A Posteriori Finite Element Bounds
with Adaptive Mesh Refinement: Application to Outputs of
the Three Dimensional Convection–Diffusion Equation

by

Hae-Won Choi

A thesis submitted in conformity with the requirements
for the degree of Master of Applied Science
Graduate Department of Mechanical and Industrial Engineering
University of Toronto

© Copyright by Hae-Won Choi, 2001.
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author’s permission.

0-612-63116-8
A Posteriori Finite Element Bounds
with Adaptive Mesh Refinement: Application to Outputs of
the Three Dimensional Convection-Diffusion Equation
by
Hae-Won Choi

A thesis submitted in conformity with the requirements
for the degree of Master of Applied Science
Graduate Department of Mechanical and Industrial Engineering
University of Toronto
2001

Abstract

Numerical simulations based on an a posteriori finite element bound method with adaptive mesh refinement is presented for linear-functional outputs of the convection-diffusion equation in three spaces dimensions. The bound method is based upon the construction of an augmented Lagrangian, in which the objective is an energy reformulation of the desired output, and the constraints are the finite element equilibrium equations and inter-subdomain continuity requirements. Outputs are functionals of the field solution obtained from partial differential equations. The bound method provides relevant, quantitative, inexpensive, and rigorous bounds to the output on a very fine discretization at a cost of the coarse mesh calculation. To achieve a derived accuracy of the bound gap at a reasonable cost, an adaptive mesh refinement technique is used to refine the mesh only where high resolution is needed. Optimal stabilization parameter $\kappa^*$ is also studied. This parameter improves the bounds, i.e. improves the sharpness of the bound gap. Finally the hybrid flux calculations on the coarse "working" mesh are compared with that on the fine "truth" mesh.
Acknowledgment

I would like to thank my supervisor Professor Marius Paraschivoiu for his help, guidance, continuous encouragement, and good advice.

I want to extend my thanks to Professor C. R. Ethier for allowing me to use AMR (Adaptive Mesh Refinement) code.

I want to acknowledge University of Toronto for the Connaught Scholarship and my supervisor for the Research Assistantship that financially supported my studies for my MASc program.

I want to give thanks to all the students of the Computational Fluid Dynamics Laboratory for providing me with a good academic and fun environment. In particular, I want to thank Zhong Cheng for helping me with Bounds method. Thanks to Simon Wong for his help with AMR code and Shahin Gomeishi for helping me Latex.

I would like to express special thanks to my lovely wife Hyun-Jin Yoon and our future baby for their grateful love, support and understanding. This thesis is dedicated to my wife and our future baby.
# Contents

1 Introduction ............................................. 1
   1.1 Motivation ........................................ 1
   1.2 Objectives ........................................ 2
   1.3 Literature Review ................................ 3
   1.4 Outline of the Thesis .............................. 6

2 Theoretical Background ................................... 7
   2.1 Sobolev Spaces ...................................... 7
   2.2 Governing Equation ................................ 9
       2.2.1 Convection-Diffusion Equation ............... 11
       2.2.2 Linear Functional Outputs .................... 13
   2.3 Galerkin Finite Element Method .................... 13
       2.3.1 Discretization ................................ 13
       2.3.2 Elemental Constructs ......................... 15
       2.3.3 The Elemental Matrix of Diffusive Term ...... 18
       2.3.4 The Elemental Matrix of Convective Term ..... 19
       2.3.5 The Elemental Vector of Force Term .......... 20
       2.3.6 Assembly ....................................... 21

3 A Posteriori Error Estimation ........................... 22
   3.1 Variational Formulation ............................ 22
       3.1.1 Governing Equation .......................... 22
       3.1.2 Elements ....................................... 23
       3.1.3 Finite Element Spaces ......................... 23
       3.1.4 Continuity Form ............................... 25
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.5</td>
<td>H-mesh and h-mesh Problems</td>
<td>26</td>
</tr>
<tr>
<td>3.1.6</td>
<td>Lagrangian Formulation</td>
<td>27</td>
</tr>
<tr>
<td>3.1.7</td>
<td>Duality</td>
<td>29</td>
</tr>
<tr>
<td>3.2</td>
<td>Bound Procedure</td>
<td>29</td>
</tr>
<tr>
<td>3.2.1</td>
<td>H-mesh Adjoint Calculation</td>
<td>29</td>
</tr>
<tr>
<td>3.2.2</td>
<td>H-mesh Hybrid Flux Calculation</td>
<td>30</td>
</tr>
<tr>
<td>3.2.3</td>
<td>h-mesh Local Neumann Problem</td>
<td>31</td>
</tr>
<tr>
<td>3.2.4</td>
<td>Bound Calculations</td>
<td>31</td>
</tr>
<tr>
<td>3.2.5</td>
<td>Bound Gap</td>
<td>32</td>
</tr>
<tr>
<td>3.3</td>
<td>Algebraic Formulation</td>
<td>33</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Adjoint Calculation</td>
<td>34</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Hybrid Flux Calculation</td>
<td>35</td>
</tr>
<tr>
<td>3.3.3</td>
<td>The FETI Procedure</td>
<td>35</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Fine &quot;truth&quot; mesh Local Neumann Problem</td>
<td>37</td>
</tr>
<tr>
<td>3.3.5</td>
<td>Bound Calculations</td>
<td>38</td>
</tr>
<tr>
<td>3.4</td>
<td>Numerical Validation</td>
<td>38</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Convergence Rate of the Average Solution Output s(1)</td>
<td>38</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Convergence Rate of Average Corner Solution Output s(2)</td>
<td>44</td>
</tr>
<tr>
<td>4</td>
<td>Adaptive Mesh Refinement</td>
<td>49</td>
</tr>
<tr>
<td>4.1</td>
<td>Local Indicators</td>
<td>49</td>
</tr>
<tr>
<td>4.2</td>
<td>Mesh Refinement</td>
<td>51</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Spatial Refinement</td>
<td>51</td>
</tr>
<tr>
<td>4.2.2</td>
<td>8-Subtetrahedron Subdivision Refinement</td>
<td>51</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Treatment of “hanging” Nodes</td>
<td>52</td>
</tr>
<tr>
<td>4.3</td>
<td>Adaptivity Procedure</td>
<td>54</td>
</tr>
<tr>
<td>4.4</td>
<td>Numerical Validation</td>
<td>54</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Uniformly Refined Meshes</td>
<td>55</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Adaptive Refinement of A Structured Initial Mesh</td>
<td>59</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Adaptive Refinement of An Unstructured Initial Mesh</td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td>Optimal Stabilization Parameter</td>
<td>70</td>
</tr>
<tr>
<td>5.1</td>
<td>Optimal Bound Formulation</td>
<td>70</td>
</tr>
</tbody>
</table>
C Analytical Solution of Rectangular Duct Flow
List of Figures

2.1 Unit Cube geometry .................................................. 12
2.2 Reference Element .................................................... 16

3.1 (a) Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ as a function of $H$ for the output $s^{(1)}$ when $\epsilon = 0.2$ (top), (b) Plots of $|((s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{pre} - s_h|$ as a function of $H$ in log-log scale for the output $s^{(1)}$ when $\epsilon = 0.2$ (bottom). ........................................ 41

3.2 (a) Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ as a function of $H$ for the output $s^{(1)}$ when $\epsilon = 0.1$ (top), (b) Plots of $|((s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{pre} - s_h|$ as a function of $H$ in log-log scale for the output $s^{(1)}$ when $\epsilon = 0.1$ (bottom). ........................................ 42

3.3 (a) Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ as a function of $H$ for the output $s^{(1)}$ when $\epsilon = 0.05$ (top), (b) Plots of $|((s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{pre} - s_h|$ as a function of $H$ in log-log scale for the output $s^{(1)}$ when $\epsilon = 0.05$ (bottom). ........................................ 43

3.4 (a) Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ as a function of $H$ for the output $s^{(2)}$ when $\epsilon = 0.2$ (top), (b) Plots of $|((s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{pre} - s_h|$ as a function of $H$ in log-log scale for the output $s^{(2)}$ when $\epsilon = 0.2$ (bottom). ........................................ 46

3.5 (a) Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ as a function of $H$ for the output $s^{(2)}$ when $\epsilon = 0.1$ (top), (b) Plots of $|((s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{pre} - s_h|$ as a function of $H$ in log-log scale for the output $s^{(2)}$ when $\epsilon = 0.1$ (bottom). ........................................ 47
3.6 (a) Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ as a function of $H$ for the output $s^{(2)}$ when $\varepsilon = 0.05$ (top), (b) Plots of $|(s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{pre} - s_h|$ as a function of $H$ in log-log scale for the output $s^{(2)}$ when $\varepsilon = 0.05$ (bottom).

4.1 $Sub_8$ Refinement of Tetrahedron $(t_{ij} = \frac{(t_i + t_j)}{2}, i < j)$

4.2 Sub-Division Patterns based on the 8-Subtetrahedron Subdivision Refinement Algorithm. (a) $Sub_2$, (b) $Sub_4$, (c) $Sub_8$, and (d) $Sub_8$.

4.3 Uniformly refined meshes: (a) $T_{(H,6)}$ has 1,296 elements, (b) $T_{(H,9)}$ has 4,374 elements, (c) $T_{(H,12)}$ has 10,368 elements, and (d) $T_{(H,15)}$ has 20,250 elements.

4.4 Isocontours (0 to 0.3 at intervals of 0.05) of temperature of subdivisions (slice at $z=0.8333$): (a) $T_{(H,6)}$ has 1,296 elements, (b) $T_{(H,9)}$ has 4,374 elements, (c) $T_{(H,12)}$ has 10,368 elements, (d) $T_{(H,15)}$ has 20,250 elements, and (e) $T_{(H,72)}$ has 2,239,488 elements.

4.5 Relative bound gap $\theta$ as a function of the number of elements in log-log scale using uniform and adaptive refinement of a structured initial mesh.

4.6 Adaptively refined meshes starting from a structured initial mesh: (a) initial mesh $T_H^0$ has 1,296 elements, (b) second refined mesh $T_H^2$ has 1,986 elements, (c) third refined mesh $T_H^3$ has 2,897 elements, (d) fourth refined mesh $T_H^4$ has 4,697 elements, (e) final mesh $T_H^5$, after 5 refinements, has 9,601 elements, and (f) uniformly refined mesh $T_H^{1/2}$ has 10,368 elements.

4.7 Isocontours (0 to 0.3 at intervals of 0.05) of temperature for adaptive refinement of a structured initial mesh (slice at $z=0.8333$): (a) first refined mesh $T_H^1$ has 1,673 elements, (b) second refined mesh $T_H^2$ has 1,986 elements, (c) third refined mesh $T_H^3$ has 2,897 elements, (d) fourth refined mesh $T_H^4$ has 4,697 elements, (e) final mesh $T_H^5$, after 5 refinements, has 9,601 elements, and (f) uniformly refined mesh $T_H^{1/2}$ has 10,368 elements.

4.8 (a) Relative bound gap $\theta$ as a function of the number of elements in log-log scale using uniform and adaptive refinement of an unstructured initial mesh (top), (b) Summary of relative bound gap $\theta$ as a function of the number of elements in log-log scale (bottom).
4.9 Adaptively refined meshes starting from an unstructured initial mesh: (a) initial mesh $\mathcal{T}_H^0$ has 1,440 elements, (b) first refined mesh $\mathcal{T}_H^1$ has 2,014 elements, (c) second refined mesh $\mathcal{T}_H^2$ has 2,639 elements, (d) third refined mesh $\mathcal{T}_H^3$ has 4,985 elements, (e) final mesh $\mathcal{T}_H^4$, after 4 refinements, has 9,739 elements and (f) uniformly refined mesh $\mathcal{T}_{H/2}^0$ has 11,520 elements.

4.10 Isocontours (0 to 0.3 at intervals of 0.05) of temperature for adaptive refinement of an unstructured initial mesh (slice at z=0.8333): (a) initial mesh $\mathcal{T}_H^0$ has 1,440 elements, (b) first refined mesh $\mathcal{T}_H^1$ has 2,014 elements, (c) second refined mesh $\mathcal{T}_H^2$ has 2,639 elements, (d) third refined mesh $\mathcal{T}_H^3$ has 4,985 elements, (e) final mesh $\mathcal{T}_H^4$, after 4 refinements, has 9,739 elements and (f) uniformly refined mesh $\mathcal{T}_{H/2}^0$ has 11,520 elements.

5.1 (a) Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ with optimal stabilization parameter $\kappa^*$ as a function of $H$ for the output $s^{(2)}$ when $\varepsilon = 0.1$ (top). (b) Plots of $|(s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{pre} - s_h|$ with optimal stabilization parameter $\kappa^*$ as a function of $H$ in log-log scale for the output $s^{(2)}$ when $\varepsilon = 0.1$ (bottom).

5.2 Relative bound gap $\theta$ with and without optimal stabilization parameter $\kappa^*$ as a function of the number of elements in log-log scale for uniformly refined meshes.

5.3 Relative bound gap $\theta$ with and without optimal stabilization parameter $\kappa^*$ as a function of the number of elements in log-log scale for uniform and adaptive refinement of a structured initial mesh.

5.4 Relative bound gap $\theta$ as a function of the number of elements in log-log scale: (a) Summary for optimal and non-optimal refinement (top) and (b) Summary for adaptive refinement of a structured initial mesh and of an unstructured initial mesh (bottom).

6.1 (a) Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ as a function of $H$ for the output $s^{(2)}$ when $\varepsilon = 0.1$ (top), (b) Plots of $|(s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{pre} - s_h|$ as a function of $H$ in log-log scale for the output $s^{(2)}$ when $\varepsilon = 0.1$ (bottom).
### 6.2 Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ with optimal stabilization parameter $\kappa^*$ as a function of $H$ for the output $s^{(2)}$ when $\epsilon = 0.1$ (top). (b) Plots of $|(s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{pre} - s_h|$ with optimal stabilization parameter $\kappa^*$ as a function of $H$ in log-log scale for the output $s^{(2)}$ when $\epsilon = 0.1$ (bottom). .................................................. 90

### 6.3 Relative bound gap $\theta$ as a function of the number of elements in log-log scale: (a) Summary for adaptive refinement of a structured initial mesh (top) and (b) Summary for adaptive refinement of an unstructured initial mesh (bottom). ........................................... 92

### 6.4 Optimal relative bound gap $\theta$ as a function of the number of elements in log-log scale: summary for adaptively refined meshes......................... 93

#### 7.1 Duct geometry ............................................. 95

#### 7.2 Velocity isocontours (0 to 1.3 at intervals of 0.1): (a) Velocity distribution as a surface (top), (b) Velocity distribution as isolines (bottom) ......................... 97

#### 7.3 Uniformly and adaptively refined meshes with optimal bound method: (a) $\mathcal{T}_H^0$ has 3,000 elements, (b) $\mathcal{T}_H^2$ has 5,023 elements, (c) $\mathcal{T}_H^3$ has 7,926 elements. (d) $\mathcal{T}_H^4$ has 10,308 elements, (e) $\mathcal{T}_H^5$ has 18,678 elements, and (f) $\mathcal{T}_{H/2}^0$ has 24,000 elements. .................................................. 101

#### 7.4 Isocontours (0 to 0.7 at intervals of 0.1) of temperature for uniformly and adaptively refined meshes with optimal bound method (slice at $z=2.0$): (a) $\mathcal{T}_H^0$ has 3,000 elements, (b) $\mathcal{T}_H^2$ has 5,023 elements, (c) $\mathcal{T}_H^3$ has 10,308 elements. (d) $\mathcal{T}_H^4$ has 18,678 elements, (e) $\mathcal{T}_{H/2}^0$ has 24,000 elements, and (f) $\mathcal{T}_{(H,36)}$ has 1.119.744 elements. .................................................. 102

#### 7.5 Relative bound gap $\theta$ as a function of the number of elements in log-log scale for uniform and adaptive refinement of a structured initial mesh using the coarse $H$-mesh hybrid flux calculations. ........................................... 103

#### 7.6 Uniformly refined meshes: (a) $\mathcal{T}_{(H,1)}$ has 3,000 elements, (b) $\mathcal{T}_{(H,6)}$ has 5,184 elements, (c) $\mathcal{T}_{(H,7)}$ has 8,232 elements, (d) $\mathcal{T}_{(H,8)}$ has 12,288 elements, and (e) $\mathcal{T}_{(H,9)}$ has 17,496 elements. .................................................. 107

#### 7.7 Adaptively refined meshes: (a) $\mathcal{T}_{H}^0$ has 3,000 elements, (b) $\mathcal{T}_{H}^1$ has 4,478 elements, (c) $\mathcal{T}_{H}^2$ has 8,734 elements, (d) $\mathcal{T}_{H}^3$ has 11,698 elements, and (e) $\mathcal{T}_{H}^4$ has 15,802 elements. .................................................. 108
7.8 Isocontours (0 to 0.7 at intervals of 0.1) of temperature for uniformly refined meshes with optimal bound method (slice at $z=2.0$): (a) $T_{(H,5)}$ has 3,000 elements, (b) $T_{(H,6)}$ has 5,184 elements, (c) $T_{(H,7)}$ has 8,232 elements, (d) $T_{(H,8)}$ has 12,288 elements, (e) $T_{(H,9)}$ has 17,496 elements, and (f) $T_{(H,36)}$ has 1,119,744 elements. 

7.9 Isocontours (0 to 0.7 at intervals of 0.1) of temperature for adaptively refined meshes with optimal bound method (slice at $z=2.0$): (a) $T^0_H$ has 3,000 elements, (b) $T^1_H$ has 4,478 elements, (c) $T^2_H$ has 8,734 elements, (d) $T^3_H$ has 11,698 elements, (e) $T^4_H$ has 15,802 elements, and (f) $T^5_H$ has 1,119,744 elements. 

7.10 Relative bound gap $\theta$ as a function of the number of elements in log-log scale for uniformly refined meshes and adaptively refined meshes using the fine $h$-mesh hybrid flux calculations. 

B.1 Adaptively refined meshes of a structured initial mesh using the non-optimal bound method: (a) $T^0_{H/2}$ has 1,296 elements, (b) $T^1_{H/2}$ has 1,660 elements, (c) $T^2_{H/2}$ has 2,211 elements, (d) $T^3_{H/2}$ has 3,399 elements, (e) $T^4_{H/2}$ has 8,084 elements, and (f) $T^5_{H/2}$ has 10,368 elements. 

B.2 Adaptively refined meshes of a structured initial mesh using the optimal bound method: (a) $T^0_H$ has 1,296 elements, (b) $T^1_H$ has 1,662 elements, (c) $T^2_H$ has 2,311 elements, (d) $T^3_H$ has 3,294 elements, (e) $T^4_H$ has 7,832 elements, and (f) $T^5_H$ has 10,368 elements. 

B.3 Relative bound gap $\theta$ with $h$-mesh hybrid flux calculations as a function of the number of elements in log-log scale using adaptive and uniform refinement: (a) without optimal stabilization parameter $\kappa^*$ (top), (b) with optimal stabilization parameter $\kappa^*$ (bottom). 

B.4 Adaptively refined mesh of an unstructured initial mesh using the non-optimal bound method: (a) $T^0_H$ has 1,440 elements, (b) $T^2_H$ has 2,010 elements, (c) $T^3_H$ has 3,121 elements, (d) $T^4_H$ has 5,620 elements, (e) $T^5_H$ has 9,624 elements, and (f) $T^6_H$ has 11,520 elements.
B.5 Adaptively refined meshes of an unstructured initial mesh using the optimal bound method: (a) $\mathcal{T}_H^0$ has 1,440 elements, (b) $\mathcal{T}_H^1$ has 1,851 elements, (c) $\mathcal{T}_H^2$ has 2,356 elements, (d) $\mathcal{T}_H^3$ has 3,679 elements, (e) $\mathcal{T}_H^4$ has 8,971 elements, and (f) $\mathcal{T}_{H/2}^0$ has 11,520 elements.

B.6 Relative bound gap $\theta$ with $h$-mesh hybrid flux calculations as a function of the number of elements in log-log scale using adaptive refinement of an unstructured initial mesh: (a) without optimal stabilization parameter $\kappa^*$ (top), (b) with optimal stabilization parameter $\kappa^*$ (bottom).

C.1 Rectangular Duct
List of Tables

3.1 Number of subdomains and refinement subdivisions for the output $s^{(1)}$ .............................. 39
3.2 Output predictor $(s_h)_{pre}$ and half bound gap $\Delta(T_H)$ for the output $s^{(1)}$ .................... 39
3.3 Convergence slope of bounds values for the output $s^{(1)}$ ....................................................... 39
3.4 Number of subdomains and refinement subdivisions for the output $s^{(2)}$ ............................... 44
3.5 Output predictor $(s_h)_{pre}$ and half bound gap $\Delta(T_H)$ for the output $s^{(2)}$ .................... 44
3.6 Convergence slope of bounds values for the output $s^{(2)}$ ....................................................... 44

4.1 Bounds for subdivisions. .............................................................................................................. 55
4.2 Number of elements (%) by half bound gap $\Delta(T_H)$ intervals for subdivisions .................. 56
4.3 Memory requirement (M) and elapsed CPU time (seconds) for computation of subdivisions .......................... 56
4.4 Bounds for adaptive refinement of a structured initial mesh .................................................. 59
4.5 Memory requirement (M) and elapsed CPU time (seconds) for computation of adaptive refinement of a structured initial mesh .............................................................. 60
4.6 Number of elements (%) by half bound gap $\Delta(T_H)$ intervals for adaptive refinement of a structured initial mesh .............................................................. 61
4.7 Bounds for adaptive refinement of an unstructured initial mesh .............................................. 64
4.8 Memory requirement (M) and elapsed CPU time (seconds) for computation of adaptive refinement of an unstructured initial mesh .............................................................. 65
4.9 Number of elements (%) by half bound gap $\Delta(T_H)$ intervals for adaptive refinement of an unstructured initial mesh .............................................................. 66

5.1 Optimal output predictor $(s_h)_{pre}$, half bound gap $\Delta(T_H)$, and relative (half) bound gap $\theta$ for each subdivision with different $R$ .......................................................... 76
5.2 Optimal bounds for subdivisions. .................................................................................................. 76
5.3 Optimal bounds for adaptive refinement of a structured initial mesh ....................................... 78
5.4 Optimal bounds for adaptive refinement of an unstructured initial mesh. ... 80

6.1 Output predictor \((s_h)_{pre}\), half bound gap \(\Delta(T_H)\), and relative (half) bound gap \(\theta\) for a sequence of refinements. .................. 87

6.2 Output predictor \((s_h)_{pre}\), half bound gap \(\Delta(T_H)\), and relative (half) bound gap \(\theta\) with the optimal stabilization parameter \(\kappa^*\) for a sequence of refinements. ... 87

7.1 Optimal bounds for uniform and adaptive refinement of a structured initial mesh. 99

7.2 Number of elements (%) by half bound gap \(\Delta(T_H)\) intervals for uniform and adaptive refinement of a structured initial mesh. .......................... 99

7.3 Optimal bounds for uniformly refined meshes. .......................... 104

7.4 Optimal bounds for adaptive refinement of a structured initial mesh. ...... 104

7.5 Number of elements (%) by half bound gap \(\Delta(T_H)\) intervals for uniformly refined meshes. .................................................. 105

7.6 Number of elements (%) by half bound gap \(\Delta(T_H)\) for adaptively refined meshes. 105

B.1 Non-optimal bounds with \(h\)-mesh hybrid flux calculations for adaptive refinement of a structured initial mesh. .......................... 123

B.2 Optimal bounds with \(h\)-mesh hybrid flux calculations for adaptive refinement of a structured initial mesh. .......................... 123

B.3 Number of element (%) by half bound gap \(\Delta(T_H)\) intervals for the non-optimal bound method. .......................... 125

B.4 Number of element (%) by half bound gap \(\Delta(T_H)\) intervals for the optimal bound method. .................................................. 125

B.5 Non-optimal bounds with \(h\)-mesh hybrid flux calculations for adaptive refinement of an unstructured initial mesh. .......................... 129

B.6 Optimal bounds with \(h\)-mesh hybrid flux calculations for adaptive refinement of an unstructured initial mesh. .......................... 129

B.7 Number of element (%) by half bound gap \(\Delta(T_H)\) intervals using the non-optimal bound method for adaptive refinement of an unstructured initial mesh. 131

B.8 Number of element (%) by half bound gap \(\Delta(T_H)\) intervals using the optimal bound method for adaptive refinement of an unstructured initial mesh. .... 131
Chapter 1

Introduction

1.1 Motivation

Numerical simulations can address a large range of engineering and natural science applications in areas such as fluid dynamics, heat transfer, aerodynamics, solid mechanics, meteorology, and geophysics. The obvious limitations of analytical solutions for complicated problems force researchers to use numerical or experimental tools. For a large number of applications, numerical simulations provide a fast, inexpensive and flexible alternative to experiments.

As a consequence, numerical simulations are an important technology for engineering analysis and design. Numerical methods are engaged in these two different categories: analysis and design. In analysis the entire field such as velocity, temperature, or displacement is relevant. For design and optimization, only a few parameters are important. A design problem exploits an input-output relationship between design variables and performance requirements. The performance can be an output or a combination of outputs which are often functionals of field solutions obtained from ordinary or partial differential equations. These functionals can either be linear or nonlinear. In context of design and optimization, the final goal of the numerical simulation is not the field solution, but rather to minimize some quantitative design output value.

The main parameter of an approximation method is the discretization size \( \delta \). For a given approximation domain, the accuracy of the approximation of the solution is closely related to the discretization size \( \delta \). In this study the finite element method is used. As the number of finite elements is increased (i.e. the discretization size \( \delta \) is decreased), not only the accuracy of numerical approximation is improved but also the requirement of CPU time is increased (which
means the computational costs are increased). Therefore an engineer faces a trade-off between computational cost and numerical accuracy. A coarse "working" mesh ($\delta = H$) approximation is relatively inexpensive but generates a solution which is not sufficiently accurate. A fine "truth" mesh ($\delta = h$) approximation is sufficiently accurate (i.e. $\Theta_h(x) \approx \Theta(x)$) but is very expensive, possibly unfeasible, in terms of computational costs.

There are several approaches that have been proposed to reduce the computational cost while keeping an acceptable accuracy. One approach consists of adapting the mesh to provide the most uniform distribution of the numerical error. The key idea of adaptive mesh refinement techniques is to define a certain norm of the error, which is constructed by adding up the contributions from each element. Adaptive mesh refinement methods accomplish computational cost reduction by varying the mesh size inside the computational domain. In this work, adaptive techniques are exploited in the context of the bound method to construct the ideal mesh that provides the derived bound gap.

1.2 Objectives

The technique adopted in this thesis does not aim at finding the field solution, but rather seeks to calculate bounds for an output of interest which is derived from the solution of a PDE (Partial Differential Equation). The goal is to improve using adaptive refinement a fast method to evaluate or approximate the fine "truth" mesh output (i.e. $s_h$) and thereby provide the "truth" validation. The idea behind the bound method is to replace the fine "truth" mesh output ($s_h$) by relevant, quantitative, inexpensive, and rigorous bounds to $s_h$ (i.e. lower and upper bounds). The bound method offers precise and reliable information at a cost of the coarse "working" mesh calculations.

More precisely, the bound method produces quantitative lower and upper bounds, i.e. $(s_h)_{UB}$ and $(s_h)_{LB}$, for $s_h = \ell^O(\Theta_h)$ which is the finite element approximation to the "exact" output $s$ on the fine "truth" mesh

$$(s_h)_{LB} \leq s_h \leq (s_h)_{UB}. \quad (1.1)$$

Inexpensively, the bounds (1.1) can also provide an estimate of $s_h$,

$$(s_h)_{pre} = \frac{1}{2} [(s_h)_{UB} + (s_h)_{LB}], \quad (1.2)$$
which will satisfy
\[ |s_h - s_{\text{pre}}| \leq \Delta, \]  \hfill (1.3)
where
\[ \Delta = \frac{1}{2} [(s_h)_{UB} - (s_h)_{LB}], \]  \hfill (1.4)
which we shall denote the (half) bound gap.

In design problems reducing the bound gap is important because an engineer is interested in bounds with a specific bound gap. The sharper bounds provide more reliable and accurate information about the "truth" output by adapting the mesh.

The objective of this thesis is to develop an error estimator for three-dimensional problems, to construct a methodology for adaptive mesh refinement on a tetrahedron, and to implement the adaptive technique in a computer code. In particular, the objectives of this thesis are as follows:

1. To extend the bound method with adaptive mesh refinement to three-dimensional spaces using a domain decomposition method with modified Lagrange multipliers and an iterative interface solution scheme—i.e. the finite element tearing and interconnecting (FETI) procedure.

2. To demonstrate the accuracy of the local error indicators which represent the sum of positive contribution of the bound gap on local elements \( T_H \) of \( T_H \).

3. To develop, based on these local bound gap error indicators, an adaptive mesh refinement methodology to reduce the bound gap.

4. To analyze the effect of the Optimal Stabilization Parameter on the bound gap.

5. To compare the hybrid flux calculations on the coarse mesh with that on the fine mesh.

6. To address a realistic problem: a convection–diffusion problem in a rectangular duct with a given velocity field.

1.3 Literature Review

The field of \textit{a posteriori} error estimation and adaptive mesh refinement is a major area in finite element analysis. The two goals are: first, inexpensive confirmation of the accuracy of
CHAPTER 1. INTRODUCTION

a particular finite element solution, and second, efficient improvement of the finite element solution by optimal adaptive mesh refinement.

The numerical error is defined by \( e = \Theta(x) - \Theta_H(x) \) (where \( \Theta(x) \) is an exact solution and \( \Theta_H(x) \) is a finite element solution discretized by \( H \)); the (pseudo)metric in which an engineer wish to measure the finite element error is denoted by \( E(e) \). An a posteriori procedure provides an estimate, \( E(T_H) \), which is typically expressed as the sum of positive contributions \( E_{T_H} \) associated within each element \( T_H \) of the \( T_H \). The elemental contributions \( E_{T_H} \) are interpreted as local indicators for the purpose of subsequent mesh refinement strategies.

The general approaches to a posteriori error estimation may be categorized as “explicit” or “implicit” methods [3]. Explicit methods have the advantage of computational efficiency, but have the drawback of the presence of constants that cannot be precisely evaluated. The implicit methods are based on the solution of residual in each element (i.e. local independent subproblems) [2, 4, 11]: the advantage is a more precise quantification of the error and the drawback is an increased complexity and computational effort. Both explicit and implicit methods typically offer error bounds in terms of constant \( C_E \) independent of \( T_H \) and \( \Theta(x) \) such that \( E(e) \leq C_E E_{T_H} \) [4], nevertheless the constant \( C_E \) is typically not known. Hence, the term “bound” for estimators for which \( C_E \) can be calculated shall be reserved.

Most early work on a posteriori error estimation focused on symmetric problems and on the natural energy measure of the error. For such cases, very effective explicit and implicit techniques [4] have been developed that require essentially no regularity assumptions and contain only minimal unknown approximation contributions to \( C_E \). Furthermore implicit procedures can be developed that offer rigorous bounds for the error; the unknown contributions to \( C_E \) are reduced to the inaccuracies acquired in the solution of the \( T_H \) local subproblems [2, 11]. Many of these explicit and implicit methods can be readily generalized to non-symmetric problems and more general error metrics [27].

Recently, a posteriori error estimation techniques has been greatly extended to error metrics more closely relevant to engineering design. It is assumed that the quantity of interest in engineering design is not the field variable or the error in the energy norm, but rather the output of the system performance which reflects specific goals and objectives of the design and optimization problem.
CHAPTER 1. **INTRODUCTION**

To reliably incorporate the numerical results into the engineering design and optimization process, only information of related to the output, $s_h$, is required to this end. An implicit procedure for the prediction of lower and upper bounds for outputs of coercive partial differential equations has been developed for one or two-dimensional space [17, 14, 19, 15, 20, 26] and for three-dimensional space [18]. The method is based upon the construction of an augmented Lagrangian, in which the objective is a quadratic energy reformulation of the desired output, and the constraints are the finite element equilibrium equations and inter-subdomain continuity requirements. The lower and upper bounds are then derived by evoking the dual maximum–minimum problem for appropriately chosen candidate Lagrange multipliers. The bound calculation is composed of two main steps: first, several global computations on the coarse “working” mesh ($T_H$); second, local computations on the fine “truth” mesh ($T_h$).

The bound method is application of the quadratic-linear duality theory proposed in [2, 11] to an augmented Lagrangian. Ladeveze procedure used to approximate the inter-subdomain connectivity in earlier work is limited to two-dimensional space [2, 11]. Herein the finite element tearing and interconnecting (FETI) procedure which is independent of dimensionality is implemented to extend the method to three-dimensional problems. This domain decomposition method is ideal to approximate the inter-subdomain connectivity (i.e. hybrid flux) in the bound method. The FETI procedure was first introduced for the structural problems and is well established in [6, 7, 8]. The FETI procedure for three-dimensional convection-diffusion equation is adopted as proposed in [18].

The Lagrangian in my approach permit the treatment of much more general error measures, such as linear-functional outputs, and the consideration of more general equations, in particular non-symmetric operators [17, 14, 19, 18]. The bound method has been developed to address a wide range of problems including the Helmholtz and Burgers equations [22] and Stokes equations [17, 15, 20]. It has been extended to the incompressible Navier-Stokes equations [10]. Furthermore, initial work has presented sensitivity derivatives to carry out design optimization problems [12].

A complete description of the bound method is presented in [17, 14, 19, 15, 20, 18]. However, it follows an *a posteriori* error estimation procedures with a modified energy objective proposed in [21]. The adaptive mesh refinement technique based on local bound-gap error
indicators is applied as suggested in [21, 10]. Previous local bound-gap error indicators were limited to two-dimensional space [21, 10]. In this thesis, the bound method—local bound gap error indicator is extended to the convection–diffusion problems in three-dimensional space. It is focused less on error estimation, and more on output prediction—i.e. it adopts the adaptive mesh refinement technique mainly as a tools of bounds improvement.

1.4 Outline of the Thesis

The outline of the remainder of this thesis is as follows. In Chapter 2 some of the major mathematical and theoretical ingredients required for this work are reviewed. In particular, the Galerkin finite element method is applied to the three-dimensional convection–diffusion equation. In Chapter 3, an a posteriori error estimation is introduced and extended to the bound procedure for the convection–diffusion equation. The FETI procedure for the bound method is also demonstrated. Convergence test results of the bound method are also presented. In Chapter 4 an adaptive mesh refinement technique is introduced. First, local error indicators which can be directly interpreted as the contribution to the bound gap from each element $T_H$ are presented. Then, some of materials for the mesh refinement technique are discussed. Finally, the adaptive procedure is summarized. In Chapter 5 the Optimal Stabilization Parameter is introduced to reduce the bound gap that is to maximize the lower bound and to minimize the upper bound. In Chapter 6 the new calculations for inter-subdomain connectivity which are the hybrid flux computations on fine $h$-mesh are presented. Then comparison of bound results between the hybrid flux calculations on the coarse $H$-mesh and the hybrid flux calculations on the fine $h$-mesh are reported. In Chapter 7 a realistic application is presented. Numerical results for this problem are presented. To conclude, in Chapter 8 some of key features of the bound method are discussed and future extensions are suggested.
Chapter 2

Theoretical Background

2.1 Sobolev Spaces

Sobolev spaces are briefly introduced in this section. The basic spaces, corresponding norms, and some of the theorems are presented. For a more complete description refer to [1, 17].

