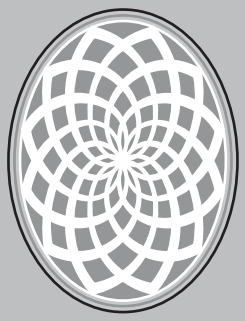

Industrial Mathematics Institute

2005:16

Adaptive finite element methods for elliptic PDEs based on conforming centroidal Voronoi Delaunay triangulations

L. Ju, M. Gunzburger and W.D. Zhao

IMI

Preprint Series

Department of Mathematics
University of South Carolina

ADAPTIVE FINITE ELEMENT METHODS FOR ELLIPTIC PDES BASED ON CONFORMING CENTROIDAL VORONOI DELAUNAY TRIANGULATIONS

LILI JU*, MAX GUNZBURGER†, AND WEIDONG ZHAO‡

Abstract. A new triangular mesh adaptivity algorithm for elliptic PDEs that combines a posteriori error estimation with centroidal Voronoi/Delaunay tessellations of domains in two dimensions is proposed and tested. The ability of the first ingredient to detect local regions of large error and the ability of the second ingredient to generate superior unstructured grids results in an mesh adaptivity algorithm that has several very desirable features, including the following. Errors are very well equidistributed over the triangles; at all levels of refinement, the triangles remain very well shaped, even if the grid size at any particular refinement level, when viewed globally, varies by several orders of magnitude; and the convergence rates achieved are the best obtainable using piecewise linear finite elements. This methodology can be easily extended to higher-order finite element approximations or mixed finite element formulations although only the linear approximation is considered in this paper.

Key words. Adaptive methods, finite element methods, centroidal Voronoi tessellation, Delaunay triangulation

AMS subject classifications. 65N50, 65N15

1. Introduction.

Adaptive grid generation techniques play an increasingly important role in the numerical solution of partial differential equations (PDEs). An essential ingredient of adaptive meshing techniques is a posteriori error estimators which are quantities that are computable once an approximate solution of the PDE has been determined. The key objectives in designing reliable and efficient a posteriori error estimators and mesh adaptivity techniques are that an existing mesh is refined in such a way that the errors in the approximate solution of the PDE on the new mesh are distributed as uniformly as possible, that those approximate solutions converge, as the mesh size decreases, to the exact solution as well as can be expected, and that the first two objectives are met with a relatively simple complexity. Both mesh adaptivity and a posteriori error estimators have been extensively studied, beginning in the late 70s [4–7] and followed by a vast literature. Here, we refer to [2, 31] for references on a posteriori error estimation and mesh adaptivity for elliptic PDEs.

The performance of adaptive methods for PDEs depends not only on the error estimators, but also on the techniques used for adaptively refining and generating meshes. In [14], a convergent adaptive algorithm was proposed for the linear finite element methods applied to the Poisson equation in two dimensions; a sequence of refined triangulations is defined based on an a posteriori error estimator and the convergence is proved. Another new family of adaptive algorithms was given in [25–27] and the convergence of the algorithms was also proved.

In many if not most adaptive methods for PDEs, the meshes are refined locally whenever some criterion based on a local error estimator is not satisfied on some elements; the mesh elsewhere in the domain is not changed. However, in an unrefined

*Department of Mathematics, University of South Carolina, Columbia, SC 29208. (ju@math.sc.edu). This work is partially supported by the University of South Carolina Research and Productive Scholarship Fund under the grant number RPS 13060-06-12306.

†School of Computational Science, Florida State University, Tallahassee, FL 32306-4120 (gunzburg@csit.fsu.edu).

‡School of Mathematics and System Sciences, Shandong University, Jinan, Shandong 250100, P.R. China (dzhao@sdu.edu.cn).

region, the errors could be so small that, because one has too many grid nodes there, computational resources are wasted. Thus, to achieve some sort of mesh optimality, it is more reasonable to coarsen the meshes in regions where errors are relatively small in addition to refining in regions where the errors are relatively large. For example, in [8], by introducing a coarsening step to the algorithm proposed in [25], an adaptive method is defined that results in certain optimal convergence rates in the energy norm.

In this paper, we propose an adaptive algorithm for linear finite element methods that can distribute the nodes in some optimal way according to a posterior error estimates, so that the error of the resulting approximate solution is distributed equally over the elements. To some extent, it is close to the mesh smoothing scheme proposed in [3]. We also would like to point out the techniques described in this paper can be easily extended to higher-order finite element approximations or mixed finite element formulations. The plan of the rest of the paper is as follows. In Sections 2 and 3, we respectively discuss the specific a posteriori error estimators and mesh generation and optimization methods that are used to define our mesh adaptivity algorithm. The mesh generation algorithm we use requires the definition of a density function which that algorithm uses to decide how grid points should be distributed. In Section 4, we first show how that density function can be related to the a posteriori error estimators and then we provide the description of our mesh adaptation algorithm. In Section 5, we use several computational experiments to demonstrate the effectiveness and efficiency of our mesh adaptation approach.

2. Error estimators for linear finite element methods. Let $\Omega \subset \mathbb{R}^2$ be a bounded domain with a Lipschitz boundary $\partial\Omega$. Consider the model elliptic partial differential equation with homogeneous boundary condition

$$\begin{cases} -\nabla \cdot (a\nabla u) = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega, \end{cases} \tag{2.1}$$

where $f \in L^2(\Omega)$ and $a \in C^1(\Omega)$ with $a(\mathbf{x}) \geq \widetilde{a} > 0$.

There are several types of a posteriori error estimators used in adaptive finite element methods, e.g., explicit error estimators, implicit error estimators, multilevel estimators, and averaging estimators. In this paper, we only use explicit a posteriori error estimators for adaptive mesh generation and refinement because they can be computed directly from the finite element approximate solution and the data of the problem. In the following, we first review some results about explicit a posteriori error estimators in the context of finite element methods for the model problem (2.1).

2.1. Finite element spaces and a priori error estimates. Assume that Ω is a polygonal domain with boundary $\partial\Omega$ and $\mathcal{T}$ is a conforming triangulation of Ω [13]. Denote by h_T the diameter of the triangle $T \in \mathcal{T}$ and by r_T the diameter of the largest circle that can be inscribed in T. Define the regularity ratio of the triangle T by $\kappa_T = h_T/r_T$. If there is a constant κ such that $\kappa_T \leq \kappa$ for all $T \in \mathcal{T}$, then we say that the triangulation $\mathcal{T}$ of Ω is regular. It is worth noting that the assumption of regularity permits partitions of the domain Ω into meshes that may contain elements of quite different sizes. This observation is very important for adaptive refinement. In the following, we will assume that $\mathcal{T}$ is regular.

Let p denote a nonnegative integer and $\mathbb{P}_p$ the space of polynomials of degree less than or equal to p. The finite element space of degree p associated with the triangulation $\mathcal{T}$ is defined by $V_h = \{v \in C(\overline{\Omega}) \mid v|_T \in \mathbb{P}_p(T) \ \forall\, T \in \mathcal{T}\}$. In this paper, for simplicity, we consider the case $p = 1$, i.e., V_h is the continuous piecewise linear

finite element space with respect to $\mathcal{T}$. But the techniques described in the remaining sections can be easily extended to other higher-order approximations.

In the a posteriori error analysis, it is also worthwhile to consider properties of certain patches of elements. Let the patch $\widetilde{T} \in \Omega$ be the union of the triangle T and the other triangles in $\mathcal{T}$ that share at least one common vertex with T. We define

$$h_{\widetilde{T}} = \max_{T' \subset \widetilde{T}} h_{T'} \qquad \text{and} \qquad r_{\widetilde{T}} = \max_{T' \subset \widetilde{T}} r_{T'};$$

then, the regularity of the patch $\widetilde{T}$ is measured by $\kappa_{\widetilde{T}} = h_{\widetilde{T}}/r_{\widetilde{T}}$. It is easy to see that the regularity of the triangulation $\mathcal{T}$ is inherited by each patch $\widetilde{T}$; see, e.g., [2].

Let V be the Hilbert space $H_0^1(\Omega)$. The weak form of problem (2.1) is to find $u \in V$ such that

$$B(u, v) = L(v) \qquad \forall\, v \in V,$$

where B is the bilinear form and L is the linear functional respectively defined by

$$B(u, v) = \int_\Omega a \nabla u \cdot \nabla v \, \mathrm{d}\mathbf{x} \qquad \text{and} \qquad L(v) = \int_\Omega f v \, \mathrm{d}\mathbf{x} \qquad \forall\, u, v \in V.$$

It is clear that $V_h \subset V$. Then, the finite element approximation $u_h \in V_h$ of the problem (2.1) is determined from the problem

$$B(u_h, v_h) = L(v_h) \qquad \forall\, v_h \in V_h.$$

For any $u \in V$, we define its energy norm $\|\cdot\|_E$ by $\|u\|_E = \big(B(u,u)\big)^{1/2}$. We denote by h the piecewise linear function with respect to $\mathcal{T}$ satisfying

$$h(\mathbf{x}) = \max_{T \in \mathcal{T} \text{ and } \mathbf{x} \in \overline{T}} h_T$$

for each vertex $\mathbf{x}$ of $\mathcal{T}$. We also assume that the exact solution $u \in H^2(\Omega)$. Let $e_h = u - u_h$ be the error of the approximate solution u_h, we then have the following classic results about a priori error estimates [13].

THEOREM 1. *There exist constants C_1 and C_2 independent of a and h such that*

$$\|e_h\|_E \le C_1 \|\sqrt{a} h^{k-1} |\nabla_k u|\|_{L^2(\Omega)}, \quad k = 1, 2, \tag{2.2}$$

and

$$\|e_h\|_{L^2(\Omega)} \le C_2 \|\sqrt{a} h^2 |\nabla_2 u|\|_{L^2(\Omega)}. \tag{2.3}$$

2.2. An explicit H^1-type a posteriori error estimator. Let $v \in V$ be chosen arbitrarily, then writing the integral over the whole domain Ω as a sum of integrals over individual triangles gives

$$B(e_h, v) = \sum_{T \in \mathcal{T}} \left\{ \int_T f v \, \mathrm{d}\mathbf{x} - \int_T (a \nabla u_h) \cdot \nabla v \, \mathrm{d}\mathbf{x} \right\}.$$

Let $\mathcal{E}_I$ denote the set of interior edges of $\mathcal{T}$. If T and T' share the common edge $\gamma \in \mathcal{E}_I$, define the jump in the normal flux across the edge γ by

$$[(a \nabla u_h) \cdot \mathbf{n}_\gamma] = (a \nabla u_h)|_T \cdot \mathbf{n}_T + (a \nabla u_h)|_{T'} \cdot \mathbf{n}_{T'}$$

where $\mathbf{n}_T$ is the unit outward normal vector to ∂T. Applying integration by parts and rearranging terms, we then can get

$$B(e_h, v) = \sum_{T\in\mathcal{T}} \int_T rv \, \mathrm{d}\mathbf{x} + \sum_{\gamma\in\mathcal{E}_I} \int_\gamma Rv \, \mathrm{d}s, \tag{2.4}$$

where $r = f + \nabla\cdot(a\nabla u_h)$ and $R = -[(a\nabla u_h)\cdot\mathbf{n}_\gamma]$.

For given $v \in V$, let $I_h v$ be the interpolant of v in V_h. Then, by the orthogonality property $B(e_h, I_h v) = 0$ and (2.4), we have

$$B(e_h, v) = \sum_{T\in\mathcal{T}} \int_T r(v - I_h v) \, \mathrm{d}\mathbf{x} + \sum_{\gamma\in\mathcal{E}_I} \int_\gamma R(v - I_h v) \, \mathrm{d}s \qquad \forall\, v \in V. \tag{2.5}$$

The identity (2.5) plays an important role, indirectly or directly, throughout an a posteriori error analysis of finite element approximations. Due to the coercivity of the bilinear form B on V, the approximation theory, and some norm equivalences, by choosing $v = e_h$, we can obtain the first a posteriori error estimate

$$\begin{aligned} \|e_h\|_E^2 &\le C\Big\{ \sum_{T\in\mathcal{T}} h_T^2 \|r\|_{L^2(T)}^2 + \sum_{\gamma\in\mathcal{E}_I} h_T \|R\|_{L^2(\gamma)}^2 \Big\} \\ &= C \sum_{T\in\mathcal{T}} \Big\{ h_T^2 \|r\|_{L^2(T)}^2 + \frac{1}{2} h_T \|R\|_{L^2(\partial T)}^2 \Big\}. \end{aligned} \tag{2.6}$$

Except for the constant C, all of the quantities on the right-hand side of (2.6) can be computed explicitly from the finite element solution u_h. Then we obtain an H^1-type local error estimator η_{T,H^1} associated with the element $T \in \mathcal{T}$ defined by

$$\eta_{T,H^1}^2 = h_T^2 \|r\|_{L^2(T)}^2 + \frac{1}{2} h_T \|R\|_{L^2(\partial T)}^2. \tag{2.7}$$

The inequality (2.6) shows that the true error e_h can be bounded from above in terms of the local error estimator η_{T,H^1}, i.e. when η_{T,H^1} is small, the true error e_h must also be small. This property is referred to as the *reliability* of the error estimator η_{T,H^1}. However, we cannot discern anything about the true error e_h on any particular triangle $T \in \mathcal{T}$ from the stability estimate (2.6). Adaptive numerical methods generally also need the fact that the true error e_h is also locally bounded from below by the local error estimator η_{T,H^1}. This type of property is referred to as the *efficiency* of the error estimator. By using properly chosen bubble functions, the efficiency of the explicit a posteriori error estimator η_{T,H^1} can also be proved. Details can be found in [2] and the references cited therein. We collect the the stability and efficiency results for the error estimator η_{T,H^1} in the following theorem.

THEOREM 2. *Let η_{T,H^1} be defined in* (2.7) *and let $\eta_{H^1}^2 = \sum_{T\in\mathcal{T}} \eta_{T,H^1}^2$. Then, there exist constants C_1 and C_2 depending only on the domain Ω, the coefficient function a, and the regularity of $\mathcal{T}$ such that*

$$C_1 \Big\{ \|e_h\|_E^2 + \sum_{T\in\mathcal{T}} h_T^2 \|f - \bar{f}\|_{L^2(T)}^2 \Big\} \le \|e_h\|_E^2 \le C_2 \eta_{H^1}^2, \tag{2.8}$$

where $\bar{f}$ denotes the mean value of f over T. Moreover, let T_γ denote the union of the triangles having γ as one of their edges; then, the local bound

$$\eta_{T,H^1}^2 \le C_2 \Big\{ \|e_h\|_{E,T_\gamma} + h_T^2 \|f - \overline{f}\|_{L^2(T_\gamma)} \Big\}$$

also holds.

