PARALLEL ADAPTIVE INFRASTRUCTURE FOR MAGNETICALLY CONFINED FUSION PLASMA SIMULATIONS

By

Fan Zhang

A Dissertation Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: MECHANICAL ENGINEERING

Approved by the
Examining Committee:

_____
Mark S. Shephard, Dissertation Adviser

_____
Assad Oberai, Member

_____
Onkar Sahni , Member

_____
Fengyan Li, Member

_____
Stephen C. Jardin, Member

Rensselaer Polytechnic Institute
Troy, New York

August 2015
(For Graduation December 2015)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

My first gratitude goes to my advisor, Dr. Mark S. Shephard. I greatly value the opportunity and experience to work as his Ph.D. student. Without his continuous support and patience, it would be impossible to finish this thesis. His rigorous attitude on the technology and his persistence to useful innovations will continue to guide me in the future.

I would like to thank the rest of the committee members: Dr. Assad Oberai, Dr. Onkar Sahni, Dr. Fengyan Li and Dr. Stephen C. Jardin for their invaluable advice on the thesis.

I am particularly grateful to Dr. Stephen C. Jardin for giving me the access to the M3D-C^1 code and the computing resources, for the responsive replies to all my questions, and for the efforts to set up the tests used in the thesis.

I thank Dr. E. Seegyoung Seol for the help on the project and the encouragement to finish the thesis. I am also grateful to the other people who helped on the project: Dr. Nathaniel M. Ferraro, Dr. Jin Chen, Dr. Danail Vassilev, and Dr. Fabien Delalondre.

I would like to thank Dr. Weiying Zheng and Dr. Shi-peng Mao for the informative discussions on error estimation during my visit to the Institute of Computational Mathematics and Scientific/Engineering Computing of the Chinese Academy of Sciences.

I am deeply grateful to the friends who have shared both my joyful and upset moments at RPI: Xiangyu Wang, Bin Wu, Jia Zhang, Lijuan Zhang, Yanheng Li, Jianfeng Liu, Yi Chen, Chu Wang, Jianguo Zhong, Yingying Wang, Jicong Cao and Qiukai Lu. Special thanks to Dr. Lingyun Li and Dr. Emily Liu for offering the precious advise on my life at RPI.

I would like to express my gratitude to my parents, Zengguang Zhang and Meixiang Wei, for their love and support throughout my life.

Lastly, I would like to thank my dear wife, Xuejiao Cao, for being on my side all theses years.

ABSTRACT

Numerical simulations of the magnetically confined plasmas play an important rule in understanding and predicting the physics in tokamak devices. In the present study, a parallel adaptive infrastructure for the magnetically confined fusion plasma simulations is developed.

An automatic meshing procedure is needed by two simulation codes under development at Princeton Plasma Physics Lab, M3D-C^1 and XGC1. M3D-C^1 requires the mesh generated and adapted on the tokamak cross-section with the option to contain a finite-thickness wall in the domain, and the 3D mesh constructed out of 2D meshes. XGC1 requires the mesh edges aligned with the magnetic flux surfaces. The mesh is one-element deep between adjacent flux surfaces and mesh improvement is applied at the X-point(s). A geometric model including both the tokamak wall structure and the magnetic flux surfaces is introduced to reflect the meshing needs. A component-based mesh generation procedure with control parameters specified on the geometric model is developed by combining unstructured triangulation, layered mesh generation by a one-element-deep marching procedure, local mesh modification, and toroidal 2D mesh extrusion to create a 3D mesh of the full reactor.

Mesh-based needs in M3D-C^1 include a procedure to form the global discrete equation, methods to estimate the simulation error, and a loop of adaptive mesh control. A mesh-adjacency based matrix assembly procedure with more efficient memory usage than the procedure using the PETSc matrix library directly is developed. The numerical conditioning of the global discrete system is improved through element-level operations. An explicit *a posterior* error estimator that calculates the mesh-dependent norm of the residual in the strong form is derived for the reduced MHD model under the large-aspect-ratio approximation. Software tools such as PUMI for mesh management and geometric model interfacing, APF for field management, MeshAdapt for mesh modification, PETSc for global equation solving, and Simmetrix for initial mesh generation are used to form the parallel adaptive loop in M3D-C^1.

CHAPTER 1
Background and Thesis Organization

1.1 Background and Motivation

Controlled fusion reactions hold the promise of providing a clean and sustainable energy source for future generations. The key issue in harnessing fusion energy is to confine the fuel such that the fusion reaction can happen in a sustainable and controlled way. Tokamak devices use magnetic confinement to achieve the sustainable fusion reaction. It is well known that the ionized fusion fuel inside the tokamak exists in a plasma state. Theoretical studies of the magnetically confined plasmas, especially through numerical simulations, play an important rule in analyzing and predicting the physics in the tokamak.

Numerical studies of plasmas in the tokamak face challenges from both the underlying complex physics and the software tools. The characteristic physics occur at multiple scales in both the spatial and temporal dimensions [1, 2] and require taking advantages of multi-level computational models. As the size of problems simulated increases, especially to the extreme scale, high-performance parallel computers must be used to properly solve the problem in a realistic time.

Mesh related needs of two codes targeted for parallel simulation of plasma physics are considered in this thesis. M3D-C^1 [3, 7, 8, 9, 6, 10] is a fusion plasma code under development at Princeton Plasma Physics Lab (PPPL). Finite elements with C^1 inter-element continuity are used to solve the fourth-order partial differential equations (PDEs) that are derived from the extended magnetohydrodynamic (MHD) equations combined with a stream function and/or potential representation for the velocity and magnetic potential vector fields [6]. The second fusion code also under development at PPPL is XGC1 which uses the particle-in-cell (PIC) method to simulate gyrokinetic particle turbulence [11, 12, 13].

Both fusion codes apply the mesh-based methods as the discretized representation of the simulation domain. In the mesh-based simulation, the reliability of results is efficiently improved through adaptive mesh control [14]. A mesh-based

adaptive analysis program includes the following components (Figure 1.1):

(A) A problem definition in terms of the mathematical model to represent the governing physics and a domain over which that physics is to be modeled.

(B) A mesh infrastructure that provides the interface from the problem definition to a discretized representation of the problem.

(C) A component to discretize the mathematical model over mesh entities. Given the inputs of the geometric domain, the mesh, and the field information describing the distribution of the material properties, loading and boundary conditions, this component produces the mesh entity discrete equations.

(D) A component to assemble the discrete equations. Given the mesh entity contributions and the ordering of the degrees of the freedom (DOFs), this component assembles the global discrete equation.

(E) A component to evaluate the current solution fields by solving the global discrete equation.

(F) A component to determine the adequacy of the current mesh discretization. Given the solution field, this component assesses the solution quality and generates the correction indication field.

(G) A component to control the quality of the mesh-based solution by improving the mesh. Given the correction indication field, the mesh is adapted and the solution fields are mapped from the original mesh to the improved mesh.

This thesis presents a parallel adaptive infrastructure for the magnetically confined fusion plasma simulations. It aims to address the following issues that arise from the simulation codes:

1. The geometry of fusion reactors needs to be defined to comply with the computational models. To meet the needs of the specific models used to discretize the mathematical model used by M3D-C^1 and XGC1, the geometry will include both the physical components from the tokamak design, and the desired

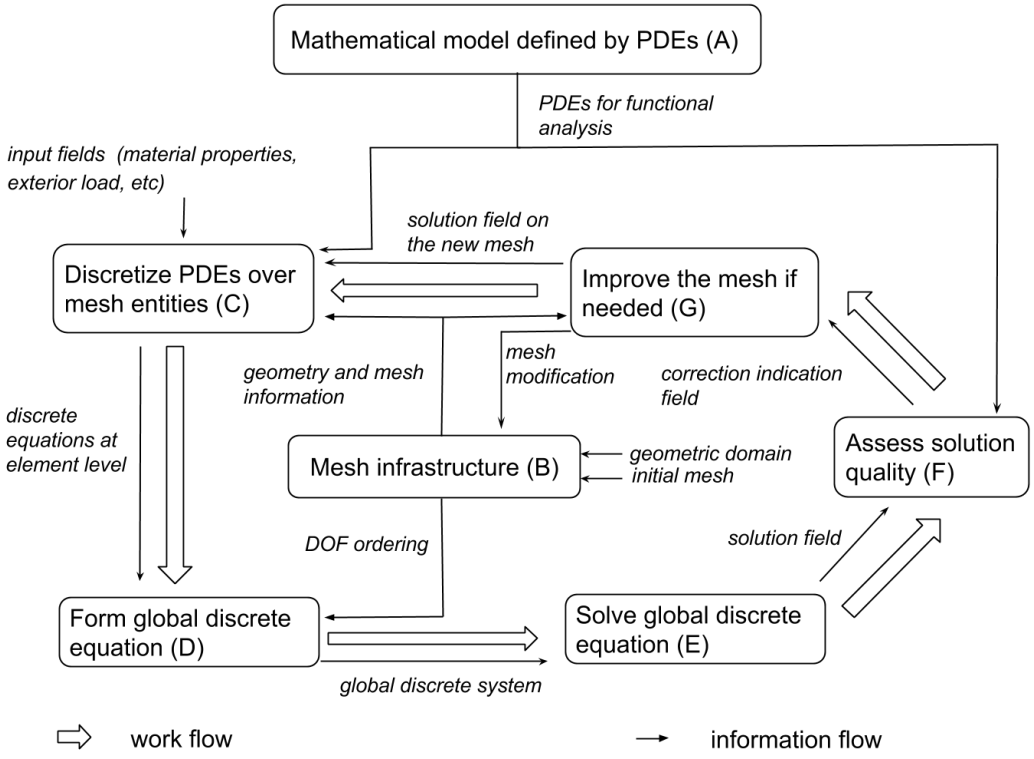


Figure 1.1: Components of a mesh-based adaptive loop.

physics components from features interior to the reactor. The geometric model with the topological representation [15] and the shape information needs to be defined as the necessary input of the mesh infrastructure (B in Figure 1.1) [16].

2. Controlled meshes must be generated and adapted (B and G in Figure 1.1). The challenges of meshing for fusion simulations lie in satisfying the specific requirements from different computational methods. Fully automatic meshing procedures that meet the specific constraints of the fusion simulation codes are needed.

3. An infrastructure that includes interacting software tools for the functional components in Figure 1.1 and methods for error estimation is needed to improve the accuracy and efficiency of the simulation by adaptive mesh control.

The entire simulation workflow needs to be a fully automatic procedure that executes the adaptive control loop illustrated in Figure 1.1. Information flow be-

tween the components of the analysis program needs to be tracked and mapped simultaneously when the mesh is adapted or the level of discretization is changed. This thesis provides methods and associated software to address each of these areas. In addition, a method to improve the numerical conditioning through element-level operations was developed to provide a better starting point to the equation solvers.

1.2 Organization of Dissertation

The following paragraphs outline the organization of the dissertation.

Chapter 2 reviews the extended MHD equations and the finite element formulation used by M3D-C^1.

Chapter 3 discusses an explicit *a posteriori* error estimator developed for M3D-C^1.

