BF

Direct and Inverse Bioelectric Field Problems

1 Introduction

Over the last two decades, computer modeling and simulation have become increasingly important to the fields of Bioengineering and Medicine. 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. Secondly, since the level of complexity one can model parallels existing hardware configurations, primarily tuned to memory, disk capacity, and CPU speed advances in computer technology have made it feasible to apply the computational modeling paradigm to complex living systems.

For these reasons, biological complexity still outstrips the capabilities of even the largest computational systems, and will for some time to come, the computational methodology has taken hold in biology and medicine and has been used successfully to suggest physiologically and clinically important scenarios and results.

No matter the size of the system to be modeled, the goal remains the same: to understand which of the system's characteristics and interactions are essential in order to quantify and represent its behavior. One always seeks biologically useful solutions and interpretations of the mathematical and computational results. Such results may help describe known behavior as well as predict unknown responses and suggest new representations. While simple to state, the aforementioned goals are difficult to realize. Once the task of defining the problem is accomplished, the modeler must be able to bring the proper tools to bear on solving the problem. While the requisite *toolset* will depend upon the application, it is sure to include techniques from a wide variety of disciplines, including computer science, mathematics, physics, and engineering.

In this case study, we will describe the techniques of modeling and simulation which can be applied to a class of bioelectric field problems. Bioelectric field problems can be found in a wide variety of biomedical applications which range from single cells [1], to organs [2], up to models which incorporate partial to full human structures [3, 4, 5, 6]. We will describe some general modeling techniques which will be applicable, in part, to all the aforementioned applications. We will focus our study on a class of direct and inverse volume conductor problems which arise in electrocardiography and electroencephalography. The solutions to these problems have applications to defibrillation studies, detection and location of arrhythmias, impedence imaging techniques, and localization and analysis of spontaneous brain activity in epileptic patients; furthermore, they can, in general, be used to estimate the electrical activity inside a volume conductor, either from potential measurements at an outer surface, or directly from the interior bioelectric sources.

Specifically, we first describe the physics and physiology of a bioelectric volume conductor, continue by describing the model construction process, emphasizing mesh generation, and follow with mathematical formulations, numerical solutions, and computational considerations of a set of direct and inverse problems. Throughout the chapter, we provide opportunities for you, the reader, to investigate various techniques in computational science yourself. We conclude with a section in which we interpret the results in terms of an error analysis and briefly introduce adaptive techniques to reduce the error due to discretization.

1.1 Bioelectric Volume Conductors

A general volume conductor can be defined as a region of volume, Ω, which has conductivity, σ, and permitivity, ε, in which resides a source current, I_V, where the V signifies per-unit volume. Solving a volume conductor problem means finding expressions for the electric, $\mathbf{E}$, and potential, Φ, fields everywhere within the volume, Ω, and/or on one of the bounded surfaces, Γ_i. While not unique to biophysics, a property that provides one with significant challenges, is the complexity of the domain, Ω, which is subdivided into several sub-domains, each with different conductivity. In some cases, these inhomogeneous regions are anisotropic (e.g. causing the field to vary with direction).

The bioelectric current sources, I_V, arise from excitable cells undergoing an activation process. Activation of cardiac tissue, for example, can be characterized as the process in which cells undergo rapid depolarization (e.g. when movement of ions across the cell membrane results in inactivation of electrical charges and a drop in potential; see [7, 8] for more details). The depolarization process causes a propagation of excitation waves to move through the myocardium (the muscular layer of the heart); these waves in turn produce an extracellular potential field, Φ. This potential field can be characterized by the geometry and conductivity of the volume conductor, and by the distance from, orientation to, and intensity of the source current, I_V.

A similar but more complicated process of depolarization occurs in the brain. While it too undergoes an activation process, the propagation occurs in a less continuous manner than in the heart. In the brain, the individual neurons determine the path that activation takes. Thus the activation process in the brain is inherently discontinuous.

Because I_V is, in general, time-varying, the resulting field quantities are governed by

Maxwell's equations. For a discussion on modeling the wave propagation through active myocardial tissue, the reader is directed to [9, 10, 11, 12, 13]. For the macroscopic volume conductor problem (in which we do not consider the individual membrane currents), we can utilize a quasi-static approximation. Because the displacement current, $(j\omega\varepsilon E)$, is much smaller than the conduction current, (σE), propagation effects are negligible, and inductive effects are minute [7]. This leads to a reduction of Maxwell's equations to one of two forms. The more general form is a reduction to Poisson's equation for electrical conduction:

$$\nabla \cdot \sigma \nabla \Phi = -I_V \qquad \text{in } \Omega \tag{1}$$

where σ is the electrical conductivity tensor, Φ is the electrostatic potential, I_V is the current source per unit volume, and Ω is the solution domain. In this form, one includes the source region and an understanding of the primary bioelectric sources, I_V, usually in the form of a simplified mathematical model. Alternatively, one can define a surface bounding the region which includes the sources and recast the formulation in terms of information on that surface, yielding Laplace's equation:

$$\nabla \cdot \sigma \nabla \Phi = 0 \qquad \text{in } \Omega. \tag{2}$$

One then solves these equations subject to an appropriate set of boundary conditions. At first glance, this seems simple: we know the governing equation and we know how to solve Poisson's and Laplace's equations. However, as we shall see, there are some significant challenges when one tries to solve a large scale, inhomogeneous, anisotropic, inverse volume conductor problem that exhibits the complicated geometry of the human body.

2 Direct and Inverse Problem Formulation

Most bioelectric field problems (volume conductor problems) can be formulated in terms of either the Poisson or Laplace equation for electrical conduction. Since Laplace's equation is the homogeneous counterpart of the Poisson equation, we will develop the treatment for a general three-dimensional Poisson problem and discuss simplifications and special cases when necessary.

As we noted previously, a *typical* bioelectric volume conductor can be posed as the following boundary value problem:

$$\nabla \cdot \sigma \nabla \Phi = -I_V \tag{3}$$

The associated boundary conditions depend on what type of problem one wishes to solve. There are usually considered to be two different types of direct and inverse volume conductor problems. One type 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 direct problem would be to solve (3) for Φ with a known description of I_V and the Neumann boundary condition:

$$\sigma\nabla\Phi\cdot\mathbf{n}=0 \qquad \text{on } \Gamma_T \tag{4}$$

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 (electroencephioliography) and ECG (electrocardiography) 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. 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) and calculates the currents and voltages within the volume conductor of the chest and voltages on the surface of the torso. 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 body. Thus one would solve (3) with the boundary conditions:

$$\Phi=\Phi_0 \qquad \text{on } \Sigma\subseteq\Gamma_T \tag{5}$$

$$\sigma\nabla\Phi\cdot\mathbf{n}=0 \qquad \text{on } \Gamma_T. \tag{6}$$

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 as seen before. While it doesn't look much different than the formulation of the direct problem, the mathematician Hadamard [14] noticed that inverse formulations of boundary value problems (or their algebraic counterpart) were often *ill-behaved.* He defined the conditions for *well-posed* and *ill-posed* problems. For a problem to be well-posed in the Hadamard sense, it must meet the following criteria:

(1) For each set of data, there exists a solution.

(2) The solution is unique.

(3) The solution depends continuously on the data.

If a problem does not meet one or more of these criteria the problem is considered to be ill-posed. The bioelectric inverse problem in terms of primary current sources that was just presented lacks two of the three criteria for being well-posed: there is not a unique solution, and the solution doesn't depend continuously on the data. The first property, lack of uniqueness, means that there is a multitude of solutions; the second property means that small errors in measurements may cause large errors in the solution. When Hadamard wrote his 1902 paper defining well- and ill-posed problems, it was with the intent of saving mathematicians and scientists substantial time and trouble. That is, he warned against trying to solve ill-posed problems, as there wasn't any way of getting accurate results from such problems. Unfortunately (or fortunately, depending on your point of view), there exist many important problems in science and engineering that are ill-posed and that are in need of solutions. These include problems in mechanical engineering (inverse kinematics, crack detection), robotics (vision), geophysics (geophysical prospecting), astrophysics, quantum mechanics (inverse scattering), medicine (ultrasound, bioelectric and biomagnetic source problems), electrical engineering (inverse optics), and elsewhere. Since there are so many important ill-posed problems, mathematicians, scientists, and engineers have developed numerous methods to *get around* the problem of ill-posedness.

While many strategies tend to be problem-dependent, there are a few which are generally applicable to bioelectric field problems and worth mentioning. One general technique for dealing with the problem of nonuniqueness transfers the problem to be solved into a model problem. Here, one usually breaks up the solution domain into a finite number of subdomains. In each subdomain, one describes a simplified model of the actual bioelectric source (such as a dipole). The problem then becomes to find the magnitude and direction of each of the simplified sources in the subdomains (as opposed as to having to find the locations of the actual sources, plus their magnitudes). Other methods work to limit the space of the solution domain [15] so that one can perhaps pick out solutions which are the only physiologically feasible ones.

Once the problem of nonuniqueness is addressed, there still exists the problem occurring when the solution does not depend continuously on the data. The linear algebra counterpart to the elliptic boundary value problem is often useful in discussing this problem of discontinuity. The numerical solution to all elliptic boundary value problems (such as the Poisson and Laplace problems) can be written in the form of a set of linear equations:

$$Az = u \quad z \in Z, u \in U \tag{7}$$

where z is the solution vector (the electrostatic potential on the surface of the brain or heart, for example), u is the vector of input data (the subset of voltage values on the surface of the skull or torso, for example), and A is the *transfer matrix* between z and u, which usually contains the geometry and physical properties (conductivities, dielectric constants, etc.) of the volume conductor. Z and U are the metric spaces where we measure z and u. The direct problem is then simply posed as solving (7) for u given z. Likewise, the inverse problem is to determine z given u.

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. When one inverts a matrix which has a very large condition number, it is highly susceptible to errors. 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 which have $\kappa(A) \approx 1$, while an ill-conditioned matrix will have $\kappa(A) \gg 1$. 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 [41] for more about the condition number of matrices.

See exercises 1 and 2.

As an example of the simplest of Tikhonov's methods, consider minimizing the following functional for the solution vector z:

$$M\{A, z, u\} = \|Az - u\|^2 + \alpha\|Iz\|^2 \quad \alpha > 0 \tag{8}$$

which yields:

$$z = [A^T A + \alpha I]^{-1}[A^T u] \tag{9}$$

This is the Tikhonov method with zeroth order regularization with regularization parameter α. The *zeroth order* corresponds to the inclusion of the identity matrix in equation (9). Other matrices that are used in practice include numerical approximations of the gradient or Laplacian, depending on which characteristic of the solution one is trying to constrain.

See exercise 3.

