Adaptive Refinement for Partial Differential Equations on Surfaces

Pravin Madhavan

Mathematics and Statistics Centre for Doctoral Training
University of Warwick

CSC Seminar
Center for Scientific Computing, 10th March 2014

Motivation - PDEs on Surfaces

Partial Differential Equations (PDEs) on surfaces arise in various areas, for instance

- materials science: enhanced species diffusion along grain boundaries,
- fluid dynamics: surface active agents,
- cell biology: phase separation on biomembranes, diffusion processes on plasma membranes, chemotaxis.

Neutrophil

Motivation - Adaptivity

In practical applications, often want to perform a simulation with *guaranteed* error bounds.

Classical error estimates can't be used for this purpose: let $\Omega \subset \mathbb{R}^2$, u the exact solution of some PDE and u_h its finite element approximation, then

$$\|u - u_h\|_{H^1(\Omega)} \leq Ch$$

where $\|u - u_h\|^2_{H^1(\Omega)} := \int_\Omega |u - u_h|^2 + |\nabla(u - u_h)|^2 \, \mathrm{d}x$. Here $v := v(x, y)$, $\nabla v := (\frac{\partial v}{\partial x}, \frac{\partial v}{\partial y})$.

Motivation - Adaptivity

In practical applications, often want to perform a simulation with *guaranteed* error bounds.

Classical error estimates can't be used for this purpose: let $\Omega \subset \mathbb{R}^2$, u the exact solution of some PDE and u_h its finite element approximation, then

$$\|u - u_h\|_{H^1(\Omega)} \leq Ch$$

where $\|u - u_h\|^2_{H^1(\Omega)} := \int_\Omega |u - u_h|^2 + |\nabla(u - u_h)|^2 \, \mathrm{d}x$. Here $v := v(x, y)$, $\nabla v := (\frac{\partial v}{\partial x}, \frac{\partial v}{\partial y})$.

- C typically depends on derivatives of the exact solution, which are generally not known.

Motivation - Adaptivity

In practical applications, often want to perform a simulation with *guaranteed* error bounds.

Classical error estimates can't be used for this purpose: let $\Omega \subset \mathbb{R}^2$, u the exact solution of some PDE and u_h its finite element approximation, then

$$\|u - u_h\|_{H^1(\Omega)} \leq Ch$$

where $\|u - u_h\|^2_{H^1(\Omega)} := \int_\Omega |u - u_h|^2 + |\nabla(u - u_h)|^2 \, \mathrm{d}x$. Here $v := v(x, y)$, $\nabla v := (\frac{\partial v}{\partial x}, \frac{\partial v}{\partial y})$.

- C typically depends on derivatives of the exact solution, which are generally not known.
- Even if C is known, error is usually severely overestimated.

Motivation - Adaptivity

In practical applications, often want to perform a simulation with *guaranteed* error bounds.

Classical error estimates can't be used for this purpose: let $\Omega \subset \mathbb{R}^2$, u the exact solution of some PDE and u_h its finite element approximation, then

$$\|u - u_h\|_{H^1(\Omega)} \leq Ch$$

where $\|u - u_h\|^2_{H^1(\Omega)} := \int_\Omega |u - u_h|^2 + |\nabla(u - u_h)|^2 \, dx$. Here $v := v(x, y)$, $\nabla v := (\frac{\partial v}{\partial x}, \frac{\partial v}{\partial y})$.

- C typically depends on derivatives of the exact solution, which are generally not known.
- Even if C is known, error is usually severely overestimated.
- No information on *where* errors are produced in domain and how they propagate.

Motivation - Adaptivity

Adaptive Grid Refinement: find optimal grid that reduces some quantity of interest below a certain user-defined tolerance with lowest computational cost.