To define the Lebesque space, let $\Omega$ be an open subset of $\mathbb{R}^d$ with boundary $\Gamma$ where $d$ is the dimension of the space of interest. Denoting Lebesque space as $L^2(\Omega)$

$$L^2(\Omega) = \{ f | \int_{\Omega} f^2 \, d\Omega < +\infty \},$$

(2.1)

which is a space of square integrable functions on $\Omega$. Associated with this space, the inner scalar product is defined as

$$(p, q)_{L^2(\Omega)} = \int_{\Omega} pq \, d\Omega,$$

(2.2)

and the following norm is,

$$||u||_{L^2(\Omega)} = \left( \int_{\Omega} u^2 \, d\Omega \right)^{\frac{1}{2}}.$$  

(2.3)

The first order Sobolev space is called the Hilbert space $\mathcal{H}^1(\Omega)$ which is the space such that both the derivatives and the functions are in $L^2(\Omega)$ on $\Omega$.

$$\mathcal{H}^1(\Omega) = \{ v | v \in L^2(\Omega), D^d v \in (L^2(\Omega))^d \},$$

(2.4)

where

$$D^d = \frac{\partial |d|}{\partial x_{d_1}^i \ldots x_{d_n}^{i}}, \quad |d| = \sum_{i=1}^{n} d_i, \quad d_i \leq d_i \leq d_n, \quad \text{integers.}$$

(2.5)

The inner scalar product of the Hilbert space is,

$$(p, q)_{\mathcal{H}^1} = \int_{\Omega} \left( \frac{\partial p}{\partial x_i} \frac{\partial q}{\partial x_i} + pq \right) \, d\Omega.$$  

(2.6)
CHAPTER 2. THEORETICAL BACKGROUND

Newton's notation can be expanded as

$$\frac{\partial p}{\partial x_i} \frac{\partial q}{\partial x_i} = \frac{\partial p}{\partial x_1} \frac{\partial q}{\partial x_1} + \frac{\partial p}{\partial x_2} \frac{\partial q}{\partial x_2} + \frac{\partial p}{\partial x_3} \frac{\partial q}{\partial x_3},$$

(2.7)

where the repeated indices denote summation of each index. The associated norm of the Hilbert space is,

$$||u||_{H^1}(\Omega) = \left( \int_{\Omega} \frac{\partial u}{\partial x_i} \frac{\partial u}{\partial x_i} + u^2 \, d\Omega \right)^{\frac{1}{2}}.$$  

(2.8)

The set of functions in $H^1(\Omega)$ that vanish on the boundary of $\Omega$ is denoted as $H^1_0(\Omega)$.

$$H^1_0(\Omega) = \{ v | v \in H^1(\Omega), v|_{\partial \Omega} = 0 \}.$$  

(2.9)

Extending the earlier definitions, I introduce the general Sobolev spaces $H^m(\Omega)$ defined for any integer $(m \geq 1)$ such that

$$H^m(\Omega) = \{ v | v \in L^2(\Omega), D^m v \in (L^2(\Omega))^d \},$$  

(2.10)

and the corresponding norm is

$$||u||_{H^m(\Omega)} = \left( \sum_{d=0}^{m} ||D^d u||_{L^2(\Omega)}^2 \right)^{\frac{1}{2}}.$$  

(2.11)

The dual space of $H^m(\Omega)$ is defined by $H^{-m}(\Omega)$, consisting of all bounded linear functions $\ell : H^m(\Omega) \rightarrow \mathbb{R}$. That is $H^{-m}(\Omega)$ is the set of linear forms that are continuous on $H^m(\Omega)$. For $\ell \in H^{-m}(\Omega)$ the associated norm is given by

$$||\ell||_{H^{-m}} = \sup_{||v||_{H^m(\Omega)}} \frac{(\ell, v)}{||v||_{H^m(\Omega)}}.$$  

(2.12)

The $L^2$-inner product is generalized by defining the duality pairing associated with these spaces: For a functional $\ell \in H^{-m}(\Omega)$ and a function $v \in H^m(\Omega)$, it can be written

$$\ell(v) = (l, v)_{H^m(\Omega)}.$$  

(2.13)

Finally, the space $H^{1/2}(\partial \Omega)$ is presented to specify the space of function associated with the boundary data $g$ or associated with linear boundary functionals. This space is defined by the norm

$$||u||_{H^{1/2}(\partial \Omega)} = \inf_{\{ u \in H^1(\Omega), u|_{\partial \Omega} = g \}} ||u||_{H^1(\Omega)}.$$  

(2.14)
CHAPTER 2. THEORETICAL BACKGROUND

Note that, this norm suggests that the boundary data can be less regular than that of $\mathcal{H}^1(\partial \Omega)$ but more than that of $L^2(\partial \Omega)$.

The Lax-Milgram theorem is now introduced to show well-posedness of certain partial differential equations. Let $a(w, v)$ be a bilinear form where $v, w$ are each defined on a Hilbert space $\mathcal{H}_0^1(\Omega)$. Then assume that the function $a(\cdot)$ is continuous

$$|a(w, v)| \geq C||w||_{\mathcal{H}_0^1(\Omega)} \cdot ||v||_{\mathcal{H}_0^1(\Omega)}, \quad \forall w, v \in \left(\mathcal{H}_0^1(\Omega)\right)^2. \quad (2.15)$$

and coercive, i.e. there exist a $C_1 > 0$ such that

$$a(w, w) \leq C_1||w||_{\mathcal{H}_0^1(\Omega)}^2 \quad \forall w \in \mathcal{H}_0^1(\Omega). \quad (2.16)$$

Then, for every bounded functional $\ell \in \mathcal{H}^{-1}(\Omega)$, there exists unique element $u_\ell \in \mathcal{H}^1(\Omega)$ such that

$$a(u_\ell, v) = \ell(v), \quad \forall v \in \mathcal{H}^1(\Omega), \quad (2.17)$$

and that satisfies the a priori estimate

$$||u||_{\mathcal{H}^1(\Omega)} \leq \frac{||\ell||_{\mathcal{H}^{-1}(\Omega)}}{C_1}. \quad (2.18)$$

Relatedly the Riesz representation theorem states that for every $\ell \in \mathcal{H}^{-1}(\Omega)$ there is a unique member of $\mathcal{H}^1(\Omega)$, $u_\ell$ such that

$$\ell(v) = (v, u_\ell)_{\mathcal{H}^1}, \quad \forall v \in \mathcal{H}^1(\Omega). \quad (2.19)$$

2.2 Governing Equation

The governing equation for the model problem is the energy equation. The energy equation can be compactly written in the general form:

$$\rho \left( \frac{\partial e}{\partial t} + U_k \frac{\partial e}{\partial x_k} \right) + \frac{\partial q}{\partial x_j} - \dot{q} = -p \left( \frac{\partial U_k}{\partial x_k} \right) + \Phi. \quad (2.20)$$

Here, $e$ is the internal energy, $q$ is the diffusive flux of heat, $\dot{q}$ is the heat generation, and $\Phi$ is called the dissipation function. The reason $\Phi$ is called the dissipation function is that it is a measure of the rate at which mechanical energy is being converted into thermal energy.
CHAPTER 2. THEORETICAL BACKGROUND

This may be readily verified by considering an incompressible fluid in a Cartesian-coordinate system. Then

\[ \Phi = \mu \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \frac{\partial U_i}{\partial x_j} \]

\[ = \frac{1}{2} \mu \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right)^2, \quad (2.21) \]

is a positive definite quantity. This shows that the dissipation function always works to increase irreversibly the internal energy of an incompressible fluid.

Using Fourier's law which is \( q = -\kappa \frac{\partial T}{\partial x_j} \), the energy equation for a Newtonian fluid becomes

\[ \rho \left( \frac{\partial e}{\partial t} + U_k \frac{\partial e}{\partial x_k} \right) = -p \left( \frac{\partial U_k}{\partial x_k} \right) + \frac{\partial}{\partial x_j} \left( \kappa \frac{\partial T}{\partial x_j} \right) + \Phi + \dot{q}. \quad (2.22) \]

The enthalpy \( h \) per unit mass is introduced next, in a search for a more convenient form of the "energy" equation, as a replacement for thermal internal energy \( e \) per unit mass. By definition,

\[ h = e + \frac{p}{\rho}. \quad (2.23) \]

Then equation (2.22) is now reformulated as

\[ \rho \left( \frac{\partial h}{\partial t} + U_k \frac{\partial h}{\partial x_k} \right) = \frac{\partial p}{\partial t} + U_k \frac{\partial p}{\partial x_k} + \frac{\partial}{\partial x_j} \left( \kappa \frac{\partial T}{\partial x_j} \right) + \Phi + \dot{q}. \quad (2.24) \]

where \( T \) is temperature. The alternative form of equation (2.24) is

\[ \frac{\rho D_h}{Dt} = \frac{Dp}{Dt} + \frac{\partial}{\partial x_j} \left( \kappa \frac{\partial T}{\partial x_j} \right) + \Phi + \dot{q}. \quad (2.25) \]

The relationship between the enthalpy and the specific heat at constant pressure is given by

\[ \frac{Dh}{Dt} = C_p \frac{DT}{Dt} + \frac{1}{\rho} (1 - \beta T) \frac{Dp}{Dt}, \quad (2.26) \]

where \( \beta \) is the coefficient of thermal expansion

\[ \beta = -\frac{1}{\rho} \left( \frac{\partial p}{\partial T} \right)_{p=\text{const}}. \quad (2.27) \]

Then equation (2.26) becomes

\[ \rho C_p \frac{DT}{Dt} = \frac{\partial}{\partial x_j} \left( \kappa \frac{\partial T}{\partial x_j} \right) + \beta T \frac{Dp}{Dt} + \Phi + \dot{q}. \quad (2.28) \]
CHAPTER 2. THEORETICAL BACKGROUND

For the calorically perfect Newtonian fluids in which \( \rho, \mu, \) and \( \kappa \) are all constants, then \( \beta = 0 \). If the viscous dissipation is negligible, then \( \Phi = 0 \). Therefore the energy equation is reformulated as

\[
\rho C_p \frac{DT}{Dt} = \frac{\partial}{\partial x_j} \left( \kappa \frac{\partial T}{\partial x_j} \right) + \dot{q}.
\]  
(2.29)

For the steady state condition, equation (2.29) becomes,

\[
- \frac{\partial}{\partial x_j} \left( \varepsilon \frac{\partial T}{\partial x_j} \right) + U_j \frac{\partial T}{\partial x_j} = f,
\]  
(2.30)

where \( \varepsilon \) is the thermal diffusivity which is defined by \( \varepsilon = \kappa/\rho C_p \) and \( f \) is force term which is denoted as \( f = \dot{q}/\rho C_p \).

2.2.1 Convection-Diffusion Equation

Equation (2.30) is a convection-diffusion equation. The model problem is: Find temperature \( \Theta(x) \) that satisfies

\[
- \frac{\partial}{\partial x_i} (\varepsilon \frac{\partial \Theta}{\partial x_i}) + U_i \frac{\partial \Theta}{\partial x_i} = f \quad \text{in } \Omega, \ i = 1, 2, 3,
\]  
(2.31)

with boundary conditions

\[
\Theta = g_D \quad \text{on } \Gamma_D,
\]  
(2.32)

\[
\varepsilon \frac{\partial \Theta}{\partial n} = g_N \quad \text{on } \Gamma_N.
\]  
(2.33)

where \( \varepsilon \) is the thermal diffusivity and \( \Omega \) is a bounded domain in \( \mathbb{R}^3 \) with Lipschitz boundary condition \( \overline{\Gamma}_D \cap \overline{\Gamma}_N = \partial \Omega \) where \( \overline{\Gamma}_D \) and \( \overline{\Gamma}_N \) are the Dirichlet and Neumann portions of the domain boundary. It is required that \( \Gamma_D \) be non-empty and \( U_i \hat{n}_i \geq 0 \) on \( \Gamma_N \). The data is assumed to be smooth, i.e. \( f \in \mathcal{H}^1(\Omega), g_D \in \mathcal{H}^{1/2}(\Gamma_D), \) and \( g_N \in \mathcal{H}^{-1/2}(\Gamma_N) \).

For this problem \( x = (x_1, x_2, x_3) \) with corresponding unit vector \( \hat{x}_1, \hat{x}_2, \hat{x}_3 \). \( \Omega \) is the unit cube, \([0,1] \times [0,1] \times [0,1] \) and six sides of which are denoted \( \Gamma^j, j = 1, \ldots, 6 \), as shown in Figure 2.1. For the first test case, which is used for validation, the homogeneous Dirichlet boundary conditions with data \( g_D = 0 \) is considered. The velocity is prescribed as \( U = (1,1,1) \). and \( f = 1 \).

The variational weak form of equation (2.31) is: Find \( \Theta \in \mathcal{H}_D^1(\Omega) \) such that

\[
\int_{\Omega} \varepsilon \frac{\partial v}{\partial x_j} \frac{\partial \Theta}{\partial x_j} dV + vU_j \frac{\partial \Theta}{\partial x_j} = \int_{\Omega} vf dV + \int_{\Gamma_N} vg_N dA, \quad \forall v \in \mathcal{H}_0^1(\Omega).
\]  
(2.34)
where

$$\mathcal{H}_D^1(\Omega) = \{ v \in \mathcal{H}^1(\Omega) | v|_{\Gamma_D} = g_D \},$$  \hspace{1cm} (2.35)

and

$$\mathcal{H}_0^1(\Omega) = \{ v \in \mathcal{H}^1(\Omega) | v|_{\Gamma_D} = 0 \},$$  \hspace{1cm} (2.36)

where \(dV\) is a differential volume element and \(dA\) is a differential area element. If there is no Neumann boundary condition, then \(g_N = 0\). Since \(\Gamma_D\) is non-empty, the three-dimensional bilinear form is coercive and uniqueness directly follows from the Lax-Milgram theorem.

A special case of the convection–diffusion equation is the Possion equation (i.e. heat conduction equation) for which \(U = (0, 0, 0)\).
2.2.2 Linear Functional Outputs

The particular linear functional outputs are described. The first output of interests is the average solution over the entire domain. The output is defined by

\[ s^{(1)}(\Theta) = \frac{1}{|\Omega|} \int_{\Omega} \Theta \, dV, \quad (2.37) \]

where \(|\Omega|\) is the volume of the entire domain.

The average value of the field solution in the vicinity of the corner \((x_1, x_2, x_3) = (1, 1, 1)\) shall also be investigated. The average solution of corner volume is the second output of interests. It is expected that the maximum value occurs inside that corner. In particular, the corner cube \(\Omega^O\), which is \([2/3, 1] \times [2/3, 1] \times [2/3, 1]\), is identified in Figure 2.1. This output is defined

\[ s^{(2)}(\Theta) = \frac{1}{|\Omega^O|} \int_{\Omega^O} \Theta \, dV, \quad (2.38) \]

where \(|\Omega^O|\) is the volume of the boundary corner cube, which is \(1/27\) of the volume of the entire cube.

2.3 Galerkin Finite Element Method

In this section the general finite element approximation is presented in matrix forms. Here only the \(H\)-mesh (coarse "working" mesh) problems are considered.

2.3.1 Discretization

For simplicity, only linear Galerkin finite element approximations on the \(H\)-mesh tetrahedra \(T_H\) are considered. The classical continuous-piecewise-polynomial in finite element subspaces is applied. The finite element subspaces \(X_H \subset H^1_0(\Omega)\) and \(\tilde{X}_H \subset H^1(\Omega)\) are

\[ X_H = \{ v|_{T_H} \in P_1(T_H), \forall T_H \in T_H \} \cap H^1_0(\Omega), \quad (2.39) \]

\[ \tilde{X}_H = \{ v|_{T_H} \in P_1(T_H), \forall T_H \in T_H \} \cap H^1(\Omega). \quad (2.40) \]

The variational weak form in equation (2.34) can be written in the discrete form: Find \(\Theta_H \in \tilde{X}_H\) such that

\[ \int_{\Omega} \varepsilon \frac{\partial w}{\partial x_j} \frac{\partial \Theta_H}{\partial x_j} \, dV + w U_j \frac{\partial \Theta_H}{\partial x_j} \, dV = \int_{\Omega} w f \, dV + \int_{\Gamma_N} w g_N \, dA, \forall w \in X_H. \quad (2.41) \]
In order to obtain the finite element approximation of the solution $\Theta(x)$, the basis are introduced.

$$\Theta_H(x) = \sum_{j=1}^{N_k} (\Theta_H)_j \phi_j(x), \quad \forall T_H \in \tilde{X}_H,$$

$$w(x) = \sum_{j=1}^{N_k} w_j \phi_j(x), \quad \forall w \in X_H,$$

where $\phi_i \in \tilde{X}_H$, $\phi_i(x_j) = \delta_{ij}$ (the Kronecker delta), and $1 \leq i, j \leq N_k$. $N_k$ is the number of elements.

Substituting equations (2.42) and (2.43) into the weak form (2.41) leads to

$$\sum_{j=1}^{N_k} \tilde{L}_{ij} \tilde{\Theta}_j = \tilde{f}_i, \quad \forall i$$

where $()$ represents all the values associated with the boundary rows.

The components in (2.44) is evaluated in terms of the linear basis functions. In particular, the discrete operators are given by

$$\tilde{L}_{ij} = \tilde{A}_{ij} + \tilde{C}_{ij} = a(\phi_i, \phi_j),$$

$$\tilde{A}_{ij} = \int_{\Omega} \sum_{m=1}^{3} \varepsilon \frac{\partial \phi_i}{\partial x_m} \frac{\partial \phi_j}{\partial x_m} dV,$$

$$\tilde{C}_{ij} = \int_{\Omega} \sum_{k=1}^{N_k} \sum_{m=1}^{3} \phi_i(U_m) \phi_j \frac{\partial \phi_k}{\partial x_m} dV,$$

$$\tilde{f}_i = \int_{\Omega} \phi_i f dV + \int_{\Gamma_N} \phi_i g_N dA,$$

where $1 \leq i, j \leq N_k$, $\tilde{A}_{ij}$ is defined as the diffusive matrix term, $\tilde{C}_{ij}$ is denoted as the convective matrix term, and $\tilde{f}_i$ is force vector term. However $\tilde{A}_{ij}$ is the symmetric matrix due to diffusion, $\tilde{C}_{ij}$ is the anti-symmetric (skew-symmetric) matrix due to convection. If there are no Neumann boundary conditions, equation (2.48) simplifies to

$$\tilde{f}_i = \int_{\Omega} \phi_i f dV = \tilde{e}^N(\phi_i).$$

Alternative form of equation (2.44) is the linear matrix form as follows.

$$\tilde{L} \tilde{\Theta}_H = \tilde{f},$$

(2.50)
where $\tilde{L}$ is non-symmetric positive-definite sparse matrix arising from the finite element discretization and $\tilde{f}$ is a right-hand side vector representing a prescribed force. The unknown nodal values of $\Theta_H$ are obtained by solving this algebraic system.

### 2.3.2 Elemental Constructs

The discrete equation (2.50) constructed the system matrices is formulated into an reference elemental level. The direct stiffness summation is used to sum up all the elemental contributions to the system. The idea is to use a transformation that converts the real elements to a reference element on which the basis function do not depend on the specific placement, orientation and size of the computed element. The linear $P_1 - \text{tetrahedron}$ and a local to global mapping $g_j^k$ are introduced. The local to global mapping $g_j^k$ takes inputs as the element number ($k = 1, \ldots, K$) and the local node number ($j = 1, 2, 3, 4$) and gives outputs as the corresponding global node number.

Every element $\Omega^k$ is mapped onto the reference element $\hat{\Omega}$ by using the affine transformation. Writing it as $x(x_1, x_2, x_3) = F^k(\xi)$ where $\xi = (\xi_1, \xi_2, \xi_3)$. In matrix form the affine transformation is

$$\xi_\alpha = C^k_\alpha + \sum_{\beta=1}^{3} D^k_{\alpha\beta} x_\beta.$$  

(2.51)

The vector $C^k_\alpha$ and the $D^k_{\alpha\beta}$ are defined as

$$C^k_\alpha = \begin{bmatrix} x_1[a+3][x_2[a+1][x_3[a+2]] - x_2[a+2][x_3[a+1]] \\ -x_2[a+3][x_1[a+1][x_3[a+2]] - x_1[a+2][x_3[a+1]] \\ +x_3[a+3][x_1[a+1][x_2[a+2]] - x_1[a+2][x_2[a+1]] \end{bmatrix} / (6 vol^k).$$

(2.52)

$$D^k_{\alpha\beta} = \frac{\partial \xi_\alpha}{\partial x_\beta},$$

(2.53)

where $\alpha = 1, 2, 3, 4$, $\beta = 1, 2, 3$, and $[\alpha] = \alpha \mod 4$. Note that, for this transformation

$$x = \sum_{\alpha=1}^{4} \hat{x}^k_\alpha \xi_\alpha,$$

(2.54)

where $\hat{x}^k_\alpha$ is the coordinate vector of local node $\alpha$ of element $k$. It follows that $dx = |J|d\xi$ where $J$ is the Jacobian. The Jacobian is defined by

$$J = \begin{vmatrix} \frac{\partial x_1}{\partial \xi_1} & \frac{\partial x_1}{\partial \xi_2} & \frac{\partial x_1}{\partial \xi_3} \\ \frac{\partial x_2}{\partial \xi_1} & \frac{\partial x_2}{\partial \xi_2} & \frac{\partial x_2}{\partial \xi_3} \\ \frac{\partial x_3}{\partial \xi_1} & \frac{\partial x_3}{\partial \xi_2} & \frac{\partial x_3}{\partial \xi_3} \end{vmatrix}.$$
CHAPTER 2. THEORETICAL BACKGROUND

Figure 2.2: Reference Element

\[
- \frac{\partial x_1}{\partial \xi_2} \left( \frac{\partial x_2}{\partial \xi_1} \frac{\partial x_3}{\partial \xi_3} - \frac{\partial x_3}{\partial \xi_2} \frac{\partial x_2}{\partial \xi_1} \right) + \frac{\partial x_1}{\partial \xi_3} \left( \frac{\partial x_2}{\partial \xi_1} \frac{\partial x_3}{\partial \xi_2} - \frac{\partial x_2}{\partial \xi_2} \frac{\partial x_3}{\partial \xi_1} \right) \tag{2.55}
\]

The reference element is described in the Figure 2.2. The reference element space \( X_H \) is introduced by

\[
P_1(\hat{\Omega}) = \{ \tilde{v} \in P_1(\Omega^k) \} \quad \text{(i.e.,} \quad \tilde{v} = \tilde{a} + \tilde{b} \xi_1 + \tilde{c} \xi_2 + \tilde{d} \xi_3 \} \tag{2.56}
\]

Its reference elemental basis is

\[
h_i \in P_1(\hat{\Omega}), \quad i = 1, 2, 3, 4, \tag{2.57}
\]

\[
h_i(\xi_j) = \delta_{ij}, \quad 1 \leq i, j \leq 4, \tag{2.58}
\]

where

\[
h_1 = \xi_1, \quad h_2 = \xi_2,
\]
Hence for any $\tilde{v} \in P_1$, having
\[ \tilde{v}(\xi) = \sum_{\alpha=1}^{4} \tilde{v}_{\alpha} h_{\alpha}(\xi). \] (2.60)

The global nodes $\{x\} = \{x_1, ..., x_{N_k}\}$, and corresponding $g_j^k$, are defined in the usual way.
- $\forall k \in 1, ..., N_k, \forall j \in 1, 2, 3, 4, \ x_i = x_{g_j^k}$,
- $x_i \neq x_j$ for $i \neq j$,
- $\forall i$ there exist at least one pair $(k, j)$, $k \in \{1, ..., K\}$, $j \in \{1, 2, 3, 4\}$.

such that $\tilde{x}_j^k = x_i$.

Associated with these global nodes, the basis functions (for $\tilde{X}_H = \{v|_{\Omega^k} \in P_1(\Omega^k)\} \cap H^1(\Omega)$)
\[ \phi_i, \ i = 1, ..., N_k, \] are defined as
\[ \phi_i \in \tilde{X}_H, \ \phi_i(x_j) = \delta_{ij}, \ 1 \leq i, j \leq N_k. \] (2.61)

Given the affine mapping, it follows that, for $k = 1, ..., K$,
\[ h_j((F^k)^{-1}(x)) = \phi_j(x)|_{\Omega^k}, \] (2.62)

and thus the assembly procedure for linear elements will directly extend.

Elemental matrices such as $\hat{A}_{\alpha\beta}$, $\hat{C}_{\alpha\beta}$, and $\hat{M}_{\alpha\beta}$ in following sections are evaluated using the numerical integration over a reference element, i.e.
\[ \int_{\Omega} f(x) \, dV = \int_{\tilde{\Omega}} f(\xi) |\mathbf{J}| \, d\xi_1 d\xi_2 d\xi_3, \] (2.63)

where $\mathbf{J}$ is the Jacobian. The integration in equation (2.63) is carried out using the numerical integration scheme [28], i.e.
\[ \int_{\tilde{\Omega}} f(\xi) |\mathbf{J}| \, d\xi_1 d\xi_2 d\xi_3 = \text{vol}^k |\mathbf{J}| \sum_{j=1}^{N_p} w_j f(\xi_{1j}, \xi_{2j}, \xi_{3j}, \xi_{4j}), \] (2.64)

where $N_p$ is the number of integration sampling points, $(\xi_{1j}, \xi_{2j}, \xi_{3j}, \xi_{4j})$ are the tetrahedral coordinates of the $j^{th}$ integration sampling point, and $w_j$ is the weight associated with the $j^{th}$ sampling point. For linear function, the weight is $w = 1$ and the reference coordinate is
\[ (\xi_1, \xi_2, \xi_3, \xi_4) = \left( \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4} \right). \] (2.65)
CHAPTER 2. THEORETICAL BACKGROUND

This quadrature is used for the calculation of the cube case where \( U_i = (1, 1, 1) \) and \( f = 1 \).

For quadratic functions arising from multiplication of two linear basis, where sampling points \( N_p = 4 \), the weights are \( w_j = 1/4 \) and reference coordinates are

\[
(a, \beta, \beta, \beta), (\beta, a, \beta, \beta), (\beta, \beta, a, \beta), (\beta, \beta, \beta, a).
\]

where \( a = 0.58541020 \), \( \beta = 0.13819660 \).

This quadrature is exploited for the computation of realistic problem, where the \( U_i \) is not uniform and needs to be integrated using the quadratic reference element.

2.3.3 The Elemental Matrix of Diffusive Term

In this section, the diffusive term \( A_{ij} \) in equation (2.46) is reformulated using the reference elemental basis.

Using equation (2.62), \( A_{ij} \) over an element \( k \) can now be written as

\[
A_{ij}^k = \int \sum_{m=1}^{3} \frac{\partial h_a}{\partial x_m} \frac{\partial h_a}{\partial x_m} dV
\]

\[
= \sum_{m=1}^{3} \int \varepsilon \left( \sum_{n=1}^{3} \frac{\partial h_a}{\partial \xi_n} \frac{\partial \xi_n}{\partial x_m} \right) \left( \sum_{n'=1}^{3} \frac{\partial h_a}{\partial \xi_{n'}} \frac{\partial \xi_{n'}}{\partial x_m} \right) dV.
\]

Let \( G_{an} = \frac{\partial h_a}{\partial \xi_n} \) and \( G_{bn'} = \frac{\partial h_a}{\partial \xi_{n'}} \), then

\[
A_{ij}^k = \sum_{m=1}^{3} \int \varepsilon \left( \sum_{n=1}^{3} G_{an} \frac{\partial \xi_n}{\partial x_m} \right) \left( \sum_{n'=1}^{3} G_{bn'} \frac{\partial \xi_{n'}}{\partial x_m} \right) |J| d\xi_1 d\xi_2 d\xi_3
\]

\[
= \varepsilon \sum_{m=1}^{3} \left( \sum_{n=1}^{3} G_{an} \frac{\partial \xi_n}{\partial x_m} \right) \left( \sum_{n'=1}^{3} G_{bn'} \frac{\partial \xi_{n'}}{\partial x_m} \right) \frac{1}{6} |J|.
\]

Note that, \( dV = |J| d\xi_1 d\xi_2 d\xi_3 \) and \( \int_{\Omega} d\xi_1 d\xi_2 d\xi_3 = \frac{1}{6} \) in above equation. Expanding equation (2.68),

\[
A_{ij}^k = \varepsilon \left[ \left( \sum_{n=1}^{3} G_{an} \frac{\partial \xi_n}{\partial x_1} \right) \left( \sum_{n'=1}^{3} G_{bn'} \frac{\partial \xi_{n'}}{\partial x_1} \right) \right]
\]

\[
+ \left( \sum_{n=1}^{3} G_{an} \frac{\partial \xi_n}{\partial x_2} \right) \left( \sum_{n'=1}^{3} G_{bn'} \frac{\partial \xi_{n'}}{\partial x_2} \right)
\]

\[
+ \left( \sum_{n=1}^{3} G_{an} \frac{\partial \xi_n}{\partial x_3} \right) \left( \sum_{n'=1}^{3} G_{bn'} \frac{\partial \xi_{n'}}{\partial x_3} \right) \frac{1}{6|J|}.
\]
Hence,

\[
\hat{A}_{\alpha\beta}^k = \frac{1}{6|J|} \left[ \left( \sum_{n=1}^{3} G_{\alpha n} a_n^k \right) \left( \sum_{n=1}^{3} G_{\beta n} a_n^k \right) + \left( \sum_{n=1}^{3} G_{\alpha n} b_n^k \right) \left( \sum_{n=1}^{3} G_{\beta n} b_n^k \right) + \left( \sum_{n=1}^{3} G_{\alpha n} c_n^k \right) \left( \sum_{n'=1}^{3} G_{\beta n'} c_n^k \right) \right],
\]

(2.70)

where

\[
G_{\alpha n} = \frac{\partial h_\alpha}{\partial \xi_n}, \\
G_{\beta n} = \frac{\partial h_\beta}{\partial \xi_n},
\]

(2.71)

and

\[
da_n^k = J \frac{\partial \xi_n}{\partial x_1} = \left( \frac{\partial x_2}{\partial \xi_{[n+1]}}, \frac{\partial x_3}{\partial \xi_{[n+1]}} - \frac{\partial x_2}{\partial \xi_{[n+2]}}, \frac{\partial x_3}{\partial \xi_{[n+2]}} \right),
\]

\[
bb_n^k = J \frac{\partial \xi_n}{\partial x_2} = \left( \frac{\partial x_1}{\partial \xi_{[n+1]}}, \frac{\partial x_3}{\partial \xi_{[n+1]}} - \frac{\partial x_1}{\partial \xi_{[n+2]}}, \frac{\partial x_3}{\partial \xi_{[n+2]}} \right),
\]

\[
cn_n^k = J \frac{\partial \xi_n}{\partial x_3} = \left( \frac{\partial x_1}{\partial \xi_{[n+1]}}, \frac{\partial x_2}{\partial \xi_{[n+1]}} - \frac{\partial x_1}{\partial \xi_{[n+2]}}, \frac{\partial x_2}{\partial \xi_{[n+2]}} \right).
\]

(2.72)

Note that, \(1 \leq \alpha, \beta \leq 4\). Its implementation is obviously very simple, however it should keep in mind that this formulation depends on the assumption that the basis functions are linear nodal functions.

### 2.3.4 The Elemental Matrix of Convective Term

Similarly to the diffusive term in equation (2.70), the convective term \(\hat{C}_{ij}\) in equation (2.47) is also reformulated using the reference elemental level.

Using once again equation (2.62), \(\hat{C}_{ij}\) over an element \(k\) can be formulated as

\[
\hat{C}_{\alpha\beta}^k = \sum_{m=1}^{3} \int_{\Omega} h_\alpha \sum_{\gamma=1}^{4} (U_m)^{\gamma} h_\gamma \frac{\partial h_\beta}{\partial x_m} dV
\]

\[
= \sum_{m=1}^{3} \sum_{\gamma=1}^{4} (U_m)^k \int_{\Omega} h_\alpha h_\gamma \frac{\partial h_\beta}{\partial \xi_n} \frac{\partial \xi_n}{\partial x_m} dV.
\]

(2.73)
CHAPTER 2. THEORETICAL BACKGROUND

Expanding equation (2.73) and use relation that \( dV = |J| d\xi_1 d\xi_2 d\xi_3 \), then

\[
\hat{C}_{\alpha\beta}^k = \sum_{\gamma=1}^{4} \int_{\Omega} \left[ (U_1)_{g\alpha} \sum_{n=1}^{3} G_{\beta\eta} \frac{\partial \xi_n}{\partial x_1} + (U_2)_{g\alpha} \sum_{n=1}^{3} G_{\beta\eta} \frac{\partial \xi_n}{\partial x_2} + (U_3)_{g\alpha} \sum_{n=1}^{3} G_{\beta\eta} \frac{\partial \xi_n}{\partial x_3} \right] h_\alpha h_\gamma |J| d\xi_1 d\xi_2 d\xi_3. \tag{2.74}
\]

Since \( \int_{\Omega} d\xi_1 d\xi_2 d\xi_3 = \frac{1}{6} \), equation (2.74) becomes

\[
\hat{C}_{\alpha\beta}^k = \sum_{\gamma=1}^{4} \left[ (U_1)_{g\alpha} \sum_{n=1}^{3} G_{\beta\eta} \frac{\partial \xi_n}{\partial x_1} + (U_2)_{g\alpha} \sum_{n=1}^{3} G_{\beta\eta} \frac{\partial \xi_n}{\partial x_2} + (U_3)_{g\alpha} \sum_{n=1}^{3} G_{\beta\eta} \frac{\partial \xi_n}{\partial x_3} \right] h_\alpha h_\gamma \frac{1}{6}. \tag{2.75}
\]

Use equations (2.72), then equation (2.75) is reformulated as

\[
\hat{C}_{\alpha\beta}^k = \frac{1}{6} \sum_{\gamma=1}^{4} h_\alpha h_\gamma \left[ (U_1)_{g\alpha} \sum_{n=1}^{3} G_{\beta\eta} a_n^k + (U_2)_{g\alpha} \sum_{n=1}^{3} G_{\beta\eta} b_n^k + (U_3)_{g\alpha} \sum_{n=1}^{3} G_{\beta\eta} c_n^k \right]. \tag{2.76}
\]

Note also that, \( 1 \leq \alpha, \beta \leq 4 \). It is easily seen from equation (2.76) that the reference elemental matrix is anti-symmetric (skew-symmetric).

2.3.5 The Elemental Vector of Force Term

In this section, the force term \( \hat{f}_i \) in equation (2.49) is reformulated for the reference elemental basis. If \( \hat{f}_i \) is a continuous function, the reference elemental basis of \( \hat{f}_i^k \) can be approximated as

\[
\hat{f}_i^k \approx \sum_{\beta=1}^{4} \int_{\Omega} h_\alpha h_\beta f(x_\beta^k) dV. \tag{2.77}
\]

Since the Jacobian is \( dV = |J| d\xi_1 d\xi_2 d\xi_3 \), then

\[
\hat{f}_i^k \approx \sum_{\beta=1}^{4} \int_{\Omega} h_\alpha h_\beta f(x_\beta^k) |J| d\xi_1 d\xi_2 d\xi_3. \tag{2.78}
\]

Introducing the mass matrix:

\[
\hat{M}_{\alpha\beta}^k = |J| \int_{\Omega} h_\alpha h_\beta d\xi_1 d\xi_2 d\xi_3 = \frac{1}{6} |J| h_\alpha h_\beta. \tag{2.79}
\]
CHAPTER 2. THEORETICAL BACKGROUND

Using equation (2.79), equation (2.78) becomes

\[ \hat{f}_\alpha^k \approx \sum_{\beta=1}^{4} \hat{M}_{\alpha\beta}^k f(\hat{x}_\beta^k). \] (2.80)

Since \( f = 1 \) for the first test case, therefore equation (2.80) is reformulated as

\[ \hat{f}_\alpha^k \approx \sum_{\beta=1}^{4} \hat{M}_{\alpha\beta}^k = \sum_{\beta=1}^{4} \frac{1}{6} |J| \ h_c h_\beta. \] (2.81)

2.3.6 Assembly

The stiffness matrix \( \hat{L} \) is obtained by assembling the diffusive elemental matrix (2.70) and convective elemental matrix (2.76). The force vector \( \hat{f} \) is obtained by assembling the force elemental vector (2.81). The elemental matrix of stiffness term \( (\hat{L}_{\alpha\beta}^k) \) is presented as

\[ \hat{L}_{\alpha\beta}^k = \hat{A}_{\alpha\beta}^k + \hat{C}_{\alpha\beta}^k. \] (2.82)

The assembly is performed in such a way that for a node common to several adjacent elements, the value in the matrix is equal to the sum of the contributions of several elements. The assembly is done by applying the following algorithm:

\[ \hat{f}_i = 0 \]
\[ \hat{L}_{ij} = 0 \]

For \( 1 \leq k \leq K \) {
  For \( 1 \leq \alpha \leq 4 \) {
    \[ i = g_{\alpha}^k \]
    \[ \hat{f}_i = \hat{f}_i + \hat{f}_\alpha^k \]
    For \( 1 \leq \beta \leq 4 \) {
      \[ j = g_{\beta}^k \]
      \[ \hat{L}_{ij} = \hat{L}_{ij} + \hat{L}_{\alpha\beta}^k \]
    }
  }
}.
Chapter 3

A Posteriori Error Estimation

In this chapter, the modified energy objective is introduced, and hence modified augmented Lagrangian that permits the calculation of the error estimator is presented. The constraints of the modified Lagrangian are the finite element equilibrium equations and inter-subdomain continuity requirements. Starting from the Lagrangian, the variational and algebraic formulation of the bound procedure for outputs of the three dimensional convection–diffusion equations will be developed. For the complete description of the previous bound method the reader should refer to [17, 14, 19, 15, 20, 18].

3.1 Variational Formulation

In this section the variational formulation for the bound calculations is presented in three space dimensions. Here the general finite element components required to construct the bounds shall be introduced.