Another bioelectric inverse formulation that appears in the literature poses both the direct and inverse problems in terms of scalar values at the surfaces. For example, both the EEG and ECG problems can be posed in this manner. For the EEG problem, one would take the surface of the brain (cortex) 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. The inverse problems are formulated as using measurements on the surface of the skull (torso) and calculating the voltages on the surface of the brain (heart). Here we are solving Laplace's equation instead of Poisson's equation, because we are solving for distributions of voltages on a surface instead of current sources within a volume.

$$\nabla \cdot \sigma \nabla \Phi = 0 \qquad \text{in } \Omega \tag{10}$$

with boundary conditions (5) and (6). This particular formulation is known as the Cauchy problem for Laplace's equation. At first glance it doesn't seem that we would gain much from this formulation; and in fact, it might appear that we are losing something in that we cannot easily infer the location of the sources that produced the voltages on the surface of the heart or skull if the sources are far from the surface. What we have gained is that the solution for such a problem is unique [21]. The problem is still ill-posed in that the solution does not depend continuously on the data, but at least we do not have to contend with the multiplicity of solutions.

Let's explore this problem further. Note that we want to solve for the voltages on the inner (heart) surface given the voltages on the outer (thorax) surface, plus a Neuman boundary condition on the outer surface. If we represent this problem as a linear system (which we will do in detail in the section on Numerical Solution and Computational Considerations 4), we begin by expressing $A\Phi = b$ for the Cauchy problem in (10) as

$$\begin{pmatrix} A_{TT} & A_{TV} & A_{TE} \\ A_{VT} & A_{VV} & A_{VE} \\ A_{ET} & A_{EV} & A_{EE} \end{pmatrix} \begin{pmatrix} \Phi_T \\ \Phi_V \\ \Phi_E \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \tag{11}$$

We can then rearrange the matrix to solve directly for the heart voltages in terms of the measured body surface voltages,

$$\Phi_E = (A_{TV}A_{VV}^{-1}A_{VE} - A_{TE})^{-1}(A_{TT} - A_{TV}A_{VV}^{-1}A_{VT})\Phi_T, \tag{12}$$

Here the subscripts, T, V, and E stand for the nodes in the regions of the torso, volume, and epicardium (surface of the heart), respectively. In general, the A_{TE} sub-matrix is zero (except for very simple models in which nodes connect directly from the torso to the heart). We can then rewrite (12) as

$$\Phi_E = (A_{VE}^{-1}A_{VV}A_{TV}^{-1})(A_{TT} - A_{TV}A_{VV}^{-1}A_{VT})\Phi_T \tag{13}$$

which reduces to

$$\Phi_E = A^{-1}\Phi_T \tag{14}$$

where the matrix A contains all the geometry and conductivity values, the so-called transfer matrix.

See exercise 4.

---

3 Model Construction and Mesh Generation

Once we have stated or derived the mathematical equations which 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 partial differential equations break up the continuous domain into discrete elements and approximate the pde using the particular numerical technique (finite element, boundary element, finite difference, or multigrid) best suited to the particular problem.

The first problem we encounter is the discretization of the solution domain into polygons, or the *mesh generation*. Because of the complex geometries often associated with bioelectric field problems, construction of the polygonal mesh can turn out to be 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 of our fundamental element. For the sake of simplicity, we will assume that we will use linear elements, triangles for two-dimensional meshes and tetrahedrons for three-dimensional domains. If we chose to use the finite difference approach, we would need to use uniform squares of cubes.

There are several different strategies for discretizing the geometry into fundamental elemements. For bioelectric field problems, there are two approaches to mesh construction that have become standard; the *divide and conquer* (or subsequent subdivision) strategy, and a method based upon finding the dual to the Voronoi diagram, the so-called *Delaunay triangulation* strategy.

Simply put, in using the divide and conquer strategy one starts with a set of points which 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 (you might be thinking about what kind of stopping criteria should be applied to stop the division). Usually, the domain is broken up into eight-node cubic elements, which can then be subdivided into five (minimally) or six tetrahedral elements for three-dimensional problems and the corresponding quadralaterals and triangles for two-dimensional problems. This methodology has the advantage of being fairly easy to program (also, commercial mesh generators exist for the divide and conquer method). For use in solving bioelectric field problems, its main disadvantage has to do with its inability to control the elements which 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 and at the same time accurately represent the geometry.

See exercise 5.

A second method of mesh generation is the Delaunay triangulation strategy. Given a two- or three-dimensional set of points which define the boundaries and interior regions of the domain to be modeled, one tessellates the point cloud into an optimal mesh of triangles or 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 the mesh can be developed to fit any predefined geometry, including subsurfaces, by starting with points which define all the necessary surfaces and subsurfaces and then adding additional interior points to minimize the aspect ratio. For triangles, the aspect ratio is defined to be either the ratio of the maximum horizontal length of the element to the maximum vertical length of the element, or the ratio of the diameter of a circumscribed circle to the maximal distance between vertices. For tetrahedra, the aspect ratio can be defined as $4\sqrt{\frac{3}{2}}\frac{\rho_k}{h_k}$ 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 "isosceles" tetrahedron (triangle) and a value of 0 for a degenerate (flat) element [22]. The closer to the value of 1, the better (it should be noted that this is not a general rule for all problems: in certain problems where there are very high gradients in a uniform direction, as in high speed fluid flow, for example, it is often better to distort the aspect ratio of the element to take into account this phenomena). The Delaunay criterion is a method for minimizing the occurrence of obtuse angles in the mesh, yielding elements which 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, two-dimensional versions, including one from netlib called sweep2.c from the directory Voronoi and at least one three-dimensional package [46].

Another drawback of the Delaunay method of mesh generation is that it can produce elements that lie within the convex hull of the point set but outside the bounding surface (contour). For example, if one is triangulating a two-dimensional kidney-shaped object, the Delaunay method will construct triangles outside the bounding contour in the concave C-shaped region. One simple way to rid the mesh of unwanted triangles (or tetrahedra) is to supplement the mesh generator with the following algorithm, based on the Gauss-Bonnet (GB) theorem of topology [23]. If one calculates the angles subtended by the points bounding a two-dimensional area about a point P (adding consecutive angles with some sign convention), the sum will equal 2π if P is inside the enclosed contour, π if P is on the boundary, and 0 if P lies outside. Thus it is a simple matter to test a questionable element by checking to see if its centroid is inside or outside the contour.

In the analogous three-dimensional version, one calculates the solid angles about the centroids of each tetrahedron, which will sum to 4π for interior points and 0 for exterior points. We can exploit this idea further to complete one last bit of preprocessing before solving the system. Since our solution domain, the head or thorax, is not homogeneous, we need to assign conductivity values to the various tissue regions.

In the FE formulation, we may need to assign a conductivity tensor to each element of the model. This is usually done by using a table look-up scheme in which each element is given a number according to the type of tissue it contains. During processing, the corresponding value from a table of conductivity tensors is used to calculate the global stiffness matrix, making it a trivial matter to change the various conductivities without changing the geometry or any other parameters of the model.

One can use the GB (or another more computationally efficient) algorithm to automatically ascertain to which conductivity group an element belongs by defining the boundaries of the various subsurfaces and determining whether the centroids of the elements are inside or outside the region in question, assigning each to an appropriate conductivity group

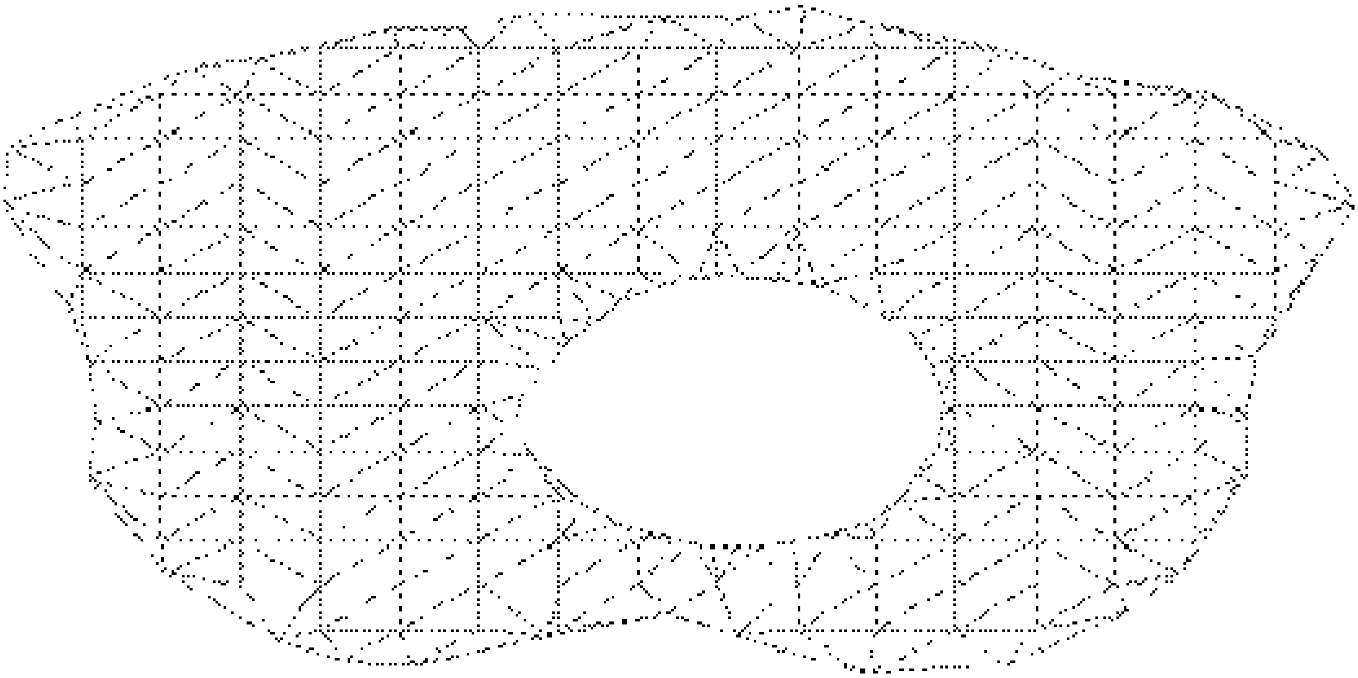

Figure 1: Delaunay triangulation of the MRI scan for a small mesh density.

number. Once all the conductivity tensors are assigned to the groups, the preprocessing is complete and we are ready to compute solutions. For more information on mesh generation and various aspects of biomedical modeling, see [24, 25, 26, 27, 28, 29, 30, 31].

See exercise 6.

Figures 1 and 2 illustrate the Delaunay triangulation of the MRI scan for two different mesh densities.

4 Numerical Solution and Computational Considerations

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).

