43

Computational Methods and Software for Bioelectric Field Problems

Christopher R. Johnson
University of Utah

43.1 Introduction

Computer modeling and simulation continue to become more important in the field of bioengineering. The reasons for this growing importance are manyfold. First, mathematical modeling has been shown to be a substantial tool for the investigation of complex biophysical phenomena. Second, since the level of complexity one can model parallels the existing hardware configurations, advances in computer architecture have made it feasible to apply the computational paradigm to complex biophysical systems. Hence, while biological complexity continues to outstrip the capabilities of even the largest computational systems, the computational methodology has taken hold in bioengineering and has been used successfully to suggest physiologically and clinically important scenarios and results.

This chapter provides an overview of numerical techniques that can be applied to a class of bioelectric field problems. Bioelectric field problems are found in a wide variety of biomedical applications, which range from single cells [59], to organs [62], up to models that incorporate partial to full human structures [44,45,50]. We describe some general modeling techniques that will be applicable, in part, to all the aforementioned applications. We focus our study on a class of bioelectric volume conductor problems that arise in electrocardiography (ECG) and electroencephalography (EEG).

We begin by stating the mathematical formulation for a bioelectric volume conductor, continue by describing the model construction process, and follow with sections on numerical solutions and computational considerations. We continue with a section on error analysis coupled with a brief introduction to adaptive methods. We conclude with a section on software.

43.2 Problem Formulation

As noted in Chapter 39, most bioelectric field problems can be formulated in terms of either the Poisson or the Laplace equation for electrical conduction. Since the Laplace equation is the homogeneous counterpart of the Poisson equation, we will develop the treatment for a general three-dimensional (3D) Poisson problem and discuss simplifications and special cases when necessary.

A *typical* bioelectric volume conductor can be posed as the following boundary value problem:

$$\nabla \cdot \sigma \nabla \Phi = -I_V \quad \text{in } \Omega, \tag{43.1}$$

where Φ is the electrostatic potential, σ is the electrical conductivity tensor, and I_V is the current per unit volume defined within the solution domain, Ω. The associated boundary conditions depend on what type of problem one wishes to solve. There are generally considered to be two different types of conductor problems: direct and inverse volume.

One type of problem deals with the interplay between the description of the bioelectric volume source currents and the resulting volume currents and volume and surface voltages. Here, the problem statement would be to solve Equation 43.1 for Φ with a known description of I_V and the Neumann boundary condition:

$$\sigma \nabla \Phi \cdot \mathbf{n} = 0 \quad \text{on } \Gamma_T, \tag{43.2}$$

which says that the normal component of the electric field is zero on the surface interfacing with air (here denoted by Γ_T). This problem can be used to solve two well-known problems in medicine, the direct EEG and the ECG volume conductor problems. In the direct EEG problem, one usually discretizes the brain and surrounding tissue and skull. One then assumes a description of the bioelectric current source within the brain (this usually takes the form of dipoles or multipoles) and calculates the field within the brain and on the surface of the scalp.

43.2.1 Example: Simulation of Focal Current Sources in the Brain

Figure 43.1 shows the simulation results from a patient-specific model of the head carried out with NeuroFEM (for source simulation) and SCIRun (for mesh generation and visualization). The mesh was composed of 179,643 nodes and 1,067,541 tetrahedral elements and the preliminary simulation was carried out with a dipole source in the right posterior region.

Similarly, in one version of the direct ECG problem, one utilizes descriptions of the current sources in the heart (either dipoles or membrane current source models such as the FitzHugh Nagumo and Beeler Reuter, among others) or defibrillation sources and calculates the currents and voltages within the heart and volume conductor of the chest and voltages on the surface of the torso.

43.2.2 Example: Simulation of Implantable Cardiac Defibrillators

The goal of these simulations was to calculate the electric potentials in the body, and especially in the fibrillating heart, that arise during a shock from an implantable cardiac defibrillator (ICD), over 90,000 of which are implanted annually in the United States alone. Of special interest was the use of such

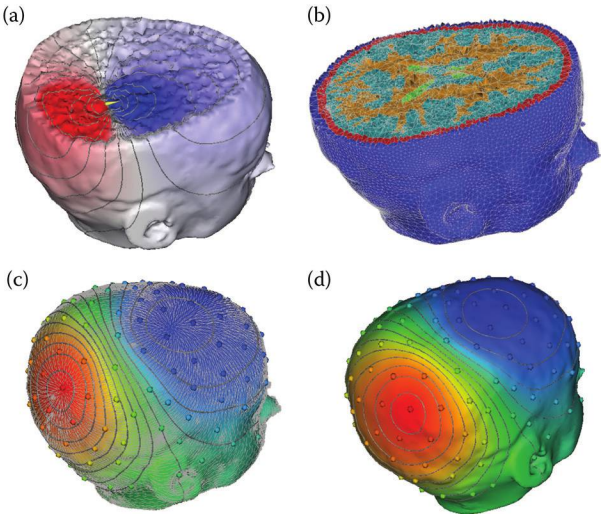

FIGURE 43.1 Illustration of the simulation of the electromagnetic field propagation in a patient-specific brain model. The figure shows a finite element method discretization of the Poisson equation with a patient-specific, five-compartment, geometrical model derived from a segmentation of brain MRI. The solid lines in the simulation images indicate isopotentials and the small white lines are electrical current streamlines.

devices in children, who are both much smaller in size than adults and almost uniformly have some form of anatomical abnormality that makes patient-specific modeling essential.

We have developed a complete pipeline for the patient-specific simulation of defibrillation fields from ICDs, starting from computed tomography (CT) or magnetic resonance imaging (MRI) image volumes and creating hexahedral meshes of the entire torso with heterogeneous mesh density to achieve acceptable computation times [48]. In these simulations, there was effectively a second modeling pipeline that executed each time the user selected a candidate set of locations for the device and the associated shock electrodes. For each such configuration, there was a customized version of the volume mesh that had to be generated and prepared for computation.

Figure 43.2 shows the steps required to implement the customized mesh for each new set of device and electrode locations. The user manipulated an interactive program implemented in SCIRun that allowed very flexible design and placement of the components of the device, an image of which is shown in the leftmost panel of the figure. Modules in SCIRun then carried out a refinement of the underlying hexahedral mesh so that the potentials applied by the device and electrodes were transferred with suitable spatial fidelity to the torso volume conductor (second panel). Then, additional modules in SCIRun computed the resulting electric field throughout the torso and visualized the results, also showing the details of the potentials at the heart and deriving from the simulations a defibrillation threshold value (last two panels of the figure). We have also carried out the initial validation of the complete system by comparing computed to measured defibrillation thresholds and obtained encouraging results [48].

The inverse problems associated with these direct problems involve estimating the current sources I_V within the volume conductor from measurements of voltages on the surface of either the head or the body. Thus, one would solve Equation 43.1 with the boundary conditions:

$$\Phi = \Phi_0 \quad \text{on } \Sigma \subseteq \Gamma_T \tag{43.3}$$

$$\sigma \nabla \Phi \cdot \mathbf{n} = 0 \quad \text{on } \Gamma_T. \tag{43.4}$$

The first is the Dirichlet condition, which says that one has a set of discrete measurements of the voltage of a subset of the outer surface. The second is the natural Neumann condition. While it does not look

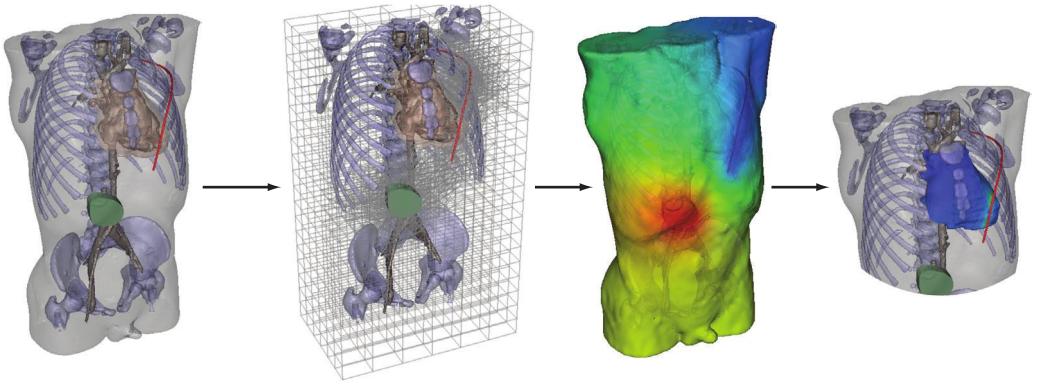

FIGURE 43.2 Pipeline for computing defibrillation potentials in children. The figures show the steps from the left to the right required to place electrodes and then compute and visualize the resulting cardiac potentials.

much different than the formulation of the direct problem, the inverse formulations are ill-posed. The bioelectric inverse problem in terms of primary current sources does not have a unique solution, and the solution does not depend continuously on the data. Thus, to obtain *useful* solutions, one must try to restrict the solution domain (i.e., number of physiologically plausible solutions) [29] for the former case, and apply the so-called *regularization* techniques to attempt to restore the continuity of the solution on the data in the latter case.

Another bioelectric direct/inverse formulation poses both the problems in terms of scalar values at the surfaces. For the EEG problem, one would take the surface of the brain (cortex) as one bounded surface and the surface of the scalp as the other surface. The direct problem would involve making measurements of voltage of the surface of the cortex at discrete locations and then calculating the voltages on the surface of the scalp. Similarly, for the ECG problem, voltages could be measured on the surface of the heart and used to calculate the voltages at the surface of the torso, as well as within the volume conductor of the thorax. To formulate the inverse problems, one uses measurements on the surface of the scalp (torso) to calculate the voltages on the surface of the cortex (heart). Here, we solve the Laplace equation instead of the Poisson equation because we are interested in the distributions of voltages on a surface instead of current sources within a volume. This leads to the following boundary value problem:

$$\nabla \cdot \sigma \nabla \Phi = 0 \quad \text{in } \Omega \tag{43.5}$$

$$\Phi = \Phi_0 \quad \text{on } \Sigma \subseteq \Gamma_T \tag{43.6}$$

$$\sigma \nabla \Phi \cdot \mathbf{n} = 0 \quad \text{on } \Gamma_T. \tag{43.7}$$

For this formulation, the solution to the inverse problem is unique [87]; however, there still exists the problem of continuity of the solution on the data. The linear algebra counterpart to the elliptic boundary value problem is often useful in discussing this problem of noncontinuity. The numerical solution to all elliptic boundary value problems (such as the Poisson and Laplace problems) can be formulated

in terms of a set of linear equations, $\mathbf{A\Phi} = \mathbf{b}$. For the solution of the Laplace equation, the system can be reformulated as

$$A\Phi_{in} = \Phi_{out}, \tag{43.8}$$

where Φ_{in} is the vector of the data on the inner surface bounding the solution domain (e.g., the electrostatic potentials on the scalp or heart), Φ_{out} is the vector of data that bounds the outer surface (e.g., the subset of voltage values on the surface of the cortex or torso), and A is the *transfer matrix* between Φ_{out} and Φ_{in}, which usually contains the geometry and physical properties (conductivities, dielectric constants, etc.) of the volume conductor. The direct problem is then simply (well) posed as solving Equation 43.8 for Φ_{out} given Φ_{in}. Likewise, the inverse problem is to determine Φ_{in} given Φ_{out}.

A characteristic of A for ill-posed problems is that it has a very large condition number. In other words, the ill-conditioned matrix A is very near to being singular. Briefly, the condition number is defined as $\kappa(A) = ||A|| \cdot ||A^{-1}||$ or the ratio of maximum to minimum singular values measured in the L_2 norm. The ideal problem conditioning occurs for orthogonal matrices that have $\kappa(A) \approx 1$, while an ill-conditioned matrix will have $\kappa(A) \gg 1$. When one inverts a matrix that has a very large condition number, the inversion process is unstable and is highly susceptible to errors. The condition of a matrix is relative. It is related to the precision level of computations and is a function of the size of the problem. For example, if the condition number exceeds a linear growth rate with respect to the size of the problem, the problem will become increasingly ill-conditioned. See Reference 27 for more about the condition number of matrices.