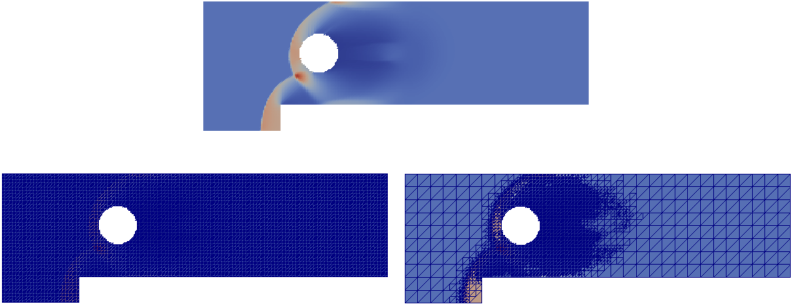
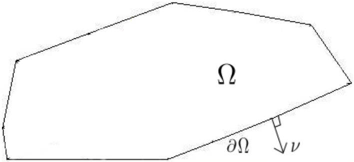


Figure: Uniform vs adaptive grid refinement for fluid flow with obstacle

Outline

Problem Formulation



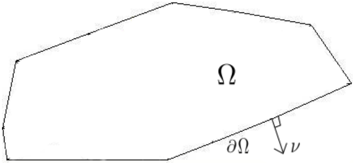
Problem: For a given function $f : \Omega \to \mathbb{R}$, find $u : \Omega \to \mathbb{R}$ such that

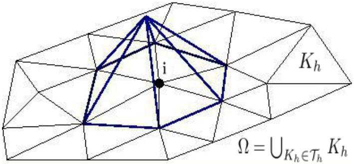
$$\begin{aligned} -\Delta u &= f \text{ in } \Omega \\ u &= 0 \text{ on } \partial\Omega. \end{aligned}$$

Here $\nabla \cdot \underline{w} = \frac{\partial w_1}{\partial x} + \frac{\partial w_2}{\partial y}$, $\Delta u := \nabla \cdot \nabla u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$.

Finite Element Approximation - Idea

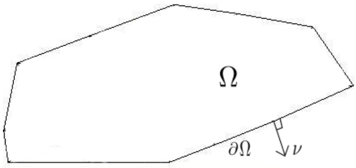
Triangulate the domain Ω.

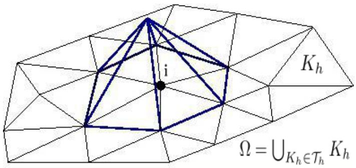




Finite Element Approximation - Idea

Triangulate the domain Ω.





Let N_h denote the number of nodes in $\mathcal{T}_h$ and $\{\phi_i^h\}_{i=1}^{N_h}$ denote a set of piecewise linear functions. The the finite element approximation u_h of u is given by

$$u_h = \sum_{i=1}^{N_h} \alpha_i \phi_i^h$$

A posteriori error estimation

Adaptive grid refinement is linked to a posteriori error estimation, which typically takes the form

$$J(u - u_h) \leq \sum_{K_h \in \mathcal{T}_h} \eta_{K_h}$$

where $J(u - u_h)$ is some quantity of interest depending on error $u - u_h$ and $\{\eta_{K_h}\}_{K_h \in \mathcal{T}_h}$ are respectively local estimators of $J(u - u_h)$.

A posteriori error estimation

Adaptive grid refinement is linked to a posteriori error estimation, which typically takes the form

$$J(u - u_h) \leq \sum_{K_h \in \mathcal{T}_h} \eta_{K_h}$$

where $J(u - u_h)$ is some quantity of interest depending on error $u - u_h$ and $\{\eta_{K_h}\}_{K_h \in \mathcal{T}_h}$ are respectively local estimators of $J(u - u_h)$.

Possible functionals $J(\cdot)$ are:

- *Energy norm*: $J(v) = \|v\|_{H^1}$.
- L^2 *norm*: $J(v) = \|v\|_{L^2}$.
- *Normal flux*: $J(v) = \int_{\partial\Omega} \nabla v \cdot \nu$.
- *Point error*: $J(v) = v(p)$ for some p in Ω.
- Some derived quantity, e.g., *lift*, *drag* or *pressure*.

One hopes that $\{\eta_{K_h}\}_{K_h \in \mathcal{T}_h}$ can be used to adaptively refine grid in such a way that $J(\cdot)$ is minimised in an optimal way.

A posteriori error estimation

$$J(u - u_h) \leq \sum_{K_h \in \mathcal{T}_h} \eta_{K_h}$$

A posteriori error estimation

$$J(u - u_h) \leq \sum_{K_h \in \mathcal{T}_h} \eta_{K_h}$$

For each $K_h \in \mathcal{T}_h$, local error indicator η_{K_h} is given by

$$\eta_{K_h} = h_{K_h} \mathcal{R}_{K_h}(u_h) + h_{K_h}^{1/2} \mathcal{R}_{\partial K_h}(u_h)$$

where
$\mathcal{R}_{K_h}(u_h) := \|f + \Delta u_h\|_{L^2(K_h)}$ is the element residual.
$R_{\partial K_h}(u_h) := \|[\nabla u_h]\|_{L^2(\partial K_h \setminus \partial\Omega)}$ is the jump residual.

