

ADAPTIVE MULTILEVEL FINITE ELEMENT SOLUTION OF THE POISSON-BOLTZMANN EQUATION I: ALGORITHMS AND EXAMPLES

M. HOLST, N. BAKER, AND F. WANG

ABSTRACT. This paper is the first of two papers on the adaptive multilevel finite element treatment of the nonlinear Poisson-Boltzmann equation (PBE), a nonlinear elliptic equation arising in biomolecular modeling. Fast and accurate numerical solution of the PBE is usually difficult to accomplish, due to presence of discontinuous coefficients, delta functions, three spatial dimensions, unbounded domain, and rapid (exponential) nonlinearity. In this first paper, we explain how adaptive multilevel finite element methods can be used to obtain extremely accurate solutions to the PBE with very modest computational resources, and we present some illustrative examples using two well-known test problems. The PBE is first discretized with piecewise linear finite elements over a very coarse simplex triangulation of the domain. The resulting nonlinear algebraic equations are solved with global inexact-Newton methods, which we have described in a paper appearing previously in this journal [55]. *A posteriori* error estimates are then computed from this discrete solution, which then drives a simplex subdivision algorithm for performing adaptive mesh refinement. The *discretize-solve-estimate-refine* procedure is then repeated, until a nearly uniform solution quality is obtained. The sequence of unstructured meshes is used to apply multilevel methods in conjunction with global inexact-Newton methods, so that the cost of solving the nonlinear algebraic equations at each step approaches optimal $O(N)$ linear complexity. All of the numerical procedures are implemented in *Manifold Code* (MC), a computer program designed and built by the first author over several years at Caltech and UC San Diego. MC is designed solve a very general class of nonlinear elliptic equations on complicated domains in two and three dimensions. We describe some of the key features of MC, and give a detailed analysis of its performance for two model PBE problems, with comparisons to the alternative methods. It is shown that the best available uniform-mesh-based finite difference or box-method algorithms, including multilevel methods, require substantially more time to reach a target PBE solution accuracy than the adaptive multilevel methods in MC. In the second paper [6], we develop an error estimator based on geometric solvent accessibility, and present a series of detailed numerical experiments for several complex biomolecules.

1. INTRODUCTION

This paper is the first in a series of two papers on the adaptive multilevel finite element treatment of the nonlinear Poisson-Boltzmann equation (PBE), a second order nonlinear partial differential equation (PDE) which is fundamental to Debye-Hückel continuum electrostatic theory [40]. The PBE determines a dimensionless potential $u(x) = e_c\Phi(x)/(k_B T)$ around a charged biological structure immersed in a salt solution, where $\Phi(x)$ is the electrostatic potential at $x \in \mathbb{R}^d$, with $d = 2$ or $d = 3$. For a 1 : 1 electrolyte, the PBE can be written as

$$(1.1) \quad -\nabla \cdot (\epsilon(x)\nabla u(x)) + \bar{\kappa}^2(x) \sinh(u(x)) = \frac{4\pi e_c^2}{k_B T} \sum_{i=1}^{N_m} z_i \delta(x - x_i), \quad u(\infty) = 0.$$

In this equation, the dielectric ϵ jumps by one or two orders of magnitude at the interface between the charged structure and the solvent, and the modified Debye-Hückel parameter $\bar{\kappa}$ (a function of the ionic strength of the solvent) is discontinuous at an ion exclusion region surrounding the

1991 *Mathematics Subject Classification*. Primary 35B40, 35Q30; Secondary 65D05, 65M60.

Key words and phrases. Poisson-Boltzmann equation, adaptive finite element methods, *a posteriori* error estimation, biomolecules, electrostatics.

biological structure. The structure itself (e.g., a biological molecule, or a membrane) is represented implicitly by ϵ and $\bar{\kappa}$, as well as explicitly by the N_m point charges $q_i = z_i e_c$ at the positions \bar{x}_i (with the distributions $\delta(\cdot)$ reflecting the point-charge behavior).

The importance of this equation in biomolecular modeling is well-established; cf. [31, 80] for thorough discussions. For a derivation of the equation from first principles and for various analytical results, see [82, 51]. Some analytical solutions are known, but only for unrealistic structure geometries, and usually only for linearizations of the equation; cf. [51] for a collection of these solutions, and for references to the large amount of literature on analytical solutions to the PBE and similar equations. References [51, 58] contain some (non-constructive) existence and uniqueness results for the nonlinear PBE in a general setting. References [58, 59] also contain a finite element approximation theory framework for the PBE, including both *a priori* and *a posteriori* error estimates for driving the adaptive numerical methods appearing in this paper.

The various features of (1.1) which cause difficulties in analytical approaches, namely the infinite domain, delta functions, discontinuities in $\epsilon(x)$ and $\bar{\kappa}(x)$, and rapid nonlinearity, also make for a formidable numerical problem. A number of papers have appeared in the literature in the last ten years presenting various numerical techniques for approximating the solution to (1.1) with widely varying degrees of accuracy and efficiency. Uniform-mesh finite difference solutions of linearizations of (1.1), as well as some integral equation-based solutions (necessarily restricted to linearizations), include [38, 46, 61, 71, 72, 80, 92]. Results with uniform-mesh finite difference methods applied to the nonlinear problem (1.1) have also appeared, including [2, 46, 55, 60, 67, 71, 73, 80, 79]. In [55, 51], a global inexact-Newton approach was combined with a fast linear multilevel method, which proved to be an extremely robust and efficient method for the nonlinear algebraic equations produced by finite difference or box-method discretizations of (1.1).

A non-adaptive, single-level finite element method, based on simplices in three dimensions (tetrahedra), was recently applied to the linearized PBE in [36, 37], employing a preconditioned conjugate gradient (PCG) method for the solution of the resulting linear algebraic systems. In a related paper [44], the linearized PBE fields are used together with an analytical technique to produce gradients for dynamical calculations. As reported in [37], an improvement of about a factor of two in solution time was observed, when compared to the time taken by an efficient finite difference uniform-mesh approach (DelPhi's linear SOR solver) to reach a solution of the same discretization accuracy. The authors of [36, 37] were able to achieve this factor of two reduction in solution time by taking advantage of the natural support for unstructured meshes inherent in finite element methods; mesh points were placed *a priori* more densely around the molecular surface, reducing the total number of mesh points required to achieve a desired accuracy when compared to a uniform-mesh approach. A simplex mesh generation technique (constrained incremental point-insertion Delaunay) was then applied to produce a set of simplices which employed the mesh points as vertices. A two-level solution procedure was also attempted in [37] as an alternative to PCG, but was unsuccessful due to the various difficult features of the PBE alluded to earlier; cf. [51] for an analysis of the difficulties. In this paper, we will show that in fact an order of magnitude or more reduction in solution time (compared to uniform-mesh approaches) are possible for finite element methods on unstructured simplex meshes, when robust *a posteriori* error estimation is used to drive adaptive mesh refinement, and when appropriate multilevel methods are used for the resulting discrete systems. Moreover, we will show how the full nonlinear problem (1.1) can be handled as well, by combining these adaptive finite element methods with the global inexact-Newton procedures described in [55]. Such a fully adaptive multilevel finite element solution strategy for two-dimensional nonlinear elliptic problems in the plane has been implemented in R. Bank's PLTMG software [9], and has been publicly available for a number of years. However, only recently have (non-commercial) codes with similar capabilities for three-dimensional nonlinear problems appeared [50, 70, 25, 26, 24].

An attempt to use adaptive finite element methods and software from the structural engineering community for the Poisson-Boltzmann equation appears in [27]. While their approach allowed for the nonlinearities in the Poisson-Boltzmann equation, they considered only axi-symmetric situations so that the problem reduced to a two-dimensional case. Moreover, the error estimator they employed (the well-known ZZ-estimator from the finite element literature) involves only gradient recovery, meaning that the estimator only indicates where the gradients in the potential are changing rapidly. The residual estimator we employ estimates not only the gradient behavior but also boundary error as well as the error in representing the charge density term in the Poisson-Boltzmann equation.

We will discretize equation (1.1) with piecewise linear ($\mathcal{P}1$) finite elements over a simplex tessellation of a truncated solvent-filled sphere around the biomolecule, producing an implicitly defined set of nonlinear algebraic equations. (The local $O(h^2)$ accuracy of the discretization with $\mathcal{P}1$ elements is comparable to the usual finite difference or box-methods, where the edge lengths of the simplices in the finite element mesh are comparable to the cell sizes in a finite difference mesh.) These nonlinear algebraic equations can be *evaluated* by traversing the simplex mesh, but can not be formed explicitly. A linearization for performing Newton-like iterations is formed explicitly as a sparse matrix. The nonlinear algebraic equations are solved with global inexact-Newton-multilevel methods, which we have described in a paper appearing previously in this journal. (These Newton-multilevel methods have provably optimal complexity in certain situations, in that the number of arithmetic operations performed to reach a converged solution is a constant multiple of the number of vertices in the simplex mesh; cf. [55, 51].) Such discrete solutions obtained on initial coarse simplex meshes are used together with *a posteriori* error estimation to drive adaptive refinement algorithms, refining simplices into smaller simplices only when needed for obtaining a target solution accuracy within each element. After several solve-estimate-refine steps, a nearly uniform distribution of discretization error is obtained, so that the target solution accuracy is reached with a nearly minimal number of discretization points. Combined with the optimal complexity methods for obtaining the solutions on each mesh, the result is an extremely efficient procedure for obtaining a final discrete solution with a desired accuracy.

All of the numerical procedures are implemented in Manifold Code (MC), a computer program designed by one of the authors (M.H.) over several years at Caltech and UC San Diego. MC is designed to solve a general class of nonlinear elliptic equations on complicated 2- and 3-dimensional domains [50], a general class of problems which includes the PBE. MC employs *a posteriori* error estimation, adaptive simplex subdivision, unstructured algebraic multilevel methods, global inexact-Newton methods, and numerical continuation methods, for the highly accurate numerical solution of this class of problems. We describe some of the key features of MC, and give a detailed analysis of its performance for two model PBE problems, with comparisons to the alternative methods. It is shown that the best available uniform-mesh-based finite difference or box-method algorithms, including multilevel methods, require substantially more time and memory to reach a target solution accuracy than the time required by MC to reach the same solution accuracy.

Outline of the paper. In §2, we derive a weak formulation of the nonlinear PBE, identifying a nonlinear integral *form*, or *functional*, representing the equation, and a bilinear form representing a linearization. Our finite element approximations will operate entirely on these two forms. In §3, we outline our finite element approximation approach, employing simplex-based piecewise-linear functions on simplex triangulations of finite regions of solvent containing the biological molecules of interest. Global inexact-Newton multilevel algorithms for solution of the resulting discrete equations are then also described entirely in terms of the two weak forms. The error estimation and adaptive simplex subdivision procedures we employ are then discussed. In §4, the MC software is described, including some of its special features. The key geometry datastructure, the “ringed vertex” is presented, and the implementation of mesh refinement, residual and matrix assembly, and other operations employing this datastructure, are outlined. Some of the other features of MC are also

described. In §5, we apply MC to two model PBE problems, and compare the results to those obtainable with the standard non-adaptive methods currently in use. We summarize the results in §6. In the second paper [6], we develop an error estimator based on geometric solvent accessibility, and present a series of detailed numerical experiments for several complex biomolecules.

2. WEAK FORMULATION OF THE NONLINEAR POISSON-BOLTZMANN EQUATION

In this section we will first give an overview of weak formulations of nonlinear elliptic equations, as required for the use of finite element methods. Some standard references for this material are [45, 48]. We finish the section with a derivation of a nonlinear weak formulation of the PBE, along with an associated bilinear linearization form, both of which are required to use MC to produce an adaptive finite element approximation. This material can be found in [51, 52].

2.1. Nonlinear elliptic equations, weak forms, and linearizations. Let $\Omega \subset \mathbb{R}^d$ be an open set, and let $\partial\Omega$ denote the boundary, which can be thought of as a set in \mathbb{R}^{d-1} . Consider now the following nonlinear scalar elliptic equation on Ω , a class of elliptic equations of which the PBE is an example:

$$(2.1) \quad -\nabla \cdot (a(x)\nabla u(x)) + b(x, u(x)) = 0 \text{ in } \Omega,$$

$$(2.2) \quad u(x) = g(x) \text{ on } \partial\Omega,$$

where

$$(2.3) \quad a : \Omega \mapsto \mathbb{R}^{3 \times 3}, \quad b : \Omega \times \mathbb{R} \mapsto \mathbb{R}, \quad g : \partial\Omega \mapsto \mathbb{R}.$$

The above form of the equation is sometimes referred to as the *strong form*, in that the solution is required to be twice differentiable for the equation to hold in the classical sense. To produce a weak formulation, which is more suitable for finite element methods in that it will require that fewer derivatives of the solution exist in the classical sense, we first choose *trial* and *test* spaces of functions. For second order scalar elliptic equations of this form, the appropriate trial and test space can be seen to be $H^1(\Omega)$, the *Sobolev* space of functions which are differentiable one time under the integral. This function space is simply the set of all scalar-valued functions over the domain Ω for which the following integral (the energy norm, or H^1 -norm) is always finite:

$$(2.4) \quad \|u\|_{H^1(\Omega)} = \left(\int_{\Omega} |\nabla u|^2 + |u|^2 dx \right)^{1/2}.$$

In other words, the set (or space) of functions $H^1(\Omega)$ is defined as:

$$(2.5) \quad H^1(\Omega) = \{ u : \|u\|_{H^1(\Omega)} < \infty \}.$$

A closely related function space is that of square-integrable functions:

$$(2.6) \quad L^2(\Omega) = \{ u : \|u\|_{L^2(\Omega)} < \infty \}, \quad \text{where } \|u\|_{L^2(\Omega)} = \left(\int_{\Omega} |u|^2 dx \right)^{1/2}.$$

(To be precise, all integrals here and below must be interpreted in the Lebesgue rather than Riemann sense, but we will ignore the distinction here.) It is important that the trial and test spaces satisfy a zero boundary condition on the boundary $\partial\Omega$ on which the Dirichlet boundary condition (2.2) holds, so that in fact we choose the following subspace of $H^1(\Omega)$:

$$(2.7) \quad H_0^1(\Omega) = \{ u \in H^1(\Omega) : u = 0 \text{ on } \partial\Omega \}.$$

Now, multiplying our elliptic equation by a test function $v \in H_0^1(\Omega)$ produces:

$$(2.8) \quad \int_{\Omega} (-\nabla \cdot (a\nabla u) + b(x, u)) v dx = 0,$$

which becomes, after applying a generalized form of Green's integral identities,

$$(2.9) \quad \int_{\Omega} (a\nabla u) \cdot \nabla v \, dx - \int_{\partial\Omega} v(a\nabla u) \cdot n \, ds + \int_{\Omega} b(x, u)v \, dx = 0.$$

Note that the boundary integral above vanishes due to the fact that the test function v vanishes on the boundary.