2.3. An explicit L^2-type a posteriori error estimator. Duality arguments can be used to derive L^2-type a posteriori error estimators. The starting point for the application of this technique is the adjoint of the model problem: find $\phi_g \in V$ such that

$$B(v, \phi_g) = (g, v) \qquad \forall v \in V \tag{2.9}$$

with $g \in L^2(\Omega)$ and where $(\cdot, \cdot)$ denotes the $L^2(\Omega)$ inner product. It is assumed that this problem is regular in the sense that the solution $\phi_g \in H^2(\Omega) \cap V$ and there exists a constant C such that

$$\|\phi_g\|_{H^2(\Omega)} \leq C \|g\|_{L^2(\Omega)}. \tag{2.10}$$

This assumption is known to hold, in particular, if the domain Ω is convex. The specific choice $g = e_h$ in (2.9) then gives

$$\|e_h\|^2_{L^2(\Omega)} = B(e_h, \phi_{e_h}).$$

Then, we have

$$\|e_h\|^2_{L^2(\Omega)} \leq \sum_{T \in \mathcal{T}} \|r\|_{L^2(T)} \|\phi_{e_h} - I_h \phi_{e_h}\|_{L^2(T)} + \sum_{\gamma \in \mathcal{E}_I} \|R\|_{L^2(\gamma)} \|\phi_{e_h} - I_h \phi_{e_h}\|_{L^2(\gamma)}. \tag{2.11}$$

By the approximation theory again, combining the inequalities (2.10) with $g = e_h$ and (2.11), we obtain

$$\|e_h\|^2_{L^2(\Omega)} \quad \leq \quad C \sum_{T \in \mathcal{T}} \left\{ h_T^4 \|r\|^2_{L^2(T)} + h_T^3 \|R\|^2_{L^2(\partial T)} \right\}$$

which is similar to (2.6), the only difference being a higher-order scaling in the mesh size; this reflects the expectation of a high-order rate of convergence with respect to the L^2 norm. Let η_{T,L^2} denote the L^2-type local error estimator defined by

$$\eta^2_{T,L^2} = h_T^4 \|r\|^2_{L^2(T)} + h_T^3 \|R\|^2_{L^2(\partial T)}. \tag{2.12}$$

We summarize the results about this local error estimator in the following theorem [2].

THEOREM 3. *Suppose that the domain Ω is convex. Let η_{T,L^2} be defined in* (2.12) *and let $\eta^2_{L^2} = \sum_{T \in \mathcal{T}} \eta^2_{T,L^2}$. Then, there exists a constant C depending on the domain Ω, the coefficient function a, and the regularity of $\mathcal{T}$ such that*

$$\|e_h\|^2_{L^2(\Omega)} \leq \quad C \eta^2_{L^2}. \tag{2.13}$$

3. Mesh generation and mesh optimization. There have been many good algorithms developed for mesh generation and mesh optimization; e.g., see [12, 16, 19, 21, 28, 29]. In this paper, we focus on centroidal Voronoi tessellation based mesh generation as proposed in [15, 16, 19].

3.1. Conforming centroidal Voronoi-Delaunay triangulation. Given an open convex domain $\Omega \in \mathbb{R}^d$ and a set of distinct points $\{\mathbf{x}_i\}_{i=1}^n \subset \Omega$, define for each point $\mathbf{x}_i$, $i = 1, \ldots, n$, the corresponding Voronoi region V_i, $i = 1, \ldots, n$, by

$$V_i = \left\{ \mathbf{x} \in \Omega \;\mid\; \|\mathbf{x} - \mathbf{x}_i\| < \|\mathbf{x} - \mathbf{x}_j\| \quad \text{for } j = 1, \ldots, n \quad \text{and} \quad j \neq i \right\}.$$

Clearly, we have $\mathbf{x}_i \in V_i$, $V_i \cap V_j = \emptyset$ for $i \neq j$, and $\cup_{i=1}^n \overline{V}_i = \overline{\Omega}$ so that $\{V_i\}_{i=1}^n$ is a tessellation of Ω. We refer to $\{V_i\}_{i=1}^n$ as the *Voronoi tessellation* (VT) of Ω associated with the point set $\{\mathbf{x}_i\}_{i=1}^n$. A point $\mathbf{x}_i$ is called a *generator* and a subdomain $V_i \subset \Omega$ is referred to as the *Voronoi region* corresponding to the generator $\mathbf{x}_i$.

It is well known that the dual tessellation (in a graph-theoretical sense) to a Voronoi tessellation of Ω is a *Delaunay triangulation* (DT). It is easy to show that the vertices of the Voronoi regions V_i's are the circumcenters of the corresponding Delaunay triangles.

Given a density function $\rho(\mathbf{x})$ defined on Ω, for any region $V \subset \Omega$, we define the *mass centroid* $\mathbf{x}^*$ of V by

$$\mathbf{x}^* = \frac{\displaystyle\int_V \mathbf{y}\rho(\mathbf{y})\,\mathrm{d}\mathbf{y}}{\displaystyle\int_V \rho(\mathbf{y})\,\mathrm{d}\mathbf{y}}.$$

Definition 1. *We refer to a Voronoi tessellation* $\{(\mathbf{x}_i, V_i)\}_{i=1}^n$ *of* Ω *as a* centroidal Voronoi tessellation (CVT) [15] *if and only if the points* $\{\mathbf{x}_i\}_{i=1}^n$ *which serve as the generators of the associated Voronoi tessellation* $\{V_i\}_{i=1}^n$ *are also the mass centroids of those regions, i.e., if and only if we have that*

$$\mathbf{x}_i = \mathbf{x}_i^* \quad \textit{for } i = 1, \ldots, n.$$

The corresponding Delaunay triangulation is referred to as a centroidal Voronoi-Delaunay triangulations (CVDT).

It is worth noting that a CVT/CVDT may not be unique; see [15]. The extension of CVTs and CVDTs to general surfaces is discussed in [17].

Given any set of points $\{\widetilde{\mathbf{x}}_i\}_{i=1}^n$ on Ω and any tessellation $\{\widetilde{V}_i\}_{i=1}^n$ of Ω, we define the corresponding *energy* by

$$\mathcal{K}\big(\{(\widetilde{\mathbf{x}}_i, \widetilde{V}_i)\}_{i=1}^n\big) = \sum_{i=1}^n \int_{\widetilde{V}_i} \rho(\mathbf{y})\|\mathbf{y} - \widetilde{\mathbf{x}}_i\|^2\,\mathrm{d}\mathbf{y}\,.$$

It has be shown that $\mathcal{K}$ is minimized only if $\{(\widetilde{\mathbf{x}}_i, \widetilde{V}_i)\}_{i=1}^n$ forms a centroidal Voronoi tessellation [15]. Although $\mathcal{K}$ may not be directly identified with an energy of some physical system, it is often naturally associated with quantities such as error distortion, variance, and cost in many practical applications.

An important and very useful property of CVTs is that the energy is equally distributed over the Voronoi regions V_i's in an asymptotic way. For example, it was shown in [15] that, in the one-dimensional case,

$$\mathcal{K}_{V_i} \approx \mathcal{K}/n \qquad \text{for } i = 1, \ldots, n,$$

where $\mathcal{K}_{V_i} = \int_{V_i} \rho(\mathbf{x})\|\mathbf{x} - \mathbf{x}_i\|^2\,d\mathbf{x}$ and $\mathcal{K} = \sum_{i=1}^n \mathcal{K}_{V_i}$. For higher-dimensional cases, this property is only a conjecture but its validity has been verified through extensive numerical studies and is widely assumed in practical applications such as vector quantization. As a consequence of this equipartition property, CVTs have important geometric features, including the following.

- For a constant density function, the generators $\{\mathbf{x}_i\}_{i=1}^n$ are uniformly distributed; the Voronoi regions $\{V_i\}_{i=1}^n$ are all almost of the same size and, in the two-dimensional case, most of them are (nearly) congruent convex hexagons [15].

- For a non-constant density function, the generators $\{\mathbf{x}_i\}_{i=1}^n$ are still locally uniformly distributed, and it is conjectured [15] that, asymptotically, for some constant C,

$$\mathcal{K}_{V_i} = C\rho(\mathbf{x}_i)h_{V_i}^{d+2} \qquad \text{and} \qquad \frac{h_{V_i}}{h_{V_j}} \approx \left(\frac{\rho(\mathbf{x}_j)}{\rho(\mathbf{x}_i)}\right)^{\frac{1}{d+2}}, \tag{3.1}$$

  where h_{V_i} denotes the diameter of V_i and d is the dimension of Ω.

Thus, in principle, one could control the distribution of generators to obtain an equal distribution of the error by connecting the density function $\rho(\mathbf{x})$ to an a posteriori error estimator.

An often used algorithm for constructing CVT/CVDT is the Lloyd's method [15].

ALGORITHM 1. **(Lloyd's Method for CVT)** *Given a domain Ω, a density function $\rho(\mathbf{x})$ defined on Ω, and a positive integer n,*

0. *select an initial set of n points $\{\mathbf{x}_i\}_{i=1}^n$ in Ω;*
1. *construct the Voronoi regions $\{V_i\}_{i=1}^n$ of Ω associated with $\{\mathbf{x}_i\}_{i=1}^n$;*
2. *determine the mass centroids of the Voronoi regions $\{V_i\}_{i=1}^n$; these centroids form the new set of points $\{\mathbf{x}_i\}_{i=1}^n$;*
3. *if the new points meet some convergence criterion, return $\{(\mathbf{x}_i, V_i)\}_{i=1}^n$ and terminate; otherwise, go to step* 1.

An important property of Lloyd's algorithm is that the energy $\mathcal{K}$ of the Voronoi tessellation $\{(\mathbf{x}_i, V_i)\}_{i=1}^n$ decreases after each iteration [15]. A probabilistic version of Lloyd's method and its parallel implementation were suggested in [24].

If a CVDT mesh is to be used within a discretization method for a PDE, e.g., in a finite element method, some modifications are needed. An obvious one is that the CVDT mesh must conform with the boundary of the domain Ω, i.e., some of the CVDT nodes should be constrained to lie on the boundary so that the boundary conditions of the PDE problem can be enforced.[1]

One can, of course, pre-define a set of boundary mesh points and then determine an interior mesh that in some sense "conforms" with the boundary mesh. We choose to instead amend the CVT definition and construction algorithm so that the boundary mesh points are automatically selected in conjunction with the interior mesh points. This results in a better "fit" of the boundary and interior meshes.

First, we generalize the CVT definition. Assume that Ω is compact and the domain boundary $\partial\Omega$ is piecewise smooth; the set of singular points, e.g., corners, is denoted by $P_S = \{\mathbf{z}_i\}_{i=1}^k$. Denote by $\mathbf{Proj}(\mathbf{x})$ the process that projects $\mathbf{x} \in \Omega$ to the closest point to $\mathbf{x}$ on the boundary $\partial\Omega$. Let

$$P_I = \{\mathbf{x}_i \mid \overline{V}_i \cap \partial\Omega = \emptyset\} \qquad \text{and} \qquad P_B = \{\mathbf{x}_i \mid \overline{V}_i \cap \partial\Omega \neq \emptyset\}$$

so that P_I, the set of *interior Voronoi generators*, denotes the set of generators that have Voronoi regions that do not intersect the boundary and P_B, the set of *boundary Voronoi generators* denotes the set of generators that have Voronoi regions that do intersect the boundary.

DEFINITION 2. *A Voronoi Tessellation $\{(\mathbf{x}_i, V_i)\}_{i=1}^n$ of Ω is called a* conforming centroidal Voronoi tessellation (CfCVT) *if and only if the following properties are satisfied:*

- $P_S \subset \{\mathbf{x}_i\}_{i=1}^n$;

[1] If $\partial\Omega = \emptyset$, e.g., if Ω is the *surface* of a sphere, Voronoi-based discretizations of PDEs have been discussed in [18].

- $\mathbf{x}_i = \mathbf{x}_i^*$ for $\mathbf{x}_i \in P_I$;
- $\mathbf{x}_i = \mathbf{Proj}(\mathbf{x}_i^*)$ for $\mathbf{x}_i \in P_B - P_S$.

The corresponding dual triangulation is then called a conforming centroidal Voronoi Delaunay triangulation (CfCVDT). It is noted that the meaning of singular (corner) points is trivial in two-dimensional space but may need to be more rigorously defined in spaces higher than two dimensions.

An algorithm for constructing a CfCVT/CfCVDT was given in [16, 19] and can be described as follows.[2] We follow Algorithm 1 except that in step 2 the new set of generators are given by

–the centers of mass of the interior Voronoi regions;

–the projections onto the boundary of the centers of mass of the boundary Voronoi regions except if the boundary Voronoi region contains a point in P_S, in which case the new generator is that point.

For this approach, both the number of mesh points on the boundary and their location are not pre-determined. However, it is not difficult to show that the number of generators lying on the boundary will never decrease after the first iteration of Lloyd's method. The reason for this is that the nodes on the boundary cannot return to the interior of the domain since their Voronoi regions are obviously always boundary Voronoi regions. Thus, for this approach, the initial position of generators must be chosen well according to the density function ρ; for example, one could determine an ordinary CVT (with no points lying on the boundary) to use as an initial set of generators for the CfCVT construction algorithm.

In practical applications, the domain Ω is often non-convex and is possibly very complicated [19], so that a main difficulty associated with Lloyd's method for constructing CfCVDTs is the construction of the Voronoi regions. For this reason, we next propose an algorithm for constructing approximate CfCVDTs in two dimensions that does not require the construction of exact Voronoi tessellations.

3.2. Approximate CfCVDT construction. In this section, we propose an algorithm to construct approximate CfCVDTs; we will later use this algorithm within our adaptive methods for mesh generation and optimization. We describe our approach for the two-dimensional case in detail; the generalization to higher dimensions follows similar lines.

Currently, for mesh generation with conforming boundary requirements, *constrained Delaunay triangulations* (CDTs) have been widely used; see, e.g., [29, 30]. The main difference between CDT and standard DT is that some geometric constraints such as predetermined node position and node connectivities are added and strictly enforced during the CDT process. For example, the boundary of the domain can be triangulated first, and the resulting boundary triangulation is then used as a constraint on the conforming triangulation of the whole domain using CDT. It is worth noting that the dual tessellations of CDT generally is not an exact Voronoi tessellation, especially near the boundary.

Our algorithm for constructing approximate CfCVDTs is based on the CDT process. Assume that $\Omega \in \mathbb{R}^2$ is a domain with a polygonal boundary.[3] Denote by $P_S = \{\mathbf{z}\}_{i=1}^k$ its corner vertex set as before. An initial conforming triangulation $\mathcal{T}_0 = \{T_i\}_{i=1}^m$ of Ω is generated using the "TRIANGLE" software package [29] that uses the CDT process with a boundary mesh as a constraint and interior Delaunay

[2] Other modified techniques for constructing CfCVTs are given in [16].

[3] For domains with curved boundaries, the projection process **Proj** can be easily effected by a damped Newton's method, see [28].

refinement techniques, or by some other means. Denote by $P = \{\mathbf{x}_i\}_{i=1}^n$ the set of vertices of $\mathcal{T}_0$, by P_B the set of boundary vertices, and by P_I the set of interior vertices. The CDT process guarantees that $P_S \subset P_B$.