3.1.1 Governing Equation

Considering a scalar linear coercive convection–diffusion problem which is characterized by the following variational statement: find $\Theta(x) \in X^D$, and $s \in R$, such that

$$a(u, \Theta) = \ell^N(u), \ \forall u \in X,$$  \hspace{1cm} (3.1)

and

$$s = \ell^O(\Theta),$$ \hspace{1cm} (3.2)
The field variable $\Theta(x)$ (i.e. temperature in this thesis) is defined over a domain $\Omega \subset \mathbb{R}^3$ with boundary $\partial \Omega = \Gamma_D$; $\Theta(x)$ must reside in $X^D = \{ v + \Theta_D | v \in X \}$, where $X = \{ v \in \mathcal{H}^1(\Omega) | v = 0 \text{ on } \Gamma_D \}$. Here $\mathcal{H}^1(\Omega)$ is the usual Hilbert space; $\Gamma_D$ is the portion of the domain boundary on which the Dirichlet boundary data, $g_D$, is imposed; $\Theta_D$ is any function in $\mathcal{H}^1(\Omega)$ satisfying the essential boundary conditions, that is, $\Theta_D|_{\Gamma_D} = g_D$. For the first test case, $g_D = 0$, which is homogeneous Dirichlet boundary conditions. Also, $a : \mathcal{H}^1(\Omega) \times \mathcal{H}^1(\Omega) \to \mathbb{R}$ is a continuous bilinear form (not necessarily symmetric), $\ell^N : \mathcal{H}^1(\Omega) \to \mathbb{R}$ is a continuous linear, and $\ell^O : \mathcal{H}^1(\Omega) \to \mathbb{R}$ is a continuous affine functional.

### 3.1.2 Elements

The elemental decomposition ("tetrahedron") is defined for the three-dimensional problem. Therefore, the subdomains are defined as non-overlapping $K$ subdomain $T_H$.

$$\overline{\Omega} = \bigcup_{T_H \in \mathcal{T}_H} \overline{T_H},$$

where the overbar denotes closure. Similarly, denoting the set of all the nodes $N$ of the tetrahedron by $\mathcal{M}(T_H)$ where $T_H$ is the subdomain discretization of $\Omega$.

Next the $h$-mesh tetrahedra $T_h$, consisting of elements $T_h$, is introduced. The $T_h$ is a uniform $R$ refinement of $T_H$.

### 3.1.3 Finite Element Spaces

Two tetrahedra of the computational domain $\Omega$ are considered: the coarse "working" or design $H$-mesh, $T_H$, consisting of $K_H$ elements of $T_H$; and the fine "truth" $h$-mesh, $T_h$, consisting of $K_h$ elements of $T_h$. The $h$-mesh tetrahedra $T_h$ is refinement of $T_H$; the geometric requirements on $T_H$ are discussed in the context of the refinement strategy.

Regular piecewise-linear continuous finite element subspaces are associated to each of these meshes,

$$X_H = \{ v \in X | v|_{T_H} \in P_1(T_H), \forall T_H \in \mathcal{T}_H \},$$
$$X_h = \{ v \in X | v|_{T_h} \in P_1(T_h), \forall T_h \in \mathcal{T}_h \},$$

where $P_1(T_\delta)$ denotes the space of linear polynomials over $T_\delta$ (where $\delta = H$ and $\delta = h$)
CHAPTER 3. A POSTERIORI ERROR ESTIMATION

respectively. Similarly, $X^D_H$ and $X^D_h$ is defined as,

\begin{align*}
X^D_H &= \{ v + \Theta_D | v \in X_H \}, \\
X^D_h &= \{ v + \Theta_D | v \in X_h \},
\end{align*}

where $\Theta_D \in H^1_D(\Omega)$ is any lifting of the Dirichlet boundary data. Note that, from the refinement hypothesis, $X_H \subset X_h \subset X$, and $X^D_H \subset X^D_h \subset X^D$.

The algorithms to be presented require that the above spaces and forms be expressed as sums of contributions over the $H$-elements $T_H$. Towards this end, the subdomain local spaces $Z_H(T_H)$ and $Z_h(T_H)$ are considered as,

\begin{align*}
Z_H(T_H) &= P_1(T_H), \quad \forall T_H \in \mathcal{T}_H, \\
Z_h(T_H) &= \{ v|_{T_h} \in P_1(T_h), \forall T_H \in \mathcal{T}_H \},
\end{align*}

where $\mathcal{R}_{T_H}$ denotes the set of $h$-mesh elements contained in $T_H$. The global representation of $Z_H(T_H)$ and $Z_h(T_H)$ are the “discontinuous subdomain” spaces $\hat{X}_H$ and $\hat{X}_h$.

\begin{align*}
\hat{X}_H &= \{ v \in L^2(\Omega) \mid v|_{T_H} \in Z_H(T_H), \forall T_H \in \mathcal{T}_H \}, \\
\hat{X}_h &= \{ v \in L^2(\Omega) \mid v|_{T_h} \in Z_h(T_H), \forall T_H \in \mathcal{T}_H \},
\end{align*}

where $L^2(\Omega)$ is the space of square-integrable functions.

Finally, the bilinear—a($\cdot$, $\cdot$), $a^*(\cdot, \cdot)$—and linear—$b^N(\cdot)$, $b^O(\cdot)$—forms are extended to accept discontinuous functions in the “discontinuous subdomain” spaces by redefining these forms as a sum of $H$-element contributions. $a(\cdot, \cdot)$ is now written as

\begin{align*}
a(w, v) = \sum_{T_H \in \mathcal{T}_H} a_{T_H}(w|_{T_H}, v|_{T_H}), \quad \forall w, v \in \hat{X}_H,
\end{align*}

where for all $T_H$ in $\mathcal{T}_H$

\begin{align*}
a_{T_H}(w, v) = \int_{T_H} \varepsilon \frac{\partial w}{\partial x_j} \frac{\partial v}{\partial x_j} + w U_j \frac{\partial v}{\partial x_j} \, dV, \quad \forall w, v \in X_H.
\end{align*}

and the symmetric part of the operator is defined as,

\begin{align*}
a^*(w, v) = \sum_{T_H \in \mathcal{T}_H} a^*_{T_H}(w|_{T_H}, v|_{T_H}), \quad \forall w, v \in \hat{X}_h,
\end{align*}
where for all $T_H$ in $\mathcal{T}_H$
\[
a^+_T(w, v) = \int_{T_H} \varepsilon \frac{\partial w}{\partial x_j} \frac{\partial v}{\partial x_j} dV + \frac{1}{2} \int_{\partial T_H \cap \Gamma_N} wU_j n_j ds, \quad \forall w, v \in X_H. \tag{3.15}
\]

The boundary term in (3.15) arises from the convection operator and only affects the Neumann boundaries. Although for problems where there are no Neumann boundaries, this term vanishes and $a^+_T(.)$ is the symmetric part of $a_T(.)$; in general $a^+_T(.)$ is not the symmetric part of $a_T(.)$.

Now the linear functional is introduced. Associated with the linear functional, it can be
\[
\ell^N(v) = \sum_{T_H \in \mathcal{T}_H} \ell^N_{T_H}(v|_{T_H}), \quad \forall v \in \hat{X}_h, \tag{3.16}
\]
where for all $T_H$ in $\mathcal{T}_H$
\[
\ell^N_{T_H}(v) = \int_{T_H} vf dV + \int_{\partial T_H \cap \Gamma_N} v g_N ds, \quad \forall v \in X_H. \tag{3.17}
\]

Associated with the output functional, it can be
\[
\ell^O(v) = \sum_{T_H \in \mathcal{T}_H} \ell^O_{T_H}(v|_{T_H}), \quad \forall v \in \hat{X}_h, \tag{3.18}
\]
such that
\[
\ell^O(v) = \ell(v), \quad \forall v \in X_H. \tag{3.19}
\]
Here $\ell()$ is the formal output functional. Note that the construction of equation (3.18) and (3.19) permits to evaluate the output associated with a field variable which is not in $\mathcal{H}^1(\Omega)$.

### 3.1.4 Continuity Form
Let $\mathcal{E}(\mathcal{T}_H)$ and $\mathcal{E}(\mathcal{T}_h)$ denote the set of open faces in the tetrahedron $\mathcal{T}_H$ and $\mathcal{T}_h$. Then introducing a space of functions over the element faces $\gamma_H \in \mathcal{E}(\mathcal{T}_H)$,
\[
\mathcal{Q}_H = \{y|_{\gamma_H} \in \mathcal{P}_1(\gamma_H), \forall \gamma_H \in \mathcal{E}(\mathcal{T}_H), y|_{\Gamma_N} = 0\}. \tag{3.20}
\]
similarly
\[
\mathcal{Q}_h = \{y|_{\gamma_h} \in \mathcal{P}_1(\gamma_h), \forall \gamma_h \in \mathcal{E}(\mathcal{T}_h) \cap \mathcal{E}(\mathcal{T}_H), y|_{\Gamma_N} = 0\}. \tag{3.21}
\]
It follow that $\mathcal{Q}_H \subset \mathcal{Q}_h \subset \mathcal{H}^{-1/2}(\mathcal{E}(\mathcal{T}_H))$; the functions in these spaces can, of course, be discontinuous.
Next, "jump" bilinear form $b$ is considered: $\hat{X}_h \times Q_h \rightarrow \mathbb{R}$,

$$b(v, q) = \sum_{\gamma_H \in \mathcal{E}(\mathcal{T}_H)} \int_{\gamma_H} J_{\gamma_H} v q|_{\gamma_H} ds,$$  \hspace{1cm} (3.22)

where $J_{\gamma_H} v$ is the jump in $v$ across $\gamma_H$ when $\gamma_H$ is an interior face, and the trace of $v$ on $\gamma_H$ when $\gamma_H$ is on the boundary $\partial \Omega$. Note that $q$ is defined only over the faces of the tetrahedron and may, of course, be discontinuous; $q$ may also be defined as the flux associated with a function $Q_d$. Effectively, equation (3.22) computes the moments of the "jumps" in $v$ over interior faces, and the moments of $v$ over boundary faces.

Similarly, the related linear functional associated with the Dirichlet boundary conditions is also required,

$$\ell^D(q) = \sum_{\gamma_H \in \mathcal{E}(\mathcal{T}_H)} \int_{\gamma_H} J_{\gamma_H} g \hat{D} q|_{\gamma_H} ds,$$  \hspace{1cm} (3.23)

which computes the moments of the imposed boundary conditions on $\Gamma_D$.

The form $b(\cdot)$ and $\ell^D(\cdot)$ can be used to enforce continuity on functions in $\hat{X}_H$ and $\hat{X}_h$: in particular,

$$X_H = \{ v \in \hat{X}_H \mid b(v, q) = 0, \ \forall q \in Q_h \},$$  \hspace{1cm} (3.24)

$$X_h = \{ v \in \hat{X}_h \mid b(v, q) = 0, \ \forall q \in Q_h \}. $$  \hspace{1cm} (3.25)

Note $b(\cdot)$ places no restriction on $v$ on natural boundaries.

3.1.5 $H$-mesh and $h$-mesh Problems

The Galerkin finite element solution on the coarse "working" mesh, $\Theta_H \in X_H^D$, satisfies

$$a(w, \Theta_H) = \ell^N(w), \ \forall w \in X_H,$$  \hspace{1cm} (3.26)

and the associated output of interest, $s_H$, is then computed as

$$s_H = \ell^O(\Theta_H).$$  \hspace{1cm} (3.27)

Similarly, the Galerkin finite element solution on the fine "truth" mesh, $\Theta_h \in X_h^D$, satisfies

$$a(w, \Theta_h) = \ell^N(w), \ \forall w \in X_h,$$  \hspace{1cm} (3.28)
and the associated fine "truth" output of interest, $s_h$, is then computed as

$$s_h = \ell^O(\Theta_h).$$

(3.29)

In the smooth case, $|s - s_H| \sim O(H^2)$, and $|s - s_h| \sim O(h^2)$, where $H$ and $h$ are length characteristics of $\mathcal{T}_H$ and $\mathcal{T}_h$, respectively.

It will be assumed that the calculation of $\Theta_H$, and hence $s_H$, can be performed at moderate cost. However, the $h$-mesh will typically be much finer than the $H$-mesh, in order to ensure that $|s - s_h|$ is suitably small; the computation of $\Theta_h$ and $s_h$ will thus, most likely, be impractical. The objective of this method is, therefore, to devise a procedure that will yield sharp upper and lower bounds for $s_h$ in an affordable manner.

### 3.1.6 Lagrangian Formulation

The approach is based on the construction of an augmented Lagrangian, in which the objective is a quadratic "energy" reformation of the desired output, and the constraints are the finite element equilibrium equations and the inter-subdomain continuity requirements.

To construct the augmented Lagrangian, $w = \Theta_h - \Theta_H$ is set in equation (3.28). Note that $\Theta_h - \Theta_H \in X_h$ and $\Theta_h \in X^D_h$, then

$$a(\Theta_h - \Theta_H, \Theta_h) - \ell^N(\Theta_h - \Theta_H) = 0.$$  

(3.30)

Equation (3.30) can be written as

$$a(\Theta_h - \Theta_H, \Theta_h - \Theta_H) + a(\Theta_h - \Theta_H, \Theta_H) - \ell^N(\Theta_h - \Theta_H) = 0.$$  

(3.31)

Introducing the error $e_h \in X_h$ as the difference between the $h$-mesh and $H$-mesh approximations, that is $e_h = \Theta_h - \Theta_H$, and equation (3.31) becomes,

$$a(e_h, e_h) + a(e_h, \Theta_H) - \ell^N(e_h) = 0.$$  

(3.32)

Also defining the residual as $\ell^E(v) = \ell^N(v) - a(v, \Theta_H)$, then

$$a(e_h, e_h) - \ell^E(e_h) = 0, \forall e_h \in X_h.$$  

(3.33)

Rewriting $a(e_h, e_h)$ in term of $a^*(e_h, e_h)$ gives

$$a(e_h, e_h) = a^*(e_h, e_h) + cu.$$  

(3.34)
where recall that $a^s(\cdot, \cdot)$ is the symmetric part of $a(\cdot, \cdot)$. The term $c_U$ in equation (3.34) arises from the convection operator which is given by

$$c_U = \frac{1}{2} \int_{\Gamma_D} \left( \Theta_H^2 - \Theta_h^2 \right) U_i \hat{n}_i \, ds. \tag{3.35}$$

Note that, since inhomogeneous Dirichlet boundary values are the same for both $H$-mesh and $h$-mesh (i.e. $\Theta_H|_{\Gamma_D} = \Theta_h|_{\Gamma_D} = g_D$), then $c_U = 0$. So that $a(\cdot, \cdot) = a^s(\cdot, \cdot)$.

Then equation (3.33) can be reformulated as

$$a^s(e_h, e_h) - \ell^E(e_h) = 0, \quad \forall e_h \in X_h. \tag{3.36}$$

Note that $||e_h||^2 = a^s(e_h, e_h)$.

The set of functions $\mathcal{S} \subset \tilde{X}_h$ is now introduced,

$$\mathcal{S} = \{ v \in \tilde{X}_h \mid a(w, \Theta_H + v) = \ell^N(w), \; \forall w \in X_h; \; b(w, q) = 0, \; \forall q \in Q_h \}. \tag{3.37}$$

It is observed that the second constraint enforces continuity and the homogeneous essential conditions, that is, $v \in X_h$, and the first constraint, from (3.28), then forces $v = e_h$. Therefore, the space $\mathcal{S}$ consists of a single function, $e_h$. The following trivial minimization statement can then be written as

$$\pm s_h = \min_{v \in \mathcal{S}} \left( a^s(v, v) - \ell^E(v) \pm \ell^O(\Theta_H + v) \right). \tag{3.38}$$

The constrained minimization problem (3.38) suggests the quadratic linear Lagrangian $\mathcal{L}: \tilde{X}_h \times X_h \times Q_h \to \mathbb{R}$, of which the constraints are the finite element equilibrium equations and the inter-subdomain continuity requirements.

$$\mathcal{L}^\pm(v, \mu, q) = \left( a^s(v, v) - \ell^E(v) \pm \ell^O(\Theta_H + v) \right) + \left[ a(\mu, \Theta_H + v) - \ell^N(\mu) \right] + b(v, q), \tag{3.39}$$

which allows to express $\pm s_h$ as

$$\pm s_h = \inf_{v \in \tilde{X}_h} \sup_{\mu \in X_h, q \in Q_h} \mathcal{L}^\pm(v, \mu, q). \tag{3.40}$$

Knowing that, at the saddle point $(\hat{e}_h, \psi_h^\pm, \lambda_h^\pm)$,

$$\hat{e}_h = \arg \min_{v \in \tilde{X}_h} \left[ \sup_{\mu \in X_h, q \in Q_h} \mathcal{L}^\pm(v, \mu, q) \right], \tag{3.41}$$

$$\psi_h^\pm, \lambda_h^\pm = \arg \max_{\mu \in X_h, q \in Q_h} \left[ \inf_{v \in \tilde{X}_h} \mathcal{L}^\pm(v, \mu, q) \right]. \tag{3.42}$$

Referring to $\mu$ as the adjoint (dual) variable and $q$ as the hybrid flux.
CHAPTER 3. A POSTERIORI ERROR ESTIMATION

3.1.7 Duality

From the quadratic-linear duality theory and equation (3.40) it follows that

\[ \pm s_h = \sup_{\mu \in X_h, \psi \in Q_h} \inf_{v \in X_h} \mathcal{L}^\pm(v, \mu, q) \geq \inf_{v \in X_h} \mathcal{L}^\pm(v, \tilde{\mu}_h^\pm, \tilde{q}_h^\pm), \quad \forall \tilde{\mu}_h^\pm \in X_h, \quad \forall \tilde{q}_h^\pm \in Q_h. \]  

(3.43)

with equality for \((\tilde{\mu}_h^\pm, \tilde{q}_h^\pm) = (\psi_h^\pm, \lambda_h^\pm)\). Upper and lower bounds for the output \(s_h, (s_h)^{LB}\) and \((s_h)^{UB}\), respectively, are now readily constructed as

\[ (s_h)^{LB} \equiv \inf_{v \in X_h} \mathcal{L}^+(v, \tilde{\mu}_h^+, \tilde{q}_h^+) \leq s_h \leq - \inf_{v \in X_h} \mathcal{L}^-(v, \tilde{\mu}_h^-, \tilde{q}_h^-) \equiv (s_h)^{UB}. \]  

(3.44)

where the superscript \((^+)\) denotes lower bounds and the superscript \((^-)\) indicates upper bounds. Note that equation (3.44) holds for any functions \(\tilde{\mu}_h^\pm \in X_h\) and \(\tilde{q}_h^\pm \in Q_h\).

3.2 Bound Procedure

Prior to calculating the bounds to the output of interest the Lagrange multiplier candidates: adjoint and hybrid flux are required. Therefore, the bound procedure involves three steps: the first step determines the adjoint; the second step determines the hybrid flux; and the third step determines the local \(h\)-mesh solution from which constructs the bounds. In three-dimensions the hybrid flux is computed between the faces of the elements.

3.2.1 \(H\)-mesh Adjoint Calculation

The problem of determining \(H\)-mesh approximations to the adjoint \(\psi_H^\pm, \mu_H^\pm\), and the hybrid flux \(\lambda_H^\pm, q_H^\pm\), that will yield sharp bounds is considered here. Towards this end, finding the saddle point of the Lagrangian (3.40) in the subspaces \(\hat{X}_H \subset X_h, X_H \subset X_h,\) and \(Q_H \subset Q_h\), reads

\[ (\psi_H^\pm, \lambda_H^\pm) = \arg \max_{\mu \in X_h, q \in Q_h} \left[ \min_{v \in \hat{X}_H} \mathcal{L}^\pm(v, \mu, q) \right]. \]  

(3.45)

followed by setting \(\tilde{\mu}_h^\pm = \psi_H^\pm\) and \(\tilde{q}_h^\pm = \lambda_H^\pm\).

From equation (3.39) the Lagrangian is expanded by putting \(v = e_H^\pm + w, \mu = \psi_H^\pm,\) and \(q = \lambda_H^\pm\)

\[ \mathcal{L}^\pm(e_H^\pm + w, \psi_H^\pm, \lambda_H^\pm) = \mathcal{L}^\pm(e_H^\pm, \psi_H^\pm, \lambda_H^\pm) + a^*(w, w) \]

\[ + 2a^*(w, e_H^\pm) - \ell^E(w) \pm \ell^O(w) \]

\[ + a(\psi_H^\pm, w) + b(w, \lambda_H^\pm). \]  

(3.46)
CHAPTER 3. A POSTERIORI ERROR ESTIMATION

Since the first variation of the Lagrange multiplier is equal to zero, the second and third line of equation (3.46) must vanish.

The stationarity conditions for equation (3.45) are obtained by requiring that the first variation of Lagrange multiplier with respect to $v$, $\mu$, and $q$ vanish:

Find $\tilde{\psi}_H^+ \in \hat{X}_H$, $\psi_H^+ \in X_H$, and $\lambda_H^+ \in Q_H$, such that

\[
\begin{align*}
    b(w, \lambda_H^+) &= -\left[2a^*(w, \tilde{\psi}_H^+) - \ell^E(w) \pm \ell^O(w) + a(\psi_H^+, w)\right], \quad \forall w \in \hat{X}_H. \\
    a(w, \Theta_H + \tilde{\psi}_H^+) &= \ell^N(w), \quad \forall w \in X_H, \\
    b(\tilde{\psi}_H^+, q) &= 0, \quad \forall q \in Q_H.
\end{align*}
\]

Equation (3.47) must be satisfied for all $w \in \hat{X}_H$, and thus, for all $w \in X_H \subset \hat{X}_H$. Expanding equation (3.48), yields

\[
a(w, \Theta_H) + a(w, \tilde{\psi}_H^+) = \ell^N(w), \quad \forall w \in X_H.
\]

Applying equation (3.26), then

\[
a(w, \tilde{\psi}_H^+) = 0, \quad \forall w \in X_H.
\]

Equation (3.51) implies that $\tilde{\psi}_H^+ = 0$. Also, from equation (3.26) knowing that

\[
\ell^E(w) = \ell^N(w) - a(w, \Theta_H) = 0, \quad \forall w \in X_H.
\]

By constraining $w$ to the continuous space $X_H$ the term $b(w, \lambda_H^+)$ is eliminated in (3.47) because $X_H$ is continuous and therefore has no inter-subdomain jumps. Finally equation (3.47) simplify as

\[
a(\psi_H^+, w) \pm \ell^O(w) = 0, \quad \forall w \in X_H.
\]

Note that, $\psi_H^+ = \pm \psi_H$ where $\psi_H$ can be calculated from

\[
a(\psi_H, w) + \ell^O(w) = 0, \quad \forall w \in X_H.
\]

3.2.2 $H$-mesh Hybrid Flux Calculation

To compute the hybrid flux, equation (3.47) is reformulated. The adjoint is known as $\psi_H^+ \in X_H$ and the problem becomes: Find $\lambda_H^+ \in Q_H$

\[
\begin{align*}
    2a^*(w, \psi_H^+) + b(w, \lambda_H^+) &= \ell^E(w) \mp \ell^O(w) - a(\psi_H^+, w), \quad \forall w \in \hat{X}_H. \\
    b(\psi_H^+, q) &= 0, \quad \forall q \in Q_H.
\end{align*}
\]
CHAPTER 3. A POSTERIORI ERROR ESTIMATION

To obtain this result, a "discontinuous subdomain" space is introduced to decouple the global system into local problems.

In this thesis I follow the FETI approach proposed in [18]. Details about the FETI procedure are discussed later in Section 3.3.3.

3.2.3 h-mesh Local Neumann Problem

Consider here the solution of the constrained minimization problems in (3.44). The minimizers of $L^\pm(v, \psi_h^\pm, \lambda_h^\pm)$, which is $\hat{e}_h^\pm \in \hat{X}_h$, will satisfy the following stationarity condition.

$$2a^s(w, \hat{e}_h^\pm) = \ell^E(w) \mp \ell^O(w) - a(\psi_h^\pm, w) - b(w, \lambda_h^\pm), \ \forall w \in \hat{X}_h,$$  \hspace{1cm} \text{(3.57)}

where $\psi_h^\pm$ is the interpolant of $\psi_h^\pm$ and $\lambda_h^\pm$ is the interpolant of $\lambda_h^\pm$.

3.2.4 Bound Calculations

In order to evaluate the lower and upper bounds, the computed values for $\hat{e}_h^\pm, \psi_h^\pm,$ and $\lambda_h^\pm$ are inserted into the Lagrangian (3.39).

$$(s_h)_{UB} \equiv -L^- (\hat{e}_h^-, \psi_h^-, \lambda_h^-) = -\eta^-.$$  \hspace{1cm} \text{(3.58)}

$$(s_h)_{LB} \equiv L^+ (\hat{e}_h^+, \psi_h^+, \lambda_h^+) = \eta^+.$$  \hspace{1cm} \text{(3.59)}

The lower and upper bounds are obtained from equations (3.39) and (3.44).

$$\eta^\pm = L^\pm (\hat{e}_h^\pm, \psi_h^\pm, \lambda_h^\pm)$$

$$= a^s(\hat{e}_h^\pm, \hat{e}_h^\pm) - \ell^E (\hat{e}_h^\pm) \pm \ell^O (\Theta_H + \hat{e}_h^\pm)$$

$$+ [a(\psi_h^\pm, \Theta_H + \hat{e}_h^\pm) - \ell^N (\psi_h^\pm)] + b(\hat{e}_h^\pm, \lambda_h^\pm).$$  \hspace{1cm} \text{(3.60)}

Expanding the above equation, gives

$$\eta^\pm = a^s(\hat{e}_h^\pm, \hat{e}_h^\pm) - \ell^E (\hat{e}_h^\pm) \pm \ell^O (\Theta_H) \pm \ell^O (\hat{e}_h^\pm)$$

$$+ a(\psi_h^\pm, \Theta_H) + a(\psi_h^\pm, \hat{e}_h^\pm) - \ell^N (\psi_h^\pm) + b(\hat{e}_h^\pm, \lambda_h^\pm).$$  \hspace{1cm} \text{(3.61)}

In particular, evoking equation (3.57) with $w = \hat{e}_h^\pm$, that is

$$b(\hat{e}_h^\pm, \lambda_h^\pm) = -2a^s(\hat{e}_h^\pm, \hat{e}_h^\pm) + \ell^E(\hat{e}_h^\pm) \mp \ell^O(\hat{e}_h^\pm) - a(\psi_h^\pm, \hat{e}_h^\pm), \ \forall w \in \hat{X}_h.$$  \hspace{1cm} \text{(3.62)}
and equation (3.26) with \( w = \psi^\pm_h \),

\[
a(\psi^\pm_h, \Theta_H) - \ell^N(\psi^\pm_h) = 0, \quad \forall w \in X_h, \tag{3.63}
\]

then finally get

\[
\eta^\pm = -a^s(\hat{e}_h^\pm, \hat{e}_h^\pm) \pm \ell^O(\Theta_H). \tag{3.64}
\]

Equations (3.58) and (3.59) now become,

\[
(s_h)_{UB} = -\eta^- = a^s(\hat{e}_h^-, \hat{e}_h^-) + \ell^O(\Theta_H), \tag{3.65}
\]

\[
(s_h)_{LB} = \eta^+ = -a^s(\hat{e}_h^+, \hat{e}_h^+) + \ell^O(\Theta_H), \tag{3.66}
\]

where the arguments of \((s_h)_{UB}\) and \((s_h)_{LB}\) indicate that the quality of bounds depends on the “coarse” working mesh \( T_H \). Equations (3.65) and (3.66) indicate that the bounds for \( s_h \) are also bounds for \( s_H \). Note that \( a^s(\hat{e}_h^-, \hat{e}_h^-) = ||\hat{e}_h^-||^2 \) and \( a^s(\hat{e}_h^+, \hat{e}_h^+) = ||\hat{e}_h^+||^2 \). Therefore I can calculate \( a^s(\hat{e}_h^-, \hat{e}_h^-) \) as well as \( a^s(\hat{e}_h^+, \hat{e}_h^+) \) and use them as an indication of the \( T_H \) error.

### 3.2.5 Bound Gap

I now introduce the (half) bound gap \( \Delta(T_H) \), which is represented as the sum of positive contributions of local indicators which are denoted as \( \Delta T_H(T_H) \).

\[
\Delta(T_H) \equiv \frac{1}{2} ((s_h)_{UB} - (s_h)_{LB}) = \frac{1}{2} \left[ a^s(\hat{e}_h^-, \hat{e}_h^-) + a^s(\hat{e}_h^+, \hat{e}_h^+) \right]. \tag{3.67}
\]

These bounds engender a predictor for \( s_h \)

\[
(s_h)_{pre}(T_H) = \frac{1}{2} [(s_h)_{LB}(T_H) + (s_h)_{UB}(T_H)]. \tag{3.68}
\]

the above equation can also be represented as

\[
(s_h)_{pre}(T_H) = \frac{1}{2} \left[ 2 \ell^O(\Theta_H) + a^s(\hat{e}_h^-, \hat{e}_h^-) - a^s(\hat{e}_h^+, \hat{e}_h^+) \right] = \ell^O(\Theta_H) + \frac{1}{2} \left[ a^s(\hat{e}_h^-, \hat{e}_h^-) - a^s(\hat{e}_h^+, \hat{e}_h^+) \right], \tag{3.69}
\]

for which

\[
|s_h - (s_h)_{pre}(T_H)| \leq \Delta(T_H). \tag{3.70}
\]

It follows that \(|s - (s_h)_{pre}| \) (respectively, \(|s - s_H| \)) is within \(|s - s_h| \) of \( \Delta \) (respectively, of \( \Delta + |(s_h)_{pre} - s_H| \)), and thus \( C_E \) is known to within the accuracy of the fine “truth” mesh.
Note that whereas $|s_h - s_H|$ may in fact be smaller than $|s_h - (s_h)_{\text{pre}}|$, the predictor $(s_h)_{\text{pre}}$ is preferred since the error bound on $(s_h)_{\text{pre}}$ will be sharper.

Finally, computational complexity is briefly summarized. On $\mathcal{T}_H$, two global calculations, i.e. one for $\Theta_H$ and one for $\psi_H$, and two local hybrid flux approximations are required. On $\mathcal{T}_h$, for each element, local symmetric Neumann operator need to be inverted for local upper and lower bounds calculation. It follows from the superlinearity of most general solution algorithms that the effort to compute the bounds will be much less than the effort to directly compute $\Theta_h$ and $s_h$. It was shown for some two dimensional problems, the computation time for the total calculations of the bound method was only 4% of the fine mesh calculations [5].

3.3 Algebraic Formulation

In this section, for the purpose of simplicity, I introduce the matrix form of the bounds formulation, which I described in the previous section. Since I already discuss the matrix form of coarse "working" mesh problem in Section 2.3, here I will present the bound procedure in a matrix notation.

The coarse "working" mesh problem in equation (3.26) is now expressed as

$$\hat{L}_H \hat{\Theta}_H = \hat{f}_H. \tag{3.71}$$

The output of interest becomes,

$$s_H = \ell_H^T \Theta_H, \tag{3.72}$$

where $\ell_H^T$ is transpose of $\ell_H$. The output $\ell_H^{(1)} = M_H$ 1 which $M_H$ is the finite element mass matrix of coarse "working" mesh and $\ell_H^{(2)} = M_H^{(2)}$ 1 which $M_H^{(2)}$ is the mass matrix for corner volume $\Omega^O$.

Similarly, the fine "truth" mesh problem in equation (3.28) becomes.

$$\hat{L}_h \hat{\Theta}_h = \hat{f}_h, \tag{3.73}$$

and the associated output of interest becomes,

$$s_h = \ell_h^T \Theta_h. \tag{3.74}$$

To avoid the expensive calculation of the system (3.73), a discontinuous space $\hat{X}_H$ with jumps across the elements $\Omega^k$ is introduced and bounds to $s_h$, i.e. $(s_h)_{LB} \leq s_h \leq (s_h)_{UB}$, are
calculated. The Lagrangian in equation (3.39) is now of the form
\[
\mathcal{L}^\pm(v, \mu, q) = \sum_{k=1}^{N_k} \left[ (v^{(k)})^T A^{(k)} v^{(k)} - (f^{(k)})^T v^{(k)} - v^{(k)}^T L^{(k)} u^{(k)}_H \right] \pm \ell^{(k)}^T \left( \Theta^{(k)}_H + v^{(k)} \right) \\
+ \sum_{k=1}^{N_k} \mu^{(k)} T \left[ L^{(k)} \left( \Theta^{(k)}_H + v^{(k)} \right) - f^{(k)} \right] + q^T \sum_{k=1}^{N_k} B^{(k)} v^{(k)},
\] (3.75)
where for \( N_k \) subdomains, \( v = \{ v^{(1)}, ..., v^{(N_k)} \} \) and \( \mu = \{ \mu^{(1)}, ..., \mu^{(N_k)} \} \). Also, \( A^{(k)} \) is the finite element discretization of the symmetric part of \( L^{(k)} \) and \( B^{(k)} \) is the sign Boolean matrix which localizes the "jumps" at the interface.

### 3.3.1 Adjoint Calculation

I consider here the \( H \)-mesh adjoint calculation. By evoking stationarity conditions of the Lagrangian (3.75), the following three equations are obtained.

Find \( e^\pm_H \in \bar{X}_H, \psi^\pm_H \in X_H \) and \( \lambda^\pm_H \in Q_H \), such that
\[
w^T B_H^T \lambda^\pm_H = - \left[ 2w^T A_H e^\pm_H - (f^T_H w - w^T L_H \Theta_H) \pm \ell^T_H w + \psi^\pm_H^T L_H w \right].
\] (3.76)
\[
w^T L_H \left( \Theta_H + e^\pm_H \right) = f^T_H w.
\] (3.77)
\[
e^\pm_H^T B_H q = 0.
\] (3.78)

Alternative forms of equations (3.76) - (3.78) are,
\[
B_H^T \lambda^\pm_H + 2A_H e^\pm_H - f_H + L_H \Theta_H + L_H^T \psi^\pm_H \pm \ell_H = 0,
\] (3.79)
\[
L_H \Theta_H + L_H e^\pm_H - f_H = 0,
\] (3.80)
\[
B_H^T e^\pm_H = 0.
\] (3.81)

Equation (3.81) forces \( e^\pm_H \in X_H \), which combined with equations (3.71) and (3.80) implies that \( e^\pm_H = 0 \). Since there are no inter-subdomain jumps for continuous spaces \( (X_H) \), then eliminating the term \( B_H \lambda^\pm_H \) in equation (3.79). Therefore, equation (3.79) is simplify as
\[
L_H^T \psi^\pm_H \pm \ell_H = 0.
\] (3.82)

Note that, \( \psi^\pm_H = \pm \psi_H \). Equation (3.82) is now reformulated as
\[
L_H^T \psi_H + \ell_H = 0.
\] (3.83)

Equation (3.83) shows that only one calculation for the adjoint is needed.
3.3.2 Hybrid Flux Calculation

Knowing the adjoint from the discrete equation (3.83), the hybrid flux can be computed using equation (3.79): Find $\lambda_H^\pm \in \mathcal{Q}_H$ such that

$$2 A_H^{(k)} e_H^{(k)} + B_H^{(k)T} \lambda_H^\pm = f_H^{(k)} - L_H^{(k)} \Theta_H^{(k)} \mp \epsilon_H^{(k)} - L_H^{(k)T} \psi_H^{\pm (k)}. \quad (3.84)$$

$$\sum_{k=1}^{N_k} B_H^{(k)} e_H^{(k)} = 0, \quad (3.85)$$

1 \leq k \leq N_k.

Note that, the term $B_H^{T} \lambda_H^\pm$ is non-zero in this case, since there exists inter-subdomain jumps for "discontinuous subdomain" space ($\hat{X}_H$). The solution to each equation of (3.84) and (3.85) only exist if the candidate Lagrange multipliers satisfy following condition.

$$\left( f_H^{(k)} - L_H^{(k)} \Theta_H^{(k)} \mp \epsilon_H^{(k)} - L_H^{(k)T} \psi_H^{\pm (k)} - B_H^{(k)T} \lambda_H^\pm \right) \perp \text{Ker} \left( A_H^{(k)} \right), \quad k = 1, \ldots, N_k. \quad (3.86)$$

It is important to stress that the Dirichlet boundary conditions are imposed through Lagrange multipliers. Therefore all subdomains are pure Neumann problems. This approach not only gives sharper bounds but also permits the same treatment of all sub-domains.