If the boundary function g is smooth enough, there is a mathematical result called the Trace Theorem [1] which guarantees the existence of a function $\bar{u} \in H^1(\Omega)$ such that $g = \bar{u}|_{\partial\Omega}$. (We will not be able to construct such a *trace* function \bar{u} in practice, but we will be able to approximate it as accurately as necessary in order to use this weak formulation in finite element methods.) Employing such a function $\bar{u} \in H^1(\Omega)$, it is easily verified that the solution u to the problem (2.1)–(2.2), if one exists, lies in the space of functions $\bar{u} + H_0^1(\Omega)$.

We have therefore shown that the solution to the original problem (2.1)–(2.2) also solves the following problem:

$$(2.10) \quad \text{Find } u \in \bar{u} + H_0^1(\Omega) \text{ such that } \langle F(u), v \rangle = 0 \quad \forall v \in H_0^1(\Omega),$$

where from equation (2.9) the scalar-valued function of u and v (also called a *form*), nonlinear in u but linear in v , is defined as:

$$(2.11) \quad \langle F(u), v \rangle = \int_{\Omega} (a\nabla u \cdot \nabla v + b(x, u)v) \, dx.$$

Clearly, the “weak” formulation of the problem given by equation (2.10) imposes only one order of differentiability on the solution u , and only in the weak sense (under an integral). Under suitable growth restrictions on the nonlinearities b and c , it can be shown that this weak formulation makes sense, in that the form $\langle F(\cdot), \cdot \rangle$ is finite for all arguments. Moreover, it can be shown under somewhat stronger assumptions that the weak formulation is well-posed, in that there exists a unique solution depending continuously on the problem data. These types of results for the PBE are shown in [51, 58, 59].

To apply a Newton iteration to solve this nonlinear problem numerically, we will need a linearization of some sort. Rather than discretize and then linearize the discretized equations, we will exploit the fact that with projection-type discretizations such as the finite element method, these operations actually commute; we can first linearize the differential equation, and then discretize the linearization in order to employ a Newton iteration. To linearize the weak form operator $\langle F(u), v \rangle$, a bilinear linearization form $\langle DF(u)w, v \rangle$ is produced as its directional (variational, Gateaux) derivative as follows:

$$(2.12) \quad \begin{aligned} \langle DF(u)w, v \rangle &= \left. \frac{d}{dt} \langle F(u + tw), v \rangle \right|_{t=0} = \left. \frac{d}{dt} \int_{\Omega} (a\nabla(u + tw) \cdot \nabla v + b(x, (u + tw))v) \, dx \right|_{t=0} \\ &= \int_{\Omega} \left(a\nabla w \cdot \nabla v + \frac{\partial b(x, u)}{\partial u} wv \right) \, dx. \end{aligned}$$

This scalar-valued function of the three arguments u, v , and w , is linear in w and v , but possibly nonlinear in u . For fixed (or frozen) u , it is referred to as a bilinear form (linear in each of the remaining arguments w and v). We will see shortly that the nonlinear weak form $\langle F(u), v \rangle$, and the associated bilinear linearization form $\langle DF(u)w, v \rangle$, together with a continuous piecewise polynomial subspace of the solution space $\bar{u} + H_0^1(\Omega)$, are all that are required to employ the finite element method for numerical solution of the original elliptic equation.

2.2. The Poisson-Boltzmann equation: weak form and linearization. In the case of the PBE on a truncated finite domain $\Omega \subset \mathbb{R}^3$, the strong form is:

$$(2.13) \quad -\nabla \cdot (\epsilon(x)\nabla u(x)) + \bar{\kappa}^2(x) \sinh(u(x)) - \left(\frac{4\pi e_c^2}{k_B T}\right) \sum_{i=1}^{N_m} z_i \delta(x - x_i) = 0 \quad \text{in } \Omega,$$

$$(2.14) \quad u(x) = g(x) \quad \text{on } \partial\Omega.$$

The boundary function $g(x)$ is usually taken to be induced by a known analytical solution to one of several possible simplifications of the linearized PBE. Far from the molecule, such analytical solutions provide a highly-accurate boundary condition approximation for the general nonlinear PBE on a truncation of \mathbb{R}^3 . Another approach to handling the boundary condition more elegantly with adaptive finite element methods is to use “infinite” elements, which we don’t consider in the present paper.

The corresponding weak formulation then follows from the previous discussion, where $a(x) = \epsilon(x)I_{3 \times 3}$, $b(x, u) = \bar{\kappa}^2(x) \sinh(u) - (4\pi e_c^2 / (k_B T)) \sum_{i=1}^{N_m} z_i \delta(x - x_i)$. For technical reasons, we assume that the sum of delta functions appearing in (2.13) is approximated by a sum of square integrable functions, which we denote as $f \in L^2(\Omega)$. This leads to:

$$(2.15) \quad \text{Find } u \in \bar{u} + H_0^1(\Omega) \text{ such that } \langle F(u), v \rangle = 0 \quad \forall v \in H_0^1(\Omega),$$

where

$$(2.16) \quad \langle F(u), v \rangle = \int_{\Omega} (\epsilon \nabla u) \cdot \nabla v + \bar{\kappa}^2 \sinh(u)v - f v \, dx.$$

The bilinear linearization form $\langle DF(u)w, v \rangle$ in the case of the PBE is again produced as the Gateaux derivative of the nonlinear form $\langle F(u), v \rangle$ above:

$$(2.17) \quad \langle DF(u)w, v \rangle = \left. \frac{d}{dt} \langle F(u + tw), v \rangle \right|_{t=0} = \int_{\Omega} (\epsilon \nabla w \cdot \nabla v + \cosh(u)wv) \, dx.$$

3. MULTILEVEL ADAPTIVE FINITE ELEMENT METHODS

3.1. The finite element method for nonlinear elliptic equations. A *Galerkin* approximation of the solution to (2.15) or (2.10) is the solution to the following subspace problem:

$$(3.1) \quad \text{Find } u_h \in \bar{u}_h + V_h \subset H_0^1(\Omega) \text{ such that } \langle F(u_h), v_h \rangle = 0, \quad \forall v_h \in V_h \subset H_0^1(\Omega),$$

for some chosen subspace V_h of $H_0^1(\Omega)$, and some approximation \bar{u}_h to \bar{u} . A *finite element* method is simply a Galerkin approximation method in which the subspace V_h is chosen to have the extremely simple form of continuous piecewise low-degree polynomials with local support, defined over a disjoint covering of the domain Ω by *elements*, typically simple polyhedra. For example, in the case of continuous piecewise linear polynomials defined over a disjoint covering with 2- or 3-simplices (cf. Figure 1), the basis functions are easily defined element-wise using the unit 2-simplex (triangle) and unit 3-simplex (tetrahedron) as follows:

$$(3.2) \quad \begin{array}{ll} \tilde{\phi}_0(\tilde{x}, \tilde{y}) & = 1 - \tilde{x} - \tilde{y} & \tilde{\phi}_0(\tilde{x}, \tilde{y}, \tilde{z}) & = 1 - \tilde{x} - \tilde{y} - \tilde{z} \\ \tilde{\phi}_1(\tilde{x}, \tilde{y}) & = \tilde{x} & \tilde{\phi}_1(\tilde{x}, \tilde{y}, \tilde{z}) & = \tilde{y} \\ \tilde{\phi}_2(\tilde{x}, \tilde{y}) & = \tilde{y} & \tilde{\phi}_2(\tilde{x}, \tilde{y}, \tilde{z}) & = \tilde{x} \\ & & \tilde{\phi}_3(\tilde{x}, \tilde{y}, \tilde{z}) & = \tilde{z} \end{array} .$$

For a simplex mesh with n vertices, a set of n *global* basis functions $\{\phi_i(x, y, z), i = 1, \dots, n\}$ are defined, as in the right-most picture in Figure 1, by combining the support regions around a given vertex, and extending the unit simplex basis functions to each arbitrary simplex using (affine) coordinate transformations.

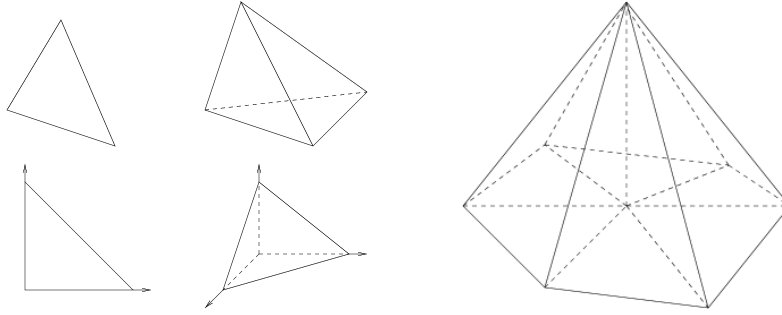


FIGURE 1. Reference and arbitrary 2- and 3-simplex elements, and a global (2D) basis function.

The above basis functions clearly do not form any subspace of $\mathcal{C}^2(\Omega)$, the space of twice continuously differentiable functions on Ω , which is the natural function space in which to look for the solutions to second order elliptic equations written in the strong form, such as equation (2.1). This is due to the fact that they are discontinuous along simplex faces and simplex vertices in the disjoint simplex covering of Ω . However, one can show that in fact [34]:

$$(3.3) \quad \text{span}\{\phi_1, \dots, \phi_n\} \subset H_0^1(\Omega), \quad \Omega \subset \mathbb{R}^d,$$

so that these continuous, piecewise defined, low-order polynomial spaces do in fact form a subspace of the solution space to the weak formulation of the problems we are considering here. Taking then $V_h = \text{span}\{\phi_1, \phi_2, \dots, \phi_n\}$, equation (3.1) reduces to a set of n nonlinear algebraic relations (implicitly defined) for n coefficients $\{\alpha_j\}$ in the expansion

$$(3.4) \quad u_h = \sum_{j=1}^n \alpha_j \phi_j.$$

In particular, regardless of the complexity of the form $\langle F(u), v \rangle$, as long as we can evaluate it for given arguments u and v , then we can evaluate the nonlinear discrete residual of the finite element approximation u_h as:

$$(3.5) \quad r_i = \langle F(\sum_{j=1}^n \alpha_j \phi_j), \phi_i \rangle, \quad i = 1, \dots, n.$$

Since the form $\langle F(u), v \rangle$ involves an integral in this setting, if we employ quadrature then we can simply sample the integrand at quadrature points; this is a standard technique in finite element technology. Given the local support nature of the functions ϕ_j , all but a small constant number of terms in the sum $\sum_{j=1}^n \alpha_j \phi_j$ are zero at a particular spatial point in the domain, so that the residual r_i is inexpensive to evaluate when quadrature is employed.

The two primary issues in applying this approximation method are then:

- (1) The approximation error $\|u - u_h\|_X$, for various norms X ,
- (2) The computational complexity of solving the n algebraic equations.

The first of these issues represents the core of finite element approximation theory, which itself rests on the results of classical approximation theory. Classical references to both topics include [34, 42, 39]. The second issue is addressed by the complexity theory of direct and iterative solution methods for sparse systems of linear and nonlinear algebraic equations, cf. [49, 85].

3.2. Error estimation, adaptive simplex subdivision, and conformity. *A priori* error analysis for the finite element method addressing the first issue is a well-understood subject [30, 34].

Much activity has recently been centered around *a posteriori* error estimation, and its use in adaptive mesh refinement algorithms [5, 4, 9, 86, 87, 91]. These estimators include weak and strong residual-based estimators [5, 4, 86], as well as estimators based on solving local problems [17, 19]. The challenge for a numerical method is to be as efficient as possible, and *a posteriori* estimates are a basic tool in deciding which parts of the solution require additional attention. While the majority of the work on *a posteriori* estimates has been for linear problems, nonlinear extensions are possible through linearization theorems (cf. [86, 87]).

The solve-estimate-refine structure in simplex-based adaptive finite element codes such as MC [50] and PLTMG [9], exploiting these *a posteriori* estimators, is as follows:

Algorithm 3.1. (*Adaptive multilevel finite element approximation*)

- While ($\|u - u_h\|_X > \epsilon$) do:
 - (1) Find $u_h \in \bar{u}_h + V_h \subset H_0^1(\Omega)$ such that $\langle F(u_h), v_h \rangle = 0, \forall v_h \in V_h \subset H_0^1(\Omega)$.
 - (2) Estimate $\|u - u_h\|_X$ over each element.
 - (3) Initialize two temporary simplex lists as empty: $Q1 = Q2 = \emptyset$.
 - (4) Place simplices with large error on the “refinement” list $Q1$.
 - (5) Bisect all simplices in $Q1$ (removing them from $Q1$),
and place any nonconforming simplices created on the list $Q2$.
 - (6) $Q1$ is now empty; set $Q1 = Q2, Q2 = \emptyset$.
 - (7) If $Q1$ is not empty, goto (5).
- End While.

The conformity loop (5)–(7), required to produce a globally “conforming” mesh (described below) at the end of a refinement step, is guaranteed to terminate in a finite number of steps (cf. [74, 75]), so that the refinements remain local. Element shape is crucial for approximation quality; the bisection procedure in step (5) is guaranteed to produce nondegenerate families if the longest edge is bisected in two dimensions [76, 81], and if marking or homogeneity methods are used in three dimensions [3, 70, 23, 22, 64, 68]. Whether longest edge bisection is nondegenerate in three dimensions apparently remains an open question.

Figure 2 shows a single subdivision of a 2-simplex or a 3-simplex using either 4-section (first figure from left), 8-section (fourth figure from left), or bisection (third and sixth figures from left). The

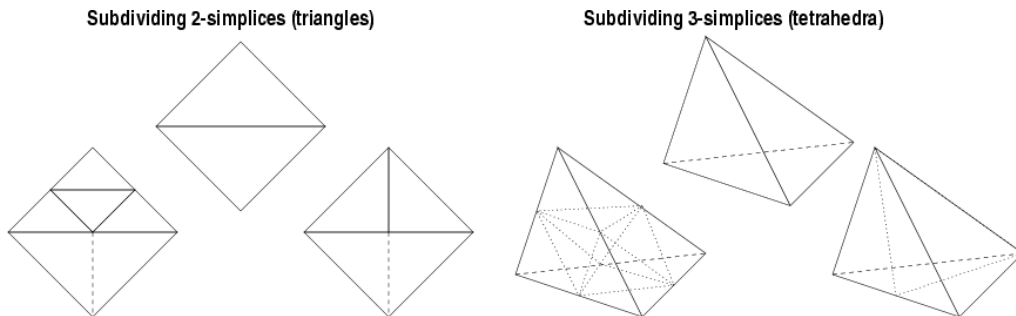


FIGURE 2. Refinement of 2- and 3-simplices using 4-section, 8-section, and bisection.

paired triangle in the 2-simplex case of Figure 2 illustrates the nature of conformity and its violation during refinement. A globally conforming simplex mesh is defined as a collection of simplices which meet only at vertices and faces; for example, removing the dotted bisection in the third figure from the left in Figure 2 produces a non-conforming mesh. Non-conforming simplex meshes create several theoretical as well as practical implementation difficulties, and the algorithms in MC (as well as those in PLTMG [9] and similar simplex-based adaptive codes [50, 70, 25, 26, 24]) enforce conformity using the above finite queue swapping strategy or a similar approach.