A number of techniques have arisen to deal with ill-posed inverse problems. These techniques include truncated singular value decomposition (TSVD), generalized singular value decomposition (GSVD), maximum entropy, and a number of generalized least squares schemes, including Twomey and Tikhonov regularization methods. Since this section is concerned more with the numerical techniques for approximating bioelectric field problems, the reader is referred to References 26, 32, 81, and 82 to further investigate the regularization of ill-posed problems. A particularly useful reference for discrete ill-posed problems is the MATLAB® package developed by Per Christian Hansen, which is available via his website [31].

43.3 Model Construction and Mesh Generation

Once we have stated or derived the mathematical equations that define the physics of the system, we must figure out how to solve these equations for the particular domain we are interested in. Most numerical methods for solving boundary value problems require that the continuous domain be broken up into discrete elements, the so-called *mesh* or *grid*, which one can use to approximate the governing equation(s) using the particular numerical technique (finite element, boundary element, finite difference, or multigrid) best suited to the problem.

Because of the complex geometries often associated with bioelectric field problems, the construction of the polygonal mesh can become one of the most time-consuming aspects of the modeling process. After deciding upon the particular approximation method to use (and the most appropriate type of element), we need to construct a mesh of the solution domain, which matches the number of degrees of freedom (DOF) of our fundamental element. For the sake of simplicity, we will assume that we will use linear elements, either tetrahedrons, which are usually used for modeling irregular 3D domains, or hexahedrons used for modeling regular, uniform domains.

There are several different strategies for discretizing the geometry into fundamental elements. For bioelectric field problems, two approaches to mesh generation have become standard: the *divide-and-conquer* (or subsequent subdivision) strategy and the so-called *Delaunay triangulation* strategy.

In using the divide-and-conquer strategy, one starts with a set of points that define the bounding surface(s) in three dimensions (contours in two dimensions). The volume (surface) is repeatedly

divided into smaller regions until a satisfactory discretization level has been achieved. Usually, the domain is broken up into eight-node cubic elements, which can then be subdivided into five (minimally) or six tetrahedral elements if so desired. This methodology has the advantage of being fairly easy to program; furthermore, commercial mesh generators exist for the divide-and-conquer method. For use in solving bioelectric field problems, its main disadvantage is that it allows elements to overlap interior boundaries. A single element may span two different conductive regions, for example, when part of an element represents muscle tissue (which could be anisotropic) and the other part of the element falls into a region representing fat tissue. It then becomes very difficult to assign unique conductivity parameters to each element and at the same time accurately represent the geometry.

A second method of mesh generation is the Delaunay triangulation strategy. Given a 3D set of points that define the boundaries and interior regions of the domain to be modeled, one tessellates the point cloud into an optimal mesh of tetrahedra. For bioelectric field problems, the advantages and disadvantages tend to be exactly contrary to those arising from the divide-and-conquer strategy. The primary advantage is that one can create the mesh to fit any predefined geometry, including subsurfaces, by starting with points that define all the necessary surfaces and subsurfaces and then adding additional interior points to minimize the aspect ratio. For tetrahedra, the aspect ratio can be defined as $4\sqrt{(3/2)}(\rho_k/h_k)$, where ρ_k denotes the diameter of the sphere circumscribed about the tetrahedron, and h_k is the maximum distance between two vertices. These formulations yield a value of 1 for an equilateral tetrahedron and a value of 0 for a degenerate (flat) element [7]. The closer to the value of 1, the better. The Delaunay criterion is a method for minimizing the occurrence of obtuse angles in the mesh, yielding elements that have aspect ratios as close to 1 as possible, given the available point set. While the ideas behind Delaunay triangulation are straightforward, the programming is nontrivial and is the primary drawback to this method. Fortunately, there exist several public domain software packages for two- and three-dimensional mesh generation (see Equation 43.6). For more information on mesh generation and various aspects of biomedical modeling, see References 4, 12, 15, 16, 19, 25, 36, 53–55, 57, 58, 66, 71, 75, 76, 79, and 80.

43.3.1 Example: Modeling of Focal Current Sources in the Brain

Figure 43.3 contains the geometric model results from a 15-year-old pediatric patient suffering from epileptic seizures. The segmentations came from a semiautomated tissue classification algorithm developed by Warfield et al. [86], followed by extensive manual inspection and hand editing of mislabeled pixels using Seg3D. The meshing component of the pipeline was implemented in BioMesh3D, a new program that incorporates separate surface fitting and mesh generation programs (e.g., TetGen) in a scripting environment [15,76]. The triangle mesh quality in Panel (a) is excellent, a result of the distributed particle method we have developed [57,58].

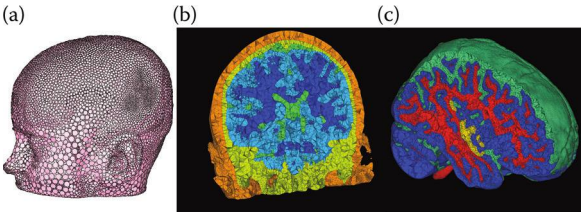

FIGURE 43.3 Example of meshing of the head in a pediatric epilepsy patient. Panel (a) shows the particle distribution over the head surface, and highlights through the variation in particle size the adaptivity of the particles over the skin. Panel (b) shows the associated tetrahedral mesh and panel (c) shows another, higher-resolution view of the mesh, highlighting the cortex and the cerebrospinal fluid.

43.4 Numerical Methods

Because of the geometrical complexity of, and numerous inhomogeneities inherent in, anatomical structures in physiological models, solutions of bioelectric field problems are usually tractable (except in the most simplified of models) only when one employs a numerical approximation method such as the finite difference (FD), the finite element (FE), boundary element (BE), or the multigrid (MG) method to solve the governing field equation(s).

43.4.1 Approximation Techniques: The Galerkin Method

The problem posed in Equation 43.1 can be solved using any of the aforementioned approximation schemes. One technique that addresses three of the previously mentioned techniques (FD, FE, and BE) can be derived by the Galerkin method. The Galerkin method is one of the most widely used methods for discretizing elliptic boundary value problems such as Equation 43.1 and for treating the spatial portion of time-dependent parabolic problems, which are common in models of cardiac wave propagation. While the Galerkin technique is not essential to the application of any of the techniques, it provides for a unifying bridge between the various numerical methods. To express our problem in a Galerkin form, we begin by rewriting Equation 43.1 as

$$A\Phi = -I_v, \tag{43.9}$$

where A is the differential operator, $A = \nabla \cdot (\sigma \nabla)$. An equivalent statement of Equation 43.9 is, find Φ such that $(A\Phi + I_v, \bar{\Phi}) = 0$. Here, $\bar{\Phi}$ is an arbitrary *test function*, which can be thought of physically as a virtual potential field, and the notation $(\phi_1, \phi_2) \equiv \int_\Omega \phi_1 \phi_2 \, d\Omega$ denotes the inner product in $L_2(\Omega)$. Applying Green's theorem, we can equivalently write

$$(\sigma \nabla \Phi, \nabla \bar{\Phi}) - \left\langle \frac{\partial \Phi}{\partial n}, \bar{\Phi} \right\rangle = -(I_v, \bar{\Phi}), \tag{43.10}$$

where the notation $\langle \phi_1, \phi_2 \rangle \equiv \int_S \phi_1 \phi_2 \, dS$ denotes the inner product on the boundary S. When the Dirichlet, $\Phi = \Phi_0$, and Neumann, $\sigma \nabla \Phi \cdot \mathbf{n} = 0$, boundary conditions are specified on S, we obtain the *weak form* of Equation 43.1:

$$(\sigma \nabla \Phi, \nabla \bar{\Phi}) = -(I_v, \bar{\Phi}). \tag{43.11}$$

It is understood that this equation must hold for all test functions, $\bar{\Phi}$, which must vanish at the boundaries where $\Phi = \Phi_0$. The Galerkin approximation ϕ to the weak form solution Φ in Equation 43.11 can be expressed as

$$\phi(\mathbf{x}) = \sum_{i=0}^{N} \phi_i \psi_i(\mathbf{x}). \tag{43.12}$$

The trial functions ψ_i, $i = 0,1,\ldots, N$ form a basis for an $N+1$-dimensional space S. We define the *Galerkin approximation* to be that element $\phi \in S$, which satisfies

$$(\sigma \nabla \phi, \nabla \psi_j) = -(I_v, \psi_j) \quad (\forall\ \psi_j \in S). \tag{43.13}$$

Since our differential operator A is positive definite and self-adjoint (i.e., $(A\Phi, \Phi) \geq \alpha\ (\Phi, \Phi) > 0$ for some nonzero positive constant α and $(A\Phi, \bar{\Phi}) = (\Phi, A\bar{\Phi})$, respectively), then we can define a space E

with an inner product defined as $(\Phi,\bar{\Phi})_E = (A\Phi,\bar{\Phi}) \equiv a(\Phi,\bar{\Phi})$ and norm (the so-called energy norm) equal to

$$(\Phi,\bar{\Phi})_E = \left\{\int_\Omega (\nabla\Phi)^2 d\Omega\right\}^{1/2} = (\Phi,\Phi)_E^{1/2}. \tag{43.14}$$

The solution Φ of Equation 43.9 satisfies

$$(A\Phi, \psi_i) = -(I_v, \psi_i) \quad (\forall\ \psi_i \in S) \tag{43.15}$$

and the approximate Galerkin solution obtained by solving Equation 43.13 satisfies

$$(A\phi, \psi_i) = -(I_v, \psi_i) \quad (\forall\ \psi_i \in S). \tag{43.16}$$

Subtracting Equation 43.15 from Equation 43.16 yields

$$(A(\phi - \Phi), \psi_i) = (\phi - \Phi, \psi_i)_E = 0 \quad (\forall\ \psi_i \in S). \tag{43.17}$$

The difference $\phi - \Phi$ denotes the error between the solution in the infinite dimensional space V and the $N+1$-dimensional space S. Equation 43.17 states that the error is orthogonal to all basis functions spanning the space of possible Galerkin solutions. Consequently, the error is orthogonal to all elements in S and must therefore be the minimum error. Thus, the Galerkin approximation is an orthogonal projection of the true solution Φ onto the given finite dimensional space of possible approximate solutions. Therefore, the Galerkin approximation is the best approximation in the energy space E. Since the operator is positive definite, the approximate solution is unique. Assume for a moment there are two solutions, ϕ_1 and ϕ_2, satisfying

$$(A\phi_1, \psi_i) = -(I_v, \psi_i)\ (A\phi_2, \psi_i) = -(I_v, \psi_i) \quad (\forall\ \psi_i \in S), \tag{43.18}$$

respectively. Then, the difference yields

$$(A(\phi_1 - \phi_2), \psi_i) = 0 \quad (\forall\ \psi_i \in S). \tag{43.19}$$

The function arising from subtracting one member from another member in S also belongs in S; hence, the difference function can be expressed by the set of A orthogonal basis functions spanning S:

$$\sum_{j=0}^{N} \Delta\phi_j (A(\psi_j, \psi_i)) = 0 \quad (\forall \psi_i \in). \tag{43.20}$$

When $i \neq j$, the terms vanish due to the basis functions being orthogonal with respect to A. Since A is positive definite:

$$(A\Phi_i, \Phi_i) > 0 \quad i = 0, \ldots, N. \tag{43.21}$$

Thus, $\Delta\phi_i = 0$, $i = 0,\ldots,N$, and by virtue of Equation 43.20, $\delta\phi = 0$, such that $\phi_1 = \phi_2$. The identity contradicts the assumption of two distinct Galerkin solutions. This proves the solution is unique [33].