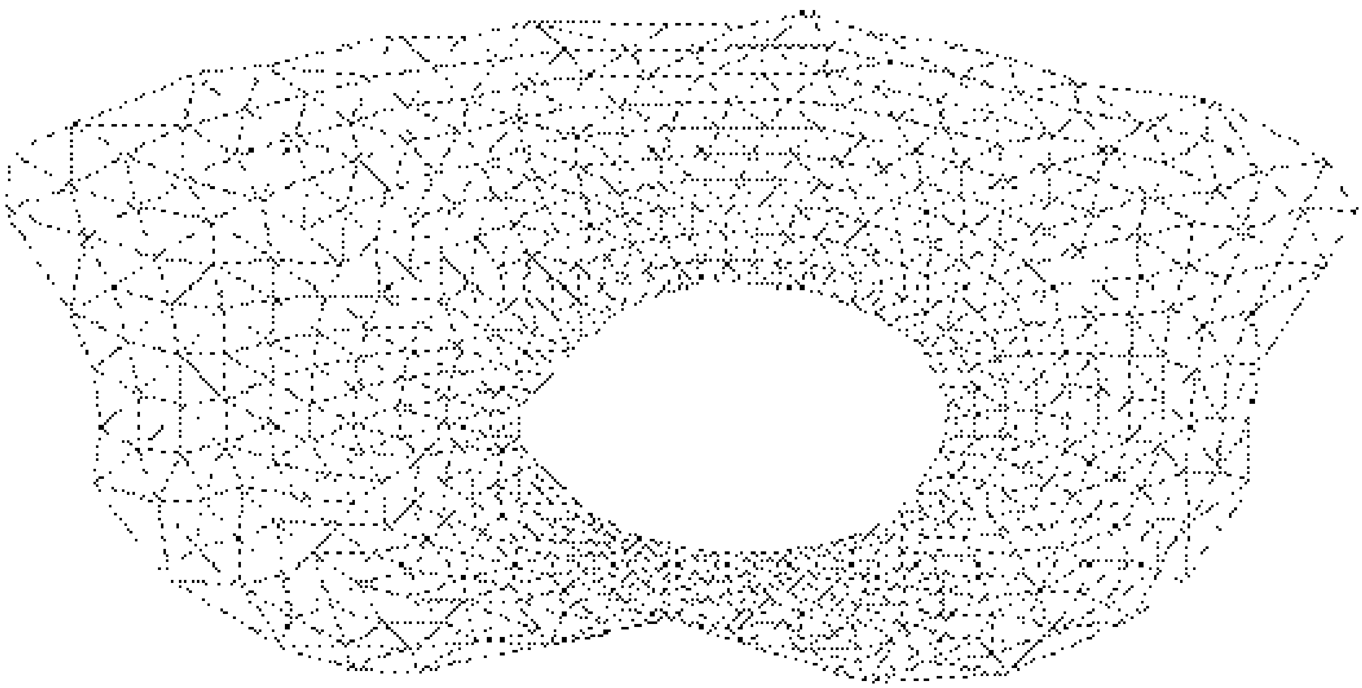

Figure 2: Delauany triangulation of the MRI scan for a medium mesh density.

4.1 The Galerkin Method - Optional

This section can be bypassed by students who have not encountered finite element, finite difference, or boundary element methods before. It does, however, provide a unifying basis from which the finite and boundary element and finite difference methods can be derived.

The problem posed in (1) can be solved using any of the aforementioned approximation schemes. One technique which 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 (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 (1), as:

$$A\Phi = -I_v \tag{15}$$

where A is the differential operator, $A = \nabla \cdot (\sigma \nabla)$. An equivalent statement of (15) is, find Φ such that $(A\Phi + I_v, \overline{\Phi}) = 0$. Here, $\overline{\Phi}$ 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 $L_2(\Omega)$, i.e. the space of square integrable functions. Applying Green's theorem, we can equivalently write,

$$(\sigma \nabla \Phi, \nabla \overline{\Phi}) - \langle \frac{\partial \Phi}{\partial n}, \overline{\Phi} \rangle = -(I_v, \overline{\Phi}) \tag{16}$$

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 (1):

$$(\sigma \nabla \Phi, \nabla \overline{\Phi}) = -(I_v, \overline{\Phi}) \tag{17}$$

It is understood that this equation must hold for all test functions, $\overline{\Phi}$, which must vanish at the boundaries where $\Phi = \Phi_0$. The Galerkin approximation ϕ to the weak form solution Φ in (17) can be expressed as:

$$\phi(\mathbf{x}) = \sum_{i=0}^{N} \phi_i \psi_i(\mathbf{x}) \tag{18}$$

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

$$(\sigma \nabla \phi, \nabla \psi_j) = -(I_v, \psi_j) \qquad (\forall \psi_j \in \mathcal{S}) \tag{19}$$

Since our differential operator A is positive definite and self adjoint (i.e., $(A\Phi, \Phi) \geq \alpha(\Phi, \Phi) > 0$ for some non-zero positive constant α and $(A\Phi, \overline{\Phi}) = (\Phi, A\overline{\Phi})$, respectively), then we can define a space E with an inner product defined as $(\Phi, \overline{\Phi})_E = (A\Phi, \overline{\Phi}) \equiv a(\Phi, \overline{\Phi})$ and norm (the so-called energy norm) equal to:

$$\|\Phi\|_E = \{ \int_\Omega (\nabla \Phi)^2 d\,\Omega \}^{\frac{1}{2}} = (\Phi, \Phi)_E^{\frac{1}{2}} \tag{20}$$

The solution Φ of (15) satisfies:

$$(A\Phi, \psi_i) = -(I_v, \psi_i) \qquad (\forall \psi_i \in \mathcal{S}) \tag{21}$$

and the approximate Galerkin solution obtained by solving (19) satisfies:

$$(A\phi, \psi_i) = -(I_v, \psi_i) \qquad (\forall \psi_i \in \mathcal{S}) \tag{22}$$

Subtracting (21) from (22) yields:

$$(A(\phi - \Phi), \psi_i) = (\phi - \Phi, \psi_i)_E = 0 \qquad (\forall \psi_i \in \mathcal{S}) \tag{23}$$

The difference $\phi - \Phi$ denotes the error between the solution in the infinite dimensional space V and the $N + 1$ dimensional space $\mathcal{S}$. Equation (23) 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 $\mathcal{S}$ 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:

$$(A\phi_1, \psi_i) = -(I_v, \psi_i) \quad (A\phi_2, \psi_i) = -(I_v, \psi_i) \qquad (\forall \psi_i \in \mathcal{S}) \tag{24}$$

respectively. Then, the difference yields:

$$(A(\phi_1 - \phi_2), \psi_i) = 0 \qquad (\forall \psi_i \in \mathcal{S}) \tag{25}$$

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

$$\sum_{j=0}^{N} \Delta\phi_j (A(\psi_j, \psi_i)) = 0 \qquad (\forall \psi_i \in \;) \tag{26}$$

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

$$(A\Phi_i, \Phi_i) > 0 \qquad i = 0, \ldots, N \tag{27}$$

Thus, $\Delta\phi_i = 0,\ i = 0, \ldots, N$, and by virtue of (26), $\delta\phi = 0$, such that $\phi_1 = \phi_2$. The identity contradicts the assumption of two distinct Galerkin solutions. This thus proves the solution is unique [32].

4.2 The Finite Difference Method

Perhaps the most traditional way to solve (1) utilizes the finite difference approach by discretizing the solution domain Ω using a grid of quadralaterals (for 2D) or cubes (for 3D). 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 (see the section on *Some High Performance Computing Issues in Partial Differential Equations*). For example:

$$\Phi_{l+1,m,n} = (\Phi + h\frac{\partial \Phi}{\partial x} + \frac{1}{2}h^2\frac{\partial^2 \Phi}{\partial x^2} + \frac{1}{6}h^3\frac{\partial^3 \Phi}{\partial x^3} + \ldots)_{l,m,n} \tag{28}$$

with similar equations for $\Phi_{l-1,m,n}, \Phi_{l,m+1,n}, \Phi_{l,m-1,n,\ldots}$. The finite difference representation of (1) is:

$$\frac{\Phi_{l+1,m,n} - 2\Phi_{l,m,n} + \Phi_{l-1,m,n}}{h^2} + \frac{\Phi_{l,m+1,n} - 2\Phi_{l,m,n} + \Phi_{l,m-1,n}}{h^2} + \frac{\Phi_{l,m,n+1} - 2\Phi_{l,m,n} + \Phi_{l,m,n-1}}{h^2} = -I_{l,m,n}(v) \tag{29}$$

or, equivalently,

$$\Phi_{l+1,m,n} + \Phi_{l-1,m,n} + \Phi_{l,m+1,n} + \Phi_{l,m-1,n} + \Phi_{l,m,n+1} + \Phi_{l,m,n-1} - 6\Phi_{l,m,n} = -h^2 I_{l,m,n}(v) \quad (30)$$

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 (30), we can reformulate (1) by its finite difference approximation in the form of the matrix equation, $A\mathbf{\Phi} = \mathbf{b}$, where $\mathbf{b}$ is a vector which 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, (1), would require evaluating (19) for the specific grid formation and specific choice of basis function:

$$\int_\Omega (\sigma_x \frac{\partial\phi}{\partial x}\frac{\partial\psi_i}{\partial x} + \sigma_y \frac{\partial\phi}{\partial y}\frac{\partial\psi_i}{\partial y} + \sigma_z \frac{\partial\phi}{\partial z}\frac{\partial\psi_i}{\partial z}) d\Omega = -\int_\Omega I_v \psi_i d\Omega \quad (31)$$

Difference quotients are then used to approximate the derivatives in (31). We note that if linear basis functions are utilized in (31), one obtains a formulation which corresponds exactly with the standard finite difference 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.

See exercise 7.

4.3 The Finite Element Method

As we have seen above, in the classical numerical treatment for partial differential equations - the finite difference 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 which 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 Finite Element Method, the solution domain can be discretized into a number of uniform or non-uniform finite elements 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 (19)). 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 $A\mathbf{\Phi} = \mathbf{b}$, which is subsequently solved for the unknown variable [33, 25].

The formulation of the finite element approximation starts with the Galerkin approximation, $(\sigma\nabla\Phi, \nabla\overline{\Phi}) = -(I_v, \overline{\Phi})$, where $\overline{\Phi}$ is our test function. We now use the finite element 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, $V_h \subset V = \{\overline{\Phi} : \overline{\Phi}$ is continuous on Ω , $\nabla\overline{\Phi}$ is piecewise continuous on $\Omega\}$. One usually defines parameters of the function $\overline{\Phi} \in V_h$ at node points, $\alpha_i = \overline{\Phi}(x_i)$, i = 0, 1, ..., 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 $\overline{\Phi} \in V_h$ as:

$$\overline{\Phi}(x) = \sum_{i=0}^{N} \alpha_i \Psi_i(x) \tag{32}$$

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

$$\text{Find } \Phi_h \in V_h \text{ such that } (\sigma\nabla\Phi_h, \nabla\overline{\Phi}) = -(I_v, \overline{\Phi}) \tag{33}$$

Furthermore, if $\Phi_h \in V_h$ satisfies (33), then we have $(\sigma\nabla\Phi_h, \nabla\Psi_i) = -(I_v, \Psi_i)$ [34]. Finally, since Φ_h itself can be expressed as the linear combination:

$$\Phi_h = \sum_{i=0}^{N} \xi_i \Psi_i(x) \qquad \xi_i = \Phi_h(x_i) \tag{34}$$

we can then write (33) as:

$$\sum_{i=0}^{N} \xi_i(\sigma_{ij}\nabla\Psi_i, \nabla\Psi_j) = -(I_v, \Psi_j) \qquad j = 0, \ldots, N \tag{35}$$

subject to the Dirichlet boundary condition. Then the finite element approximation of (1) can equivalently be expressed as a system of N equations with N unknowns $\xi_i, \ldots, \xi_N$ (the electrostatic potentials, for example). 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 of 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).