Chapter 4 gives an overview of the M3D-C^1 code and the extensions towards an automated version.

Chapter 5 presents the tokamak mesh definition procedure for the fusion plasma codes, M3D-C^1 and XGC1 [17]. A geometric model that includes both the reactor wall structure and the magnetic flux surfaces is introduced to support the meshing procedure. Mesh control parameters to meet the needs of the simulation procedure are identified and can be specified on that geometric model. The mesh generation procedure is constructed by combining the meshing operations that satisfies the constraints from the simulation procedures while creating well controlled graded meshes.

Chapter 6 presents a procedure to improve the numerical conditioning of the resulting matrix systems for M3D-C^1. The preconditioning method is based on the knowledge of the scales associated with the element-level physics.

Chapter 7 demonstrates the results of the parallel adaptive simulations with M3D-C^1.

Chapter 8 concludes the work carried out and discusses possible future works.

CHAPTER 2
Finite Element Formulation of Extended MHD Equation

This chapter presents an overview of the extended MHD equation and its finite element discretization in M3D-C^1. The overview is extracted from [3, 7, 8, 9, 6, 10].

2.1 Governing Equations

The governing equations solved by M3D-C^1 combine the Navier-Stokes equations of fluid dynamics and Maxwell's equations of electromagnetism. For simplicity of the presentation, the form considered does not include the particle source, the gyroviscosity tensor [8] or the two-fluid terms. In addition, the fluid viscosity and electrical resistivity are assumed constant.

The symbols in the equations are defined as:

ρ: ion density,

$\mathbf{B}$: magnetic field,

$\mathbf{E}$: electric field,

$\mathbf{J}$: electric current density,

$\mathbf{V}$: ion fluid velocity,

P: ion fluid pressure,

μ: dynamic viscosity,

$\mathbf{\Pi}_\mu$: deviatoric stress tensor due to fluid viscosity,

η: electrical resistivity.

The differential operators are defined and expanded in the Cartesian coordinate by the index notation that follows [18] as

∇: gradient operator, $\nabla F \equiv F_{,i}$

$\nabla\cdot$: divergence operator, $\nabla \cdot \mathbf{F} \equiv F_{i,i}$

$\nabla\times$: curl operator, $\nabla \times \mathbf{F} \equiv \epsilon_{ijk} F_{k,j}$

∇^2: Laplace operator, $\nabla^2 F \equiv F_{,ii}$

$\mathbf{I}$: identity tensor, $\mathbf{I} = \delta_{ij}$

Also see Section 7.3 for the differential operators defined on the RZ plane in the cylindrical coordinate system.

2.1.1 Continuity Equation

The continuity equation describes the conservation law of mass in the ion fluid. The equation is

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{V}) = 0. \tag{2.1}$$

The fluid is considered as "incompressible" if the change of particle density, ρ, is negligible [18]. The continuity equation for the incompressible fluid reduces to

$$\nabla \cdot \mathbf{V} = 0. \tag{2.2}$$

2.1.2 Momentum Equation

The momentum equation describes the conservation law of momentum in the ion fluid. The equation takes the form as

$$\rho \left(\frac{\partial \mathbf{V}}{\partial t} + \mathbf{V} \cdot \nabla \mathbf{V} \right) = \nabla \cdot \mathbf{\Pi}_\mu + \mathbf{J} \times \mathbf{B} - \nabla P. \tag{2.3}$$

For Newtonian fluid, the deviatoric stress tensor, $\mathbf{\Pi}_\mu$, takes the form as [18]

$$\mathbf{\Pi}_\mu = \mu(\nabla \mathbf{V} + (\nabla \mathbf{V})^T) - \frac{2}{3}\mu(\nabla \cdot \mathbf{V})\mathbf{I}. \tag{2.4}$$

For an incompressible fluid that satisfies $\nabla \cdot \mathbf{V} = 0$, the deviatoric stress tensor

and its contribution to the momentum equation is simplified as

$$\begin{aligned} \mathbf{\Pi}_\mu &= \mu(\nabla\mathbf{V} + (\nabla\mathbf{V})^T), && \text{(2.5a)} \\ \nabla \cdot \mathbf{\Pi}_\mu &= \mu\nabla^2\mathbf{V}. && \text{(2.5b)} \end{aligned}$$

$\mathbf{J} \times \mathbf{B}$ is the Lorentz force applied to the electric current by the magnetic field [19]. This force couples the fluid motion with the electromagnetic field in the momentum equation.

2.1.3 Maxwell-Faraday Equation

The Maxwell-Faraday equation relates the magnetic field varying in time with the electric field varying in space by [19]

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E}. \tag{2.6}$$

Ohm's law in the conductive fluid is [20]

$$\mathbf{E} + \mathbf{V} \times \mathbf{B} = \eta\mathbf{J}, \tag{2.7}$$

and Ampere's circuital law (without Maxwell's addition) is [19]

$$\mathbf{J} = \nabla \times \mathbf{B}. \tag{2.8}$$

Note that the physics quantities are non-dimensionalized and the magnetic constant does not appear in Ampere's circuital law. Equation 2.7 and Equation 2.8 are used to eliminate the electric field from the Maxwell-Faraday equation. The equation describing the dynamics of the magnetic field is rewritten in terms of the velocity as

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{V} \times \mathbf{B}) - \eta\nabla \times (\nabla \times \mathbf{B}). \tag{2.9}$$

Note that the term of $\mathbf{V} \times \mathbf{B}$ in Equation 2.9 couples the velocity field and the magnetic field.

2.1.4 Equation Set Considered

Combining the equations discussed in Section 2.1.1, 2.1.2 and 2.1.3, and substituting Ampere's circuital law (Equation 2.8) into the momentum equation, the equation set for the incompressible MHD is

$$\rho\left(\frac{\partial \mathbf{V}}{\partial t} + \mathbf{V}\cdot\nabla\mathbf{V}\right) = \mu\nabla^2\mathbf{V} + (\nabla\times\mathbf{B})\times\mathbf{B} - \nabla P, \tag{2.10a}$$

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla\times(\mathbf{V}\times\mathbf{B}) - \eta\nabla\times(\nabla\times\mathbf{B}). \tag{2.10b}$$

where $\mathbf{V}$ and $\mathbf{B}$ satisfy

$$\nabla\cdot\mathbf{V} = 0, \tag{2.11a}$$

$$\nabla\cdot\mathbf{B} = 0. \tag{2.11b}$$

Note that Equation 2.11b is Gauss's law for magnetism.

2.2 Fourth-order PDE and its Weak Formulation

Equation 2.10 and its extended form that includes the compressible part of the fluid [6] only contain differential operators up to second order. Two additional steps are taken to derive the fourth-order PDE that will be written in a cylindrical reference frame.

Define two sets of the scalar fields as $[U,\omega,\chi]$ and $[\psi,f]$, and the poloidal gradient operator as $\nabla_\perp \equiv \hat{R}\frac{\partial}{\partial R} + \hat{Z}\frac{\partial}{\partial Z}$. The velocity vector, $\mathbf{V}$, and the magnetic field, $\mathbf{B}$, are represented by the scalar fields in the (R,Z,φ) coordinate system (Figure 2.1) as [6]

$$\mathbf{V} = R^2\nabla U\times\nabla\varphi + R^2\omega\nabla\varphi + \frac{1}{R^2}\nabla_\perp\chi, \tag{2.12a}$$

$$\mathbf{B} = \nabla\psi\times\nabla\varphi - \nabla_\perp\frac{\partial f}{\partial\varphi} + B_\varphi\nabla\varphi, \tag{2.12b}$$

where $B_\varphi = F_0 + R^2\nabla\cdot\nabla_\perp f$ such that $\nabla\cdot B = 0$. F_0 is the constant defined from

the total toroidal current, I_0, as [6]

$$F_0 = \frac{I_0}{2\pi}. \tag{2.13}$$

$\mathbf{V}$ written in the general form of Equation 2.12a contains the compressible part. We leave the scope of the incompressible MHD defined by Equation 2.10 for a moment. The following three operators are applied to the momentum equation (Equation 2.3) [6]:

$$\nabla\varphi \cdot \nabla_\perp \times R^2, \tag{2.14a}$$

$$R^2 \nabla\varphi \cdot, \tag{2.14b}$$

$$-\nabla_\perp \cdot \frac{1}{R^2}, \tag{2.14c}$$

such that three scalar equations from the momentum equation are obtained as

$$\nabla\varphi \cdot \nabla_\perp \times R^2 \left\{ \rho \left(\frac{\partial \mathbf{V}}{\partial t} + \mathbf{V} \cdot \nabla \mathbf{V} \right) \right\} = \nabla\varphi \cdot \nabla_\perp \times R^2 \left\{ \nabla \cdot \mathbf{\Pi}_\mu + \mathbf{J} \times \mathbf{B} - \nabla P \right\}, \tag{2.15a}$$

$$R^2 \nabla\varphi \cdot \left\{ \rho \left(\frac{\partial \mathbf{V}}{\partial t} + \mathbf{V} \cdot \nabla \mathbf{V} \right) \right\} = R^2 \nabla\varphi \cdot \left\{ \nabla \cdot \mathbf{\Pi}_\mu + \mathbf{J} \times \mathbf{B} - \nabla P \right\}, \tag{2.15b}$$

$$-\nabla_\perp \cdot \frac{1}{R^2} \left\{ \rho \left(\frac{\partial \mathbf{V}}{\partial t} + \mathbf{V} \cdot \nabla \mathbf{V} \right) \right\} = -\nabla_\perp \cdot \frac{1}{R^2} \left\{ \nabla \cdot \mathbf{\Pi}_\mu + \mathbf{J} \times \mathbf{B} - \nabla P \right\}. \tag{2.15c}$$

Note that deviatoric stress tensor, $\mathbf{\Pi}_\mu$ is defined by Equation 2.4.

The two operators, $\nabla\varphi \cdot \nabla_\perp \times$ and $\nabla\varphi \cdot$, are applied to Equation 2.6 to obtain the two scalar equations from the Maxwell-Faraday Equation as

$$\nabla\varphi \cdot \nabla_\perp \times \left\{ \frac{\partial \mathbf{B}}{\partial t} \right\} = \nabla\varphi \cdot \nabla_\perp \times \left\{ -\nabla \times \mathbf{E} \right\}, \tag{2.16a}$$

$$\nabla\varphi \cdot \left\{ \frac{\partial \mathbf{B}}{\partial t} \right\} = \nabla\varphi \cdot \left\{ -\nabla \times \mathbf{E} \right\}. \tag{2.16b}$$

Note that $\mathbf{E}$ is substituted by Ohm's law (Equation 2.7).

The rest of the section illustrates the procedure that derives the fourth-order

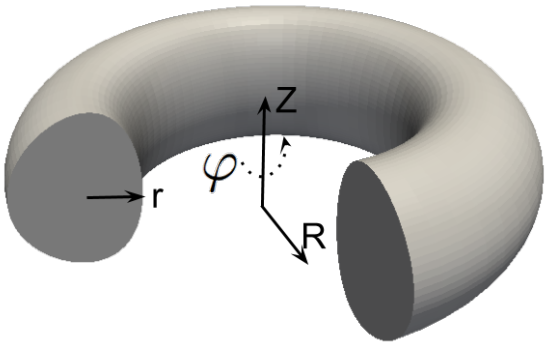


Figure 2.1: Cylindrical coordinate system used in M3D-C^1.