3.3.3 The FETI Procedure

The classical FETI approach is applied to calculate the inter-subdomain connectivity (i.e. hybrid flux) on the coarse "working" mesh $\mathcal{T}_H$. The right hand side of equation (3.84) is defined as

$$q_H^{(k)} = f_H^{(k)} - L_H^{(k)} \Theta_H^{(k)} \mp \epsilon_H^{(k)} - L_H^{(k)T} \psi_H^{\pm (k)}. \quad (3.87)$$

and also denoting $R_H^{(k)} = \text{Ker} \left( A_H^{(k)} \right)$ in equation (3.86). Then equations (3.84) and (3.85) are reformulated as

$$K_H^{(k)} e_H^{(k)} + B_H^{(k)T} \lambda_H^\pm = q_H^{(k)}, \quad (3.88)$$

$$\sum_{k=1}^{N_k} B_H^{(k)} e_H^{(k)} = 0, \quad (3.89)$$

1 \leq k \leq N_k,
where \( K_H^{(k)} = 2A_H^{(k)} \). Equation (3.88) is substituted into equation (3.89) and the set of amplitudes that specifies the contribution of the \( R_H^{(k)} \) to the solution, which denoted as \( \alpha_H^{(k)} \), is added
\[
\sum_{k=1}^{N_k} B_H^{(k)} e_H^{(k)}(k) = \sum_{k=1}^{N_k} B_H^{(k)} \left[ K_H^{(k)+} \left( q_H^{(k)} - B_H^{(k)T} \lambda_H^{\pm} \right) + R_H^{(k)} \alpha_H^{(k)} \right] = 0. \tag{3.90}
\]

These coefficients \( \alpha_H^{(k)} \) and \( R_H^{(k)} \) can be determined by requiring that each subdomain problem be mathematically solvable — i.e., each floating subdomain be self-equilibrated — which is
\[
\sum_{k=1}^{N_k} R_H^{(k)T} \left( q_H^{(k)} - B_H^{(k)T} \lambda_H^{\pm} \right) = 0. \tag{3.91}
\]

Rearranging equation (3.90) and (3.91) gives
\[
B_H^{(k)} K_H^{(k)+} B_H^{(k)T} \lambda_H^{\pm} - B_H^{(k)} R_H^{(k)} \alpha_H^{(k)} = B_H^{(k)} K_H^{(k)+} q_H^{(k)}. \tag{3.92}
\]
\[
-\frac{R_H^{(k)T} B_H^{(k)T} \lambda_H^{\pm}}{k} = -\frac{R_H^{(k)T} q_H^{(k)}}{k}. \tag{3.93}
\]

1 \( \leq k \leq N_k \).

The FETI interface problem, i.e. equation (3.92) and (3.93), is presented in matrix form:
\[
\begin{bmatrix}
F_H & -G_H \\
-G_H^T & 0
\end{bmatrix}
\begin{bmatrix}
\lambda_H \\
\alpha_H
\end{bmatrix}
= \begin{bmatrix}
d_H \\
e_H
\end{bmatrix}. \tag{3.94}
\]

where each of these terms is given by
\[
F_H = \sum_{k=1}^{N_k} B_H^{(k)} K_H^{(k)+} B_H^{(k)T}, \tag{3.95}
\]
\[
G_H = \begin{bmatrix}
B_H^{(1)} R_H^{(1)} & \ldots & B_H^{(N_k)} R_H^{(N_k)}
\end{bmatrix}, \tag{3.96}
\]
\[
\lambda_H = \lambda_H^{\pm}, \tag{3.97}
\]
\[
\alpha_H = \begin{bmatrix}
\alpha_H^{(1)} & \ldots & \alpha_H^{(N_k)}
\end{bmatrix}, \tag{3.98}
\]
\[
d_H = \sum_{k=1}^{N_k} B_H^{(k)} K_H^{(k)+} q_H^{(k)}, \tag{3.99}
\]
\[
e_H = \begin{bmatrix}
B_H^{(1)T} q_H^{(1)} & \ldots & B_H^{(N_k)T} q_H^{(N_k)}
\end{bmatrix}. \tag{3.100}
\]

where \( K_H^{(k)+} \) is a generalized inverse of \( K_H^{(k)} \). The constraint \( G_H^T \lambda_H = e_H \) ensures that
\[
\left( f_H^{(k)} - L_H^{(k)} \Theta_H^{(k)} + \psi_H^{(k)} T \psi_H^{(k)} \right) \in \text{range } A_H^{(k)} \text{ for all } k = 1, \ldots, N_k.
The FETI method iterates on $\lambda_H$, given an initial $\lambda_H^0$ which satisfies the constraint $G_H^T \lambda_H^0 = e_H$. Subsequent increments of the inter-subdomain connectivity satisfy followings.

$$P_H F_H \Delta \lambda_H = P_H d_H,$$

$$G_H^T \Delta \lambda_H = 0,$$

where $P_H$ is an orthogonal projector onto $\text{Ker}(G_H^T)$ defined by

$$P_H = I_H - G_H(G_H^T G_H)^{-1} G_H^T.$$

The FETI algorithm can be regarded as a two-step pre-conditioned conjugate gradient method to solve the interface problem and can be summarized as in [18, 7]:

1. Initialize

$$\hat{\lambda}_H^0 = G_H(G_H^T G_H)^{-1} e_H$$

$$w^0 = P^T (d_H - F_H \hat{\lambda}_H^0)$$

2. Iterate $n = 0, 1, \ldots$ until convergence

$$y^n = P_H \hat{F}_H^{-1} w^n$$

$$p^n = y^n - \sum_{i=0}^{n-1} \frac{y^{nT} F_H p^i}{p^{iT} F_H p^i} p^i$$

$$\eta^n = \frac{y^{nT} w^n}{p^{iT} F_H p^n}$$

$$\hat{\lambda}_H^{n+1} = \hat{\lambda}_H^n + \eta^n p^n$$

$$w^{n+1} = w^n - \eta^n P_H^T F_H p^n$$

### 3.3.4 Fine “truth” mesh Local Neumann Problem

The coarse “working” mesh adjoint ($\psi_H$), hybrid flux ($\lambda_H$), and field variable ($\Theta_H$) are linearly interpolated onto the fine “truth” mesh, i.e., $\psi_H \rightarrow \psi_h$, $\lambda_H \rightarrow \lambda_h$, and $\Theta_H \rightarrow \hat{\Theta}_h$. The minimizers of $\mathcal{L}^\pm(v, \psi_h^\pm, \lambda_h^\pm)$, which is $\hat{e}_h^\pm \in \hat{X}_h$, will satisfy the following equation.

$$2 A_h^{(k)}(\hat{e}_h^\pm)^{(k)} = f_h^{(k)} - I_h^{(k)} \psi_h^{(k)} \mp \hat{e}_h^{(k)} - L_h^{(k)} T \psi_h^{\pm(k)} - B_h^{(k)} T \lambda_h^\pm,$$

$$k = 1, \ldots, N_k.$$
3.3.5 Bound Calculations

The lower and upper bounds can now be evaluated by substituting the \( v = \hat{e}_h^\pm \), \( \mu = \psi_h^\pm \), and \( q = \lambda_h^\pm \) into the Lagrangian (3.75), i.e.

\[
\eta^\pm = \mathcal{L}^\pm(\hat{e}_h^\pm, \psi_h^\pm, \lambda_h^\pm).
\]  

Hence, the lower and upper bounds become,

\[
(s_h)_{UB} = -\eta^- = s_H + \sum_{k=1}^{N_k} (\hat{e}_h^-)^{(k)} A_h^{(k)} (\hat{e}_h^-)^{(k)},
\]

\[
(s_h)_{LB} = \eta^+ = s_H - \sum_{k=1}^{N_k} (\hat{e}_h^+)^{(k)} A_h^{(k)} (\hat{e}_h^+)^{(k)}.
\]

3.4 Numerical Validation

Bounds of the output of the convection–diffusion equation are investigated for a cube where \( U_l = (1,1,1) \), \( f = 1 \), and \( \varepsilon = 0.2,0.1,0.05 \). Two outputs are considered: the first output \( s^{(1)} \) is the average solution of the entire domain on the fine discretization \( \mathcal{T}_h \) and the second output \( s^{(2)} \) is the average solution in corner cube \( \Omega^O \), which is \( 2/3,1[\times]2/3,1[\times]2/3,1[\times]2/3,1[\times]2/3,1[\times]2/3,1 \) on the fine discretization \( \mathcal{T}_h \). Computational results are presented here to illustrate the accuracy of bound method. To begin, a calculation for both bounds and a conventional finite element approach with a LASPack sparse solver [24] implementation is performed. Furthermore, all the computations are carried on a single with processor Pentium III 933 MHz CPU having 1536M of memory.

3.4.1 Convergence Rate of the Average Solution Output \( s^{(1)} \)

The influence of the number of subdomains on the sharpness of the bounds is analyzed for the average solution output \( s^{(1)} \). Five different coarse subdivisions are considered: i.e. \( \mathcal{T}_{(H,5)} \), \( \mathcal{T}_{(H,6)} \), \( \mathcal{T}_{(H,10)} \), \( \mathcal{T}_{(H,12)} \), and \( \mathcal{T}_{(H,15)} \). The following notation, \( \mathcal{T}_{(H,N)} \), is utilized to identify the coarse discretizations where \( N \) is the number of subdomains per edge of cube, i.e. \( N \times N \times N \times N \times N \times N \times 6 \) subdomains in the entire computational domain. The information in Table 3.1 reports the number of elements and the degrees-of-freedom associated with each refinement \( (R) \) investigated. The fine “truth” mesh \( (\mathcal{T}_h) \) for first output consists of 226,981 degrees-of-freedom (nodes) and 1,296,000 tetrahedral elements.
CHAPTER 3. A POSTERIORI ERROR ESTIMATION

<table>
<thead>
<tr>
<th>Coarse mesh</th>
<th>Refinement $(R)$</th>
<th>Number of subdomains</th>
<th>Degrees of freedom per subdomain</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{T}_{(H,5)}$</td>
<td>12</td>
<td>750</td>
<td>457</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,6)}$</td>
<td>10</td>
<td>1,296</td>
<td>288</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,10)}$</td>
<td>6</td>
<td>6,000</td>
<td>85</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,12)}$</td>
<td>5</td>
<td>10,368</td>
<td>57</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,15)}$</td>
<td>4</td>
<td>20,250</td>
<td>36</td>
</tr>
</tbody>
</table>

Table 3.1: Number of subdomains and refinement subdivisions for the output $s^{(1)}$.

<table>
<thead>
<tr>
<th>Coarse mesh</th>
<th>Refinement $(R)$</th>
<th>$\varepsilon = 0.2$</th>
<th>$\varepsilon = 0.1$</th>
<th>$\varepsilon = 0.05$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{T}_{(H,5)}$</td>
<td>12</td>
<td>0.08142 ± 0.0210</td>
<td>0.12959 ± 0.0609</td>
<td>0.18080 ± 0.2093</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,6)}$</td>
<td>10</td>
<td>0.08258 ± 0.0158</td>
<td>0.13025 ± 0.0462</td>
<td>0.17895 ± 0.1594</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,10)}$</td>
<td>6</td>
<td>0.08447 ± 0.0066</td>
<td>0.13151 ± 0.0194</td>
<td>0.17669 ± 0.0671</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,12)}$</td>
<td>5</td>
<td>0.08479 ± 0.0047</td>
<td>0.13174 ± 0.0139</td>
<td>0.17632 ± 0.0479</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,15)}$</td>
<td>4</td>
<td>0.08506 ± 0.0031</td>
<td>0.13190 ± 0.0091</td>
<td>0.17598 ± 0.0309</td>
</tr>
</tbody>
</table>

Table 3.2: Output predictor $(s_h)_{pre}$ and half bound gap $\Delta(\mathcal{T}_H)$ for the output $s^{(1)}$.

<table>
<thead>
<tr>
<th>Coarse mesh</th>
<th>Refinement $(R)$</th>
<th>$\varepsilon = 0.2$</th>
<th>$\varepsilon = 0.1$</th>
<th>$\varepsilon = 0.05$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\log</td>
<td>(s_h)_{UB} - s_h</td>
<td>/log H$</td>
<td>1.77</td>
<td>1.88</td>
</tr>
<tr>
<td>$\log</td>
<td>s_H - s_h</td>
<td>/log H$</td>
<td>2.03</td>
<td>2.03</td>
</tr>
<tr>
<td>$\log</td>
<td>(s_h)_{LB} - s_h</td>
<td>/log H$</td>
<td>1.98</td>
<td>1.96</td>
</tr>
</tbody>
</table>

Table 3.3: Convergence slope of bounds values for the output $s^{(1)}$. 

---
In order to analyze the sharpness of the bounds (i.e. upper and lower bounds), the values of the bounds and their convergence for different coarse "working" meshes are reported in Table 3.2, Table 3.3, and Figure 3.1 to Figure 3.3 for the output \( s^{(1)} \). Clearly, a finer coarse mesh provides sharper lower and upper bounds but at a higher computational cost. In Table 3.2, the output predictor, \((s_h)_{\text{pre}}\) and the half bound gap, \(\Delta(T_H)\) are summarized for different thermal diffusivity and in Table 3.3, the convergence slope is reported for different thermal diffusivity (i.e. \( \epsilon = 0.2, 0.1, \) and 0.05), respectively.

Figure 3.1a shows \((s_h)_{UB}/s_h\), \((s_h)_{LB}/s_h\), \(s_H/s_h\) and \((s_h)_{\text{pre}}/s_h\) as a function of \( H \) and Figure 3.1b presents \(|(s_h)_{UB} - s_h|\), \(|(s_h)_{LB} - s_h|\), \(|s_H - s_h|\) and \(|(s_h)_{\text{pre}} - s_h|\) as a function of \( H \) in log-log scale for the output \( s^{(1)} \) when the thermal diffusivity is 0.2 (\( \epsilon = 0.2 \)). Similarly, Figure 3.2 presents for \( \epsilon = 0.1 \) and Figure 3.3 illustrates for \( \epsilon = 0.05 \), respectively. It is observed that the bounds are quite accurate and the average output of bounds \((s_h)_{\text{pre}}\) is the most accurate estimator of the fine "truth" mesh output \( s_h \). The output predictor \((s_h)_{\text{pre}}\) is from 17.1% to 2.3% when \( \epsilon = 0.2 \), from 17.6% to 2.3% when \( \epsilon = 0.1 \), and from 20% up to 2.6% when \( \epsilon = 0.05 \) more accurate than the coarse "working" mesh output \( s_H \).

The relative (half) bound gap \( \theta \), which is defined as the bound gap \( \Delta(T_H) \) over the output predictor \((s_h)_{\text{pre}}\), is decreased from 25.8% to 3.7% (\( \epsilon = 0.2 \)), from 47% to 6.9% (\( \epsilon = 0.1 \)), and from 115.7% to 17.6% (\( \epsilon = 0.05 \)). The bound gap increases whenever the thermal diffusivity (\( \epsilon \)) is decreased. This indicated that a more convection dominant problem requires a finer coarse mesh to obtain the same bound gap. From Table 3.2 and Figure 3.1b, 3.2b, and 3.3b, the bounds converge to \( O(H^2) \) as \( H \to h \), which agrees with theory [16]. Note three exceptions which are: the upper bound converges to \( O(H^{1.8}) \) for \( \epsilon = 0.2 \) and \( O(H^{1.9}) \) for \( \epsilon = 0.1, 0.05 \), respectively.
Figure 3.1: (a) Plots of \((s_h)_{UB}/s_h\), \((s_h)_{LB}/s_h\), \(s_H/s_h\) and \((s_h)_{pre}/s_h\) as a function of \(H\) for the output \(s^{(1)}\) when \(\varepsilon = 0.2\) (top), (b) Plots of \(|(s_h)_{UB} - s_h|\), \(|(s_h)_{LB} - s_h|\), \(|s_H - s_h|\), and \(|(s_h)_{pre} - s_h|\) as a function of \(H\) in log-log scale for the output \(s^{(1)}\) when \(\varepsilon = 0.2\) (bottom).
CHAPTER 3. A POSTERIORI ERROR ESTIMATION

Figure 3.2: (a) Plots of $(s_h)UB/s_h$, $(s_h)LB/s_h$, $s_H/s_h$ and $(s_h)prec/s_h$ as a function of $H$ for the output $s^{(1)}$ when $\varepsilon = 0.1$ (top), (b) Plots of $|(s_h)UB - s_h|$, $|(s_h)LB - s_h|$, $|s_H - s_h|$, and $|(s_h)prec - s_h|$ as a function of $H$ in log-log scale for the output $s^{(1)}$ when $\varepsilon = 0.1$ (bottom).

CHAPTER 3. A POSTERIORI ERROR ESTIMATION

Figure 3.3: (a) Plots of \((s_h)_{UB}/s_h\), \((s_h)_{LB}/s_h\), \(s_H/s_h\), and \((s_h)_{pre}/s_h\) as a function of \(H\) for the output \(s^{(1)}\) when \(\varepsilon = 0.05\) (top), (b) Plots of \(|(s_h)_{UB} - s_h|\), \(|(s_h)_{LB} - s_h|\), \(|s_H - s_h|\), and \(|(s_h)_{pre} - s_h|\) as a function of \(H\) in log-log scale for the output \(s^{(1)}\) when \(\varepsilon = 0.05\) (bottom).
Table 3.4: Number of subdomains and refinement subdivisions for the output $s^{(2)}$.

<table>
<thead>
<tr>
<th>Coarse mesh</th>
<th>Refinement $(R)$</th>
<th>Number of subdomains</th>
<th>Degrees of freedom per subdomain</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{T}_{(H,6)}$</td>
<td>6</td>
<td>1,296</td>
<td>85</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,9)}$</td>
<td>4</td>
<td>4,374</td>
<td>36</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,12)}$</td>
<td>3</td>
<td>10,368</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 3.5: Output predictor $(s_h)_\text{pre}$ and half bound gap $\Delta(\mathcal{T}_H)$ for the output $s^{(2)}$.

<table>
<thead>
<tr>
<th>Coarse mesh</th>
<th>Refinement $(R)$</th>
<th>$\varepsilon = 0.2$</th>
<th>$\varepsilon = 0.1$</th>
<th>$\varepsilon = 0.05$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{T}_{(H,6)}$</td>
<td>6</td>
<td>0.08048 $\pm$ 0.0963</td>
<td>0.18677 $\pm$ 0.1788</td>
<td>0.34888 $\pm$ 0.3750</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,9)}$</td>
<td>4</td>
<td>0.08730 $\pm$ 0.0486</td>
<td>0.19875 $\pm$ 0.0866</td>
<td>0.35956 $\pm$ 0.1742</td>
</tr>
<tr>
<td>$\mathcal{T}_{(H,12)}$</td>
<td>3</td>
<td>0.08980 $\pm$ 0.0279</td>
<td>0.20297 $\pm$ 0.0487</td>
<td>0.36300 $\pm$ 0.0953</td>
</tr>
</tbody>
</table>

3.4.2 Convergence Rate of Average Corner Solution Output $s^{(2)}$

Similarly, the average corner solution output $s^{(2)}$ is investigated by using three different coarse subdivisions: i.e. $\mathcal{T}_{(H,6)}$, $\mathcal{T}_{(H,9)}$, and $\mathcal{T}_{(H,12)}$. The information in Table 3.3 reports the number of elements and the degrees-of-freedom constructed each refinement $(R)$. The fine “truth” mesh $(\mathcal{T}_h)$ for second output consists of 50,653 degrees-of-freedom (nodes) and 279,936 tetrahedral elements.

To analyze the sharpness of the bounds (i.e. upper and lower bounds) for average corner solution output $s^{(2)}$, the values of the bounds and their convergence for different coarse “working” meshes are reported in Table 3.5, Table 3.6, and Figure 3.4 to Figure 3.6. In Table 3.5 the output predictor, $(s_h)_\text{pre}$ and the half bound gap, $\Delta(\mathcal{T}_H)$ are summarized for each thermal

<table>
<thead>
<tr>
<th></th>
<th>$\varepsilon = 0.2$</th>
<th>$\varepsilon = 0.1$</th>
<th>$\varepsilon = 0.05$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\log[(s_h)_UB - s_h]/\log H$</td>
<td>1.90</td>
<td>1.98</td>
<td>2.09</td>
</tr>
<tr>
<td>$\log[s_h - (s_h)_L]/\log H$</td>
<td>2.02</td>
<td>2.05</td>
<td>2.03</td>
</tr>
<tr>
<td>$\log[(s_h)_LB - s_h]/\log H$</td>
<td>1.96</td>
<td>2.02</td>
<td>2.10</td>
</tr>
</tbody>
</table>

Table 3.6: Convergence slope of bounds values for the output $s^{(2)}$. 
diffusivity and in Table 3.6 the convergence slope is reported for different thermal diffusivity (i.e. \( \varepsilon = 0.2, 0.1, \) and 0.05), respectively.

Figure 3.4a shows that \((s_h)_{UB}/s_h, (s_h)_{LB}/s_h, s_H/s_h\) and \((s_h)_{pre}/s_h\) as a function of \(H\) and Figure 3.4b presents \(|(s_h)_{UB} - s_h|, |(s_h)_{LB} - s_h|, |s_H - s_h|, \) and \(|(s_h)_{pre} - s_h|\) as a function of \(H\) in log-log scale for the output \(s^{(2)}\) when the thermal diffusivity is 0.2 (\(\varepsilon = 0.2\)). Similarly, Figure 3.5 and Figure 3.6 present the same graphs for \(\varepsilon = 0.1\) and \(\varepsilon = 0.05\), respectively. The average output of bounds \((s_h)_{pre}\) is more accurate than the coarse “working” mesh output \(s_H\) from 23.2% to 6% (\(\varepsilon = 0.2\)), from 21% to 5.4% (\(\varepsilon = 0.1\)), and from 15.8% up to 4.6% (\(\varepsilon = 0.05\)).

When \(\varepsilon = 0.2\), the relative (half) bound gap \(\theta\), which is defined as the bound gap \(\Delta(T_H)\) over the output predictor \((s_h)_{pre}\), is decreased from 119.7% to 31%, for \(\varepsilon = 0.1\) the relative bound gap is changed from 95.7% to 23.97%, and for the case of \(\varepsilon = 0.05\) the relative bound gap is reduced from 107.5% to 26.3%. Unlike the average solution output \(s^{(1)}\), the bound gap do not depend on the thermal diffusivity \(\varepsilon\). Note that, the bound gap is rather higher than that of the output \(s^{(1)}\). As it is noticed from Table 3.6 and Figure 3.4b, 3.5b, and 3.6b, only except the upper bound which converges to \(O(H^{1.9})\) (\(\varepsilon = 0.2\)), the bounds converge to \(O(H^2)\) as \(H \to h\), which again conforms with theory [16]. If hybrid flux calculations are performed on fine “truth” \(h\)-mesh, not only they improve the accuracy of convergence rate but also reduce the relative (half) bound gap. The calculation of hybrid flux directly on the fine \(h\)-mesh is discussed in Chapter 6.

While the bound gap is within respectable values, it is worth while to appreciate that the bound method always guarantees that the output of interest is within these values. Note that the output predictor, \((s_h)_{pre}\), offers a better prediction of the output than the coarse mesh output, \(s_H\), while the bounds themselves guarantee that the output is within \((s_h)_{UB}\) and \((s_h)_{LB}\).

Uniform subdivisions of the mesh are exploited for the convergence rate but it is not an optimum method for reducing bounds. The improvement of the sharpness of bounds and uniform distribution of the error can be achieved by changing the mesh based on an error estimator. The adaptive mesh refinement technique will be introduced in next chapter.
Figure 3.4: (a) Plots of \((s_h)_{UB}/s_h\), \((s_h)_{LB}/s_h\), \(s_H/s_h\) and \((s_h)_{pre}/s_h\) as a function of \(H\) for the output \(s^{(2)}\) when \(\varepsilon = 0.2\) (top), (b) Plots of \(|(s_h)_{UB} - s_h|\), \(|(s_h)_{LB} - s_h|\), \(|s_H - s_h|\), and \(|(s_h)_{pre} - s_h|\) as a function of \(H\) in log-log scale for the output \(s^{(2)}\) when \(\varepsilon = 0.2\) (bottom).
Figure 3.5: (a) Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ as a function of $H$ for the output $s^{(2)}$ when $\varepsilon = 0.1$ (top), (b) Plots of $|(s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{pre} - s_h|$ as a function of $H$ in log-log scale for the output $s^{(2)}$ when $\varepsilon = 0.1$ (bottom).
Figure 3.6: (a) Plots of \((s_h)_{UB}/s_h\), \((s_h)_{LB}/s_h\), \(s_H/s_h\) and \((s_h)_{pre}/s_h\) as a function of \(H\) for the output \(s^{(2)}\) when \(\epsilon = 0.05\) (top), (b) Plots of \(|(s_h)_{UB} - s_h|\), \(|(s_h)_{LB} - s_h|\), \(|s_H - s_h|\), and \(|(s_h)_{pre} - s_h|\) as a function of \(H\) in log-log scale for the output \(s^{(2)}\) when \(\epsilon = 0.05\) (bottom).
Chapter 4

Adaptive Mesh Refinement

An engineer may be interested in bounds with a specific bound gap and therefore improving the sharpness of bounds or bound gap is a major issue in the application of this method. One approach to improve the bounds is adapting the coarse mesh. In this section, the procedure of adaptive mesh refinement is presented. This technique is validated on a simple test case.

4.1 Local Indicators

The adaptive mesh refinement procedure can be implemented on any pair of meshes, $\mathcal{T}_H$ and $\mathcal{T}_h$, which satisfy the requirement that $\mathcal{T}_h$ is a refinement of $\mathcal{T}_H$. Since the cost of computing the bounds is essentially a function of number of elements $K_H$ in $\mathcal{T}_H$, it is desirable to construct optimized tetrahedra that maximize the bound accuracy (minimize the bound gaps) for the minimum number of elements $\mathcal{T}_H$. In this section an adaptive algorithm for generating such optimized grids proposed in [21, 10] is extended to three dimensional spaces.

Note that the bound gap (3.67) can be expressed as a sum of elemental contributions, i.e.

$$\Delta (\mathcal{T}_H) = \sum_{T_H \in \mathcal{T}_H} \Delta_{T_H} (\mathcal{T}_H), \quad (4.1)$$

where

$$\Delta_{T_H} (\mathcal{T}_H) = \frac{1}{2} \left[ a_{T_H}^+(\hat{e}_h^-, \hat{e}_h^-) + a_{T_H}^-(\hat{e}_h^+, \hat{e}_h^+) \right]. \quad (4.2)$$

Note that $\Delta_{T_H} (\mathcal{T}_H)$ is non-negative and can thus be directly interpreted as the contribution to the bound gap from element $T_H$.

If the optimal stabilization parameter $\kappa^*$ (developed in Chapter 5) is used in the bound
calculation, the local indicators should be formulated as

\[ \Delta(T_H) = \sum_{T_H \in T_H} \Delta(T_H, \kappa^*), \]

where

\[ \Delta(T_H, \kappa^*) = \frac{1}{\kappa^*} a_{T_H}^q (\hat{\epsilon}_{0H}, \hat{\epsilon}_{0H}) + \kappa^* a_{T_H}^q (\hat{\epsilon}_{1H}, \hat{\epsilon}_{1H}), \]

\[ = \sqrt{\frac{a^q (\hat{\epsilon}_{0H}, \hat{\epsilon}_{0H}) a_{T_H}^q (\hat{\epsilon}_{0H}, \hat{\epsilon}_{0H}) + \sqrt{a^q (\hat{\epsilon}_{1H}, \hat{\epsilon}_{1H}) a_{T_H}^q (\hat{\epsilon}_{1H}, \hat{\epsilon}_{1H})}}. \]

The detail information about the optimal stabilization parameter \( \kappa^* \) will be explained in Chapter 5 of this thesis.

Now I can describe the adaptive mesh refinement strategy suggested in [21, 10]. Starting from an initial grid \( T_H^n \) is a corresponding bound gaps \( \Delta(T_H^n), n = 1, 2, 3, \ldots \) such that each tetrahedron \( T_H^n \) is a refinement of the preceding tetrahedra \( T_{H-1}^n \). This approach does not guarantee that \( T_H^n \leq T_{H-1}^n \) for any particular \( n \), but it does ensure that, for a sufficiently large \( n \), \( T_H^n \leq \Delta_{\text{target}} \), where \( \Delta_{\text{target}} > 0 \) is a specified positive gap target.

In order to identify the elements in \( T_H^{n-1} \) that need to be refined, first the largest elemental contribution \( \Delta_{\text{max}}^{n-1} \) to the bound gap \( \Delta(T_H^{n-1}) \) can be calculated as

\[ \Delta_{\text{max}}^{n-1} = \max_{T_H \in T_H^{n-1}} \Delta(T_H^{n-1}), \]

and then all elements \( T_H \in T_H^{n-1} \) for which

\[ \Delta(T_H^{n-1}) \geq \alpha \Delta_{\text{max}}^{n-1}, \]

can be selected for refinement. \( \alpha \) is a parameter which controls the fraction of elements to be refined at each adaptive cycle \( 0 < \alpha < 1 \). At present, this parameter is specified \textit{a priori}, and is independent of \( n \). More sophisticated strategies for choosing \( \alpha \) can, of course, be devised; for instance, I can determine \( \alpha^n \) such that, based on a \textit{priori} scalings, a given target gap \( \Delta_{\text{target}}^{n} \) should be satisfied by the refined grid \( T_H^n \).

The adaptive mesh refinement process is terminated when the accuracy goal is achieved. In the applications that follow, the accuracy measure is the relative (half) bound gap \( \theta^n \), given by

\[ \theta^n = \frac{\Delta(T_H^n)}{(s_H)_{\text{pre}}(T_H^n)}. \]
The adaptive process is thus halted when $\theta^n \leq \theta^{obj}$ where $\theta^{obj}$ is the prescribed accuracy. The adaptive process is also stopped whenever reaching a maximum number of iterations.

4.2 Mesh Refinement

4.2.1 Spatial Refinement

Knowing the elements which are locally flagged for refinement, the next step is to refine the mesh by adding more nodes. The method, which is simple and efficient, involves subdivision of elements based on a set of primitive geometric operations. Meshes using triangles and tetrahedra are well suited for such a refinement. A major advantage of this refinement is that new elements are fully nested within their forming "coarse" elements [23].

The refinement procedure based on dividing tetrahedral elements into 2, 4, or 8 elements is implemented herein. There are two main reasons for implementing this technique. First, in term of high computational cost, using fine "coarse" mesh everywhere is very expensive for the large and complex three-dimensional simulations. Second, the nested property of refinement allows all the element history information to be easily stored so that it can be extended to mesh coarsening. However, the mesh coarsening is not applied in this thesis. There are two main rules to be followed during the refinement procedure:

1. Excessively distorted or stretched elements should be avoided. The generation of too many large aspect ratio elements causes ill-conditioned matrices, which adversely affects the accuracy and stability [23, 25]. Hence, it is crucial that the shape of each element should not be deteriorated excessively.

2. The refined mesh should be conforming. A conforming mesh is defined as one in which the union of any two elements is a face, an edge, a node, or empty [23, 13]. To meet the conforming criterion, all the elements should be subdivided to share either a face or an edge with a previous element.

4.2.2 8-Subtetrahedron Subdivision Refinement

The refinement method involves subdividing a tetrahedron by eight sections, which is resulting in eight subdivided elements [23, 13, 9]. However, this method alone cannot guarantee a locally adapted conforming mesh and causes various kinds of subdivision patterns. Although
this method is far more difficult to implement and to analyze, this method offers fast refinement since the element volume changes by a factor of eight. Therefore, this refinement method is more suitable for mesh adaptation intended for complex three-dimensional simulations.

The 8-subtetrahedron subdivision refinement procedure \cite{23, 13} begins with \( \text{Sub}_8 \) refinement of the element, illustrated in Figure 4.1. Every edge of a tetrahedron \( T_H \) is bisected into two sub-edges and every face is bisected into the sub-faces by introducing three new edges that connect the nodes bisecting the original edges. All new edges are then connected to make the new faces interior to the original tetrahedron. This procedure forms the corner elements and one interior octahedron, then divided into the inner elements by inserting a mid-edge node on one of its the diagonals. The shortest diagonals is chosen to ensure that the inner elements have the best aspect ratio \cite{23, 9}. 8-Subtetrahedra are \( T_1(t_0, t_{01}, t_{02}, t_{03}) \), \( T_2(t_{01}, t_1, t_{12}, t_{13}) \), \( T_3(t_{02}, t_{12}, t_2, t_{23}) \), \( T_4(t_{03}, t_{13}, t_{23}, t_3) \), \( T_5(t_{01}, t_{13}, t_{03}, t_{02}) \), \( T_6(t_{01}, t_{12}, t_{13}, t_{02}) \), \( T_7(t_{23}, t_{02}, t_{12}, t_{13}) \), and \( T_8(t_{23}, t_{03}, t_{02}, t_{13}) \). After refinement the mesh is no longer conforming, since it contains a number of elements that are left with "hanging" nodes on some or all of their six edges.

### 4.2.3 Treatment of "hanging" Nodes

A unified procedure for tetrahedral mesh refinement can deal with all possible "hanging" node configurations proposed in \cite{23, 13}.

The unique patterns of sub-division can occur from the eight sections refinement (8-subtetrahedron subdivision refinement). These are illustrated in Figure 4.2 with the patterns...
denoted as "Sub_2", "Sub_4", "Sub_6", and "Sub_8". A tetrahedron with only one "hanging" node is subdivided using "Sub_2" refinement; a tetrahedron with three "hanging" nodes on the same face is subdivided using "Sub_4" refinement; a tetrahedron with two "hanging" nodes on a pair of opposite edges is subdivided using "Sub_6" refinement; and a tetrahedron with six "hanging" nodes on each edge is subdivided using "Sub_8" refinement. The "Sub_2", "Sub_4", "Sub_6", and "Sub_8" refinement procedures are known as "directional" refinement. Elements resulting from directional refinement are called as "transition" elements and must not be directionally refined more than once, since doing so can generate excessively stretched elements. To avoid undue stretching, parent should be recovered and then subdivided using "Sub_8". The detail 8-tetrahedron subdivision refinement algorithm is outlined in [23, 13]. This algorithm always guarantees a conforming mesh.
4.3 Adaptivity Procedure

The adaptive mesh refinement procedure is presented in this section. Here are the four steps required for the entire procedure:

1. Run the bound calculation code using an initial "coarse" working mesh $T^0_H$ with corresponding bound gap $\Delta(T^0_H)$. Then calculate the largest elemental contribution $\Delta^0_{\text{max}}$ to the bound gap $\Delta(T^0_H)$ and then tag elements which are needed to be refined based on equation (4.7). There are two ways to set the value of $\alpha$; First, $\alpha$ can be fixed ($0 < \alpha < 1$) for all the refinements. In this case, the number of elements for each refinement cannot be predicted 'priori'. Second, $\alpha$ can be varied for each refinement provided that the number of tagged elements is fixed.

2. Since the mesh refinement code is developed for 'quadratic' tetrahedral elements, one mid-node on each edge of every element is added. Then the mesh is refined based on the tagged elements from step 1. Although the 'linear' tetrahedral element is used for the present study, I will increase the order of shape function in the tetrahedral element for future study. If the 'quadratic' tetrahedron is used for the bound calculation, it is not necessary to add the mid-node in each edge of every element.

3. Change the 'quadratic' tetrahedral element to the 'linear' tetrahedral element by removing the mid-node on each edge of every element. If the 'quadratic' tetrahedron is used for the bound calculation, this step is not needed.

4. Apply the refined mesh to perform the bound calculation.

The procedure from step 1 to step 4 is repeated until a selected accuracy is achieved.

4.4 Numerical Validation

In this section, the convection–diffusion equation is investigated with the adaptive mesh refinement technique for the cube test case where $U_i = (1, 1, 1)$, $f = 1$, and $\varepsilon = 0.1$. The second output $s^{(2)}$, which is the average corner solution is considered for the purpose of demonstrating the adaptive methodology.
Chapter 4. Adaptive Mesh Refinement

Computational results are presented here to illustrate the coarse mesh solution. The bound values, the bound gap information and the relative (half) bound gap as a function of number of elements. Three different meshes, which are the uniformly refined mesh (i.e. $\mathcal{T}_{(H,6)}$, $\mathcal{T}_{(H,9)}$, $\mathcal{T}_{(H,12)}$, and $\mathcal{T}_{(H,15)}$), the adaptively refined mesh starting from a structured mesh, and the adaptively refined mesh starting from an unstructured mesh are used for computations. Note that, unlike the calculation of previous chapter the refinement of local subdomain is fixed at 6 ($R = 6$) for all the bound calculations. When $R = 6$, each local subdomain has 84 nodes and 216 tetrahedral elements. In the adaptive process, elements are selected for refinement based on equation (4.7) with $\alpha = 0.4$ for all refinements.