4.3.1 Some Details for 2-D Domains

Now that we have a feeling for how the finite element method works on a theoretical level, let's now work up the practical machinery for handling 2D problems. We note that the extension to 3D is straightforward in terms of the theory and algorithms. The difficulties when extending to 3D (the so-called *curse of dimensionality*) have to do with the size of the resulting system and in creating the finite element mesh.

Let's start with an arbitrary 2D planar domain Ω and break it up into discrete elements to form a finite dimensional subspace, Ω_h. For 2D planar domains we have the choice of representing our function (if we assume for now that we will use linear elements) as either triangles,

$$\Phi \approx \tilde{u} = \alpha_1 + \alpha_2 x + \alpha_3 y \tag{36}$$

or as quadralaterals,

$$\Phi \approx \tilde{u} = \alpha_1 + \alpha_2 x + \alpha_3 y + \alpha_4 xy. \tag{37}$$

For now, let's restrict our development to triangles, knowing that it is easy to modify our formulae for quadralaterals. We take out a specific triangle from our finite dimensional subspace and apply the previous formulations for the three vertices,

$$\begin{aligned}\tilde{\Phi}_1 &= \alpha_1 + \alpha_2 x_1 + \alpha_3 y_1 \\ \tilde{\Phi}_2 &= \alpha_1 + \alpha_2 x_2 + \alpha_3 y_2 \\ \tilde{\Phi}_3 &= \alpha_1 + \alpha_2 x_3 + \alpha_3 y_3 \end{aligned} \tag{38}$$

or in matrix form,

$$\begin{pmatrix} \tilde{\Phi}_1 \\ \tilde{\Phi}_2 \\ \tilde{\Phi}_3 \end{pmatrix} = \begin{pmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix}. \tag{39}$$

We can now solve for the coefficients:

$$\alpha_1 = \frac{1}{2A} \begin{vmatrix} \tilde{\Phi}_1 & x_1 & y_1 \\ \tilde{\Phi}_2 & x_2 & y_2 \\ \tilde{\Phi}_3 & x_3 & y_3 \end{vmatrix} \tag{40}$$

$$\alpha_2 = \frac{1}{2A} \begin{vmatrix} 1 & \tilde{\Phi}_1 & y_1 \\ 1 & \tilde{\Phi}_2 & y_2 \\ 1 & \tilde{\Phi}_3 & y_3 \end{vmatrix} \tag{41}$$

$$\alpha_3 = \frac{1}{2A} \begin{vmatrix} 1 & x_1 & \tilde{\Phi}_1 \\ 1 & x_2 & \tilde{\Phi}_2 \\ 1 & x_3 & \tilde{\Phi}_3 \end{vmatrix} \tag{42}$$

where

$$2A = \begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix}. \tag{43}$$

If we now define,

$$\begin{aligned} a_1 &= x_2y_3 - x_3y_2 \\ b_1 &= y_2 - y_3 \\ c_1 &= x_3 - x_2 \end{aligned} \tag{44}$$

we can express our coefficients as

$$\begin{aligned} \alpha_1 &= (a_1\tilde{\Phi}_1 + a_2\tilde{\Phi}_2 + a_3\tilde{u\Phi}_3)/2A \\ \alpha_2 &= (b_1\tilde{\Phi}_1 + b_2\tilde{\Phi}_2 + b_3\tilde{\Phi}_3)/2A \\ \alpha_3 &= (c_1\tilde{\Phi}_1 + c_2\tilde{\Phi}_2 + c_3\tilde{\Phi}_3)/2A \end{aligned} \tag{45}$$

with

$$2A = a_i + b_ix_i + c_iy_i \tag{46}$$

were we can permute the indicies $1, 2, 3$ to find $a_2, b_2, etc.$ Now we can write an expression for a general point in our triangle in terms of the values at the vertices,

$$\tilde{\Phi} = \frac{1}{2A}[(a_1 + b_1x + c_1y)\tilde{\Phi}_1 + (a_2 + b_2x + c_2y)\tilde{\Phi}_2 + (a_3 + b_3x + c_3y)\tilde{\Phi}_3] \tag{47}$$

or, most succintly,

$$\tilde{\Phi} = \sum_i N_i\tilde{\Phi}_i \tag{48}$$

with $\tilde{\Phi}_i$, the solution value at node i, and N_i is called 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 finite element analysis):

$$\Phi_j(N_i) = N_i(x, y) = f_i(x, y) \equiv \frac{a_i + b_ix + c_iy}{2A}. \tag{49}$$

From our definition of $2A$ we can see that $N_1 = 1$ at node 1 and $N_1 = 0$ at nodes $2, 3$. In general this can be expressed as,

$$\Phi_j(N_i) \equiv \delta_{ij} \tag{50}$$

where δ_{ij} is the Kroneker delta.

Our expression for the shape functions is, in fact, a transformation from cartesian coordinates to a natural set of coordinates, often called *area coordinates* or *barycentric coordinates.* If we look at the transformation for N_2 for a point within the element with coordinates (x_p, y_p), we note that,

$$N_2 = \frac{a_2 + b_2x_p + c_2y_p}{2A} = \frac{Area(P13)}{Area(123)} \tag{51}$$

which is just a ratio of the areas of the triangles $(P13)$ and $(123) = A$. Thus we have a transformation from a point inside an element in terms of its barycentric coordinates to the global cartesian system,

$$\begin{aligned} x &= N_1x_1 + N_2x_2 + N_3x_3 \\ y &= N_1y_1 + N_2y_2 + N_3y_3 \\ 1 &= N_1 + N_2 + N_3. \end{aligned} \tag{52}$$

Now that we have a suitable set of basis functions, we can find the finite element approximation to our 2D problem. Recall that our orginal problem can be formulated as,

$$a(u, v) = (I_v, v) \quad \forall v \in \Omega \tag{53}$$

where

$$a(u, v) = \int_\Omega \nabla u \cdot \nabla v dx \tag{54}$$

and

$$(I_v, v) = \int_\Omega I_v \cdot v dx \tag{55}$$

The finite element approximation to the original boundary value problem is

$$a(u_h, v) = (I_v, v) \quad \forall v \in \Omega_h \tag{56}$$

which has the equivalent form

$$\sum_{i=1}^{N} \xi_i a(\Phi_i, \Phi_j) = (I_v, \Phi_j) \tag{57}$$

where

$$a(\Phi_i, \Phi_j) = a(\Phi_i(N_j), \Phi_j(N_i)) \tag{58}$$

which can be expressed by the matrix and vector elements,

$$(a_{ij}) = \int_{\Omega_E} (\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}) d\Omega \tag{59}$$

and

$$I_i = \int_{\Omega_E} N_i I_v \, d\Omega \tag{60}$$

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

$$\int_{\Omega_h} N_1^a N_2^b N_3^c \, d\Omega = 2A \frac{a!b!c!}{(a+b+c+2)!}. \tag{61}$$

Therefore,

$$(a_{ij}) = \int_{\Omega_E} \frac{b_ib_j + c_ic_j}{4A^2} \, d\Omega = \frac{b_ib_j + c_ic_j}{4A} \tag{62}$$

and for the right hand side we have, assuming constant sources,

$$I_i = \int_{\Omega_E} \frac{a_i + b_i x + c_i y}{2A} I_v \, d\Omega = \frac{AI_v}{3} \tag{63}$$

Thus the local, element matrix looks like

$$\frac{1}{4A}\begin{pmatrix} b_1^2 + c_1^2 & b_1 b_2 + c_1 c_2 & b_1 b_3 + c_1 c_3 \\ & b_2^2 + c_2^2 & b_2 b_3 + c_2 c_3 \\ \text{Sym.} & & b_3^2 + c_3^2 \end{pmatrix}\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \frac{AI_v}{3}\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \tag{64}$$

which have the compact forms,

$$a_{ij}^{(n)} = \frac{1}{4A}(b_i^{(n)} b_j^{(n)} + c_i^{(n)} c_j^{(n)}) \tag{65}$$

and

$$I_i^{(n)} = \frac{AI_v}{3} \quad \text{for constant sources.} \tag{66}$$

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

$$\sum_{n=1}^{Nel} (a_{ij}^{(n)})(\xi_i) = (I_i^{(n)}) \tag{67}$$

where Nel is equal to the total number of elements in the discretized solution domain and i represents the node numbers (vertices). Let's say we want to add the contributions from two elements which have global numbers (134) for element one and (324) for element two. The local element matrices look like

$$\begin{pmatrix} a_{11}^1 & a_{12}^1 & a_{13}^1 \\ & a_{22}^1 & a_{23}^1 \\ S & & a_{33}^1 \end{pmatrix} \tag{68}$$

and

$$\begin{pmatrix} a_{11}^2 & a_{12}^2 & a_{13}^2 \\ & a_{22}^2 & a_{23}^2 \\ S & & a_{33}^2 \end{pmatrix} \tag{69}$$

where the S denotes symmetry. The compenents of (a_{ij}) represent the connectivities in the global matrix and are the contributions from the (Φ_i, Φ_j) pair. Thus the two element system would have the following global matrix in terms of the local contributions:

$$\begin{pmatrix} a_{13}^1 & 0 & a_{31}^1 & a_{32}^1 \\ & a_{11}^2 & a_{13}^2 & a_{12}^2 \\ & & a_{11}^1 + a_{33}^2 & a_{12}^1 + a_{32}^2 \\ S & & & a_{22}^1 + a_{22}^2 \end{pmatrix}. \tag{70}$$

This process continues until we have added in all our element contributions to form the global system. Now we have only to take into account of our boundary conditions and then solve our system.