PDE and its weak formulation using the reduced MHD model under the large-aspect-ratio approximation as the example.

2.2.1 2D Incompressible MHD under Large-aspect-ratio Approximation

We consider the case when the ratio between the major axis and minor axis of the torus is large ($\frac{R_0}{r_0} >> 1$, see Figure 2.1). Under the large-aspect-ratio approximation, $\nabla\varphi = \frac{1}{R}\hat{\varphi} \sim \frac{1}{R_0}\hat{\varphi}$, and the set of the unit vectors $[\hat{Z}, \hat{R}, \hat{\varphi}]$ on a constant φ plane is assumed to be a set of Cartesian coordinates.

Assume $R_0 = 1$ and we have $\hat{\varphi} = \nabla\varphi$. Consider the reduced MHD model that only contains the 2D stream function, $U = U(R, Z)$, and the poloidal magnetic flux, $\psi = \psi(R, Z)$. $\mathbf{V}$ and $\mathbf{B}$ are represented by U and ψ in the form as (see Equation 2.12 by replacing $R = 1$)

$$\mathbf{V} = \nabla U \times \hat{\varphi} = \nabla \times (U\hat{\varphi}) = -\frac{\partial U}{\partial Z}\hat{R} + \frac{\partial U}{\partial R}\hat{Z}, \tag{2.17a}$$

$$\mathbf{B} = \nabla \psi \times \hat{\varphi} = \nabla \times (\psi\hat{\varphi}) = -\frac{\partial \psi}{\partial Z}\hat{R} + \frac{\partial \psi}{\partial R}\hat{Z}. \tag{2.17b}$$

The following useful identities are listed to derive the desired fourth-order

PDE:

$$\begin{aligned}
\nabla \times (\mathbf{f} \times \mathbf{g}) &\equiv (\mathbf{g} \cdot \nabla)\mathbf{f} + (\nabla \cdot \mathbf{g})\mathbf{f} - (\mathbf{f} \cdot \nabla)\mathbf{g} - (\nabla \cdot \mathbf{f})\mathbf{g}, && \text{(2.18a)}\\
\nabla \times \nabla \times \mathbf{f} &\equiv \nabla(\nabla \cdot \mathbf{f}) - \nabla^2\mathbf{f}, && \text{(2.18b)}\\
\nabla \cdot (\mathbf{f} \times \mathbf{g}) &\equiv (\nabla \times \mathbf{f}) \cdot \mathbf{g} - \mathbf{f} \cdot (\nabla \times \mathbf{g}), && \text{(2.18c)}\\
\nabla \times (\nabla f) &\equiv 0, && \text{(2.18d)}\\
\mathbf{f} \times (\mathbf{g} \times \mathbf{h}) &\equiv (\mathbf{f} \cdot \mathbf{h})\mathbf{g} - (\mathbf{f} \cdot \mathbf{g})\mathbf{h}. && \text{(2.18e)}
\end{aligned}$$

Gauss's law for magnetism ($\nabla \cdot \mathbf{B} = 0$) and the fluid incompressibility ($\nabla \cdot \mathbf{V} = 0$) are satisfied by the scalar representation defined by Equation 2.17. In addition, Equation 2.17 defines $\mathbf{V}$ and $\mathbf{B}$ on the 2D constant φ plane in the sense that $\hat{\varphi} \cdot \mathbf{B} = 0$ and $\hat{\varphi} \cdot \mathbf{V} = 0$.

For simplicity of notation, define Poisson bracket as

$$< F, G > \equiv -(\nabla F \times \nabla G) \cdot \hat{\varphi} \equiv \frac{\partial F}{\partial R}\frac{\partial G}{\partial Z} - \frac{\partial F}{\partial Z}\frac{\partial G}{\partial R}. \qquad (2.19)$$

From the definition of $\mathbf{V}$ and $\mathbf{B}$ by Equation 2.17,

$$\mathbf{V} \times \mathbf{B} = (\nabla U \times \hat{\varphi}) \times (\nabla \psi \times \hat{\varphi}) = -\hat{\varphi} \cdot (\nabla \psi \times \nabla U)\hat{\varphi} =< \psi, U > \hat{\varphi}, \qquad (2.20)$$

and

$$\nabla^2(\mathbf{V} \times \mathbf{B}) = \nabla^2 < \psi, U > \hat{\varphi}. \qquad (2.21)$$

Applying Identity 2.18c and noticing that $\hat{\varphi} \cdot \mathbf{B} = 0$ and $\hat{\varphi} \cdot \mathbf{V} = 0$, we obtain

$$\nabla \cdot (\mathbf{V} \times \mathbf{B}) = 0. \qquad (2.22)$$

2.2.2 Fourth-order PDE

To obtain the fourth-order PDE with regard to U and ψ, we substitute Equation 2.17 into Equation 2.10 and apply the operator $\hat{\varphi} \cdot \nabla_\perp \times$ to the both sides (see Equation 2.15a and 2.16a by replacing $R = 1$). Since we only consider the 2D case,

we have $\hat{\varphi} \cdot \nabla \times \equiv \hat{\varphi} \cdot \nabla_{\perp} \times$ and we use $\hat{\varphi} \cdot \nabla \times$ for the simple notation.

First consider the $\rho \mathbf{V}$ term in Equation 2.10. Noticing ρ is constant, $\frac{\partial U}{\partial \varphi} = 0$, and Identity 2.18b, it can be seen that

$$\hat{\varphi} \cdot (\nabla \times \rho \mathbf{V}) = \rho \hat{\varphi} \cdot \nabla \times (\nabla \times U \hat{\varphi}) = -\rho \hat{\varphi} \cdot (\nabla^2 U \hat{\varphi}) = -\rho \nabla^2 U. \tag{2.23}$$

Similarly,

$$\hat{\varphi} \cdot (\nabla \times \mathbf{B}) = -\nabla^2 \psi. \tag{2.24}$$

Now consider the viscosity force in Equation 2.10. The viscosity force is reduced to

$$\nabla \cdot \Pi_{\mu} = \mu \nabla^2 (\nabla U \times \hat{\varphi}). \tag{2.25}$$

Considering that $\hat{\varphi} \cdot (\nabla U \times \hat{\varphi}) = \nabla U \cdot (\hat{\varphi} \times \hat{\varphi}) = 0$, $\hat{\varphi} \cdot \nabla \times (\nabla^2 \mathbf{f}) = \nabla^2 (\hat{\varphi} \cdot \nabla \times \mathbf{f})$ and Identity 2.18a, we have

$$\begin{aligned} \hat{\varphi} \cdot \nabla \times (\nabla \cdot \Pi_{\mu}) &= \hat{\varphi} \cdot \nabla \times [\mu \nabla^2 (\nabla U \times \hat{\varphi})] \\ &= \mu \nabla^2 [\hat{\varphi} \cdot \nabla \times (\nabla U \times \hat{\varphi})] \\ &= -\mu \nabla^2 [\hat{\varphi} \cdot (\nabla \cdot \nabla U \hat{\varphi})] \\ &= -\mu \nabla^4 U. \end{aligned} \tag{2.26}$$

From Identity 2.18b and $\nabla \cdot \mathbf{B} = 0$, it can be seen that $\nabla \times \nabla \times \mathbf{B} = -\nabla^2 \mathbf{B}$. The electrical resistivity term in Equation 2.10 after applying $\hat{\varphi} \cdot \nabla \times$ is obtained similarly as the viscosity force:

$$\hat{\varphi} \cdot \nabla \times (\eta \nabla \times \nabla \times \mathbf{B}) = \eta \nabla^4 \psi. \tag{2.27}$$

Consider the convection term $\mathbf{V} \cdot \nabla \mathbf{V}$ after applying $\hat{\varphi} \cdot \nabla \times$ in Equation 2.10:

$$\hat{\varphi} \cdot \nabla \times (\mathbf{V} \cdot \nabla \mathbf{V}) = \hat{R} \cdot \frac{\partial (\mathbf{V} \cdot \nabla \mathbf{V})}{\partial Z} - \hat{Z} \cdot \frac{\partial (\mathbf{V} \cdot \nabla \mathbf{V})}{\partial R} = < \nabla^2 U, U > . \tag{2.28}$$

Similarly, from the properties defined by Equation 2.20 and 2.22, and Identity 2.18a and 2.18b, and the property defined by Equation 2.22, the terms of $\nabla \times \mathbf{B} \times \mathbf{B}$ and

$\mathbf{V} \times \mathbf{B}$ after applying $\hat{\varphi} \cdot \nabla \times$ are

$$\begin{aligned} \hat{\varphi} \cdot \nabla \times (\nabla \times \mathbf{B} \times \mathbf{B}) &= \hat{\varphi} \cdot [(\mathbf{B} \cdot \nabla) \nabla \times \mathbf{B}] \\ &= < \nabla^2 \psi, \psi >, \qquad (2.29) \\ \hat{\varphi} \cdot \nabla \times [\nabla \times (\mathbf{V} \times \mathbf{B})] &= -\hat{\varphi} \cdot \nabla^2 (\mathbf{V} \times \mathbf{B}) \\ &= -\nabla^2 < \psi, U > . \qquad (2.30) \end{aligned}$$

From Identity 2.18d, the pressure term, ∇P, after applying $\hat{\varphi} \cdot \nabla \times$ does not appear in the equation.

Combining the intermediate properties derived in this section, the fourth-order PDE for U and ψ is summarized as follows:

$$\begin{aligned} -\rho \nabla^2 \frac{\partial U}{\partial t} + \rho < \nabla^2 U, U > - < \nabla^2 \psi, \psi > + \mu \nabla^4 U &= 0, \qquad (2.31a) \\ -\nabla^2 \frac{\partial \psi}{\partial t} + \nabla^2 < \psi, U > + \eta \nabla^4 \psi &= 0. \qquad (2.31b) \end{aligned}$$

2.2.3 Integration Identities

Equation 2.31 is converted to the weak form by integrating the test-function weighted integral form by parts [21]. The integration by parts is performed by applying Stokes' theorem and Gauss's theorem [19] that take the form as

$$\begin{aligned} \int_\Omega \nabla \times \mathbf{F} \cdot d\mathbf{S} &= \oint_{\partial\Omega} \mathbf{F} \cdot d\mathbf{r}, \qquad (2.32a) \\ \int_\Omega \nabla \cdot \mathbf{F} d\Omega &= \oint_{\partial\Omega} \mathbf{F} \cdot \hat{n} dS. \qquad (2.32b) \end{aligned}$$

Specifically, we define the integration on the RZ plane with the constant φ. We have

$$\begin{aligned} d\Omega &= dR dZ, \qquad (2.33a) \\ d\mathbf{S} &= \hat{\varphi} d\Omega, \qquad (2.33b) \end{aligned}$$

and $d\mathbf{r}$ and $\hat{n} dS$ are infinitesimal changes along the tangent and normal direction of the boundary.