4.4.1 Uniformly Refined Meshes

It is now important to investigate the influence of uniformly subdivided meshes and various bound values for the average corner solution output $s^{(2)}$. To begin, different coarse subdivisions are considered: i.e. $\mathcal{T}_{(H,6)}$, $\mathcal{T}_{(H,9)}$, $\mathcal{T}_{(H,12)}$, and $\mathcal{T}_{(H,15)}$.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$\mathcal{T}_{(H,6)}$</th>
<th>$\mathcal{T}_{(H,9)}$</th>
<th>$\mathcal{T}_{(H,12)}$</th>
<th>$\mathcal{T}_{(H,15)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td># of nodes</td>
<td>1,296</td>
<td>4,374</td>
<td>10,368</td>
<td>20,250</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
<td>0.3655</td>
<td>0.2927</td>
<td>0.2613</td>
<td>0.2451</td>
</tr>
<tr>
<td>$s_H$</td>
<td>0.1476</td>
<td>0.1795</td>
<td>0.1920</td>
<td>0.1982</td>
</tr>
<tr>
<td>$(s_h)_{LB}$</td>
<td>0.0080</td>
<td>0.1074</td>
<td>0.1482</td>
<td>0.1687</td>
</tr>
<tr>
<td>$(s_h)_{pre}$</td>
<td>0.1868</td>
<td>0.2000</td>
<td>0.2047</td>
<td>0.2069</td>
</tr>
<tr>
<td>$\pm \Delta(T_H)$</td>
<td>$\pm 0.1788$</td>
<td>$\pm 0.0927$</td>
<td>$\pm 0.0566$</td>
<td>$\pm 0.0382$</td>
</tr>
<tr>
<td>$\theta$ (%)</td>
<td>95.7</td>
<td>46.3</td>
<td>27.6</td>
<td>18.5</td>
</tr>
</tbody>
</table>

Table 4.1: Bounds for subdivisions.

For a sequence of uniform refinement results of the bound method of four different subdivisions are summarized in Table 4.1, in which $\Delta(T_H)$ and $\theta$ denote the (half) bound gap and the relative (half) bound gap of each subdivision, respectively. Table 4.1 shows that a reduction of the relative (half) bound gap from 95.7% to 18.5% is achieved by uniform refinement of the coarse mesh.

Figure 4.3 presents the uniformly refined meshes: $\mathcal{T}_{(H,6)}$, $\mathcal{T}_{(H,9)}$, $\mathcal{T}_{(H,12)}$, and $\mathcal{T}_{(H,15)}$. Figure 4.4 shows isocontours of temperature for a section at $z = 0.83333$ of each subdivision, in which
CHAPTER 4. ADAPTIVE MESH REFINEMENT

Table 4.2: Number of elements (%) by half bound gap $\Delta(T_H)$ intervals for subdivisions.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$\mathcal{T}_{(H,6)}$</th>
<th>$\mathcal{T}_{(H,9)}$</th>
<th>$\mathcal{T}_{(H,12)}$</th>
<th>$\mathcal{T}_{(H,15)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta(T_H) \geq 10^{-3}$</td>
<td>1,296 (5%)</td>
<td>4,374 (0%)</td>
<td>10,368 (0%)</td>
<td>20,250 (0%)</td>
</tr>
<tr>
<td>$5 \times 10^{-4} \leq \Delta(T_H) &lt; 10^{-3}$</td>
<td>65 (5%)</td>
<td>2 (0%)</td>
<td>0 (0%)</td>
<td>0 (0%)</td>
</tr>
<tr>
<td>$10^{-1} \leq \Delta(T_H) &lt; 5 \times 10^{-4}$</td>
<td>15 (1.2%)</td>
<td>18 (0.4%)</td>
<td>0 (0%)</td>
<td>0 (0%)</td>
</tr>
<tr>
<td>$5 \times 10^{-5} \leq \Delta(T_H) &lt; 10^{-4}$</td>
<td>91 (7%)</td>
<td>223 (5.1%)</td>
<td>108 (1%)</td>
<td>15 (0.1%)</td>
</tr>
<tr>
<td>$10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>71 (5.5%)</td>
<td>86 (2%)</td>
<td>224 (2.2%)</td>
<td>84 (0.4%)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>542 (41.8%)</td>
<td>434 (9.9%)</td>
<td>560 (5.4%)</td>
<td>832 (4.1%)</td>
</tr>
<tr>
<td>$10^{-6} \leq \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>311 (24%)</td>
<td>516 (11.8%)</td>
<td>406 (3.9%)</td>
<td>537 (2.6%)</td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 10^{-6}$</td>
<td>201 (15.5%)</td>
<td>1,968 (45%)</td>
<td>2,280 (22%)</td>
<td>2,179 (10.8%)</td>
</tr>
</tbody>
</table>

Table 4.3: Memory requirement (M) and elapsed CPU time (seconds) for computation of subdivisions.

<table>
<thead>
<tr>
<th></th>
<th>$\mathcal{T}_{(H,6)}$</th>
<th>$\mathcal{T}_{(H,9)}$</th>
<th>$\mathcal{T}_{(H,12)}$</th>
<th>$\mathcal{T}_{(H,15)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory requirement</td>
<td>28 (5%)</td>
<td>140 (2%)</td>
<td>582 (21%)</td>
<td>1,889 (22%)</td>
</tr>
<tr>
<td>Coarse mesh calculations</td>
<td>24 (4%)</td>
<td>153 (5%)</td>
<td>615 (22%)</td>
<td>2,163 (25%)</td>
</tr>
<tr>
<td>Fine mesh calculations</td>
<td>15 (3%)</td>
<td>53 (2%)</td>
<td>160 (6%)</td>
<td>386 (5%)</td>
</tr>
<tr>
<td>Total calculations</td>
<td>39 (7%)</td>
<td>206 (8%)</td>
<td>775 (28%)</td>
<td>2,549 (30%)</td>
</tr>
</tbody>
</table>

The value of isocontours range from 0 to 0.3 at intervals of 0.05. The right square box of each figure denotes the domain projected in two dimensions (i.e. $[2/3, 1] \times [2/3, 1]$) of the average corner solution output $s^{(2)}$. In addition, a high resolution solution associated with $\mathcal{T}_{(H,72)}$ mesh is presented in Figure 4.4. It is easily noticed that the isocontour of temperature is improved as the number of elements is increased and isocontours of $\mathcal{T}_{(H,15)}$ are the closest to that of $\mathcal{T}_{(H,72)}$.

Table 4.2 summarizes the number of elements (%) by half bound gap $\Delta(T_H)$ intervals of each subdivision. As the number of elements is increased, the bound error (bound gap) is decreased correspondingly. For the final subdivision $\mathcal{T}_{(H,15)}$, it is observed that the 82% of elements have the value of (half) bound gap $\Delta(T_H)$ less than $10^{-6}$. However the range of bound gaps of the other elements is from $5 \times 10^{-4}$ to $10^{-6}$, which is rather wide. It is mainly because each subdivision uses the uniform mesh over the entire domain and causes the higher contribution to the bound error to concentrate on the domain of $\Omega^0$, which is natural for
Figure 4.3: Uniformly refined meshes: (a) $\mathcal{T}_{(H,6)}$ has 1,296 elements, (b) $\mathcal{T}_{(H,9)}$ has 4,374 elements, (c) $\mathcal{T}_{(H,12)}$ has 10,368 elements, and (d) $\mathcal{T}_{(H,15)}$ has 20,250 elements.

convection dominated problems.

Table 4.3 reports the memory requirement and the elapsed CPU time for computation of each subdivision. The memory and the CPU time increases for larger number of subdivisions.
Figure 4.4: Isocontours (0 to 0.3 at intervals of 0.05) of temperature of subdivisions (slice at \( z=0.8333 \)):
(a) \( \mathcal{T}_{(H,6)} \) has 1,296 elements, (b) \( \mathcal{T}_{(H,9)} \) has 4,374 elements, (c) \( \mathcal{T}_{(H,12)} \) has 10,368 elements, (d) \( \mathcal{T}_{(H,15)} \) has 20,250 elements, and (e) \( \mathcal{T}_{(H,72)} \) has 2,239,488 elements.
### 4.4.2 Adaptive Refinement of A Structured Initial Mesh

In order to decrease the bound gap significantly and distribute the local bound gap error more uniformly, the adaptive mesh refinement technique is proposed. In this section, some of results of the bound method are presented using adaptively refined methodology applied to an initial structured mesh.

Results of the bound method of adaptively refined meshes are summarized in Table 4.4. Table 4.4 shows that a reduction of the relative (half) bound gap from 95.7% to 24.9% is achieved by using adaptive refinement, while the uniform refinement only decreases the relative (half) bound gap from 95.7% to 33.2% with even more elements ($T_{H/2}^0$ has 10,368 elements) than that of adaptive refinement ($T_{H}^0$ has 9,785 elements).

Figure 4.6 presents the adaptively refined meshes starting from an initial structured mesh: i.e. $T_{H}^0$, $T_{H}^1$, $T_{H}^2$, $T_{H}^3$, $T_{H}^4$ and $T_{H}^5$. Figure 4.7 shows the isocontours of temperature sliced at $z = 0.83333$ of adaptively refined meshes, in which the value of isocontours from 0 to 0.3 at intervals of 0.05 is used. The isocontour of temperature is improved as the number of elements is increased. Isocontours of final adapted mesh $T_{H}^5$ after 5 refinements are smoother than that of the uniformly refined mesh $T_{H/2}^0$ in the domain of the output.

Table 4.5 presents the memory requirement and the elapsed CPU time for computation of adaptively refined meshes. The fourth adapted mesh $T_{H}^4$ (4,697 elements) which has 34% relative bound gap $\theta$ requires 1/2 CPU time of the uniformly refined mesh $T_{H/2}^0$ (10,368 elements) which achieves 33.2% relative bound gap $\theta$. Note that, final adapted mesh $T_{H}^5$ has 9,601

---

Table 4.4: Bounds for adaptive refinement of a structured initial mesh.

<table>
<thead>
<tr>
<th></th>
<th>$T_{H}^0$</th>
<th>$T_{H}^1$</th>
<th>$T_{H}^2$</th>
<th>$T_{H}^3$</th>
<th>$T_{H}^4$</th>
<th>$T_{H}^5$</th>
<th>$T_{H/2}^0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$#$ of elements</td>
<td>1,296</td>
<td>1,673</td>
<td>1,986</td>
<td>2,897</td>
<td>4,697</td>
<td>9,601</td>
<td>10,368</td>
</tr>
<tr>
<td>$#$ of nodes</td>
<td>343</td>
<td>430</td>
<td>492</td>
<td>698</td>
<td>1,063</td>
<td>1,966</td>
<td>2,096</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
<td>0.3655</td>
<td>0.3319</td>
<td>0.3080</td>
<td>0.2924</td>
<td>0.2858</td>
<td>0.2663</td>
<td>0.2733</td>
</tr>
<tr>
<td>$s_{H}$</td>
<td>0.1476</td>
<td>0.1844</td>
<td>0.1850</td>
<td>0.1927</td>
<td>0.1989</td>
<td>0.2012</td>
<td>0.1920</td>
</tr>
<tr>
<td>$(s_h)_{LB}$</td>
<td>0.0080</td>
<td>0.0769</td>
<td>0.1014</td>
<td>0.1269</td>
<td>0.1407</td>
<td>0.1599</td>
<td>0.1372</td>
</tr>
<tr>
<td>$\pm \Delta (T_{H})$</td>
<td>$\pm 0.1788$</td>
<td>$\pm 0.1275$</td>
<td>$\pm 0.1032$</td>
<td>$\pm 0.0828$</td>
<td>$\pm 0.0725$</td>
<td>$\pm 0.0523$</td>
<td>$\pm 0.0680$</td>
</tr>
<tr>
<td>$\theta$ (%)</td>
<td>95.7</td>
<td>62.4</td>
<td>50.5</td>
<td>39.5</td>
<td>34.0</td>
<td>24.9</td>
<td>33.2</td>
</tr>
</tbody>
</table>
elements. It requires higher memory requirement and longer CPU time than uniformly refined mesh $T_H^{0}$ which has 10,368 elements. It is mainly because the adaptively refined mesh has more stretched elements than that of uniformly refined mesh, therefore it requires more FETI iterations. The stopping criteria of FETI iterations is $10^{-3}$ in this case. Overall computational cost can be improved with a better implementation of the FETI method.

Table 4.6 presents the number of elements (%) by half bound gap $\Delta(T_H)$ intervals from the bound method with adaptive refinement starting from an initial structured mesh. As the number of elements is increased, the bound error (bound gap) is decreased correspondingly. Final adapted mesh $T_H^{3}$ (9,601 elements) only has 12 elements (0.1%) which have value of bound gap greater than $5 \times 10^{-5}$ while the uniformly refined mesh $T_H^{0}$ (10.368 elements) has 360 elements (3.5%) and these elements cause a larger bound gap (local bound gap error).

Figure 4.5 shows the relative bound gap $\theta$ as a function of the number of elements using an uniform and an adaptive refinement of a structured initial mesh. From Figure 4.5 it can be clearly observed that the values of relative (half) bound gap $\theta$ for the adaptively refined meshes presents better results than that of the uniformly refined meshes.
Table 4.6: Number of elements (%) by half bound gap $\Delta(T_H)$ intervals for adaptive refinement of a structured initial mesh.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^1$</th>
<th>$T_H^2$</th>
<th>$T_H^3$</th>
<th>$T_H^4$</th>
<th>$T_H^{1/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta(T_H) \geq 10^{-3}$</td>
<td>65 (5)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$5 \times 10^{-4} \leq \Delta(T_H) &lt; 10^{-3}$</td>
<td>15 (1.2)</td>
<td>7 (0.4)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$10^{-4} \leq \Delta(T_H) &lt; 5 \times 10^{-4}$</td>
<td>91 (7)</td>
<td>268 (13.5)</td>
<td>22 (0.5)</td>
<td>0 (0)</td>
<td>136 (1.3)</td>
</tr>
<tr>
<td>$5 \times 10^{-5} \leq \Delta(T_H) &lt; 10^{-4}$</td>
<td>71 (5.5)</td>
<td>330 (16.6)</td>
<td>242 (5.1)</td>
<td>12 (0.1)</td>
<td>224 (2.2)</td>
</tr>
<tr>
<td>$10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>542 (41.8)</td>
<td>950 (47.8)</td>
<td>1,897 (40.4)</td>
<td>1,304 (13.6)</td>
<td>713 (6.9)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>311 (24)</td>
<td>303 (15.3)</td>
<td>1,225 (26.1)</td>
<td>2,166 (22.6)</td>
<td>671 (6.5)</td>
</tr>
<tr>
<td>$10^{-6} \leq \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>201 (15.5)</td>
<td>128 (6.4)</td>
<td>1,232 (26.2)</td>
<td>5,420 (56.4)</td>
<td>3,221 (31)</td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 10^{-6}$</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>79 (1.7)</td>
<td>699 (7.3)</td>
<td>5,403 (52.1)</td>
</tr>
</tbody>
</table>

Figure 4.5: Relative bound gap $\theta$ as a function of the number of elements in log-log scale using uniform and adaptive refinement of a structured initial mesh.
Figure 4.6: Adaptively refined meshes starting from a structured initial mesh: (a) initial mesh $\mathcal{T}_H^0$ has 1,296 elements, (b) second refined mesh $\mathcal{T}_H^2$ has 1,986 elements, (c) third refined mesh $\mathcal{T}_H^3$ has 2,897 elements, (d) fourth refined mesh $\mathcal{T}_H^4$ has 4,697 elements, (e) final mesh $\mathcal{T}_H^5$, after 5 refinements, has 9,601 elements, and (f) uniformly refined mesh $\mathcal{T}_{H/2}^6$ has 10,368 elements.
Figure 4.7: Isocontours (0 to 0.3 at intervals of 0.05) of temperature for adaptive refinement of a structured initial mesh (slice at z=0.8333): (a) first refined mesh $\mathcal{T}_H^1$ has 1,673 elements, (b) second refined mesh $\mathcal{T}_H^2$ has 1,986 elements, (c) third refined mesh $\mathcal{T}_H^3$ has 2,897 elements, (d) fourth refined mesh $\mathcal{T}_H^4$ has 4,697 elements, (e) final mesh $\mathcal{T}_H^5$, after 5 refinements, has 9,601 elements and (f) uniformly refined mesh $\mathcal{T}_{H/2}^0$ has 10,368 elements.
Table 4.7: Bounds for adaptive refinement of an unstructured initial mesh.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$\mathcal{T}_H^0$</th>
<th>$\mathcal{T}_H^1$</th>
<th>$\mathcal{T}_H^2$</th>
<th>$\mathcal{T}_H^3$</th>
<th>$\mathcal{T}_H^4$</th>
<th>$\mathcal{T}_{H/2}^0$</th>
</tr>
</thead>
<tbody>
<tr>
<td># of nodes</td>
<td>1,440</td>
<td>2,014</td>
<td>2,639</td>
<td>4,985</td>
<td>9,739</td>
<td>11,520</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
<td>0.3295</td>
<td>0.2953</td>
<td>0.2809</td>
<td>0.2723</td>
<td>0.2547</td>
<td>0.2555</td>
</tr>
<tr>
<td>$s_H$</td>
<td>0.1710</td>
<td>0.1946</td>
<td>0.1960</td>
<td>0.2008</td>
<td>0.2041</td>
<td>0.1991</td>
</tr>
<tr>
<td>$(s_h)_{LB}$</td>
<td>0.0661</td>
<td>0.1241</td>
<td>0.1400</td>
<td>0.1559</td>
<td>0.1733</td>
<td>0.1630</td>
</tr>
<tr>
<td>$(s_h)_{pre}$</td>
<td>0.1978</td>
<td>0.2097</td>
<td>0.2105</td>
<td>0.2141</td>
<td>0.2140</td>
<td>0.2093</td>
</tr>
<tr>
<td>$\pm \Delta (T_H)$</td>
<td>$\pm 0.1317$</td>
<td>$\pm 0.0856$</td>
<td>$\pm 0.0704$</td>
<td>$\pm 0.0582$</td>
<td>$\pm 0.0407$</td>
<td>$\pm 0.0463$</td>
</tr>
<tr>
<td>$\Theta$ (%)</td>
<td>66.6</td>
<td>40.8</td>
<td>33.5</td>
<td>27.2</td>
<td>19</td>
<td>22.1</td>
</tr>
</tbody>
</table>

4.4.3 Adaptive Refinement of An Unstructured Initial Mesh

The continuous refinement of a structured mesh can generate many stretched elements, that is elements with high value of aspect ratio or edge ratio, as it is observed from Figure 4.6. Many stretched elements not only degrade numerical accuracy but also increase the computational cost. Because of the shape of the mesh, the refinement of a structured initial mesh can generate more stretched elements than that of an unstructured initial mesh. Therefore, the adaptive refinement of an unstructured initial mesh gives better results. Note that, the unstructured mesh is the most common type of mesh and is easy to generate using commercial mesh generating packages.

The influence of the adaptive mesh refinement technique for unstructured meshes is investigated in this section. To generate an unstructured initial mesh $\mathcal{T}_H^0$, the commercial mesh generating package Gambit (© Fluent Inc.) is used.

Results of the bound method with adaptively generated meshes starting from an unstructured initial mesh are summarized in Table 4.7. Table 4.7 shows that a reduction of the relative (half) bound gap from 66.6% to 19% is achieved by using adaptive refinement. While the uniform refinement only decreases the relative (half) bound gap from 66.6% to 22.1% with even more elements ($\mathcal{T}_{H/2}^0$ has 11,520 elements) than that of adaptive refinement ($\mathcal{T}_H^1$ has 9,739 elements).

Similarly to the structured mesh analysis, Figure 4.9 illustrates the adaptively refined meshes starting from an unstructured initial mesh: i.e. $\mathcal{T}_H^0, \mathcal{T}_H^1, \mathcal{T}_H^2, \mathcal{T}_H^3, \mathcal{T}_H^4$, and $\mathcal{T}_{H/2}^0$. 
Table 4.8: Memory requirement (M) and elapsed CPU time (seconds) for computation of adaptive refinement of an unstructured initial mesh.

<table>
<thead>
<tr>
<th># of elements</th>
<th>(T_{H}^0)</th>
<th>(T_{H}^1)</th>
<th>(T_{H}^2)</th>
<th>(T_{H}^3)</th>
<th>(T_{H}^{1/2})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory requirement</td>
<td>43</td>
<td>79</td>
<td>11</td>
<td>265</td>
<td>696</td>
</tr>
<tr>
<td>Coarse mesh calculations</td>
<td>41</td>
<td>110</td>
<td>172</td>
<td>522</td>
<td>1324</td>
</tr>
<tr>
<td>Fine mesh calculations</td>
<td>16</td>
<td>25</td>
<td>33</td>
<td>72</td>
<td>167</td>
</tr>
<tr>
<td>Total calculations</td>
<td>57</td>
<td>135</td>
<td>205</td>
<td>594</td>
<td>1491</td>
</tr>
</tbody>
</table>

Table 4.8: Memory requirement (M) and elapsed CPU time (seconds) for computation of adaptive refinement of an unstructured initial mesh.

Figure 4.10 shows the isocontours of temperature for a slice at \(z = 0.83333\) of computational domain. The final adapted mesh \(T_{H}^3\) (9,739 elements) gives smoother isocontours than that of the uniformly refined mesh \(T_{H/2}^0\) (11,520 elements).

Table 4.8 presents the memory requirement and the elapsed CPU time of the bound method with adaptively refined meshes starting from an unstructured initial mesh. Although the final adapted mesh \(T_{H}^3\) (9,739 elements) has less elements than uniformly refined mesh \(T_{H/2}^0\) (11,520 elements), the final adapted mesh \(T_{H}^3\) (9,739 elements) requires longer CPU time than that of uniformly refined mesh \(T_{H/2}^0\) (11,520 elements). It is mainly because the adaptively refined mesh has more stretched elements than that of the uniformly refined mesh and it requires more FETI iterations.

Table 4.9 presents the number of elements (%) by half bound gap \(\Delta(T_{H})\) intervals of adaptively refined meshes starting from an unstructured initial mesh. As the further refinement is performed, the bound error (bound gap) is decreased correspondingly. Final adapted mesh \(T_{H}^3\) (9,739 elements) only has 1 element (0%) in value of bound gap greater than \(5 \times 10^{-5}\) while the uniformly refined mesh \(T_{H/2}^0\) (11,520 elements) has 211 elements (1.9%) and these elements still cause the higher value of bound gap (bound error).

The relative bound gap \(\theta\) as a function of the number of elements using uniform and adaptive refinement of an unstructured initial mesh is shown in Figure 4.8a. In addition, the relative bound gap \(\theta\) as a function of the number of elements is summarized in Figure 4.8b. From Figure 4.8a and Figure 4.8b it is clearly observed that the adaptive refinement of an unstructured initial mesh provides up to 6% better results of relative bound gap over the adaptive refinement of a structured initial mesh.
CHAPTER 4. ADAPTIVE MESH REFINEMENT

The continuous refinement of an unstructured initial mesh can also generate stretched elements, as it is observed from Figure 4.9. Further refinement does not improve local bound gap error (or reduce the bound gap) significantly and causes higher memory and longer CPU time requirement for FETI iterations. In order to reduce the local bound gap error further, the optimal stabilization parameter $\kappa^*$ should be applied to the bound calculations. The optimal stabilization parameter $\kappa^*$ will be introduced later in Chapter 5.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^3$</th>
<th>$T_H^4$</th>
<th>$T_H^4$</th>
<th>$T_H^4$</th>
<th>$T_H^{1/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta(T_H) \geq 10^{-3}$</td>
<td>1,440</td>
<td>2,639</td>
<td>4,985</td>
<td>9,739</td>
<td>11,520</td>
</tr>
<tr>
<td>$5 \times 10^{-4} \leq \Delta(T_H) &lt; 10^{-3}$</td>
<td>34 (2.4)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$10^{-4} \leq \Delta(T_H) &lt; 5 \times 10^{-4}$</td>
<td>44 (3.0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>90 (6.3)</td>
<td>95 (3.6)</td>
<td>4 (0.1)</td>
<td>0 (0)</td>
<td>53 (0.5)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>75 (5.2)</td>
<td>287 (10.9)</td>
<td>100 (2.0)</td>
<td>1 (0)</td>
<td>158 (1.4)</td>
</tr>
<tr>
<td>$10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>507 (35.2)</td>
<td>1,382 (52.4)</td>
<td>1,870 (37.5)</td>
<td>884 (9.1)</td>
<td>751 (6.5)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>392 (27.2)</td>
<td>543 (20.6)</td>
<td>1,327 (26.6)</td>
<td>1,712 (17.6)</td>
<td>420 (3.6)</td>
</tr>
<tr>
<td>$10^{-6} \leq \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>297 (20.6)</td>
<td>329 (12.4)</td>
<td>1,487 (29.8)</td>
<td>5,511 (56.6)</td>
<td>2,599 (22.6)</td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 10^{-6}$</td>
<td>1 (0.1)</td>
<td>3 (0.1)</td>
<td>197 (4.0)</td>
<td>1,631 (16.7)</td>
<td>7,539 (65.4)</td>
</tr>
</tbody>
</table>

Table 4.9: Number of elements (%) by half bound gap $\Delta(T_H)$ intervals for adaptive refinement of an unstructured initial mesh.
Figure 4.8: (a) Relative bound gap $\theta$ as a function of the number of elements in log-log scale using uniform and adaptive refinement of an unstructured initial mesh (top). (b) Summary of relative bound gap $\theta$ as a function of the number of elements in log-log scale (bottom).
CHAPTER 4. ADAPTIVE MESH REFINEMENT

Figure 4.9: Adaptively refined meshes starting from an unstructured initial mesh: (a) initial mesh $\mathcal{T}_H^2$ has 1,440 elements, (b) first refined mesh $\mathcal{T}_H^1$ has 2,014 elements, (c) second refined mesh $\mathcal{T}_H^2$ has 2,639 elements, (d) third refined mesh $\mathcal{T}_H^3$ has 4,985 elements, (e) final mesh $\mathcal{T}_H^4$, after 4 refinements, has 9,739 elements, and (f) uniformly refined mesh $\mathcal{T}_{H/2}^0$ has 11,520 elements.
Figure 4.10: Isocontours (0 to 0.3 at intervals of 0.05) of temperature for adaptive refinement of an unstructured initial mesh (slice at $z=0.8333$): (a) initial mesh $T_H^0$ has 1,440 elements. (b) first refined mesh $T_H^1$ has 2,014 elements, (c) second refined mesh $T_H^2$ has 2,639 elements. (d) third refined mesh $T_H^3$ has 4,985 elements, (e) final mesh $T_H^4$, after 4 refinements, has 9,739 elements and (f) uniformly refined mesh $T_{H/2}^0$ has 11,520 elements.
Chapter 5

Optimal Stabilization Parameter

In this section a procedure which improves the sharpness of the bounds by maximizing the lower bound and minimizing the upper bound is presented. To do this, a positive real number $\kappa$, to scale the output of interest is introduced. Both the lower and upper bounds to this scaled output are calculated. It is obvious that the case of $\kappa = 1$ the usual, non-optimal bounds are obtained, which were discussed in Section 3.2 and 3.3. Therefore for any real value of $\kappa$ the lower and upper bounds will be obtained. I now look for the parameter $\kappa$ that will yield the sharpest bounds. A $\kappa$-optimization procedure is presented in matrix form. Variational optimal bound formulation is presented in Appendix A.

5.1 Optimal Bound Formulation

The strategy is to write all variables as linear functions in $\kappa$ and then derive the bounds as a function in $\kappa$. This procedure does not change the bounding theory and the bounds remain rigorous. For the purpose of simplicity, the optimized bound formulation is presented in matrix form.

The Lagrangian is now expressed with stabilization parameter $\kappa$:

$$\mathcal{L}^{\pm}(v, \mu, q) = \sum_{k=1}^{N_k} \left[ \kappa \left( \begin{array}{c} v^{(k)T} A^{(k)} v^{(k)} - f^{(k)T} v^{(k)} + v^{(k)T} L^{(k)} \Theta_{H}^{(k)} \end{array} \right) \pm \varepsilon^{(k)T} \left( \Theta_{H}^{(k)} + v^{(k)} \right) \right]$$

$$+ \sum_{k=1}^{N_k} \mu^{(k)T} \left[ L^{(k)} (\Theta_{H}^{(k)} + v^{(k)}) - f^{(k)} \right]$$

$$+ q^{T} \sum_{k=1}^{N_k} B^{(k)} v^{(k)}$$

(5.1)
CHAPTER 5. OPTIMAL STABILIZATION PARAMETER

where, for \( N_k \) subdomains, \( v = \{ v^{(1)}, \ldots, v^{(N_k)} \} \) and \( \mu = \{ \mu^{(1)}, \ldots, \mu^{(N_k)} \} \). Also, \( A^{(k)} \) is the finite element discretization of the symmetric part of \( L^{(k)} \) and \( B^{(k)} \) is the sign Boolean matrix which localizes the “jumps” at the interface.

5.1.1 Adjoint Calculation

The \( H \)-mesh adjoint calculation is considered here. By evoking stationarity conditions of the Lagrangian (5.1), three equations are obtained:

Find \( e_H^\pm \in \tilde{X}_H, \psi_H^\pm \in X_H \) and \( \lambda_H^\pm \in Q_H \), such that

\[
B_H^T \lambda_H^\pm + \kappa \left( 2A_H e_H^\pm - f_H + L_H \Theta_H \right) + L_H^T \psi_H^\pm \pm \ell_H = 0. \tag{5.2}
\]

\[
L_H \Theta_H + L_H e_H^\pm - f_H = 0. \tag{5.3}
\]

\[
B_H^T e_H^\pm = 0. \tag{5.4}
\]

For the continuous space \( X_H \),

\[
L_H^T \psi_H + \ell_H = 0. \tag{5.5}
\]

Note that, the adjoint calculation is same as equation (3.83), since it is independent of parameter \( \kappa \).

5.1.2 Hybrid Flux Calculation

Knowing the adjoints, the hybrid flux can be evaluated: Find \( \lambda_H^\pm \in Q_H \) such that

\[
B_H^T \lambda_H^\pm = \kappa [f_H - L_H \Theta_H] \mp \ell_H - L_H^T \psi_H^\pm. \tag{5.6}
\]

Alternatively,

\[
B_H^T \lambda_0 H = - \ell_H - L_H^T \psi_H. \tag{5.7}
\]

\[
B_H^T \lambda_1 H = f_H - L_H \Theta_H. \tag{5.8}
\]

and then set \( \lambda_H^\pm = \pm \lambda_0 H + \kappa \lambda_1 H \). The right hand side of equations (5.7) and (5.8) are denoted as the residual of the adjoint and the field solution, respectively. Note that, equations (5.7) and (5.8) represent a solvable but indeterminate system.
If FETI method is applied, equations (5.7) and (5.8) become

\[
2A_H^{(k)}e_{0H}^{(k)} + B_H^{(k)^T}\lambda_0 = -\ell_{H}^{(k)} - L_H^{(k)^T}\psi_{H}^{(k)},
\]

\[
\sum_{k=1}^{N_k} B_H^{(k)^T}e_{0H}^{(k)} = 0,
\]

\[
k = 1, \ldots, N_k.
\]  

(5.9)

Similarly,

\[
2A_H^{(k)}e_{1H}^{(k)} + B_H^{(k)^T}\lambda_1 = f_{H}^{(k)} - L_H^{(k)^T}\Theta_{H}^{(k)},
\]

\[
\sum_{k=1}^{N_k} B_H^{(k)^T}e_{1H}^{(k)} = 0,
\]

\[
k = 1, \ldots, N_k.
\]  

(5.10)

Note that, equations (5.9) and (5.10) can be reformulated into an interface problem for the FETI procedure. The FETI procedure for the optimal bound formulation is discussed in Section 6.2.

5.1.3 Local Neumann Problem

The minimizers of \( L^\pm (v, \psi^\pm_h, \lambda^\pm_h) \), which is \( \tilde{e}_h^\pm \in \tilde{X}_h \), will satisfy the following equation.

\[
2\kappa A_H^{(k)}(\tilde{e}_h^\pm)^{(k)} = \kappa \left[ f_{H}^{(k)} - L_H^{(k)^T}\tilde{\Theta}_{h}^{(k)} \right] \mp \ell_{h}^{(k)} - L_H^{(k)^T}\psi_{h}^{\pm(k)} - B_h^{(k)^T}\lambda_{h}^{\pm(k)}.
\]

\[
k = 1, \ldots, N_k.
\]  

(5.11)

Alternatively, in terms of \( \tilde{e}_{0h} \) and \( \tilde{e}_{1h} \)

\[
2A_H^{(k)}\tilde{e}_{0h}^{(k)} = -\ell_{h}^{(k)} - L_H^{(k)^T}\psi_{h}^{(k)} - B_h^{(k)^T}\lambda_{0h},
\]

\[
2A_H^{(k)}\tilde{e}_{1h}^{(k)} = f_{h}^{(k)} - L_H^{(k)^T}\tilde{\Theta}_{h}^{(k)} - B_h^{(k)^T}\lambda_{1h},
\]

\[
k = 1, \ldots, N_k,
\]  

(5.12)

(5.13)

(5.14)

where \( \tilde{e}_h^\pm = \pm \frac{1}{\kappa} \tilde{e}_{0h} + \tilde{e}_{1h} \).

5.1.4 Bound Calculations

The lower and upper bounds can be derived from Lagrangian (5.1):

\[
(s_H)_{UB} = s_H + \sum_{k=1}^{N_k} \kappa(\tilde{e}_h^\pm)^{(k)^T}A_h^{(k)}(\tilde{e}_h^\pm)^{(k)},
\]  

(5.15)
\begin{equation}
(s_h)_{LB} = s_H - \sum_{k=1}^{N_k} \kappa \left( \hat{e}_h^{(k)} \right)^T A_h^{(k)} \left( \hat{e}_h^{(k)} \right).
\end{equation}

Alternatively,
\begin{align}
(s_h)_{UB} &= s_H - \sum_{k=1}^{N_k} 2 \left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)} \\
&\quad + \frac{1}{\kappa} \left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{0h}^{(k)} + \kappa \left( \varepsilon_{1h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)}, \\
(s_h)_{LB} &= s_H - \sum_{k=1}^{N_k} 2 \left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)} \\
&\quad - \frac{1}{\kappa} \left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{0h}^{(k)} - \kappa \left( \varepsilon_{1h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)}.
\end{align}

The half bound gap in matrix form becomes:
\begin{equation}
\Delta(T_H, \kappa) = \sum_{k=1}^{N_k} \frac{1}{\kappa} \left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{0h}^{(k)} + \kappa \left( \varepsilon_{1h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)}.
\end{equation}

The output predictor is rewritten in matrix form as
\begin{equation}
(s_h)_{prc}(T_H) = s_H - \sum_{k=1}^{N_k} 2 \left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)}.
\end{equation}

Note that, equation (5.20) is independent of the parameter \( \kappa \).

5.1.5 Optimal Bounds

The optimal parameter \( \kappa^* \) is evaluated in matrix form.
\begin{equation}
\kappa^* = \sqrt{\frac{\left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{0h}^{(k)}}{\left( \varepsilon_{1h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)}}}, \quad k = 1, \ldots, N_k.
\end{equation}

Then, the optimized bounds are rewritten as
\begin{align}
(s_h)_{UB} &= s_H - \sum_{k=1}^{N_k} 2 \left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)} \\
&\quad + \frac{1}{\kappa^*} \left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{0h}^{(k)} + \kappa^* \left( \varepsilon_{1h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)}, \\
(s_h)_{LB} &= s_H - \sum_{k=1}^{N_k} 2 \left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)} \\
&\quad - \frac{1}{\kappa^*} \left( \varepsilon_{0h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{0h}^{(k)} - \kappa^* \left( \varepsilon_{1h}^{(k)} \right)^T A_h^{(k)} \varepsilon_{1h}^{(k)}.
\end{align}
and the associated bound gap becomes,
\[
\Delta(T_H, \kappa^*) = \sum_{k=1}^{N_k} \frac{1}{\kappa^*} (e_{0h}^{(k)})^T A_h^{(k)} e_{0h}^{(k)} + \kappa^* (e_{1h}^{(k)})^T A_h^{(k)} e_{1h}^{(k)}. \tag{5.24}
\]

Since the choice of parameter \(\kappa\) has no influence on the output predictor, equation (5.20) will also be used for the output predictor for the optimized bounds.

5.2 Numerical Validation

A cube test case where \(U_i = (1, 1, 1), f = 1,\) and \(\varepsilon = 0.1\) described in Chapter 4 is reconsidered for the purpose of demonstrating the optimal adaptive methodology.