A posteriori error estimation

$$J(u - u_h) \leq \sum_{K_h \in \mathcal{T}_h} \eta_{K_h}$$

For each $K_h \in \mathcal{T}_h$, local error indicator η_{K_h} is given by

$$\eta_{K_h} = h_{K_h} \mathcal{R}_{K_h}(u_h) + h_{K_h}^{1/2} \mathcal{R}_{\partial K_h}(u_h)$$

where
$\mathcal{R}_{K_h}(u_h) := \|f + \Delta u_h\|_{L^2(K_h)}$ is the element residual.
$R_{\partial K_h}(u_h) := \|[\nabla u_h]\|_{L^2(\partial K_h \setminus \partial\Omega)}$ is the jump residual.

$\eta_{K_h} =$ "*element* residual" $+$ "*jump* residual"

A posteriori error estimation

$$J(u - u_h) \leq \sum_{K_h \in \mathcal{T}_h} \eta_{K_h}$$

For each $K_h \in \mathcal{T}_h$, local error indicator η_{K_h} is given by

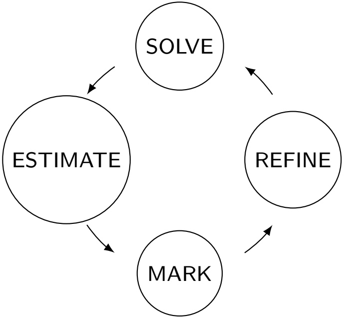
$$\eta_{K_h} = h_{K_h} \mathcal{R}_{K_h}(u_h) + h_{K_h}^{1/2} \mathcal{R}_{\partial K_h}(u_h)$$

where
$\mathcal{R}_{K_h}(u_h) := \|f + \Delta u_h\|_{L^2(K_h)}$ is the element residual.
$R_{\partial K_h}(u_h) := \|[\nabla u_h]\|_{L^2(\partial K_h \setminus \partial\Omega)}$ is the jump residual.

$\eta_{K_h} =$ "*element* residual" + "*jump* residual"

- The exact solution u does not appear in our local estimators!
- Local indicators η_{K_h} can be used to find regions in Ω where error is large and hence where smaller grid elements should be used!

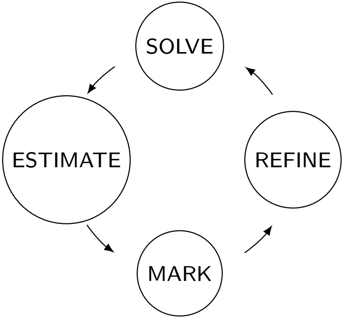
Adaptive Refinement Algorithm



Start with an intitial grid $\mathcal{T}_h^0$. Then for $n \geq 0$:

- SOLVE: compute a finite element approximation u_h of u.

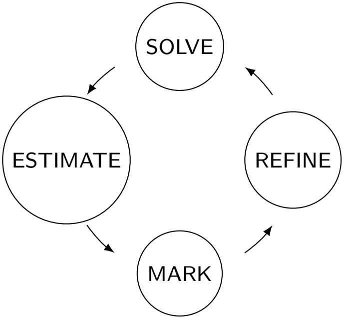
Adaptive Refinement Algorithm



Start with an intitial grid $\mathcal{T}_h^0$. Then for $n \geq 0$:

- SOLVE: compute a finite element approximation u_h of u.
- ESTIMATE: use u_h to compute local indicators $\{\eta_{K_h}\}_{K_h \in \mathcal{T}_h}$. If $\sum_{K_h \in \mathcal{T}_h} \eta_{K_h} < \mathrm{TOL}$, break.