43.4.2 Finite Difference Method

Perhaps the most traditional way to solve Equation 43.1 utilizes the FD approach by discretizing the solution domain Ω using a grid of uniform hexahedral elements. The coordinates of a typical grid point

are $x = lh$, $y = mh$, $z = nh$ (l,m,n = integers), and the value of $\Phi(x, y, z)$ at a grid point is denoted by $\Phi_{l,m,n}$. Taylor's theorem can then be utilized to provide the difference equations. For example

$$\Phi_{l+1,m,n} = \left(\Phi + h\frac{\partial \Phi}{\partial x} + \frac{1}{2}h^2 \frac{\partial^2 \Phi}{\partial x^2} + \frac{1}{6}h^3 \frac{\partial^3 \Phi}{\partial x^3} + \cdots \right)_{l,m,n} \tag{43.22}$$

with similar equations for $\Phi_{l-1,m,n}$, $\Phi_{l,m+1,n}$, $\Phi_{l,m-1,n}$,…. The FD representation of Equation 43.1 is

$$\frac{\Phi_{l+1,m,n} - 2\Phi_{l,m,n} + \Phi_{l-1,m,n}}{h^2} + \frac{\Phi_{l,m+1,n} - 2\Phi_{l,m,n} + \Phi_{l,m-1,n}}{h^2} + \frac{\Phi_{l,m,n+1} - 2\Phi_{l,m,n} + \Phi_{l,m,n-1}}{h^2} = -I_{l,m,n}(v) \tag{43.23}$$

or, equivalently

$$\Phi_{l+1,m,n} + \Phi_{l-1,m,n} + \Phi_{l,m+1,n} + \Phi_{l,m-1,n} + \Phi_{l,m,n+1} + \Phi_{l,m,n-1} - 6\Phi_{l,m,n} = -h^2 I_{l,m,n}(v). \tag{43.24}$$

If we define the vector $\mathbf{\Phi}$ to be $[\Phi_{1,1,1}\ldots\Phi_{1,1,N-1};\ldots\Phi_{1,N-1},1\ldots\Phi_{N-1,N-1,N-1}]^T$ to designate the $(N-1)^3$ unknown grid values, and pull out all the known information from Equation 43.24, we can reformulate Equation 43.1 by its FD approximation in the form of the matrix equation $A\mathbf{\Phi} = \mathbf{b}$, where $\mathbf{b}$ is a vector that contains the sources and modifications due to the Dirichlet boundary condition.

Unlike the traditional Taylor's series expansion method, the Galerkin approach utilizes basis functions, such as linear piecewise polynomials, to approximate the true solution. For example, the Galerkin approximation to the sample problem Equation 43.1 would require evaluating Equation 43.13 for the specific grid formation and specific choice of basis function:

$$\int_{\Omega} \left(\sigma_x \frac{\partial \phi}{\partial x}\frac{\partial \psi_i}{\partial x} + \sigma_y \frac{\partial \phi}{\partial y}\frac{\partial \psi_i}{\partial y} + \sigma_z \frac{\partial \phi}{\partial z}\frac{\partial \psi_i}{\partial z} \right) d\Omega = -\int_{\Omega} I_v \psi_i \, d\Omega. \tag{43.25}$$

Difference quotients are then used to approximate the derivatives in Equation 43.25. We note that if linear basis functions are utilized in Equation 43.25, one obtains a formulation that corresponds exactly with the standard FD operator. Regardless of the difference scheme or order of basis function, the approximation results in a linear system of equations of the form $A\mathbf{\Phi} = \mathbf{b}$, subject to the appropriate boundary conditions.

43.4.3 Finite Element Method

As we have seen above, in the classical numerical treatment for partial differential equations—the FD method—the solution domain is approximated by a grid of uniformly spaced nodes. At each node, the governing differential equation is approximated by an algebraic expression that references adjacent grid points. A system of equations is obtained by evaluating the previous algebraic approximations for each node in the domain. Finally, the system is solved for each value of the dependent variable at each node. In the FE method, the solution domain can be discretized into a number of uniform or nonuniform FEs that are connected via nodes. The change of the dependent variable with regard to location is approximated within each element by an interpolation function. The interpolation function is defined relative to the values of the variable at the nodes associated with each element. The original boundary value problem is then replaced with an equivalent integral formulation (such as Equation 43.13). The interpolation functions are then substituted into the integral equation, integrated, and combined with the results from all other elements in the solution domain. The results of this procedure can be

reformulated into a matrix equation of the form $A\Phi = \mathbf{b}$, which is subsequently solved for the unknown variable [3,36].

The formulation of the FE approximation starts with the Galerkin approximation, $(\sigma\nabla\Phi, \nabla\bar{\Phi}) = -(I_v, \bar{\Phi})$, where $\bar{\Phi}$ is our test function. We now use the FE method to turn the continuous problems into a discrete formulation. First, we discretize the solution domain, $\Omega = \cup_{e=1}^{E} \Omega_e$, and define a finite dimensional subspace, $V_h \subset V = \{\bar{\Phi}: \bar{\Phi}$ is continuous on Ω, $\nabla\bar{\Phi}$ is piecewise continuous on $\Omega\}$. One usually defines parameters of the function $\bar{\Phi} \in V_h$ at node points $\alpha_i = \bar{\Phi}(x_i)$, $i = 0,1,\ldots,N$. If we now define the basis functions, $\psi_i \in V_h$, as linear continuous piecewise functions that take the value 1 at node points and zero at other node points, then we can represent the function $\bar{\Phi} \in V_h$ as

$$\bar{\Phi}(x) = \sum_{i=0}^{N} \alpha_i \Psi_i(x), \tag{43.26}$$

such that each $\bar{\Phi} \in V_h$ can be written in a unique way as a linear combination of the basis functions $\Psi_i \in V_h$. Now, the FE approximation of the original boundary value problem can be stated as

$$\text{Find } \Phi_h \in V_h \quad \text{such that } (\sigma\nabla\Phi_h, \nabla\bar{\Phi}) = -(I_v, \bar{\Phi}). \tag{43.27}$$

Furthermore, if $\Phi_h \in V_h$ satisfies Equation 43.27, then we have $(\sigma\nabla\Phi_h, \nabla\Psi_i) = -(I_v, \Psi_i)$ [42,47]. Finally, since Φ_h itself can be expressed as the linear combination

$$\Phi_h = \sum_{i=0}^{N} \xi_i \Psi_i(x) \quad \xi_i = \Phi_h(x_i), \tag{43.28}$$

we can then write Equation 43.27 as

$$\sum_{i=0}^{N} \xi_i (\sigma_{ij}\nabla\Psi_i, \nabla\Psi_j) = -(I_v, \Psi_j) \quad j = 0,\ldots,N, \tag{43.29}$$

subject to the Dirichlet boundary condition. Then the FE approximation of Equation 43.1 can equivalently be expressed as a system of N equations with N unknowns $\xi_i,\ldots,\xi_N$ (e.g., the electrostatic potentials). In matrix form, the above system can be written as $A\xi = b$, where $A = (a_{ij})$ is called the global stiffness matrix and has elements $(a_{ij}) = (\sigma_{ij}\nabla\Psi_i, \nabla\Psi_j)$, while $b_i = -(I_v, \Psi_i)$ and is usually termed the load vector.

For volume conductor problems, A contains all the geometry and conductivity information of the model. The matrix A is symmetric and positive definite; thus, it is nonsingular and has a unique solution. Because the basis function differs from zero for only a few intervals, A is sparse (only a few of its entries are nonzero).

43.4.3.1 Application of the FE Method for 3D Domains

We now illustrate the concepts of the FE method by considering the solution of Equation 43.1 using linear 3D elements. We start with a 3D domain Ω that represents the geometry of our volume conductor and break it up into discrete elements to form a finite dimensional subspace, Ω_h. For 3D domains, we have the choice of representing our function as either tetrahedra

$$\tilde{\Phi} = \alpha_1 + \alpha_2 x + \alpha_3 y + \alpha_4 z, \tag{43.30}$$

or hexahedra

$$\tilde{\Phi} = \alpha_1 + \alpha_2 x + \alpha_3 y + \alpha_4 z + \alpha_5 xy + \alpha_6 yz + \alpha_7 xz + \alpha_8 xyz. \tag{43.31}$$

Because of space limitations, we restrict our development to tetrahedra, knowing that it is easy to modify our formulae for hexahedra. We take out a specific tetrahedra from our finite dimensional subspace and apply the previous formulations for the four vertices

$$\begin{Bmatrix} \tilde{\Phi}_1 \\ \tilde{\Phi}_2 \\ \tilde{\Phi}_3 \\ \tilde{\Phi}_4 \end{Bmatrix} = \begin{pmatrix} 1 & x_1 & y_1 & z_1 \\ 1 & x_2 & y_2 & z_2 \\ 1 & x_3 & y_3 & z_3 \\ 1 & x_4 & y_4 & z_4 \end{pmatrix} \begin{Bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{Bmatrix}, \tag{43.32}$$

or

$$\tilde{\Phi}_\mathbf{i} = \mathbf{C}\alpha, \tag{43.33}$$

which define the coordinate vertices, and

$$\alpha = \mathbf{C}^{-1}\tilde{\Phi}_\mathbf{i}, \tag{43.34}$$

which defines the coefficients. From Equations 43.30 and 43.34, we can express $\tilde{\Phi}$ at any point within the tetrahedra

$$\tilde{\Phi} = [1, x, y, z]\alpha = \mathbf{S}\alpha = \mathbf{S}\mathbf{C}^{-1}\tilde{\Phi}_\mathbf{i} \tag{43.35}$$

or, most succinctly

$$\tilde{\Phi} = \sum_i N_i \tilde{\Phi}_i. \tag{43.36}$$

$\tilde{\Phi}_i$ is the solution value at node i, and $\mathbf{N} = \mathbf{S}\mathbf{C}^{-1}$ is the local *shape function* or *basis function*. This can be expressed in a variety of ways in the literature (depending, usually, on whether you are reading engineering or mathematical treatments of FE analysis):

$$\Phi_j(N_i) = N_i(x, y, z) = f_i(x, y, z) \equiv \frac{a_i + b_i x + c_i y + d_i z}{6V}, \tag{43.37}$$

where

$$6V = \begin{vmatrix} 1 & x_1 & y_1 & z_1 \\ 1 & x_2 & y_2 & z_2 \\ 1 & x_3 & y_3 & z_3 \\ 1 & x_3 & y_3 & z_4 \end{vmatrix} \tag{43.38}$$

defines the volume of the tetrahedra, V.

Now that we have a suitable set of basis functions, we can find the FE approximation to our 3D problem. Our original problem can be formulated as

$$a(u, v) = (I_v, v) \quad \forall\ v \in \Omega, \tag{43.39}$$

where

$$a(u,v) = \int_{\Omega} \nabla u \cdot \nabla v \, d\Omega \tag{43.40}$$

and

$$(I_v,v) = \int_{\Omega} I_v \cdot v \, d\Omega. \tag{43.41}$$

The FE approximation to the original boundary value problem is

$$a(u_h, v) = (I_v, v) \quad \forall\ v \in \Omega_h, \tag{43.42}$$

which has the equivalent form

$$\sum_{i=1}^{N} \xi_i a(\Phi_i,\Phi_j) = (I_v,\Phi_j), \tag{43.43}$$

where

$$a(\Phi_i, \Phi_j) = a(\Phi_i(N_j), \Phi_j(N_i)), \tag{43.44}$$

which can be expressed by the matrix and vector elements

$$a_{ij} = \int_{\Omega_E} \left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y} + \frac{\partial N_i}{\partial z}\frac{\partial N_j}{\partial z} \right) d\Omega \tag{43.45}$$

and

$$I_i = \int_{\Omega_E} N_i I_v \, d\Omega. \tag{43.46}$$

Fortunately, the above quantities are easy to evaluate for linear tetrahedra. As a matter of fact, there are closed form solutions for the matrix elements (a_{ij}):

$$\int_{\Omega_h} N_1^a N_2^b N_3^c N_4^d \, d\Omega = 6V \frac{a!b!c!d!}{(a+b+c+d+3)!}. \tag{43.47}$$

Therefore

$$a_{ij} = \int_{\Omega_E} \frac{b_i b_j + c_i c_j + d_i d_j}{6V^2} \, d\Omega = \frac{b_i b_j + c_i c_j + d_i d_j}{6V}, \tag{43.48}$$

and, for the right-hand side (RHS), we have, assuming constant sources,

$$I_i = \int_{\Omega_E} \frac{a_i + b_i x + c_i y + d_i z}{6V} I_v \, d\Omega = \frac{V I_v}{4}. \tag{43.49}$$

which have the compact forms

$$a_{ij}^{(n)} = \frac{1}{6V}(b_i^{(n)}b_j^{(n)} + c_i^{(n)}c_j^{(n)} + d_i^{(n)}d_j^{(n)}) \tag{43.50}$$

and

$$I_i^{(n)} = \frac{VI_v}{4} \quad \text{for constant sources.} \tag{43.51}$$

Now, we add up all the contributions from each element into a global matrix and global vector.

$$\sum_{n=1}^{Nel}(a_{ij}^{(n)})(\xi_i) = (I_i^{(n)}), \tag{43.52}$$

where *Nel* is equal to the total number of elements in the discretized solution domain and *i* represents the node numbers (vertices). This yields a linear system of equations of the form $\mathbf{A\Phi} = \mathbf{b}$, where $\mathbf{\Phi}$ is our solution vector of voltages, $\mathbf{A}$ represents the geometry and conductivity of our volume conductor, and $\mathbf{b}$ represents the contributions from the current sources and boundary conditions.