The convergence rate with optimal stabilization parameter \(\kappa^*\) is investigated for coarse mesh solution \(s_H\), upper and lower bounds (i.e. \((s_h)_{UB}\) and \((s_h)_{LB}\)), and output predictor \((s_h)_{pre}\). To perform, three different coarse subdivisions are used: i.e. \(T_{(H_3)}\), \(T_{(H_9)}\), and \(T_{(H_{12})}\). Different refinement of local subdomain is used for each subdivision to conserve the same fine “truth” mesh.

Then various results of bound calculations are illustrated for the optimum adaptive methodology. In order to compare results from previous chapter, that is the results of the bound method without optimal stabilization parameter, the refinement of local subdomain is again fixed at 6 \((R = 6)\) for all the bound calculations. When \(R = 6\), the local subdomain has 84 nodes and 216 tetrahedral elements. In the adaptive process, elements are selected for refinement based on equation (4.7) with \(\alpha = 0.4\) for all the refinements.

5.2.1 Convergence Rate of the Average Corner Solution Output \(s^{(2)}\)

The information in Table 5.1 reports optimal results of the bound method for the uniformly refined mesh with different refinement \((R)\). Note that, the reduction of relative (half) bound gap \(\theta\) from 65.5% to 16.5% is achieved with optimal stabilization parameter \(\kappa^*\). while without optimal stabilization parameter it only changes from 95.7% to 24%. The optimal stabilization parameter \(\kappa^*\) is converged to the 2.51 as \(H \rightarrow h\).

To analyze the sharpness of the bounds (i.e. upper and lower bounds) for average corner solution output \(s^{(2)}\), the convergence rate for different coarse “working” meshes are illustrated on Figure 5.1. The bounds with optimal stabilization parameter in Figure 5.1a are sharper than bounds without optimal stabilization parameter in Figure 3.5a.
Figure 5.1: (a) Plots of \((s_h)_{UB}/s_h\), \((s_h)_{LB}/s_h\), \(s_H/s_h\) and \((s_h)_{pre}/s_h\) with optimal stabilization parameter \(\kappa^*\) as a function of \(H\) for the output \(s^{(2)}\) when \(\varepsilon = 0.1\) (top). (b) Plots of \(|(s_h)_{UB} - s_h|\), \(|s_H - s_h|\), \(|(s_h)_{LB} - s_h|\), and \(|(s_h)_{pre} - s_h|\) with optimal stabilization parameter \(\kappa^*\) as a function of \(H\) in log-log scale for the output \(s^{(2)}\) when \(\varepsilon = 0.1\) (bottom).
CHAPTER 5. OPTIMAL STABILIZATION PARAMETER

Table 5.1: Optimal output predictor \((s_h)_{pre}\), half bound gap \(\Delta(T_H)\), and relative (half) bound gap \(\theta\) for each subdivision with different \(R\).

<table>
<thead>
<tr>
<th># of elements</th>
<th>(T_{(H,5)})</th>
<th>(T_{(H,9)})</th>
<th>(T_{(H,12)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refinement (R)</td>
<td>6</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>((s_h)_{pre})</td>
<td>0.1867</td>
<td>0.1987</td>
<td>0.2030</td>
</tr>
<tr>
<td>(\pm\triangle(T_H))</td>
<td>(\pm 0.1013)</td>
<td>(\pm 0.0506)</td>
<td>(\pm 0.0287)</td>
</tr>
<tr>
<td>(\theta)</td>
<td>65.5</td>
<td>30</td>
<td>16.5</td>
</tr>
<tr>
<td>(\kappa^*)</td>
<td>2.53</td>
<td>2.51</td>
<td>2.51</td>
</tr>
</tbody>
</table>

The slopes of \(\log|(s_h)_{UB} - s_h|/\log H\), \(\log|s_H - s_h|/\log H\), and \(\log|(s_h)_{LB} - s_h|/\log H\) are converged to 1.98, 2.05 and 2.04. As noticed from these values and Figure 5.1b, the bounds converge to \(O(H^2)\) as \(H \to h\) conforming with theory [16].

Table 5.2: Optimal bounds for subdivisions.

<table>
<thead>
<tr>
<th># of elements</th>
<th>(T_{(H,5)})</th>
<th>(T_{(H,9)})</th>
<th>(T_{(H,12)})</th>
<th>(T_{(H,15)})</th>
</tr>
</thead>
<tbody>
<tr>
<td># of nodes</td>
<td>1,296</td>
<td>4,374</td>
<td>10,368</td>
<td>20,250</td>
</tr>
<tr>
<td>((s_h)_{UB})</td>
<td>0.3091</td>
<td>0.2642</td>
<td>0.2441</td>
<td>0.2337</td>
</tr>
<tr>
<td>(s_H)</td>
<td>0.1476</td>
<td>0.1795</td>
<td>0.1920</td>
<td>0.1982</td>
</tr>
<tr>
<td>((s_h)_{LB})</td>
<td>0.0644</td>
<td>0.1259</td>
<td>0.1653</td>
<td>0.1801</td>
</tr>
<tr>
<td>((s_h)_{pre})</td>
<td>0.1867</td>
<td>0.2000</td>
<td>0.2047</td>
<td>0.2069</td>
</tr>
<tr>
<td>(\pm\triangle(T_H))</td>
<td>(\pm 0.1223)</td>
<td>(\pm 0.0641)</td>
<td>(\pm 0.0394)</td>
<td>(\pm 0.0268)</td>
</tr>
<tr>
<td>(\theta) (%)</td>
<td>65.5</td>
<td>32.1</td>
<td>19.2</td>
<td>12.9</td>
</tr>
<tr>
<td>(\kappa^*)</td>
<td>2.53</td>
<td>2.49</td>
<td>2.47</td>
<td>2.44</td>
</tr>
</tbody>
</table>

5.2.2 Influence of \(\kappa^*\) on the Uniformly Refined Meshes

The influence of optimal stabilization parameter \(\kappa^*\) is investigated for the average corner solution output \(s^{(2)}\). To start, four different coarse subdivisions in Figure 4.3 are reused here.

Table 5.2 shows that a reduction of the relative (half) bound gap from 65.5% to 12.9% is achieved by using the optimal stabilization parameter \(\kappa^*\). Recall that the reduction of relative bound gap \(\theta\) without optimal stabilization parameter is only from 95.7% to 18.5% with uniformly refined meshes. Note that, optimal bound method improves the relative bound...
Figure 5.2: Relative bound gap $\theta$ with and without optimal stabilization parameter $\kappa^*$ as a function of the number of elements in log-log scale for uniformly refined meshes.

gap from 30.2% to 5.6%. The optimal stabilization parameter $\kappa^*$ is converged to the 2.41 for this case.

Figure 5.2 shows that the relative bound gap $\theta$ with and without optimal stabilization parameter $\kappa^*$ as a function of the number of elements using uniformly refined meshes. It is clearly observed that relative (half) bound gap $\theta$ with optimal stabilization parameter $\kappa^*$ presents better than that of without optimal stabilization parameter $\kappa^*$. 
Table 5.3: Optimal bounds for adaptive refinement of a structured initial mesh.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_{H}^0$</th>
<th>$T_{H}^1$</th>
<th>$T_{H}^2$</th>
<th>$T_{H}^3$</th>
<th>$T_{H}^4$</th>
<th>$T_{H}^{1/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,296</td>
<td>1.673</td>
<td>1.986</td>
<td>2.897</td>
<td>4.697</td>
<td>9.601</td>
<td>10.368</td>
</tr>
<tr>
<td>343</td>
<td>0.3091</td>
<td>0.3018</td>
<td>0.2913</td>
<td>0.2837</td>
<td>0.2805</td>
<td>0.2652</td>
</tr>
<tr>
<td>$s_{H}$</td>
<td>0.1476</td>
<td>0.1844</td>
<td>0.1850</td>
<td>0.1927</td>
<td>0.1989</td>
<td>0.2012</td>
</tr>
<tr>
<td>$s_{H}^{LB}$</td>
<td>0.0644</td>
<td>0.1072</td>
<td>0.1180</td>
<td>0.1356</td>
<td>0.1460</td>
<td>0.1611</td>
</tr>
<tr>
<td>$(s_{h})^{LB}$</td>
<td>0.1867</td>
<td>0.2045</td>
<td>0.2047</td>
<td>0.2096</td>
<td>0.2132</td>
<td>0.2131</td>
</tr>
<tr>
<td>$\pm \Delta(T_{H})$</td>
<td>0.1223</td>
<td>0.0973</td>
<td>0.0866</td>
<td>0.0741</td>
<td>0.0672</td>
<td>0.0520</td>
</tr>
<tr>
<td>$\theta$ (%)</td>
<td>65.5</td>
<td>47.6</td>
<td>42.3</td>
<td>35.3</td>
<td>31.5</td>
<td>24.4</td>
</tr>
<tr>
<td>$\kappa^*$</td>
<td>2.53</td>
<td>2.16</td>
<td>1.84</td>
<td>1.62</td>
<td>1.48</td>
<td>1.24</td>
</tr>
</tbody>
</table>

5.2.3 Influence of $\kappa^*$ on Adaptive Refinement of A Structured Initial Mesh

The adaptively refined meshes starting from a structured initial mesh in Figure 4.6 are re-investigated in this section. Results of optimal bound method for adaptively generated meshes is summarized in Table 5.3. Table 5.3 shows that a reduction of the relative (half) bound gap from 65.5% to 24.4% is achieved by using optimal stabilization parameter $\kappa^*$ and adaptive refinement, while the uniform refinement with optimal stabilization parameter $\kappa^*$ decreases the relative (half) bound gap from 65.5% to 28.4% with even more elements ($T_{H}^{1/2}$ has 10,368 elements) than that of adaptive refinement ($T_{H}^4$ has 9,601 elements). Note that, the reduction of relative bound gap $\theta$ without optimal stabilization parameter is only accomplished from 95.7% to 24.9% with adaptive refinement of a structured initial mesh.

Figure 5.3 shows that the relative bound gap $\theta$ with and without optimal stabilization parameter $\kappa^*$ as a function of the number of elements using adaptive and uniform refinement of a structured initial mesh. The adaptively refined meshes with and without optimal stabilization parameter $\kappa^*$ presents smaller relative (half) bound gap $\theta$ than the uniformly refined meshes.

Relative bound gap with the optimal bound method always presents better than relative bound gap with the non-optimal bound method. For initial mesh $T_{H}^0$ (1,296 elements), 30.2% of relative bound gap $\theta$ improvement is achieved using the optimal stabilization parameter $\kappa^*$. However, for final adapted mesh $T_{H}^5$ (9,601 elements) only 0.5% of relative bound gap $\theta$ improves using the optimal bound method. Due to increased number of adaptive cycles, elements are stretched and thereby the interpolation error of the hybrid flux is increased.
Figure 5.3: Relative bound gap $\theta$ with and without optimal stabilization parameter $\kappa^*$ as a function of the number of elements in log-log scale for uniform and adaptive refinement of a structured initial mesh.
5.2.4 Influence of $\kappa^*$ on Adaptive Refinement of An Unstructured Initial Mesh

The adaptively refined meshes starting from an unstructured initial mesh presented in Figure 4.9 is reconsidered here. Results of the optimal bound method for adaptive refinement of an unstructured initial mesh is summarized in Table 5.4. Table 5.4 shows that a reduction of the relative (half) bound gap from 49.1% to 18.2% is achieved by using the optimal stabilization parameter $\kappa^*$ with adaptive refinement, while the uniform refinement with the optimal stabilization parameter $\kappa^*$ decreases the relative (half) bound gap from 49.1% to 18.2% with even more elements ($T^0_{H/2}$ has 11,520 elements) than that of adaptive refinement ($T^1_H$ has 9,739 elements). Recall that the reduction of relative bound gap $\theta$ without the optimal stabilization parameter is from 66.6% to 19% with adaptive refinement of an unstructured initial mesh.

Figure 5.4a shows the relative bound gap $\theta$ with and without optimal stabilization parameter $\kappa^*$ as a function of the number of elements using adaptive and uniform refinement of an unstructured initial mesh. Figure 5.4b presents the comparison of the relative bound gap $\theta$ using adaptive refinement of a structured initial mesh and of an unstructured initial mesh. The optimal bound method presents better results than the non-optimal bound method. The bound method with adaptive refinement gives better improvement than that with uniform refinement. For the bounds and relative bound gap, the optimal bound method using adaptive refinement of an unstructured initial mesh provides better performance than that using adaptive refinement of a structured initial mesh.

<table>
<thead>
<tr>
<th>Table 5.4: Optimal bounds for adaptive refinement of an unstructured initial mesh.</th>
</tr>
</thead>
<tbody>
<tr>
<td># of elements</td>
</tr>
<tr>
<td># of nodes</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
</tr>
<tr>
<td>$s_H$</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
</tr>
<tr>
<td>$(s_h)_{prec}$</td>
</tr>
<tr>
<td>$\pm \Delta (T_H)$</td>
</tr>
<tr>
<td>$\theta$ (%)</td>
</tr>
<tr>
<td>$\kappa^*$</td>
</tr>
</tbody>
</table>
Bounds gap (the upper and lower bound gap) $\Delta(T_H)$ and relative (half) bound gap $\theta$ with the stabilization parameter $\kappa^*$ are improved significantly up to second or third refinement of the initial mesh. However, after four or five refinements the bound gap and relative bound gap does not improve noticeably even with the stabilization parameter $\kappa^*$. It is mainly due to interpolation error of hybrid flux. In order to achieve further improvement of the bounds values, a different approach which calculates the hybrid flux on the fine $h$-mesh will be investigated. Computations of the hybrid flux on the fine “truth” $h$-mesh are discussed in Chapter 6.
Figure 5.4: Relative bound gap $\theta$ as a function of the number of elements in log-log scale: (a) Summary for optimal and non-optimal refinement (top) and (b) Summary for adaptive refinement of a structured initial mesh and of an unstructured initial mesh (bottom).
Chapter 6

Hybrid Flux on Fine Mesh

In this section a new approach to calculate the hybrid flux is investigated. Previously the hybrid flux is calculated on a coarse $H$-mesh and then interpolated on a fine $h$-mesh, i.e. $\lambda_H \rightarrow \lambda_h$. The $H$-mesh hybrid flux calculations have a computational cost advantage—which means it is inexpensive to approximate hybrid flux on the fine $h$-mesh. However, this approach has increased interpolation error—i.e. increasing the bound gap.

In order to improve the sharpness of the bound gap, a new approach to compute the hybrid flux is investigated. Hybrid flux calculations on the fine “truth” $h$-mesh are guaranteed to increase the computational accuracy by reducing the bound gap. Since the hybrid flux calculations are performed on the fine “truth” $h$-mesh, it requires higher memory and longer CPU time—which means the computational cost is more expensive than that of hybrid flux calculations on the coarse “working” mesh.

The numerical procedure for the hybrid flux calculations on the fine $h$-mesh is discussed and some results are presented. Furthermore, the optimal stabilization parameter $\kappa^*$ to bound calculations is applied. Note that, in case of $\kappa = 1$ the usual, non-optimal bounds are obtained. For the purpose of simplicity, the bounds formulation in matrix form are presented herein.

6.1 $h$-mesh Hybrid Flux Calculation

Since the coarse “working” mesh calculations for solving the field variable $\Theta_H$ and adjoint $\psi_H$ are same as in the previous section, I now focus on the numerical procedure of $h$-mesh inter-subdomain connectivity (i.e. hybrid flux) calculations.

Knowing the coarse $H$-mesh field variable and adjoint, the $h$-mesh hybrid flux can be
calculated using the interpolated values of the field variable and adjoint:

Find $h$-mesh hybrid flux $\lambda^{\pm}_h \in \mathcal{Q}_h$ such that

$$B_h^T \lambda^{\pm}_h = \kappa [f_h - L_h T_h] \mp \ell_h - L_h^T \psi^{\pm}_h,$$

(6.1)

where $\kappa$ is stabilization parameter and it can be any real value. Alternative forms of equation (6.1) are

$$B_h^T \lambda_0 = -\ell_h - L_h^T \psi_h,$$

(6.2)

$$B_h^T \lambda_{1_h} = f_h - L_h T_h,$$

(6.3)

where $\lambda^{\pm}_h$ was replaced by $\pm \lambda_0 + \kappa \lambda_{1_h}$. Note that, equations (6.3) and (6.3) are solvable but indeterminate systems.

Employing the $h$-mesh hybrid fluxes $\lambda_0$ and $\lambda_{1_h}$, the $h$-mesh local Neumann calculations, which solve for $\epsilon_{0h}$ and $\epsilon_{1h}$, can be evaluated. In addition, the upper and lower bound calculations for $(s_h)_{UB}$ and $(s_h)_{UB}$ can be performed. Numerical procedures regarding the $h$-mesh local Neumann problem and bound calculations have already been discussed in Chapter 5 of this thesis.

6.2 New FETI Procedure on the $h$-mesh

The new FETI approach is applied to calculate the hybrid flux on the fine “truth” mesh $T_h$. Equations (6.3) and (6.3), which are solvable but indeterminate systems, can now be treated as interface problems using FETI procedure.

The superscript $(k)$ denotes each local subdomain problem in this domain decomposition method ($1 \leq k \leq N_k$). Defining $q_{0h}^{(k)}$ and $q_{1h}^{(k)}$ as

$$q_{0h}^{(k)} = -\epsilon^{(k)}_h - L^{(k)}_h T^{(k)}_h \psi^{(k)}_h,$$

(6.4)

$$q_{1h}^{(k)} = f^{(k)}_h - L^{(k)}_h \Theta^{(k)}_h,$$

(6.5)

and denoting $R^{(k)}_h = \text{Ker} \left(A^{(k)}_h\right)$. Then equations (6.1) and (6.3) are reformulated as

$$K^{(k)}_h e^{(k)}_{0h} + B^{(k)}_h^T \lambda_0 = q^{(k)}_{0h},$$

$$\sum_{k=1}^{N_k} B^{(k)}_h e^{(k)}_{0h} = 0,$$

(6.6)
and
\[ K_h^{(k)} e^{(k)}_{1h} + B_h^{(k)^T} \lambda_{1h} = q_{1h}^{(k)}, \]
\[ \sum_{k=1}^{N_k} B_h^{(k)} e_{1h}^{(k)} = 0, \]  
(6.7)
where \( K_h^{(k)} = 2 A_h^{(k)} \). Substituting equation (6.6) into equation (6.7) and enforcing solvability, then
\[ \sum_{k=1}^{N_k} B_h^{(k)} e_{0h}^{(k)} = \sum_{k=1}^{N_k} B_h^{(k)} \left[ K_h^{(k)+} \left( q_{0h}^{(k)} - B_h^{(k)^T} \lambda_{0h}^{(k)} \right) + R_h^{(k)} \alpha_{0h}^{(k)} \right] = 0. \]  
(6.8)
and
\[ \sum_{k=1}^{N_k} B_h^{(k)} e_{1h}^{(k)} = \sum_{k=1}^{N_k} B_h^{(k)} \left[ K_h^{(k)+} \left( q_{1h}^{(k)} - B_h^{(k)^T} \lambda_{1h}^{(k)} \right) + R_h^{(k)} \alpha_{1h}^{(k)} \right] = 0. \]  
(6.9)
where \( \alpha_{h}^{(k)} \) is the set of amplitudes that specifies the contribution of the \( R_h^{(k)} \) to the solution. These coefficients can be determined by requiring that each subdomain problem be mathematically solvable — i.e., each floating subdomain be self-equilibrated — which is
\[ \sum_{k=1}^{N_k} R_h^{(k)^T} \left( q_{0h}^{(k)} - B_h^{(k)^T} \lambda_{0h}^{(k)} \right) = 0, \]
\[ \sum_{k=1}^{N_k} R_h^{(k)^T} \left( q_{1h}^{(k)} - B_h^{(k)^T} \lambda_{1h}^{(k)} \right) = 0. \]  
(6.10)
Rearranging equations (6.8), (6.9) and (6.10) gives
\[ B_h^{(k)} K_h^{(k)+} B_h^{(k)^T} \lambda_{0h} - B_h^{(k)} R_h^{(k)} \alpha_{0h}^{(k)} = B_h^{(k)} K_h^{(k)+} q_{0h}^{(k)}. \]
\[ -R_h^{(k)^T} B_h^{(k)^T} \lambda_{0h} = -R_h^{(k)^T} q_{0h}^{(k)}. \]  
(6.11)
and
\[ B_h^{(k)} K_h^{(k)+} B_h^{(k)^T} \lambda_{1h} - B_h^{(k)} R_h^{(k)} \alpha_{1h}^{(k)} = B_h^{(k)} K_h^{(k)+} q_{1h}^{(k)}, \]
\[ -R_h^{(k)^T} B_h^{(k)^T} \lambda_{1h} = -R_h^{(k)^T} q_{1h}^{(k)}. \]  
(6.12)
This leads to the FETI interface problem, i.e. equations (6.11) and (6.12). in matrix form are:
\[ \begin{bmatrix} F_h & -G_h \\ -G_h^T & 0 \end{bmatrix} \begin{bmatrix} \lambda_{0h} \\ \alpha_{0h} \end{bmatrix} = \begin{bmatrix} d_{0h} \\ -e_{0h} \end{bmatrix}, \]  
(6.13)
and
\[
\begin{bmatrix}
F_h & -G_h \\
-G_h^T & 0
\end{bmatrix}
\begin{bmatrix}
\lambda_{1h} \\
\alpha_{1h}
\end{bmatrix}
= \begin{bmatrix}
d_{1h} \\
e_{1h}
\end{bmatrix},
\] (6.14)

where each of these terms is given by
\[
F_h = \sum_{k=1}^{N_k} B_h^{(k)} K_h^{(k)+} B_h^{(k)T},
\]
\[
G_h = \begin{bmatrix}
B_h^{(1)} R_h^{(1)} & \ldots & B_h^{(N_k)} R_h^{(N_k)}
\end{bmatrix},
\]
\[
\alpha_{0h} = \begin{bmatrix}
\alpha_{0h}^{(1)} \\
\vdots \\
\alpha_{0h}^{(N_k)}
\end{bmatrix},
\]
\[
\alpha_{1h} = \begin{bmatrix}
\alpha_{1h}^{(1)} \\
\vdots \\
\alpha_{1h}^{(N_k)}
\end{bmatrix},
\]
\[
d_{0h} = \sum_{k=1}^{N_k} B_h^{(k)} K_h^{(k)+} q_{0h}^{(k)},
\]
\[
d_{1h} = \sum_{k=1}^{N_k} B_h^{(k)} K_h^{(k)+} q_{1h}^{(k)},
\]
\[
e_{0h} = \begin{bmatrix}
R_h^{(1)} q_{0h}^{(1)} \\
\vdots \\
R_h^{(N_k)} q_{0h}^{(N_k)}
\end{bmatrix},
\]
\[
e_{1h} = \begin{bmatrix}
R_h^{(1)} q_{1h}^{(1)} \\
\vdots \\
R_h^{(N_k)} q_{1h}^{(N_k)}
\end{bmatrix},
\] (6.18)

where \(K_h^{(k)+}\) is a generalized inverse of \(K_h^{(k)}\). The constraints \(G_h^T\lambda_{0h} = e_{0h}\) and \(G_h^T\lambda_{1h} = e_{1h}\) ensure that \((-\varepsilon_h^{(k)} - L_h^{(k)} T_h^{(k)}) \in \text{range } A_h^{(k)}\) and \((f_h^{(k)} - L_h^{(k)} T_h^{(k)}) \in \text{range } A_h^{(k)}\) for all \(k = 1, \ldots, N_k\).

The New FETI method iterates on \(\lambda_{0h}\) and \(\lambda_{1h}\), given an initial \(\lambda_{0h}^0\) and \(\lambda_{1h}^0\) which satisfy the constraints \(G_h^T\lambda_{0h} = e_{0h}\) and \(G_h^T\lambda_{1h} = e_{1h}\). The FETI algorithm can be considered as a two-step pre-conditioned conjugate gradient method to calculate the interface problem and it is already summarized in Chapter 3 of this thesis. Note that in equations (6.13) and (6.14) only the right hand side is different.

### 6.3 Numerical Validation

In this section, the convection–diffusion equation is revisited with the fine \(h\)-mesh hybrid flux calculations for the cube case where \(U_i = (1, 1, 1)\), \(f = 1\), and \(\varepsilon = 0.1\). The second output \(s^{(2)}\) of interest, which is the average solution on corner cube \(\Omega^O\) is reconsidered.

The convergence rate of the coarse mesh solution \(s_H\), upper and lower bounds (i.e. \((s_h)_{UB}\) and \((s_h)_{LB}\)), and the output predictor \((s_h)_{pre}\) with the fine \(h\)-mesh hybrid flux calculations
Table 6.1: Output predictor \((s_h)_{\text{pre}}\), half bound gap \(\Delta(T_H)\), and relative (half) bound gap \(\theta\) for a sequence of refinements.

<table>
<thead>
<tr>
<th>Refinement (R)</th>
<th>(T_{(H,6)})</th>
<th>(T_{(H,9)})</th>
<th>(T_{(H,12)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>1,296</td>
<td>4,374</td>
<td>10,368</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0.1769</td>
<td>0.1900</td>
</tr>
<tr>
<td>3</td>
<td>0.1245</td>
<td>0.0693</td>
<td>0.0333</td>
</tr>
<tr>
<td>(\theta)</td>
<td>70.4</td>
<td>31.3</td>
<td>16.7</td>
</tr>
</tbody>
</table>

Table 6.2: Output predictor \((s_h)_{\text{pre}}\), half bound gap \(\Delta(T_H)\), and relative (half) bound gap \(\theta\) with the optimal stabilization parameter \(\kappa^*\) for a sequence of refinements.

<table>
<thead>
<tr>
<th>Refinement (R)</th>
<th>(T_{(H,6)})</th>
<th>(T_{(H,9)})</th>
<th>(T_{(H,12)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>1,296</td>
<td>4,374</td>
<td>10,368</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0.1769</td>
<td>0.1900</td>
</tr>
<tr>
<td>3</td>
<td>0.1245</td>
<td>0.0693</td>
<td>0.0333</td>
</tr>
<tr>
<td>(\kappa^*)</td>
<td>2.89</td>
<td>3.0</td>
<td>3.07</td>
</tr>
</tbody>
</table>

are verified. To study the convergence, three different coarse subdivisions are used: i.e.\(T_{(H,6)}\), \(T_{(H,9)}\), and \(T_{(H,12)}\). Note that, different refinement \((R)\) of local subdomain is used for each subdivision.

Various results of bound calculations are now illustrated. To start, two different meshes, which are an adaptively refined meshes staring from a structured initial mesh, and an adaptively refined meshes starting from an unstructured initial mesh are used for computations. Note that, the refinement of local subdomain is again fixed at 6 \((R = 6)\) for all the bound calculations. When \(R = 6\), the local subdomain has 84 nodes and 216 tetrahedral elements. In the adaptive process, elements are selected for refinement based on equation (4.7) with \(\alpha = 0.4\) for all the refinements. The global FETI tolerance is set to \(10^{-3}\) similarly to previous coarse \(H\)-mesh hybrid flux calculations.
6.3.1 Convergence Rate of the Average Corner Solution Output $s^{(2)}$

Table 6.1 reports the output predictor $(s_h)_{\text{pre}}$, the half bound gap $\Delta(T_H)$, and the relative (half) bound gap $\theta$ without optimal stabilization parameter $\kappa^*$ for a sequence of refinements. Similarly, Table 6.2 describes for the optimal stabilization parameter $\kappa^*$. The new hybrid flux calculations on the fine $h$-mesh using optimal bound method yield a reduction of relative (half) bound gap $\theta$ from 43.4% to 9.8%. The optimal stabilization parameter $\kappa^*$ is converged to the 3.07 as $H \to h$. Without the optimal stabilization parameter it only changes from 70.1% to 16.7%. Recall that, the previous coarse $H$-mesh hybrid flux calculations provide the reduction up to 16.5% with the optimal stabilization parameter $\kappa^*$. Note that, the fine $h$-mesh hybrid flux calculations provide 22.1% to 6.7% improvements relative to the bound gap over the coarse $H$-mesh hybrid flux calculations.

To validate the output $s^{(2)}$, the convergence rate for different coarse “working” meshes are reported in Figure 6.1 and 6.2. Figure 6.1a shows that $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{\text{pre}}/s_h$ as a function of $H$ and Figure 6.1b illustrates $|(s_h)_{UB} - s_h|$, $|(s_h)_{LB} - s_h|$, $|s_H - s_h|$, and $|(s_h)_{\text{pre}} - s_h|$ as a function of $H$ in log-log scale when the thermal diffusivity is 0.1 ($\varepsilon = 0.1$). Similarly, Figure 6.2 presents for optimal stabilization parameter $\kappa^*$. The optimal bounds in Figure 6.2a are much sharper than the non-optimal bounds in Figure 6.1a.

The slopes of $\log((s_h)_{UB} - s_h)/\log H$, $\log|s_H - s_h|/\log H$, and $\log|(s_h)_{LB} - s_h|/\log H$ are converged to 2.10, 2.05 and 2.04 with the non-optimal bound method. The slopes of $\log((s_h)_{UB} - s_h)/\log H$, $\log|s_H - s_h|/\log H$, and $\log|(s_h)_{LB} - s_h|/\log H$ are converged to 2.26, 2.05 and 2.08 with the optimal bound method. As it is noticed from these results, the bounds converge to $O(H^2)$ as $H \to h$ conforming with theory [16].

The hybrid flux calculations on the fine $h$-mesh provide not only better normalized bounds (i.e. $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{\text{pre}}/s_h$) but also better slopes of the convergence rate (i.e. $\log((s_h)_{UB} - s_h)/\log H$, $\log((s_h)_{LB} - s_h)/\log H$, $\log|s_H - s_h|/\log H$) but at a higher computational cost.
Figure 6.1: (a) Plots of \( \frac{(s_h)_{UB}}{s_h} \), \( \frac{(s_h)_{LB}}{s_h} \), \( \frac{s_H}{s_h} \), and \( \frac{(s_h)_{pre}}{s_h} \) as a function of \( H \) for the output \( s^{(2)} \) when \( \varepsilon = 0.1 \) (top), (b) Plots of \( |(s_h)_{UB} - s_h| \), \( |(s_h)_{LB} - s_h| \), \( |s_H - s_h| \), and \( |(s_h)_{pre} - s_h| \) as a function of \( H \) in log-log scale for the output \( s^{(2)} \) when \( \varepsilon = 0.1 \) (bottom).
Figure 6.2: (a) Plots of $(s_h)_{UB}/s_h$, $(s_h)_{LB}/s_h$, $s_H/s_h$ and $(s_h)_{pre}/s_h$ with optimal stabilization parameter $\kappa^*$ as a function of $H$ for the output $s^{(2)}$ when $\varepsilon = 0.1$ (top). (b) Plots of $|(s_h)_{UB} - s_h|$, $|s_H - s_h|$, $|(s_h)_{LB} - s_h|$, and $|(s_h)_{pre} - s_h|$ with optimal stabilization parameter $\kappa^*$ as a function of $H$ in log-log scale for the output $s^{(2)}$ when $\varepsilon = 0.1$ (bottom).
6.3.2 Summary of the \( h \)-mesh Hybrid Flux Calculations of Adaptive Mesh

In this section, numerical validation for the hybrid flux calculations on the fine \( h \)-mesh is presented for adaptively refined meshes. The relative half bound gap is summarized for adaptive refinement of a structured initial mesh and of an unstructured initial mesh. For more detail results of the fine \( h \)-mesh hybrid flux calculations using adaptively refined meshes refer to Appendix B.

The relative bound gap \( \theta \) as a function of the number of elements using adaptive refinement of a structured initial mesh is presented in Figure 6.3a and using adaptive refinement of an unstructured initial mesh is shown in Figure 6.3b, respectively. Figure 6.3a shows that the improvement of relative bound gap \( \theta \) of adaptive refined structured mesh using optimal bound method is achieved from 27% to 3.9% over using non-optimal bound method. Similarly from Figure 6.3b, the relative bound gap \( \theta \) with adaptive refinement of an unstructured initial mesh with optimal stabilization parameter \( \kappa^* \) improves from 16.7% to 1.7% over without optimal stabilization parameter \( \kappa^* \). Figure 6.4 illustrates the comparison of optimal relative bound gap \( \theta \) between adaptive refinement of a structured initial mesh and of an unstructured initial mesh. The optimal bound method using adaptive refinement of an unstructured initial mesh presents smaller relative bound gap \( \theta \) from 13.3% to 2.3% than using that of a structured initial mesh. Note that, the adaptive refinement always provides a sharper bound gap than the uniform refinement. The both adaptive refinement of a structured and an unstructured initial mesh with optimal bound method present better performance than those with non-optimal bound method. In terms of bounds or relative bound gap, the adaptive refinement of an unstructured initial mesh always gives better results than that of a structured initial mesh.

The fine \( h \)-mesh hybrid flux calculations provide better bounds and the relative bound gap. In terms of relative bound gap using the optimal bound method, the final adapted mesh (7,832 elements) of a structured initial mesh provides 8.7% and the final adapted mesh (8,971 elements) of an unstructured initial mesh performs 6.4%. Recall that for the coarse \( H \)-mesh hybrid flux calculations using the optimal bound method the final adapted mesh (9,601 elements) of a structured initial mesh gives 24.4% and the final adapted mesh (9,739 elements) of an unstructured initial mesh provides 18.2% of the relative bound gap \( \theta \), respectively.

It is noticed that the optimal bound method with adaptive refinement of an unstructured
Figure 6.3: Relative bound gap $\theta$ as a function of the number of elements in log-log scale: (a) Summary for adaptive refinement of a structured initial mesh (top) and (b) Summary for adaptive refinement of an unstructured initial mesh (bottom).
initial mesh is the most desirable methodology to improve the sharpness of bounds and to distribute the local bound gap error more uniformly. In addition, the direct calculations of hybrid flux on the fine $h$-mesh always presents sharper bounds than $H$-mesh hybrid calculations but at a higher computational expense.
Chapter 7

Realistic Test Case

7.1 Problem Statement

In this section a more realistic application is addressed. The convection-diffusion equation is applied for computing the temperature field in a rectangular duct with a given laminar velocity field of which the analytical solution is available.

The optimal and non-optimal bound method is applied to the three dimensional convection-diffusion equation in a rectangular duct geometry. The adaptive mesh refinement technique is applied to reduce the bound gap and therefore improve the validated prediction for the output. The calculation for inter-subdomain connectivity (i.e hybrid flux) are performed both on the coarse "working" $H$-mesh and on the fine "truth" $h$-mesh.

7.1.1 Governing Equation

The convection-diffusion equation in three space dimensions are used to describe the behavior of temperature $\Theta$ in a rectangular duct flow. The temperature, $\Theta$, satisfies

$$-\frac{\partial}{\partial x_i}(\varepsilon \frac{\partial \Theta}{\partial x_i}) + U_i \frac{\partial \Theta}{\partial x_i} = f \text{ in } \Omega, \ i = 1, 2, 3. \tag{7.1}$$

with boundary conditions

$$\Theta = \Theta_i, \text{ on } \Gamma_1, \tag{7.2}$$

$$\Theta = 0, \text{ on } \Gamma_2 \text{ and } \Gamma_4, \tag{7.3}$$

$$\varepsilon \frac{\partial \Theta}{\partial n} = q_j, \ j = 1, \ldots, 4, \text{ on } \Gamma_3, \tag{7.4}$$

$$\frac{\partial \Theta}{\partial n} = 0, \text{ on } \Gamma_5, \tag{7.5}$$
where $\varepsilon$ is the thermal diffusivity and $\Omega$ is a bounded domain in $\mathbb{R}^3$ in which $\Gamma_1$ is inhomogeneous Dirichlet boundary, $\Gamma_2$ and $\Gamma_4$ are homogeneous Dirichlet boundary, $\Gamma_3$ is inhomogeneous Neumann boundary, and $\Gamma_5$ is homogeneous Neumann boundary.

For this problem $x = (x_1, x_2, x_3)$ with corresponding unit vector $\hat{x}_1, \hat{x}_2, \hat{x}_3$. $\Omega$ is the rectangular duct, $[0,1] \times [0,1] \times [0,4]$ and outer sides of duct are denoted $\Gamma_j$, $j = 1, ..., 5$. as shown in Figure 7.1. The inlet temperature is given as $\Theta_1 = \sin(\pi x_1) \times \sin(\pi x_2)$ and the heat flux is given as $q_{1,3} = 0.5 \times \sin(\pi x_2) \times \sin[\pi (x_3 - 1)]$ and $q_{2,4} = 0.5 \times \sin(\pi x_1) \times \sin[\pi (x_3 - 1)]$. To avoid singularities on each side of this geometry, a sinusoidal distribution of inlet temperature and heat flux is given. The inlet velocity is prescribed as $U_i$, which is described analytically in Section 7.1.3. The force term $f = 0$, assumes no heat source inside the volume.