3.3. Inexact-Newton methods and linear multilevel methods. Addressing the complexity of Step 1 of the algorithm above, Newton methods are often the most effective:

Algorithm 3.2. (*Damped-inexact-Newton*)

- Given an initial u
- While ($|\langle F(u), v \rangle| > TOL$ for some v) do:
 - (1) Find δ such that $\langle DF(u)\delta, v \rangle = -\langle F(u), v \rangle + r, \forall v$
 - (2) Set $u = u + \lambda\delta$
- end while

The bilinear form $\langle DF(u)w, v \rangle$ in the algorithm above is the (Gateaux) linearization of the nonlinear form $\langle F(u), v \rangle$, defined earlier. This form is easily computed from most nonlinear forms $\langle F(u), v \rangle$ which arise from second order nonlinear elliptic problems, as we saw in the case of the PBE. The possibly nonzero “residual” term r is to allow for inexactness in the Jacobian solve for efficiency, which is quite effective in many cases; cf. [16, 41, 43]. The parameter λ brings robustness to the algorithm [14, 15, 43]. If folds or bifurcations are present, then the iteration is modified to incorporate path-following [13, 62].

As was the case for the nonlinear residual $\langle F(\cdot), \cdot \rangle$, the matrix representing the bilinear form in the Newton iteration is easily assembled, regardless of the complexity of the bilinear form $\langle DF(\cdot), \cdot \rangle$. In particular, the matrix equation for $w = \sum_{j=1}^n \beta_j \phi_j$ has the form:

$$(3.6) \quad AU = F,$$

where

$$(3.7) \quad A_{ij} = \langle DF(\sum_{k=1}^n \alpha_k \phi_k) \phi_j, \phi_i \rangle, \quad F_i = \langle F(\sum_{j=1}^n \alpha_j \phi_j), \phi_i \rangle, \quad U_i = \beta_i.$$

As long as the integral-based forms $\langle F(\cdot), \cdot \rangle$ and $\langle DF(\cdot), \cdot \rangle$ can be evaluated at individual points in the domain, then quadrature can be used to build the Newton equations, regardless of the complexity of the forms. This is one of the most powerful features of the finite element method, and is exploited to an extreme in the code MC (see §4 and [50]).

It can be shown that the Newton iteration above is dominated by the computational complexity of solving the n linear algebraic equations in each iteration; cf. [16, 47]. Multilevel methods are the only known provably optimal or nearly optimal methods for solving these types of linear algebraic equations, resulting from discretizations of a large class of general linear elliptic problems [10, 47, 88]. An obstacle to applying multilevel methods to the PBE is the presence of the geometrically complex discontinuities in the dielectric ϵ and in the Debye-Hückel parameter $\bar{\kappa}$; their presence destroys multilevel method efficiency, and can in fact cause divergence of multilevel methods. This problem effects both adaptive and uniform mesh approaches; a feasible solution is to employ algebraic multilevel methods [20, 21, 28, 32, 33, 29, 77, 83, 84], which effectively “smear” the discontinuities appropriately on coarser meshes so that multilevel methods can be made convergent again, and can even regain their near optimal complexity in some cases. Such an approach was taken in [51, 55] for meshes with non-uniform but Cartesian structure; we employ an unstructured version of this algebraic multilevel approach here. In particular, we enforce the following condition algebraically at each level of the multilevel hierarchy:

$$(3.8) \quad A_c = P^T AP,$$

where, A is the discretization of the linearized PBE on a given mesh, A_c is the discretization on a coarser mesh, and where P is the interpolation operator relating finite element basis functions on the two meshes. It can be shown that a multilevel method based on such a construction is provably convergent, and even optimal under certain conditions. See §4 below and also [50] for more details on the implementation of this triple sparse matrix product in MC.

4. THE COMPUTER PROGRAM MC

MC [50] (see also [53, 8]) is an adaptive multilevel finite element code developed by one of the authors (M.H.) over several years at Caltech and UC San Diego. It is designed to produce provably accurate numerical solutions to a general class of nonlinear elliptic equations on complicated 2- and 3-dimensional domains in an optimal or nearly-optimal way. MC employs *a posteriori* error estimation, adaptive simplex subdivision, unstructured algebraic multilevel methods, global inexact-Newton methods, and numerical continuation methods, for the highly accurate numerical solution of this class of problems.

4.1. The overall design of MC. The MC code is an implementation of Algorithm 3.1, employing Algorithm 3.2 for nonlinear elliptic systems that arise in Step 1 of Algorithm 3.1. The linear Newton equations in each iteration of Algorithm 3.2 are solved with algebraic multilevel methods, and the algorithm is supplemented with a continuation technique when necessary. Several of the features of MC are somewhat unusual, allowing for the treatment of very general nonlinear elliptic systems of tensor equations on domains with the structure of 2- and 3-manifolds. In particular, some of these features are:

- *Abstraction of the elliptic system:* The elliptic system is defined only through a nonlinear weak form over the domain manifold, along with an associated linearization form, also defined everywhere on the domain manifold (precisely the forms $\langle F(u), v \rangle$ and $\langle DF(u)w, v \rangle$ in the discussions above). To use the *a posteriori* error estimator, a third function must also be provided (essentially the strong form of the problem).
- *Abstraction of the domain manifold:* The domain manifold is specified by giving a polyhedral representation of the topology, along with an abstract set of coordinate labels of the user's interpretation, possibly consisting of multiple charts. MC works only with the topology of the domain, the connectivity of the polyhedral representation. The geometry of the domain manifold is provided only through the form definitions, which contain the manifold metric information, and through a `oneChart()` routine that the user provides to resolve chart boundaries.
- *Dimension independence:* Exactly the same code paths are taken for both two- and three-dimensional problems (as well as for higher-dimensional problems). To achieve this dimension independence, MC employs the simplex as its fundamental geometrical object for defining finite element bases.

As a consequence of the abstract weak form approach to defining the problem, the complete definition of the nonlinear Poisson-Boltzmann requires writing only 1000 lines of C to define the two weak forms, and to define the `oneChart()` routine. Changing to a different problem, e.g. large deformation nonlinear elasticity, involves providing only a different definition of the forms and a different domain description.

4.2. Topology and geometry representation in MC. A datastructure referred to as the *ringed-vertex* (cf. [50]) is used to represent meshes of d -simplices of arbitrary topology. This datastructure is illustrated in Figure 3. The ringed-vertex datastructure is similar to the winged-edge, quad-edge, and edge-facet datastructures commonly used in the computational geometry community for representing 2-manifolds [69], but it can be used more generally to represent arbitrary d -manifolds, $d = 2, 3, \dots$. It maintains a mesh of d -simplices with near minimal storage, yet for shape-regular (non-degenerate) meshes, it provides $O(1)$ -time access to all information necessary for refinement, un-refinement, and discretization of an elliptic operator. The ringed-vertex datastructure also allows for dimension independent implementations of mesh refinement and mesh manipulation, with one implementation covering arbitrary dimension d . An interesting feature of this datastructure is that the C structures used for vertices, simplices, and edges are all of fixed

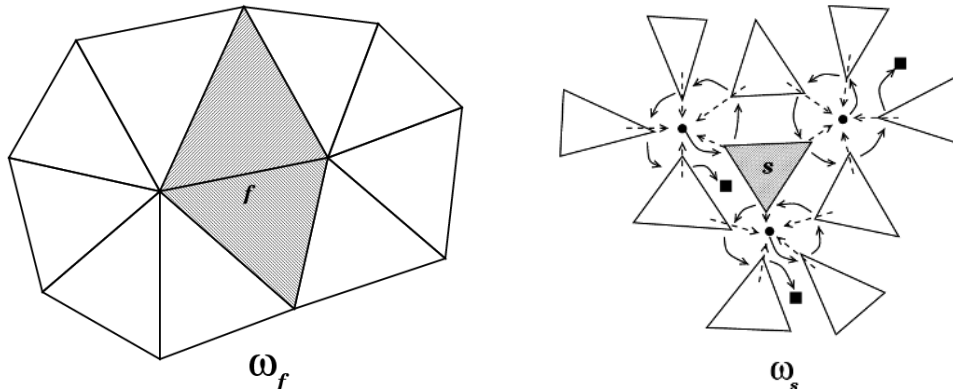


FIGURE 3. Polyhedral manifold representation. The figure on the left shows two overlapping polyhedral (vertex) charts consisting of the two rings of simplices around two vertices sharing an edge. The region consisting of the two darkened triangles around the face f is denoted ω_f , and represents the overlap of the two vertex charts. Polyhedral manifold topology is represented by MC using the *ringed vertex* datastructure. The datastructure is illustrated for a given simplex s in the figure on the right; the topology primitives are vertices and d -simplices. The collection of the simplices which meet the simplex s at its vertices (which then includes those simplices that share faces as well) is denoted as ω_s . (The set ω_s includes s itself.) Edges are temporarily created during subdivision but are then destroyed (a similar ring datastructure is used to represent the edge topology).

size, so that a fast array-based implementation is possible, as opposed to a less-efficient list-based approach commonly taken for finite element implementations on unstructured meshes. A detailed description of the ringed-vertex datastructure, along with a complexity analysis of various traversal algorithms, can be found in [50].

Since MC is based entirely on the d -simplex, for adaptive refinement it employs simplex bisection, using one of the simplex bisection strategies outlined earlier. Bisection is first used to refine an initial subset of the simplices in the mesh (selected according to some error estimates, discussed below), and then a closure algorithm is performed in which bisection is used recursively on any non-conforming simplices, until a conforming mesh is obtained. If it is necessary to improve element shape, MC attempts to optimize the following simplex shape measure function for a given d -simplex s , in an iterative fashion, similar to the approach taken in [18]:

$$(4.1) \quad \eta(s, d) = \frac{2^{2(1-\frac{1}{d})} 3^{\frac{d-1}{2}} |s|^{\frac{2}{d}}}{\sum_{0 \leq i < j \leq d} |e_{ij}|^2}.$$

The quantity $|s|$ represents the (possibly negative) volume of the d -simplex, and $|e_{ij}|$ represents the length of the edge that connects vertex i to vertex j in the simplex. For $d = 2$ this is the shape-measure used in [18] with a slightly different normalization. For $d = 3$ this is the shape-measure developed in [63] again with a slightly different normalization.

4.3. Discretization and adaptivity in MC. Given a nonlinear weak form $\langle F(u), v \rangle$, its linearization bilinear form $\langle DF(u)w, v \rangle$, a Dirichlet function \bar{u} , and collection of simplices representing the domain, MC uses a default linear element to produce and then solve the implicitly defined nonlinear algebraic equations for the basis function coefficients in the expansion (3.4). The user can also provide his own element, specifying the number of degrees of freedom to be located on vertices, edges, faces, and in the interior of simplices, along with a quadrature rule, and the values of the

basis functions at the quadrature points on the master element. Different element types may be used for different components of a coupled elliptic system.

Once the equations are assembled and solved (discussed below), *a posteriori* error estimates are computed from the discrete solution to drive adaptive mesh refinement. The idea of adaptive error control in finite element methods is to estimate the behavior of the actual solution to the problem using only a previously computed numerical solution, and then use the estimate to build an improved numerical solution by increasing the polynomial order (*p*-refinement) or refining the mesh (*h*-refinement) where appropriate. Note that this approach to adapting the mesh (or polynomial order) to the local solution behavior affects not only approximation quality, but also solution complexity: if a target solution accuracy can be obtained with fewer mesh points by their judicious placement in the domain, the cost of solving the discrete equations is reduced (sometimes dramatically) because the number of unknowns is reduced (again, sometimes dramatically). Generally speaking, if an elliptic equation has a solution with local singular behavior, such as would result from the presence of abrupt changes in the coefficients of the equation (e.g., the functions ϵ and $\bar{\kappa}$ in the present case), then adaptive methods tend to give dramatic improvements over non-adaptive methods in terms of accuracy achieved for a given complexity price. Two examples illustrating bisection-based adaptivity patterns (driven by a completely geometrical “error” indicator simply for illustration) are shown in Figure 4.

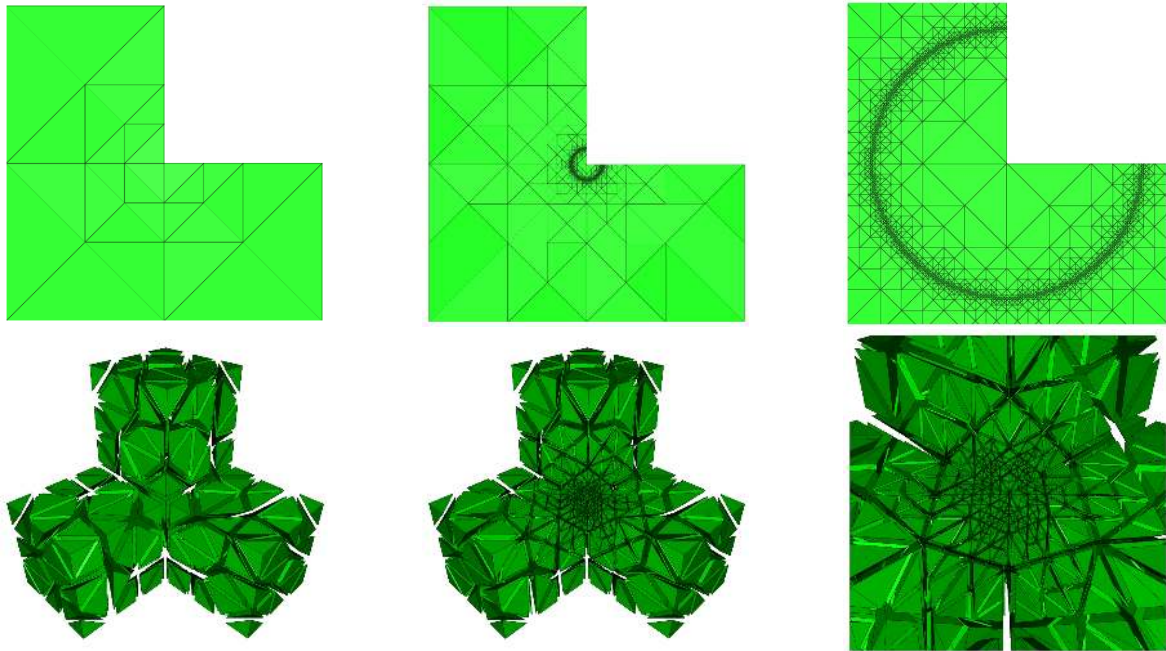


FIGURE 4. Examples illustrating the 2D and 3D adaptive mesh refinement algorithms in MC. The right-most figure in each row shows a close-up of the area where most of the refinement occurred in each example.