For the FD method, it turns out that the Dirichlet boundary condition is easy to apply while the Neumann condition takes a little extra effort. For the FE method, it is just the opposite. The Neumann boundary condition

$$\nabla\Phi \cdot \mathbf{n} = 0 \tag{43.53}$$

is satisfied automatically within the Galerkin and variational formulations. This can be seen by using Green's divergence theorem

$$\int_\Omega \nabla \cdot \mathbf{A}\, dx = \int_\Gamma \mathbf{A} \cdot \mathbf{n}\, dS, \tag{43.54}$$

and applying it to the left-hand side of the Galerkin FE formulation:

$$\begin{aligned}\int_\Omega \nabla v \cdot \nabla w\, d\Omega &\equiv \int_\Omega \left[\frac{\partial v}{\partial x_1}\frac{\partial w}{\partial x_1} + \frac{\partial v}{\partial x_2}\frac{\partial w}{\partial x_2}\right] d\Omega \\ &= \int_\Gamma \left[v\frac{\partial w}{\partial x_1}n_1 + v\frac{\partial w}{\partial x_2}n_2\right] dS - \int_\Omega v\left[\frac{\partial^2 w}{\partial x_1^2} + \frac{\partial^2 w}{\partial x_2^2}\right] d\Omega \\ &= \int_\Gamma v\frac{\partial w}{\partial n}dS - \int_\Omega v\nabla^2 w\, d\Omega.\end{aligned} \tag{43.55}$$

If we multiply our original differential equation, $\nabla^2\Phi = -I_v$, by an arbitrary test function and integrate, we obtain

$$(I_v, v) = -\int_\Omega (\nabla^2\Phi)v\, d\Omega = -\int_\Gamma \frac{\partial \Phi}{\partial n}\, v\, dS + \int_\Omega \nabla\Phi \cdot \nabla v\, d\Omega = a(\Phi, v), \tag{43.56}$$

where the boundary integral term, $\partial\Phi/\partial n$ vanishes and we obtain the standard Galerkin FE formulation.

To apply the Dirichlet condition, we have to work a bit harder. To apply the Dirichlet boundary condition directly, one usually modifies the (a_{ij}) matrix and b_i vector such that one can use standard linear system solvers. This is accomplished by implementing the following steps.

Assuming we know the ith value of u_i

1. Subtract from the ith member of the RHS the product of a_{ij} and the known value of Φ_i (call it $\bar{\Phi}_i$); this yields the new RHS, $\hat{b}_i = b_i - a_{ij}\bar{\Phi}_j$.
2. Zero the ith row and column of A: $\hat{a}_{ij} = \hat{a}_{ji} = 0$.
3. Assign $\hat{a}_{ii} = 1$.
4. Set the jth member of the RHS equal to $\bar{\Phi}_i$.
5. Continue for each Dirichlet condition.
6. Solve the augmented system, $\hat{A}\Phi = \widehat{b_v}$.

43.4.4 Boundary Element Method

For bioelectric field problems with isotropic domains (and few inhomogeneities), another technique, called the BE method, may be utilized. This technique utilizes information only upon the boundaries of interest, and thus reduces the dimension of any field problem by one. For differential operators, the response at any given point to sources and boundary conditions depends only on the response at neighboring points. The FD and FE methods approximate differential operators defined on subregions (volume elements) in the domain; hence, direct mutual influence (connectivity) exists only between neighboring elements, and the coefficient matrices generated by these methods have relatively few nonzero coefficients in any given matrix row. As is demonstrated by Maxwell's laws [39], equations in differential forms can often be replaced by equations in integral forms, for example, the potential distribution in a domain is uniquely defined by the volume sources and the potential and current density on the boundary. The BE method utilizes this fact by transforming the differential operator defined in the domain to integral operators defined on the boundary. In the BE method [6,13,40], only the boundary is discretized; hence, the mesh generation is considerably simpler for this method than for the volume methods. Boundary solutions are obtained directly by solving the set of linear equations; however, potentials and gradients in the domain can be evaluated only after the boundary solutions have been obtained. As this method has a rich history in bioelectric field problems, the reader is referred to some of the classic references for further information regarding the application of the BE method to bioelectric field problems [5,30,67,69].

43.4.5 Solution Methods and Computational Considerations

The application of each of the previous approximation methods to Equation 43.1 yields a system of linear equations of the form $\mathbf{A\Phi} = \mathbf{b}$, which must be solved to obtain the final solution. There are a plethora of available techniques for the solutions of such systems. The solution techniques can be broadly categorized as *direct* and *iterative* solvers. Direct solvers include Gaussian elimination and lower-upper (LU) decomposition, while iterative methods include Jacobi, Gauss–Seidel, successive overrelaxation (SOR), and conjugate gradient (CG) methods, among others. The choice of the particular solution method is highly dependent upon the approximation technique employed to obtain the linear system, upon the size of the resulting system, and upon accessible computational resources. For example, the linear system resulting from the application of the FD or FE method will yield a matrix **A** that is symmetric, positive definite, and sparse. The matrix resulting from the FD method will have a specific band-diagonal structure that is dependent on the order of difference equations one uses to approximate the governing equation. The matrix resulting from the FE method will be exceedingly sparse so that only a few of the off diagonal elements will be nonzero. The application of the BE method, on the other hand, will yield a matrix **A** that is dense and nonsymmetric and thus requires a different choice of solver.

The choice of the optimal solver is further complicated by the size of the system versus access to computational resources. Sequential direct methods are usually confined to single workstations and thus the size of the system should fit in memory for optimal performance. Sequential iterative methods can be employed when the size of the system exceeds the memory of the machine; however, one pays a price in terms of performance as direct methods are usually much faster than iterative methods. In many cases, the size of the system exceeds the computational capability of a single workstation and one must resort to the use of clusters of workstations and/or parallel computers.

While new and improved methods continue to appear in the numerical analysis literature, the author's studies comparing various solution techniques for direct and inverse bioelectric field problems have resulted in the conclusion that the preconditioned CG methods and MG methods are the best overall performers for volume conductor problems computed on single workstations. Specifically, the incomplete Choleski conjugate gradient (ICCG) method works well for the FE method* and the preconditioned biconjugate gradient (BCG) methods are often utilized for BE methods. When clusters of workstations and/or parallel architectures are considered, the choice is less clear. For use with some high-performance architectures that contain large amounts of memory, parallel direct methods such as LU decomposition become attractive; however, preconditioned CG methods still perform well.

A discussion of parallel computing methods for the solution of biomedical field problems could fill an entire text. Thus, the reader is directed to the following references on parallel scientific computing [18,23,28].

43.4.6 Comparison of Methods

Since we do not have space to provide a detailed, quantitative description of each of the previously mentioned methods, we give an abbreviated summary of the applicability of each method in solving different types of bioelectric field problems.

As outlined above, the FD, FE, and BE methods can all be used to approximate the boundary value problems that arise in biomedical research problems. The choice depends on the nature of the problem. The FE and FD methods are similar in that the entire solution domain must be discretized, while with the BE method, only the bounding surfaces must be discretized. For regular domains, the FD method is generally the easiest method to code and implement, but the FD method usually requires special modifications to define irregular boundaries, abrupt changes in material properties, and complex boundary conditions. While typically more difficult to implement, the BE and FE methods are preferred for problems with irregular, inhomogeneous domains and mixed boundary conditions. The FE method is superior to the BE method for representing nonlinearity and true anisotropy, while the BE method is superior to the FE method for problems where only the boundary solution is of interest or where solutions are wanted in a set of highly irregularly spaced points in the domain. Because the computational mesh is simpler for the BE method than for the FE method, the BE program requires less bookkeeping than an FE program. For this reason, BE programs are often considered easier to develop than FE programs; however, the difficulties associated with singular integrals in the BE method are often highly underestimated. In general, the FE method is preferred for problems where the domain is highly heterogeneous, whereas the BE method is preferred for highly homogeneous domains.

43.5 Adaptive Methods

Thus far, we have discussed how one formulates the problem, discretizes the geometry, and finds an approximate solution. We are now faced with answering the difficult question pertaining to the accuracy

* This is specifically for the FE method applied to elliptic problems. Such problems yield a matrix that is symmetric and positive definite. The Choleski decomposition only exists for symmetric, positive-definite matrices.

of our solution. Without reference to experimental data, how can we judge the validity of our solutions? To give yourself an intuitive feel for the problem (and possible solution), consider the approximation of a two-dimensional region discretized into triangular elements. We will apply the FE method to solve the Laplace equation in the region.

First, consider the approximation of the potential field $\Phi(x, y)$ by a two-dimensional Taylor's series expansion about a point (x, y):

$$\Phi(x+h, y+k) = \Phi(x,y) + \left[h\frac{\partial\Phi(x,y)}{\partial x} + k\frac{\partial\Phi(x,y)}{\partial y}\right] + \frac{1}{2!}\left[h^2\frac{\partial^2\Phi(x,y)}{\partial^2 x} + 2hk\frac{\partial^2\Phi(x,y)}{\partial x\partial y} + k^2\frac{\partial^2\Phi(x,y)}{\partial^2 y}\right] + \cdots \tag{43.57}$$

where h and k are the maximum x and y distances within an element. Using the first two terms (up to first-order terms) in the above Taylor's expansion, we can obtain the standard linear interpolation function for a triangle:

$$\frac{\partial\Phi(x_i,y_i)}{\partial x} = \frac{1}{2A}[\Phi_i(y_j - y_m) + \Phi_m(y_i - y_j) + \Phi_j(y_m - y_i)], \tag{43.58}$$

where A is the area of the triangle. Likewise, one could calculate the interpolant for the other two nodes and discover that

$$\frac{\partial\Phi(x_i,y_i)}{\partial x} = \frac{\partial\Phi(x_j,y_j)}{\partial x} = \frac{\partial\Phi(x_m,y_m)}{\partial x} \tag{43.59}$$

is constant over the triangle (and thus so is the gradient in y as well). Thus, we can derive the standard linear interpolation formulas on a triangle that represent the first two terms of the Taylor's series expansion. This means that the error due to discretization (from using linear elements) is proportional to the third term of the Taylor's expansion:

$$\epsilon \approx \frac{1}{2!}\left[h^2\frac{\partial^2\Phi(x,y)}{\partial^2 x} + 2hk\frac{\partial^2\Phi(x,y)}{\partial x\partial y} + k^2\frac{\partial^2\Phi(x,y)}{\partial^2 y}\right], \tag{43.60}$$

where Φ is the exact solution. We can conjecture, then, that the error due to discretization for first-order linear elements is proportional to the second derivative. If Φ is a linear function over the element, then the first derivative is a constant and the second derivative is zero and there is no error due to discretization. This implies that the gradient must be constant over each element. If the function is not linear, or the gradient is not constant over an element, the second derivative will not be zero and is proportional to the error incurred due to "improper" discretization. Examining Equation 43.60, we can easily see that one way to decrease the error is to decrease the size of h and k. As h and k go to zero, the error tends to zero as well. Thus, decreasing the mesh size in places of high errors due to high gradients decreases the error. As an aside, we note that if one divides Equation 43.9 by hk, one can also express the error in terms of the elemental aspect ratio h/k, which is a measure of the relative shape of the element. It is easy to see that one must be careful to maintain an aspect ratio as close to unity as possible.