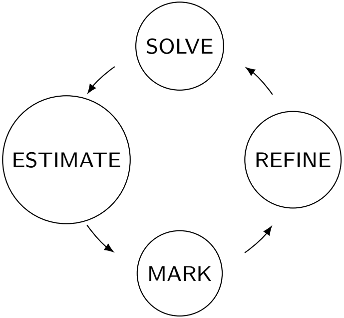
Adaptive Refinement Algorithm



Start with an intitial grid $\mathcal{T}_h^0$. Then for $n \geq 0$:

- SOLVE: compute a finite element approximation u_h of u.
- ESTIMATE: use u_h to compute local indicators $\{\eta_{K_h}\}_{K_h \in \mathcal{T}_h}$. If $\sum_{K_h \in \mathcal{T}_h} \eta_{K_h} < \mathrm{TOL}$, break.
- MARK: depending on value of local indicator η_{K_h}, mark corresponding element K_h for refinement or not.

Adaptive Refinement Algorithm



Start with an intitial grid $\mathcal{T}_h^0$. Then for $n \geq 0$:

- SOLVE: compute a finite element approximation u_h of u.
- ESTIMATE: use u_h to compute local indicators $\{\eta_{K_h}\}_{K_h \in \mathcal{T}_h}$. If $\sum_{K_h \in \mathcal{T}_h} \eta_{K_h} < \mathrm{TOL}$, break.
- MARK: depending on value of local indicator η_{K_h}, mark corresponding element K_h for refinement or not.
- REFINE: Refine marked elements $K_h \in \mathcal{T}_h^n$ to construct new grid $\mathcal{T}_h^{n+1}$.

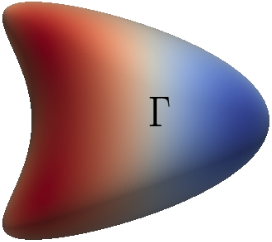
Outline

1. A Posteriori Error Analysis and Adaptive Refinement

2. Adaptive Refinement on Surfaces

3. Geometric Adaptive Refinement

Problem Formulation



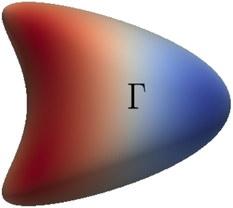
Problem: For a given function $f : \Gamma \to \mathbb{R}$, find $u : \Gamma \to \mathbb{R}$ such that

$$-\Delta_{\Gamma} u + u = f \text{ in } \Gamma$$

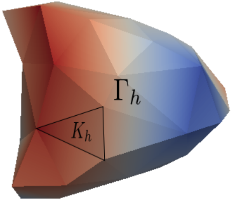
where Δ_{Γ} is the *Laplace-Beltrami* operator.

Triangulated Surfaces

- Γ is approximated by a polyhedral surface Γ_h composed of planar triangles K_h.
- The vertices sit on Γ $\quad\Rightarrow\quad$ Γ_h is its linear interpolation.
- Triangulate Γ_h as we have done for Ω in the flat case.



Γ



$\Gamma_h = \bigcup_{K_h \in \mathcal{T}_h} K_h$

A Posteriori Error Estimates on Surfaces

$$J(u - u_h) \leq \sum_{K_h \in \mathcal{T}_h} \eta_{K_h}$$

A Posteriori Error Estimates on Surfaces

$$J(u - u_h) \leq \sum_{K_h \in \mathcal{T}_h} \eta_{K_h}$$

For each $K_h \in \mathcal{T}_h$, local error indicator η_{K_h} is given by

$$\eta_{K_h} = h_{K_h} \mathcal{R}_{K_h}(u_h) + h_{K_h}^{1/2} \mathcal{R}_{\partial K_h}(u_h) + \mathcal{G}_{K_h}(u_h)$$

where
$\mathcal{R}_{K_h}(u_h) := \|f_h \delta_h + \Delta_{\Gamma_h} u_h - u_h \delta_h\|_{L^2(K_h)}$ is the element residual.
$\mathcal{R}_{\partial K_h}(u_h) := \|[\nabla_{\Gamma_h} u_h]\|_{L^2(\partial K_h)}$ is the jump residual.
$\mathcal{G}_{K_h}$ is the geometric residual encompassing the error caused by approximating smooth surface Γ.