4.4. *A posteriori* error estimation in MC. There are several approaches to adaptive error control, although the approaches based on *a posteriori* error estimation are usually the most effective and most general. While most existing work on *a posteriori* estimates has been for linear problems, extensions to the nonlinear case can be made through linearization. For example, consider the nonlinear problem:

$$(4.2) \quad F(u) = 0, \quad F \in C^1(\mathcal{B}_1, \mathcal{B}_2^*),$$

where \mathcal{B}_1 and \mathcal{B}_2 are typically function spaces such as our now familiar $H_0^1(\Omega)$. The notation \mathcal{B}^* above represents the *dual space* of bounded linear functionals on the space \mathcal{B} , and features prominently in partial differential operator equations and in error estimation analysis. Consider now also a discretization of the equation above:

$$(4.3) \quad F_h(u_h) = 0, \quad F_h \in C^0(U_h, V_h^*),$$

where $U_h \subset \mathcal{B}_1$ and $V_h \subset \mathcal{B}_2$. For example, this could be the nonlinear Poisson-Boltzmann equation and a finite element discretization. In this case, the function spaces would be taken to be $\mathcal{B}_1 = \mathcal{B}_2 = H_0^1(\Omega)$. The nonlinear residual $F(u_h)$ can be used to estimate the error in the approximation $\|u - u_h\|_{\mathcal{B}_1}$, through the use of a *linearization theorem* [65, 86]. In particular, under fairly mild assumptions, one can establish the following set of inequalities:

$$(4.4) \quad C_1 \|F(u_h)\|_{\mathcal{B}_2^*} \leq \|u - u_h\|_{\mathcal{B}_1} \leq C_2 \|F(u_h)\|_{\mathcal{B}_2^*}.$$

The error measure $\|v\|_{\mathcal{B}_1}$ would typically be the H^1 -norm defined in (2.4). The measure $\|f\|_{\mathcal{B}_2^*}$ is the *dual space norm* of a functional in the dual space of \mathcal{B}_2 ; it is defined clearly in Appendix A. The effect of linearization is absorbed by the positive constants C_1 and C_2 , and one focuses on two-sided estimates for the nonlinear residual $\|F(u_h)\|_{\mathcal{B}_2^*}$ appearing on each side of (4.4). However, since we typically construct highly refined meshes where needed, such local linearized estimates are reasonable. Note that $\|F(u_h)\|_{\mathcal{B}_2^*}$ can be estimated in several ways, including

- (1) Approximation by $\|F_h(u_h)\|_{\mathcal{B}_2^*}$ (strong and weak residual estimates) [86, 87, 65, 66].
- (2) Solution of local problems (Dirichlet or Neumann) [17, 19].

The two approaches are often equivalent up to constants (cf. [86]). For reasons of efficiency in the case of elliptic systems in spatial dimension three, we employ the first approach in MC, referred to as estimation by strong residuals. The error estimator can also be user-specified in MC; the user is given all of the information about an element and the current discrete solution in the element, and is asked to produce an error estimate for the element. This can be as simple as computing the average of the jumps in the normal derivatives of the solution gradient across the edges shared with neighboring elements, or the complete calculation of a bound on the strong form of the residual [86]. In particular, one employs the linearization theorem above, together with some derived (and computable) upper and lower bounds on the nonlinear residual $\|F(u_h)\|_{\mathcal{B}_2^*}$ given by the following pair of inequalities:

$$(4.5) \quad A \leq \|F(u_h)\|_{\mathcal{B}_2^*} \leq B.$$

While it is clear that the upper bound B is the key to bounding the error, the lower bound A can also be quite useful; it can help to ensure that the adaptive procedure doesn't do too much work by over-refining an area where it is unnecessary. The effectiveness of an adaptive finite element code can hinge on the implementation details of the estimator, and implementing it efficiently can be quite an art form (cf. [86, 17, 19]).

We employ a residual error estimator in MC, based on computing an upper bound on the nonlinear residual as in (4.5). Appendix A contains the detailed derivation of the upper bound in equation (4.5) that is used in MC for general nonlinear elliptic equations. In Appendix B, we instantiate the estimator in the case of the nonlinear Poisson-Boltzmann equation, and we give the error estimator which we employ for the experiments here and in the a follow-on paper [6].

4.5. Solution of linear and nonlinear systems with MC. When a system of nonlinear finite element equations must be solved in MC, the global inexact-Newton Algorithm 3.2 is employed, where the linearization systems are solved by linear multilevel methods. This is the algorithm originally presented in [55], but now used in conjunction with error estimation and adaptivity. When necessary, the Newton procedure in Algorithm 3.2 is supplemented with a user-defined normalization equation for performing an augmented system continuation algorithm. The linear systems

arising as the Newton equations in each iteration of Algorithm 3.2 are solved using a completely algebraic multilevel algorithm. Either refinement-generated prolongation matrices P_k , or user-defined prolongation matrices P_k in a standard YSMP-row-wise sparse matrix format, are used to define the multilevel hierarchy algebraically. In particular, once the single “fine” mesh is used to produce the discrete nonlinear problem $F(u) = 0$ along with its linearization $Au = f$ for use in the Newton iteration in Algorithm 3.2, a J -level hierarchy of linear problems is produced algebraically using the following recursion:

$$(4.6) \quad A_{k+1} = P_k^T A_k P_k, \quad k = 1, \dots, J-1, \quad A_1 \equiv A.$$

As a result, the underlying multilevel algorithm is provably convergent in the case of self-adjoint-positive matrices [57]. Moreover, the multilevel algorithm has provably optimal $O(N)$ convergence properties under the standard assumptions for uniform refinements [88], and is nearly-optimal $O(N \log N)$ under very weak assumptions on adaptively refined problems [11].

Coupled with the superlinear convergence properties of the outer inexact Newton iteration in Algorithm 3.2, this leads to an overall complexity of $O(N)$ or $O(N \log N)$ for the solution of the discrete nonlinear problems in Step 1 of Algorithm 3.1. Combining this low-complexity solver with the judicious placement of unknowns only where needed due to the error estimation in Step 2 and the subdivision algorithm in Steps 3-7 of Algorithm 3.1, leads to a very effective low-complexity approximation technique for solving a general class of nonlinear elliptic equations such as the Poisson-Boltzmann equation.

4.6. Parallel computing with MC through local estimate decoupling. MC incorporates a new approach to the use of parallel computers with adaptive finite element methods, based on the idea of decoupling the problem through *local error estimation* [8, 12]. The idea of the algorithm, described in detail in [8], is as follows. MC solves a small problem sequentially on one processor, employing an initially very coarse mesh. The mesh is then partitioned (e.g., inertially or spectrally), where the partitioning strategy is *weighted* by *a posteriori* error estimates. The entire coarse mesh is then distributed to all of the processors, along with the list of elements that the particular processor “owns”, in which it is to compute *a posteriori* error estimates as it performs its solve-estimate-refine loop (Step 2 through Step 7 in Algorithm 3.1). When a processor has reached an error tolerance locally, computation stops on that processor. An example showing the types of local refinements that occur within each subdomain is depicted in Figure 5.

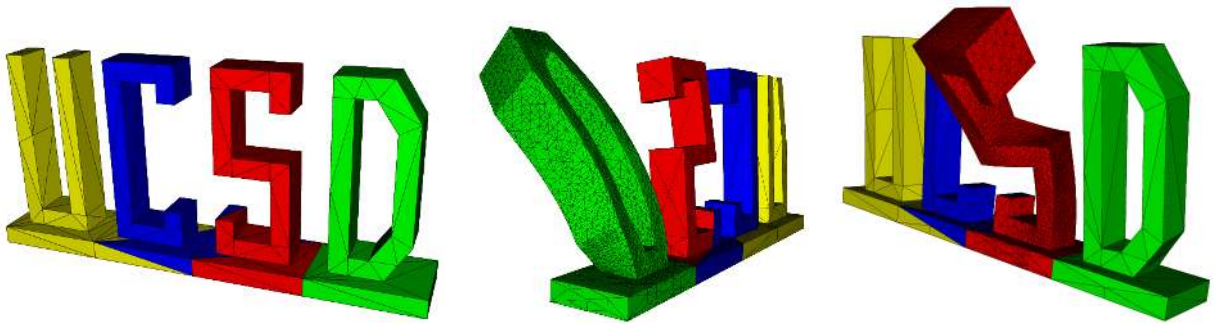


FIGURE 5. An example showing the types of local refinements that are created by the local estimate-based parallel algorithm. The distinct colors correspond to the elements assigned to a particular processor. The initial spectrally-bisected coarse mesh is shown in the left-most figure, followed by the locally adapted subdomains on two of four processors at the end of the computation. No communication took place during the calculations.

Mesh conformity algorithms will necessarily lead the refinement regions into the surrounding areas, which will in most cases lead to nonconforming interfaces between the regions. This is not a concern in that the subdomain solutions themselves are taken to be the final discrete solution represented subdomain-wise. To evaluate the solution at any point in the domain, one simply must determine which subdomain contains the point, and then fetch the solution value from the particular subdomain. While this approach seems naive, some recent theoretical results [91] support this as provably good, and even optimal in some cases. The principle idea underlying the results in [91] is that while elliptic problems are globally coupled, this global coupling is essentially a “low-frequency” coupling, and can be handled on the initial mesh which is much coarser than that required for approximation accuracy considerations. This idea has been exploited, for example, in [89, 90], and is in fact why the construction of a coarse problem in overlapping domain decomposition methods is the key to obtaining convergence rates which are independent of the number of subdomains (cf. [88]). A complete description of the algorithm and an explanation of why it actually works can be found in [8], along with examples using MC and the 2D adaptive code PLTMG [9],

4.7. Availability of MC. A fully-functional MATLAB version of MC for 2-manifolds, called *MClab*, is available under the GNU copyleft license at the following website:

<http://www.scicomp.ucsd.edu/~mholst/>

MClab employs the ringed-vertex datastructure and implements the same adaptivity and solution algorithms that are used in MC. A number of additional tools developed for use with MC and MClab are also available under a GNU license at the site above, include MCsg (an OpenGL-based X11/Win32 polygon display tool) and MCPs (an OpenGL-to-Postscript generator used to generate the pictures of finite element meshes appearing in this paper; it is described in [7]).

5. NUMERICAL RESULTS FOR TWO MODEL PROBLEMS

The effectiveness of the adaptive multilevel finite element method will now be demonstrated by using two well-known model problems (illustrated by the two cartoons in Figure 6) with known analytical solutions. The application to a number of more complex systems appears in [6]. To

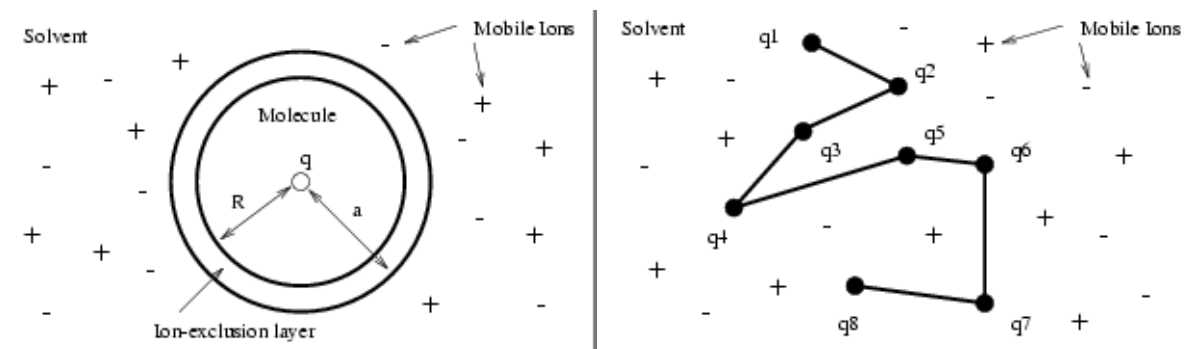


FIGURE 6. Two-dimensional cartoons illustrating the form of the two model problems.

make the following numerical results as interesting as possible, we will tabulate the accuracy, discretization cell (tetrahedron) number and cell size (tetrahedral diameter), memory usage, and solution time of the adaptive method implemented in MC when applied to the two model problems. We will then compare to the same measures (where the discretization cells are now cubical *boxes*) for the highly optimized (non-uniform) Cartesian mesh algebraic multigrid Poisson-Boltzmann solver *PMG*. The *PMG* solver has been described in detail in two papers appearing previously in this journal [54, 55]. Although the *PMG* solver is non-adaptive and can only employ non-uniform

Cartesian meshes, it shares a number of crucial core algorithms with MC. These include the global inexact Newton algorithm described in [52, 55], and the linear complexity algebraic multilevel solver described in [52, 54]. We believe that PMG is one of the fastest and most robust of the Cartesian mesh solvers for the nonlinear Poisson-Boltzmann equation that are publically available in source form. The version of PMG we employed for this paper has been freely available under the GNU copyleft license for more than four years, and can be downloaded from the website given in Section 4.7. Therefore, the following results should be viewed as a fair but conservative demonstration of what might be gained by the use of an adaptive method.

It should be noted that there are a number of uniform mesh solvers for these types of problems that are publically available. In addition, several codes written specifically for the Poisson-Boltzmann equation, while not publically available, may be obtained for a small fee. Chief among these is the well-known highly-optimized DelPhi software from the Honig lab at Columbia University. For small problem sizes, such as cubical meshes with 63 mesh lines in each coordinate direction or less, DelPhi and similar approaches are probably unmatched in producing a solution in the least amount of execution time. However, our interest here is developing techniques for producing highly accurate solutions for very large and complex systems, and uniform mesh approaches require much larger meshes than these codes allow. For example, in DelPhi one may only employ cubical meshes with 63 mesh lines in each direction. Moreover, the algorithm employed by DelPhi is based on SOR with an intelligent (nearly optimal) parameter estimation scheme (described in detail in [71]). It is well-known (cf. [94]) that SOR with optimal parameter has complexity $O(N^{1.33}\ln N)$, and decays to $O(N^{1.67}\ln N)$ if the parameter is not optimal, where N is the number of unknowns in the discrete system. The multilevel methods employed here and previously in [54, 55] have both theoretical and empirically demonstrated complexity of $O(N)$, and a simple back-of-the-envelope calculation will convince the reader that for large problem sizes one has no choice but to employ multilevel methods. Since PMG is the only publically available PB solver that demononstrates this linear scaling with the problem size, we will use PMG in our numerical experiments.