The problem with the preceding heuristic argument is that one has to know the exact solution *a priori* before one can estimate the error. This is certainly a drawback considering we are trying to accurately approximate Φ.

43.5.1 Convergence of a Sequence of Approximate Solutions

Let us try to quantify our error a bit further. When we consider the preceding example, it seems to make sense that if we increase the number of DOF we used to approximate our function, the accuracy must approach the true solution. That is, we would hope that the sequence of approximate solutions will *converge* to the exact solution as the number of DOF increases indefinitely:

$$\Phi(x) - \tilde{\Phi}_n(x) \to 0 \quad \text{as } N \to \infty. \tag{43.61}$$

This is a statement of *pointwise convergence*. It describes the approximate solution as approaching arbitrarily close to the exact solution at each point in the domain as the number of DOF increases.

Measures of convergence often depend on how the *closeness* of measuring the distance between functions is defined. Another common description of measuring convergence is *uniform convergence*, which requires that the maximum value of $\Phi(x) - \tilde{\Phi}_n(x)$ in the domain vanish as $N \to \infty$. This is stronger than pointwise convergence as it requires a uniform rate of convergence at every point in the domain. Two other commonly used measures are *convergence in energy* and *convergence in mean*, which involve measuring an *average* of a function of the pointwise error over the domain [14].

In general, proving pointwise convergence is very difficult except in the simplest cases, while proving the convergence of an averaged value, such as energy, is often easier. Of course, scientists and engineers are often much more interested in assuring that their answers are accurate in a pointwise sense than in an energy sense because they typically want to know values of the solution $\Phi(x)$ and gradients $\nabla\Phi(x)$ at specific places.

One intermediate form of convergence is called the *Cauchy convergence*. Here, we require the sequences of two different approximate solutions to approach arbitrarily close to each other:

$$\Phi_m(x) - \tilde{\Phi}_n(x) \to 0 \quad \text{as } M,N \to \infty. \tag{43.62}$$

While the pointwise convergence expression would imply the previous equation, it is important to note that the Cauchy convergence does not imply pointwise convergence, as the functions could converge to an answer other than the true solution.

While we cannot be assured of pointwise convergence of these functions for all but of the simplest cases, there do exist theorems that ensure that a sequence of approximate solutions must converge to the exact solution (assuming no computational errors) if the basis functions satisfy certain conditions. The theorems can only ensure convergence in an average sense over the entire domain but it is usually the case that if the solution converges in an average sense (energy, etc.), then it will converge in the pointwise sense as well.

43.5.2 Energy Norms

The error in energy, measured by the *energy norm*, is defined in general as [88–90]

$$e = \left\{ \int_{\Omega} e^T L e \, d\Omega \right\}^{1/2}, \tag{43.63}$$

where $e = \Phi(x) - \tilde{\Phi}_n(x)$ and L is the differential operator for the governing differential equation (i.e., it contains the derivatives operating on $\Phi(x)$ and any function multiplying $\Phi(x)$). For physical problems, this is often associated with the energy density.

Another common measure of convergence utilizes the L_2 norm. This can be termed the average error and can be associated with errors in any quantity. The L_2 norm is defined as

$$(e)_{L_2} = \left\{ \int_{\Omega} e^T e \, d\Omega \right\}^{1/2}. \tag{43.64}$$

While the norms given above are defined on the whole domain, one can note that the square of each can be obtained by summing element contributions

$$(e)^2 = \sum_{i=1}^{M} (e)_i^2, \tag{43.65}$$

where i represents an element contribution and m the total element number. Often for an *optimal* FE mesh, one tries to make the contributions to this square of the norm equal for all elements.

While the absolute values given by the energy or L_2 norms have little value, one can construct a relative percentage error that can be more readily interpreted:

$$\eta = \frac{(e)}{(\Phi)} \times 100. \tag{43.66}$$

This quantity, in effect, represents a weighted RMS error. The analysis can be determined for the whole domain or for element subdomains. One can use it in an adaptive algorithm by checking element errors against some predefined tolerance, η_0, and increasing the DOF only of those areas above the predefined tolerance.

Two other methods, the *p* and the *hp* methods, have been found, in most cases, to converge faster than the *h* method. The *p* method of refinement requires that one increase the order of the basis function that was used to represent the interpolation (i.e., linear to quadratic to cubic, etc.). The *hp* method is a combination of the *h* and *p* methods and has recently been shown to converge the fastest of the three methods (but, as you might imagine, it is the hardest to implement). To find out more about adaptive refinement methods, see References 2, 14, 22, 43, 47, 71, and 88.

43.6 Software for Bioelectric Field Problems

In the past few years, there have been a number of research software systems that have been created for the computational study of biomedical problems, including bioelectric field problems. Below, I have listed several open source software that are useful for computational bioelectric field problems. The list is meant to be representative and not comprehensive and I apologize for inevitable omissions.

- *SCIRun* (software.sci.utah.edu/scirun) is our own example of a general-purpose, problem-solving environment that has found extremely broad application both within biomedicine [34,46,48,77,85] and in areas as diverse as nuclear physics [49,70] and combustion [63]. An overview of SCIRun will be presented below.
- *CMISS* (www.cmiss.org) also has a very broad technical scope and application domain [9] and is the basis of many simulation studies in bioelectric fields and biomechanics of the heart and other organs [24,35,60], respiratory physiology [78], and bioelectric fields in the gastrointestinal system [68].
- *Simbios* (simbios.stanford.edu) is a newly emerging software system from the NIH-funded "Center for Physics-Based Simulation of Biological Structures" [72]. The biological coverage of Simbios is very broad, with the goal to help biomedical researchers understand biological

form and function as they create novel drugs, synthetic tissues, medical devices, and surgical interventions [8,10,11,20].

- *3D Slicer* (www.slicer.org) is a multiplatform, open source set of tools for visualization and image computing. It is also from an NIH NCBC Center, the "National Alliance for Medical Image Computing" (NA-MIC) (www.na-mic.org) [65]. Slicer includes a wide variety of image processing and visualization capabilities, including segmentation, registration, and analysis [52,56].
- *Seg3D* (www.seg3d.org) is a lightweight 3D segmentation program, which includes interactive volume visualization capabilities [91].
- *Brainstorm* (neuroimage.usc.edu/brainstorm) is an integrated toolkit dedicated to visualization and processing of data recorded from magnetoencephalography (MEG) and EEG. Brainstorm provides a comprehensive set of tools for researchers interested in MEG/EEG [41,61,74].
- *SimBio* and *NeuroFEM* (www.simbio.de and www.neurofem.com) is a combination of programs directed at source localization in the brain using patient-specific FE models with multiple conductivities and even anisotropic conductivity [85].
- *Continuity* (www.continuity.ucsd.edu) is a problem-solving environment for multiscale modeling in bioengineering and physiology with special emphasis on cardiac biomechanics, transport, and electrophysiology.
- *PCEnv* (www.cellml.org/downloads/pcenv) is the Physiome CellML Environment, an integrated software that provides an interface to the cell simulation models of the CellML project.
- *Virtual Cell* (www.nrcam.uchc.edu) is a software system for a wide range of scientists, from experimental cell biologists to theoretical biophysicists, who wish to create models of cellular structure and chemical, electrical, or mechanical function.
- *Neuron* (www.neuron.yale.edu/neuron) is a simulation environment for modeling individual neurons and networks of neurons, which is especially well suited to comparisons with experimental data. It has a very user-friendly interface that provides tools for building, managing, and using models in a way that is numerically sound and computationally efficient.
- *Genesis* (www.genesis-sim.org) has a very similar application domain as Neuron as a general-purpose simulation platform to simulate neural systems ranging from subcellular organelles and biochemical reactions to complex models of single neurons, large networks, and systems-level models.
- *TetGen* (tetgen.berlios.de) creates tetrahedral volume meshes from volume data made from triangulated surfaces for solving partial differential equations by FE or finite volume methods.
- *BioMesh3D* (www.sci.utah.edu/software) is a 3D tetrahedral and hexahedral mesh generator [92].
- *ITK* (www.itk.org), the Insight ToolKit, is a comprehensive set of software functions to perform image processing or analysis. ITK is the basis of many other tools (e.g., SCIRun and Seg3D) as they lack a graphical user interface (UI) and exist only as a C++ class library [37].
- *VTK* (www.vtk.org), the Visualization ToolKit, consists of an extensive library for visualization functions that is a component in many larger systems, for example, 3D slicer [73].
- *ImageVis3D* (www.sci.utah.edu/software) is a volume visualization program that allows for large-scale interactive visualization of scalar field datasets using isosurface extraction and volume rendering. ImageVis3D [93] works on multiple platforms, including desktops and laptops, the iPhone and iPad, and large distributed high-performance computers via the *VisIT* software system.
- *VisIT* (www.vacet.org) is a scalable, parallel software system for visualizing results of large-scale computational simulations. *VisIT* was created as part of the DOE ASCI and SciDAC programs. Research and development of *VisIT* continues as part of the DOE Visualization and Analytics Center for Enabling Technologies (VACET).
- *ECGSim* (www.ecgsim.org) is a program that computes the body surface potentials from the heart and allows the user to make changes in the electrical characteristics of the cells in any region of the heart. Its goal is to provide an educational tool but also a way to study the relationship between

the electric activity of the ventricular myocardium and the resulting potentials on the thorax under both normal and pathological conditions.

- *LabHeart* (www.labheart.org) is primarily a teaching tool that simulates the cardiac action potential, including the individual ionic currents and the fluctuations in intracellular calcium concentration.
- *iCell* is an Internet-based simulation program that allows the user to generate action potentials from a wide range of cell types [21].

43.6.1 SCIRun

This section provides a brief overview of the SCIRun and BioPSE problem-solving environments and presents examples of their use for the solution of bioelectric field problems.

The SCIRun* software system is an integrated, extensible, visualization-driven, open source, problem-solving environment that has been developed at the University of Utah's Scientific Computing and Imaging Institute [38].

For an application developer, SCIRun provides a software platform, upon which other applications can be rapidly constructed. SCIRun provides native support for interprocess communication, resource management (e.g., thread migration, memory management), and parallel computing. These operating system type services enable the dataflow aspects of the system. In addition to these low-level services, SCIRun also provides a number of built-in libraries and data structures that developers can use and can build upon. And at the highest level, SCIRun provides a rich set of algorithms for modeling, simulation, and visualization. All these levels of functionality can be leveraged by the developer when constructing new algorithms or applications in SCIRun [46,64,84].

The application program interface (API) to SCIRun is the visual dataflow environment called the network editor. Within the network editor, programs can be visually assembled from the library of available algorithms. The dataflow network for a sample bioelectric field simulation is shown in Figure 43.4.

The boxes in the network are called *modules*, and the lines connecting them are called *datapipes*. The point of attachment, where a datapipe attaches to a module, is called a *dataport*; the dataports on the tops of the modules are input ports, and the ports on the bottoms of the modules are output ports. In SCIRun, the dataports are color-coded to indicate the type of the data. For example, the blue datapipes are for matrices, and the yellow datapipes are for fields. Fields are used to represent 3D geometry as well as the data values that are defined over that geometry. Taken as a whole, the collection of modules and datapipes in a dataflow application is called a *network*, or *net*. Each module can have an optional UI button on its module; if the user presses the UI button, a separate window appears, with controls for viewing and modifying the state of the module's parameters.