4.3.2 Boundary Conditions

For the finite difference method, it turns out that the Dirichlet boundary conditions is very easy to apply while the Neumann condition takes a little extra effort. For the finite element method it is just the opposite. The Neumann boundary condition

$$\nabla\Phi \cdot \mathbf{n} = 0 \tag{71}$$

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

$$\int_\Omega \nabla \cdot \mathbf{A} dx = \int_\Gamma \mathbf{A} \cdot \mathbf{n}\, dS \tag{72}$$

and applying it to the left hand side of the Galerkin finite element formulation

$$\begin{aligned}\int_\Omega \nabla v \cdot \nabla w \, d\Omega \equiv \int_\Omega [\frac{\partial v}{\partial x_1}\frac{\partial w}{\partial x_1} + \frac{\partial v}{\partial x_2}\frac{\partial w}{\partial x_2}]\, d\Omega \\ = \int_\Gamma [v\frac{\partial w}{\partial x_1}n_1 + v\frac{\partial w}{\partial x_2}n_2]\, dS - \int_\Omega v[\frac{\partial^2 w}{\partial x_1^2} + \frac{\partial^2 w}{\partial x_2^2}]\, d\Omega \\ = \int_\Gamma v\frac{\partial w}{\partial n}\, dS - \int_\Omega v\nabla^2 w \, d\Omega.\end{aligned} \tag{73}$$

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

$$(I_v, v) = -\int_\Omega (\nabla^2\Phi) v \, d\Omega = -\int_\Gamma \frac{\partial\Phi}{\partial n} v \, dS + \int_\Omega \nabla\Phi \cdot \nabla v \, d\Omega = a(\Phi, v) \tag{74}$$

where the boundary integral term, $\frac{\partial\Phi}{\partial n}$ vansishes and we obtain the standard Galerkin finite element 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 f_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 r.h.s. the product of a_{ij} and the known value of Φ_i (call it $\bar{\Phi}_i$; this yields the new right hand side, $\hat{f}_i = f_i - a_{ij}\bar{\Phi}_j$.
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 r.h.s. equal to $\bar{\Phi}_i$.
5. Continue for each Dirichlet condition.
6. Solve the augmented system, $\hat{A}\Phi = \hat{I}_v$.

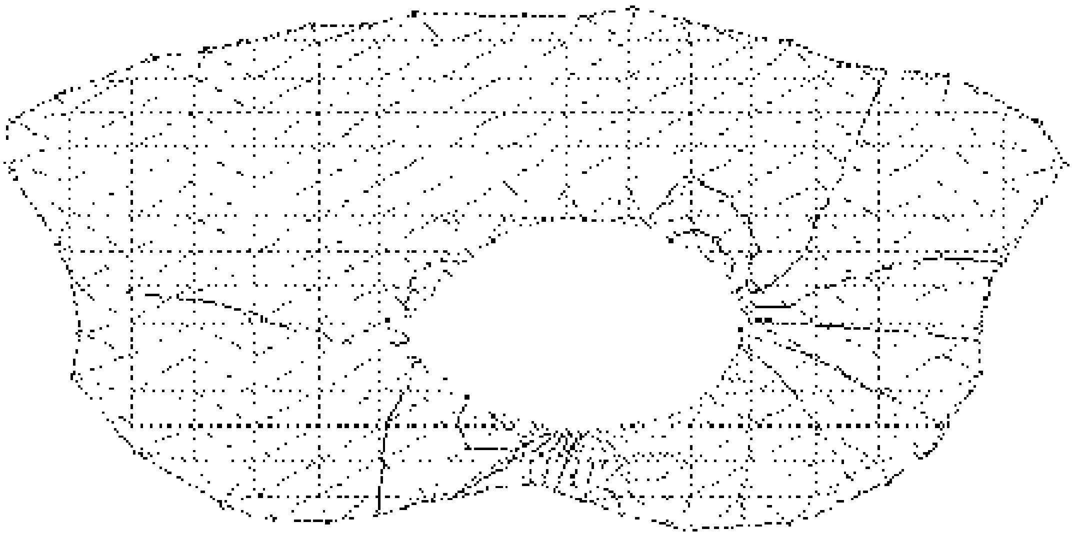

Figure 3: Finite element solution in the form of isovoltage contours overlayed on the finite element mesh for a small mesh density.

Consider, for example, the 3×3 system,

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \end{pmatrix} = \begin{pmatrix} I_{v1} \\ I_{v2} \\ I_{v3} \end{pmatrix}. \tag{75}$$

where Φ_3 is known, $\bar{\Phi}_3$. Following the steps enumerated above, we obtain the following augmented system,

$$\begin{pmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \end{pmatrix} = \begin{pmatrix} I_{v1} - a_{13}\bar{\Phi}_3 \\ I_{v2} - a_{23}\bar{\Phi}_3 \\ \bar{\Phi}_3 \end{pmatrix}. \tag{76}$$

Solving this system using a standard solver for $Ax = b$ yields the solution set, $(\Phi_1, \Phi_2, \bar{\Phi}_3)$.

As one might imagine, the process of preparing the augmented system is somewhat time consuming if one has a large number of Dirichlet boundary conditions.

Now that we've enforced our boundary condition, we *just* need to solve our system...

See exercises 8 and 9.

Figures 3 and 4 depict the finite element solution in the form of isovoltage contours overlayed on the finite element mesh at two mesh densities.

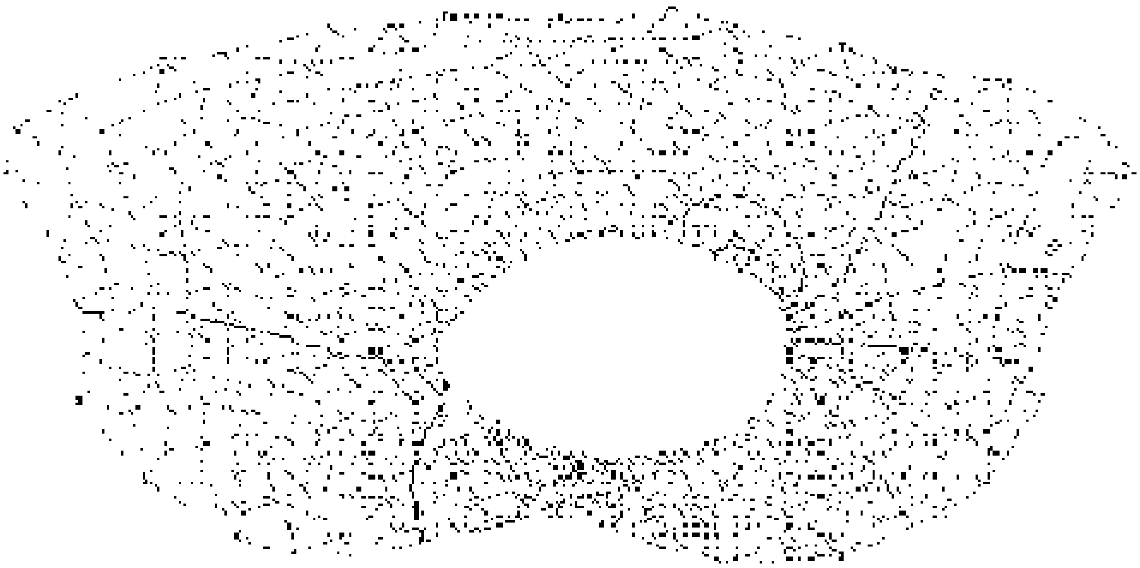

Figure 4: Finite element solution in the form of isovoltage contours overlayed on the finite element mesh for a medium mesh density.

4.4 The Boundary Element Method

For bioelectric field problems with isotropic domains (and few inhomogeneities), another, somewhat less familiar technique, called the boundary element 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 are 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 non-zero coefficients in any given matrix row. As is demonstrated by Maxwell's laws [35], equations in differential forms can often be replaced by equations in integral forms; e.g. the potential distribution in a domain is uniquely defined by the volume sources and the potential and current density on the boundary. The boundary element method utilizes this fact by transforming the differential operator defined in the domain to integral operators defined on the boundary. In the boundary element method [36, 37, 38], 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.

See project in section 9.

---

5 Comparison of Methods

Since we don't have space to give 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 which 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 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 book-keeping than a 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.

6 Adaptive Methods

See exercise 10.

We stated in the introduction that one of the goals of the modeler is to accurately approximate the solution of bioelectric field problems using a numerical technique. We have discussed how one formulates the problem, discretizes the geometry, and finds an approximate solution. We are now faced with answering the question posed in the problem above pertaining to the accuracy of our solution. 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'll apply the finite element method to solve Laplace's 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):

$$\Phi(x+h,y+k) = \Phi(x,y) + [h\frac{\partial\Phi(x,y)}{\partial x} + k\frac{\partial\Phi(x,y)}{\partial y}] +$$
$$\frac{1}{2!}[h^2\frac{\partial^2\Phi(x,y)}{\partial^2 x} + 2hk\frac{\partial^2\Phi(x,y)}{\partial x\partial y} + k^2\frac{\partial^2\Phi(x,y)}{\partial^2 y}] + \ldots \quad (77)$$

where h and k are the maximum x and y distances within an element respectively. 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:

$$\frac{\partial\Phi(x_i,y_i)}{\partial x} = \frac{1}{2A}[\Phi_i(y_j - y_m) + \Phi_m(y_i - y_j) + \Phi_j(y_m - y_i) \quad (78)$$

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

$$\frac{\partial\Phi(x_i,y_i)}{\partial x} = \frac{\partial\Phi(x_j,y_j)}{\partial x} = \frac{\partial\Phi(x_m,y_m)}{\partial x} \quad (79)$$

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 which 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:

$$\epsilon \approx \frac{1}{2!}[h^2\frac{\partial^2\Phi(x,y)}{\partial^2 x} + 2hk\frac{\partial^2\Phi(x,y)}{\partial x\partial y} + k^2\frac{\partial^2\Phi(x,y)}{\partial^2 y}] \quad (80)$$

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 (80) 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 (15) by hk, one can also express the error in terms of the elemental aspect ratio $\frac{h}{k}$, 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.

See exercise 11.

6.0.1 Convergence of a Sequence of Approximate Solutions

Let's try to quantitfy our error a bit further. When we consider the preceding example, it seems to make sense that if we increase the number of degrees of freedom 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 degrees of freedom (DOF) increases indefinitely.

$$\|\Phi(x) - \tilde{\Phi}_n(x)\| \rightarrow 0 \quad \text{as } N \rightarrow \infty \tag{81}$$

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 $N \rightarrow \infty$. This is stronger than pointwise convergence as it requires a unform 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 [45].

In general, proving pointwise convergence is very difficult except in the simplest cases, while proving the convergence of an averged 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 an energy sense because they typically want to know values of the solution $\Phi(x)$ and graidents, $\nabla\Phi(x)$ 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,

$$\|\Phi_m(x) - \tilde{\Phi}_n(x)\| \rightarrow 0 \quad \text{as } M, N \rightarrow \infty \tag{82}$$

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 an answer other than the true solution.

While we cannot be assured pointwise convergence of these functions for all but the simplist 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 averge sense (energy, etc.), then it will converge in the pointwise sense as well.

6.0.2 Energy Norms

The error in energy, measured by the *energy norm*, is defined in general as [42, 43, 44],

$$\|e\| = \left(\int_\Omega e^T L e \, d\Omega\right)^{\frac{1}{2}} \tag{83}$$

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,

$$\|e\|_{L_2} = (\int_\Omega e^T e \, d\Omega)^{\frac{1}{2}}. \tag{84}$$

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,

$$\|e\|^2 = \sum_{i=1}^{M} \|e\|_i^2 \tag{85}$$

where i represents an element contribution and m the total element number. Often for an *optimal* finite element 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:

$$\eta = \frac{\|e\|}{\|u\|} \times 100. \tag{86}$$

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 those areas which are above the predefined tolerance.