For each triangle $T_i = (\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, \mathbf{x}_{i_3}) \in \mathcal{T}_0$, we define

$$\mathbf{x}_{T_i} = \begin{cases} \text{circumcenter of } T_i & \text{if } T_i \text{ is an acute triangle;} \\ \text{the middle point of the longest edge of } T_i & \text{otherwise.} \end{cases}$$

Clearly, $\mathbf{x}_{T_i} \in \overline{T}_i$. For each vertex $\mathbf{x}_i$, we denote by $\{T_{i_k}\}_{k=1}^{m_i} \subset \mathcal{T}_0$ the set of triangles for which $\mathbf{x}_i$ is a vertex, counting in the counterclockwise direction.

Interior vertices. First, consider the case $\mathbf{x}_i \in P_I$, i.e., $\mathbf{x}_i$ is an interior vertex. Define U_i by

$$U_i = \text{the polygon formed by } \{\mathbf{x}_{T_{i_k}}\}_{k=1}^{m_i};$$

see Fig. 3.1. The polygon U_i can be regarded as an approximation to the Voronoi region V_i associated with $\mathbf{x}_i$. Let $\overline{\mathbf{x}}_i$ denote the center of mass of the U_i with respect to the density function ρ. Denote by $\{\alpha_{i_k}\}_{k=1}^{m_i}$ the associated angles around $\mathbf{x}_i$ corresponding to $\{T_{i_k}\}_{k=1}^{m_i}$. Define

$$\alpha = \begin{cases} \max\{\alpha_{i_k} \mid \overline{T}_{i_k} \cap \partial\Omega \neq \emptyset\} & \text{if } \overline{T}_{i_k} \cap \partial\Omega \neq \emptyset \text{ for some } i_k; \\ 0 & \text{otherwise} \end{cases}$$

and $\mathbf{e}_i$ denote the corresponding boundary edge opposite to the angle α_{i_k} such that $\alpha_{i_k} = \alpha$; see Fig. 3.1 for illustrations of some cases.

Now, select a parameter θ_{max} ($\pi > \theta_{max} > \pi/2$). Then, define

$$\mathbf{y}_i = \begin{cases} \overline{\mathbf{x}}_i & \text{if } \alpha < \theta_{max}; \\ \mathbf{Proj}_{\mathbf{e}_i}\mathbf{x}_i & \text{otherwise} \end{cases} \tag{3.2}$$

where $\mathbf{Proj}_{\mathbf{e}_i}\mathbf{x}_i$ denotes the projection of $\mathbf{x}_i$ onto the boundary edge $\mathbf{e}_i$. It is clear that $\mathbf{y}_i$ is still an interior vertex if $\alpha < \theta_{max}$; otherwise, it is a boundary vertex although $\mathbf{x}_i$ is an interior node.

Boundary vertices. Next, consider the case $\mathbf{x}_i \in P_B$, i.e., $\mathbf{x}_i$ is a boundary vertex. Let $\mathbf{e}_1$ and $\mathbf{e}_2$ denote the two boundary edges having $\mathbf{x}_i$ as the common end point, and let $\mathbf{z}_1$ and $\mathbf{z}_2$ denote the midpoints of $\mathbf{e}_1$ and $\mathbf{e}_2$, respectively; see Fig. 3.2. The approximate Voronoi region U_i of $\mathbf{x}_i$ is defined by

$$U_i = \text{the polygon formed by } \mathbf{z}_1, \{\mathbf{x}_{T_{i_k}}\}_{k=1}^{m_i}, \text{ and } \mathbf{z}_2;$$

see Fig. 3.2. Let $\overline{\mathbf{x}}_i$ denote the center of mass of the U_i associated with the density function ρ.

If $\mathbf{x}_i \in P_B - P_S$, denote by β_1 and β_2 the angles facing the boundary edges $\mathbf{e}_1$ and $\mathbf{e}_2$, respectively, in $\{T_{i_k}\}_{k=1}^{m_i}$; see Fig. 3.2 (right). Let

$$\beta = \max(\beta_1, \beta_2)$$

and select a parameter θ_{min} ($\pi/3 > \theta_{min} > 0$). Then, define

$$\mathbf{y}_i = \begin{cases} \mathbf{x}_i & \text{if } \mathbf{x}_i \in P_S; \\ \mathbf{Proj}_{\overline{\mathbf{z}_1\mathbf{z}_2}}\overline{\mathbf{x}}_i & \text{if } \mathbf{x}_i \in P_B - P_S \text{ and } \beta > \theta_{min}; \\ \overline{\mathbf{x}}_i & \text{if } \mathbf{x}_i \in P_B - P_S \text{ and } \beta \leq \theta_{min} \end{cases} \tag{3.3}$$

where $\mathbf{Proj}_{\overline{\mathbf{z}_1\mathbf{z}_2}}\mathbf{x}_i$ denotes the projection of $\mathbf{x}_i$ onto the segment $\overline{\mathbf{z}_1\mathbf{z}_2}$. It is clear that $\mathbf{y}_i$ is also a boundary vertex if $\mathbf{x}_i$ is a corner vertex, or $\mathbf{x}_i$ is a non-corner vertex but $\beta > \theta_{min}$; otherwise, $\mathbf{y}_i$ becomes an interior vertex although $\mathbf{x}_i$ is on the boundary (We also call it a *lifting process*).

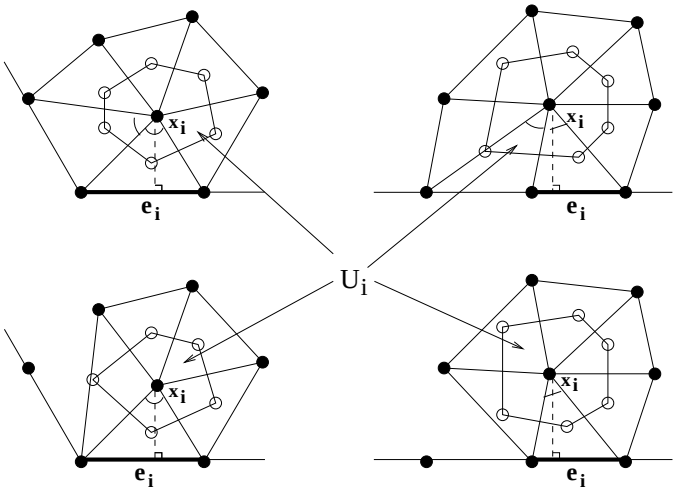

Fig. 3.1: The approximate Voronoi region U_i for the interior vertex $\mathbf{x}_i$.

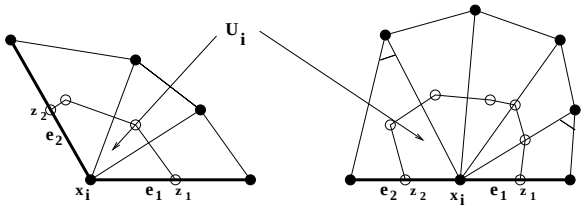

Fig. 3.2: The approximate Voronoi region U_i for the boundary vertex $\mathbf{x}_i$; left: corner vertex; right: non-corner vertex.

The approximate CfCVDT construction algorithm. We can now describe an algorithm for constructing an approximate CfCVDT of the domain Ω.

ALGORITHM 2. **(Modified Lloyd's method for approximate CfCVDT)** *Given a domain Ω, a density function $\rho(\mathbf{x})$ defined on Ω, and an initial triangulation $\mathcal{T}_0$ of Ω with vertices $\{\mathbf{x}_i\}_{i=1}^n$ generated using CDT,*

1. *determine $\{\mathbf{y}_i\}_{i=1}^n$ from $\{\mathbf{x}_i\}_{i=1}^n$ according to* (3.2) *and* (3.3)*;*
2. *set $\{\mathbf{x}_i\}_{i=1}^n = \{\mathbf{y}_i\}_{i=1}^n$ and reconstruct the boundary segments $\mathcal{E}_B$ from the new $\{\mathbf{x}_i\}_{i=1}^n$;*
3. *re-triangulate the domain Ω using CDT with $\{\mathbf{x}_i\}_{i=1}^n$ as the vertices and $\mathcal{E}_B$ as the boundary edges; the resulting triangulation is the new $\mathcal{T}$;*
4. *if the triangulation $\mathcal{T}$ meets some convergence criterion, return $\mathcal{T}$ and terminate; otherwise, go to step* 1.

In the remainder of this paper, we will use the notation $\mathcal{T}$=CfCVDT($\mathcal{T}_0$,Ω,ρ) to represent the output of Algorithm 2.

REMARK 1. *To prevent some vertices from frequently jumping back and forth between the boundary and the interior of the domain, more sophisticated controls are needed; for the sake of simplicity, we omit some details in Algorithm 2.*

REMARK 2. *Two user-defined parameters θ_{max} and θ_{min} corresponding respec-*

tively to the projection process and the lifting process are used to avoid bad-shaped triangles in the region close to the boundary. In our computational experiments, we set $\theta_{max} = 5\pi/9$ and $\theta_{min} = \pi/6$. These are only empirical values, but many experiments lead us to believe they are good choices.

4. CfCVDT-based adaptive finite element methods. Adaptive meshing methods for solving PDEs often takes the following standard form:

0. generate a coarse mesh $\mathcal{T}^{(0)}$ of the domain Ω and set $\ell = 0$;
1. solve the system produced by discretizing the PDE based on $\mathcal{T}^{(\ell)}$ and calculate the local error estimators;
2. if some convergence criteria is satisfied, terminate; otherwise, go to step 3;
3. refine the mesh $\mathcal{T}^{(\ell)}$ based on the local error estimators to get the next level of mesh $\mathcal{T}^{(\ell+1)}$ and set $\ell = \ell + 1$, then go to step 1.

In our adaptive method, we use CfCVDTs to refine and optimize the mesh at each level, but first we need to determine, from the error estimators, the density function used in the CfCVDT algorithm.

4.1. Determination of the density function. Let $\mathcal{T}^{(\ell)}$ denote the triangulation of Ω with vertices $\{\mathbf{x}_i^{(\ell)}\}_{i=1}^{n^{(\ell)}}$ at the refinement level ℓ. Let $\eta_{T,H^1}^{(\ell)}$ and $\eta_{T,L^2}^{(\ell)}$ represent the corresponding local H^1-type and L^2-type error estimators on $T \in \mathcal{T}^{(\ell)}$ at level ℓ defined by (2.7) and (2.12), respectively. A comparison of (2.2) and (2.8) and of (2.3) and (2.13) reveals that it is reasonable to divide both $\eta_{T,H^1}^{(\ell)}$ and $\eta_{T,L^2}^{(\ell)}$ by $\sqrt{a}$ in order to reflect the local variations of true error more accurately. Thus, we define

$$(\xi_{T,H^1}^{(\ell)})^2 = \frac{(\eta_{T,H^1}^{(\ell)})^2}{a_T} \qquad \text{and} \qquad (\xi_{T,L^2}^{(\ell)})^2 = \frac{(\eta_{T,L^2}^{(\ell)})^2}{a_T}, \tag{4.1}$$

where a_T is the mean value of $a(\mathbf{x})$ on the triangle T, i.e., $a_T = \int_T a(\mathbf{x})\, d\mathbf{x}/\text{Area}(T)$. In order to minimize

$$(\xi_{H^1}^{(\ell)})^2 = \sum_{T\in\mathcal{T}^{(\ell)}} (\xi_{T,H^1}^{(\ell)})^2 \qquad \text{or} \qquad (\xi_{L^2}^{(\ell)})^2 = \sum_{T\in\mathcal{T}^{(\ell)}} (\xi_{T,H^1}^{(\ell)})^2,$$

we need to distribute $(\xi_{T,H^1}^{(\ell)})^2$ or $(\xi_{T,L^2}^{(\ell)})^2$ equally over all triangles of $\mathcal{T}^{(\ell)}$.

Set

$$\widetilde{\rho}_{T,H^1}^{(\ell)} = \frac{(\xi_{T,H^1}^{(\ell)})^2}{h_T^4} \qquad \text{and} \qquad \widetilde{\rho}_{T,L^2}^{(\ell)} = \frac{(\xi_{T,L^2}^{(\ell)})^{4/3}}{h_T^4}. \tag{4.2}$$

We then uniquely determine two piecewise linear functions (with respect to $\mathcal{T}^{(\ell)}$) $\rho_{H^1}^{(\ell+1)}$ and $\rho_{L^2}^{(\ell+1)}$ on Ω such that for any vertex $\mathbf{x}_i^{(\ell)}$ of $\mathcal{T}^{(\ell)}$,

$$\rho_{H^1}^{(\ell+1)}(\mathbf{x}_i^{(\ell)}) = \frac{\sum\limits_{T\in S_i} \widetilde{\rho}_{T,H^1}^{(\ell)}}{\text{card}(S_i)} \qquad \text{and} \qquad \rho_{L^2}^{(\ell+1)}(\mathbf{x}_i^{(\ell)}) = \frac{\sum\limits_{T\in S_i} \widetilde{\rho}_{T,L^2}^{(\ell)}}{\text{card}(S_i)}, \tag{4.3}$$

where

$$S_i = \{T \in \mathcal{T}^{(\ell)} \mid \mathbf{x}_i^{(\ell)} \in \overline{T}\}.$$

Note that, in some sense, if the solution $u \in H^2(\Omega)$, we have

$$(\xi^{(\ell)}_{T,H^1})^2 \approx a_T |\nabla_2 u|^2 h_T^4 \qquad \text{and} \qquad (\xi^{(\ell)}_{T,L^2})^2 \approx a_T |\nabla_2 u|^2 h_T^6. \tag{4.4}$$

Combining (4.4) with the CVT/CVDT property (3.1) for $d = 2$, i.e.,

$$\frac{h_{V_i}}{h_{V_j}} \approx \Big(\frac{\rho(\mathbf{x}_j)}{\rho(\mathbf{x}_i)}\Big)^{1/4}$$

for a CVT $\{(\mathbf{x}_i, V_i)\}_{i=1}^n$ of Ω with respect to the density function ρ, it is then not difficult to verify that the CfCVDT mesh $\mathcal{T}^{(\ell+1)}$ generated by the density function $\rho^{(\ell+1)}_{H^1}$ or $\rho^{(\ell+1)}_{H^1}$ will approximately have the property that

$$(\xi^{(\ell+1)}_{T_i,H^1})^2 \approx (\xi^{(\ell+1)}_{T_j,H^1})^2 \qquad \text{or} \qquad (\xi^{(\ell+1)}_{T_i,L2})^2 \approx (\xi^{(\ell+1)}_{T_j,L^2})^2,$$

respectively, for any triangles $T_i, T_j \in \mathcal{T}^{(\ell+1)}$.

We will refer to the density functions $\rho^{(\ell+1)}_{H^1}$ and $\rho^{(\ell+1)}_{L^2}$ as the H^1-based and L^2-based density functions, respectively. From their defining formulas, it is easy to see that $\rho^{(\ell+1)}_{H^1}$ varies more rapidly than does $\rho^{(\ell+1)}_{L^2}$. We expect that CCDVT meshes generated using $\rho^{(\ell+1)}_{H^1}$ will produce a finite element approximation with smaller H^1 norm or energy error while those generated using $\rho^{(\ell+1)}_{L^2}$ will tend to have smaller L^2 norm error.