43.6.2 BioPSE

SCIRun comes with a set of general-purpose modules that are not specific to any particular application. Modules can also be generated for a specific application, or for adding a set of optional functionality (such as raster data processing), in which case they are organized into a *package*. The package that has been primarily used and extended in this work is called BioPSE [17]. BioPSE stands for biomedical problem-solving environment, and contains all the functionality that is specific to bioelectric field problems.

The example network in Figure 43.4 is solving a bioelectric field problem for a dipolar source in a volume conductor model of a head. The domain is discretized with linear tetrahedral FEs, with five

* SCIRun is pronounced "ski-run" and derives its name from the Scientific Computing and Imaging (SCI) Institute, which is pronounced "ski" as in "Ski Utah."

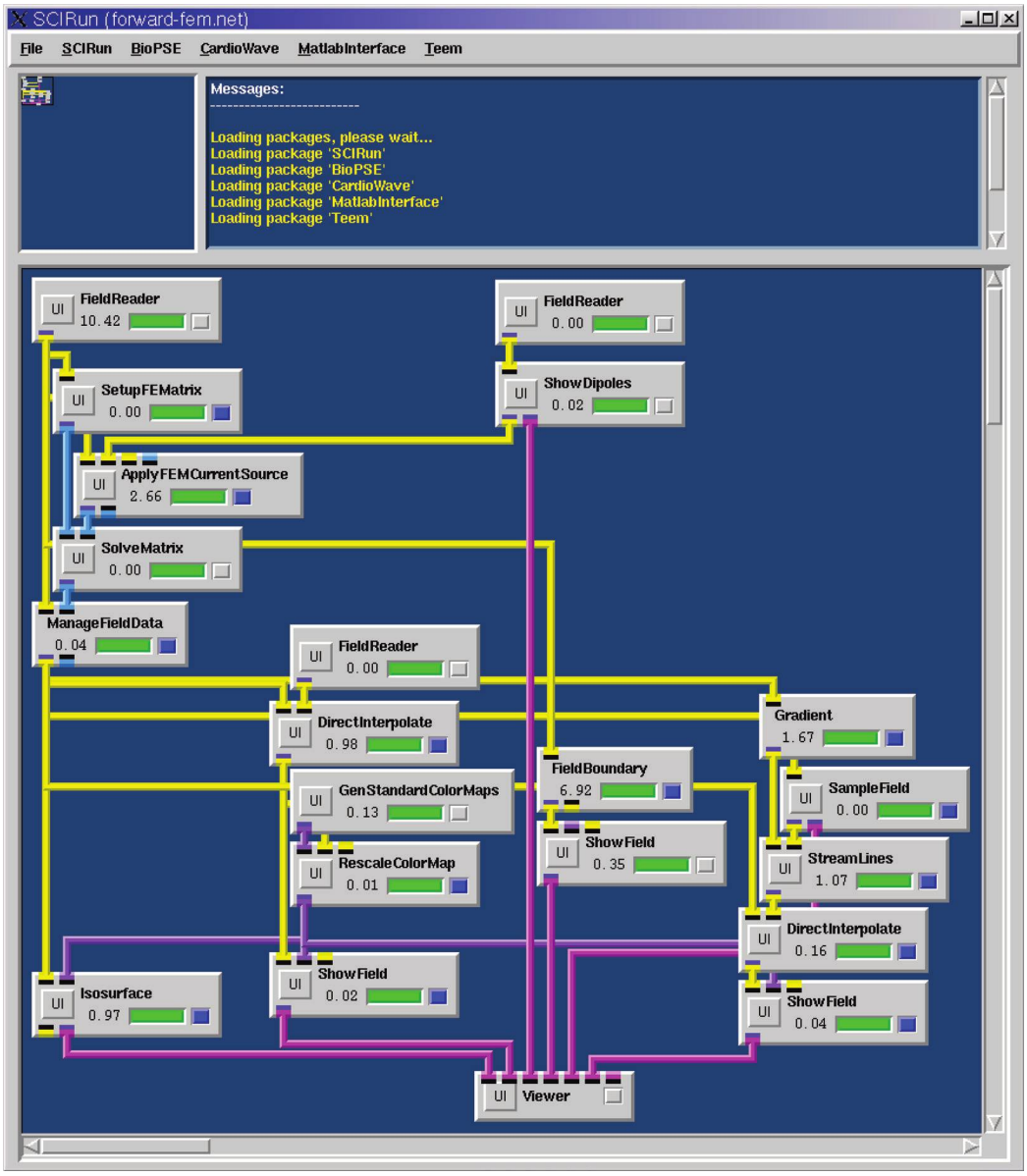

FIGURE 43.4 BioPSE dataflow network for modeling, simulating, and visualizing the bioelectric field generated in a realistic head model due to a single dipole source.

different conductivity types assigned through the volume. The problem is numerically approximated with a linear system, and is solved using the CG method. A set of virtual electrode points are rendered as pseudocolored spheres, to visualize the potentials at those locations on the scalp, and an isopotential surface and several pseudocolored electric field streamlines are also shown.

The BioPSE network implements this simulation and visualization with a collection of interconnected modules. The tetrahedral FE mesh with conductivity values is read in with one of the FieldReaders. That Field is then passed into the SetupFEMatrix module, which produces a stiffness matrix, **A**, as output. The RHS of the linear system, **b**, is generated by the ApplyFEMCurrentSource module, which applies

the dipole source as a boundary condition. The linear system $\mathbf{A\Phi} = \mathbf{b}$ is then solved by the SolveMatrix module to recover the potentials at all the nodes in the domain. This solution is then attached to the geometry with the ManageFieldData module, and the results are visualized. A complete description of this application is available in the tutorial section of the SCIRun User's Guide, and can be downloaded from the SCI Institute's website [1].

In addition to the BioPSE modules that appear in the above net, BioPSE also contains modules for generating and using FE lead fields, for constructing separating surfaces from segmented volumes or planar contours, for running boundary element method (BEM) simulations, and for visualizing lead potentials over time.

43.6.3 PowerApps

Historically, one of the major hurdles to SCIRun becoming a tool for the scientist as well as the engineer has been SCIRun's dataflow interface. While visual programming is natural for computer scientists and engineers, who are accustomed to writing software and building algorithmic pipelines, it is overly cumbersome for application scientists. Even when a dataflow network implements a specific application

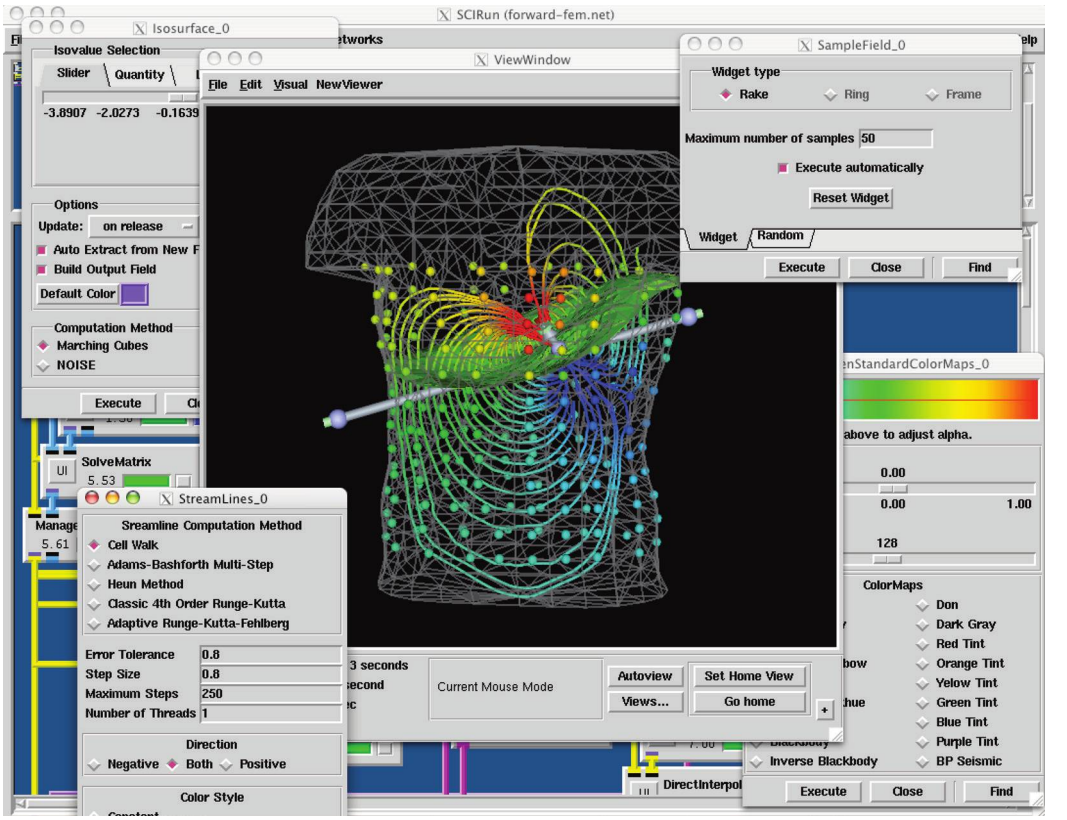

FIGURE 43.5 BioPSE dataflow interface to the forward bioelectric field application. The underlying dataflow network implements the application with modular interconnected components called modules. Data are passed between the modules as input and output parameters to the algorithms. While this is a useful interface for prototyping, it is nonintuitive for end-users; it is confusing to have a separate user interface window to control the settings for each module. Moreover, the entries in the user interface windows fail to provide a semantic context for their settings. For example, the text-entry field on the SampleField user interface that is labeled "Maximum number of samples" is controlling the number of electric field streamlines that are produced for the visualization.

(such as the forward bioelectric field simulation network provided with BioPSE and detailed in the BioPSE tutorial), the UI components of the network are presented to the user in separate UI windows, without any semantic context for their settings. For example, SCIRun provides file browser UIs for reading in data. However, on the dataflow network, all the file browsers have the same generic presentation. Historically, there has not been a way to present the filename entries in their semantic context, for example, to indicate that one entry should identify the electrodes input file and another should identify the FE mesh file.

While this interface shortcoming has long been identified, it has only recently been addressed. With the 1.20 release of BioPSE/SCIRun (in October 2003), we introduced *PowerApps*. A PowerApp is a customized interface built atop a dataflow application network. The dataflow network controls the execution and synchronization of the modules that comprise the application, but the generic UI windows are replaced with entries that are placed in the context of a single application-specific interface window.

With the 1.20 release of BioPSE, we released a PowerApp called BioFEM. BioFEM has been built atop the dataflow network shown in Figure 43.4, and provides a useful example for demonstrating the differences between the dataflow and PowerApp views of the same functionality. In Figure 43.5, the dataflow version of the application is shown: the user has separate interface windows for controlling different aspects of the simulation and visualization. In contrast, the PowerApp version is shown in Figure 43.6; here, the application has been wrapped up into a single interface window, with logically arranged and semantically labeled UI elements composed within panels and notetabs.