For the convergence of the above norms to be valid, we need to make sure our basis functions meet certain criteria, namely those of *completeness* and *continuity.* These can be stated as follows [45]:

Completeness condition. The element trial solution, $\tilde{\Phi}^{(e)}$, and any of its derivatives up to order m appearing in the integrals of the Galerkin (or variational/weak) formulation, should be able to assume any constant value within an element when, in the limit, the size of the element decreases to zero.

Continuity condition. At interelement boundaries, the element trial solutions should be C^{m-1}-continuous; that is, $\tilde{\Phi}^{(e)}$ and its derivatives up to order $m-1$ should be continuous.

If the element basis functions, $N_i^{(e)}$, are constructed so that $\tilde{\Phi}^{(e)}$ satifies both conditions, then a sequence of approximate solutions, corresponding to a sequence of successively refined meshes, will converge energywise to the exact solution (again, assuming no other errors) in the limit as $h_{\max} \to 0$, where h represents the size of the element. This is often referred to as *h-convergence.*

The completeness condition is necessary for convergence to the exact solution. An incomplete element will generally converge to some other limiting function. The continuity condition is not necessary for convergence. The purpose of the continuity condition is to ensure that discontinuities at the interelement boundaries are not severe enough to introduce errors in addition to the discretization error.

If continuity and completeness conditions are satisfied, convergence is assured. An element which satisfies both conditions is called a *conforming* or *compatible* element. An element which satifies completeness but not continuity is called a *nonconforming* or *incompatible* element. For an expanded discussion of completeness and continuity see [45, 34].

See exercise 12.

Two other methods, the p and the hp methods, have been found, in most cases, to converge faster. 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). Take a minute to estimate the number of additional elements that will be added for each iteration and the total number of elements for say, five iterations of the adaptive algorithm. Remember that this is a two-dimensional problem and that the data you are working with is only one of 116 such MRI sections. You should be able to get an idea of how the size of the problem increases with accuracy. Obviously, due to limited CPU, memory, and time, one has to make some choices concerning accuracy versus computational costs. One possible practical solution to this problem is to calculate the number of nodes that your particular workstation can handle and then refine only up to that particular number.

See exercise 13.

---

7 Selected Results

The figures on the preceding pages are some selected results on modeling bioelectric fields within the body due to the electrical activity of the heart. The figures depict various portions of the modeling and simulation process for the three-dimensional thorax model as described in [3, 4, 27, 39].

Figure 5 shows a collage of the different modeling aspects. Here we see a composite containing a triangulated surface of a lung, a portion of the finite element mesh, and the rendered heart, all superimposed upon the original MRI scan contained within a rendered portion of the torso surface.

Figure 6 shows a simulation using the inhomogeneous three-dimensional model. The

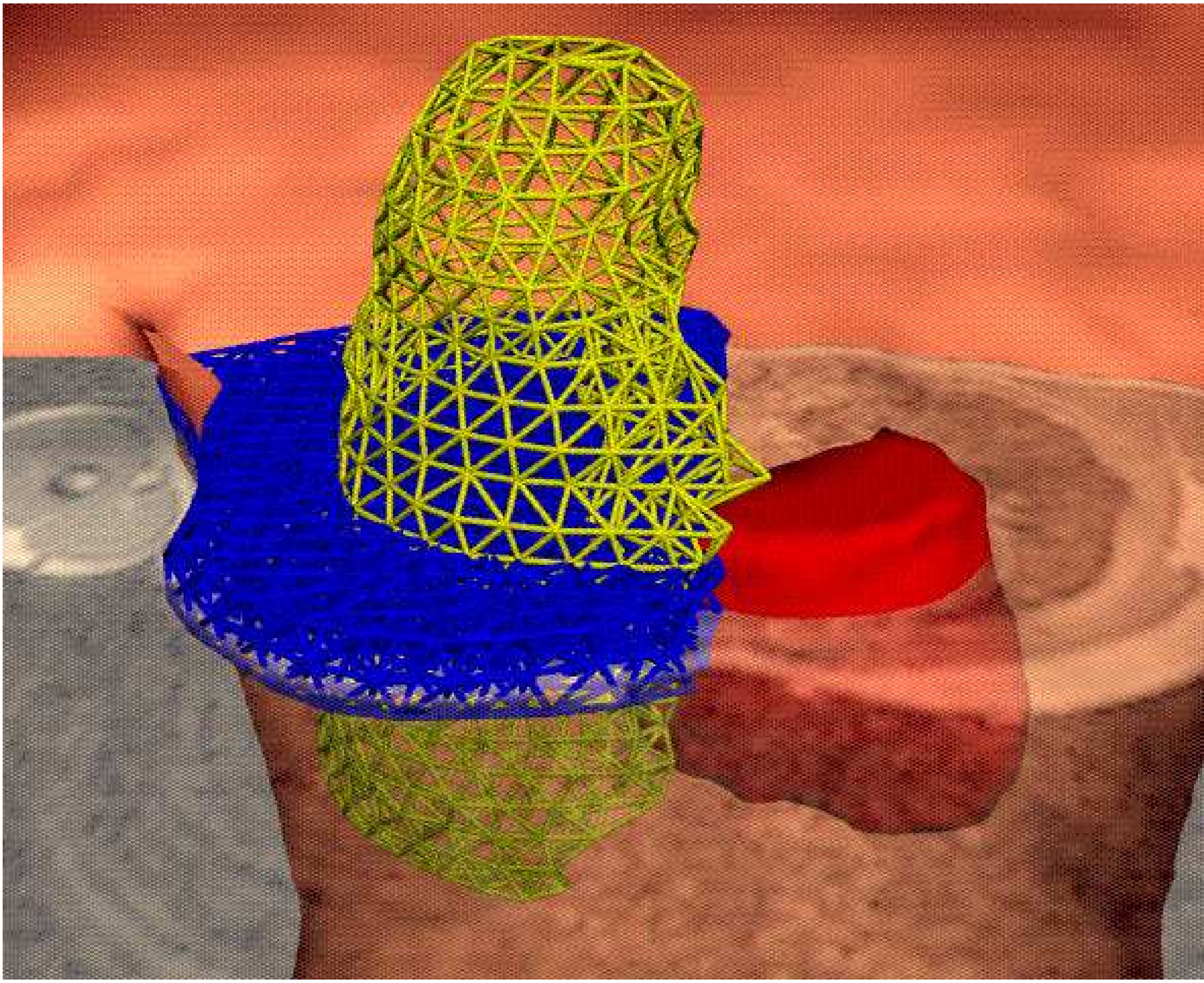

Figure 5: Collage of the different modeling aspects.

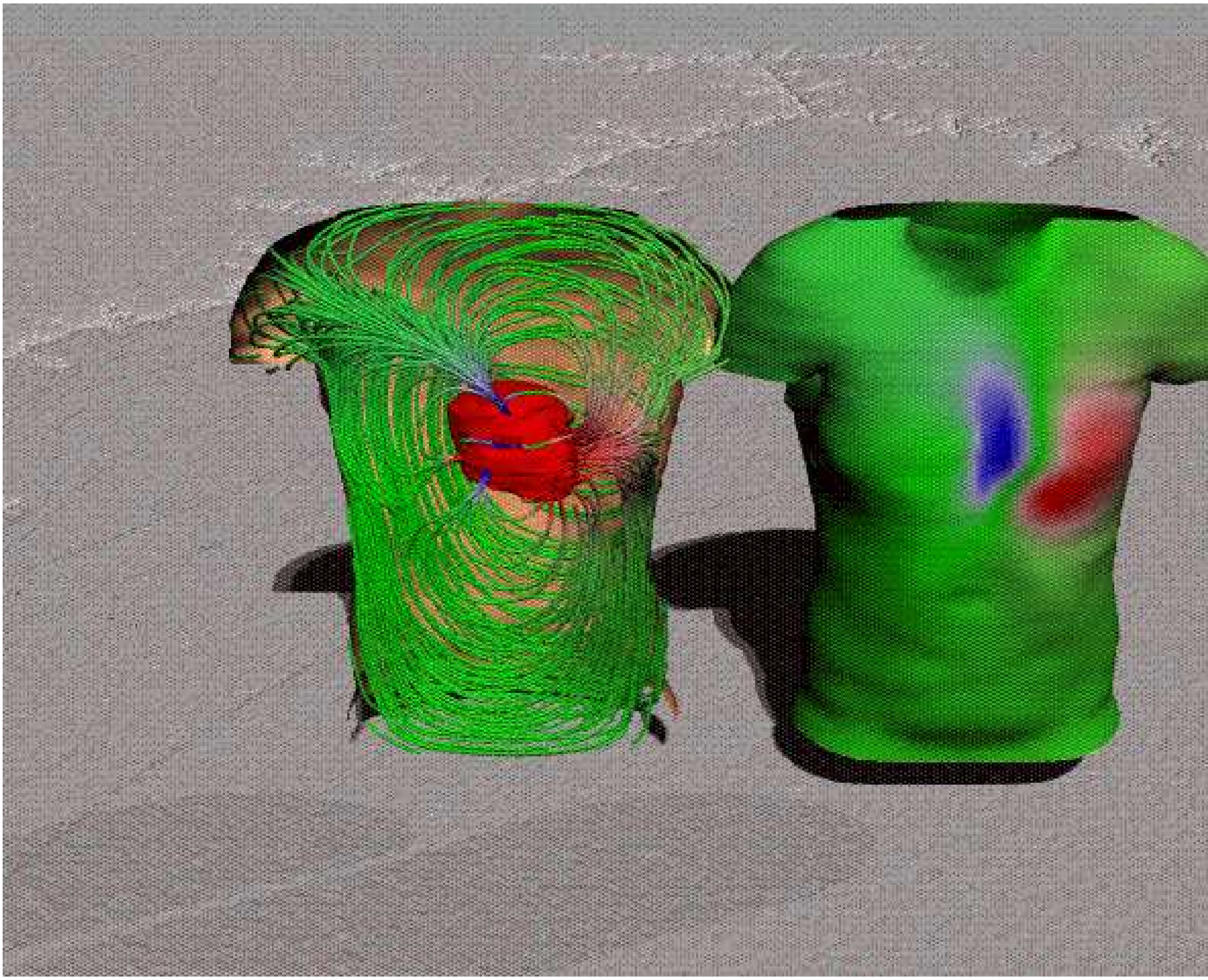

Figure 6: Simulation using the inhomogeneous three-dimensional model.

boundary conditions were derived from unipolar electrograms recorded from the heart surface during open chest surgery. The torso on the right shows a color-coded, rendered isovoltage map, while the torso on the left has been opened to reveal the electric current loops which form in the thorax.

8 Exercises

Exercise 1 *Effects of an ill-conditioned matrix.*

A simple demonstration of the effects of an ill-conditioned matrix on the accuracy of obtaining the inverse of a matrix - and thus the subsequent solution of a system - can be found by considering the following matrix:

$$A = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n-1} \end{pmatrix} \tag{87}$$

Obtain the subroutines *dgeco.f, dgedi.f, dgesl.f,* and *dsvdc.f* from the linpack directory of netlib. This can be accomplished by sending email to netlib@ornl.gov with the line, send filename.f from linpack, in the text section. Netlib will automatically send the code back to you in an email message.