The most time consuming step in the calculations of $\rho^{(\ell+1)}_{H^1}(\mathbf{x})$ and $\rho^{(\ell+1)}_{L^2}(\mathbf{x})$ for any $\mathbf{x} \in \Omega$ is the nearest neighbor search operation since they are defined by interpolation with respect to an unstructured mesh. However, this task can be effected efficiently using the software package "ANN" [1] that is based on the K-D tree algorithm.

Remark 3. *In many practical applications, the coefficient in the model equation* (2.1) *is often a tensor product, i.e., a symmetric, positive definite matrix*

$$A(\mathbf{x}) = \begin{pmatrix} a_{11}(\mathbf{x}) & a_{12}(\mathbf{x}) \\ a_{21}(\mathbf{x}) & a_{22}(\mathbf{x}) \end{pmatrix}$$

rather than a scalar-valued function $a(\mathbf{x})$. Under this situation, if the difference between $a_{11}(\mathbf{x})$ and $a_{22}(\mathbf{x})$ is not large locally, then it is still reasonable to scale these estimators by

$$(\xi^{(\ell)}_{T,H^1})^2 = \frac{(\eta^{(\ell)}_{T,H^1})^2}{A_T} \qquad \textit{and} \qquad (\xi^{(\ell)}_{T,L^2})^2 = \frac{(\eta^{(\ell)}_{T,L^2})^2}{A_T}, \tag{4.5}$$

where $A_T = \int_T \sqrt{\det(A(\mathbf{x}))}\, d\mathbf{x}/Area(T)$; If $A(\mathbf{x})$ is strongly anisotropic, then it could not be handled correctly in this framework; the anisotropic CVT meshes proposed in [20], perhaps with some variations, may be able to deal with this case.

Remark 4. *If a higher-order finite element approximation is used, for example, $V_h = \{v \in C(\overline{\Omega}) \mid v|_T \in \mathbb{P}_k(T) \ \ \forall\, T \in \mathcal{T}\}$ where $k > 2$, then the discrete solution u_h is of k-th order convergence in H^1 norm and $(k+1)$-th order in L^2 norm. Let $\eta^{(\ell)}_{T,H^1}$ and $\eta^{(\ell)}_{T,L^2}$ be the accordingly derived local H^1 and L^2 type error estimators for this approximation; then by similar analysis, it is better to replace* (4.2) *by*

$$\widetilde{\rho}^{(\ell)}_{T,H^1} = \frac{(\xi^{(\ell)}_{T,H^1})^{\frac{4}{k+1}}}{h_T^4} \qquad \textit{and} \qquad \widetilde{\rho}^{(\ell)}_{T,L^2} = \frac{(\xi^{(\ell)}_{T,L^2})^{\frac{4}{k+2}}}{h_T^4}. \tag{4.6}$$

REMARK 5. *If the mixed finite element method is used, the density fucntion can be determined similar to the standard finite element approximation since all we need are just explicit a posterior local error estimators and their orders with respect to the local mesh size* h_T.

4.2. Adaptive algorithms based on CfCVDT. Let $\{E_i\}_{i=1}^{k^{(\ell)}}$ denote the set of edges of the ℓ-th level triangulation $\mathcal{T}^{(\ell)}$. Set $\rho_{E_i} = \rho(\mathbf{z}_i)$ for any density function ρ, where $\mathbf{z}_i$ denotes the midpoint of the edge E_i. We can now define our adaptive finite element method as follows.

ALGORITHM 3. **(CfCVDT-based adaptive finite element method)** *Given a domain* Ω, *an integer* $N_{max} > 0$ *(the maximum allowable number of mesh vertices), an integer* L_{max} *(the maximum allowable levels of refinements), and a parameter* $0 < \theta \leq 1$.

0. *Preprocessing: generate an initial coarse triangulation* $\mathcal{T}$ *of* Ω *using CDT or some other means, solve the PDE using a finite element method (FEM) on* $\mathcal{T}$, *and then determine the local error estimators* η_{T,H^1} *(or* η_{T,L^2}*) for all* $T \in \mathcal{T}$. *Construct the density function* ρ_{H^1} *(or* ρ_{L^2}*) using* (4.3) *and optimize* $\mathcal{T}$ *to obtain* $\mathcal{T}^{(0)} = CfCVDT(\mathcal{T}, \Omega, \rho_{H^1})$ *(or* $\mathcal{T}^{(0)} = CfCVDT(\mathcal{T}, \Omega, \rho_{L^2})$*) that becomes the new initial coarse mesh; let* $n^{(0)}$ *denote the number of vertices of* $\mathcal{T}^{(0)}$ *and set* $\ell = 0$.
1. *Solve the PDE using the FEM on* $\mathcal{T}^{(\ell)}$. *If* $\ell > L_{max}$ *or* $n^{(\ell)} > N_{max}$, *terminate; otherwise, go to step* 2.
2. *Determine the local error estimator* $\eta^{(\ell)}_{T,H^1}$ *(or* $\eta^{(\ell)}_{T,L^2}$*) for all* $T \in \mathcal{T}^{(\ell)}$.
3. *Construct* $\rho^{(\ell+1)}_{H^1}$ *(or* $\rho^{(\ell+1)}_{L^2}$*) using* (4.3) *and set the density function* $\rho = \rho^{(\ell+1)}_{H^1}$ *(or* $\rho = \rho^{(\ell+1)}_{L^2}$*)*.
4. *Determine* $\{\rho_{E_i}\}_{i=1}^{k^{(\ell)}}$ *and sort them in decreasing order.*
5. *Add* $\{\mathbf{z}_i\}_{i=1}^{k_\theta}$ *into the triangulation* $\mathcal{T}^{(\ell)}$, *where*

$$k_\theta = \max\Big\{k^* \;\Big|\; \sum_{i=1}^{k^*} \rho_{E_i} < \theta \sum_{i=1}^{k^{(l)}} \rho_{E_i}\Big\},$$

and then form, using CDT, the new intermediate triangulation $\widetilde{\mathcal{T}}^{(\ell+1)}$ *with* $n^{(\ell+1)} = n^{(\ell)} + k_\theta$ *vertices.*

6. *Optimize* $\widetilde{\mathcal{T}}^{(\ell+1)}$ *to obtain* $\mathcal{T}^{(\ell+1)} = CfCVDT(\widetilde{\mathcal{T}}^{(\ell+1)}, \Omega, \rho)$, *set* $\ell \leftarrow \ell + 1$, *then go to step* 1.

The parameter θ in Algorithm 3 is used here to control the refinement process [26]. The sorting procedure in step 4 can be implemented efficiently using a quick sorting algorithm.

5. Computational experiments. In this section, using computational experiments, we illustrate the effectiveness of CfCVDT-based adaptive finite element methods. Consider the problem

$$\begin{cases} -\nabla \cdot (a\nabla u) + bu = f & \text{in } \Omega, \\ u = g & \text{on } \partial\Omega, \end{cases} \tag{5.1}$$

where $b \in L^\infty(\Omega)$ with $b \geq 0$. Correspondingly, $r = f$ in (2.7) and (2.12) is changed to $r = f - bu_h$. Initial coarse meshes are either chosen to be a uniform Cartesian grid or are produced using the "TRIANGLE" package and are subsequently repeatedly

refined by our adaptive methods, i.e., Algorithm 3. For the sake of having something to compare to, we also find finite element approximations of the problem (5.1) using uniform refinements of the initial meshes. We set $L_{max} = 15$, N_{max} =20,000, and $\theta = 0.4$ for Algorithm 3. For our adaptive methods, the convergence rate CR with respect to the norm $\|\cdot\|$ at the refinement level ℓ is roughly computed by

$$\mathrm{CR} = \frac{2\log(\|e_{h,\ell}\|/\|e_{h,\ell-1}\|)}{\log(n_{\ell-1}/n_\ell)}, \tag{5.2}$$

where n_ℓ denotes the number of nodes and $e_{h,\ell}$ denotes the error $u - u_h$ at the refinement level ℓ.

We apply the commonly used q measure [22] to evaluate the quality of triangular meshes, where, for any triangle T, q is defined to be twice the ratio of the radius R_T of the largest inscribed circle and the radius r_T of the smallest circumscribed circle, i.e.,

$$q(T) = 2\frac{R_T}{r_T} = \frac{(b+c-a)(c+a-b)+a+b-c)}{abc},$$

where a, b, and c are side lengths of T. For a given triangulation $\mathcal{T}$, we define

$$q_{min} = \min_{T\in\mathcal{T}} q(T) \qquad \text{and} \qquad q_{avg} = \frac{1}{\mathrm{card}(\mathcal{T})}\sum_{T\in\mathcal{T}} q(T); \tag{5.3}$$

q_{min} measures the quality of the worst triangle and q_{avg} measures the average quality of the mesh $\mathcal{T}$.

5.1. Smooth solution with large gradients. The first illustrative problem is given as follows.

EXAMPLE 1. *Set* $\Omega = [-1,1]\times[-1,1]$, $a(x,y) = 10.0\cos(y)$, *and* $b(x,y) = x^2+y^2$. *The exact solution* u *is chosen to be*

$$u(x,y) = \frac{1.0}{(x-0.5)^2+(y-0.5)^2+0.01} - \frac{1.0}{(x+0.5)^2+(y+0.5)^2+0.01} \tag{5.4}$$

and f *and* g *are determined from* u *so that* (5.1) *is satisfied.*

It is easy to see that the exact solution u given in (5.4) is a smooth function, i.e., certainly, $u \in C^2(\Omega)$. Note that u achieves its maximum value $99\frac{101}{201}$ at the point $(0.5, 0.5)$ and its minimum value $-99\frac{101}{201}$ at the point $(-0.5, -0.5)$, but decays very quickly away from its extrema and thus has large gradients near these two points. Note also that $a(x,y)$ and $f(x,y)$ also have relatively rapid variations over Ω.

The initial coarse mesh (the input for step 0 of Algorithm 3) used for the solution of Example 1 is a uniform Cartesian grid consisting of 81 nodes; see Fig. 5.1. The corresponding CfCVDT meshes (the output of step 0) with the same number of nodes produced using the density functions ρ_{H^1} and ρ_{L^2} are also given in Fig. 5.1. Fig. 5.2 presents repeatedly refined meshes at some levels generated using Algorithm 3. The distributions of nodes in the CfCVDT-based adaptive meshes clearly show the accumulation of nodes in the vicinity of the two points near which large gradients in the solution occur. The CfCVDT-based adaptive meshes are "optimal" in the sense that all triangles remain well-shaped at all refinement levels, an observation which is supported by the values of q_{min} and q_{avg} given in Table 5.1. It is also clear that the CfCVDT meshes generated using the density function ρ_{H^1} tend to distribute the

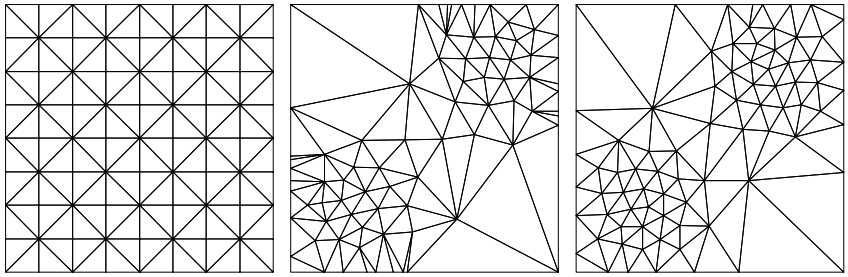

Fig. 5.1: Initial meshes for Example 1; left: a uniform Cartesian grid with 81 nodes; middle and right: the corresponding CfCVDT meshes with the same number of nodes generated using the density functions ρ_{H^1} and ρ_{L^2}, respectively.

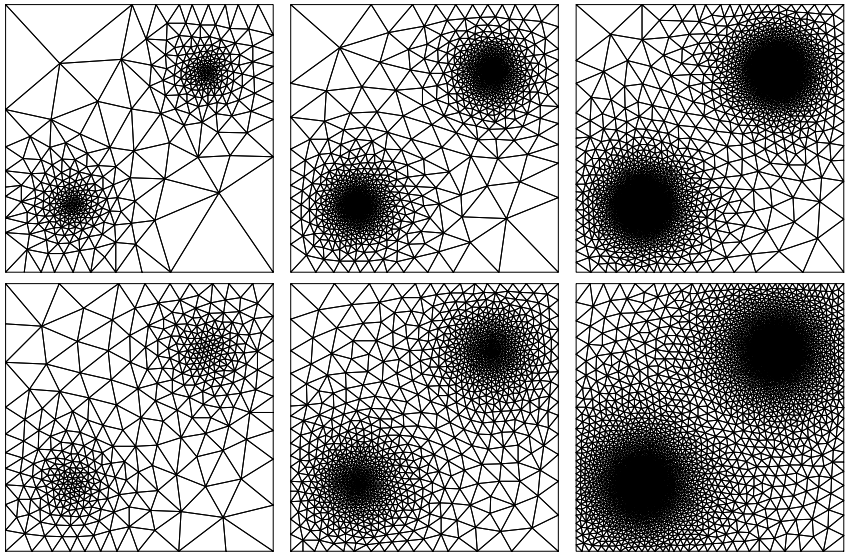

Fig. 5.2: Repeatedly refined adaptive meshes at some levels generated by the CfCVDT-based adaptive method for Example 1; top: 394, 1287, 4644 nodes using the density function ρ_{H^1}; bottom: 370, 1280, 4629 nodes using the density function ρ_{L^2}.

nodes in a slightly less uniform manner than those generated using the density function ρ_{L^2}; this observation which is also verified by the values of h_{max}/h_{min} in Table 5.1, can be explained by the fact that ρ_{H^1} has larger variations than does ρ_{L^2}.

Table 5.1 contains information about mesh quality, solution errors, and convergence rates at all refinement levels for different refinement strategies for Example 1; the corresponding plots of the error norms ($\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$) vs. the number of nodes are given in Fig. 5.3 where $|e_h|_{H^1(\Omega)}$ denotes the semi-H^1 norm defined by $\|\nabla e_h\|_{L^2(\Omega)}$. One observes that the two CfCVDT-based adaptive methods and the uniform refinement strategy achieve almost perfect convergence rates, i.e., 2 and 1

for $\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$, respectively.[4] These, of course, are the expected rates since the exact solution u of Example 1 belongs to $H^2(\Omega)$. The values of q_{min} and q_{avg} given in Table 5.1 demonstrate that the shape quality of the meshes resulting from the CfCVDT-based adaptive strategy is always very good at all levels for both density functions ρ_{L^2} and ρ_{H^1}, although the mesh sizes vary a lot over the Ω, e.g., h_{max}/h_{min} reaches 176.3 at the last level when ρ_{H^1} is used. Note that h_{max}/h_{min} tends to converge for our adaptive methods since u is smooth. Also, as expected, the adaptive method using ρ_{L^2} as the density function generated approximate solutions u_h having smaller $\|e_h\|_{L^2(\Omega)}$ relative to those obtained using ρ_{H^1}; on the other hand, the latter generated approximate solutions with (slightly) smaller $|e_h|_{H^1(\Omega)}$.