From Gauss's theorem and the following two identities that take the form as

$$\nu\nabla^2 F \equiv \nabla\cdot(\nu\nabla F) - \nabla\nu\cdot\nabla F, \tag{2.34a}$$

$$\nu\nabla^4 F \equiv \nabla\cdot(\nu\nabla(\nabla^2 F)) - \nabla\cdot(\nabla^2 F\nabla\nu) + \nabla^2\nu\nabla^2 F, \tag{2.34b}$$

we have

$$\int_\Omega \nu\nabla^2 F d\Omega = \oint_{\partial\Omega} \nu\hat{n}\cdot\nabla F dS - \int_\Omega \nabla\nu\cdot\nabla F d\Omega, \tag{2.35a}$$

$$\int_\Omega \nu\nabla^4 F d\Omega = \oint_{\partial\Omega} \nu\hat{n}\cdot\left\{\nabla(\nabla^2 F)\right\} dS - \oint_{\partial\Omega} \nabla\nu\cdot\hat{n}\nabla^2 F dS + \int_\Omega \nabla^2\nu\nabla^2 F d\Omega. \tag{2.35b}$$

From Identity 2.18d, we have the identity as

$$\nabla\times(F\nabla G) \equiv \nabla F\times\nabla G + F(\nabla\times\nabla G) \equiv \nabla F\times\nabla G. \tag{2.36}$$

From Identity 2.36 and also noticing $\mathbf{F}\times\mathbf{G} = -\mathbf{G}\times\mathbf{F}$, we have

$$\nu(\nabla F\times\nabla G) \equiv \nu\left\{\nabla\times(F\nabla G)\right\} \equiv \nabla\times(\nu F\nabla G) + F\nabla G\times\nabla\nu. \tag{2.37}$$

Applying Stokes' theorem, we have the integration identity

$$\int_\Omega \nu(\nabla F\times\nabla G)\cdot d\mathbf{S} = \oint_{\partial\Omega} \nu F\nabla G\cdot d\mathbf{r} + \int_\Omega F\nabla G\times\nabla\nu\cdot d\mathbf{S}. \tag{2.38}$$

Taking the $\hat{\varphi}$ component of Equation 2.38, using the definition of the Poisson bracket in the vector form ($< F, G >= -(\nabla F\times\nabla G)\hat{\varphi}$), and reversing the direction of integral, $\oint$, (along the direction $-\hat{\varphi}$), we have the integration identity of Poisson bracket as

$$\int_\Omega \nu < F, G > d\Omega = \oint_{\partial\Omega} \nu F\left(\frac{\partial G}{\partial R}dR + \frac{\partial G}{\partial Z}dZ\right) + \int_\Omega F < G, \nu > d\Omega. \tag{2.39}$$

Specifically, replacing F with $\nabla^2 F$ in Equation 2.39, we have

$$\int_\Omega \nu < \nabla^2 F, G > d\Omega = \oint_{\partial\Omega} \nu \nabla^2 F \left(\frac{\partial G}{\partial R} dR + \frac{\partial G}{\partial Z} dZ \right) + \int_\Omega \nabla^2 F < G, \nu > d\Omega. \tag{2.40}$$

2.2.4 Weak Formulation

For simplicity of the presentation, we assume homogeneous Dirichlet boundary conditions for both U and ψ in Equation 2.31, that is, $U = \nabla U \cdot \hat{n} = \psi = \nabla\psi \cdot \hat{n} = 0$ at the boundary. Define the test function space as

$$\mathcal{V} = \{\nu \,|\, \nu \in H^2,\, \nu = \nabla\nu \cdot \hat{n} = 0 \, at \, \partial\Omega\}, \tag{2.41}$$

and the solution space as

$$\mathcal{W} = \mathcal{V}. \tag{2.42}$$

Apply Identity 2.35a to the operator ∇^2, Identity 2.35b to the operator ∇^4, and Identity 2.40 to swap $\nabla^2(\cdot)$ and the test function in Poisson bracket. The weak form of the problem defined by Equation 2.31 is stated as:

Find $(U, \psi) \in \mathcal{W} \times \mathcal{W}$, *such that for all* $(\nu, q) \in \mathcal{V} \times \mathcal{V}$,

$$\int_\Omega \left\{ \rho \nabla \frac{\partial U}{\partial t} \cdot \nabla\nu + \rho \nabla^2 U < U, \nu > - \nabla^2 \psi < \psi, \nu > + \mu \nabla^2 U \nabla^2 \nu \right\} d\Omega = 0, \tag{2.43a}$$

$$\int_\Omega \left\{ \nabla \frac{\partial \psi}{\partial t} \cdot \nabla q - \nabla < \psi, U > \cdot \nabla q + \eta \nabla^2 \psi \nabla^2 q \right\} d\Omega = 0, \tag{2.43b}$$

where the initial condition is $(U, \psi) = (U^0, \psi^0)$ at $t = 0$.

2.3 C^1 Finite Elements

C^1 finite elements are applied for spatial discretization of the fourth-order PDEs (such as Equation 2.43). The C^1 reduced quintic triangle element [3] is used on the cross-section of the tokamak. The 3D C^1 wedge element is obtained by introducing the cubic Hermite polynomial in the toroidal direction to the C^1

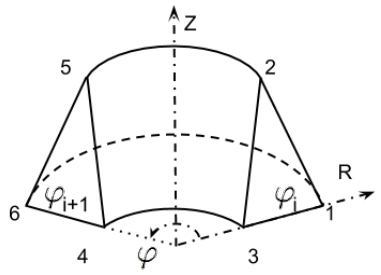


Figure 2.2: Wedge mesh element between two cross-sections in M3D-C^1.

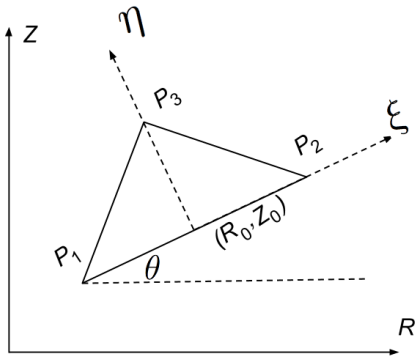


Figure 2.3: Triangle element with (ξ, η) as the local coordinate and (R, Z) as the global coordinate [3].

reduced quintic triangle element [6]. Figure 2.2 illustrates the wedge mesh element by connecting the triangle elements on two tokamak cross-sections on the $(RZ\varphi)$ coordinate system.

2.3.1 Reduced Quintic Triangle Element

A set of shape functions on the triangle with constrained fifth-order polynomials is defined to have C^1 inter-element continuity [3]. The mapping from the local coordinate (ξ, η) to the global coordinate (R, Z) is defined as

$$\begin{bmatrix} R \\ Z \end{bmatrix} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \begin{bmatrix} \xi \\ \eta \end{bmatrix} + \begin{bmatrix} R_0 \\ Z_0 \end{bmatrix}, \tag{2.44}$$

where (R_0, Z_0) is the origin point of the (ξ, η) coordinate system in Figure 2.3.

Eighteen shape functions $\{\mu_i\}$ that are fifth-order polynomials of (ξ, η) need

Table 2.1: Multipliers of the shape functions and corresponding field values associated with the vertex in the C^1 triangle element.

multiplier	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
field value	U	$U_{,\xi}$	$U_{,\eta}$	$U_{,\xi\xi}$	$U_{,\xi\eta}$	$U_{,\eta\eta}$

to be defined. The shape functions are expanded as

$$\mu_i = \sum_{k=1}^{21} \alpha_{ki} \xi^{m_k} \eta^{n_k}, \quad 0 \leq m_k + n_k \leq 5, \quad i = 1, 2, ..., 18 \tag{2.45}$$

There are twenty-one coefficients $\{\alpha_{ki}\}$ in each fifth-order polynomial. The coefficients are constrained to satisfy the following conditions which yield an eighteen-DOF C^1 element:

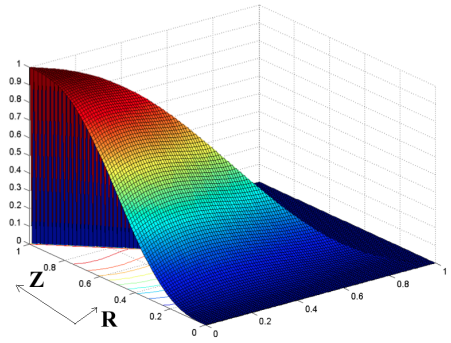
- The desired DOFs are the terms in field value, first-order and second-order derivatives at the vertices. It gives eighteen desired equations.
- Three constraints are applied that force the normal derivatives of the shape functions along the edges reduced to cubic polynomials so that C^1 inter-element continuity can be satisfied using the eighteen DOFs.

Figure 2.4 plots the first six shape functions derived by solving the linear system with twenty unknowns (the coefficient of the term $\xi^4\eta$ is zero) to obtain the coefficient of the fifth-order polynomials for each shape function. The matrix of the linear system can be found in Appendix A of Reference [3]. Due to the inclusion of the constraints, the order of polynomial completeness is fourth.

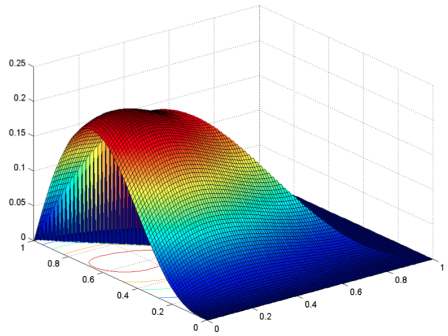
A scalar field U can be interpolated with the eighteen shape functions on the triangle as

$$U(\xi, \eta) = \sum_{i=1}^{18} \lambda_i \mu_i(\xi, \eta), \tag{2.46}$$

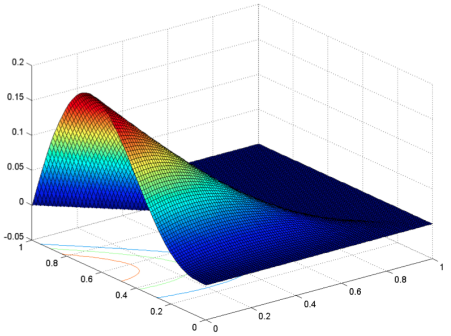
where λ_i is the multiplier associated with μ_i. From the construction of the shape functions, $\{\lambda_i\}$ equals to the value of U, $U_{,\xi}$, $U_{,\eta}$, $U_{,\xi\xi}$, $U_{,\xi\eta}$, and $U_{,\eta\eta}$ at the three vertices respectively. The first six multipliers associated with vertex P_1 in Figure 2.3 and the corresponding field values evaluated at the vertex are listed in Table 2.1.