To use MC to adaptively solve the Poisson-Boltzmann equation requires several preprocessing steps. To begin, our coarse simplex mesh generation procedure (described in more detail in [6]) involves using the charge locations from the PDB file as the interior vertices of an initial coarse tetrahedral mesh. After scattering some additional shells of vertices around the charges, with sparsity of the shell vertices increasing with the shell diameter, we employ a standard 3D Delaunay mesh generator to produce an initial mesh of tetrahedra. The tetrahedral meshes produced by this procedure for the two model problems considered here appear in Figures 7 and 8. Since the

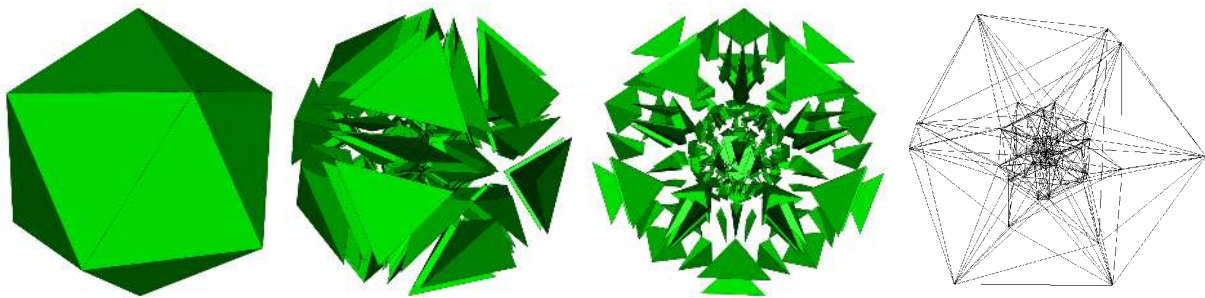


FIGURE 7. The initial coarse tetrahedral mesh for the single atom example. The mesh contains 59 vertices and 318 tetrahedra.

solution to the PBE changes rapidly near the “molecular surface” as defined by the discontinuities in the dielectric ϵ and the Debye-Hückel parameter $\bar{\kappa}$, and near the delta functions in the source term representing the charges in the protein, one expects the residual-based error estimator (derived

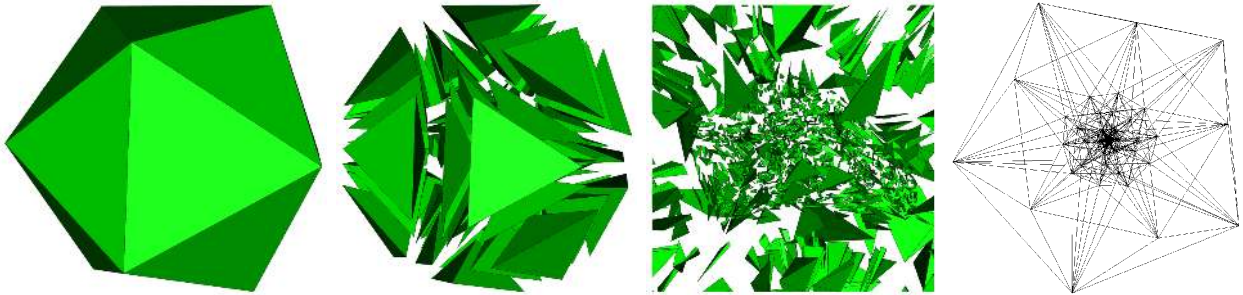


FIGURE 8. The initial coarse tetrahedral mesh for the rod-like molecule example. The mesh contains 258 vertices and 1,621 tetrahedra.

in Appendices A and B) to automatically refine the mesh near the dielectric boundaries and the charge locations. In the first test problem, we can define the molecular surface quite easily, since the problem contains only a single atom. The refinement in this case will occur at both the molecular surface and the charge locations. In the second test problem, there is no such surface; the biomolecule is taken to be a fictitious rod-like molecule with 200 atoms of varying charges, with complete solvent penetration. The refinement in this example should occur primarily near the charge locations.

To apply the finite element approximation techniques presented here to more complex problems would require an expensive calculation to determine whether or not a point in the domain is solvent accessible, given the description of complex molecule from a PDB file. Since we have no underlying Cartesian mesh to exploit, this calculation typically involves an expensive traversal of the tetrahedral mesh. In any event, the calculation determines (implicitly) the piecewise-constant dielectric ϵ and the Debye-Hückel parameter $\bar{\kappa}$. We discuss two efficient approaches for handling the solvent accessibility problem in the second paper [6]. The first approach involves the use of the Solvent Accessible Volume library of Bashford and You [93]. We present an alternative approach in [6] based on atomic hash lists, which was developed by one of the authors (N.B.). It appears to be somewhat faster than the approach in [93], and is fast enough to make solvent accessibility evaluation negligible compared to the other computational costs in MC. This allows MC to be used effectively for arbitrarily complex charged biological structures with arbitrarily complicated solvent accessibility surfaces. These issues are discussed at length in [6].

5.1. A single atom. The first test problem consists of a single atom of radius 2 \AA , centered at the origin. This allows for an analytical solution to the linearized Poisson-Boltzmann equation for the evaluation of accuracy. Due to spherical symmetry, there is only radial dependence in the solution to the linearized PBE, which is easily verified to have the following analytical form in the three subregions of \mathbb{R}^3 (cf. [51, 52]):

$$(5.1) \quad u(r) = \frac{q}{\epsilon_w R} \left(1 - \frac{\kappa R}{1 + \kappa a} \right), \quad r \leq R,$$

$$(5.2) \quad u(r) = \frac{q}{\epsilon_w r} \left(1 - \frac{\kappa r}{1 + \kappa a} \right), \quad R < r < a,$$

$$(5.3) \quad u(r) = \frac{q e^{\kappa a}}{\epsilon_w (1 + \kappa a)} \cdot \frac{e^{-\kappa r}}{r}, \quad r \geq R.$$

Here, $a = R$ is the 2 \AA radius of the atom in the left frame in Figure 6, q is the charge of the atom, ϵ_w is the dielectric over all of \mathbb{R}^3 , $\kappa = \bar{\kappa}/\sqrt{\epsilon_w}$, and where the other terms are as described following (1.1). For both the uniform and adaptive calculations, we use (5.3) as an exact boundary

condition on the exterior surface of the solvent-filled box containing the atom (in the adaptive case, we employ a solvent-filled sphere). We use a 12 Å box in the uniform case; in the adaptive case, we use a 100 Å sphere to illustrate the advantage of the unstructured mesh approach. Note that since the boundary condition we employ is exact, the smaller domain size gives the uniform mesh method an advantage. It is more typical that the exact boundary condition is not known, in which case the much larger domain gives the adaptive method a more accurate boundary condition approximation.

For both the non-adaptive code PMG and the adaptive code MC, Table 1 lists the number of refinements, the smallest cell size (box length or tetrahedral diameter), the accuracy in the solution (relative error) at a selected point in the solvent near the atom (at $(x, y, z) = (0, 0, 2.1)$), the number of vertices and cells (boxes or tetrahedra), the memory usage, and the total CPU time required to solve the problem. By extrapolating from the discretization error reduction in the table for the uniform mesh method as the mesh is uniformly refined, it is clear from the table that at least an additional order of magnitude in memory and solution time are required by the uniform method to reach the same accuracy and resolution as the adaptive method. Moreover, using the adaptive method we can obtain cell resolutions smaller than would be obtainable with uniform methods using the largest supercomputers available (again extrapolating the memory and CPU time to achieve the cell diameter potential of the adaptive method).

TABLE 1. Performance comparison: The single atom example is first solved with the non-adaptive Cartesian code PMG, and is then solved adaptively using MC. The “Diameter” column refers to the smallest cell size in the mesh. The “Error” column refers to the point-wise error in the solution at a selected point near the atom. All calculations are in double precision, and execution times are for a 350Mhz Pentium II running Linux.

Method	Levels	Diameter	Error	#Vertices	#Cells	Memory	Time
PMG	5	3.8e-1 Å	2.2e-1	35,937	32,768	9 MB	5.6 sec
PMG	6	1.9e-1 Å	5.8e-2	274,625	262,144	66 MB	47.3 sec
PMG	7	9.4e-2 Å	1.6e-2	2,146,689	2,097,152	515 MB	393.0 sec
MC	10	7.0e-3 Å	5.2e-3	67,288	373,978	109 MB	122.2 sec

Figure 9 shows the uniform mesh, a cutting plane, and a projected isosurface for the solution produced by the Cartesian mesh code PMG. Shown also are collection of isosurfaces of varying magnitudes, illustrating the lack of resolution obtainable with the uniform method. In Figure 10 we show the corresponding calculation using the adaptive code MC. First shown is the adapted mesh, a cutting plane through the unstructured mesh, followed by a sequence of highly-resolved isosurfaces of the solution.

5.2. A rod-like molecule with complete solvent penetration. The second test problem consists of a collection of 200 atoms at arbitrary locations, of which there are nine atom types with differing charges. The charges all lie within a box of length 20 Å on each side. The solvent is assumed to completely penetrate the (fictitious) molecule, which allows for an analytical solution to the linearized Poisson-Boltzmann equation for the evaluation of accuracy. In this case, the linearized PBE is known to have the following analytical solution in all of \mathbb{R}^3 (cf. [51, 52]):

$$(5.4) \quad u(x) = \left(\frac{e_c^2}{k_B T} \right) \sum_{i=1}^{N_m} \frac{e^{-\kappa|x-x_i|}}{\epsilon_w |x-x_i|} z_i,$$

where ϵ_w is the dielectric over all of \mathbb{R}^3 , $\kappa = \bar{\kappa}/\sqrt{\epsilon_w}$, and where the other terms are as described following (1.1). For both the uniform and adaptive calculations, we use (5.4) as an exact boundary

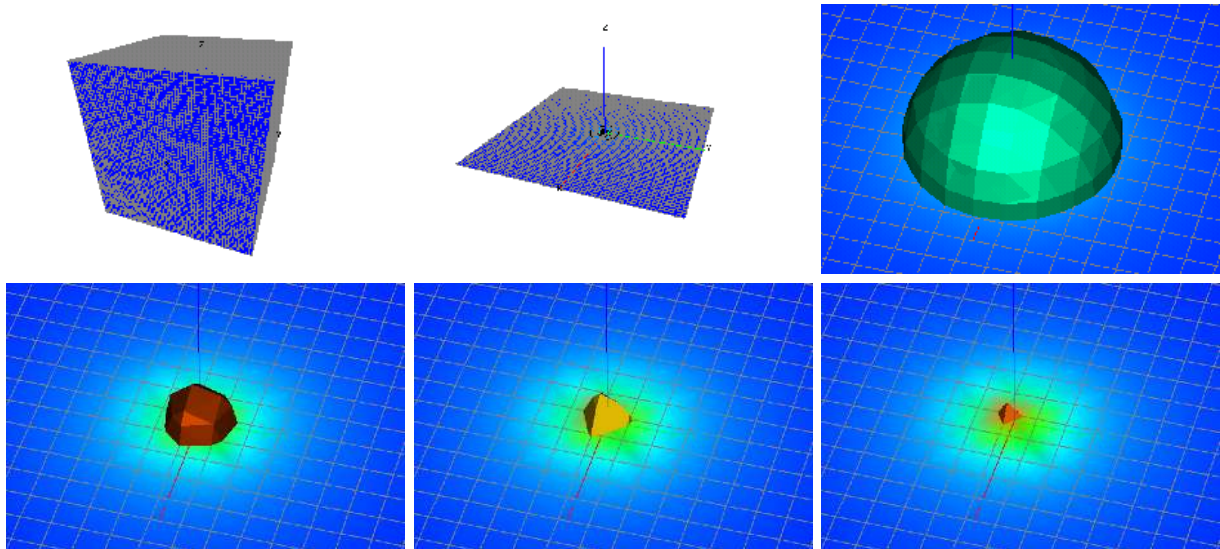


FIGURE 9. The single atom example solved non-adaptively by the Cartesian mesh code PMG. The uniform $65 \times 65 \times 65$ mesh used is shown, which contains 262,144 box elements and 274,625 box vertices. Also shown are several isosurfaces of the solution at the 274,625 box vertices. The $x - y$ plane is also shown cutting through the solution, with isosurfaces projected onto the plane. The uniform mesh is also projected onto the cutting plane, illustrating the large uniform cell volumes. Although the mesh contains an enormous number of cells and vertices, note that the isosurfaces near the singularity are very poorly resolved.

condition on the exterior surface of the solvent-filled box containing the atom (again, in the adaptive case, we employ a solvent-filled sphere). We use a box with 60 \AA sides in the uniform case; in the adaptive case, we use a 500 \AA diameter sphere to again illustrate the flexibility of the unstructured mesh approach. (Again, since the boundary condition we employ is exact, the smaller domain size gives the uniform mesh method an advantage.)

For both the non-adaptive code PMG and the adaptive code MC, Table 2 lists the number of refinements, the smallest cell size (box length or tetrahedral diameter), the accuracy in the solution (relative error) at a selected point in the solvent (at $(x, y, z) = (0, 0, 0)$), the number of vertices and cells (boxes or tetrahedra), the memory usage, and the total CPU time required to solve the problem. It is again clear from the table that at least an additional order of magnitude in memory and solution time are required by the uniform method in order to reach the same accuracy and resolution as the adaptive method.

Figure 11 shows the uniform mesh, a cutting plane, a projected isosurface for the solution produced by the uniform mesh calculation, and then a collection of isosurfaces of varying magnitudes, illustrating the lack of resolution obtainable with the uniform method. In Figure 12, we show the corresponding calculation with the adaptive methods. First shown is the adapted mesh, a cutting plane through the unstructured mesh, followed by a sequence of highly-resolved isosurfaces of the solution.

6. SOME REMARKS ON THE THEORETICAL JUSTIFICATION OF “FOCUSSING”

A common technique used with numerical solution of the Poisson-Boltzmann equation has been “focussing”. This is a technique where one solves the equation on a uniform mesh (typically fixed for all time at $65 \times 65 \times 65$) using a large mesh spacing containing the molecule well within the

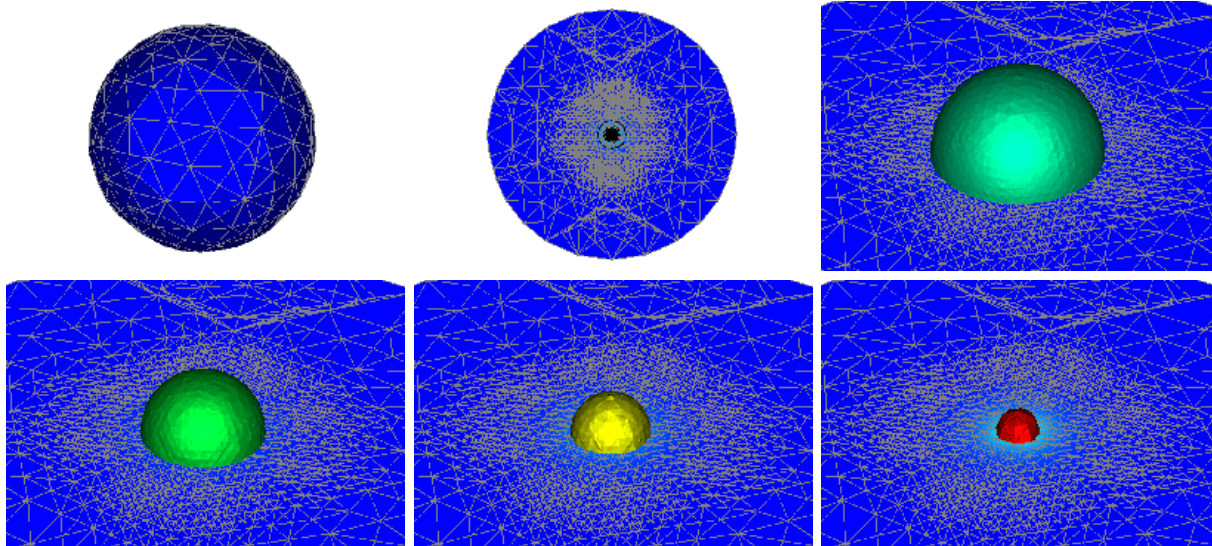


FIGURE 10. The single atom example solved adaptively using MC, where the adapted mesh containing 67,288 vertices and 373,978 tetrahedra is automatically determined adaptively, without human intervention. Shown is the final adapted mesh and several isosurfaces of the adapted solution computed by MC at the 67,288 tetrahedral vertices. The $x - y$ plane is also shown cutting through the solution, with isosurfaces projected onto the plane. The adapted mesh is also projected onto the cutting plane, illustrating the small non-uniform cell volumes near the dielectric boundary and near the singularity.