A Posteriori Error Estimates on Surfaces

$$J(u - u_h) \leq \sum_{K_h \in \mathcal{T}_h} \eta_{K_h}$$

For each $K_h \in \mathcal{T}_h$, local error indicator η_{K_h} is given by

$$\eta_{K_h} = h_{K_h} \mathcal{R}_{K_h}(u_h) + h_{K_h}^{1/2} \mathcal{R}_{\partial K_h}(u_h) + \mathcal{G}_{K_h}(u_h)$$

where
$\mathcal{R}_{K_h}(u_h) := \|f_h \delta_h + \Delta_{\Gamma_h} u_h - u_h \delta_h\|_{L^2(K_h)}$ is the element residual.
$\mathcal{R}_{\partial K_h}(u_h) := \|[\nabla_{\Gamma_h} u_h]\|_{L^2(\partial K_h)}$ is the jump residual.
$\mathcal{G}_{K_h}$ is the geometric residual encompassing the error caused by approximating smooth surface Γ.

$\eta_{K_h} =$ "*element* residual" + "*jump* residual" + "*geometric* residual"

Outline

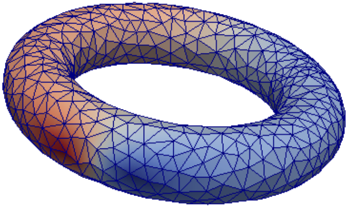
Software

Dune

Distributed and Unified Numerics Environment

- All simulations have been performed using the Distributed and Unified Numerics Environment (DUNE).
- Initial mesh generation made use of 3D surface mesh generation module of the Computational Geometry Algorithms Library (CGAL).
- Further information about DUNE and CGAL can be found respectively on http://www.dune-project.org/ and http://www.cgal.org/

Test Problem 1



Model problem:

$$-\Delta_\Gamma u + u = f$$

on the torus

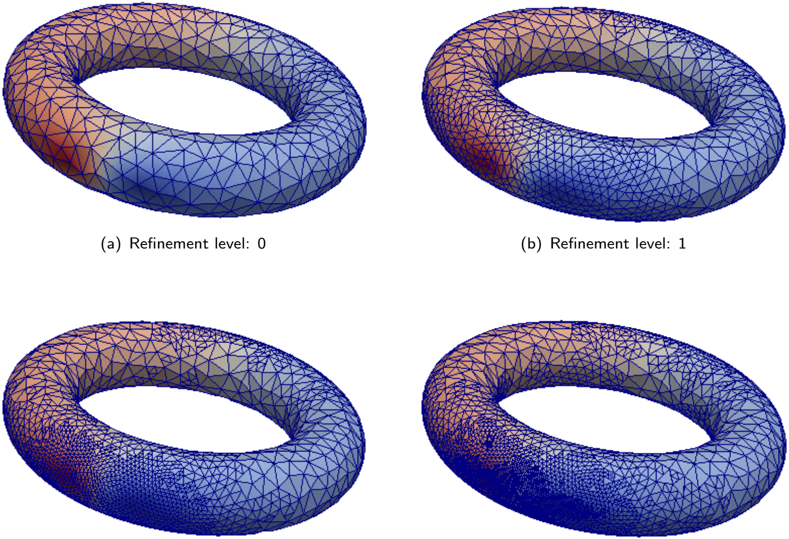
$$\Gamma = \{x \in \mathbb{R}^3 \ : \ x_3^2 + \left(1 - \sqrt{x_1^2 + x_2^2}\right)^2 - 0.0625 = 0\}.$$

The right-hand side f is chosen such that

$$u(x_1, x_2, x_3) = e^{\frac{1}{1.85 - x_1^2}} \sin(x_2).$$

is the exact solution.

Adaptive Refinement Algorithm for Test Problem 1



(a) Refinement level: 0

(b) Refinement level: 1

(c) Refinement level: 2

(d) Refinement level: 3

Test Problem 2

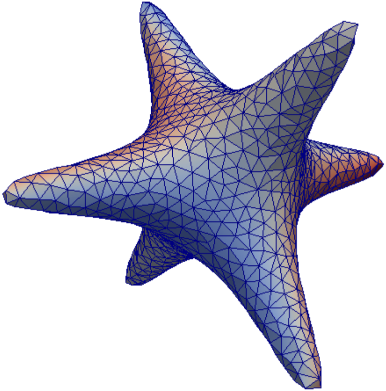
Model problem on the *Enzensberger-Stern surface*

$$\Gamma = \{x \in \mathbb{R}^3 \ : \ 400(x_1^2 x_2^2 + x_2^2 x_3^2 + x_1^2 x_3^2) - (1 - x_1^2 - x_2^2 - x_3^2)^3 - 40 = 0.\}$$

The right-hand side f is chosen such that

$$u(x) = x_1 x_2$$

is the exact solution.