The variational weak form of the governing equation is as follows: Find $\Theta \in \mathcal{H}^1(\Omega)$ such that

$$
\int_{\Omega} \varepsilon \frac{\partial v}{\partial x_j} \frac{\partial \Theta}{\partial x_j} + vU_j \frac{\partial \Theta}{\partial x_j} \, dV = \int_{\Gamma_3} vq \, dA, \quad \forall v \in \mathcal{H}^1_0(\Omega).
$$

(7.6)

where $dV$ is a differential volume element and $dA$ is a differential area element. The right hand side of equation (7.6) is a inhomogeneous Neumann boundary term arising from left hand side.
CHAPTER 7. REALISTIC TEST CASE

Note that, the velocity field \( U_i \) is not uniform and needs to be integrated. Chapter 2 describes more details about the integration procedure of \( U_i \).

7.1.2 Output of Interest

The particular linear functional that is of interest is the average temperature output denoted by \( s^{(3)} \) on a specific slice (denoted as \( \Gamma^O \)) of the domain \( \Omega \). Figure 7.1 indicates the surface \( \Gamma^O \), which is \([0, 1[\times]0, 1[ \) at \( x_3 = 2.0 \).

The average temperature output linear functional is simply written as

\[
s^{(3)} = \ell^{(3)}(\Theta) = \frac{1}{|\Gamma^O|} \int_{\Gamma^O} \Theta dA, \tag{7.7}
\]

where \( |\Gamma^O| \) is the area of the surface \( \Gamma^O \). The value of \( |\Gamma^O| = 1 \) in this case.

7.1.3 Velocity Field

The velocity distribution is prescribed as \( U_i \) and is obtained analytically. It is assumed that the secondary flow in the rectangular duct is negligible, that is velocities in \( x_1 \) and \( x_2 \) direction are negligible if they are compared with velocity in \( x_3 \)-direction. The velocity field is laminar and it is driven by a negative pressure gradient.

The velocity in the \( x_3 \)-direction has a parabolic distribution and it is shown in Figure 7.2. The analytical solution for a rectangular duct flow is formulated by

\[
U_1 = U_2 = 0, \\
U_3(x_1, x_2, x_3) = -\frac{2}{\mu} \frac{d\rho}{dx_3} \left[ \frac{1}{4} \left( \frac{1}{4} - (x_1 - 1/2)^2 \right) \right.
\]

\[
-2 \sum_{n=0}^{\infty} \frac{(-1)^n \cosh \lambda_n (x_2 - 1/2)}{\cosh \lambda_n/2} \cos \lambda_n (x_1 - 1/2) \right]. \tag{7.8}
\]

where

\[
\lambda_n = 2 \left( n\pi + \frac{\pi}{2} \right). \tag{7.9}
\]

Note that, the velocity depends on the coordinates \( x_1 \) and \( x_2 \), \( \mu \) (fluid viscosity), and \( \frac{d\rho}{dx_3} \) (pressure gradient in the \( x_3 \)-direction). The procedure to calculate the analytical solution for the rectangular duct flow is described in Appendix C.
Figure 7.2: Velocity isocontours (0 to 1.3 at intervals of 0.1): (a) Velocity distribution as a surface (top), (b) Velocity distribution as isolines (bottom)
7.2 Numerical Validation

In this section, the convection-diffusion equation is investigated for the rectangular duct flow where the thermal diffusivity is 0.1 (i.e. $\varepsilon = 0.1$). The average temperature $s^{(3)}$ over domain $\Gamma^O$ is considered.

First the influence of the coarse “working” $H$-mesh hybrid flux calculations on the various results of bound method is investigated. For this case, the uniform and adaptive refinement of a structured initial mesh is used. Then the influence of the fine “truth” $h$-mesh hybrid flux calculations on several results of bound method is explored. The uniformly refined meshes and adaptively refined meshes starting from a structured initial mesh are employed for this validation.

All the computations are performed on a single processor with Pentium III 933 MHz CPU having 1536M memory. Due to the memory limitation of the computational resource, all the bound calculations are carried out with a refinement of 4 ($R = 4$). When $R = 4$, each local subdomain has 35 nodes and 64 tetrahedral elements. In the adaptive process, elements are selected for refinement based on equation (4.7) with $\alpha = 0.4$ for all the refinements. The global FETI tolerance is set to $10^{-3}$ for both hybrid flux calculations.

7.2.1 $H$-mesh Hybrid Flux Calculations

It is now important to demonstrate the adaptive mesh refinement technique. Here, the hybrid flux calculations are applied on the coarse $H$-mesh. Recall that the goal of the adaptive technique is to improve the sharpness of bound gap and distribute the local bound gap error more uniformly over the entire domain. Results of the optimal bound method using uniform and adaptive refinement of a structured initial mesh is summarized in Table 7.1. Table 7.1 shows that a reduction of the relative (half) bound gap from 51% to 21.6% is achieved by using the optimal adaptive refinement, while the optimal uniform refinement only decreases the relative (half) bound gap from 51% to 24.2% with even more elements ($T^0_{H/2}$ has 24,000 elements) than that of adaptive refinement ($T^0_H$ has 18,678 elements).

Table 7.2 reports the number of elements (%) by half bound gap $\Delta(T_H)$ intervals for uniform and adaptive refinement of a structured initial mesh with optimal stabilization parameter $\kappa^*$. As more adaptive refinement cycles are performed, the maximum value of the half bound
CHAPTER 7. REALISTIC TEST CASE

Table 7.1: Optimal bounds for uniform and adaptive refinement of a structured initial mesh.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^0$</th>
<th>$T_H^1$</th>
<th>$T_H^2$</th>
<th>$T_H^3$</th>
<th>$T_H^4$</th>
<th>$T_H^5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3,000</td>
<td>4,029</td>
<td>5,023</td>
<td>7,926</td>
<td>10,308</td>
<td>18,678</td>
<td>24,000</td>
</tr>
<tr>
<td># of nodes</td>
<td>756</td>
<td>991</td>
<td>1,186</td>
<td>1,823</td>
<td>2,285</td>
<td>4,003</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
<td>1.3909</td>
<td>1.4566</td>
<td>1.4139</td>
<td>1.4234</td>
<td>1.4271</td>
<td>1.4394</td>
</tr>
<tr>
<td>$(s_h)_{LB}$</td>
<td>0.7055</td>
<td>0.8443</td>
<td>0.8772</td>
<td>0.9673</td>
<td>1.0106</td>
<td>1.0797</td>
</tr>
<tr>
<td>$(s_h)_{pre}$</td>
<td>0.4518</td>
<td>0.5998</td>
<td>0.6606</td>
<td>0.7694</td>
<td>0.8312</td>
<td>0.9285</td>
</tr>
<tr>
<td>$\pm \Delta(T_H)$</td>
<td>± 0.4696</td>
<td>± 0.4284</td>
<td>± 0.3766</td>
<td>± 0.3291</td>
<td>± 0.2979</td>
<td>± 0.2554</td>
</tr>
<tr>
<td>$\theta$ (%)</td>
<td>51.0</td>
<td>41.7</td>
<td>36.3</td>
<td>30.0</td>
<td>36.4</td>
<td>21.6</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>1.87</td>
<td>1.76</td>
<td>1.60</td>
<td>1.42</td>
<td>1.39</td>
<td>1.35</td>
</tr>
</tbody>
</table>

Table 7.2: Number of elements (%) by half bound gap $\Delta(T_H)$ intervals for uniform and adaptive refinement of a structured initial mesh.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^0$</th>
<th>$T_H^1$</th>
<th>$T_H^2$</th>
<th>$T_H^3$</th>
<th>$T_H^4$</th>
<th>$T_H^5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3,000</td>
<td>116 (3.9)</td>
<td>12 (0.2)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>3 (0)</td>
<td>52 (0.2)</td>
</tr>
<tr>
<td>$5 \times 10^{-4} \leq \Delta(T_H) &lt; 10^{-3}$</td>
<td>105 (3.5)</td>
<td>88 (1.8)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>52 (0.2)</td>
<td></td>
</tr>
<tr>
<td>$10^{-4} \leq \Delta(T_H) &lt; 5 \times 10^{-4}$</td>
<td>540 (18)</td>
<td>918 (18.3)</td>
<td>635 (6.2)</td>
<td>163 (0.9)</td>
<td>361 (1.5)</td>
<td></td>
</tr>
<tr>
<td>$5 \times 10^{-5} \leq \Delta(T_H) &lt; 10^{-4}$</td>
<td>295 (9.8)</td>
<td>738 (14.7)</td>
<td>1,033 (10.0)</td>
<td>763 (4.1)</td>
<td>553 (2.3)</td>
<td></td>
</tr>
<tr>
<td>$10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>781 (26)</td>
<td>1,967 (39.1)</td>
<td>4,610 (44.7)</td>
<td>6,265 (23.5)</td>
<td>3,271 (13.6)</td>
<td></td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>226 (7.5)</td>
<td>362 (7.2)</td>
<td>1,808 (17.5)</td>
<td>4,487 (24)</td>
<td>2,580 (10.8)</td>
<td></td>
</tr>
<tr>
<td>$10^{-6} \leq \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>407 (13.6)</td>
<td>426 (8.5)</td>
<td>1,651 (16.0)</td>
<td>5,934 (31.8)</td>
<td>6,941 (28.9)</td>
<td></td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 10^{-6}$</td>
<td>530 (17.7)</td>
<td>511 (10.2)</td>
<td>571 (5.5)</td>
<td>1,066 (5.7)</td>
<td>10,239 (42.7)</td>
<td></td>
</tr>
</tbody>
</table>

The gap $\Delta(T_H)$ is decreased and the range of the bound gap is more uniformly distributed over the entire domain. It is observed that the final adapted mesh $T_H^5$ (18,678 elements) has 163 elements (0.9%) that have values of bound gap greater than $10^{-4}$ while the uniformly refined mesh $T_H^{3/2}$ (24,000 elements) has 416 elements (1.7%). Note that, these elements cause the increase of bound gap (bound error).

Figure 7.3 illustrates surface meshes generated using uniform and adaptive refinement of a structured initial mesh. Note that, the domain used for the output $s^{(3)}$ (i.e. $[0, 1] \times [0, 1]$ at $z = 2$) is refined as more adaptive refinement cycles are performed (see the adaptively refined meshes in Figure 7.3). Figure 7.4 shows the isocontours of temperature in domain $\Gamma$. 
CHAPTER 7. REALISTIC TEST CASE

for uniform and adaptive refinements of a structured initial mesh. The value of isocontours ranges from 0 to 0.7 at intervals of 0.1. In addition, in Figure 7.4 isocontours of temperature associated with $\mathcal{T}^{(H,36)}_{(1,119,744\text{ elements})}$ are presented as a high resolution solution. As the refinement is performed, the isocontours become smoother. Final isocontours on mesh $\mathcal{T}^{7}_{H}$ (18,678 elements) catch the boundary layer near the wall more precisely than that of uniformly refined mesh $\mathcal{T}^{0}_{H/2}$ (24,000 elements). After a four refinements, the method identifies the mesh in the middle region where there is not much refinement required and thereby the sharpness of bounds or relative bound gap is not improved significantly. Error is mainly due to the interpolation error of the hybrid flux so that the direct calculation of hybrid flux on the fine $h$-mesh should improve the bound gap. This effect is analyzed in the Section 7.2.2.

Figure 7.5 shows the relative bound gap $\theta$ with and without stabilization parameter $\kappa^*$ as a function of the number of elements over domain $\Gamma^O$ using uniform and adaptive refinements of a structured initial mesh. It can be observed from Figure 7.5 that the adaptive refinement of a structured initial mesh using optimal bound method performs best in terms of relative bound gap than any other method. Because of the interpolation error of hybrid flux on the fine $h$-mesh, the relative (half) bound gap $\theta$ is still over 20%. To improve the sharpness of the bound gap and to reduce the relative bound gap $\theta$, not only optimal stabilization parameter $\kappa^*$ but also direct calculations of hybrid flux on the fine "truth" $h$-mesh must be used. In next section the influence of the optimal stabilization parameter and direct calculations of the hybrid flux on the fine $h$-mesh is investigated on the sharpness of bounds and relative (half) bound gap.
Figure 7.3: Uniformly and adaptively refined meshes with optimal bound method: (a) $\mathcal{T}_H^0$ has 3,000 elements, (b) $\mathcal{T}_H^2$ has 5,023 elements, (c) $\mathcal{T}_H$ has 7,926 elements, (d) $\mathcal{T}_H^1$ has 10,308 elements, (e) $\mathcal{T}_H^5$ has 18,678 elements, and (f) $\mathcal{T}_{H/2}^0$ has 24,000 elements.
Figure 7.4: Isocontours (0 to 0.7 at intervals of 0.1) of temperature for uniformly and adaptively refined meshes with optimal bound method (slice at $z=2.0$): (a) $T_H^0$ has 3,000 elements, (b) $T_H$ has 5,023 elements, (c) $T_H^1$ has 10,308 elements, (d) $T_H^2$ has 18,678 elements, (e) $T_{H/2}$ has 24,000 elements, and (f) $T_{(H,36)}$ has 1,119,744 elements.
Figure 7.5: Relative bound gap $\theta$ as a function of the number of elements in log-log scale for uniform and adaptive refinement of a structured initial mesh using the coarse $H$-mesh hybrid flux calculations.
### Table 7.3: Optimal bounds for uniformly refined meshes.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_{(H,5)}$</th>
<th>$T_{(H,6)}$</th>
<th>$T_{(H,7)}$</th>
<th>$T_{(H,8)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td># of nodes</td>
<td>3,000</td>
<td>5,184</td>
<td>8,232</td>
<td>12,288</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
<td>1.2442</td>
<td>1.2278</td>
<td>1.2192</td>
<td>1.2148</td>
</tr>
<tr>
<td>$s_H$</td>
<td>0.7055</td>
<td>0.7818</td>
<td>0.8397</td>
<td>0.8851</td>
</tr>
<tr>
<td>$(s_h)_{LB}$</td>
<td>0.5895</td>
<td>0.6948</td>
<td>0.7719</td>
<td>0.8308</td>
</tr>
<tr>
<td>$(s_h)_{pre}$</td>
<td>0.9168</td>
<td>0.9613</td>
<td>0.9956</td>
<td>1.0228</td>
</tr>
<tr>
<td>$\pm \Delta(T_H)$</td>
<td>$\pm 0.3273$</td>
<td>$\pm 0.2665$</td>
<td>$\pm 0.2236$</td>
<td>$\pm 0.1920$</td>
</tr>
<tr>
<td>$\theta$ (%)</td>
<td>35.7</td>
<td>27.7</td>
<td>22.5</td>
<td>18.8</td>
</tr>
<tr>
<td>$\kappa^*$</td>
<td>2.1</td>
<td>2.1</td>
<td>2.26</td>
<td>2.31</td>
</tr>
</tbody>
</table>

### Table 7.4: Optimal bounds for adaptive refinement of a structured initial mesh.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^A$</th>
<th>$T_H^B$</th>
<th>$T_H^C$</th>
<th>$T_H^D$</th>
</tr>
</thead>
<tbody>
<tr>
<td>total d.o.f.</td>
<td>756</td>
<td>1,067</td>
<td>1,953</td>
<td>2,583</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
<td>1.2442</td>
<td>1.2405</td>
<td>1.2212</td>
<td>1.2253</td>
</tr>
<tr>
<td>$s_H$</td>
<td>0.7055</td>
<td>0.8843</td>
<td>0.9866</td>
<td>1.0166</td>
</tr>
<tr>
<td>$(s_h)_{LB}$</td>
<td>0.5895</td>
<td>0.7830</td>
<td>0.9146</td>
<td>0.9514</td>
</tr>
<tr>
<td>$(s_h)_{pre}$</td>
<td>0.9168</td>
<td>1.0117</td>
<td>1.0679</td>
<td>1.0884</td>
</tr>
<tr>
<td>$\pm \Delta(T_H)$</td>
<td>$\pm 0.3273$</td>
<td>$\pm 0.2287$</td>
<td>$\pm 0.1532$</td>
<td>$\pm 0.1369$</td>
</tr>
<tr>
<td>$\theta$ (%)</td>
<td>35.7</td>
<td>22.6</td>
<td>14.3</td>
<td>12.6</td>
</tr>
<tr>
<td>$\kappa^*$</td>
<td>2.1</td>
<td>2.1</td>
<td>1.86</td>
<td>1.84</td>
</tr>
<tr>
<td>$\kappa^*$</td>
<td>2.1</td>
<td>2.1</td>
<td>1.86</td>
<td>1.84</td>
</tr>
</tbody>
</table>

#### 7.2.2 $h$-mesh Hybrid Flux Calculations

The calculations of hybrid flux are now applied on the fine "truth" $h$-mesh. Results of the optimal bound method using uniformly refined meshes is summarized in Table 7.3. Similarly, Table 7.4 reports optimal bounds for adaptive refinement of a structured initial mesh. Table 7.3 and Table 7.4 show a reduction of the relative (half) bound gap from 35.7% to 9.8% by using optimal adaptive refinement of a structured initial mesh. The third adapted mesh $T_H^A$ (11,698 elements) gives 12.6% relative bound gap $\theta$ while the subdivision $T_{(H,8)}$ (12,228 elements) provides only 18.8% with even more elements. Recall that the relative bound gap $\theta$ with the coarse $H$-mesh hybrid flux calculations achieved only 21.6% on the final adapted mesh $T_H^A$ (18,678 elements). Note that, the direct calculations of hybrid flux on the fine $h$-mesh
Table 7.5: Number of elements (%) by half bound gap $\Delta(T_H)$ intervals for uniformly refined meshes.

<table>
<thead>
<tr>
<th>$\Delta(T_H)$</th>
<th>$T_{(H,5)}$</th>
<th>$T_{(H,6)}$</th>
<th>$T_{(H,7)}$</th>
<th>$T_{(H,8)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\geq 10^{-3}$</td>
<td>3,000</td>
<td>5,184</td>
<td>8,232</td>
<td>12,288</td>
</tr>
<tr>
<td>$5 \times 10^{-4} \leq \Delta(T_H) &lt; 10^{-3}$</td>
<td>80 (2.7)</td>
<td>55 (1.1)</td>
<td>25 (0.3)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$10^{-4} \leq \Delta(T_H) &lt; 5 \times 10^{-4}$</td>
<td>390 (13)</td>
<td>349 (6.7)</td>
<td>305 (3.7)</td>
<td>294 (2.4)</td>
</tr>
<tr>
<td>$5 \times 10^{-5} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>299 (10)</td>
<td>374 (7.2)</td>
<td>303 (3.7)</td>
<td>270 (2.2)</td>
</tr>
<tr>
<td>$10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>871 (29)</td>
<td>1,248 (24.1)</td>
<td>1,587 (19.3)</td>
<td>1,669 (13.6)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-6}$</td>
<td>270 (9)</td>
<td>777 (15.0)</td>
<td>980 (11.9)</td>
<td>1,203 (9.8)</td>
</tr>
<tr>
<td>$10^{-6} \leq \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>346 (11.5)</td>
<td>922 (17.8)</td>
<td>2,307 (28)</td>
<td>3,870 (31.5)</td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 10^{-6}$</td>
<td>655 (21.8)</td>
<td>1,394 (26.9)</td>
<td>2,643 (32.1)</td>
<td>4,902 (39.9)</td>
</tr>
</tbody>
</table>

Table 7.6: Number of elements (%) by half bound gap $\Delta(T_H)$ for adaptively refined meshes.

<table>
<thead>
<tr>
<th>$\Delta(T_H)$</th>
<th>$T^0_H$</th>
<th>$T^1_H$</th>
<th>$T^2_H$</th>
<th>$T^3_H$</th>
<th>$T^4_H$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\geq 10^{-3}$</td>
<td>3,000</td>
<td>4,478</td>
<td>8,734</td>
<td>11,698</td>
<td>15,802</td>
</tr>
<tr>
<td>$5 \times 10^{-4} \leq \Delta(T_H) &lt; 10^{-3}$</td>
<td>80 (2.7)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$10^{-4} \leq \Delta(T_H) &lt; 5 \times 10^{-4}$</td>
<td>390 (13)</td>
<td>575 (12.8)</td>
<td>219 (2.5)</td>
<td>110 (0.9)</td>
<td>3 (0)</td>
</tr>
<tr>
<td>$5 \times 10^{-5} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>299 (10)</td>
<td>478 (10.7)</td>
<td>579 (6.6)</td>
<td>421 (3.6)</td>
<td>171 (1.1)</td>
</tr>
<tr>
<td>$10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>871 (29)</td>
<td>1,781 (40)</td>
<td>2,788 (31.9)</td>
<td>2,973 (25.4)</td>
<td>2,910 (18.4)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-6}$</td>
<td>270 (9)</td>
<td>525 (11.7)</td>
<td>1,910 (21.9)</td>
<td>2,298 (19.6)</td>
<td>2,807 (17.8)</td>
</tr>
<tr>
<td>$10^{-6} \leq \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>346 (11.5)</td>
<td>467 (10.4)</td>
<td>2,469 (28.3)</td>
<td>4,661 (39.8)</td>
<td>7,477 (47.3)</td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 10^{-6}$</td>
<td>655 (21.8)</td>
<td>613 (13.7)</td>
<td>769 (8.8)</td>
<td>1,235 (10.6)</td>
<td>2,434 (15.4)</td>
</tr>
</tbody>
</table>

provide over 10% reduction in terms of relative (half) bound gap $\theta$.

Table 7.5 reports the number of elements (%) by bound gap $\Delta(T_H)$ intervals with optimal stabilization parameter $\kappa^*$ using uniformly refined meshes. Similarly, the number of elements (%) by half bound gap $\Delta(T_H)$ intervals using optimal adaptive refinement of a structured initial mesh is summarized in Table 7.6. The third adapted mesh $T^3_H$ (11,698 elements) has only 110 elements (0.9%) which have values of local bound gap error greater than $10^{-1}$ while the subdivision $T_{(H,8)}$ (12,288 elements) has 324 elements (3.1%) and these elements cause the higher value of bound gap (bound error). As the adaptive refinement sequence proceeds, the maximum value of half bound gap $\Delta(T_H)$ is decreased and local bound gap error is more
uniformly distributed over the entire domain. Note that, the bound gap (or local bound gap error) distribution obtained from uniformly refined meshes contribute almost 4 (3.7) times higher maximum bound gap than that of adaptive refinement. It is because the adaptive refinement technique refines meshes only where higher local bound gap error occurs.

Figure 7.6 and 7.7 shows uniformly refined meshes and adaptively refined meshes, respectively. The domain \( \Gamma^O \) (i.e. \([0,1] \times [0,1] \) at \( z = 2 \)) is identified and refined as the adaptive refinement cycle proceeds. Isocontours of temperature (from 0 to 0.7 at intervals) in domain \( \Gamma^O \) for uniformly refined meshes with optimal bound method and for adaptively refined mesh with optimal bound method are shown in Figure 7.8 and Figure 7.9, respectively. As the refinement cycle progresses, the isocontours become smoother especially near the wall. Final isocontours of adapted mesh \( \mathcal{T}_H \) (15,802 elements) catch the boundary layer near the wall more precisely than that of the subdivision \( \mathcal{T}_{(H,3)} \) (17,496 elements).

The relative bound gap \( \theta \) as a function of the number of elements is illustrated with and without optimal stabilization parameter in Figure 7.10. Clearly it is observed from Figure 7.10 that the relative (half) bound gap \( \theta \) for optimal adaptive refinement of a structured initial mesh performs better than not only that of uniformly refined meshes but also that of non-optimal adaptive refinement of a structured initial mesh. Since the computations of the hybrid flux are directly performed on the fine \( h \)-mesh, the sharpness of bound gap is improved and the relative (half) bound gap under 10% (9.8%) is achieved. Nevertheless there is a higher computational cost associated with these calculations. It is most likely that furthermore improvement to the sharpness of the bounds can be obtained for additional refinement cycles. However due to the memory limitation of the computational resource used herein, the further refinement could not be performed.
Figure 7.6: Uniformly refined meshes: (a) $\mathcal{T}_{(H,5)}$ has 3,000 elements, (b) $\mathcal{T}_{(H,8)}$ has 5,184 elements, (c) $\mathcal{T}_{(H,7)}$ has 8,232 elements, (d) $\mathcal{T}_{(H,8)}$ has 12,288 elements, and (e) $\mathcal{T}_{(H,9)}$ has 17,496 elements.
Figure 7.7: Adaptively refined meshes: (a) $\mathcal{T}_H^0$ has 3,000 elements, (b) $\mathcal{T}_H^1$ has 4,478 elements, (c) $\mathcal{T}_H^2$ has 8,734 elements, (d) $\mathcal{T}_H^3$ has 11,698 elements, and (e) $\mathcal{T}_H^4$ has 15,802 elements.
Figure 7.8: Isocontours (0 to 0.7 at intervals of 0.1) of temperature for uniformly refined meshes with optimal bound method (slice at z=2.0): (a) $\mathcal{T}_{(H,5)}$ has 3,000 elements, (b) $\mathcal{T}_{(H,6)}$ has 5,184 elements, (c) $\mathcal{T}_{(H,7)}$ has 8,232 elements, (d) $\mathcal{T}_{(H,8)}$ has 12,288 elements, (e) $\mathcal{T}_{(H,9)}$ has 17,496 elements, and (f) $\mathcal{T}_{(H,36)}$ has 1,119,744 elements.
Figure 7.9: Isocontours (0 to 0.7 at intervals of 0.1) of temperature for adaptively refined meshes with optimal bound method (slice at z=2.0): (a) $\mathcal{T}_H^2$ has 3,000 elements, (b) $\mathcal{T}_H^4$ has 4,478 elements, (c) $\mathcal{T}_H^8$ has 8,734 elements, (d) $\mathcal{T}_H^{16}$ has 11,698 elements, (e) $\mathcal{T}_H^{32}$ has 15,802 elements, and (f) $\mathcal{T}_{(H,36)}$ has 1,119,744 elements.
Figure 7.10: Relative bound gap $\theta$ as a function of the number of elements in log-log scale for uniformly refined meshes and adaptively refined meshes using the fine $h$-mesh hybrid flux calculations.
Chapter 8

Conclusion

8.1 Summary and Limitations

In this thesis an a posteriori finite element bound method with adaptive mesh refinement technique is investigated for linear-functional outputs of convection-diffusion equation in three space dimensions.

The bound method in this thesis is based on the modified energy objective—hence modified Lagrangian which gives information regarding the contribution of each element to the bound gap. The adaptive mesh refinement technique refines the meshes where the error is concentrated and thereby controls the errors of the elements in such a way that the output of interest is bounded more accurately. The goal of adaptive methodology is to reduce the bound gap and distribute the local bound gap error more uniformly. An optimal stabilization parameter $\kappa^*$ is applied to optimize the bounds and thereby improve the sharpness of the bound gap. Finally the direct calculations of hybrid flux on the fine "truth" $h$-mesh are presented to reduce the bound gap furthermore.

The resulting methods are applied to three dimensional convection-diffusion problems. First, validation for the convection-diffusion equation applied to a cube, has been investigated. The average solution output $s^{(1)}$ and the average corner solution output $s^{(2)}$ are considered as the linear functional outputs of interest. It is observed that in most cases the bounds converge to $O^2(H)$ as $H \rightarrow h$ conforming with theory [16]. Second, demonstration of the method on a more realistic problem, describing the temperature in a rectangular duct flow, was investigated. Boundary conditions include inlet temperature $\Theta_i$ and heat flux $q_j$ as sinusoidal distribution. The average surface solution output $s^{(3)}$ in domain $\Gamma^O$ (i.e. section of the duct $]0,1[\times]0,1[$ at
Consider $z = 2.0$ as output.

Adaptive mesh refinement is applied to a structured initial mesh and an unstructured initial mesh. In particular, some of key features of the bound method with adaptive mesh refinement technique are as follows:

- Bounds for the adaptively refined mesh always presents sharper values than that of the uniformly refined mesh.

- Bounds calculated with the optimal stabilization parameter $\kappa^*$ gives better results than bounds without the optimal stabilization parameter.

- For the validation test case with the coarse $H$-mesh hybrid flux calculations using optimal bound method, the half bound gap within $\pm 24.4\%$ are obtained for the final adapted mesh (9,601 elements) of a structured initial mesh. Similarly, the half bound gap within $\pm 18.2\%$ are achieved for the final adapted mesh (9,739 elements) of an unstructured initial mesh.

- Calculations of hybrid flux on the fine $h$-mesh improves the sharpness of bounds but at a higher computational cost.

- For the validation test case with the hybrid flux calculations on the fine $h$-mesh using optimal bound method, the final adapted mesh (7,832 elements) of a structured initial mesh provides the half bound gap within $\pm 8.7\%$. Similarly, the final adapted mesh (8,971 elements) of a unstructured initial mesh performs the half bound gap within $\pm 6.4\%$.

- For the realistic test case using optimal bound method, the coarse $H$-mesh hybrid flux calculations provides the half bound gap within $\pm 21.6\%$ for the final adapted mesh (18,678 elements).

- Finally, the hybrid flux calculations on the fine $h$-mesh for the final adapted mesh (15,802 elements) performs the half bound gap within $\pm 9.8\%$.

Because of memory limitation, the further refinement was not performed herein. However, the sharpness of bounds and relative bound gap can be improved further on large computers or when using distributed memory computers.
8.2 Future Work

I have classified into three main areas of future improvement. These areas are related to the computer science, numerical method, and engineering.

In view of computer science, the parallel computation should be implemented. The domain decomposition intrinsic to the bound method enables a straightforward parallelization. Parallel processing can be applied not only the fine $h$-mesh local Neumann computation but also the coarse $H$-mesh field variable and adjoint calculation. Note that the hybrid flux calculations on the fine $h$-mesh, which requires huge computational resource, may be solved in parallel and thereby allows large speedups. Parallelization can be performed using existing parallelization algorithms and software libraries such as MPI. Furthermore, better adaptively generated mesh can be constructed to improve the sharpness of the bound gap. To reduce the stretched elements, i.e. a way to improve the sharpness of bounds or relative bound gap, the technique such as smoothing the faces or volumes and relocating the nodes can be applied.

In the field of numerical methods in particular the finite element method, the order of elements can be increased. The linear reference elements can be extended to quadratic reference elements. The bounds in present linear finite element space perform at the order of 2. $O(H^2)$ in convergence rate as $H \rightarrow h$. Bounds in quadratic finite element space will be converged in order of 3, $O(H^3)$ as $H \rightarrow h$. In addition, the algorithms in particular the hybrid flux calculations can be improved. Such algorithm improves the accuracy of hybrid flux interpolations on the fine $h$-mesh while keeping the hybrid flux calculations on the coarse $H$-mesh.

Regarding engineering extensions, the bound method with adaptive refinement technique will be extended to handle the three dimensional Navier-Stokes equations. Presently the three dimensional convection-diffusion equation is considered with a given velocity field. For Navier-Stokes problem, not only velocity field but also pressure field will be calculated. The bound method with adaptive mesh refinement technique can also be applied to treat the Structure or Acoustic problems.
Bibliography


Appendix A

Optimal Variational Bound Formulation

The strategy is to write all variables as linear functions in $\kappa$ and then derive the bounds as a function in $\kappa$. This procedure does not change the bounding theory and the bounds remain rigorous. The following trivial minimization statement can be rewritten as,

$$\pm s_h = \min_{v \in S} \left[ \kappa(a^*(v, v) - \ell^E(v)) \pm \ell^O(\Theta_H + v) \right]. \quad (A.1)$$

where $\kappa \in \mathcal{R}_+$ is a non-negative parameter that will be used later to optimize the computed bounds.

The quadratic Lagrangian $\mathcal{L}: \hat{X}_h \times X_h \times Q_h \rightarrow \mathcal{R}$, is reformulated as follows.

$$\mathcal{L}^\pm(v, \mu, q) = \kappa(a^*(v, v) - \ell^E(v)) \pm \ell^O(\Theta_H + v)$$

$$+ [a(\mu, \Theta_H + v) - \ell^N(\mu)] + b(v, q). \quad (A.2)$$

The expansion of equation (A.2) becomes,

$$\mathcal{L}^\pm(e_H^\pm + w, \psi_H^\pm, \lambda_H^\pm) = \mathcal{L}^\pm(e_H^\pm, \psi_H^\pm, \lambda_H^\pm) + a^*(w, w)$$

$$+ \kappa \left[2a^*(w, e_H^\pm) - \ell^E(w)\right] \pm \ell^O(w)$$

$$+ a(\psi_H^\pm, w) + b(w, \lambda_H^\pm). \quad (A.3)$$

where the first variation of the Lagrangian must vanish.
H-mesh Adjoint Calculation

The stationarity conditions for equation (A.2) leads to the following statement:

Find $e_H^\pm \in \hat{X}_H, \psi_H^\pm \in X_H$, and $\lambda_H^\pm \in Q_H$, such that

$$b(w, \lambda_H^\pm) = -\left\{ \kappa \left[ 2a^s(w, e_H^\pm) - \ell(w) \right] \pm \ell^O(w) + a(\psi_H^\pm, w) \right\}, \forall w \in \hat{X}_H. \quad (A.4)$$

$$a(w, \Theta_H + e_H^\pm) = \ell^N(w), \forall w \in X_H, \quad (A.5)$$

$$b(e_H^\pm, q) = 0, \forall q \in Q_H, \quad (A.6)$$

Note that, $e_H^\pm = 0$ and $b(w, \lambda_H^\pm) = 0$ when $w$ in continuous spaces ($X_H$).

With $\psi_H^\pm = \pm \psi_H$,

$$a(\psi_H, w) + \ell^O(w) = 0, \forall w \in X_H. \quad (A.7)$$

Note also that, the adjoint calculation is same as equation (3.54) and independent of the parameter $\kappa$.

H-mesh Hybrid Flux Calculation

Knowing the adjoint that $\psi_H^\pm \in X_H$, the hybrid flux, $\lambda_H^\pm \in Q_H$, can now be evoked from equation (A.4).

$$b(w, \lambda_H^\pm) = \kappa \ell^F(w) \mp \ell^O(w) - a(\psi_H^\pm, w), \forall w \in \hat{X}_H. \quad (A.8)$$

Alternatively, find $\lambda_0H \in Q_H$ and $\lambda_1H \in Q_H$ such that

$$b(w, \lambda_0H) = -\ell^O(w) - a(\psi_H, w), \forall w \in \hat{X}_H, \quad (A.9)$$

$$b(w, \lambda_1H) = \ell^F(w) = \ell^N(w) - a(w, \Theta_H), \forall w \in \hat{X}_H. \quad (A.10)$$

and then set $\lambda_H^\pm = \pm \lambda_0H + \kappa \lambda_1H$. Note that $\lambda_0H$ equilibrates the $H$-mesh adjoint (A.7), whereas $\lambda_1H$ equilibrates the original $H$-mesh solution.

Equations (A.9) and (A.10) represent a solvable but indeterminate system. If FETI method is applied, they become an interface problem. The computation of an acceptable compatible solution is known as the equilibration problem.
h-mesh Local Neumann Calculation

The minimizers of \( \mathcal{L}^\pm(v, \psi_h^\pm, \lambda_h^\pm) \), which is \( \hat{e}_h^\pm \in \hat{X}_h \), will satisfy the following stationarity condition.

\[
2a^s(w, \hat{e}_h^\pm) = \kappa \ell^E(w) + \ell^O(w) - a(\psi_h^\pm, w) - b(w, \lambda_h^\pm), \quad \forall w \in \hat{X}_h.
\]

(A.11)

where \( \psi_h^\pm \) is the interpolant of \( \psi_H^\pm \) and \( \lambda_h^\pm \) is the interpolant of \( \lambda_H^\pm \).

Equation (A.11) can be rearranged into equations for \( \hat{e}_{0h} \) and \( \hat{e}_{1h} \).

\[
2a^s(w, \hat{e}_{0h}) = -\ell^O(w) - a(\psi_h, w) - b(w, \lambda_{0h}), \quad \forall w \in \hat{X}_h.
\]

(A.12)

\[
2a^s(w, \hat{e}_{1h}) = \ell^O(w) - a(w, \Theta_H) - b(w, \lambda_{1h}), \quad \forall w \in \hat{X}_h.
\]

(A.13)

where \( \hat{e}_h^\pm = \pm \frac{1}{\kappa} \hat{e}_{0h} + \hat{e}_{1h} \). Note that \( \hat{e}_{0h} \) and \( \hat{e}_{1h} \) are related to the error in the adjoint \( \psi_H \) and the error in the solution \( \Theta_H \), respectively.