TABLE 2. Performance comparison: the rod-like molecule example is first solved with the non-adaptive Cartesian code PMG, and is then solved adaptively using MC. The “Diameter” column refers to the smallest cell size in the mesh. The “Error” column refers to the point-wise error in the solution at a selected point near the atom. All calculations are in double precision, and execution times are for a 350Mhz Pentium II running Linux.

Method	Levels	Diameter	Error	#Vertices	#Cells	Memory	Time
PMG	5	6.25e0 Å	1.3e0	35,937	32,768	9 MB	7.2 sec
PMG	6	3.13e0 Å	3.6e-1	274,625	262,144	66 MB	68.7 sec
PMG	7	1.56e0 Å	1.1e-1	2,146,689	2,097,152	515 MB	453.3 sec
MC	11	1.7e-3 Å	4.1e-2	78,113	434,804	131 MB	158.4 sec

interior, so that well-known analytical solutions can be used as accurate approximations to the boundary conditions. One then solves the problem a second time, but using a mesh spacing that results in the uniform $65 \times 65 \times 65$ mesh barely covering the biomolecule, with boundary conditions provided by the earlier coarse mesh solution. This technique has been used a number of times in the literature (cf. [71, 80]).

While this technique seems intuitively reasonable in some sense, there has so far been no theoretical justification of the approach. However, some recent results in finite element error estimation appearing in [91] can be used to justify this two-mesh technique. The principle idea underlying the results in [91] is that while elliptic problems are globally coupled (in particular, to their boundary conditions), this global coupling is essentially a “low-frequency” coupling, and can be handled on

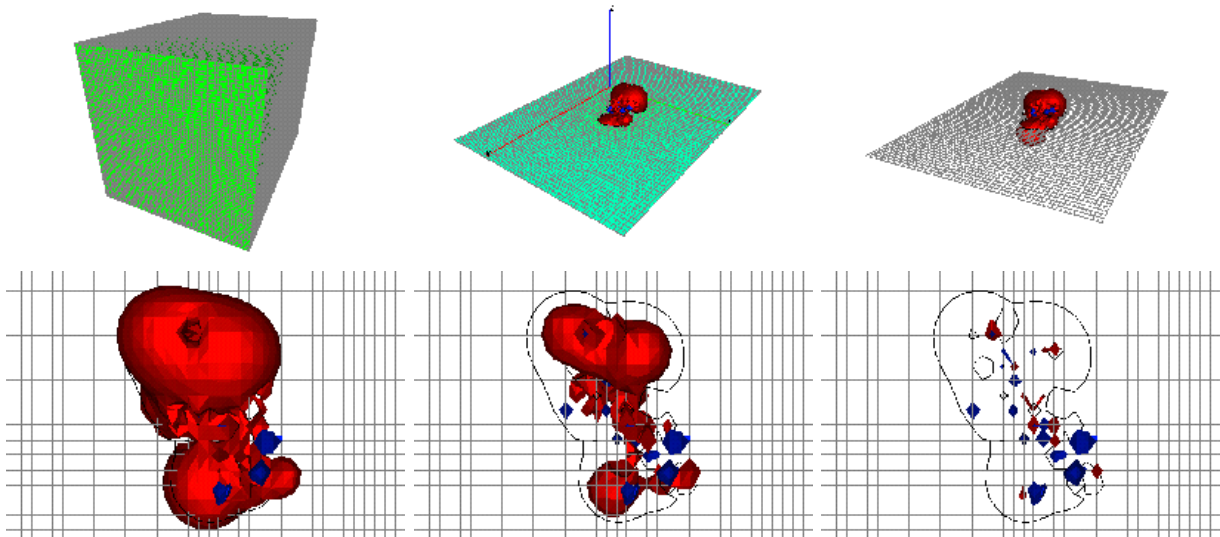


FIGURE 11. The rod-like molecule example solved non-adaptively by the Cartesian mesh code PMG. The uniform $65 \times 65 \times 65$ mesh used is shown, which contains 262,144 box elements and 274,625 box vertices. Also shown are several isosurfaces of the uniform mesh rod-like molecule solution computed by PMG at the 274,625 box vertices. The $x-y$ plane is also shown cutting through the solution, with isosurfaces projected onto the plane. The uniform mesh is also projected onto the cutting plane, illustrating the large uniform cell volumes. Again, while the mesh contains a very large number of cells and vertices, the isosurfaces near the singularities are poorly resolved, and reflect a very inaccurate, “smeared” potential.

a mesh which is much coarser than that required for approximation accuracy considerations. This idea has been exploited for example in [89, 90], and is in fact why the construction of a coarse problem in overlapping domain decomposition methods is the key to obtaining convergence rates which are independent of the number of subdomains (cf. [88]).

The key result in [91] for our purposes is the following *a priori* error estimate. To explain, let Ω_k be the collection of disjoint subregions of the domain Ω , and let Ω_k^0 to be an extension of the disjoint Ω_k , such that $\Omega_k \subset \subset \Omega_k^0$, and so that the sizes of the overlap regions $\Omega_i^0 \cap \Omega_j^0$ are on the order of the sizes of the regions Ω_k . We can think about one of the Ω_k regions as being a mesh which barely covers the biomolecule, and Ω_k^0 as a slightly larger mesh that includes a small amount of additional solvent region around the biomolecule. Now, under some reasonable assumptions about the approximation properties of a finite element space S_0^h defined over Ω , the following *a priori* error estimate holds for the global Galerkin solution u_h to Poisson-like equations similar to the Poisson-Boltzmann equation:

$$\|u - u_h\|_{H^1(\Omega_k)} \leq C \left(\inf_{v \in S_0^h} \|u - v\|_{H^1(\Omega_k^0)} + \|u - u_h\|_{L^2(\Omega)} \right).$$

This *a priori* result states that the error in the global Galerkin solution u_h restricted to a subdomain Ω_k can be bounded by the error in the best approximation from the finite element space S_0^h measured only over the extended subdomain Ω_k^0 , plus a higher-order global term (the global L^2 -norm of the error). In the context of the focussing technique, if the large coarse mesh used during the first step of focussing is quasi-uniform and shape-regular with element diameter H , and if we make some reasonable smoothness assumptions about the solution u , then standard finite element

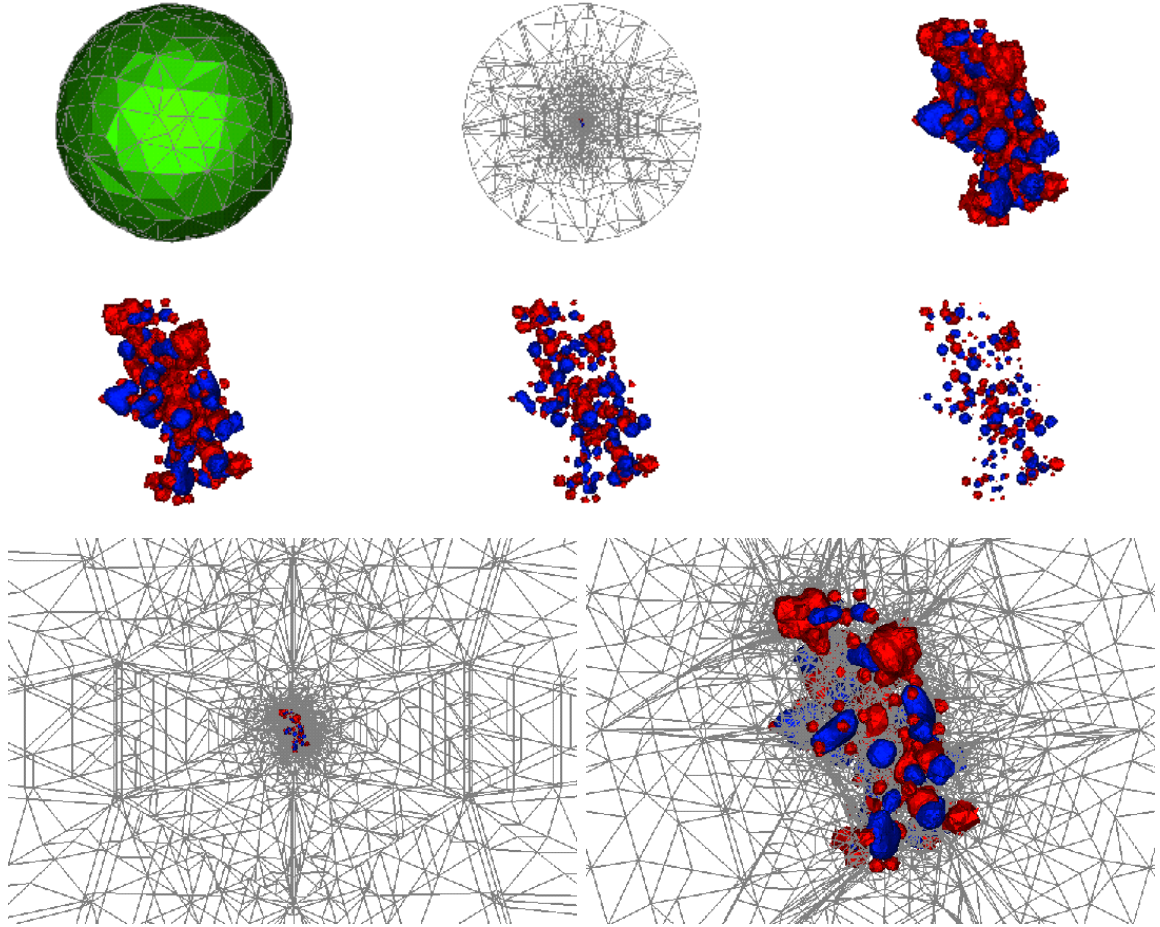


FIGURE 12. The rod-like molecule example solved adaptively using MC, where the adapted mesh containing 78,113 vertices and 434,804 tetrahedra is automatically determined adaptively (without human intervention). Shown is the final adapted mesh and several isosurfaces of the solution computed by MC at the 78,113 tetrahedral vertices. One advantage of the unstructured mesh is clear in the second frame: the boundaries can be placed quite far from the biomolecule, which provides for accurate boundary condition approximation. The remarkable resolution gain over the non-adaptive method becomes clear when one compares the isosurfaces above with those in Figure 11. The $x-y$ plane is also shown cutting through the solution in the last two frames, illustrating the extremely small non-uniform cell volumes near the singularities in the source, which drives the adaptivity through the residual-based error estimator.

interpolation theory can be used to bound the global term by an $O(H^2)$ factor. Moreover, if the mesh used in the second step of focussing covers only the extended subdomain Ω_k^0 but is again quasi-uniform and shape-regular, and now has smaller element diameter h , then the local term can be bounded using standard interpolation theory by an $O(h)$ factor. This implies that if the two-step focussing technique respects the relationship $h = O(H^2)$ between the coarse and fine regions, then asymptotically the global term based on the much coarser mesh outside Ω_k^0 does not pollute the accuracy of the adapted solution in the subdomain Ω_k .

The above discussion justifies the focussing approach for finite element discretizations of the Poisson-Boltzmann equation. An algorithm similar in spirit to focussing, but constructed for purely parallel computing considerations, is employed in MC; the algorithm is discussed in detail in [8], and was described briefly in Section 4.6.

7. SUMMARY

In this paper we derived a weak form for the nonlinear PBE, and calculated the bilinear form representing a linearization for use in Newton-like iterations. We outlined our finite element approximation approach, employing simplex-based piecewise-linear functions on simplex triangulations of finite regions of solvent containing the biological molecule. Global inexact-Newton multilevel algorithms were then described for solution of the resulting discrete equations. The error estimation and adaptive simplex subdivision procedures we employ were then discussed, and a detailed derivation of the general error estimator was given in Appendix A, followed by the estimator specifically for the Poisson-Boltzmann equation in Appendix B. We described the implementation of these techniques in the MC software package, and outlined some of its more interesting features. The geometry datastructures were described, and the implementation of mesh refinement, residual and matrix assembly, and other operations employing this datastructure, were outlined. We then applied MC to two simple model PBE problems, and compared the results to those obtainable with the standard non-adaptive methods currently in use. It was seen that substantial reductions on computational costs, as well as memory usage, are possible when compared to a uniform-mesh-based approach reaching the same accuracy. In the second paper [6], we develop an improved error estimator based on geometric solvent accessibility, and present a series of detailed numerical experiments for several complex biomolecules.

ACKNOWLEDGMENTS

M. Holst and N. Baker thank R. Bank and A. McCammon at UC San Diego for many enlightening discussions, and for their support and encouragement during the period in which this paper was written. M. Holst was supported in part by a UCSD Hellman Fellowship, and in part by NSF CAREER Award 9875856. N. Baker is a predoctoral fellow of the Howard Hughes Medical Institute and the Burroughs Wellcome Fund La Jolla Interfaces in Science training program. Additional support for N. Baker was provided by NSF and NIH grants to J. A. McCammon.

APPENDIX A. THE *a posteriori* ERROR ESTIMATOR IN MC

We now outline the derivation of the *a posteriori* error estimator that we use in MC for Galerkin approximations (3.1) to the solutions of nonlinear elliptic systems of the form (2.1)–(2.2). This is a special case of the derivations appearing in [50, 53] for more general nonlinear elliptic systems of tensor equations. In Appendix B, instantiate this estimator for the case of the nonlinear Poisson-Boltzmann equation. Our derivation here and in [50, 53] follows closely that of Verfürth [86, 87].

The starting point for our residual-based error estimator is the linearization inequality (4.4). In our setting of the weak formulation (2.10)–(2.11), we have:

$$(A.1) \quad C_1 \|F(u_h)\|_{H^{-1}(\Omega)} \leq \|u - u_h\|_{H^1(\Omega)} \leq C_2 \|F(u_h)\|_{H^{-1}(\Omega)}.$$

The linearization constants in (4.4) have been denoted as C_1 and C_2 . The norm of the nonlinear residual $F(\cdot)$ in the dual space of bounded linear functionals on $H^1(\Omega)$ is defined in a standard way:

$$(A.2) \quad \|F(u)\|_{H^{-1}(\Omega)} = \sup_{0 \neq v \in H^1(\Omega)} \frac{\langle F(u), v \rangle}{\|v\|_{H^1(\Omega)}}.$$

The numerator is the general nonlinear weak form $\langle F(u), v \rangle$ appearing in (2.11). In order to derive a bound on the weak form in the numerator we must first introduce quite a bit of notation that we have managed to avoid until now.