ℓ	n_ℓ	q_{min}	q_{avg}	h_{max}/h_{min}	$\|e_h\|_{L^2(\Omega)}$	CR	$\|e_h\|_{H^1(\Omega)}$	CR
Uniform refinement								
0	81	0.828	0.828	1.0	2.3260e+01		1.8187e+02	
1	289	0.828	0.828	1.0	2.2931e+00	3.34	1.0422e+02	0.80
2	1089	0.828	0.828	1.0	9.6260e-01	1.25	6.5176e+01	0.68
3	4225	0.828	0.828	1.0	2.7745e-01	1.80	3.5677e+01	0.87
4	16441	0.828	0.828	1.0	7.1278e-02	1.96	1.8242e+01	0.97
5	66409	0.828	0.828	1.0	1.8071e-02	1.98	9.2255e+00	0.98
Adaptive refinement using ρ_{H^1}								
0	81	0.280	0.805	6.6	7.6651e+00		1.5021e+02	
1	114	0.541	0.916	8.8	2.9722e+00	5.54	9.0709e+01	2.95
2	143	0.459	0.876	20.8	2.2858e+00	2.31	6.1156e+01	3.47
3	229	0.579	0.919	22.5	1.1666e+00	2.86	4.5971e+01	1.21
4	394	0.618	0.926	40.7	7.5734e-01	1.59	3.3524e+01	1.16
5	703	0.574	0.936	45.9	4.6479e-01	1.69	2.4081e+01	1.14
6	1287	0.611	0.941	59.5	2.9602e-01	1.49	1.7575e+01	1.04
7	2426	0.622	0.942	80.9	1.8046e-01	1.56	1.2760e+01	1.01
8	4644	0.651	0.944	114.5	9.7005e-02	1.91	9.2654e+00	0.99
9	8921	0.621	0.945	130.9	5.6155e-02	1.68	6.7420e+00	0.97
10	17095	0.598	0.944	164.5	3.1818e-02	1.75	4.8327e+00	1.02
11	33192	0.610	0.944	176.3	1.6909e-02	1.91	3.4671e+00	1.00
Adaptive refinement using ρ_{L^2}								
0	81	0.563	0.898	5.7	5.7395e+00		1.3966e+02	
1	127	0.679	0.931	7.1	2.1686e+00	4.33	9.0681e+01	1.92
2	211	0.579	0.924	13.4	1.0831e+00	2.74	5.9141e+01	1.68
3	370	0.638	0.938	11.9	5.8917e-01	2.17	4.0737e+01	1.33
4	681	0.675	0.941	17.7	3.2326e-01	1.97	2.9245e+01	1.09
5	1280	0.679	0.940	20.6	1.7163e-01	2.01	2.1150e+01	1.03
6	2412	0.601	0.944	23.5	9.4164e-02	1.90	1.4562e+01	1.18
7	4629	0.680	0.944	27.7	4.8922e-02	2.01	1.0621e+01	0.97
8	8883	0.649	0.944	34.9	2.6327e-02	1.90	7.4636e+00	1.08
9	17099	0.602	0.944	37.4	1.3566e-02	2.03	5.4503e+00	0.96
10	32875	0.609	0.944	41.5	7.2308e-03	1.93	3.8497e+00	1.06

Table 5.1: Mesh quality, solution errors, and convergence rates for different refinement strategies for Example 1.

From Table 5.1 and Fig. 5.3, one also observes that the CfCVDT-based adaptive methods are much more efficient relative to the uniform refinement strategy. For example, for the uniform refinement method, we have that $\|e_h\|_{L^2(\Omega)}$ =1.8071e-02 and $|e_h|_{H^1(\Omega)}$ =9.2255e+00 on the refined mesh with 66,049 nodes. However, for the adaptive methods, the values of these norms are 5.6155e-02 and 6.7420e+00,

[4]The convergence rate for $\|e_h\|_{L^2(\Omega)}$ for the adaptive method using ρ_{H^1} being a little erratic and slightly lower than 2.

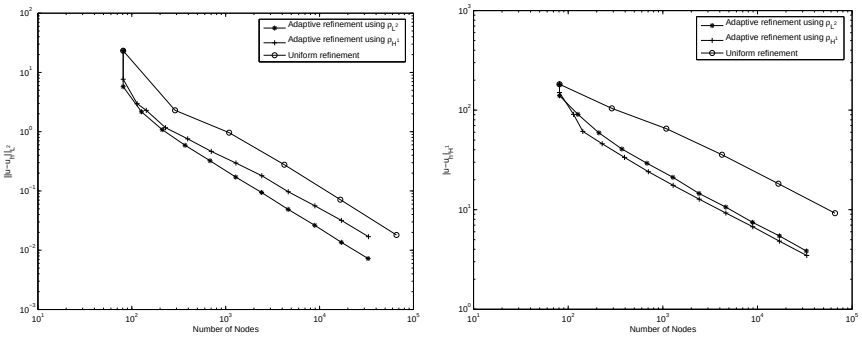

Fig. 5.3: Error norms vs. number of nodes for different refinement strategies for Example 1; left: $\|e_h\|_{L^2(\Omega)}$; right: $|e_h|_{H^1(\Omega)}$.

respectively, on the mesh with only $8,921$ nodes when ρ_{H^1} is used as the density function, and 2.6327e-02 and 7.4636e+00, respectively, on the mesh with only $8,883$ nodes when ρ_{L^2} is used as the density function. This means that, in the case of the $|e_h|_{H^1(\Omega)}$ norm, the CfCVDT-based adaptive methods are more than 8 times more efficient than the uniform refinement method if only considering the size of the resulting system.

An important optimal property of CfCVDT-based adaptive methods is the equidistribution of the errors over Ω. In order to verify this, we plot, in Fig. 5.4, a representative approximate solution u_h and the errors e_h for different refinement methods. It is obvious that the adaptive methods do indeed distribute the errors much more equally than does the uniform refinement method; note that the same scale is used for e_h in Fig. 5.4 for all the refinement methods.

5.2. Geometric singularity. The second illustrative problem is given as follows.

EXAMPLE 2. *Let* $\Omega = \Omega_{R_1} \cup \Omega_{R_2}$ *where* $\Omega_{R_1} = [-1,0] \times [0,1]$ *and* $\Omega_{R_2} = [0,1] \times [-1,1]$ *so that* Ω *is a non-convex,* Γ*-shaped region which induces a geometrically based singularity in the solution of the PDE at the origin* $(0,0)$*. In* (5.1)*, we set* $a(x,y) = 1$ *and* $b(x,y) = 0$*. We use the polar coordinates* (r,θ) *instead of the Cartesian coordinates* $(x,y) = (r\cos\theta, r\sin\theta)$ *to describe the exact solution* u *which is chosen to be*

$$u(r,\theta) = s\left(\frac{r-\delta_1}{\delta_2-\delta_1}\right) r^{2/3}\sin\left(\frac{2}{3}\theta\right) + w(r\cos\theta, r\sin\theta) \tag{5.5}$$

with $\delta_1 = 0.02$ *and* $\delta_2 = 0.25$*, where* s *is the cut-off function*

$$s(t) = \begin{cases} 1, & t < 0, \\ -6t^5 + 15t^4 - 10t^3 + 1, & 0 \le t \le 1, \\ 0, & t > 1, \end{cases}$$

and $w(x,y) = (x - x^3)(y^2 - y^4)$*. Note that* w *is a smooth function with* $w|_{\partial\Omega} = 0$*. Then,* f *and* g *are again determined from* u *so that* (5.1) *is satisfied.*

The exponent and angle factor $2/3$ in the exact solution (5.5) emulates the typical singular behavior of solutions of (5.1) in the Γ-shaped domain that has an interior

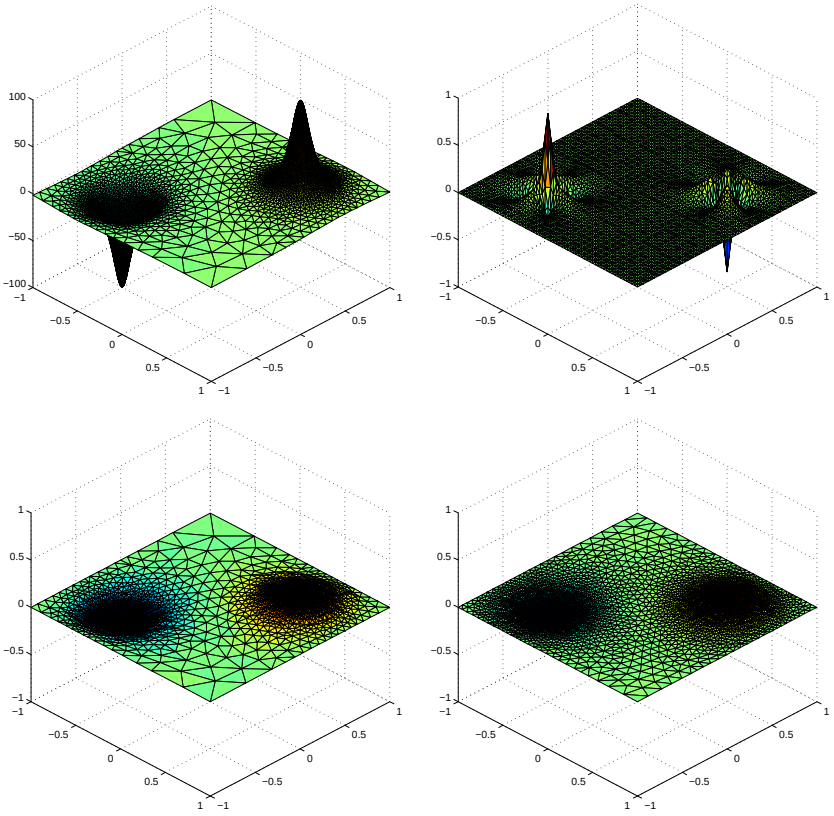

Fig. 5.4: Plots of an approximate solution and error distributions for different refinement methods for Example 1; top-left: a representative approximate solution u_h; top-right: the error e_h on the mesh with 4225 nodes obtained using the uniform refinement method; bottom-left: e_h on the mesh with 4644 nodes obtained using the CfCVDT-based adaptive method using ρ_{H^1}; bottom-right: e_h on the mesh with 4629 nodes obtained using the CfCVDT-based adaptive method using ρ_{L^2}.

angle equal to $3\pi/2$; see [23]. It is then easy to show that the exact solution u given in (5.5) only belongs to $H^{\frac{5}{3}-\epsilon}(\Omega)$ for any $\epsilon > 0$ and has a strong singularity at the origin. Again, this is the typical regularity one can expect for solutions of (5.1) in an Γ-shaped domain. Note that, in particular, $u \notin H^2(\Omega)$.

The initial coarse mesh (the input for step 0 of Algorithm 3) used for the solution of Example 2 is a uniform Cartesian grid consisting of 65 nodes; see Fig. 5.5. The corresponding CfCVDT meshes (the output of step 0) with the same number of nodes produced using the density functions ρ_{H^1} and ρ_{L^2} are also given in Fig. 5.5. Fig. 5.6 presents repeatedly refined meshes at some levels generated using Algorithm 3. The distributions of nodes in the CfCVDT-based adaptive meshes clearly show the accumulation of nodes near the origin where the singularity in the solution occurs. In order to better visualize the extent of the accumulation, 16-fold magnifications of the meshes near the origin are also included in Fig. 5.6. It is easy to see that again CfCVDT meshes generated using the density function ρ_{H^1} have a little higher con-

centration of nodes near the singular point than the ones generated using ρ_{L^2}. Also, once again, all triangles remain well-shaped at all refinement levels, an observation that is supported by the values of q_{min} and q_{avg} listed in Table 5.2.

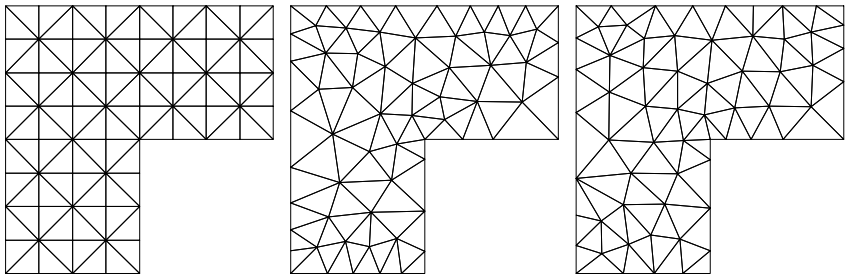

Fig. 5.5: Initial meshes for Example 2; left: a uniform Cartesian grid with 65 nodes; middle and right: the corresponding CfCVDT meshes with the same number of nodes generated using the density functions ρ_{H^1} and ρ_{L^2}, respectively.

Table 5.2 contains information about mesh quality, solution errors, and convergence rates at all refinement levels for different refinement strategies for Example 2; the corresponding plots of the error norms ($\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$) vs. the number of nodes are given in Fig. 5.7. The convergence rates for the uniform refinement method are about 1.48 and 0.85 for $\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$, respectively. These rates are a little better than the values 4/3 and 2/3, respectively, that finite element theory predicts for an exact solution $u \in H^{\frac{5}{3}-\epsilon}(\Omega)$; this behavior can possibly be explained by the superconvergence property of piecewise linear finite element approximations on uniform grids and by the fact that the mesh resolution may still not be fine enough to achieve asymptotic convergence rates; see [9]. However, one sees that the CfCVDT-based adaptive methods still achieve almost perfect convergence rates, i.e., 2 and 1 for $\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$, respectively. The values of q_{min} and q_{avg} given in Table 5.2 demonstrate that the quality of the meshes produced by the CfCVDT-based adaptive methods is always very good for both density functions ρ_{H^1} and ρ_{L^2}, even though the mesh sizes vary greatly over the Ω, e.g., h_{max}/h_{min} reaches 194.7 at the last level when the density function ρ_{H^1} is used. It is interesting to observe that, since u does not belong to $H^2(\Omega)$, h_{max}/h_{min} tends to monotonically increase for the adaptive methods. The CfCVDT adaptive method using the density function ρ_{L^2} generated approximate solutions with almost the same values of $\|e_h\|_{L^2(\Omega)}$ as that obtained using the density function ρ_{H^1}; on the other hand, the latter generated approximate solutions with (slightly) smaller values of $|e_h|_{H^1(\Omega)}$.

From Table 5.2 and Fig. 5.7, one also observes that the CfCVDT-based adaptive methods are much more efficient relative to the uniform refinement strategy. For example, for the uniform refinement method, we have that $\|e_h\|_{L^2(\Omega)}$ =1.9910e-04 and $|e_h|_{H^1(\Omega)}$ =2.8144e-02 on the refined mesh with $49,665$ nodes. However, for the adaptive methods, the values of these norms are 1.9454e-04 and 2.8889e-02, respectively, on the mesh with only $4,455$ nodes when ρ_{H^1} is used as the density function, and 1.2593e-04 and 2.6637e-02, respectively, on the mesh with only $6,847$ nodes when ρ_{L^2} is used as the density function. This means that, in the case of the $\|e_h\|_{H^1(\Omega)}$ norm, the CfCVDT-based adaptive methods are more than 10 times more efficient than the uniform refinement method if only considering the size of the resulting system.