Test Problem 2

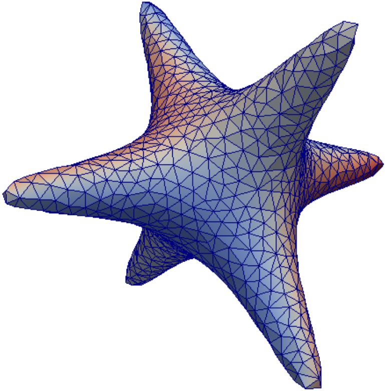
Model problem on the *Enzensberger-Stern surface*

$$\Gamma = \{x \in \mathbb{R}^3 \ : \ 400(x_1^2 x_2^2 + x_2^2 x_3^2 + x_1^2 x_3^2) - (1 - x_1^2 - x_2^2 - x_3^2)^3 - 40 = 0.\}$$

The right-hand side f is chosen such that

$$u(x) = x_1 x_2$$

is the exact solution.



- Notice that solution is *smooth* but initial mesh *poorly resolves* areas of high curvature.

Test Problem 2

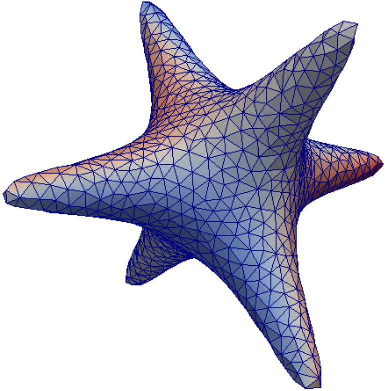
Model problem on the *Enzensberger-Stern surface*

$$\Gamma = \{x \in \mathbb{R}^3 \ : \ 400(x_1^2x_2^2 + x_2^2x_3^2 + x_1^2x_3^2) - (1 - x_1^2 - x_2^2 - x_3^2)^3 - 40 = 0.\}$$

The right-hand side f is chosen such that

$$u(x) = x_1x_2$$

is the exact solution.



- Notice that solution is *smooth* but initial mesh *poorly resolves* areas of high curvature.
- Geometric residual in estimator *very large* in those areas and drives adaptive grid refinement algorithm.

Test Problem 2

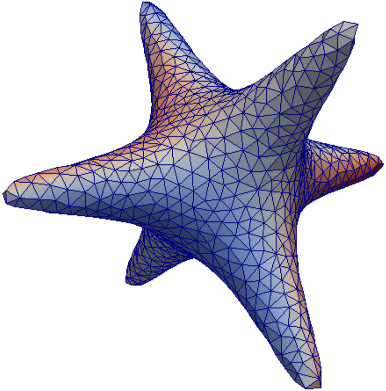
Model problem on the *Enzensberger-Stern surface*

$$\Gamma = \{x \in \mathbb{R}^3 \ : \ 400(x_1^2 x_2^2 + x_2^2 x_3^2 + x_1^2 x_3^2) - (1 - x_1^2 - x_2^2 - x_3^2)^3 - 40 = 0.\}$$

The right-hand side f is chosen such that

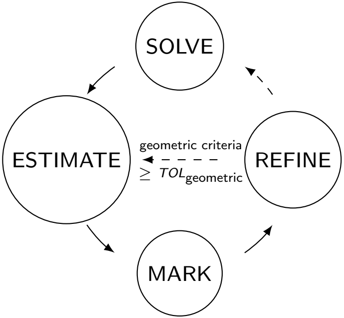
$$u(x) = x_1 x_2$$

is the exact solution.



- Notice that solution is *smooth* but initial mesh *poorly resolves* areas of high curvature.
- Geometric residual in estimator *very large* in those areas and drives adaptive grid refinement algorithm.
- Recomputation of u_h in adaptive grid refinement algorithm (SOLVE) is *costly* and does not significantly reduce overall residual when geometric residual dominates.

Geometric Adaptive Refinement Algorithm



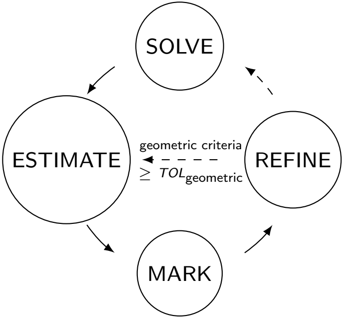
Start with an intitial grid $\mathcal{T}_h^0$.