To begin, we assume that the domain Ω has been exactly triangulated with a set \mathcal{S} of shape-regular d -simplices (the finite dimension d is arbitrary throughout this discussion). A family of simplices will be referred to here as shape-regular if for all simplices in the family the ratio of the diameter of the circumscribing sphere to that of the inscribing sphere, is bounded by an absolute fixed constant, independent of the numbers and sizes of the simplices that may be generated through refinements. (For a more careful definition of shape-regularity and related concepts, see [34].) It will be convenient to introduce the following notation:

$$\begin{aligned}
\mathcal{S} &= \text{The complete set of shape-regular simplices which (exactly) triangulates } \Omega \\
\mathcal{N}(s) &= \text{The union of the faces contained in the set of simplices } s \text{ which lie on } \partial\Omega \\
\mathcal{I}(s) &= \text{The union of the faces contained in the set of simplices } s \text{ not contained in } \mathcal{N}(s) \\
\mathcal{F}(s) &= \mathcal{N}(s) \cup \mathcal{I}(s) \\
\omega_s &= \bigcup \{ \tilde{s} \in \mathcal{S} \mid s \cap \tilde{s} \neq \emptyset, \text{ where } s \in \mathcal{S} \} \\
\omega_f &= \bigcup \{ \tilde{s} \in \mathcal{S} \mid f \cap \tilde{s} \neq \emptyset, \text{ where } f \in \mathcal{F} \} \\
h_s &= \text{The diameter of the simplex } s \text{ (either circumscribing or inscribing sphere diameter)} \\
h_f &= \text{The diameter of the face } f \text{ (either circumscribing or inscribing sphere diameter)}.
\end{aligned}$$

When the argument to one of the face set functions \mathcal{N} , \mathcal{I} , or \mathcal{F} is in fact the entire set of simplices \mathcal{S} , we will leave off the explicit dependence on \mathcal{S} without danger of confusion. Referring back to Figure 4.2 is now convenient. The two darkened triangles in the left picture in Figure 4.2 represents the set ω_f for the face f shared by the two triangles. The clear triangles in the right picture in Figure 4.2 represents the set ω_s for the darkened triangle s in the center (the set ω_s also includes the darkened triangle).

Finally, we will also need some notation to represent discontinuous jumps in function values across faces interior to the triangulation. To begin, for any face $f \in \mathcal{N}$, let n_f denote the unit outward normal; for any face $f \in \mathcal{I}$, take n_f to be an arbitrary (but fixed) choice of one of the two possible face normal orientations. Now, for any $v \in L^2(\Omega)$ such that $v \in C^0(s) \forall s \in \mathcal{S}$, define the *jump function*:

$$(A.3) \quad [v]_f(x) = \lim_{\epsilon \rightarrow 0^+} v(x + \epsilon n_f) - \lim_{\epsilon \rightarrow 0^-} v(x - \epsilon n_f).$$

We now begin the analysis by splitting the volume and surface integrals in (2.11) into sums of integrals over the individual elements and faces, and we then employ the Green's integral identities again to work backward towards the strong form in each element:

$$\begin{aligned}
\langle F(u), v \rangle &= \int_{\Omega} (a \nabla u \cdot \nabla v + b(u)v) \, dx = \sum_{s \in \mathcal{S}} \int_s (a \nabla u \cdot \nabla v + b(u)v) \, dx \\
(A.4) \quad &= \sum_{s \in \mathcal{S}} \int_s (b(u) - \nabla \cdot (a \nabla u))v \, dx + \sum_{s \in \mathcal{S}} \int_{\partial s} n \cdot (a \nabla u)v \, ds
\end{aligned}$$

Using the fact that (3.1) holds for the solution to the discrete problem, we employ the jump function and write

$$\begin{aligned}
|\langle F(u_h), v \rangle| &= |\langle F(u_h), v - v_h \rangle| \\
&= \sum_{s \in \mathcal{S}} \int_s (b(u_h) - \nabla \cdot (a \nabla u_h))(v - v_h) \, dx + \sum_{s \in \mathcal{S}} \int_{\partial s} n \cdot (a \nabla u_h)(v - v_h) \, ds \\
&= \sum_{s \in \mathcal{S}} \int_s (b(u_h) - \nabla \cdot (a \nabla u_h))(v - v_h) \, dx + \sum_{f \in \mathcal{I}} \int_f [n \cdot (a \nabla u_h)]_f (v - v_h) \, ds
\end{aligned}$$

$$(A.5) \leq \sum_{s \in \mathcal{S}} (\|b(u_h) - \nabla \cdot (a \nabla u_h)\|_{L^2(s)} \|v - v_h\|_{L^2(s)}) + \sum_{f \in \mathcal{I}} (\|[n \cdot (a \nabla u_h)]_f\|_{L^2(f)} \|v - v_h\|_{L^2(f)}),$$

where we have applied the Cauchy-Schwarz inequality to each integral.

In order to bound the sums on the right, we will employ a standard technical tool known as a H^1 -quasi-interpolant I_h . An example of such an interpolant is due to Scott and Zhang [78], which we refer to as the SZ-interpolant (see also Clément's interpolant in [35]). Unlike point-wise polynomial interpolation, which is not always well-defined for functions in $H^1(\Omega)$, the SZ-interpolant I_h can be constructed quite generally for H^1 -functions on shape-regular meshes of 2- and 3-simplices. Moreover, it can be shown to have the following remarkable local approximation properties: For all $v \in H^1(\Omega)$, it holds that

$$(A.6) \quad \|v - I_h v\|_{L^2(s)} \leq C_s h_s \|v\|_{H^1(\omega_s)}, \text{ and } \|v - I_h v\|_{L^2(f)} \leq C_f h_f^{1/2} \|v\|_{H^1(\omega_f)}.$$

For the construction of the SZ-interpolant, and for a proof of the approximation inequalities in L^p -spaces for $p \neq 2$, see [78]. A simple construction and the proof of the first inequality can also be found in the appendix of [56]. (Note that this interpolant is only employed as a proof technique here; it is not required in the actual implementation of the error estimator.)

Employing now the SZ-interpolant by taking $v_h = I_h v$ in (A.5), using (A.6), we have

$$(A.7) \quad \begin{aligned} |\langle F(u_h), v \rangle| &\leq \sum_{s \in \mathcal{S}} C_s h_s \|b(u_h) - \nabla \cdot (a \nabla u_h)\|_{L^2(s)} \|v\|_{H^1(\omega_s)} \\ &\quad + \sum_{f \in \mathcal{I}} C_f h_f^{1/2} \|[n \cdot (a \nabla u_h)]_f\|_{L^2(f)} \|v\|_{H^1(\omega_f)} \\ &\leq \max\{C_s, C_f\} \left(\sum_{s \in \mathcal{S}} h_s^2 \|b(u_h) - \nabla \cdot (a \nabla u_h)\|_{L^2(s)}^2 \right)^{1/2} \left(\sum_{s \in \mathcal{S}} \|v\|_{H^1(\omega_s)}^2 \right)^{1/2} \\ &\quad + \left(\sum_{f \in \mathcal{I}} h_f \|[n \cdot (a \nabla u_h)]_f\|_{L^2(f)}^2 \right)^{1/2} \left(\sum_{f \in \mathcal{I}} \|v\|_{H^1(\omega_f)}^2 \right)^{1/2} \end{aligned}$$

where we have used the discrete Cauchy-Schwarz inequality on each separate sum to obtain the last inequality.

It is not difficult to show (cf. [86]) that the simplex shape regularity assumption makes it possible to establish the following two inequalities:

$$(A.8) \quad \sum_{s \in \mathcal{S}} \|v\|_{H^1(s)}^2 \leq D_s \|v\|_{H^1(\Omega)}^2, \text{ and } \sum_{f \in \mathcal{F}} \|v\|_{H^1(f)}^2 \leq D_f \|v\|_{H^1(\Omega)}^2,$$

where D_s and D_f depend on the shape regularity constants. Therefore, since $\mathcal{I} \subset \mathcal{F}$ and $\mathcal{N} \subset \mathcal{F}$, we have finally that

$$(A.9) \quad \begin{aligned} |\langle F(u_h), v \rangle| &\leq C_3 \|v\|_{H^1(\Omega)} \\ &\cdot \left(\sum_{s \in \mathcal{S}} h_s^2 \|b(u_h) - \nabla \cdot (a \nabla u_h)\|_{L^2(s)}^2 + \sum_{f \in \mathcal{I}} h_f \|[n \cdot (a \nabla u_h)]_f\|_{L^2(f)}^2 \right)^{1/2}, \end{aligned}$$

where $C_3 = \max\{C_s, C_f\} \times \max\{D_s^{1/2}, D_f^{1/2}\}$ depends on the shape regularity of the simplices in \mathcal{S} .

We can finally use this last result in (A.2) to achieve the upper bound in (A.1):

$$\|u - u_h\|_{H^1(\Omega)} \leq C_2 \|F(u_h)\|_{H^{-1}(\Omega)} = C_2 \sup_{0 \neq v \in H^1(\Omega)} \frac{\langle F(u_h), v \rangle}{\|v\|_{H^1(\Omega)}}$$

$$(A.10) \quad \leq C_2 C_3 \left(\sum_{s \in \mathcal{S}} h_s^2 \|b(u_h) - \nabla \cdot (a \nabla u_h)\|_{L^2(s)}^2 + \sum_{f \in \mathcal{I}} h_f \| [n \cdot (a \nabla u_h)]_f \|_{L^2(f)}^2 \right)^{1/2}.$$

We now make one final transformation that will turn this into a sum of element-wise error indicators that will be easier to work with in an implementation. We only need to account for the interior face integrals (which would otherwise be counted twice) when we combine the sum over the faces into the sum over the elements. This leave us with the following

Theorem A.1. *Let $u \in H^1(\Omega)$ be a weak solution of (2.1)–(2.2), or equivalently of (2.10)–(2.11). If the linearization inequality (4.4) is valid, then the following a posteriori error estimate holds for a Galerkin approximation u_h satisfying (3.1):*

$$(A.11) \quad \|u - u_h\|_{H^1(\Omega)} \leq C_0 \left(\sum_{s \in \mathcal{S}} \eta_s^2 \right)^{1/2},$$

where

$$C_0 = C_2 \cdot \max\{C_s, C_f\} \cdot \max\{D_s^{1/2}, D_f^{1/2}\},$$

and where the element-wise error indicator η_s is defined as:

$$(A.12) \quad \eta_s = \left(h_s^2 \|b(u_h) - \nabla \cdot (a \nabla u_h)\|_{L^2(s)}^2 + \frac{1}{2} \sum_{f \in \mathcal{I}(s)} h_f \| [n \cdot (a \nabla u_h)]_f \|_{L^2(f)}^2 \right)^{1/2}.$$

Proof. See the discussion above. \square

The element-wise error indicator in (A.12) is employed in MC in the general case of a nonlinear elliptic system of the form (2.1)–(2.2).

Of course, we can't perform the integrals in (A.12) exactly in most cases, so we employ quadrature in MC. Note that η_s is computable by quadrature, since most terms appearing in the definition depend only on the (available) computed solution u_h ; the remaining quantities (the normal vector n_q and the mesh parameters h_s and h_f) are completely geometrical and can be computed from the local simplex geometry information. The indicator is very inexpensive when compared to the typical cost of producing the discrete solution u_h itself; the number of function evaluations and arithmetic operations (for performing quadrature) is always linear in the total number of simplices.

APPENDIX B. AN *a posteriori* ERROR ESTIMATOR FOR THE POISSON-BOLTZMANN EQUATION

In Appendix A we derived the general form of the error estimator used in MC. Here, we use the estimator from Appendix A to instantiate an estimator specifically for the Poisson-Boltzmann equation. In this case, we have where $a(x) = \epsilon(x)I_{3 \times 3}$, $b(x, u) = \bar{\kappa}^2(x) \sinh(u) - f(x)$, where $f(x) = (4\pi e_c^2 / (k_B T)) \sum_{i=1}^{N_m} z_i \delta(x - x_i)$. This produces the following element-wise error indicator:

$$(B.1) \quad \eta_s = \left(h_s^2 \|\bar{\kappa}^2 \sinh(u_h) - f - \nabla \cdot (\epsilon \nabla u_h)\|_{L^2(s)}^2 + \frac{1}{2} \sum_{f \in \mathcal{I}(s)} h_f \| [n \cdot (\epsilon \nabla u_h)]_f \|_{L^2(f)}^2 \right)^{1/2}.$$

The element-wise error estimator in (B.1) is used for all numerical examples in the present paper. It is used in conjunction with a geometric solvent accessibility-based error estimator the numerical results appearing in the companion paper [6]. The jump discontinuities in the dielectric are detected by the second term in the sum containing the jump function, leading to refinement at the dielectric boundary as seen in Section 5 and in [6]. The delta functions produce steep gradients in the solution, which are also detected by the jump function, and the discrete approximation to the delta

functions themselves are detected directly by the volume integral in the first term in the estimator above.