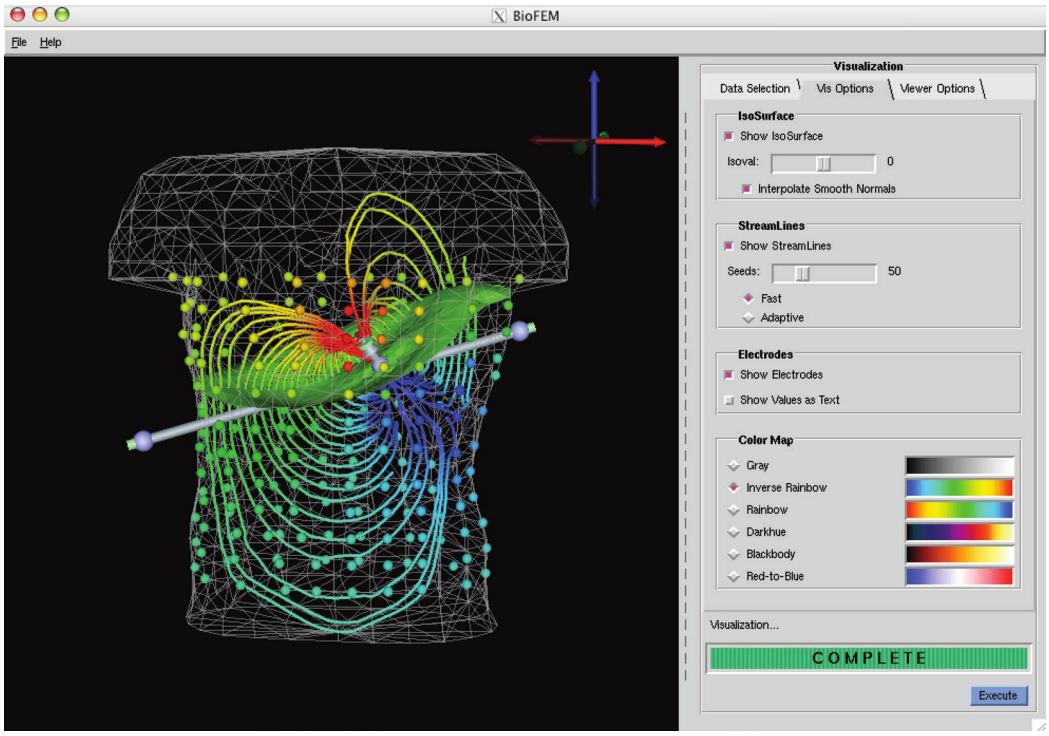

FIGURE 43.6 The BioFEM custom interface. Though the application is functionality equivalent to the dataflow version shown in Figures 43.4 and 43.5, this PowerApp version provides an easier-to-use custom interface. Everything is contained within a single window; the user is led through the steps of loading and visualizing the data with the tabs on the right; the generic control settings have been replaced with contextually appropriate labels; and application-specific tooltips (not shown) appear when the user places the cursor over any user interface element.

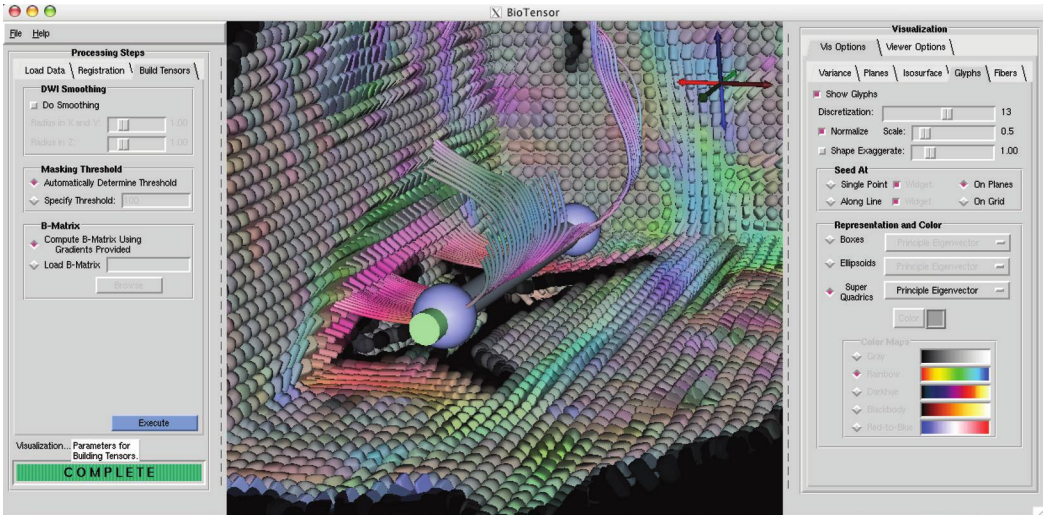

FIGURE 43.7 The BioTensor PowerApp. Just as with BioFEM, we have wrapped up a complicated dataflow network into a custom application. In the left panel, the user is guided through the stages of loading the data, co-registering the diffusion-weighted images, and constructing diffusion tensors. On the right panel, the user has controls for setting the visualization options. In the rendering window in the middle, the user can render and interact with the dataset.

In addition to bioelectric field problems, the BioPSE system can also be used to investigate other biomedical applications. For example, we have wrapped the tensor and raster data processing functionality of the Teem toolkit into the Teem package of BioPSE, and we have used that increased functionality to develop the BioTensor PowerApp, as seen in Figure 43.7. BioTensor presents a customized interface to a 140-module dataflow network. With BioTensor, the user can visualize diffusion-weighted imaging (DWI) datasets to investigate the anisotropic structure of biological tissues. The application supports the import of DICOM and Analyze datasets, and implements the latest diffusion tensor visualization techniques, including superquadric glyphs [51] and tensorlines [83] (both shown).

Acknowledgments

Support for this research comes largely from the NIH Center for Integrative Biomedical Computing (www.sci.utah.edu/cibc), funded by grants from the National Center for Research Resources (5P41RR012553-14) and the National Institute of General Medical Sciences (8 P41 GM103545-14) from the National Institutes of Health. I thank David Weinstein, Jeroen Stinstra, and Rob MacLeod for their contribution to the illustrative examples and the section on software.

References

1. 2010. Scientific Computing and Imaging Institute (SCI), University of Utah, www.sci.utah.edu.
2. A. Ainsworth and J.T. Oden. *A Posteriori Error Estimation in Finite Element Analysis*. Wiley-Interscience, New York, 2000.
3. J.E. Akin. *Finite Element Analysis for Undergraduates*. Academic Press, New York, 1986.
4. Th. Apel, M. Berzins, P.K. Jimack, G. Kunert, A. Plaks, I. Tsukerman, and M. Walkley. Mesh shape and anistropic elements: Theory and practice. *The Mathematics of Finite Elements and Applications X*, p. 367–376, 2000.
5. R.C. Barr, T.C. Pilkington, J.P. Boineau, and M.S. Spach. Determining surface potentials from current dipoles, with application to electrocardiography. *IEEE Trans Biomed Eng*, 13:88–92, 1966.

6. G. Beer and J.O. Watson. *Introduction to Finite and Boundary Element Methods for Engineers*. Wiley, New York, 1992.
7. O. Bertrand. 3D finite element method in brain electrical activity studies. In J. Nenonen, H.M. Rajala, and T. Katila, editors, *Biomagnetic Localization and 3D Modeling*, p. 154–171. Helsinki University of Technology, Helsinki, 1991.
8. T.F. Besier, G.E. Gold, S.L. Delp, M. Fredericson, and G.S. Beaupre. The influence of femoral internal and external rotation on cartilage stresses within the patellofemoral joint. *J Orthop Res*, 26(12):1627–1635, 2008.
9. S. Blackett, D. Bullivant, C. Stevens, and P. Hunter. Open source software infrastructure for computational biology and visualization. *Conf Proc IEEE Eng Med Biol Soc*, 6:6079–6080, 2005.
10. S.S. Blemker, D.S. Asakawa, G.E. Gold, and S.L. Delp. Image-based musculoskeletal modeling: Applications, advances, and future opportunities. *J Magn Reson Imaging*, 25(2):441–451, 2007.
11. G.R. Bowman, X. Huang, Y. Yao, J. Sun, G. Carlsson, L.J. Guibas, and V.S. Pande. Structural insight into RNA hairpin folding intermediates. *J Am Chem Soc*, 130(30):9676–9678, 2008.
12. A. Bowyer. Computing Dirichlet tesselations. *Comput J*, 24:162–166, 1981.
13. C.A. Brebbia and J. Dominguez. *Boundary Elements: An Introductory Course*. McGraw-Hill, Boston, 1989.
14. D.S. Burnett. *Finite Element Method*. Addison Wesley, Reading, MA, 1988.
15. M. Callahan, M.J. Cole, J.F. Shepherd, J.G. Stinstra, and C.R. Johnson. A meshing pipeline for biomedical computing. *Eng Comput*, 25(1):115–130, 2009.
16. C.D. Carbonera and J.F. Shepherd. A constructive approach to constrained hexahedral mesh generation. In *Proceedings of the 15th International Meshing Roundtable*, September 2006.
17. CIBC, 2009. BioPSE: Problem solving environment for modeling, simulation, image processing, and visualization for biomedical computing applications. Scientific Computing and Imaging Institute (SCI), Download from: http://www.scirun.org.
18. E.F. Van de Velde. *Concurrent Scientific Computing*. Springer-Verlag, New York, 1994.
19. C. Deitrich, C.E. Scheidegger, J. Schreiner, J. Comba, L.P. Nedel, and C.T. Silva. Edge transformations for improving mesh quality of marching cubes. *IEEE Trans Vis Comput Graphics*, 15(1):150–159, 2009.
20. S.L. Delp, F.C. Anderson, A.S. Arnold, P. Loan, A. Habib, C.T. John, E. Guendelman, and D.G. Thelen. Opensim: Open-source software to create and analyze dynamic simulations of movement. *IEEE Trans Biomed Eng*, 54(11):1940–1950, 2007.
21. S.S. Demir. Interactive cell modeling web-resource, iCell, as a simulation-based teaching and learning tool to supplement electrophysiology education. *Ann Biomed Eng*, 34:1077–1087, 2006.
22. J.E. Flaherty. *Adaptive Methods for Partial Differential Equations*. SIAM, Philadelphia, 1989.
23. T.L. Freeman and C. Phillips. *Parallel Numerical Algorithms*. Prentice Hall, New York, 1992.
24. A. Garny, P. Kohl, P.J. Hunter, M.R. Boyett, and D. Noble. One-dimensional rabbit sinoatrial node models: Benefits and limitations. *J Cardiovasc Electrophysiol*, 14(10 Suppl):S121–S132, 2003.
25. P.L. George. *Automatic Mesh Generation*. Wiley, New York, 1991.
26. V.B. Glasko. *Inverse Problems of Mathematical Physics*. American Institute of Physics, New York, 1984.
27. G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins, Baltimore, 1989.
28. G.H. Golub and J.M. Ortega. *Scientific Computing: An Introduction with Parallel Computing*. Academic Press, Boston, 1993.
29. F. Greensite, G. Huiskamp, and A. van Oosterom. New quantitative and qualitative approaches to the inverse problem of electrocardiology: Their theoretical relationship and experimental consistency. *Med Phys*, 17(3):369–379, 1990.
30. R.M. Gulrajani, F.A. Roberge, and G.E. Mailloux. The forward problem of electrocardiography. In P.W. Macfarlane and T.D. Veitch Lawrie, editors, *Comprehensive Electrocardiology*, p. 197–236. Pergamon Press, Oxford, England, 1989.