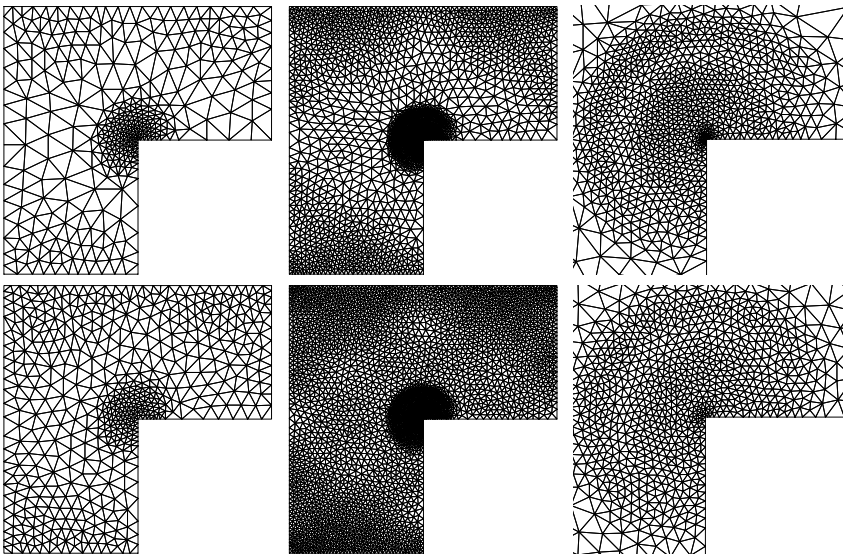

Fig. 5.6: Refined adaptive meshes at some levels generated by the CfCVDT-based adaptive method for Example 2; top, left to right: for the density function ρ_{H^1}, 400 and 2379 node meshes and the 2379 node case near the singular point magnified 16 times; bottom, left to right: for the density function ρ_{L^2}, 565 and 3617 nodes and the 3617 node case near the singular point magnified 16 times.

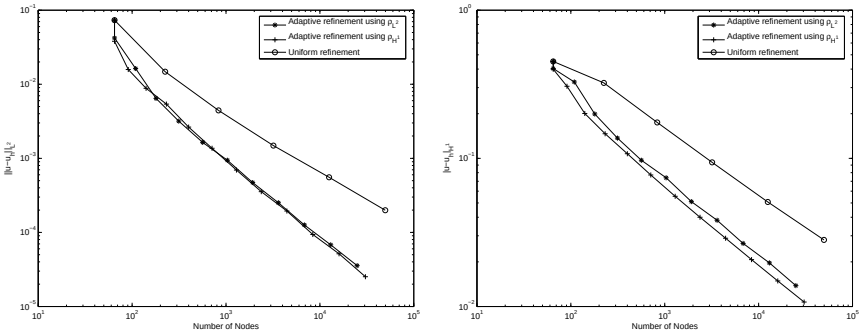

Fig. 5.7: Error norms vs. number of nodes for different refinement strategies for Example 2; left: $\|e_h\|_{L^2(\Omega)}$; right: $|e_h|_{H^1(\Omega)}$.

We display a representative approximate solution u_h and the errors e_h for different refinement methods in Fig. 5.8. It is again obvious that the CfCVDT-based adaptive methods distribute the errors much more equally over the triangles than does the uniform refinement method.

5.3. Interface singularity. The third illustrative problem is given as follows.

ℓ	n_ℓ	q_{min}	q_{avg}	h_{max}/h_{min}	$\|e_h\|_{L^2(\Omega)}$	CR	$\|e_h\|_{H^1(\Omega)}$	CR
Uniform refinement								
0	65	0.828	0.828	1.0	7.3322e-02		4.5081e-01	
1	225	0.828	0.828	1.0	1.4759e-02	2.31	3.2282e-01	0.48
2	833	0.828	0.828	1.0	4.4383e-03	1.73	1.7465e-01	0.88
3	3201	0.828	0.828	1.0	1.4867e-03	1.58	9.3882e-02	0.89
4	12545	0.828	0.828	1.0	5.5462e-04	1.42	5.0745e-02	0.89
5	49665	0.828	0.828	1.0	1.9910e-04	1.48	2.8144e-02	0.85
Adaptive refinement using ρ_{H^1}								
0	65	0.745	0.929	2.1	3.8000e-02		3.9914e-01	
1	91	0.653	0.915	3.9	1.5791e-02	5.22	3.0636e-01	1.57
2	141	0.602	0.911	6.2	8.8165e-03	2.66	2.0079e-01	1.93
3	232	0.536	0.915	9.2	5.4002e-03	1.97	1.4655e-01	1.27
4	400	0.505	0.923	10.3	2.6396e-03	2.63	1.0745e-01	1.14
5	711	0.579	0.934	15.2	1.3676e-03	2.29	7.7155e-02	1.15
6	1293	0.477	0.934	19.0	6.9433e-04	2.27	5.5410e-02	1.11
7	2379	0.442	0.937	32.2	3.5445e-04	2.21	3.9999e-02	1.07
8	4455	0.490	0.940	67.4	1.9454e-04	1.91	2.8889e-02	1.04
9	8419	0.480	0.941	86.7	9.3771e-05	2.29	2.0725e-02	1.04
10	16005	0.595	0.944	196.6	5.1481e-05	1.87	1.4871e-02	1.03
11	30576	0.517	0.943	294.7	2.5224e-05	2.20	1.0716e-02	1.01
Adaptive refinement using ρ_{L^2}								
0	65	0.645	0.918	2.2	4.2120e-02		4.0369e-01	
1	109	0.731	0.928	2.5	1.6311e-02	3.67	3.2760e-01	0.81
2	180	0.602	0.931	3.6	6.4383e-03	3.71	1.9912e-01	1.99
3	314	0.661	0.933	4.4	3.1812e-03	2.53	1.3687e-01	1.35
4	565	0.648	0.932	5.5	1.6431e-03	2.25	9.7091e-02	1.17
5	1035	0.703	0.938	6.1	9.4153e-04	1.84	7.3907e-02	0.90
6	1925	0.642	0.941	7.9	4.7074e-04	2.23	5.1002e-02	1.20
7	3617	0.649	0.941	9.6	2.5132e-04	1.99	3.8175e-02	0.92
8	6847	0.579	0.944	15.5	1.2593e-04	2.17	2.6637e-02	1.13
9	13048	0.576	0.945	19.0	6.7958e-05	1.91	1.9695e-02	0.94
10	24933	0.579	0.945	28.4	3.5607e-05	2.00	1.3835e-02	1.09

Table 5.2: Mesh quality, solution errors, and convergence rates for different refinement strategies for Example 2.

EXAMPLE 3. *Let* $\Omega = [-1,1]\times[-1,1]$, $b(x,y) = 0$, *and*

$$a(x,y) = \begin{cases} 1 & (x,y) \in \Omega_1 \cup \Omega_3 \\ 5 & (x,y) \in \Omega_2 \cup \Omega_4, \end{cases}$$

where $\Omega_1 = (0,1)\times(0,1)$, $\Omega_2 = (-1,0)\times(0,1)$, $\Omega_3 = (-1,0)\times(-1,0)$ *and* $\Omega_4 = (0,1)\times(-1,0)$*; see the top-left image in Fig. 5.9. Note that the coefficient* a *is discontinuous across the two lines* $x = 0$ *and* $y = 0$. *This problem is thus an interface problem* [10]. *The exact solution* u *is chosen to be*

$$u(r,\theta) = r^\alpha\big(p_i\cos(\alpha\theta) + q_i\sin(\alpha\theta)\big) \quad \text{in } \Omega_i, \tag{5.6}$$

with $0 < \alpha < 1$ *and for* $i = 1,2,3,4$. *With a normalization condition such as*

$$\sum_{i=1}^{4}(p_i + q_i) = 1,$$

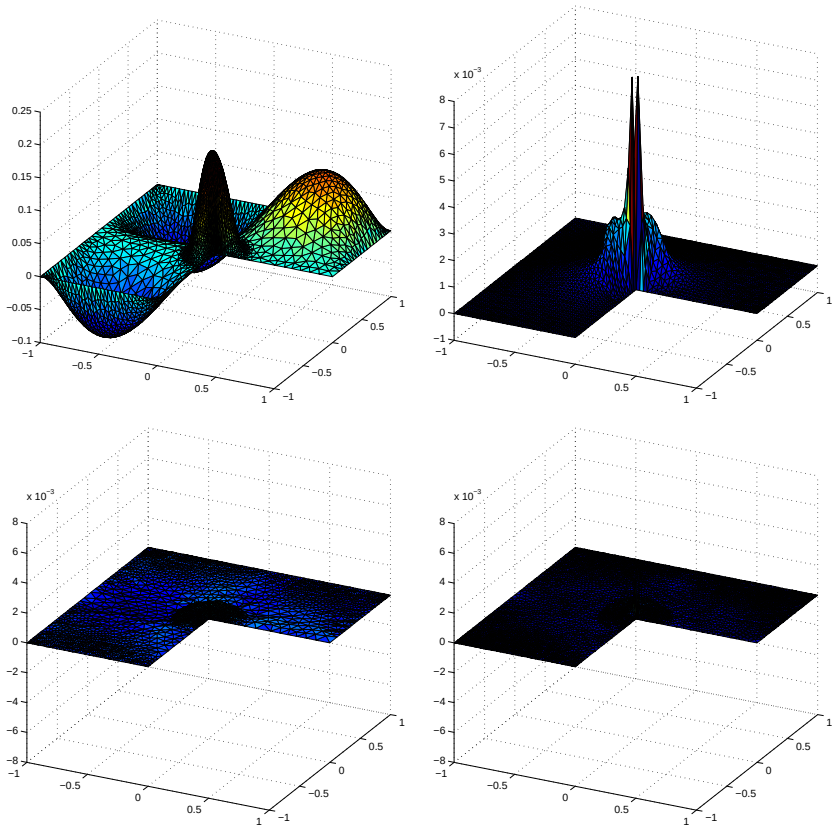

Fig. 5.8: Plots of an approximate solution and error distributions for different refinement methods for Example 2; top-left: a representative approximate solution u_h; top-right: the error e_h on the mesh with 3201 nodes obtained using the uniform refinement method; bottom-left: e_h on the mesh with 2379 nodes obtained using the CfCVDT-based adaptive method using ρ_{H^1}; bottom-right: e_h on the mesh with 3617 nodes obtained using the CfCVDT-based adaptive method using ρ_{L^2}.

we can easily solve a Sturm-Liouville problem to find that if $\alpha \approx 0.53544094560$ *and*

$$\begin{aligned}\{(p_i,q_i)\}_{i=1}^4 \quad \approx \quad &\big\{(14.535673,-0.839562),\,(0.429608,1.621108),\\ &(-13.043759,6.469236),\,(-0.478922,-0.693383)\big\},\end{aligned}$$

then u *given by* (5.6) *satisfies*

$$\nabla\cdot(a\nabla u)=0 \quad \textit{in } \Omega_i,$$

and the interface conditions

$$\lim_{\theta\to(i\pi/2)^+} u(r,\theta)=\lim_{\theta\to(i\pi/2)^-} u(r,\theta)$$

and

$$\lim_{\theta\to(i\pi/2)^+} a\frac{\partial u(r,\theta)}{\partial\theta}=\lim_{\theta\to(i\pi/2)^-} a\frac{\partial u(r,\theta)}{\partial\theta}$$

for $i = 1, 2, 3, 4$. Note that $u \in H^{1+\alpha-\epsilon}(\Omega)$ for any $\epsilon > 0$ and has a strong singularity at the origin.

The initial coarse mesh, i.e., the input for step 0 of Algorithm 2, used for the solution of Example 3 is the same as that used for Example 1, i.e., a uniform Cartesian grid consisting of 81 nodes; see Fig. 5.9. For this interface problem, we need to impose a restriction on the computational meshes: no triangle at any refinement level can straddle the interface lines $x = 0$ and $y = 0$, i.e., the triangulations have to conform to the interfaces, so that the discontinuities of a only occur across mesh edges. Uniformly refined grids automatically meet this constraint. Some modifications can be made to Algorithm 2 so that CfCVDT adapted meshes also satisfy this requirement. The initial CfCVDT meshes, i.e., the output of step 0 of Algorithm 2, determined using the density functions ρ_{H^1} and ρ_{L^2} are also given in Fig. 5.9. Fig. 5.10 displays refined meshes at some levels generated by the CfCVDT-based adaptive method. The distributions of nodes in the CfCVDT-adapted meshes clearly show the accumulation of nodes near the origin. In order to better visualize the extent of the accumulation, a portion of the meshes near the origin, magnified 1024 times, are also included in Fig. 5.10. It is easy to see that the CfCVDT-adapted meshes generated using the density function ρ_{H^1} have a much higher concentration of nodes near the singular point, i.e., the origin, than those generated using ρ_{L^2}. This observation is supported by the values of h_{max}/h_{min} listed in Table 5.3. Again, all triangles are well shaped at all refinement levels, an observation that is supported by the values of q_{min} and q_{avg} listed in Table 5.3.

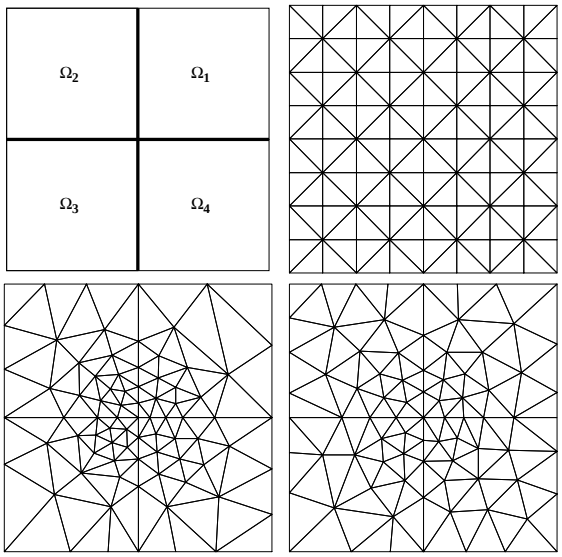

Fig. 5.9: The domain Ω and the initial meshes for Example 3; top-left: the domain Ω and its partitions; top-right: a uniform Cartesian grid with 81 nodes; bottom: the corresponding CfCVDT meshes generated using the density functions ρ_{H^1} (left) and ρ_{L^2} (right).