Upper and Lower Bounds Calculation

To evaluate the bounds, it remains only to insert the computed values for \( \hat{e}_{0h}^\pm, \psi_H^\pm \), and \( \lambda_H^\pm \) into the Lagrangian (A.2).

\[
(s_h)_{UB}(T_H, \kappa) = \ell^O(\Theta_H) + \kappa a^s(\hat{e}_{0h}^-, \hat{e}_{1h}^-)
\]

(A.14)

\[
(s_h)_{LB}(T_H, \kappa) = \ell^O(\Theta_H) - \kappa a^s(\hat{e}_{0h}^+, \hat{e}_{1h}^+)
\]

(A.15)

where the bounds depend on the “coarse” working mesh \( T_H \) and \( \kappa \).

In terms of \( \hat{e}_{0h} \) and \( \hat{e}_{1h} \)

\[
(s_h)_{UB}(T_H, \kappa) = \ell^O(\Theta_H) - 2a^s(\hat{e}_{0h}, \hat{e}_{1h}) + \frac{1}{\kappa} a^s(\hat{e}_{0h}, \hat{e}_{0h}) + \kappa a^s(\hat{e}_{1h}, \hat{e}_{1h}).
\]

(A.16)

\[
(s_h)_{LB}(T_H, \kappa) = \ell^O(\Theta_H) - 2a^s(\hat{e}_{0h}, \hat{e}_{1h}) - \frac{1}{\kappa} a^s(\hat{e}_{0h}, \hat{e}_{0h}) - \kappa a^s(\hat{e}_{1h}, \hat{e}_{1h}).
\]

(A.17)

Note that, equations (A.16) and (A.17) are valid for any positive value of the parameter \( \kappa \).

The half bound gap becomes,

\[
\Delta(T_H, \kappa) = \frac{1}{\kappa} a^s(\hat{e}_{0h}, \hat{e}_{0h}) + \kappa a^s(\hat{e}_{1h}, \hat{e}_{1h}).
\]

(A.18)

where the choice of \( \kappa \) will affect the (half) bound gap.

The output predictor is also reformulated as,

\[
s_{\text{pre}}(T_H) = \ell^O(\Theta_H) - 2a^s(\hat{e}_{0h}, \hat{e}_{1h}),
\]

(A.19)
where the selection of \( \kappa \) will not affect the output predictor.

**Optimized Bounds**

Since \( \hat{e}_{0h} \) and \( \hat{e}_{1h} \) do not depend on the choice of \( \kappa \), I can readily find that \( \kappa \) which minimizes the bound gap and hence renders the lower and upper bounds as sharp as possible. The optimal parameter \( \kappa^* \) is defined by

\[
\kappa^* = \sqrt{\frac{a^s(e_{0h}, \hat{e}_{0h})}{a^s(e_{1h}, \hat{e}_{1h})}}.
\] (A.20)

Hence the optimized lower and upper bounds are reformulated as

\[
(s_h)_{UB}(T_H, \kappa^*) = \epsilon^O(\Theta_H) - 2a^s(\hat{e}_{0h}, \hat{e}_{1h}) + \frac{1}{\kappa^*}a^s(\hat{e}_{0h}, \hat{e}_{0h}) + \kappa^*a^s(\hat{e}_{1h}, \hat{e}_{1h}). \tag{A.21}
\]

\[
(s_h)_{LB}(T_H, \kappa^*) = \epsilon^O(\Theta_H) - 2a^s(\hat{e}_{0h}, \hat{e}_{1h}) - \frac{1}{\kappa^*}a^s(\hat{e}_{0h}, \hat{e}_{0h}) - \kappa^*a^s(\hat{e}_{1h}, \hat{e}_{1h}). \tag{A.22}
\]

the associated bound gap is

\[
\Delta(T_H, \kappa^*) = \frac{1}{\kappa^*}a^s(\hat{e}_{0h}, \hat{e}_{0h}) + \kappa^*a^s(\hat{e}_{1h}, \hat{e}_{1h}). \tag{A.23}
\]

**Local Indicators**

In this section, the local indicators in terms of optimal parameter \( \kappa^* \) is presented again (It was already introduced in Chapter 4 of this thesis). Here, the sum of elemental contributions is reformulated as,

\[
\Delta(T_H) = \sum_{T_H \in \mathcal{T}_H} \Delta_{T_H}^T(T_H, \kappa^*), \tag{A.24}
\]

where

\[
\Delta_{T_H}(T_H, \kappa^*) = \frac{1}{\kappa^*}a_{T_H}^s(\hat{e}_{0h}, \hat{e}_{0h}) + \kappa^*a_{T_H}^s(\hat{e}_{1h}, \hat{e}_{1h}) \tag{A.25}
\]

\[
= \sqrt{\frac{a^s(\hat{e}_{1h}, \hat{e}_{1h})}{a^s(\hat{e}_{0h}, \hat{e}_{0h})}a_{T_H}^s(\hat{e}_{0h}, \hat{e}_{0h}) + \sqrt{a^s(\hat{e}_{0h}, \hat{e}_{0h})a_{T_H}^s(\hat{e}_{1h}, \hat{e}_{1h})}. \tag{A.26}
\]

Note that \( \Delta_{T_H}(T_H, \kappa^*) \) is non-negative and can thus be directly interpreted as the contribution to the bound gap from element \( T_H \).
Appendix B

Numerical Validation on fine mesh Hybrid Flux

The direct calculations of hybrid flux are introduced in Chapter 6. The detail results of bound method with the adaptive mesh refinement are presented herein. Two different initial coarse meshes, i.e. an adaptively refined structured mesh and an adaptively generated unstructured mesh, are considered. The average corner solution output $s^{(2)}$ is considered as the output of interest herein.

Adaptive Refinement of A Structured Initial Mesh

The adaptive mesh refinement technique with hybrid flux calculations on the coarse $H$-mesh reduces the bound gap and distributes the bounds error uniformly, but bounds still remains higher values. In order to reduce the bound gap further more, the adaptive mesh refinement technique with the direct calculations of hybrid flux on the fine $h$-mesh is essentially required. In this section, results of bound method with the fine $h$-mesh hybrid flux calculations on adaptively refined meshes of a structured initial mesh are presented and compared with bounds of the coarse $H$-mesh hybrid flux.

Table B.1 presents the non-optimal bounds with the $h$-mesh hybrid flux calculations for adaptive refinement. Similarly, Table B.2 reports for the optimal bounds. Table B.1 and Table B.2 show that a reduction of the optimal relative bound gap $\theta$ from 43.4% to 8.7% is achieved with the fine $h$-mesh hybrid flux calculations for adaptive refinement, while the non-optimal relative bound gap decreases from 70.4% to 12.6%. Recall that the reduction of optimal bound
APPENDIX B. NUMERICAL VALIDATION ON FINE MESH HYBRID FLUX

### Table B.1: Non-optimal bounds with $h$-mesh hybrid flux calculations for adaptive refinement of a structured initial mesh.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^0$</th>
<th>$T_H^1$</th>
<th>$T_H^2$</th>
<th>$T_H^3$</th>
<th>$T_H^4$</th>
<th>$T_H^{1/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>total d.o.f.</td>
<td>1,296</td>
<td>1,660</td>
<td>2,211</td>
<td>3,399</td>
<td>8,084</td>
<td>10,368</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
<td>0.3013</td>
<td>0.2774</td>
<td>0.2541</td>
<td>0.2442</td>
<td>0.2316</td>
<td>0.2401</td>
</tr>
<tr>
<td>$s_H$</td>
<td>0.1476</td>
<td>0.1820</td>
<td>0.1888</td>
<td>0.1964</td>
<td>0.2016</td>
<td>0.1920</td>
</tr>
<tr>
<td>$(s_h)_{LB}$</td>
<td>0.0522</td>
<td>0.1124</td>
<td>0.1433</td>
<td>0.1623</td>
<td>0.1796</td>
<td>0.1610</td>
</tr>
<tr>
<td>$(s_h)_{pre}$</td>
<td>0.1768</td>
<td>0.1949</td>
<td>0.1987</td>
<td>0.2028</td>
<td>0.2056</td>
<td>0.2006</td>
</tr>
<tr>
<td>$\pm \Delta (T_H)$</td>
<td>± 0.1245, ± 0.0825</td>
<td>± 0.0554, ± 0.0415</td>
<td>± 0.0260, ± 0.0395</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\theta$ (%)</td>
<td>70.4</td>
<td>42.3</td>
<td>27.9</td>
<td>20.5</td>
<td>12.6</td>
<td>19.7</td>
</tr>
</tbody>
</table>

### Table B.2: Optimal bounds with $h$-mesh hybrid flux calculations for adaptive refinement of a structured initial mesh.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^0$</th>
<th>$T_H^1$</th>
<th>$T_H^2$</th>
<th>$T_H^3$</th>
<th>$T_H^4$</th>
<th>$T_H^{1/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>total d.o.f.</td>
<td>1,296</td>
<td>1,660</td>
<td>2,211</td>
<td>3,399</td>
<td>8,084</td>
<td>10,368</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
<td>0.2537</td>
<td>0.2496</td>
<td>0.2360</td>
<td>0.2314</td>
<td>0.2226</td>
<td>0.2240</td>
</tr>
<tr>
<td>$s_H$</td>
<td>0.1746</td>
<td>0.1815</td>
<td>0.1887</td>
<td>0.1951</td>
<td>0.2004</td>
<td>0.1920</td>
</tr>
<tr>
<td>$(s_h)_{LB}$</td>
<td>0.1001</td>
<td>0.1394</td>
<td>0.1613</td>
<td>0.1725</td>
<td>0.1871</td>
<td>0.1771</td>
</tr>
<tr>
<td>$(s_h)_{pre}$</td>
<td>0.1769</td>
<td>0.1945</td>
<td>0.1987</td>
<td>0.2019</td>
<td>0.2048</td>
<td>0.2005</td>
</tr>
<tr>
<td>$\pm \Delta (T_H)$</td>
<td>± 0.0768, ± 0.0551</td>
<td>± 0.0374, ± 0.0294</td>
<td>± 0.0178, ± 0.0235</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\theta$ (%)</td>
<td>43.4</td>
<td>28.3</td>
<td>18.8</td>
<td>14.6</td>
<td>8.7</td>
<td>11.7</td>
</tr>
<tr>
<td>$\kappa^*$ (%)</td>
<td>2.89</td>
<td>2.70</td>
<td>2.53</td>
<td>2.62</td>
<td>2.75</td>
<td>3.04</td>
</tr>
</tbody>
</table>
gap \( \theta \) with the coarse \( H \)-mesh hybrid flux calculations provides only from 65.5% to 24.4%. Note that, the relative bound gap \( \theta \) for adaptive refinement always sharper than that of uniform refinement with or without the optimal stabilization parameter \( \kappa^* \).

Table B.3 presents the number of element (%) by half bound gap \( \Delta(T_H) \) intervals using the non-optimal bound method for adaptive refinement of a structured initial mesh. Similarly, Table B.4 presents for the optimal bound method. As the number of element is increased, the bound error (bound gap) is decreased correspondingly. The final adapted mesh \( T^1 \) (8,084 elements) using the non-optimal bound method has 495 elements (6.1%) which have value of local bound gap error greater equal than \( 10^{-5} \) while the uniformly refined mesh \( T_{H/2}^0 \) (10,368 elements) has 774 elements (7.4%). Similarly, the final optimal adapted mesh \( T_h^1 \) (7,832 elements) has 185 elements (2.4%) which have value of bound gap greater equal than \( 10^{-5} \) while the uniformly refined mesh \( T_{H/2}^0 \) (10,368 elements) has 578 elements (5.5%). Note that, the optimal bound method provides the sharper bound gap and distributes the local bound gap error more uniformly.

Figure B.1 illustrates the adaptively refined meshes using the non-optimal bound method. Similarly, Figure B.2 shows the adaptively refined meshes using the optimal bound method. The shape of the mesh in Figure B.1 and Figure B.2 is slightly different, but however the finer elements are concentrated on the region of output (i.e. \( [2/3,1[\times]2/3,1[\times]2/3,1[\times] \)) on both meshes.

Figure B.3a shows the relative bound gap \( \theta \) for the non-optimal bound method as a function of the number of elements. Similarly, Figure B.3b presents the same graphs for the optimal bound method. It is clearly observed from Figure B.3 that the relative (half) bound gap \( \theta \) with optimal stabilization parameter \( \kappa^* \) less than that without stabilization parameter: i.e the optimal bound method provides sharper bounds than the non-optimal bound method. Note that, the adaptive refinement provides always better performance than the uniform refinement.
Table B.3: Number of element (%) by half bound gap $\Delta(T_H)$ intervals for the non-optimal bound method.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^0$</th>
<th>$T_H^3$</th>
<th>$T_H^1$</th>
<th>$T_{H/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta(T_H) \geq 10^{-3}$</td>
<td>1,296</td>
<td>3,399</td>
<td>8,084</td>
<td>10.368</td>
</tr>
<tr>
<td>$5 \times 10^{-4} \leq \Delta(T_H) &lt; 10^{-3}$</td>
<td>36 (2.8)</td>
<td>3 (0.1)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$10^{-4} \leq \Delta(T_H) &lt; 5 \times 10^{-4}$</td>
<td>42 (3.2)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$5 \times 10^{-5} \leq \Delta(T_H) &lt; 10^{-4}$</td>
<td>68 (5.2)</td>
<td>4 (0.1)</td>
<td>0 (0)</td>
<td>57 (0.5)</td>
</tr>
<tr>
<td>$10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>65 (5.0)</td>
<td>91 (2.7)</td>
<td>0 (0)</td>
<td>114 (1.1)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>342 (26.4)</td>
<td>1,234 (36.3)</td>
<td>495 (16.1)</td>
<td>603 (5.8)</td>
</tr>
<tr>
<td>$10^{-6} \leq \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>249 (19.2)</td>
<td>962 (28.3)</td>
<td>893 (11.0)</td>
<td>318 (3.1)</td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 10^{-6}$</td>
<td>488 (37.6)</td>
<td>1,060 (31.2)</td>
<td>4,463 (55.2)</td>
<td>1,545 (14.9)</td>
</tr>
</tbody>
</table>

Table B.4: Number of element (%) by half bound gap $\Delta(T_H)$ intervals for the optimal bound method.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^0$</th>
<th>$T_H^3$</th>
<th>$T_H^1$</th>
<th>$T_{H/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta(T_H) \geq 10^{-3}$</td>
<td>1,296</td>
<td>3,399</td>
<td>7,832</td>
<td>10.368</td>
</tr>
<tr>
<td>$5 \times 10^{-4} \leq \Delta(T_H) &lt; 10^{-3}$</td>
<td>13 (1.0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$10^{-4} \leq \Delta(T_H) &lt; 5 \times 10^{-4}$</td>
<td>18 (1.4)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$5 \times 10^{-5} \leq \Delta(T_H) &lt; 10^{-4}$</td>
<td>124 (9.6)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>12 (0.1)</td>
</tr>
<tr>
<td>$10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>110 (8.5)</td>
<td>23 (0.7)</td>
<td>0 (0)</td>
<td>34 (0.3)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>296 (22.8)</td>
<td>960 (29.1)</td>
<td>185 (2.4)</td>
<td>532 (5.1)</td>
</tr>
<tr>
<td>$10^{-6} \leq \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>166 (12.8)</td>
<td>751 (22.8)</td>
<td>625 (8.0)</td>
<td>498 (4.8)</td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 10^{-6}$</td>
<td>522 (40.3)</td>
<td>1,414 (42.9)</td>
<td>3,969 (50.7)</td>
<td>1,717 (16.6)</td>
</tr>
</tbody>
</table>

| $\Delta(T_H) < 10^{-6}$ | 47 (3.6) | 146 (4.4) | 3,053 (39.0) | 7,575 (73.1) |
Figure B.1: Adaptively refined meshes of a structured initial mesh using the non-optimal bound method: (a) $\mathcal{T}_{H}^0$ has 1,296 elements, (b) $\mathcal{T}_{H}^1$ has 1,660 elements, (c) $\mathcal{T}_{H}^2$ has 2,211 elements, (d) $\mathcal{T}_{H}^3$ has 3,399 elements, (e) $\mathcal{T}_{H}^4$ has 8,084 elements, and (f) $\mathcal{T}_{H/2}^6$ has 10,368 elements.
Figure B.2: Adaptively refined meshes of a structured initial mesh using the optimal bound method: (a) $\mathcal{T}_H^0$ has 1,296 elements, (b) $\mathcal{T}_H^1$ has 1,662 elements, (c) $\mathcal{T}_H^2$ has 2,311 elements, (d) $\mathcal{T}_H^3$ has 3,294 elements, (e) $\mathcal{T}_H^4$ has 7,832 elements, and (f) $\mathcal{T}_{H/2}^0$ has 10,368 elements.
Figure B.3: Relative bound gap $\theta$ with $h$-mesh hybrid flux calculations as a function of the number of elements in log-log scale using adaptive and uniform refinement: (a) without optimal stabilization parameter $\kappa^*$ (top), (b) with optimal stabilization parameter $\kappa^*$ (bottom).
APPENDIX B. NUMERICAL VALIDATION ON FINE MESH HYBRID FLUX

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^0$</th>
<th>$T_H^1$</th>
<th>$T_H^2$</th>
<th>$T_H^3$</th>
<th>$T_H^4$</th>
<th>$T_H^{1/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>total d.o.f.</td>
<td>1,440</td>
<td>1,851</td>
<td>2,010</td>
<td>3,121</td>
<td>5,620</td>
<td>9,624</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
<td>0.2785</td>
<td>0.2534</td>
<td>0.2499</td>
<td>0.2361</td>
<td>0.2301</td>
<td>0.2241</td>
</tr>
<tr>
<td>$s_H$</td>
<td>0.1710</td>
<td>0.1930</td>
<td>0.1966</td>
<td>0.1996</td>
<td>0.2033</td>
<td>0.2048</td>
</tr>
<tr>
<td>$(s_h)_{LB}$</td>
<td>0.1008</td>
<td>0.1482</td>
<td>0.1557</td>
<td>0.1730</td>
<td>0.1827</td>
<td>0.1903</td>
</tr>
<tr>
<td>$(s_h)_{pre}$</td>
<td>0.1897</td>
<td>0.2008</td>
<td>0.2028</td>
<td>0.2045</td>
<td>0.2064</td>
<td>0.2072</td>
</tr>
<tr>
<td>$\pm \Delta(T_H)$</td>
<td>$\pm 0.0888$</td>
<td>$\pm 0.0546$</td>
<td>$\pm 0.0471$</td>
<td>$\pm 0.0316$</td>
<td>$\pm 0.0237$</td>
<td>$\pm 0.0169$</td>
</tr>
<tr>
<td>$\theta$ (%)</td>
<td>46.8</td>
<td>26.2</td>
<td>23.2</td>
<td>15.4</td>
<td>11.5</td>
<td>8.1</td>
</tr>
<tr>
<td>$\kappa^*$ (%)</td>
<td>2.74</td>
<td>2.29</td>
<td>2.19</td>
<td>2.21</td>
<td>2.38</td>
<td>2.81</td>
</tr>
</tbody>
</table>

Table B.5: Non-optimal bounds with h-mesh hybrid flux calculations for adaptive refinement of an unstructured initial mesh.

<table>
<thead>
<tr>
<th># of elements</th>
<th>$T_H^0$</th>
<th>$T_H^1$</th>
<th>$T_H^2$</th>
<th>$T_H^3$</th>
<th>$T_H^4$</th>
<th>$T_H^{1/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>total d.o.f.</td>
<td>393</td>
<td>486</td>
<td>518</td>
<td>747</td>
<td>1,242</td>
<td>1,996</td>
</tr>
<tr>
<td>$(s_h)_{UB}$</td>
<td>0.2469</td>
<td>0.2393</td>
<td>0.2340</td>
<td>0.2272</td>
<td>0.2198</td>
<td>0.2203</td>
</tr>
<tr>
<td>$s_H$</td>
<td>0.1710</td>
<td>0.1930</td>
<td>0.1976</td>
<td>0.2001</td>
<td>0.2035</td>
<td>0.1991</td>
</tr>
<tr>
<td>$(s_h)_{LB}$</td>
<td>0.1325</td>
<td>0.1622</td>
<td>0.1730</td>
<td>0.1823</td>
<td>0.1931</td>
<td>0.1881</td>
</tr>
<tr>
<td>$(s_h)_{pre}$</td>
<td>0.1897</td>
<td>0.2008</td>
<td>0.2035</td>
<td>0.2048</td>
<td>0.2065</td>
<td>0.2042</td>
</tr>
<tr>
<td>$\pm \Delta(T_H)$</td>
<td>$\pm 0.0572$</td>
<td>$\pm 0.0385$</td>
<td>$\pm 0.0305$</td>
<td>$\pm 0.0224$</td>
<td>$\pm 0.0133$</td>
<td>$\pm 0.0161$</td>
</tr>
<tr>
<td>$\theta$ (%)</td>
<td>30.1</td>
<td>19.2</td>
<td>15.0</td>
<td>10.9</td>
<td>6.4</td>
<td>7.8</td>
</tr>
<tr>
<td>$\kappa^*$ (%)</td>
<td>2.74</td>
<td>2.29</td>
<td>2.19</td>
<td>2.21</td>
<td>2.38</td>
<td>2.81</td>
</tr>
</tbody>
</table>

Table B.6: Optimal bounds with h-mesh hybrid flux calculations for adaptive refinement of an unstructured initial mesh.

Adaptive Refinement of An Unstructured Initial Mesh

The adaptive refinement of an unstructured initial mesh is investigated for the purpose of demonstrating the effect of h-mesh hybrid flux calculations. Results of bound method with the fine h-mesh hybrid flux calculations on adaptive refinement of an unstructured initial mesh are presented and compared with results on adaptive refinement of a structured initial mesh.

Table B.5 presents the non-optimal bounds with the fine h-mesh hybrid flux calculations for adaptive refinement of an unstructured initial mesh. Similarly, Table B.6 reports for the optimal bounds. Table B.5 and Table B.6 show a reduction of the relative bound gap $\theta$ using optimal bound method from 30.1% to 6.4% with h-mesh hybrid flux calculations, while relative
bound gap using the non-optimal bound method decreases from 46.8% to 8.1%. Recall that the reduction of bound gap $\theta$ using optimal bound method with the coarse $H$-mesh hybrid flux calculations provides only from 49.1% to 18.2%. Also the reduction of relative bound gap using optimal bound method for adaptive refinement of a structured initial mesh decreases only from 43.4% to 8.7%.

Table B.7 presents the number of element (%) by half bound gap $\Delta(T_H)$ intervals using the non-optimal bound method for adaptive refinement of an unstructured initial mesh. Similarly, Table B.8 presents for the optimal bound method. As the number of element is increased, the bound error (bound gap) is decreased correspondingly. The final adapted mesh $\mathcal{T}_5$ (9.624 elements) using the non-optimal bound method has 123 elements (1.4%) which have value of local bound gap error greater equal than $10^{-5}$ while the uniformly refined mesh $\mathcal{T}_{H/2}^0$ (11.520 elements) has 647 elements (5.6%). Similarly, the final optimal adapted mesh $\mathcal{T}_H^1$ (8.971 elements) using the optimal bound method has 53 elements (0.6%) which have value of bound gap greater equal than $10^{-5}$ while the uniformly refined mesh $\mathcal{T}_{H/2}^0$ (11,520 elements) has 304 elements (2.6%). Note that, adaptive refinement of an unstructured initial mesh with optimal bound method provides the sharper bound gap and distributes the local bound gap error more uniformly than any other method.

Figure B.3 shows the adaptively refined meshes of an unstructured initial mesh using the non-optimal bound method. Similarly, Figure B.2 presents meshes for the optimal bound method. The shape of meshes in Figure B.4 and Figure B.5 is slightly different. but however the finer elements are concentrated on the region of output (i.e. $\frac{2}{3} [x] \times \frac{2}{3} [x] \times \frac{2}{3} [x]$) on both meshes. Since the convection is stronger than diffusion in cube test case, the boundary layer is generated in the domain of output (i.e. inside of corner cube $\Omega^O$). To catch the right boundary layer in corner volume $\Omega^O$, the finer grids are required in this region.

Figure B.6a shows that the relative bound gap $\theta$ with the non-optimal bound method as a function of the number of elements using uniform and adaptive refinement of an unstructured initial mesh. Note that, the adaptive refinement always provides a sharper bound gap than the uniform refinement. Also the bound method with the optimal stabilization parameter presents better performance than that without the optimal stabilization parameter.

It is noticed that the optimal bound method with adaptive refinement of an unstructured
Table B.7: Number of element (%) by half bound gap $\Delta(T_H)$ intervals using the non-optimal bound method for adaptive refinement of an unstructured initial mesh.

<table>
<thead>
<tr>
<th>$\Delta(T_H)$</th>
<th>$T_H^1$</th>
<th>$T_H^2$</th>
<th>$T_H^3$</th>
<th>$T_H^4$</th>
<th>$T_H^{1/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\geq 10^{-3}$</td>
<td>1,440</td>
<td>2,010</td>
<td>5,620</td>
<td>9,624</td>
<td>11.520</td>
</tr>
<tr>
<td>$5 \times 10^{-4} \leq \Delta(T_H) &lt; 10^{-3}$</td>
<td>18 (1.2)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$5 \times 10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-4}$</td>
<td>37 (2.6)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>100 (6.9)</td>
<td>88 (4.4)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$10^{-5} &lt; \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>24 (1.7)</td>
<td>168 (8.3)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>390 (27.1)</td>
<td>816 (10.6)</td>
<td>574 (10.2)</td>
<td>123 (1.4)</td>
<td>572 (5.0)</td>
</tr>
<tr>
<td>$10^{-6} &lt; \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>316 (21.9)</td>
<td>397 (19.8)</td>
<td>961 (17.1)</td>
<td>612 (6.4)</td>
<td>364 (3.2)</td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>500 (34.7)</td>
<td>491 (24.4)</td>
<td>2,898 (51.6)</td>
<td>3,964 (41.2)</td>
<td>1,220 (10.6)</td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 10^{-6}$</td>
<td>55 (3.8)</td>
<td>50 (2.5)</td>
<td>1,187 (21.1)</td>
<td>4,916 (51.1)</td>
<td>9,289 (80.6)</td>
</tr>
</tbody>
</table>

Table B.8: Number of element (%) by half bound gap $\Delta(T_H)$ intervals using the optimal bound method for adaptive refinement of an unstructured initial mesh.

<table>
<thead>
<tr>
<th>$\Delta(T_H)$</th>
<th>$T_H^1$</th>
<th>$T_H^2$</th>
<th>$T_H^3$</th>
<th>$T_H^4$</th>
<th>$T_H^{1/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\geq 10^{-3}$</td>
<td>1,440</td>
<td>2,356</td>
<td>3,679</td>
<td>8,971</td>
<td>11.520</td>
</tr>
<tr>
<td>$5 \times 10^{-4} \leq \Delta(T_H) &lt; 10^{-3}$</td>
<td>2 (0.1)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$5 \times 10^{-5} \leq \Delta(T_H) &lt; 5 \times 10^{-4}$</td>
<td>14 (1.0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>115 (8.0)</td>
<td>10 (0.4)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$10^{-5} &lt; \Delta(T_H) &lt; 5 \times 10^{-5}$</td>
<td>98 (6.8)</td>
<td>68 (2.9)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>$5 \times 10^{-6} \leq \Delta(T_H) &lt; 10^{-5}$</td>
<td>441 (30.6)</td>
<td>897 (38.1)</td>
<td>689 (18.5)</td>
<td>53 (0.6)</td>
<td>290 (2.5)</td>
</tr>
<tr>
<td>$10^{-6} &lt; \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>151 (10.5)</td>
<td>494 (21.0)</td>
<td>771 (20.9)</td>
<td>364 (4.0)</td>
<td>450 (3.9)</td>
</tr>
<tr>
<td>$10^{-6} &lt; \Delta(T_H) &lt; 5 \times 10^{-6}$</td>
<td>499 (34.6)</td>
<td>773 (32.8)</td>
<td>1,920 (52.2)</td>
<td>3,868 (43.1)</td>
<td>2,253 (19.5)</td>
</tr>
<tr>
<td>$\Delta(T_H) &lt; 10^{-6}$</td>
<td>120 (8.3)</td>
<td>114 (4.8)</td>
<td>308 (8.4)</td>
<td>4,686 (52.2)</td>
<td>8,513 (73.9)</td>
</tr>
</tbody>
</table>

Initial mesh is the most desirable methodology to improve the sharpness of bounds and to distribute the local bound gap error more uniformly. The direct calculations of hybrid flux on the fine $h$-mesh always provide better numerical accuracy than the coarse $H$-mesh hybrid calculations but at a higher computational expense.
Figure B.4: Adaptively refined mesh of an unstructured initial mesh using the non-optimal bound method: (a) $\mathcal{T}_H^0$ has 1,440 elements, (b) $\mathcal{T}_H^1$ has 2,010 elements, (c) $\mathcal{T}_H^2$ has 3,121 elements, (d) $\mathcal{T}_H^3$ has 5,620 elements, (e) $\mathcal{T}_H^4$ has 9,624 elements, and (f) $\mathcal{T}_{H/2}^0$ has 11,520 elements.
Figure B.5: Adaptively refined meshes of an unstructured initial mesh using the optimal bound method: (a) $\mathcal{T}_H^0$ has 1,440 elements, (b) $\mathcal{T}_H^1$ has 1,851 elements, (c) $\mathcal{T}_H^2$ has 2,356 elements, (d) $\mathcal{T}_H^3$ has 3,679 elements, (e) $\mathcal{T}_H^4$ has 8,971 elements, and (f) $\mathcal{T}_{H/2}^0$ has 11,520 elements.
Figure B.6: Relative bound gap $\theta$ with $h$-mesh hybrid flux calculations as a function of the number of elements in log-log scale using adaptive refinement of an unstructured initial mesh: (a) without optimal stabilization parameter $\kappa^*$ (top), (b) with optimal stabilization parameter $\kappa^*$ (bottom).
Appendix C

Analytical Solution of Rectangular Duct Flow

The governing equation for a rectangular duct flow is the three-dimensional Navier Stokes equation where \( U_1 = U_2 = 0 \) and depends on the velocity in \( x_3 \)-direction, \( U_3 \) and the negative pressure gradient (i.e. \( \frac{dp}{dx_3} \)).

Starting from simplified governing equation: Find \( U_3 \) such that

\[
\frac{\partial^2 U_3}{\partial x_1^2} + \frac{\partial^2 U_3}{\partial x_2^2} = \frac{1}{\mu} \frac{dp}{dx_3} \tag{C.1}
\]

with boundary conditions

\[
\frac{\partial U_3}{\partial x_1}(0, x_2) = 0, \quad \tag{C.2}
\]

\[
\frac{\partial U_3}{\partial x_2}(x_1, 0) = 0, \quad \tag{C.3}
\]

\[
U_3(a, x_2) = 0, \quad \tag{C.4}
\]

\[
U_3(x_1, a) = 0, \quad \tag{C.5}
\]

where \( a \) is the distance from the origin \((0,0)\) to the wall (indicated in Figure C.1). \( a = 1/2 \) in this case.

Here setting the right hand side of equation (C.1) as \( \frac{dp}{dx_3} = -f \) than equation (C.1) becomes,

\[
\frac{\partial^2 U_3}{\partial x_1^2} + \frac{\partial^2 U_3}{\partial x_2^2} + f = 0. \quad \tag{C.6}
\]

By using the separation of variables such that

\[
U_3(x_1, x_2) = \psi(x_1, x_2) + \phi(x_1), \quad \tag{C.7}
\]
then
\[ \frac{\partial^2 \psi}{\partial x_1^2} + \frac{\partial^2 \psi}{\partial x_2^2} + \frac{\partial^2 \phi}{\partial x_1^2} + f = 0. \] (C.8)

Note that, equation (C.8) can be changed into two parts as follows:

1. \[ \frac{d^2 \phi}{dx_1^2} = -f. \] (C.9)

with boundary conditions
\[ \phi(a) = 0, \] (C.10)
\[ \frac{d\phi}{dx_1}(0) = 0. \] (C.11)

2. \[ \frac{\partial^2 \psi}{\partial x_1^2} + \frac{\partial^2 \psi}{\partial x_2^2} = 0, \] (C.12)

with boundary conditions
\[ \frac{\partial \psi}{\partial x_1}(0, x_2) = 0, \] (C.13)
\[ \frac{\partial \psi}{\partial x_2}(x_1, 0) = 0, \] (C.14)
\[ \psi(a, x_2) = 0, \] (C.15)
\[ \psi(x_1, a) = -\phi(x_1). \] (C.16)
Using equation (C.9) and applying boundary conditions which are equations (C.10) and (C.11) then
\[ \phi(x_1) = \frac{1}{2} f(a^2 - x_1^2). \tag{C.17} \]

The separation of variables, which is \( \phi(x_1, x_2) = X_1(x_1)X_2(x_2) \), is applied in equation (C.12) then
\[ \frac{1}{X_1} \frac{\partial^2 X_1}{\partial x_1^2} = -\frac{1}{X_2} \frac{\partial^2 X_2}{\partial x_2^2} = -k. \tag{C.18} \]

Here equation (C.18) is separated into two parts as follows:
\[ X_1'' + kX_1 = 0, \tag{C.19} \]

with boundary conditions
\[ \frac{dX_1}{dx_1}(0) = 0, \text{ and } X_1(a) = 0. \tag{C.20} \]

Similarly,
\[ X_2'' - kX_2 = 0, \tag{C.21} \]

with boundary conditions
\[ \frac{dX_2}{dx_2}(0) = 0. \tag{C.22} \]

Solving for equation (C.19) and applying the boundary conditions (C.20) then
\[ X_1(x_1) = B \cos \frac{1}{a} (n\pi + \frac{\pi}{2})x_1, \quad n = 0, 1, 2, \ldots \tag{C.23} \]

where \( B \) is a constant. Similarly, from equations (C.21) and (C.22)
\[ X_2(x_2) = 2C \cosh \left[ \frac{1}{a} (n\pi + \frac{\pi}{2})x_2 \right], \quad n = 0, 1, 2, \ldots \tag{C.24} \]

here \( C \) is a constant. Combining equations (C.23) and (C.24) then
\[ \psi(x_1, x_2) = \sum_{n=0}^{\infty} A_n \cos(\lambda_n x_1) \cosh(\lambda_n x_2). \tag{C.25} \]

where
\[ \lambda_n = \frac{1}{a} (n\pi + \frac{\pi}{2}). \tag{C.26} \]

and \( A_n \) is also a constant and can be determined by applying boundary condition (C.16). Hence,
\[ A_n \cosh(\lambda_n a) = \frac{f}{a} \int_0^a x_1^2 \cos(\lambda_n x_1) - a^2 \cos(\lambda_n x_1) \, dx_1 \]
\[ = \frac{2f}{a\lambda_n^3} (-1)^n. \tag{C.27} \]
APPENDIX C. ANALYTICAL SOLUTION OF RECTANGULAR DUCT FLOW

Then $A_n$ becomes

$$A_n = \frac{2f}{a\lambda_n^3} \frac{(-1)^n}{cosh(\lambda_n a)}.$$  \hspace{1cm} (C.28)

Therefore equation (C.25) can be reformulated as follow.

$$\psi(x_1, x_2) = \sum_{n=0}^{\infty} -\frac{2f}{a} \frac{(-1)^n}{\lambda_n^3} \frac{cosh(\lambda_n x_2)}{cosh(\lambda_n a)} \cos(\lambda_n x_1).$$ \hspace{1cm} (C.29)

Combining equations (C.17) and (C.29) then

$$U_3(x_1, x_2) = \phi(x_1) + \psi(x_1, x_2)$$

$$= \frac{1}{2} f(a^2 - x_1^2) - \frac{2f}{a} \sum_{n=0}^{\infty} \frac{(-1)^n}{\lambda_n^3} \frac{cosh(\lambda_n x_2)}{cosh(\lambda_n a)} \cos(\lambda_n x_1).$$ \hspace{1cm} (C.30)

where

$$\lambda_n = \frac{1}{a} (n\pi + \frac{\pi}{2}).$$ \hspace{1cm} (C.31)

Applying $a = 1/2$ and $f = -\frac{1}{\mu} \frac{dp}{dx_3}$ then the analytical solution of a rectangular duct flow is now formulated as follow.

$$U_1 = U_2 = 0,$$

$$U_3(x_1, x_2, x_3) = -\frac{2}{\mu} \frac{dp}{dx_3} \left[ \frac{1}{4} \left( \frac{1}{4} - x_1^2 \right) - 2 \sum_{n=0}^{\infty} \frac{(-1)^n}{\lambda_n^3} \frac{cosh(\lambda_n x_2)}{cosh(\lambda_n/2)} \cos(\lambda_n x_1) \right].$$ \hspace{1cm} (C.32)

where

$$\lambda_n = 2 \left( n\pi + \frac{\pi}{2} \right).$$ \hspace{1cm} (C.33)

Note that the above analytical solution has the domain of $[-\frac{1}{2}, \frac{1}{2}]$.