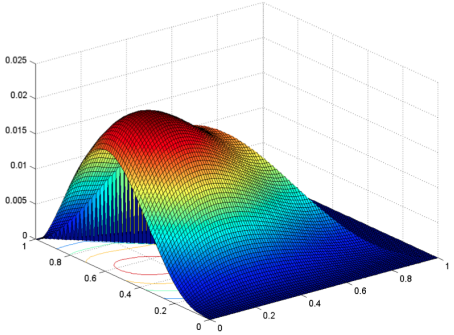
(a) function value



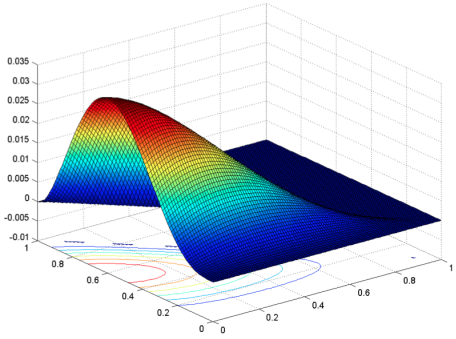
(b) 1^{st}-order derivative in R direction



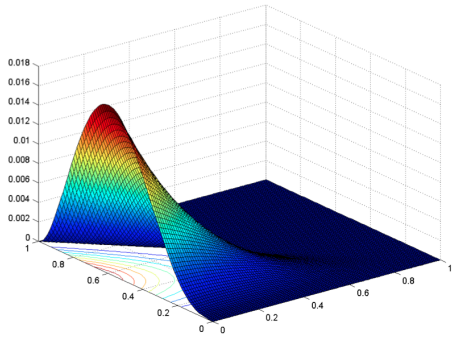
(c) 1^{st}-order derivative in Z direction



(d) 2^{nd}-order derivative in R direction



(e) twisted derivative in R, Z direction



(f) 2^{nd}-order derivative in Z direction

Figure 2.4: Shape functions associated with the vertex located at (0,1).

The desired DOFs, $[U,\ U_{,R},\ U_{,Z},\ U_{,RR},\ U_{,RZ},\ U_{,ZZ}]$, are values in the cylindrical coordinate. Thus the final steps convert the values from the local (ξ, η) coordinate system to the global (R, Z) coordinate system. The first six DOFs associated with vertex P_1 and the corresponding field values evaluated at P_1 are listed in Table 2.2.

The relations to convert the derivatives are constructed from the chain rule as

Table 2.2: DOFs and corresponding field values associated with the vertex in the C^1 triangle element.

DOF	d_1	d_2	d_3	d_4	d_5	d_6
field value	U	$U_{,R}$	$U_{,Z}$	$U_{,RR}$	$U_{,RZ}$	$U_{,ZZ}$

follows (noticing that the transformation defined by Equation 2.44 is linear):

$$U_{,\xi} = U_{,R}R_{,\xi} + U_{,Z}Z_{,\xi} = U_{,R}cos(\theta) + U_{,Z}sin(\theta), \tag{2.47a}$$

$$U_{,\eta} = U_{,R}R_{,\eta} + U_{,Z}Z_{,\eta} = -U_{,R}sin(\theta) + U_{,Z}cos(\theta), \tag{2.47b}$$

$$\begin{aligned} U_{,\xi\xi} &= U_{,RR}R_{,\xi}^2 + 2U_{,RZ}R_{,\xi}Z_{,\xi} + U_{,ZZ}Z_{,\xi}^2 \\ &= U_{,RR}cos^2(\theta) + 2U_{,RZ}sin(\theta)cos(\theta) + U_{,ZZ}sin^2(\theta), \end{aligned} \tag{2.47c}$$

$$\begin{aligned} U_{,\xi\eta} &= U_{,RR}R_{,\xi}R_{,\eta} + U_{,RZ}(R_{,\xi}Z_{,\eta} + R_{,\eta}Z_{,\xi}) + U_{,ZZ}Z_{,\xi}Z_{,\eta} \\ &= -U_{,RR}cos(\theta)sin(\theta) + U_{,RZ}[cos^2(\theta) - sin^2(\theta)] + U_{,ZZ}sin(\theta)cos(\theta), \end{aligned} \tag{2.47d}$$

$$\begin{aligned} U_{,\eta\eta} &= U_{,RR}R_{,\eta}^2 + 2U_{,RZ}R_{,\eta}Z_{,\eta} + U_{,ZZ}Z_{,\eta}^2 \\ &= U_{,RR}sin^2(\theta) - 2U_{,RZ}sin(\theta)cos(\theta) + U_{,ZZ}cos^2(\theta). \end{aligned} \tag{2.47e}$$

The full set of relations between the multipliers and DOFs is

$$\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & cos(\theta) & sin(\theta) & 0 & 0 & 0 \\ 0 & -sin(\theta) & cos(\theta) & 0 & 0 & 0 \\ 0 & 0 & 0 & cos^2(\theta) & 2sin(\theta)cos(\theta) & sin^2(\theta) \\ 0 & 0 & 0 & -sin(\theta)cos(\theta) & cos^2(\theta) - sin^2(\theta) & sin(\theta)cos(\theta) \\ 0 & 0 & 0 & sin^2(\theta) & -2sin(\theta)cos(\theta) & cos^2(\theta) \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{bmatrix}. \tag{2.48}$$

2.3.2 Cubic Hermite Polynomial

The C^1 cubic Hermite polynomial is used to interpolate the field in the toroidal φ direction (See Figure 2.1 for the coordinate system). Define $\tilde{\varphi} \in [0, 1]$ as the local

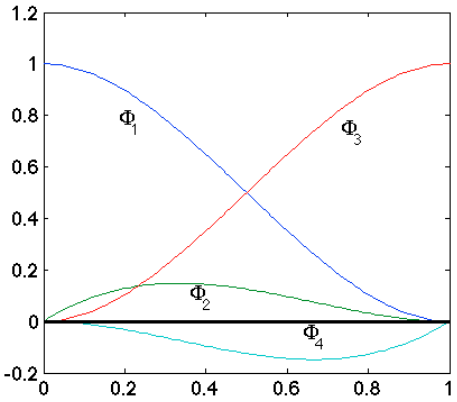


Figure 2.5: Hermite cubic polynomial defined on $[0, 1]$**.**

coordinate and the mapping to the global coordinate φ as

$$\varphi = \varphi_i + (\varphi_{i+1} - \varphi_i)\tilde{\varphi}, \tag{2.49}$$

where φ_i and φ_{i+1} are the toroidal angles of the two cross-sections (Figure 2.2).

Figure 2.5 plots the shape functions defined on the interval $[0, 1]$. The two shape functions associated with the point $\tilde{\varphi} = 0$ are

$$\Phi_1(\tilde{\varphi}) = (\tilde{\varphi} - 1)^2(2\tilde{\varphi} + 1), \tag{2.50}$$

$$\Phi_2(\tilde{\varphi}) = \tilde{\varphi}(\tilde{\varphi} - 1)^2, \tag{2.51}$$

such that $\Phi_1(0) = 1, \Phi_1'(0) = \Phi_1(1) = \Phi_1'(1) = 0$ and $\Phi_2'(0) = 1, \Phi_2(0) = \Phi_1(1) = \Phi_1'(1) = 0$. The two shape functions associated with the point $\tilde{\varphi} = 1$ are

$$\Phi_3(\tilde{\varphi}) = \tilde{\varphi}^2(3 - 2\tilde{\varphi}), \tag{2.52}$$

$$\Phi_4(\tilde{\varphi}) = \tilde{\varphi}^2(\tilde{\varphi} - 1). \tag{2.53}$$

A scalar field U can be interpolated with the four shape functions on the interval as

$$U(\tilde{\varphi}) = \sum_{i=1}^{4} \lambda_i \Phi_i(\tilde{\varphi}), \tag{2.54}$$

where λ_i is the multiplier associated with Φ_i. From the construction of the shape

Table 2.3: Multipliers of the shape functions and corresponding field values associated with the vertex in the C^1 cubic Hermite Element.

multiplier	λ_1	λ_2	λ_3	λ_4
field value	$U(0)$	$U_{,\tilde{\varphi}}(0)$	$U(1)$	$U_{,\tilde{\varphi}}(1)$

Table 2.4: DOFs and corresponding field values associated with the vertex in the C^1 cubic Hermite element.

DOFs	d_1	d_2	d_3	d_4
field value	$U(\varphi_i)$	$U'(\varphi_i)$	$U(\varphi_{i+1})$	$U'(\varphi_{i+1})$

functions, $\{\lambda_i\}$ equal to the value of $U, U_{,\tilde{\varphi}}$ at the two points of the interval. The four multipliers associated with the two points $\tilde{\varphi} = 0, 1$ and the corresponding field values are listed in Table 2.3.

Four DOFs are defined in the 1D cubic Hermite element and the values are equal to U and $U_{,\varphi}$ at the two points, respectively. The DOFs and the corresponding field values on $[\varphi_i, \varphi_{i+1}]$ are listed in Table 2.4 ($U' \equiv U_{,\varphi}$ is used). The relations between λ_i and d_i for the cubic Hermite element from the chain rule are

$$\lambda_1 = d_1, \tag{2.55a}$$

$$\lambda_2 = (\varphi_{i+1} - \varphi_i) d_2, \tag{2.55b}$$

$$\lambda_3 = d_3, \tag{2.55c}$$

$$\lambda_4 = (\varphi_{i+1} - \varphi_i) d_4. \tag{2.55d}$$

2.3.3 C^1 Wedge Element

The 3D C^1 element is defined on the wedge (Figure 2.2) by combining the C^1 triangle element and the C^1 Hermite polynomial. Define the local coordinate $(\xi, \eta, \tilde{\varphi})$ and the mapping to the global coordinate (R, Z, φ) as

$$\begin{bmatrix} R \\ Z \\ \varphi \end{bmatrix} = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & \varphi_{i+1} - \varphi_i \end{bmatrix} \begin{bmatrix} \xi \\ \eta \\ \tilde{\varphi} \end{bmatrix} + \begin{bmatrix} R_0 \\ Z_0 \\ \varphi_i \end{bmatrix}. \tag{2.56}$$

Field U is interpolated in the wedge mesh element (Figure 2.2) with the shape

Table 2.5: Multipliers of the shape functions and corresponding field values associated with the vertex in the C^1 wedge element.

multiplier	λ_{11}	λ_{21}	λ_{31}	λ_{41}	λ_{51}	λ_{61}
field value	U	$U_{,\xi}$	$U_{,\eta}$	$U_{,\xi\xi}$	$U_{,\xi\eta}$	$U_{,\eta\eta}$
multiplier	λ_{12}	λ_{22}	λ_{32}	λ_{42}	λ_{52}	λ_{62}
field value	$U, \tilde{\varphi}$	$U_{,\xi\tilde{\varphi}}$	$U_{,\eta\tilde{\varphi}}$	$U_{,\xi\xi\tilde{\varphi}}$	$U_{,\xi\eta\tilde{\varphi}}$	$U_{,\eta\eta\tilde{\varphi}}$

functions of the C^1 triangle $\{\mu_i\}$ and the cubic Hermite $\{\Phi_i\}$ as

$$U(\xi,\eta,\tilde{\varphi}) = \left(\sum_{i=1}^{18} \alpha_i \mu_i(\xi,\eta)\right) \cdot \left(\sum_{j=1}^{4} \beta_j \Phi_j(\tilde{\varphi})\right), \tag{2.57}$$

where α_i and β_j are coefficients of μ_i and Φ_j, respectively.