Table 5.3 contains information about mesh quality, solution errors, and convergence rates at all refinement levels for different refinement strategies for Example 3; the corresponding plots of the error norms ($\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$) vs. the number of

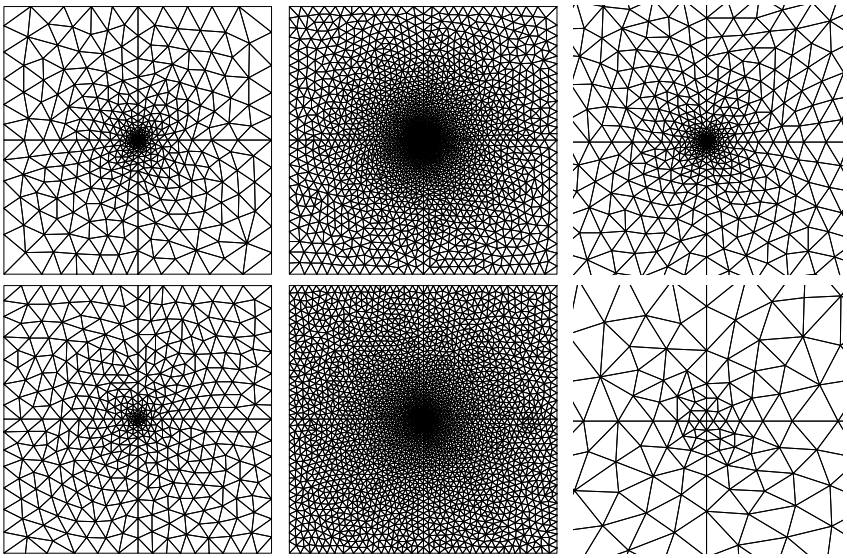

Fig. 5.10: Refined adaptive meshes at some levels generated by the CfCVDT-based adaptive method for Example 3; top, left to right: for the density function ρ_{H^1}, 476 and 2910 node meshes and the 2910 node case near the singular point magnified 1024 times; bottom, left to right: for the density function ρ_{L^2}, 436 and 2868 nodes and the 2868 node case near the singular point magnified 1024 times.

nodes are given in Fig. 5.11. The convergence rates for the uniform refinement method are about 1.08 and 0.53 for $\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$, respectively. These match very well with those predicted by finite element theory, i.e., 2α and α, respectively, for $u \in H^{\alpha-\epsilon}(\Omega)$). However, one sees that the CfCVDT-based adaptive methods still achieve almost perfect convergence rates, i.e., 1.0 and 1.8 for $|e_h|_{H^1(\Omega)}$ and $|e_h|_{L^2(\Omega)}$, respectively. The values of q_{min} and q_{avg} given in Table 5.3 demonstrate that the quality of the meshes produced by the CfCVDT-based adaptive methods is always very good at all refinement levels and for both density functions ρ_{H^1} and ρ_{L^2}, although the mesh sizes vary greatly over the Ω, e.g., h_{max}/h_{min} reaches 53393.1 at the last level when the density function ρ_{H^1} is used. Again, since u is not in $H^2(\Omega)$, we see that h_{max}/h_{min} tends to monotonically increase for the adaptive methods. Note that in this case, the CfCVDT-based adaptive method using the density function ρ_{H^1} generated approximate solutions having both smaller $\|e_h\|_{L^2(\Omega)}$ and $\|e_h\|_{H^1(\Omega)}$ than using ρ_{L^2}. We believe this phenomenon is due to the strong singularity of the exact solution u while the density functions are constructed based on the assumption that $u \in H^2(\Omega)$.

From Table 5.3 and Fig. 5.11, one also observes that the CfCVDT-based adaptive methods are much more efficient relative to the uniform refinement strategy. For example, for the uniform refinement method, we have that $\|e_h\|_{L^2(\Omega)}$ =1.4193e-03 and $|e_h|_{H^1(\Omega)}$ =1.0975e-01 on the refined mesh with $66,409$ nodes. However, for the adaptive methods, the values of these norms are 1.3335e-03 and 1.1379e-01, respectively, on the mesh with only 853 nodes when ρ_{H^1} is used as the density function, and 1.2532e-03 and 1.1830e-01, respectively, on the mesh with only $1,512$ nodes when ρ_{L^2} is used as the density function. This means that, in the case of the $|e_h|_{H^1(\Omega)}$ norm,

ℓ	n_ℓ	q_{min}	q_{avg}	h_{max}/h_{min}	$\|e_h\|_{L^2(\Omega)}$	CR	$\|e_h\|_{H^1(\Omega)}$	CR
Uniform refinement								
0	81	0.828	0.828	1.0	6.1349e-02		6.5596e-01	
1	289	0.828	0.828	1.0	2.9422e-02	1.06	4.7058e-01	0.48
2	1089	0.828	0.828	1.0	1.3654e-02	1.11	3.2931e-01	0.52
3	4225	0.828	0.828	1.0	6.3877e-03	1.09	2.2900e-01	0.52
4	16441	0.828	0.828	1.0	3.0046e-03	1.09	1.5869e-01	0.53
5	66409	0.828	0.828	1.0	1.4193e-03	1.08	1.0975e-01	0.53
Adaptive refinement using ρ_{H^1}								
0	81	0.489	0.896	5.4	2.8158e-02		5.1027e-01	
1	115	0.557	0.901	12.1	1.4766e-02	3.68	3.8016e-01	1.68
2	176	0.483	0.883	33.7	9.7944e-03	1.93	2.8124e-01	1.42
3	282	0.489	0.899	87.3	5.3705e-03	2.63	2.1415e-01	1.19
4	476	0.522	0.919	206.1	3.2539e-03	1.86	1.5543e-01	1.19
5	853	0.526	0.925	469.2	1.3335e-03	3.04	1.1379e-01	1.07
6	1561	0.455	0.933	1141.8	7.2023e-04	2.04	8.2305e-02	1.07
7	2910	0.447	0.939	3210.3	3.4939e-04	2.32	6.0055e-02	1.01
8	5517	0.484	0.941	6897.3	1.9200e-04	1.87	4.3156e-02	1.03
9	10518	0.487	0.942	13927.4	1.0982e-04	1.73	3.2185e-02	0.91
10	19935	0.521	0.943	30373.2	6.3599e-05	1.71	2.2651e-02	1.10
11	38024	0.596	0.944	53393.1	3.5049e-05	1.85	1.6798e-02	0.93
Adaptive refinement using ρ_{L^2}								
0	81	0.668	0.920	2.7	3.4814e-02		5.4208e-01	
1	139	0.523	0.918	3.7	1.8234e-02	2.40	4.1472e-01	0.99
2	241	0.689	0.929	7.1	8.3887e-03	2.82	2.9469e-01	1.24
3	436	0.529	0.932	11.5	4.3692e-03	2.20	2.1264e-01	1.10
4	808	0.518	0.936	15.8	3.4153e-03	0.80	1.7719e-01	0.60
5	1512	0.659	0.939	23.7	1.2532e-03	3.20	1.1830e-01	1.29
6	2868	0.640	0.944	33.2	7.1078e-04	1.77	8.7174e-02	0.95
7	5475	0.551	0.942	56.8	3.9287e-04	1.83	6.1441e-02	1.08
8	10491	0.619	0.944	90.7	2.1873e-04	1.80	4.6519e-02	0.86
9	20121	0.545	0.944	112.2	1.1865e-04	1.88	3.3943e-02	0.97

Table 5.3: Mesh quality, solution errors, and convergence rates for different refinement strategies for Example 3.

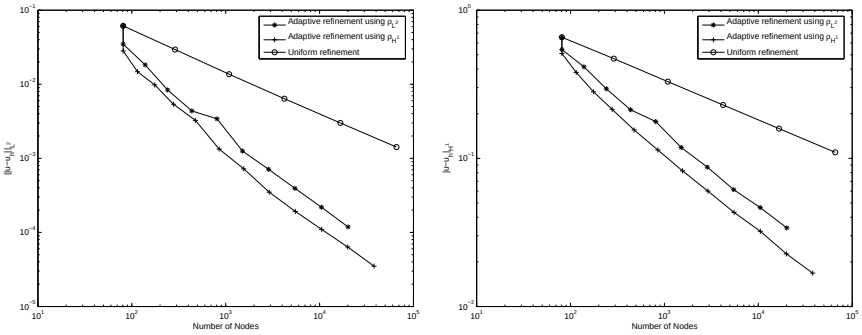

Fig. 5.11: Error norms vs. number of nodes for different refinement strategies for Example 3; left: $\|e_h\|_{L^2(\Omega)}$; right: $|e_h|_{H^1(\Omega)}$.

the CfCVDT-based adaptive methods are more than 60 times more efficient than the uniform refinement method if only considering the size of the resulting system.

We display a representative approximate solution u_h and the errors e_h for different

refinement methods in Fig. 5.12. It is again obvious that the CfCVDT-based adaptive methods distribute the errors much more equally over the triangles than does the uniform refinement method.

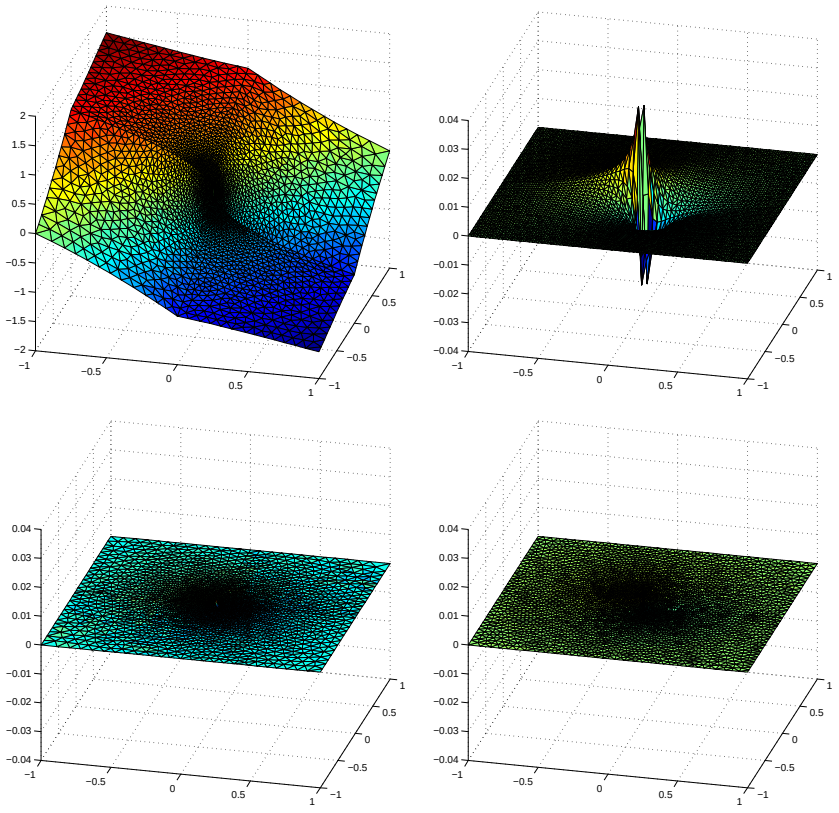

Fig. 5.12: Plots of an approximate solution and error distributions for different refinement methods for Example 3; top-left: a representative approximate solution u_h; top-right: the error e_h on the mesh with 4225 nodes obtained using the uniform refinement method; bottom-left: e_h on the mesh with 2910 nodes obtained using the CfCVDT-based adaptive method using ρ_{H^1}; bottom-right: e_h on the mesh with 2910 nodes obtained using the CfCVDT-based adaptive method using ρ_{L^2}.

5.4. More complicated geometries. Our last two examples involve more complicated geometries and serve to illustrate the robustness and effectiveness of our adaptive CfCVDT-based mesh generation algorithm.

EXAMPLE 4. *Let* $\Omega = \Omega_S - \Omega_{H^1} \cup \Omega_{H_2}$, *where* $\Omega_S = [0,1] \times [0,1]$ *and* Ω_{H^1} *and* Ω_{H_2} *are two open hexagons formed by the set of vertices* $\{0.25+0.1\cos(j-1)\theta,\ 0.75+0.1\sin(j-1)\theta)\}_{j=1}^6$ *and* $\{0.6+0.1\cos(j-1)\theta,\ 0.4+0.1\sin(j-1)\theta)\}_{j=1}^6$ *with* $\theta = \pi/3$, *respectively. Clearly,* Ω *is a square domain having two hexagonal holes; see Fig.* 5.13. *In general, solutions of* (5.1) *possess large geometric singularities at the vertices of the two interior hexagons. Set*

$$a(x,y) = 1 + 10x^2 + y^2,\ \ b(x,y) = 1 + x^2,\quad f(x,y) = 1,\ \ and\ \ g(x,y) = 0. \qquad (5.7)$$

Although an analytic form of the exact solution u of (5.1) *with the data* (5.7) *is not known, it is known that u only belongs to* $H^{\frac{7}{4}-\epsilon}(\Omega)$ *for any* $\epsilon > 0$ *and not to* $H^2(\Omega)$.

EXAMPLE 5. *Let* $\Omega = \Omega_{S_1} - \Omega_{S_2}$, *where* $\Omega_{S_1} = [0,1]\times[0,1]$ *and* $\Omega_{S_2} = [-0.5, 0.5]\times[-0.5, 0.5]$. *Set*

$$b(x,y) = 0, \quad f(x,y) = 1, \quad g(x,y) = 0 \tag{5.8}$$

and

$$a(x,y) = \begin{cases} 1 & (x,y) \in \Omega_1 \\ 20 & (x,y) \in \Omega_2 \\ 20 & (x,y) \in \Omega_3 \\ 400 & (x,y) \in \Omega_4, \end{cases} \tag{5.9}$$

where Ω_i *for* $i = 1,2,3,4$ *are defined in Fig.* 5.13. *Note that a is discontinuous in* Ω. *We again do not know an analytic form of the exact solution u of* (5.1) *with the data* (5.8) *and* (5.9), *but we do know that globally u only belongs to* $H^1(\Omega)$ *(and not* $H^2(\Omega)$*), but* $u|_{\Omega_i} \in H^2(\Omega_i)$ *for* $i = 1,2,3,4$.

The initial coarse meshes used for the solution of Examples 4 and 5 are displayed in Fig. 5.13. Fig. 5.14 and Fig. 5.15 present repeatedly refined meshes at different levels generated by the CfCVDT-based adaptive methods for Examples 4 and 5, respectively for both density functions ρ_{H^1} and ρ_{L^2}. Information about mesh quality is given in Tables 5.4 and 5.5, respectively. Fig. 5.16 displays approximate solutions u_h computed by our adaptive methods for the two examples.

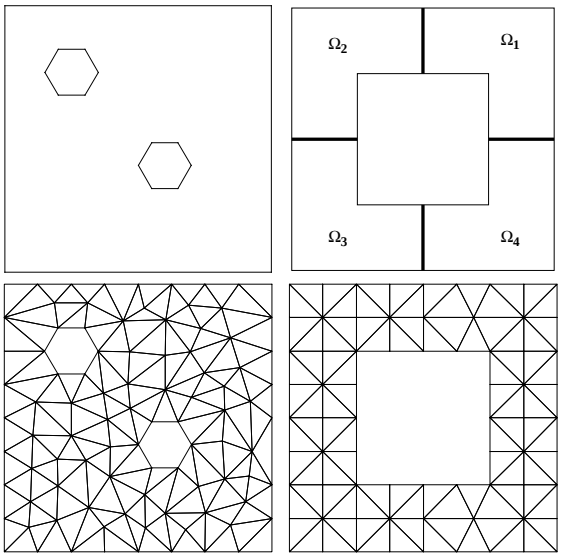

Fig. 5.13: Top: the domain Ω for Examples 4 (left) and 5 (right); bottom: initial meshes for Examples 4 (left) and 5 (right).