REFERENCES

1. R. A. Adams, *Sobolev spaces*, Academic Press, San Diego, CA, 1978.
2. S. A. Allison, J. J. Sines, and A. Wierzbicki, *Solutions of the full Poisson-Boltzmann equation with application to diffusion-controlled reactions*, J. Phys. Chem. **93** (1989), 5819–5823.
3. D.N. Arnold, A. Mukherjee, and L. Pouly, *Locally adapted tetrahedral meshes using bisection*, 1997.
4. I. Babuška and W.C. Rheinboldt, *Error estimates for adaptive finite element computations*, SIAM J. Numer. Anal. **15** (1978), 736–754.
5. ———, *A posteriori error estimates for the finite element method*, Int. J. Numer. Meth. Engrg. **12** (1978), 1597–1615.
6. N. Baker, M. Holst, and F. Wang, *Adaptive multilevel finite element solution of the Poisson-Boltzmann equation II: refinement at solvent accessible surfaces in biomolecular systems*, J. Comput. Chem. (Submitted; in review).
7. R. Bank and M. Holst, *Painting finite element meshes*, (In preparation).
8. ———, *A new paradigm for parallel adaptive mesh refinement*, SIAM J. Sci. Statist. Comput. (Accepted; to appear).
9. R. E. Bank, *PLTMG: A software package for solving elliptic partial differential equations, users' guide 8.0*, Software, Environments and Tools, Vol. 5, SIAM, Philadelphia, PA, 1998.
10. R. E. Bank and T. F. Dupont, *An optimal order process for solving finite element equations*, Math. Comp. **36** (1981), no. 153, 35–51.
11. R. E. Bank, T. F. Dupont, and H. Yserentant, *The hierarchical basis multigrid method*, Numer. Math. **52** (1988), 427–458.
12. R. E. Bank, M. Holst, B. Mantel, J. Periaux, and C. H. Zhou, *CFD PPLTMG: Using a posteriori error estimates and domain decomposition*, ECCOMAS 98 (New York, NY), John Wiley & Sons, 1998.
13. R. E. Bank and H. D. Mittelmann, *Stepsize selection in continuation procedures and damped Newton's method*, J. Computational and Applied Mathematics **26** (1989), 67–77.
14. R. E. Bank and D. J. Rose, *Parameter selection for Newton-like methods applicable to nonlinear partial differential equations*, SIAM J. Numer. Anal. **17** (1980), no. 6, 806–822.
15. ———, *Global Approximate Newton Methods*, Numer. Math. **37** (1981), 279–295.
16. ———, *Analysis of a multilevel iterative method for nonlinear finite element equations*, Math. Comp. **39** (1982), no. 160, 453–465.
17. R. E. Bank and R. K. Smith, *A posteriori error estimates based on hierarchical bases*, SIAM J. Numer. Anal. **30** (1993), no. 4, 921–935.
18. ———, *Mesh smoothing using a posteriori error estimates*, SIAM J. Numer. Anal. **34** (1997), 979–997.
19. R. E. Bank and A. Weiser, *Some a posteriori error estimators for elliptic partial differential equations*, Math. Comp. **44** (1985), no. 170, 283–301.
20. R. E. Bank and J. Xu, *The hierarchical basis multigrid method and incomplete LU decomposition*, Seventh International Symposium on Domain Decomposition Methods for Partial Differential Equations (D. Keyes and J. Xu, eds.), AMS, Providence, Rhode Island, 1994, pp. 163–173.
21. ———, *An algorithm for coarsening unstructured meshes*, Numerische Mathematik **73** (1996), 1–36.
22. E. Bänsch, *An adaptive finite-element strategy for the three-dimensional time-dependent Navier-Stokes equations*, Journal of Computational and Applied Mathematics **36** (1991), 3–28.
23. ———, *Local mesh refinement in 2 and 3 dimensions*, Impact of Computing in Science and Engineering **3** (1991), 181–191.
24. P. Bastian, K. Birken, K. Johannsen, S. Lang, N. Neuss, H. Rentz-Reichert, and C. Wieners, *UG – a flexible software toolbox for solving partial differential equations*, 1998.
25. R. Beck, B. Erdmann, and R. Roitzsch, *KASKADE 3.0: An object-oriented adaptive finite element code*, Tech. Report TR95–4, Konrad-Zuse-Zentrum for Informationstechnik, Berlin, 1995.
26. J. Bey, *Adaptive grid manager: AGM3D manual*, Tech. Report 50, SFB 382, Math. Inst. Univ. Tübingen, 1996.
27. W. R. Bowen and A. O. Sharif, *Adaptive finite-element solution of the nonlinear Poisson-Boltzmann equation: A charged spherical particle at various distances from a charged cylindrical pore in a charged planar surface*, Journal of Colloid and Interface Science **187** (1997), 363–374.
28. A. Brandt, *Algebraic multigrid theory: The symmetric case*, Appl. Math. Comp. **19** (1986), 23–56.
29. Achi Brandt, Steve McCormick, and John Ruge, *Algebraic multigrid (AMG) for sparse matrix equations*, Sparsity and Its Applications (D.J. Evans, ed.), Cambridge Univ. Press, 1984.

30. S. C. Brenner and L. R. Scott, *The mathematical theory of finite element methods*, Springer-Verlag, New York, NY, 1994.
31. J. M. Briggs and J. A. McCammon, *Computation unravels mysteries of molecular biophysics*, *Computers in Physics* **6** (1990), no. 3, 238–243.
32. T. F. Chan, B. Smith, and J. Zou, *Overlapping Schwarz methods on unstructured meshes using non-matching coarse grids*, Tech. Report CAM 94-8, Department of Mathematics, UCLA, 1994.
33. T.F. Chan, S. Go, and L. Zikatanov, *Lecture notes on multilevel methods for elliptic problems on unstructured meshes*, Tech. report, Dept. of Mathematics, UCLA, 1997.
34. P. G. Ciarlet, *The finite element method for elliptic problems*, North-Holland, New York, NY, 1978.
35. Ph. Clément, *Approximation by finite element functions using local regularization*, R.A.I.R.O. **2** (1975), 77–84.
36. C. M. Cortis and R. A. Friesner, *An automatic three-dimensional finite element mesh generation system for the poisson-boltzmann equation*, *J. Comput. Chem.* **18** (1997), no. 13, 1570–1590.
37. ———, *Numerical solution of the Poisson-Boltzmann equation using tetrahedral finite-element meshes*, *J. Comput. Chem.* **18** (1997), no. 13, 1591–1608.
38. M. E. Davis and J. A. McCammon, *Solving the finite difference linearized Poisson-Boltzmann equation: A comparison of relaxation and conjugate gradient methods*, *J. Comput. Chem.* **10** (1989), no. 3, 386–391.
39. P. J. Davis, *Interpolation and approximation*, Dover Publications, Inc., New York, NY, 1963.
40. P. Debye and E. Hückel, *Physik. Z.* **24** (1923), 185.
41. R. S. Dembo, S. C. Eisenstat, and T. Steihaug, *Inexact Newton Methods*, *SIAM J. Numer. Anal.* **19** (1982), no. 2, 400–408.
42. R. A. DeVore and G. G. Lorentz, *Constructive approximation*, Springer-Verlag, New York, NY, 1993.
43. S. C. Eisenstat and H. F. Walker, *Globally Convergent Inexact Newton Methods*, Tech. report, Dept. of Mathematics and Statistics, Utah State University, 1992.
44. M. Friedrichs, R. Zhou, S Edinger, and R. Friesner, *Poisson-Boltzmann analytical gradients for molecular modeling calculations*, *J. Phys. Chem. B* **103** (1999), 3057–3061.
45. S. Fucik and A. Kufner, *Nonlinear differential equations*, Elsevier Scientific Publishing Company, New York, NY, 1980.
46. M. K. Gilson, K. A. Sharp, and B. H. Honig, *Calculating the electrostatic potential of molecules in solution: Method and error assessment*, *J. Comput. Chem.* **9** (1988), no. 4, 327–335.
47. W. Hackbusch, *Multi-grid methods and applications*, Springer-Verlag, Berlin, Germany, 1985.
48. ———, *Elliptic differential equations*, Springer-Verlag, Berlin, Germany, 1992.
49. ———, *Iterative solution of large sparse systems of equations*, Springer-Verlag, Berlin, Germany, 1994.
50. M. Holst, *Adaptive multilevel finite element methods on manifolds and their implementation in MC*, (In preparation; currently available as a technical report and User’s Guide to the MC software).
51. ———, *Multilevel methods for the Poisson-Boltzmann equation*, Ph.D. thesis, Numerical Computing Group, University of Illinois at Urbana-Champaign, 1993, Also published as Technical Report UIUCDCS-R-03-1821.
52. ———, *The Poisson-Boltzmann equation: Analysis and multilevel numerical solution (Monograph based on the Ph.D. Thesis [51])*, Tech. report, Applied Mathematics and CRPC, California Institute of Technology, 1994.
53. M. Holst and D. Bernstein, *Adaptive finite element solution of the initial-value problem in general relativity: Theory and algorithms*, *Comm. Math. Phys.* (Submitted; in review).
54. M. Holst and F. Saied, *Multigrid solution of the Poisson-Boltzmann equation*, *J. Comput. Chem.* **14** (1993), no. 1, 105–113.
55. ———, *Numerical solution of the nonlinear Poisson-Boltzmann equation: Developing more robust and efficient methods*, *J. Comput. Chem.* **16** (1995), no. 3, 337–364.
56. M. Holst and E. Titi, *Determining projections and functionals for weak solutions of the Navier-Stokes equations*, *Recent Developments in Optimization Theory and Nonlinear Analysis* (Providence, Rhode Island) (Yair Censor and Simeon Reich, eds.), *Contemporary Mathematics*, vol. 204, American Mathematical Society, 1997.
57. M. Holst and S. Vandewalle, *Schwarz methods: to symmetrize or not to symmetrize*, *SIAM J. Numer. Anal.* **34** (1997), no. 2, 699–722.
58. M. Holst and J. Xu, *The Poisson-Boltzmann Equation I: Approximation theory*, (In preparation).
59. ———, *The Poisson-Boltzmann Equation II: Fast numerical solution based on singular functions*, (In preparation).
60. B. Jayaram, K. A. Sharp, and B. Honig, *The electrostatic potential of B-DNA*, *Biopolymers* **28** (1989), 975–993.
61. A. H. Juffer, E. F. F. Botta, B. A. M. van Keulen, A. van der Ploeg, and H. J. C. Berendsen, *The electric potential of a macromolecule in a solvent: A fundamental approach*, *J. Comput. Phys.* **97** (1991), 144–171.
62. H. B. Keller, *Numerical methods in bifurcation problems*, Tata Institute of Fundamental Research, Bombay, India, 1987.
63. A. Liu and B. Joe, *Relationship between tetrahedron shape measures*, *BIT* **34** (1994), 268–287.

64. ———, *Quality local refinement of tetrahedral meshes based on bisection*, SIAM J. Sci. Statist. Comput. **16** (1995), no. 6, 1269–1291.
65. J.L. Liu and W.C. Rheinboldt, *A posteriori finite element error estimators for indefinite elliptic boundary value problems*, Numer. Funct. Anal. and Optimiz. **15** (1994), no. 3, 335–356.
66. ———, *A posteriori finite element error estimators for parametrized nonlinear boundary value problems*, Numer. Funct. Anal. and Optimiz. **17** (1996), no. 5, 605–637.
67. B. A. Luty, M. E. Davis, and J. A. McCammon, *Solving the finite-difference non-linear Poisson-Boltzmann equation*, J. Comput. Chem. **13** (1992), no. 9, 1114–1118.
68. J.M. Maubach, *Local bisection refinement for N -simplicial grids generated by relection*, SIAM J. Sci. Statist. Comput. **16** (1995), no. 1, 210–277.
69. E.P. Mucke, *Shapes and implementations in three-dimensional geometry*, Ph.D. thesis, Dept. of Computer Science, University of Illinois at Urbana-Champaign, 1993.
70. A. Mukherjee, *An adaptive finite element code for elliptic boundary value problems in three dimensions with applications in numerical relativity*, Ph.D. thesis, Dept. of Mathematics, The Pennsylvania State University, 1996.
71. A. Nicholls and B. Honig, *A rapid finite difference algorithm, utilizing successive over-relaxation to solve the Poisson-Boltzmann equation*, J. Comput. Chem. **12** (1991), no. 4, 435–445.
72. C. Niedermeier and K. Schulten, *Molecular dynamics simulations in heterogeneous dielectrics and Debye-Hückel media – application to the protein bovine pancreatic trypsin inhibitor*, Tech. report, Department of Physics and Beckman Institute, University of Illinois at Urbana-Champaign, 1990.
73. A. A. Rashin and J. Malinsky, *New method for the computation of ionic distribution around rod-like polyelectrolytes with helical distribution of charges. I. General approach and a nonlinearized Poisson-Boltzmann equation*, J. Comput. Chem. **12** (1991), no. 8, 981–993.
74. M.C. Rivara, *Algorithms for refining triangular grids suitable for adaptive and multigrid techniques*, International Journal for Numerical Methods in Engineering **20** (1984), 745–756.
75. ———, *Local modification of meshes for adaptive and/or multigrid finite-element methods*, Journal of Computational and Applied Mathematics **36** (1991), 79–89.
76. I.G. Rosenberg and F. Stenger, *A lower bound on the angles of triangles constructed by bisecting the longest side*, Math. Comp. **29** (1975), 390–395.
77. J. W. Ruge and K. Stüben, *Algebraic multigrid (AMG)*, Multigrid Methods (S. F. McCormick, ed.), Frontiers in Applied Mathematics, vol. 3, SIAM, Philadelphia, PA, 1987, pp. 73–130.
78. L. R. Scott and S. Zhang, *Finite element interpolation of nonsmooth functions satisfying boundary conditions*, Math. Comp. **54** (1990), no. 190, 483–493.
79. K. A. Sharp and B. Honig, *Calculating total electrostatic energies with the nonlinear Poisson-Boltzmann equation*, J. Phys. Chem. **94** (1990), 7684–7692.
80. ———, *Electrostatic interactions in macromolecules: Theory and applications*, Annu. Rev. Biophys. Biophys. Chem. **19** (1990), 301–332.
81. M. Stynes, *On faster convergence of the bisection method for all triangles*, Math. Comp. **35** (1980), 1195–1201.
82. C. Tanford, *Physical chemistry of macromolecules*, John Wiley & Sons, New York, NY, 1961.
83. P. Vanek, J. Mandel, and M. Brezina, *Algebraic multigrid on unstructured meshes*, Tech. Report UCD/CCM 34, Center for Computational Mathematics, University of Colorado at Denver, 1994.
84. ———, *Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems*, Tech. Report UCD/CCM 36, Center for Computational Mathematics, University of Colorado at Denver, 1995.
85. R. S. Varga, *Matrix iterative analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1962.
86. R. Verfürth, *A posteriori error estimates for nonlinear problems. Finite element discretizations of elliptic equations*, Math. Comp. **62** (1994), no. 206, 445–475.
87. ———, *A review of a posteriori error estimation and adaptive mesh-refinement techniques*, John Wiley & Sons Ltd, New York, NY, 1996.
88. J. Xu, *Iterative methods by space decomposition and subspace correction*, SIAM Review **34** (1992), no. 4, 581–613.
89. ———, *A novel two-grid method for semilinear elliptic equations*, Tech. report, Dept. of Mathematics, Penn State University, 1992.
90. ———, *Two-grid finite element discretization for nonlinear elliptic equations*, Tech. report, Dept. of Mathematics, Penn State University, 1992.
91. J. Xu and A. Zhou, *Local error estimates and parallel adaptive algorithms*, 1997, Preprint.
92. B. J. Yoon and A. M. Lenhoff, *A boundary element method for molecular electrostatics with electrolyte effects*, J. Comput. Chem. **11** (1990), no. 9, 1080–1086.
93. T. You and D. Bashford, *An analytical algorithm for the rapid determination of the solvent accessibility of points in a three-dimensional lattice around a solute molecule*, J. Comput. Chem. **16** (June, 1995), no. 6, 743–757.

94. D. M. Young, *Iterative solution of large linear systems*, Academic Press, New York, NY, 1971.

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF CALIFORNIA AT SAN DIEGO, LA JOLLA, CA 92093, USA.

E-mail address: `mholst@math.ucsd.edu`

DEPARTMENT OF CHEMISTRY AND DEPARTMENT OF MATHEMATICS SCIENTIFIC COMPUTATION GROUP, UNIVERSITY OF CALIFORNIA AT SAN DIEGO, LA JOLLA, CA 92093, USA.

E-mail address: `nbaker@wasabi.ucsd.edu`

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF CALIFORNIA AT IRVINE, IRVINE, CA 92697, USA.

E-mail address: `fwang@math.uci.edu`