31. P.C. Hansen. Regularization tools: A MATLAB package for analysis and solution of discrete ill-posed problems. Technical report, Technical University of Denmark, 1992. Available via netlib in the library numeralgo/no4.
32. P.C. Hansen. Analysis of discrete ill-posed problems by means of the L-curve. *SIAM Rev*, 34(4):561–580, 1992.
33. C.S. Henriquez, C.R. Johnson, K.A. Henneberg, L.J. Leon, and A.E. Pollard. Large scale biomedical modeling and simulation: From concept to results. In N. Thakor, editor, *Frontiers in Biomedical Computing*. IEEE Press, Philadelphia, 1995.
34. C.S. Henriquez, J.V. Tranquillo, D.M. Weinstein, E.W. Hsu, and C.R. Johnson. Three-dimensional propagation in mathematic models: Integrative model of the mouse heart. In D.P. Zipes and J. Jalife, editors, *Cardiac Electrophysiology: From Cell to Bedside*, Chapter 30, p. 273–281. Saunders, Philadelphia, PA, 4th edition, 2004.
35. D.A. Hooks, K.A. Tomlinson, S.G. Marsden, I.J. LeGrice, B.H. Smaill, A.J. Pullan, and P.J. Hunter. Cardiac microstructure: Implications for electrical propagation and defibrillation in the heart. *Circ Res*, 91(4):331–338, 2002.
36. S.R.H. Hoole. *Computer-Aided Analysis and Design of Electromagnetic Devices*. Elsevier, New York, 1989.
37. L. Ibanez and W. Schroeder. *The ITK Software Guide 2.4*. Kitware, 2005.
38. SCI Institute, 2010. SCIRun: A scientific computing problem solving environment, Scientific Computing and Imaging Institute (SCI), Download from: http://www.scirun.org.
39. J.D. Jackson. *Classical Electrodynamics*. John Wiley & Sons, New York, 1975.
40. M.A. Jawson and G.T. Symm. *Integral Equation Methods in Potential Theory and Elastostatics*. Academic Press, London, 1977.
41. K. Jerbi, J.P. Lachaux, K. N'Diaye, D. Pantazis, R.M. Leahy, L. Garnero, and S. Baillet. Coherent neural representation of hand speed in humans revealed by MEG imaging. *Proc Natl Acad Sci USA*, 104(18):7676–7681, 2007.
42. C.R. Johnson and R.S. MacLeod. Computer models for calculating electric and potential fields in the human thorax. *Ann Biomed Eng*, 19:620, 1991. In *Proceedings of the 1991 Annual Fall Meeting of the Biomedical Engineering Society*.
43. C.R. Johnson and R.S. MacLeod. Expedition in das herz (in German). *GEO*, 1993.
44. C.R. Johnson, R.S. MacLeod, and P.R. Ershler. A computer model for the study of electrical current flow in the human thorax. *Comput Biol Med*, 22(3):305–323, 1992.
45. C.R. Johnson, R.S. MacLeod, and M.A. Matheson. Computer simulations reveal complexity of electrical activity in the human thorax. *Comput Phys*, 6(3):230–237, 1992.
46. C.R. Johnson, S.G. Parker, D. Weinstein, and S. Heffernan. Component-based problem solving environments for large-scale scientific computing. *J Conc Comp Prac Exper*, 14:1337–1349, 2002.
47. C.R. Johnson and A.E. Pollard. Electrical activation of the heart: Computational studies of the forward and inverse problems in electrocardiography. In *Computer Assisted Analysis and Modeling*, p. 583–628. MIT Press, Cambridge, MA, 1990.
48. M. Jolley, J. Stinstra, S. Pieper, R.S. MacLeod, D.H. Brooks, F. Cecchin, and J.K. Triedman. A computer modeling tool for comparing novel ICD electrode orientations in children and adults. *Heart Rhythm J*, 5(4):565–572, 2008.
49. C. Jones, K.-L. Ma, A.R. Sanderson, and L. Myers. Visual interrogation of gyrokinetic particle simulations. *J Phys*, 78:012033 (6pp), 2007.
50. Y. Kim, J.B. Fahy, and B.J. Tupper. Optimal electrode designs for electrosurgery. *IEEE Trans Biomed Eng*, 33:845–853, 1986.
51. G. Kindlmann. Superquadric tensor glyphs. In *Proceedings of the IEEE TVCG/EG Symposium on Visualization 2004*, p. 147–154, May 2004.
52. S. Lankton and A. Tannenbaum. Localizing region-based active contours. *IEEE Trans Imag Proc*, pp. 2029–2039, 11 2008.

53. M. Lizier, J.F. Shepherd, L.G. Nonato, J. Comba, and C.T. Silva. Comparing techniques for tetrahedral mesh generation. In *Proceedings of the Inaugural International Conference of the Engineering Mechanics Institute (EM 2008)*, pp. 1–8, 2008.
54. R.S. MacLeod, C.R. Johnson, and M.A. Matheson. Visualization of cardiac bioelectricity—A case study. In *Proceedings of the IEEE Visualization 92*, p. 411–418. IEEE CS Press, 1992.
55. R.S. MacLeod, C.R. Johnson, and M.A. Matheson. Visualization tools for computational electrocardiography. In *Visualization in Biomedical Computing*, p. 433–444. Bellingham, WA, 1992. Proceedings of the SPIE #1808.
56. M. Maddah, W. Grimson, S. Warfield, and W. Wells. A unified framework for clustering and quantitative analysis of white matter fiber tracts. *Med Image Anal*, pp. 191–202, 04 2008.
57. M. Meyer, P. Georgel, and R.T. Whitaker. Robust particle systems for curvature dependent sampling of implicit surfaces. In *Proceedings of the International Conference on Shape Modeling and Applications (SMI)*, p. 124–133, June 2005.
58. M. Meyer, B. Nelson, R.M. Kirby, and R.T. Whitaker. Particle systems for efficient and accurate finite element visualization. *IEEE Trans Vis Comput Graphics* 13(5): 1015–1026, 2007.
59. C.E. Miller and C.S. Henriquez. Finite element analysis of bioelectric phenomena. *Crit Rev Biomed Eng*, 18:181–205, 1990.
60. A.J. Pullan M.P. Nash. Challenges facing validation of noninvasive electrical imaging of the heart. *Ann Noninvasive Electrocardiol*, 10(1):73–82, 2005.
61. K. N'Diaye, R. Ragot, L. Garnero, and V. Pouthas. What is common to brain activity evoked by the perception of visual and auditory filled durations? a study with MEG and EEG co-recordings. *Brain Res Cogn Brain Res*, 21(2):250–268, 2004.
62. J. Nenonen, H.M. Rajala, and T. Katilia. *Biomagnetic Localization and 3D Modelling*. Helsinki University of Technology, Espoo, Finland, 1992. Report TKK-F-A689.
63. S.G. Parker. Component-based multi-physics simulations of fires and explosions. In *Proceedings of the 12th SIAM Conference on Parallel Processing for Scientific Computing*, 2006. Presented at the Minisymposium on Parallel Dynamic Data Management Infrastructures for Scientific & Engineering Applications.
64. S.G. Parker, D.M. Weinstein, and C.R. Johnson. The SCIRun computational steering software system. In E. Arge, A.M. Bruaset, and H.P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, p. 1–40. Birkhauser Press, Boston, 1997.
65. S. Pieper, B. Lorensen, W. Schroeder, and R. Kikinis. The NA-MIC kit: ITK, VTK, Pipelines, Grids and 3D Slicer as an open platform for the medical image computing community. In *Proceedings of the IEEE International Symposium on Biomedical Imaging*, 4 2006.
66. T.C. Pilkington, B. Loftis, J.F. Thompson, S. L-Y. Woo, T.C. Palmer, and T.F. Budinger. *High-Performance Computing in Biomedical Research*. CRC Press, Boca Raton, FL, 1993.
67. R. Plonsey. *Bioelectric Phenomena*. McGraw-Hill, New York, 1969.
68. A. Pullan, L. Cheng, R. Yassi, and M. Buist. Modelling gastrointestinal bioelectric activity. *Prog Biophys Mol Biol*, 85(2–3):523–550, 2004.
69. Y. Rudy and B.J. Messinger-Rapport. The inverse solution in electrocardiography: Solutions in terms of epicardial potentials. *Crit Rev Biomed Eng*, 16:215–268, 1988.
70. A.R. Sanderson, C.R. Johnson, and R.M. Kirby. Display of vector fields using a reaction diffusion model. In *Proceedings of IEEE Visualization 2004*, p. 115–122, 2004.
71. J.A. Schmidt, C.R. Johnson, J.C. Eason, and R.S. MacLeod. Applications of automatic mesh generation and adaptive methods in computational medicine. In I. Babuska, J.E. Flaherty, W.D. Henshaw, J.E. Hopcroft, J.E. Oliger, and T. Tezduyar, editors, *Modeling, Mesh Generation, and Adaptive Methods for Partial Differential Equations*, p. 367–390. Springer-Verlag, 1995.
72. J.P. Schmidt, S.L. Delp, M.A. Sherman, C.A. Taylor, V.S. Pande, and R.B. Altman. The Simbios national center: Systems biology in motion. *Proc IEEE*, 96(8):1266–1280, 2008.

73. W. Schroeder, K. Martin, and B. Lorensen. *Visualization Toolkit: An Object-Oriented Approach to 3D Graphics, 4th Edition*. Kitware, Clifton Park, NY, 2006.
74. C. Sergent, S. Baillet, and S. Dehaene. Timing of the brain events underlying access to consciousness during the attentional blink. *Nat Neurosci*, 8(10):1391–1400, 2005.
75. J.F. Shepherd. *Topologic and Geometric Constraint-Based Hexahedral Mesh Generation*. PhD thesis, School of Computing, University of Utah, May 2007.
76. J.F. Shepherd and C.R. Johnson. Hexahedral mesh generation constraints. *J Eng Comput*, 24(3):195–213, 2008.
77. J.G. Stinstra, S. Shome, B. Hopenfeld, and R.S. MacLeod. Modeling the passive cardiac electrical conductivity during ischemia. *Med Biol Eng Comput*, 43(6):776–782, 2005.
78. M.H. Tawhai, A.J. Pullan, and P.J. Hunter. Generation of an anatomically based three-dimensional model of the conducting airways. *Annal Biomed Eng*, 28(7):793–802, 2000.
79. J. Thompson, Z. Warsi, and C. Mastin. *Numerical Grid Generation Foundations and Applications*. North Holland, New York, 1985.
80. J. Thompson and N.P. Weatherill. Structed and unstructed grid generation. In T.C. Pilkington, B. Loftis, J.F. Thompson, S. L-Y. Woo, T.C. Palmer, and T.F. Budinger, editors, *High-Performance Computing in Biomedical Research*, p. 63–112. CRC Press, Boca Raton, FL, 1993.
81. A. Tikhonov and V. Arsenin. *Solution of Ill-Posed Problems*. Winston, Washington, DC, 1977.
82. A.N. Tikhonov and A.V. Goncharsky. *Ill-Posed Problems in the Natural Sciences*. MIR Publishers, Moscow, 1987.
83. D. Weinstein, G. Kindlmann, and E. Lundberg. Tensorlines: Advection-diffusion based propagation through diffusion tensor fields. In *Proceedings of IEEE Visualization 1999*, p. 249–253, 1999.
84. D.M. Weinstein, S.G. Parker, J. Simpson, K. Zimmerman, and G.M. Jones. Visualization in the sci-run problem-solving environment. In C.D. Hansen and C.R. Johnson, editors, *The Visualization Handbook*, p. 615–632. Elsevier, Burlington, MA, 2005.
85. C.H. Wolters, A. Anwander, X. Tricoche, D.M. Weinstein, M.A. Koch, and R.S. Macleod. Influence of tissue conductivity anisotropy on EEG/MEG field and return current computation in a realistic head model: A simulation and visualization study using high-resolution finite element modeling. *Neuroimage*, 30(3):813–826, 2006.
86. Y. Wu, S.K. Warfield, I.L. Tan, W.M. Wells, D.S. Meier, R.A. van Schijndel, F. Barkhof, and C.R. Guttmann. Automated segmentation of multiple sclerosis lesion subtypes with multichannel MRI. *Neuroimage*, 32(3):1205–1215, 2006.
87. Y. Yamashita. Theoretical studies on the inverse problem in electrocardiography and the uniqueness of the solution. *IEEE Trans Biomed Eng*, 29:719–725, 1982.
88. O.C. Zienkiewicz. *The Finite Element Method in Engineering Science*. McGraw-Hill, New York, 1971.
89. O.C. Zienkiewicz and J.Z. Zhu. A simple error estimate and adaptive procedure for practical engineering analysis. *Int J Num Meth Eng*, 24:337–357, 1987.
90. O.C. Zienkiewicz and J.Z. Zhu. Adaptivity and mesh generation. *Int J Num Meth Eng*, 32:783–810, 1991.
91. CIBC, 2013. Volumetric image segmentation and visualization. Scientific Computing and Imaging Institute (SCI), Download from: http://www.seg3d.org.
92. SCI Institute, 2013. BioMesh3D: Quality mesh generator for biomedical applications. Scientific Computing and Imaging Institute (SCI). Download from http://www.biomesh3d.org.
93. CIBC, 2013. ImageVis3D: A SCIRun Power App for interactive visualization of vary large image volumes. Scientific Computing and Imaging Institute (SCI), Download from: http://www.imagevis3d.org.