It can be further written as

$$U = \sum_{i=1}^{18}\sum_{j=1}^{4} \lambda_{ij}\mu_i(\xi,\eta)\Phi_j(\tilde{\varphi}), \tag{2.58}$$

where $\lambda_{ij} = \alpha_i\beta_j$ is the multiplier of the shape function $\mu_i(\xi,\eta)\Phi_j(\tilde{\varphi})$.

There are totally seventy-two (18×4) multipliers in the wedge elements. From the properties of μ_i and Φ_j, the seventy-two multipliers equal to the values of U, $U_{,\xi}$, $U_{,\eta}$, $U_{,\xi\xi}$, $U_{,\xi\eta}$, $U_{,\eta\eta}$, $U_{,\tilde{\varphi}}$, $U_{,\xi\tilde{\varphi}}$, $U_{,\eta\tilde{\varphi}}$, $U_{,\xi\xi\tilde{\varphi}}$, $U_{,\xi\eta\tilde{\varphi}}$ and $U_{,\eta\eta\tilde{\varphi}}$ evaluated at the six vertices of the wedge mesh element. The twelve multipliers associated with vertex 1 in Figure 2.2 and the corresponding field values evaluated at the vertex are listed in Table 2.5.

Seventy-two DOFs are defined in the C^1 wedge element. The DOFs equal to values of U, $U_{,R}$, $U_{,Z}$, $U_{,RR}$, $U_{,RZ}$, $U_{,ZZ}$, $U_{,\varphi}$, $U_{,R\varphi}$, $U_{,Z\varphi}$, $U_{,RR\varphi}$, $U_{,RZ\varphi}$ and $U_{,ZZ\varphi}$ at the six vertices, respectively. The first twelve DOFs associated with vertex 1 in Figure 2.2 and the corresponding field values evaluated at the vertex are listed in Table 2.6.

The relation between the multipliers of the shape functions and DOFs are obtained by combing Equation 2.48 and Equation 2.55.

Table 2.6: DOFs and corresponding field values associated with the vertex in the C^1 wedge element.

DOF	d_1	d_2	d_3	d_4	d_5	d_6
field value	U	$U_{,R}$	$U_{,Z}$	$U_{,RR}$	$U_{,RZ}$	$U_{,ZZ}$
DOF	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}
field value	$U_{,\varphi}$	$U_{,R\varphi}$	$U_{,Z\varphi}$	$U_{,RR\varphi}$	$U_{,RZ\varphi}$	$U_{,ZZ\varphi}$

CHAPTER 3
Explicit *a posteriori* Estimator

Adaptive mesh control [22] is an efficient method to improve the reliability of simulation results. The procedure of mesh adaptation is driven by the mesh size field that is aimed to equally distribute the estimated error between mesh elements [23, 24]. In order to obtain the desired mesh size field, it is necessary to estimate the error distribution in the solution.

There are limited studies on the error estimation in the area of adaptive MHD simulations. The error is estimated either from *a priori* knowledge of the solution property or from *a posteriori* evaluation of the solution field.

The *a priori* approach requires of a reasonable understanding and prediction of the physics. Error indicators based on the post processed fields of specific solution properties are used in the MHD simulation [25, 26, 27, 28]. The approach has been applied by M3D-C^1 using the magnetic flux field in the plasma equilibrium ([29], also see Section 5.4.2).

The *a posteriori* approach [22, 30] is better suited to explicitly control the discretization error. M3D-C^1 uses a C^1 finite element method and it is difficult to apply methods by recovering first-order derivatives such as the superconvergent patch recovery technique [31, 24]. Richardson extrapolation is applied in multilevel adaptive mesh refinement (AMR) [32]. However, it requires a multilevel mesh representation. [33] applies the error indicator by minimizing a mesh energy functional with moving mesh techniques.

This chapter presents an explicit *a posteriori* estimator [30, 34, 35, 36] developed for M3D-C^1. The model for 2D incompressible MHD (Section 2.2.1) is used for simplicity of the presentation. Section 3.1 states the problem in terms of the bilinear and trilinear forms, and defines the corresponding error equation. Section 3.1 follows the methodology proposed by [37]. Section 3.2 makes use of the results from [34, 36, 22, 38] and defines the error estimator. Section 3.3 discusses the aspects of implementation. Section 3.4 illustrates the effectiveness of the error estimator.

3.1 Model Problem

The problem defined by Equation 2.43 in Section 2.2.4 is considered. For simplicity of the notation, assume $\rho = 1$. Define the following two bilinear forms:

$$a_1(u,v) = \int_\Omega \nabla u \cdot \nabla v d\Omega, \tag{3.1a}$$

$$a_2(u,v) = \int_\Omega \nabla^2 u \nabla^2 v d\Omega. \tag{3.1b}$$

Define the following two trilinear forms:

$$c_1(u,v,w) = \int_\Omega \nabla^2 u < v,w > d\Omega, \tag{3.2a}$$

$$c_2(u,v,w) = -\int_\Omega \nabla < u,v > \cdot \nabla w d\Omega, \tag{3.2b}$$

where $<,>$ is the Poisson bracket defined as (also see Equation 2.19)

$$< F,G > \equiv \frac{\partial F}{\partial R}\frac{\partial G}{\partial Z} - \frac{\partial F}{\partial Z}\frac{\partial G}{\partial R}. \tag{3.3}$$

Note that the Poisson bracket defined by Equation 3.3 is a bilinear form.

With the definitions of the bilinear and trilinear forms, the problem defined by Equation 2.43 is stated as:

Find $(U,\psi) \in \mathcal{W} \times \mathcal{W}$*, such that for all* $(\nu,q) \in \mathcal{V} \times \mathcal{V}$,

$$a_1(\frac{\partial U}{\partial t},\nu) + c_1(U,U,\nu) - c_1(\psi,\psi,\nu) + \mu a_2(U,\nu) = 0, \tag{3.4a}$$

$$a_1(\frac{\partial \psi}{\partial t},q) + c_2(\psi,U,q) + \eta a_2(\psi,\nu) = 0, \tag{3.4b}$$

where the initial condition is $(U,\psi) = (U^0,\psi^0)$ at $t = 0$, and $\mathcal{V}$ and $\mathcal{W}$ are the sub-spaces of H^2 defined by Equation 2.41 and Equation 2.42.

Define the finite-dimensional test function space, $\mathcal{V}_h \in \mathcal{V}$, and the finite-dimensional solution space, $\mathcal{W}_h \in \mathcal{W}$. Assume $(U_h,\psi_h) \in \mathcal{W}_h \times \mathcal{W}_h$ is the finite-element solution. Define the spatial discretization error as

$$(e,E) = (U - U_h, \psi - \psi_h) \in \mathcal{V} \times \mathcal{V}. \tag{3.5}$$

Substituting $U = U_h + e$ and $\psi = \psi_h + E$ into Equation 3.4, we obtain the error equation as [37]

$$a_1(\frac{\partial e}{\partial t}, \nu) + \left\{ \begin{array}{c} c_1(e, U_h, \nu) \\ +c_1(U_h, e, \nu) \\ +c_1(e, e, \nu) \end{array} \right\} - \left\{ \begin{array}{c} c_1(E, \psi_h, \nu) \\ +c_1(\psi_h, E, \nu) \\ +c_1(E, E, \nu) \end{array} \right\} + \mu a_2(e, \nu)$$

$$= \left\{ \begin{array}{c} -a_1(\frac{\partial U_h}{\partial t}, \nu) \\ -c_1(U_h, U_h, \nu) \\ +c_1(\psi_h, \psi_h, \nu) \\ -\mu a_2(U_h, \nu) \end{array} \right\}, \qquad (3.6a)$$

$$a_1(\frac{\partial E}{\partial t}, q) + \left\{ \begin{array}{c} c_2(E, U_h, q) \\ +c_2(\psi_h, e, q) \\ +c_2(E, e, q) \end{array} \right\} + \eta a_2(E, q) = \left\{ \begin{array}{c} -a_1(\frac{\partial \psi_h}{\partial t}, q) \\ -c_2(\psi_h, U_h, q) \\ -\eta a_2(\psi_h, q) \end{array} \right\}. \qquad (3.6b)$$

Note that the subscripted quantity by letter h, $(\cdot)_h$, is the spatial discretized form of the quantity, $(\cdot)$.

We assume the temporal discretization error will be kept small compared with the spatial discretization error by choosing a sufficiently small time step, δt [37]. Under this assumption, a simple scheme is used for the temporal discretization. Notationally, the superscript n, $(\cdot)^n$, is used to represent the discretized form of $(\cdot)$ at $t = n\delta t$. Applying the implicit backward Euler method for the bilinear terms, and the explicit forward Euler for the trilinear terms in Equation 3.6, we obtain the temporal discretized error equation as

$$\frac{a_1(e^{n+1}, \nu)}{\delta t} + \mu a_2(e^{n+1}, \nu) = \left\{ \begin{array}{c} -\frac{a_1(U_h^{n+1} - U_h^n, \nu)}{\delta t} \\ -c_1(U_h^n, U_h^n, \nu) \\ +c_1(\psi_h^n, \psi_h^n, \nu) \\ -\mu a_2(U_h^{n+1}, \nu) \end{array} \right\} + \mathcal{R}_h^U(e^n, E^n), \qquad (3.7a)$$

$$\frac{a_1(E^{n+1}, q)}{\delta t} + \eta a_2(E^{n+1}, q) = \left\{ \begin{array}{c} -\frac{a_1(\psi_h^{n+1} - \psi_h^n, q)}{\delta t} \\ -c_2(\psi_h^n, U_h^n, q) \\ -\eta a_2(\psi_h^{n+1}, q) \end{array} \right\} + \mathcal{R}_h^\psi(e^n, E^n). \qquad (3.7b)$$

where

$$\mathcal{R}_h^U(e^n, E^n) = \frac{a_1(e^n, \nu)}{\delta t} - \left\{ \begin{array}{c} c_1(e^n, U_h^n, \nu) \\ +c_1(U_h^n, e^n, \nu) \\ +c_1(e^n, e^n, \nu) \end{array} \right\} + \left\{ \begin{array}{c} c_1(E^n, \psi_h^n, \nu) \\ +c_1(\psi_h^n, E^n, \nu) \\ +c_1(E^n, E^n, \nu) \end{array} \right\}, \tag{3.8a}$$

$$\mathcal{R}_h^\psi(e^n, E^n) = \frac{a_1(E^n, q)}{\delta t} - \left\{ \begin{array}{c} c_2(E^n, U_h^n, q) \\ +c_2(\psi_h^n, e^n, q) \\ +c_2(E^n, e^n, q) \end{array} \right\}. \tag{3.8b}$$

$\mathcal{R}_h^U(e^n, E^n)$ and $\mathcal{R}_h^\psi(e^n, E^n)$ are set to zero by arguing that the history-accumulated error does not represent the mesh areas that need better resolution at this time [37].

Alternatively, Equation 3.7 can also be obtained by applying both the temporal and spatial discretization, subtracting the fully-discretized solution from Equation 3.4, and dropping the truncation error of the temporal discretization.