Note: If you have not used netlib before, you might want to look at the available software and read the section on netlib in the Linear Algebra chapter of CSEP.

To continue, just send the message, send index from netlib, in your email text. Extract the programs from the email messages and compile them. To use the subroutines, you will need first to write a main program which defines the dimensions and calls the subroutines. Once the main program is in place, factor the matrix A using *dgeco.f.* This program will also estimate the condition number of the matrix. Using *dgesl.f*[*], solve a linear system, $Ax = b$ using the matrix A given above and a right hand side vector with each entry equal to 1.0. Save your answer. Now change the first entry in the right hand side from 1.0 to 1.1 and solve the system again. Try this for different sizes of n. Since this is a linear system, small changes in the inputs should yield small changes in the output. So what has happened? Use *dsvdc.f* on the matrix A and calculate the singular values. Take the ratio of the maximum to minimum singular value. This will give you another estimate of the condition number of

[*]Alternatively, you can do this excercise with Maple, Mathematica, Matlab, or a similar symbolic manipulation package

the matrix A. Compare this number with the one calculated by the program *dgeco.f*. The condition number tells you how many bits of information you can expect to lose during an inversion procedure. Given the condition number you calculated, what precision can you expect in the solution?

So what can we do now? The solution is so terrible because it does not depend continuously on the data, so small errors in the input can lead to large changes in the output. Methods which try to restore the solution back onto the data are called regularization methods [16, 17, 18].

Exercise 2 *Effects of regularization.*

As an exercise in regularization, calculate the singular values for the matrix A, where A is 4×4. Look at how the values are distributed and notice that there is a large jump between the first few values and the other values. Recalculate the solution of the system, this time using the SVD program, and zero out the entry that correspond to the smallest singular value.

You will find, perhaps to your surprise, that the solution now gives you reasonable results (at least more reasonable than your first, unregularized solution). What you have just done is a form of regularization known as truncated singular value decomposition (TSVD). In effect, you have applied a filter to the system, which filters out the high frequency *noise*. At the same time, you are also filtering out useful information. Basically, you are giving up some information for an added degree of smoothness of the solution. Unfortunately, this method doesn't work well for all ill-conditioned matrices (you might try to figure out for which cases the truncated SVD method would work well and for which cases it would not). Other methods have been developed which try to regularize the system, not by completely removing the singular values, but by modifying them so that they can contribute to the information content. Such methods were developed by the Russian mathematician Tikhonov [16].

Exercise 3 *Program for Tikhonov's methods.*

Now write a program that solves equation (9) using the same matrix A as in the previous problem. Try several different values for the regularization parameter α and compare these solutions with those you obtained using the singular value decomposition method. You might notice the similarity of equation (9) with the least squares solution ($\alpha = 0$). Try setting $\alpha = 0$ and see what happens (explain).

To investigate the regularization of ill-posed problems further, see [16, 17, 18, 19]. A particularly useful reference for discrete ill-posed problems is the Matlab package developed by Per Christian Hansen which is available via netlib [20].

Exercise 4 *Effect of the regularization parameter.*

To see just how ill-conditioned the transfer matrix in the previous equation, calculate the condition number of, A_transfer.dat, using the singular value decomposition (A_transfer.dat was constructed by using the finite element method, which will be described in detail later in the chapter). First solve equation (14) using all the singular values using the data files, Phi_H.dat and Phi_T.dat. Then solve the system by discarding the ones near zero (i.e. by using the TSVD approach). Next solve (14) using the Tikhonov regularization technique by varying the regularization parameter α.

As you continue solving the inverse problem for different regularization parameters, you might realize that you don't really know what you are looking for in terms of an answer (except to compare it to the SVD solution). Assuming we don't have a "true" solution to compare with, how is it then, that we know which regularization to choose *a priori*? This is a very good question and one that has intrigued (plagued) scientists and mathematicians working on inverse problems for years. While there has been much research into methods for choosing the best regularization parameter in an *a priori* manner, most schemes that have been put forth depend in some way on the data and/or constraints placed on the solution, and, thus, are problem dependent.

For an excellent overview of the problem of finding optimal a priori *estimates of the regularization parameter and some solutions, see [19].*

Exercise 5 *Generation of a finite element mesh.*

To familiarize yourself with the divide and conquer technique, use the x,y coordinates given in *mesh.pts* to generate a finite element mesh by subdividing the region into square sections. These points are derived from segmenting a transthoracic MRI scan for the epicardial and thorax boundaries. Usually one creates a master block and then subdivides each until a reasonable discretization level has been achieved (you might ask yourself what *reasonable* means here). Once you have obtained the subdivision into blocks, further subdivide the region into triangles (save the quadrilateral connectivities for a later exercise). You can print out the resulting mesh using the program *GridDraw* or view it using a postscript viewer such as Ghostscript. Save the triangle connectivities for later use with the finite element solver.

Exercise 6 *Challenges of nonuniform mesh generation.*

Now an example of the challenges of nonuniform mesh generation. Obtain the program *sweep2.c* from the *Voronoi* directory from netlib. Once you have received *sweep2.c*, extract it and compile it. Run *sweep2* with the -t option on the complete set of x,y grid points in *mesh.pts*. This will output the triangle connectivities needed for the finite element calculation. Look at the mesh you've created with *GridDraw*. You should notice that you have triangles that have been created outside (inside) the contours of interest. This is because, while the triangles are outside the contour, they are inside the convex hull of the point set, which is the property that *sweep2* uses to create the triangulation. Now write a program which extracts the triangles outside (or inside) the region of interest using the GB (or other) algorithm. There are other ways to extract the extra triangles. See if you can figure out a more computationally efficient method (for some problems, the process of extracting unwanted elements can take a significant amount of time). Once the extraneous triangles are removed, save the connectivities for later input into the finite element solver.

Exercise 7 *Finite difference program.*

Using the quadralaterals you created with the divide and conquer scheme in excercise 5 as a finite difference mesh. Write a two-dimensional finite difference program to solve (1).

Exercise 8 *Determination of the bandwidth and the non-zero values.*

The features of the matrix A (symmetric, positive definitive, and sparse), resulting from a FE formulation, significantly affect the strategy for solving the system of equations. Write a program to calculate the bandwidth of the matrix A in *A_transfer.dat* and estimate the minimum size of the matrix (assuming double precision reals) based upon your bandwidth estimate. Now write a program to calculate the number of nonzero entries in the matrix A. You should find that the proportion of nonzero to zero entries is quite small. Because of the "wasted space" all those zeros take up, methods have been developed to store the matrix in a form such that one doesn't ever have to store a zero.

Referencing the chapter on Numerical Linear Algebra and the references therein, write a routine to store the matrix A in a compressed format. Compare the size of the resulting size of the matrix (and/or vectors) with the original size.

Exercise 9 *2D FE solver.*

Write a 2D FE solver to solve (2) with the uniform mesh you created previously using the divide and conquer technique (a set of points, triangle connectivities, and boundary conditions are available in the data file, *regular_mesh_c.dat or regular_mesh_f.dat.* The difference in the two data files is that one starts indexing at 0 and the other at 1. Alternatively, you can use (modify) one of the FE solvers *fe2d_code.c or fel.f.* Compare it to the solution you obtain using the nonuniform mesh you created using the Delaunay technique. Continue to subdivide the uniform mesh and note whether or not your solution begins to converge to the solution you computed using the nonuniform mesh (why or why not?). Also compare the number of uniform and nonuniform elements it took to conform to the geometry.

Exercise 10 *Estimates of accuracy.*

In considering the solution you calculated using the finite element method in excercise 9, ask yourself the following difficult questions: "How accurate is my solution?" and "How do I figure out how to find out how accurate my solution is?" These embarrassingly simple, but tremendously important questions can be difficult to answer. The best way to answer the first question is to know what the answer should be, by comparing it with either an experimental result or an analytical solution. However, for the majority of cases there isn't an analytical solution and/or there are no experimental results. This is one reason why the field of computational science has become so important to so many research areas. In many instances, either experiments cannot be performed or it would be too expensive or time consuming to perform an experiment to confirm computational results. However, we still need to estimate the accuracy of our solutions - and while we're at it, figure out how we can make our solution more accurate.

The topic of providing accuracy estimates of numerical solutions and providing algorithms to automatically increase the accuracy of a solution has (and continues to be) of considerable interest and research. This type of research can be found under the heading *adaptive methods for partial differential equations*, and in many books on finite element methods. Before reading the brief introduction to adaptive methods given below, try to figure out how to estimate the accuracy of your numerical solution and give some thought to how you could increase the accuracy of the solution.

Exercise 11 *Some thoughts on errors.*

Before going on, try to figure out how you could utilize the results of the heuristic argument to identify elements which have large errors associated with them.

Exercise 12 *Error estimation program.*

Write a program which estimates the error in the potential field at each node using a simple energy norm estimator. Utilize a simple averaging procedure to identify which elements need to be refined (can you think of a better way to do this?). Add an extra node at the centroid of the element, which has been identified for refinement (Be careful about how you treat the elements which border a surface. Why?).

Use the *sweep2* program to re-triangulate the mesh. After removing the extraneous elements, calculate the FE solution and compare it to the original solution. As you continue the process, you are bound to notice a couple of things. One, after a few iterations, the amount of change in your solution from the previous solution will be minimal. Secondly, the number of degrees of freedom in your system will increase rapidly, such that you may soon not be able to calculate the solution on your workstation (especially if it has limited memory or you have limited time). This may mean that you are converging to the true solution (if you have done the refinements correctly).

Exercise 13 *Some thoughts on adaptive mesh generation.*

Another exercise involves fixing the number of nodes to the maximum number and then moving the nodes so that they are positioned in the optimal locations (the so-called *r* method or relocation method). Still another possibility is to use an de-refinement strategy, in which regions of the discretized domain that were too finely meshed in the original mesh can be "unrefined" and help balance the total number of degrees of freedom. At some point, however, for most large three-dimensional problems, one must face the option of parallelizing the algorithms so that they can be run on either a distributed system of high-end workstations or a massively parallel machine. The use of such techniques for adaptive mesh generation is just in its infancy, but is certainly a topic of considerable importance to researchers who work on large-scale finite element systems.

To find out more about parallel adaptive refinement methods and adaptive methods in general, see [45, 34, 39, 40, 42].

9 Project

Obtain the references, [36, 38] and/or others, and write a boundary element program (with linear elements) to solve the two-dimensional Laplace's equation (10) for the isotropic, homogeneous case using the data given in *regular_mesh.dat or irregular_mesh.dat*. As a continuation, derive a boundary element formulation for the isotropic, inhomogeneous case.

References

[1] C.E. Miller and C.S. Henriquez. Finite element analysis of bioelectric phenomena. *Crit. Rev. in Biomed. Eng.*, 18:181–205, 1990. This represents the first review paper on the use of the finite element method as applied to biomedical problems. As the authors note, bioengineers came to these methods only fairly recently, as compared to other engineers. It contains a good survey of applications.