- SOLVE: compute a finite element approximation u_h of u.

Then for $n \geq 0$:

- ESTIMATE: use u_h to compute local indicators $\{\eta_{K_h}\}_{K_h \in \mathcal{T}_h}$. If $\sum_{K_h \in \mathcal{T}_h} \eta_{K_h} < \text{TOL}$, break.
- MARK: depending on value of local indicator η_{K_h}, mark corresponding element K_h for refinement or not.
- REFINE: Refine marked elements $K_h \in \mathcal{T}_h^n$ to construct new grid $\mathcal{T}_h^{n+1}$.

Geometric Adaptive Refinement Algorithm



Start with an intitial grid $\mathcal{T}_h^0$.

- SOLVE: compute a finite element approximation u_h of u.

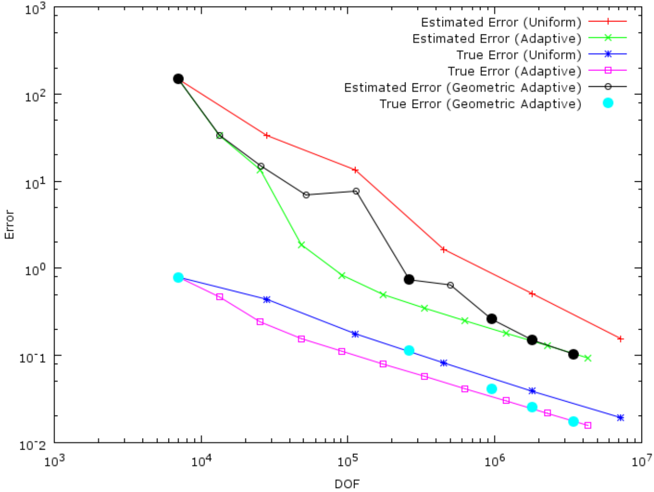
Then for $n \geq 0$:

- ESTIMATE: use u_h to compute local indicators $\{\eta_{K_h}\}_{K_h \in \mathcal{T}_h}$. If $\sum_{K_h \in \mathcal{T}_h} \eta_{K_h} < \text{TOL}$, break.
- MARK: depending on value of local indicator η_{K_h}, mark corresponding element K_h for refinement or not.
- REFINE: Refine marked elements $K_h \in \mathcal{T}_h^n$ to construct new grid $\mathcal{T}_h^{n+1}$.
- While

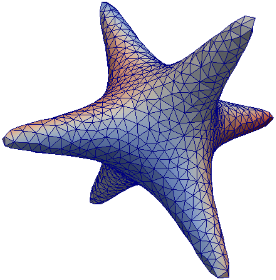
$$\frac{\sum_{K_h \in \mathcal{T}_h} \mathcal{G}_{K_h}}{\sum_{K_h \in \mathcal{T}_h} \eta_{K_h}} \geq TOL_{\text{geometric}}$$

go to ESTIMATE else SOLVE.

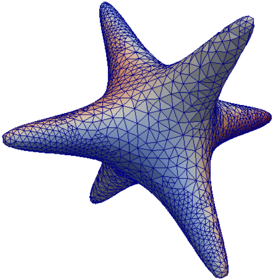
Geometric Adaptive Refinement Algorithm for Test Problem 2



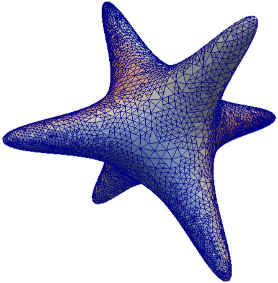
Geometric Adaptive Refinement Algorithm for Test Problem 2



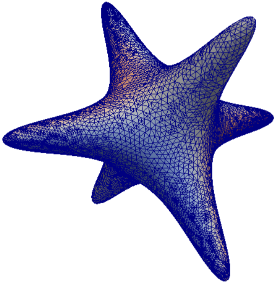
(e) Refinement level: 0



(f) Refinement level: 1



(g) Refinement level: 2



(h) Refinement level: 3

Thanks for your attention!

Acknowledgement: EPSRC Engineering and Physical Sciences Research Council grant EP/H023364/1 MASDOC Mathematics and Statistics Centre for Doctoral Training