3.2 Explicit *a posteriori* Estimator

We apply the explicit *a posteriori* estimator of [30] that directly computes a mesh-dependent norm of the residual in the strong form [34]. The same type of the error estimators are studied for the Stokes equation with the Dirichlet boundary conditions [34], the Poisson equation with the Dirichlet-Neumann boundary conditions [36], the biharmonic equation with the Dirichlet boundary conditions [22, 38], and the convection-diffusion equations with the Dirichlet-Neumann boundary conditions [39]. We summarize the key steps in constructing an explicit error estimator using the results from the papers referred to, particularly [30].

Key terms used in describing the error estimator include:

$\mathcal{P}$: triangulation of the 2D domain,

$\mathcal{K} \in \mathcal{P}$: a triangle element in $\mathcal{P}$ with the boundary as $\partial\mathcal{K}$ that is made up of three edges,

$\Gamma \in \partial\mathcal{P}$: a single edge element,

γ: angle between edge Γ and $\hat{R}$,

$h_{\mathcal{K}} = diam(\mathcal{K})$: diameter of triangle $\mathcal{K}$ (the length of the longest edge in the triangle is used here),

$\|F\|^2_{L_2(\mathcal{K})} = \int_{\mathcal{K}} F^2 d\Omega$: L_2 integration of f on element $\mathcal{K}$,

$\|F\|^2_{L_2(\Gamma)} = \int_{\Gamma} F^2 d\Gamma$: L_2 integration of f on edge Γ,

$[\cdot]_{\Gamma} = (\cdot)_{\Gamma\in\mathcal{K}} - (\cdot)_{\Gamma\in\mathcal{J}}$ will represent the jump discontinuity along the interior mesh edge, Γ, that is shared by element $\mathcal{K}$ and $\mathcal{J}$. Note that the normal and tangent directions of edge Γ are defined in element $\mathcal{K}$.

Figure 3.1 illustrates the terms defined in the section. The interior edge, Γ, is shared by Element $\mathcal{K}$ and $\mathcal{J}$. Assume $\hat{n}$, $\hat{n}'$, $\hat{\tau}$, and $\hat{\tau}'$ are the unit vectors in the normal and tangent directions of edge Γ in Element $\mathcal{K}$ and $\mathcal{J}$, respectively. Note that

$$\hat{n}' = -\hat{n}, \tag{3.9a}$$

$$\hat{\tau}' = -\hat{\tau}. \tag{3.9b}$$

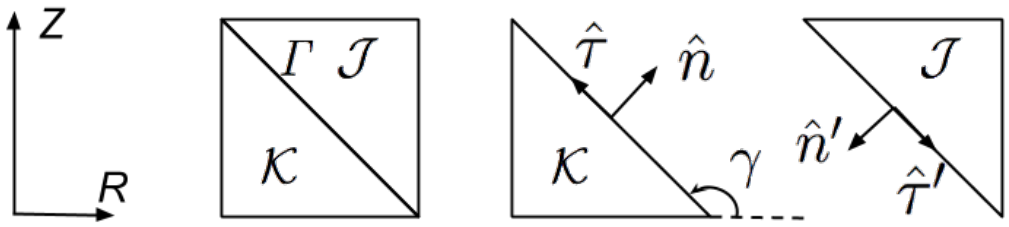


Figure 3.1: Illustration of terms used by Section 3.2.

Assume the normal and tangent directions of edge Γ are $\hat{n}$ and $\hat{\tau}$ (the same as is defined in element $\mathcal{K}$). Two types of jump discontinuity are considered:

- Normal flux jump comes from the discontinuity of the term, $\frac{\partial F}{\partial n} \equiv \nabla F \cdot \hat{n}$, due to the application of Gauss's theorem (Equation 2.32b) and integration

identities derived from it (Equation 2.35b). We have the following relation in Figure 3.1:

$$\begin{aligned}\left(\frac{\partial F}{\partial n}\right)_{\Gamma\in\mathcal{K}}+\left(\frac{\partial F}{\partial n'}\right)_{\Gamma\in\mathcal{J}} &= (\nabla F)_{\Gamma\in\mathcal{K}}\cdot\hat{n}+(\nabla F)_{\Gamma\in\mathcal{J}}\cdot\hat{n}' \\ &= \left(\frac{\partial F}{\partial n}\right)_{\Gamma\in\mathcal{K}}-\left(\frac{\partial F}{\partial n}\right)_{\Gamma\in\mathcal{J}}. \end{aligned} \tag{3.10}$$

It defines the jump discontinuity of the normal flux as

$$\left[\frac{\partial F}{\partial n}\right]_{\Gamma}=\left(\frac{\partial F}{\partial n}\right)_{\Gamma\in\mathcal{K}}-\left(\frac{\partial F}{\partial n}\right)_{\Gamma\in\mathcal{J}}. \tag{3.11}$$

- Discontinuity of the "force", $\mathbf{F}$, in the tangent direction is due to the application of Stokes' theorem (Equation 2.32a) and integration identities derived from it (Equation 2.40). We have the following relation in Figure 3.1:

$$(\mathbf{F}\cdot\hat{\tau})_{\Gamma\in\mathcal{K}}+(\mathbf{F}\cdot\hat{\tau'})_{\Gamma\in\mathcal{J}}=(\mathbf{F}\cdot\hat{\tau})_{\Gamma\in\mathcal{K}}-(\mathbf{F}\cdot\hat{\tau})_{\Gamma\in\mathcal{J}}\,. \tag{3.12}$$

Specifically, we replace $\mathbf{F}$ with ∇F and use the following relation:

$$\hat{\tau}=cos(\gamma)\hat{R}+sin(\gamma)\hat{Z}, \tag{3.13}$$

where γ is the angle between $\hat{\tau}$ and the R axis (Figure 3.1). This defines the jump discontinuity in the tangent direction as

$$\begin{aligned}[\nabla F\cdot\hat{\tau}]_{\Gamma} &= \left[\frac{\partial F}{\partial R}cos(\gamma)+\frac{\partial F}{\partial Z}sin(\gamma)\right]_{\Gamma} \\ =\left(\frac{\partial F}{\partial R}cos(\gamma)+\frac{\partial F}{\partial Z}sin(\gamma)\right)_{\Gamma\in\mathcal{K}} &-\left(\frac{\partial F}{\partial R}cos(\gamma)+\frac{\partial F}{\partial Z}sin(\gamma)\right)_{\Gamma\in\mathcal{J}}. \end{aligned} \tag{3.14}$$

Note that the solution using the finite element space, $\mathcal{V}_h$, for the spatial discretization, and the temporal discretization of the forward Euler on the bilinear

forms and the backward Euler on the trilinear forms is U_h^n that satisfies

$$\frac{a_1(U_h^{n+1}-U_h^n,\nu_h)}{\delta t}+c_1(U_h^n,U_h^n,\nu_h)-c_1(\psi_h^n,\psi_h^n,\nu_h)+\mu a_2(U_h^{n+1},\nu_h) = 0, \tag{3.15a}$$

$$\frac{a_1(\psi_h^{n+1}-\psi_h^n,q_h)}{\delta t}+c_2(U_h^n,\psi_h^n,q_h)+\eta a_2(\psi_h^{n+1},q_h) = 0. \tag{3.15b}$$

Adding Equation 3.15 to the right-hand-side of Equation 3.7 (Noticing $\mathcal{R}_h^U(e^n,E^n)$ and $\mathcal{R}_h^\psi(e^n,E^n)$ are set to zero), we obtain

$$\frac{a_1(e^{n+1},\nu)}{\delta t}+\mu a_2(e^{n+1},\nu) = \left\{\begin{array}{l} -\frac{a_1(U_h^{n+1}-U_h^n,\nu-\nu_h)}{\delta t} \\ -c_1(U_h^n,U_h^n,\nu-\nu_h) \\ +c_1(\psi_h^n,\psi_h^n,\nu-\nu_h) \\ -\mu a_2(U_h^{n+1},\nu-\nu_h) \end{array}\right\}, \tag{3.16a}$$

$$\frac{a_1(E^{n+1},q)}{\delta t}+\eta a_2(E^{n+1},q) = \left\{\begin{array}{l} -\frac{a_1(\psi_h^{n+1}-\psi_h^n,q-q_h)}{\delta t} \\ -c_2(\psi_h^n,U_h^n,q-q_h) \\ -\eta a_2(\psi_h^{n+1},q-q_h) \end{array}\right\}. \tag{3.16b}$$

It can be seen that the weighting functions on the right-hand-side of Equation 3.16 are changed to $(\nu-\nu_h,q-q_h)$ compared with Equation 3.7. To be more exact, $(\nu_h,q_h)\in\mathcal{V}_h\times\mathcal{V}_h$ in Equation 3.16 is the interpolant of $(\nu,q)\in\mathcal{V}\times\mathcal{V}$ that satisfies Theorem 1.1 in [30]. [40] gives the definition of the interpolant in Hilbert spaces.

Integrate the right-hand-side of Equation 3.16 by parts (see Equations 2.35a,

2.35b and 2.40). Specifically, we have from Equation 2.35b

$$
\begin{aligned}
a_2(U_h^{n+1}, \nu - \nu_h) &= \int_\Omega \nabla^2 U_h^{n+1} \nabla^2(\nu - \nu_h) d\Omega \\
&= \sum_{\mathcal{K}\in\mathcal{P}} \int_{\mathcal{K}} \nabla^2 U_h^{n+1} \nabla^2(\nu - \nu_h) d\Omega \\
&= \sum_{\mathcal{K}\in\mathcal{P}} \left\{ \begin{array}{l} \int_{\mathcal{K}} (\nu - \nu_h) \nabla^4 U_h^{n+1} d\Omega \\ - \oint_{\partial\mathcal{K}} (\nu - \nu_h) \hat{n} \cdot (\nabla \nabla^2 U_h^{n+1}) dS \\ + \oint_{\partial\mathcal{K}} \nabla(\nu - \nu_h) \cdot \hat{n} \nabla^2 U_h^{n+1} dS \end{array} \right\} \\
&= \sum_{\mathcal{K}\in\mathcal{P}} \int_{\mathcal{K}} (\nu - \nu_h) \nabla^4 U_h^{n+1} d\Omega \\
&\quad + \sum_{\Gamma\in\partial\mathcal{P}\setminus\partial\Omega} \int_\Gamma \left\{ \begin{array}{l} (\nu - \nu_h) \left[-\frac{\partial \nabla^2 U_h^{n+1}}{\partial n} \right]_\Gamma \\ + \nabla(\nu - \nu_h) \cdot \hat{n} \left[\nabla^2 U_h^{n+1} \right]_\Gamma \end{array} \right\} dS.
\end{aligned}
\tag{3.17a}
$$

Note that we applied the following properties:

$$
\hat{n} \cdot (\nabla \nabla^2 U_h^{n+1}) = \frac{\partial \nabla^2 U_h^{n+1}}{\partial n} \tag{3.18a}
$$

$$
\left[-(\nu - \nu_h) \frac{\partial \nabla^2 U_h^{n+1}}{\partial n} \right]_\Gamma = (\nu - \nu_h) \left[-\frac{\partial \nabla^2 U_h^{n+1}}{\partial n} \right]_\Gamma, \tag{3.18b}
$$

$$
\left[\nabla(\nu - \nu_h) \cdot \hat{n} \nabla^2 U_h^{n+1} \right]_\Gamma = \nabla(\nu - \nu_h) \cdot \hat{n} \left[\nabla^2 U_h^{n+1} \right]_\Gamma. \tag{3.18c}
$$

Similarly, we have from Equation 2.35a

$$
\begin{aligned}
a_1(U_h^{n+1} - U_h^n, \nu - \nu_h) &= \int_\Omega \nabla(U_h^{n+1} - U_h^n) \cdot \nabla(\nu - \nu_h) d\Omega \\
&= \sum_{\mathcal{K}\in\mathcal{P}} \int_{\mathcal{K}} \nabla(U_h^{n+1} - U_h^n) \cdot \nabla(\nu - \nu_h) d\Omega \\
&= \sum_{\mathcal{K}\in\mathcal{P}} \left\{ \begin{array}{l} - \int_{\mathcal{K}} (\nu - \nu_h) \nabla^2 (U_h^{n+1} - U_h^n) d\Omega \\ + \oint_{\partial\mathcal{K}} (\nu - \nu_h) \hat{n} \cdot \nabla (U_h^{n+1} - U_h^n) \end{array} \right\} dS \\
&= \sum_{\mathcal{K}\in\mathcal{P}} - \int_{\mathcal{K}} (\nu - \nu_h) \nabla^2 (U_h^{n+1} - U_h^n) d\Omega.
\end{aligned}
\tag{3.19a}
$$

Note that we used the homogeneous Dirichlet boundary conditions and the property

of the C^1 continuity in the solution.