[2] J. Nenonen, H.M. Rajala, and T. Katilia. *Biomagnetic Localization and 3D Modelling.* Helsinki University of Technology, Espoo, Finland, 1992. Report TKK-F-A689.

Biomagnetic Localization and 3D Modelling. Helsinki University of Technology, Espoo, Finland, 1992. Report TKK-F-A689. This is a collection of papers devoted to inverse magneto-encephalography (MEG) and magneto-cardiography (MCG) problems. While perhaps difficult to obtain, it contains several interesting papers on the subject.

[3] C.R. Johnson, R.S. MacLeod, and P.R. Ershler. A computer model for the study of electrical current flow in the human thorax. *Computers in Biology and Medicine*, 22(3):305–323, 1992. This is a paper by the author which details the construction of the three-dimensional model, part of which is described in the chapter.

[4] C.R. Johnson, R.S. MacLeod, and M.A. Matheson. Computer simulations reveal complexity of electrical activity in the human thorax. *Comp. in Physics*, 6(3):230–237, May/June 1992. Another paper by the author. This one deals with some of the computational and visualization aspects of the work. It contains some nice color figures (including one on the cover of the journal).

[5] O. Gandhi. *Biological Effects and Medical Applications of Electromagnetic Energy.* Prentice-Hall, Englewood Cliffs, 1990. This textbook surveys the subject of the effect of electromagnetic sources (powerlines, cellular phones, etc.) on the human body.

[6] Y. Kim, J.B. Fahy, and B.J. Tupper. Optimal electrode designs for electrosurgery. *IEEE Trans. Biomed. Eng.*, 33:845–853, 1986. This paper discusses an example of the modeling of bioelectric fields for applications in surgery.

[7] R. Plonsey. *Bioelectric Phenomena.* McGraw-Hill, New York, 1969. This is the first text using physics, mathematics, and engineering principles to quantify bioelectric phenomena.

[8] R.K. Hobbie. *Intermediate Physics for Medicine and Biology.* John Wiley and Sons, New York, 1988. This textbook is written for undergraduate science and engineering majors. It covers a wide variety of topics and contains a couple of chapters devoted to bioelectric phenomena.

[9] J.D. Murray. *Mathematical Biology.* Springer-Verlag, New York, 1989. A textbook aimed at first year graduate mathematics students interested in applying mathematics to biological problems.

[10] C.S. Henriquez and R. Plonsey. Simulation of propagation along a bundle of cardiac tissue. I. mathematical formulation. *IEEE Trans Biomed Eng*, 37:850–860, 1990. This paper and the companion paper below, describe the bidomain approach to the propagation of electrical signals through active cardiac tissue.

[11] C.S. Henriquez and R. Plonsey. Simulation of propagation along a bundle of cardiac tissue. II. results of simulation. *IEEE Trans Biomed Eng*, 37:861–887, 1990.

[12] J.P. Keener. Waves in excitable media. *SIAM J. Appl. Math.*, 46:1039–1056, 1980. This paper describes mathematical models for wave propagation in various excitable media. Both chemical and physiological systems are addressed.

[13] V.S. Zykov. *Modelling Wave Processes in Excitable Media.* Manchester University Press, Manchester, 1988. The first textbook devoted to the mathematical treatment of wave propagation in excitable media.

[14] J. Hadamard. Sur les problemes aux derivees parielies et leur signification physique. *Bull. Univ. of Princeton*, pages 49–52, 1902. This is Hadamard's original paper describing the concepts of well- and ill-posedness. In French.

[15] 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. *Medical Physics*, 17(3):369–379, 1990. This paper describes methods for constraining the inverse problem of electrocardiology in terms of sources. Methods are developed which put bounds on the space of acceptable solutions.

[16] A. Tikhonov and V. Arsenin. *Solution of Ill-posed Problems.* Winston, Washington, DC, 1977. This is the book in which Tikhonov describes his method of regularization for ill-posed problems.

[17] A.N. Tikhonov and A.V. Goncharsky. *Ill-Posed Problems in the Natural Sciences.* MIR Publishers, Moscow, 1987. This is a collection of research papers from physics, geophysics, optics, medicine, etc., which describe ill-posed problems and the solution techniques the authors have developed.

[18] V.B. Glasko. *Inverse Problems of Mathematical Physics.* American Institute of Physics, New York, 1984. This book has several introductory chapters on the mathematics of ill-posed problems followed by several chapters on specific applications.

[19] P.C. Hansen. Analysis of discrete ill-posed problems by means of the L-curve. *SIAM Review*, 34(4):561–580, 1992. This is an excellent review paper which describes the various techniques developed to solve ill-posed problems. Special attention is paid to the selection of the *a priori* approximation of the regularization parameter.

[20] P.C. Hansen. Regularization Tools: A Matlab package for analysis and solution of discrete ill-posed problems. Available via netlib in the library numeralgo/no4. This is an excellent set of tools for experimenting with and analyzing discrete ill-posed problems. The netlib library contains several Matlab routines as well as a postscript version of the accompanying technical report/manual.

[21] Y. Yamashita. Theoretical studies on the inverse problem in electrocardiography and the uniqueness of the solution. *IEEE Trans Biomed Eng*, 29:719–725, 1982. The first paper to prove the uniqueness of the inverse problem in electrocardiography.

[22] 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*, pages 154–171. Helsinki University of Technology, Helsinki, 1991. This paper describes the inverse MEG problem using the finite element method.

[23] I.M. Singer and J.A. Thorpe. *Lecture Notes on Elementary Topology and Geometry.* Springer-Verlag, New York, 1967. An excellent elementary textbook on topology and geometry.

[24] A Bowyer. Computing Dirichlet tesselations. *Computer J.*, 24:162–166, 1981. One of the first papers on the Delaunay triangulation in 3-space. If you read this paper, don't use Bowyer's algorithm for creating the triangulations as it in not computationally efficient.

[25] S.R.H. Hoole. *Computer-Aided Analysis and Design of Electromagnetic Devices.* Elsevier, New York, 1989. While the title wouldn't make you think so, this is an excellent introductory text on the use of numerical techniques to solve boundary value problems in electrodynamics. The text also contains sections on mesh generation and solution

methods. Furthermore, it provides pseudocode for most of the algorithms discussed throughout the text.

[26] R.S. MacLeod, C.R. Johnson, and M.A. Matheson. Visualization tools for computational electrocardiography. In *Visualization in Biomedical Computing*, pages 433–444, 1992. This paper, and the paper which follows, concern the visualization aspect of the research described in the chapter.

[27] R.S. MacLeod, C.R. Johnson, and M.A. Matheson. Visualization of cardiac bioelectricity — a case study. In *IEEE Visualization 92*, pages 411–418, 1992. See previous comment.

[28] 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, 1993. This edited collection of papers gives an overview of the state of the art in high performance computing (as of 1993) as it pertains to biomedical research. While certainly not comprehensive, the text does showcase several interesting applications and methods.

[29] 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*, pages 63–112. CRC Press, Boca Raton, 1993. This paper contains some extensions of Thompson's classic textbook on numerical grid generation.

[30] P.L. George. *Automatic Mesh Generation*. Wiley, New York, 1991. This is an excellent introduction to mesh generation. It contains a suvey of all the major mesh generation schemes.

[31] J.F. Thompson, Z.U.A. Warsi, and C.W. Mastin. *Numerical Grid Generation*. North-Holland, New York, 1985. This is the classic on mesh generation. The mathematical level is higher than that of George's book, and most of the applications are directed towards computational fluid dynamics.

[32] 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, 1994 (to appear). This paper describes the process of large scale modeling along with computational and visualization issues pertaining to bioelectric field problems.

[33] J.E. Akin. *Finite Element Analysis for Undergraduates*. Academic Press, New York, 1986. This is an easy to read, self contained text on the finite element method aimed at undergraduate engineering students.

[34] C. Johnson. *Numerical solution of Partial Differential Equations by the Finite Element Method*. Cambridge University Press, Cambridge, 1990. An excellent introductory book on the finite element method. The text assumes mathematical background of a first year graduate students in applied mathematics and computer science. An excellent introduction to the theory of adaptive methods.

[35] J.D. Jackson. *Classical Electrodynamics*. John Wiley, New York, 1975. One of the classic books on electrodynamics and the standard graduate textbook for many first year graduate physics courses on the subject.

[36] C.A. Brebbia and J. Dominguez. *Boundary Elements: An Introductory Course*. McGraw-Hill, Boston, 1989. This is an introductory book on the boundary element method by one of the foremost experts on the subject (C.A.B.).

[37] M.A. Jawson and G.T. Symm. *Integral Equation Methods in Potential Theory and Elastostatics*. Academic Press, London, 1977. An introduction to the boundary integral method as applied to potential theory and elastostatics.

[38] G. Beer and J.O. Watson. *Introduction to Finite and Boundary Element Methods for Engineers*. Wiley, New York, 1992. This is an excellent first book for those wishing to learn about the practical aspects of the numerical solution of boundary value problems. The book covers not only finite and boundary element methods, but also sections on mesh generation and the solution of large systems.

[39] C.R. Johnson and R.S. MacLeod. Nonuniform spatial mesh adaption using a posteriori error estimates: applications to forward and inverse problems. *Applied Numerical Mathematics*, **vol. 14**, pp. 331-326, 1994. This is a paper by the author which describes the application of the h-method of mesh refinement for large scale two- and three-dimensional bioelectric field problems.

[40] J.E. Flaherty. *Adaptive Methods for Partial Differential Equations*. SIAM, Philadelphia, 1989. This is a collection of papers on adaptive methods, many by the founders in the field. Most of the papers are applied to adaptive methods for finite element methods and deal with both theory and applications.

[41] G.H. Golub, and C.F. Van Loan. *Matrix Computations*. Johns Hopkins, Baltimore, 1989. This is a classic reference for matrix computations and is highly recommended.

[42] O.C. Zienkiewicz. *The Finite Element Method in Engineering Science*. McGraw-Hill, New York, 1971. This is a classic text on the finite element method. It is now in its fourth edition, 1991.

[43] 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. This is a classic paper which describes the use of the energy norm to globally refine the mesh based upon *a priori* error estimates.

[44] O.C. Zienkiewicz and J.Z Zhu. Adaptivity and mesh generation. *Int. J. Num. Meth. Eng.*, **32**, 783-810, 1991. This is another good paper on adaptive methods which describes some more advanced methods than their 1987 paper.

[45] D.S. Burnett *Finite Element Method.* Addison Wesley, Reading, Mass., 1988. This is an excellent introduction to the finite element method. It covers all the basics and introduces more advanced concepts in an readily understandable context.

[46] C.B. Barber, D.P. Dobkin, and H. Huhdanpaa The quickhull algorithm for convex hull. Geometry Cneter Technical Report GCG53. A public domain two- and three-dimensional Delaunay mesh generation code. Available via anonymous ftp from: geom.umn.edu:/pub/software/qhull.tar.Z. There is also a geometry viewer available from geom.umn.edu:/pub/software/geomview/geomview-sgi.tar.Z.