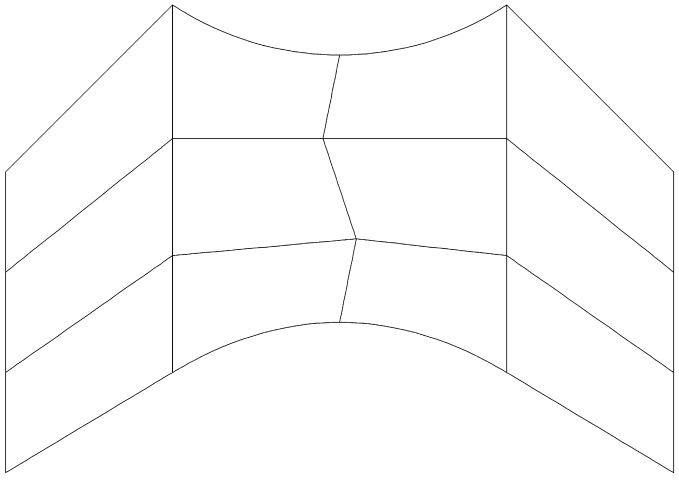


Fig. 8. A test mesh for the 2D spectral-element Helmholtz solver. The domain extends from 0 to 2 in x, and from 0 to 1.4 in y. The order and orientation of the elements are random. Essential boundary conditions are applied on the left and right vertical edges, and natural boundary conditions are applied on the top and bottom edges.

1	20	2.796×10^{-2}	2.796×10^{-2}
2	63	4.269×10^{-3}	4.269×10^{-3}
3	130	1.689×10^{-4}	1.689×10^{-4}
4	221	1.872×10^{-5}	1.872×10^{-5}
5	336	1.372×10^{-6}	1.372×10^{-6}
6	475	6.679×10^{-8}	6.679×10^{-8}
7	638	1.557×10^{-8}	1.557×10^{-8}
8	825	1.711×10^{-9}	1.711×10^{-9}
9	1036	2.400×10^{-10}	2.400×10^{-10}
10	1271	3.991×10^{-11}	3.991×10^{-11}
11	1530	3.710×10^{-12}	3.767×10^{-12}
12	1813	8.668×10^{-13}	9.022×10^{-13}
13	2120	1.309×10^{-13}	1.404×10^{-13}
14	2451	3.932×10^{-14}	8.187×10^{-14}
15	2806	2.706×10^{-14}	1.000×10^{-13}
16	3185	5.859×10^{-14}	1.146×10^{-13}
17	3588	6.007×10^{-14}	1.008×10^{-13}

It is evident that there is exponential convergence to the true answer for both the direct and the conjugate-gradient solvers. The errors are identical up to order 10; for higher orders, there are slight differences between the

two solvers, and the maximum accuracy that can be reached by the conjugate-gradient solver is just a bit less than that of the direct solver. However, the difference is not very significant, and both solvers can compute nearly-exact solutions.

The three-dimensional solver has been tested in a very similar manner, with similar results.

ACKNOWLEDGMENTS

We are grateful to Raphy Coifman and Steve Orszag for introducing us to each other and to Dennis Healy for his steadfast support and interest. We thank David Busath for providing the experimental gramicidin data against which these simulations were compared. This work was made possible by DARPA Grant N65236-98-1-5409.

REFERENCES

Abramowitz, M., and Stegun, I. A., (ed.) (1972). *Handbook of Mathematical Functions*, National Bureau of Standards.

Apostol, T. M., (1967). *Calculus*, 2nd ed., Wiley.

Barrett, R., Berry, M., Chan, T., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., and van der Vorst, H. (1994). *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM Press.

Bockris, J. O'M., and Reddy, A. K. N. (1998). *Modern Electrochemistry*, 2nd ed., Plenum.

Chen, D.-P., Lear, J., and Eisenberg, R. S. (1997). Permeation through an open channel. Poisson–Nernst–Planck theory of a synthetic ionic channel. *Biophys. J.* **72**, 97.

Courant, R., and Hilbert, D. (1989). *Methods of Mathematical Physics* (translation of german original), Wiley.

Ghaddar, N. K., Karniadakis, G. E., and Patera, A. T. (1986). a conservative isoparametric spectral element method for forced convection; application to fully developed flow in periodic geometries. *Num. Heat Transfer* **9**, 227.

Golub, G. H., and Van Loan, C. F. (1989). *Matrix Computations*, 2nd ed., Johns Hopkins.

Gummel, H. K. (1964). A self-consistent iterative scheme for one-dimensional steady-state transistor calculations. *IEEE Trans. Electron Devices* **ED-11**, 445.

Hestenes, M., and Stiefel, E. (1952). Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Stand.* **49**, 409.

Hollerbach, U., Chen, D.-P., Busath, D. D., Eisenberg, and R. S. (2000). *Predicting Function from Structure Using the Poisson–Nernst–Planck Equations: Sodium Currents in the Gramicidin A Channels*, Langmuir, Vol. 16, p. 5509.

Jackson, J. D. (1975). *Classical Electrodynamics*, 2nd ed., Wiley.

Jacoboni, C., and Lugli, P. (1989). *The Monte Carlo Method for Semiconductor Device Simulation*, Springer Verlag.

Jerome, J. W. (1996). *Analysis of Charge Transport: A Mathematical Study of Semiconductors*, Springer Verlag.

Kurnikova, M. G., Coalson, R. D., Graf, P., and Nitzan A. (1999). A lattice relaxation algorithm for 3D Poisson–Nernst–Planck theory with application to ion transport through the gramicidin A channel. *Biophys. J.* **76**, 642.

Nicholson, D. R. (1983). *Introduction to Plasma Theory*, Wiley.

Patera, A. T. (1984). A spectral element method for fluid dynamics: laminar flow in a channel expansion. *J. Comput. Phys.* **54**, 468.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1986). *Numerical Recipes*, Cambridge University Press.

Robinson, R. A., and Stokes, R. H. (1959). *Electrolyte Solutions*, Butterworths Scientific Publications.

Roux, B. (1997). Influence of the membrane potential on the free energy of an intrinsic protein. *Biophys. J.* **73**, 2980.

Roux, B. (1999). Statistical mechanical equilibrium theory of selective ion channels. *Biophys. J.* **77**, 139.

Selberherr, S. (1984). *Analysis and Simulation of Semiconductor Devices*, Springer-Verlag.

Strang, G., and Fix, G. J. (1973). *An Analysis of the Finite Element Method*, Prentice-Hall.

Zienkiewicz, O. C. (1971). *The Finite Element Method in Engineering Science*, 2nd ed., McGraw-Hill.