6. Conclusions. In this paper, we presented an efficient adaptive mesh refining algorithm for elliptic PDEs that combines a posteriori error estimation with centroidal Voronoi/Delaunay tessellations of domains in two dimensions. The two ingredients

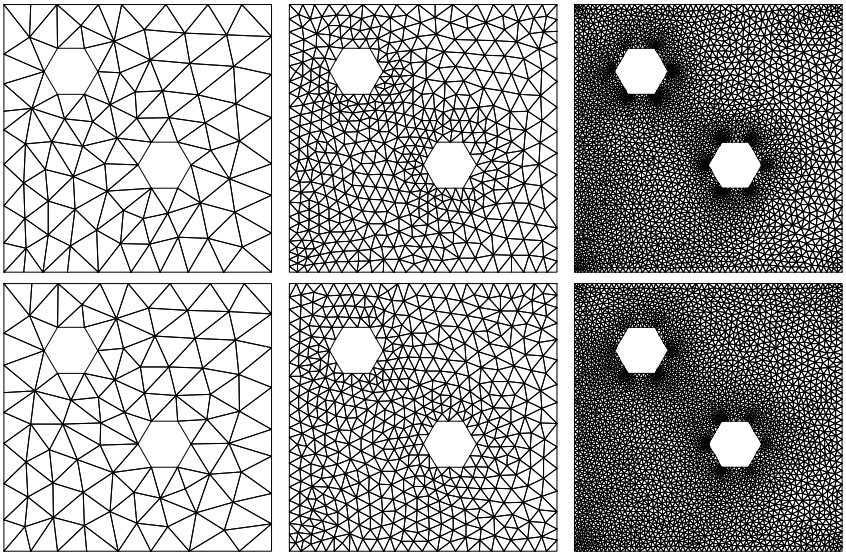

Fig. 5.14: Repeatedly refined meshes at different levels generated by the CfCVDT-based adaptive methods for Example 4; top: 94, 507, and 3096 nodes using the density function ρ_{H^1}; bottom: 94, 524, and 3373 nodes using the density function ρ_{L^2}.

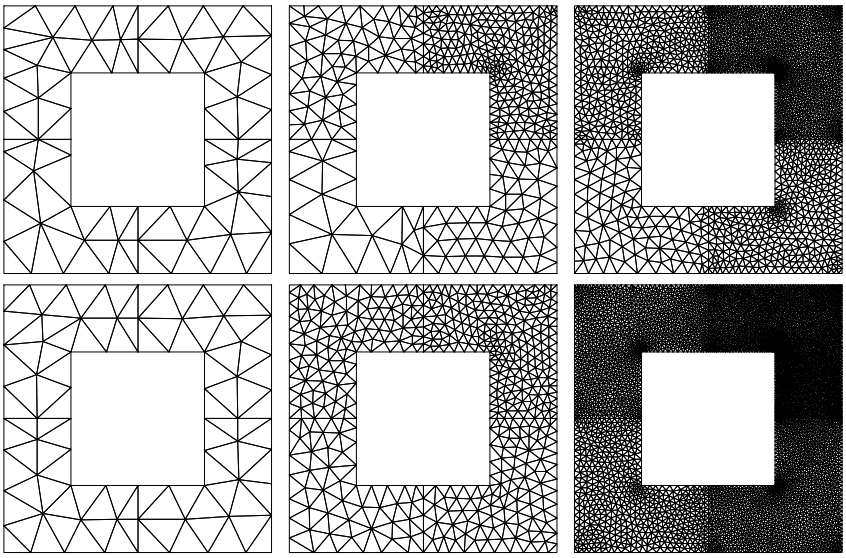

Fig. 5.15: Repeatedly refined meshes at different levels generated by the CfCVDT-based adaptive methods for Example 5. top: 70, 408, and 2474 nodes using the density function ρ_{H^1}; bottom: 70, 492, and 3020 nodes using the density function ρ_{L^2}.

are well linked together by the fact that the density function required by the second one is defined and computed from the first one with standard interpolations. Various numerical experiments were carried out and showed that our techniques always ob-

Uniform refinement						
ℓ	0	1	2	3	4	5
n_l	94	336	1261	4875	19159	75951
q_{min}	0.467	0.467	0.467	0.467	0.467	0.467
q_{avg}	0.862	0.862	0.862	0.862	0.862	0.862
h_{max}/h_{min}	2.4	2.4	2.4	2.4	2.4	2.4
Adaptive refinement using ρ_{H^1}						
ℓ	0	1	2	3	4	5
n_l	94	146	280	507	921	1680
q_{min}	0.721	0.727	0.614	0.715	0.636	0.649
q_{avg}	0.919	0.936	0.941	0.942	0.936	0.941
h_{max}/h_{min}	1.7	2.5	2.7	3.2	3.9	5.8
ℓ	6	7	8	9		
n_l	3096	5770	10838	20506		
q_{min}	0.538	0.556	0.567	0.529		
q_{avg}	0.940	0.941	0.941	0.945		
h_{max}/h_{min}	9.4	10.9	18.8	30.7		
Adaptive refinement using ρ_{L^2}						
ℓ	0	1	2	3	4	5
n_l	94	160	287	524	968	1797
q_{min}	0.721	0.661	0.628	0.655	0.663	0.620
q_{avg}	0.918	0.935	0.939	0.942	0.938	0.941
h_{max}/h_{min}	1.6	2.1	2.2	2.7	3.0	3.8
ℓ	6	7	8	9		
n_l	3373	6384	12168	23237		
q_{min}	0.625	0.465	0.608	0.571		
q_{avg}	0.942	0.944	0.943	0.944		
h_{max}/h_{min}	4.0	5.9	6.5	8.6		

Table 5.4: Mesh quality by different refinement methods for Example 4.

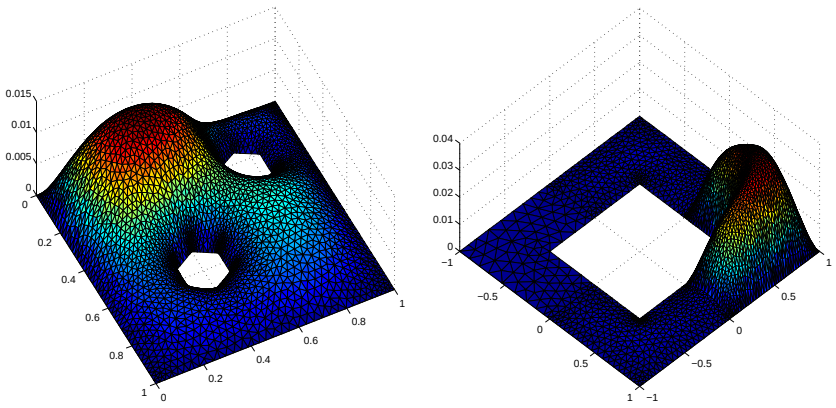

Fig. 5.16: Plots of the approximate solutions u_h obtained using the CfCVDT-based adaptive methods. Left: u_h for Example 4 on the mesh with 3096 nodes generated using ρ_{H^1}; right: u_h for Example 5 on the mesh with 2474 nodes generated using ρ_{H^1}.

tained optimal convergence rates for the piecewise linear finite elements with respect to both H^1 and L^2 norms and worked pretty robust. This mesh adaptation strategy can be easily generalized and applied to higher-order finite element approximations or mixed finite element formulations. We also would like to remark that we are currently

Uniform Refinement						
ℓ	0	1	2	3	4	5
n_l	70	232	832	3136	12160	47872
q_{min}	0.828	0.828	0.828	0.828	0.828	0.828
q_{avg}	0.845	0.845	0.845	0.845	0.845	0.845
h_{max}/h_{min}	1.3	1.3	1.3	1.3	1.3	1.3
Adaptive refinement using ρ_{H^1}						
ℓ	0	1	2	3	4	5
n_l	70	90	137	233	409	734
q_{min}	0.548	0.653	0.681	0.583	0.518	0.541
q_{avg}	0.903	0.917	0.931	0.930	0.929	0.38
h_{max}/h_{min}	1.5	2.4	3.9	5.8	9.6	15.1
ℓ	6	7	8	9	10	11
n_l	1344	2474	4608	8685	16519	31555
q_{min}	0.524	0.540	0.493	0.531	0.472	0.419
q_{avg}	0.935	0.939	0.942	0.943	0.944	0.942
h_{max}/h_{min}	19.9	33.6	86.4	170.0	288.1	381.9
Adaptive refinement using ρ_{L^2}						
ℓ	0	1	2	3	4	5
n_l	94	141	276	492	888	1625
q_{min}	0.747	0.720	0.697	0.690	0.534	0.612
q_{avg}	0.921	0.931	0.936	0.935	0.935	0.943
h_{max}/h_{min}	1.38	2.2	4.8	4.5	6.8	6.6
ℓ	6	7	8	9	10	
n_l	3020	5612	10477	19611	38354	
q_{min}	0.635	0.502	0.567	0.595	0.582	
q_{avg}	0.941	0.944	0.945	0.945	0.944	
h_{max}/h_{min}	9.1	11.9	13.9	15.5	25.2	

Table 5.5: Mesh quality by different refinement methods for Example 5.

studying the extension of this methodology to problems in three dimensions and to time dependent problems.

Acknowledgments. The authors would like to thank the referees for their valuable suggestions which improved the paper a lot.

REFERENCES

[1] S. Arya and D. Mount, Approximate nearest neighbor searching, *Proceedings of 4th Ann. ACM-SIAM Symposium on Discrete Algorithms (SODA'93)*, 1993, pp. 271-280.

[2] M. Ainsworth and J. Oden, *A Posteriori Error Estimation in Finite Element Analysis*, Wiley, 2002.

[3] R.E. Bank and R.K. Smith, Mesh smoothing using a posterior error estimates, *SIAM J. Numer. Anal.* **34**, 1997, pp. 979-997.

[4] I. Babuska and A. Miller, A feedback finite element method with a posteriori error estimation Part I, *Comput. Meth. Appl. Mech. Engrg* **61**, 1987, pp. 1-40.

[5] I. Babuska and W.C. Rheinboldt, Error estimates for adaptive finite element computations, *SIAM J. Numer. Anal.* **18**, 1978, pp. 736–754.

[6] I. Babuska and W.C. Rheinboldt, A posteriori error estimates for the finite element method, *Int. J. Numer. Meth. Engrg.* **12**, 1978, pp. 1597-1615.

[7] I. Babuska and W.C. Rheinboldt, A posteriori error analysis of finite element solutions for one dimensional problems, *SIAM J. Numer. Anal.* **18**, 1981, pp. 565-589.

[8] P. Binev, W. Dahmen, and R. DeVore, Adaptive finite element methods with convergence rates, *Numer. Math.* **97**, 2004, pp. 219-268.

[9] S. Brenner, Multigrid methods for the computation of singular solutions and stress intensity factors I: Corner singularities, *Math. Comp.* **68**, 2003, pp. 559-583.

[10] S. Brenner and L.-Y. Sung, Multigrid methods for the computation of singular solutions and stress intensity factors III: Interface singularities, *Comp. Meth. Appl. Mech. Engrg.* **192**, 2003, pp. 4687-4702.
[11] C. Carstensen, All first-order averaging techiniques for a posteriori finite element error control on unstructured grids are efficient and reliable, *Math. Comp.* **73**, 2003, pp. 1153-1165.
[12] L. Chen and J. Xu, Optimal Delaunay triangulation, *J. Comp. Math.* **22**, 2004, pp. 299-308.
[13] P.G. Ciarlet, *The Finite Element Method for Elliptic Problems*, North-Holland, 1978.
[14] W. Dörfler, A Convergent adaptive algorithm for Poisson's equation, *SIAM J Numer. Anal.* **33**, 1996, pp. 1106-1124.
[15] Q. Du, V. Faber, and M. Gunzburger, Centroidal Voronoi tessellations: Applications and algorithms, *SIAM Review* **41**, 1999, pp. 637-676.
[16] Q. Du and M. Gunzburger, Grid generation and optimization based on centroidal Voronoi tessellations, *Appl. Math. Comptut.* **133**, 2002, pp. 591-607.
[17] Q. Du, M. Gunzburger, and L. Ju, Constrained centroidal Voronoi tessellations on general surfaces, *SIAM J. Sci. Comput.* **24**, 2003, pp. 1488-1506.
[18] Q. Du, M. Gunzburger, and L. Ju, Voronoi-based finite volume methods, optimal Voronoi meshes and PDEs on the sphere, *Comp. Meth. Appl. Mech. Engrg.* **192**, 2003, pp. 3933-3957.
[19] Q. Du and D. Wang, Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations, *Int. J. Numer. Methods Engrg.* **56**, 2003, pp. 1355-1373.
[20] Q. Du and D. Wang, Anisotropic centroidal Voronoi tessellations and their applications, *SIAM. J. Sci. Comput.* **26**, 2005, pp. 737-761.
[21] D. Field, Laplacian smoothing and Delaunay triangulation, *Comm. Appl. Numer. Methods* **4**, 1988, pp. 709-712.
[22] D. Field, Quantitative measures for initial meshes, *Int. J. Numer. Methods Engrg.* **47**, 2000, pp. 887-906.
[23] P. Grisvard, *Elliptic Problems in Nonsmooth Domains*, Pitman, Boston, 1985.
[24] L. Ju, Q. Du, and M. Gunzburger, Probabilistic methods for centroidal Voronoi tessellations and their parallel implementations, *Parallel Comput.* **28**, 2002, pp. 1477-1500.
[25] P. Morin, R. Nochetto, and K. Siebert, Data oscillation and convergence of adaptive FEM, *SIAM J. Numer. Anal.* **38**, 2000, pp. 466-488.
[26] P. Morin, R. Nochetto, and K. Siebert, Convergence of adaptive finite element methods, *SIAM Review* **44**, 2002, pp. 631-658.
[27] P. Morin, R. Nochetto, and K. Siebert, Local problems on stars: A posteriori error estimators, convergence, and performance, *Math. Comp.* **72**, 2003, pp. 1067-1097.
[28] P.-O. Persson and G. Strang, A simple mesh generator in Matlab, *SIAM Review* **46**, 2004, pp.329-345.
[29] J. Shewchuk, Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator, *Lecture Notes in Comput. Sci.* **1148**, Springer, New York, 1996, pp. 203-222.
[30] J. Shewchuk, What is a good linear element? Interpolation, conditioning and quality measures, *Proceedings of the 11th International Meshing Roundtable*, Sandia National Laboratories, Albuquerque, 2002, pp. 115-126.
[31] R. Verfürth, *A Review of A Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*, Wiley-Teubner, Chichester, 1996.