The term of $c_1(U_h^n, U_h^n, \nu - \nu_h)$ is integrated by parts from Equation 2.40 as

$$\begin{aligned} c_1(U_h^n, U_h^n, \nu - \nu_h) &= \int_\Omega \nabla^2 U_h^n < U_h^n, \nu - \nu_h > d\Omega \\ &= \sum_{\mathcal{K}\in\mathcal{P}} \int_{\mathcal{K}} \nabla^2 U_h^n < U_h^n, \nu - \nu_h > d\Omega \\ &= \sum_{\mathcal{K}\in\mathcal{P}} \left\{ \begin{array}{l} \int_{\mathcal{K}} (\nu - \nu_h) < \nabla^2 U_h^n, U_h^n > d\Omega \\ - \oint_{\partial\mathcal{K}} (\nu - \nu_h) \nabla^2 U_h^n \left(\frac{\partial U_h^n}{\partial R} dR + \frac{\partial U_h^n}{\partial Z} dZ \right) \end{array} \right\} \\ &= \sum_{\mathcal{K}\in\mathcal{P}} \int_{\mathcal{K}} (\nu - \nu_h) < \nabla^2 U_h^n, U_h^n > d\Omega \\ &\quad + \sum_{\Gamma\in\partial\mathcal{P}\setminus\partial\Omega} - \int_\Gamma (\nu - \nu_h) \left[\nabla^2 U_h^n \left(\begin{array}{l} \frac{\partial U_h^n}{\partial R} cos(\gamma) \\ + \frac{\partial U_h^n}{\partial Z} sin(\gamma) \end{array} \right) \right]_\Gamma dS. \end{aligned} \tag{3.20a}$$

Note that we used the following property (also see Equation 3.13):

$$dR = cos(\gamma) dS, \tag{3.21a}$$

$$dZ = sin(\gamma) dS. \tag{3.21b}$$

The other terms in Equation 3.16 are integrated by parts similarly. This leads to the error equation written as the element residual and the jump discontinuity across the mesh edge:

$$\frac{a_1(e^{n+1}, \nu)}{\delta t} + \mu a_2(e^{n+1}, \nu) = \left\{ \begin{array}{l} \sum_{\mathcal{K}\in\mathcal{P}} \int_{\mathcal{K}} (\nu - \nu_h) \mathcal{R}_1^U d\Omega \\ + \sum_{\Gamma\in\partial\mathcal{P}\setminus\partial\Omega} \int_\Gamma (\nu - \nu_h) [\mathcal{R}_2^U]_\Gamma dS \\ + \sum_{\Gamma\in\partial\mathcal{P}\setminus\partial\Omega} \int_\Gamma \nabla(\nu - \nu_h) \cdot \hat{n} [\mathcal{R}_3^U]_\Gamma dS \end{array} \right\}, \tag{3.22a}$$

$$\frac{a_1(E^{n+1}, q)}{\delta t} + \eta a_2(E^{n+1}, q) = \left\{ \begin{array}{l} \sum_{\mathcal{K}\in\mathcal{P}} \int_{\mathcal{K}} (q - q_h) \mathcal{R}_1^\psi d\Omega \\ + \sum_{\Gamma\in\partial\mathcal{P}\setminus\partial\Omega} \int_\Gamma (q - q_h) [\mathcal{R}_2^\psi]_\Gamma dS \\ + \sum_{\Gamma\in\partial\mathcal{P}\setminus\partial\Omega} \int_\Gamma \nabla(q - q_h) \cdot \hat{n} [\mathcal{R}_3^\psi]_\Gamma dS \end{array} \right\}. \tag{3.22b}$$

The R terms are defined as:

$$\mathcal{R}_1^U = \frac{1}{\delta t}\nabla^2(U_h^{n+1} - U_h^n) + \left\{ \begin{array}{l} - < \nabla^2 U_h^n, U_h^n > \\ + < \nabla^2 \psi_h^n, \psi_h^n > \end{array} \right\} - \mu\nabla^4 U_h^{n+1}, \quad (3.23a)$$

$$\mathcal{R}_1^\psi = \frac{1}{\delta t}\nabla^2(\psi_h^{n+1} - \psi_h^n) - \nabla^2 < \psi_h^n, U_h^n > -\eta\nabla^4\psi_h^{n+1}, \quad (3.23b)$$

$$\mathcal{R}_2^U = \left\{ \begin{array}{l} \nabla^2 U_h^n \left(\frac{\partial U_h^n}{\partial R} cos(\gamma) + \frac{\partial U_h^n}{\partial Z} sin(\gamma) \right) \\ -\nabla^2 \psi_h^n \left(\frac{\partial \psi_h^n}{\partial R} cos(\gamma) + \frac{\partial \psi_h^n}{\partial Z} sin(\gamma) \right) \end{array} \right\} + \mu \frac{\partial \nabla^2 U_h^{n+1}}{\partial n}, \quad (3.23c)$$

$$\mathcal{R}_2^\psi = \frac{\partial}{\partial n} < \psi_h^n, U_h^n > +\eta \frac{\partial \nabla^2 \psi_h^{n+1}}{\partial n}, \quad (3.23d)$$

$$\mathcal{R}_3^U = -\mu\nabla^2 U_h^{n+1}, \quad (3.23e)$$

$$\mathcal{R}_3^\psi = -\eta\nabla^2\psi_h^{n+1}, \quad (3.23f)$$

Define the energy norms for U and ψ on $\mathcal{V}$ as

$$|||e|||_U = \sqrt{\frac{a_1(e,e)}{\delta t} + \mu a_2(e,e)}, \quad (3.24a)$$

$$|||E|||_\psi = \sqrt{\frac{a_1(E,E)}{\delta t} + \eta a_2(E,E)}. \quad (3.24b)$$

It can be seen that the energy norms are equivalent to the H^2 norms defined on $\mathcal{V}$ [34].

From the approximation theory in H^2 space [40, 30] (setting $s = r = l = 2$ and $m = 0$ for Theorem 1.1 in [30]), and the procedure for defining the explicit *a posteriori* estimator in [30] (setting $\nu = e^{n+1}$ and $q = E^{n+1}$ in Equation 3.22, applying Cauchy-Schwarz Inequality and Theorem 1.1 in [30], and using the equivalence

of the energy norm as the H^2 norm), we obtain the *a posteriori* estimate as

$$|||e^{n+1}|||_U \leq C \left\{ \begin{array}{l} \sum_{\mathcal{K}\in\mathcal{P}} h_{\mathcal{K}}^4 \left\|\mathcal{R}_1^U\right\|_{L_2(\mathcal{K})}^2 \\ +\sum_{\Gamma\in\partial\mathcal{P}\setminus\partial\Omega} h_{\mathcal{K}}^3 \left\|\left[\mathcal{R}_2^U\right]_\Gamma\right\|_{L_2(\Gamma)} \\ +\sum_{\Gamma\in\partial\mathcal{P}\setminus\partial\Omega} h_{\mathcal{K}} \left\|\left[\mathcal{R}_3^U\right]_\Gamma\right\|_{L_2(\Gamma)} \end{array} \right\}^{\frac{1}{2}}, \tag{3.25a}$$

$$|||E^{n+1}|||_\psi \leq C \left\{ \begin{array}{l} \sum_{\mathcal{K}\in\mathcal{P}} h_{\mathcal{K}}^4 \left\|\mathcal{R}_1^\psi\right\|_{L_2(\mathcal{K})}^2 \\ +\sum_{\Gamma\in\partial\mathcal{P}\setminus\partial\Omega} h_{\mathcal{K}}^3 \left\|\left[\mathcal{R}_2^\psi\right]_\Gamma\right\|_{L_2(\Gamma)} \\ +\sum_{\Gamma\in\partial\mathcal{P}\setminus\partial\Omega} h_{\mathcal{K}} \left\|\left[\mathcal{R}_3^\psi\right]_\Gamma\right\|_{L_2(\Gamma)} \end{array} \right\}^{\frac{1}{2}}. \tag{3.25b}$$

The local error estimator on element $\mathcal{K}$ is defined by

$$\epsilon_{U\mathcal{K}}^2 = h_{\mathcal{K}}^4 ||\mathcal{R}_1^U||_{L_2(\mathcal{K})}^2 + \frac{1}{2} \sum_{\Gamma\in\partial\mathcal{K}\setminus\partial\Omega} \left(h_{\mathcal{K}}^3 \left\|\left[\mathcal{R}_2^U\right]_\Gamma\right\|_{L_2(\Gamma)}^2 + h_{\mathcal{K}} \left\|\left[\mathcal{R}_3^U\right]_\Gamma\right\|_{L_2(\Gamma)}^2 \right), \tag{3.26a}$$

$$\epsilon_{\psi\mathcal{K}}^2 = h_{\mathcal{K}}^4 ||\mathcal{R}_1^\psi||_{L_2(\mathcal{K})}^2 + \frac{1}{2} \sum_{\Gamma\in\partial\mathcal{K}\setminus\partial\Omega} \left(h_{\mathcal{K}}^3 \left\|\left[\mathcal{R}_2^\psi\right]_\Gamma\right\|_{L_2(\Gamma)}^2 + h_{\mathcal{K}} \left\|\left[\mathcal{R}_3^\psi\right]_\Gamma\right\|_{L_2(\Gamma)}^2 \right). \tag{3.26b}$$

Note that the local error estimator reduces to the result in [38] for the biharmonic equation with the Dirichlet boundary conditions by dropping the non-linear terms and the time-derivative terms.

3.3 Implementation: Evaluation of Derivatives

The shape functions are written as the fifth-order polynomials of the local (ξ, η) coordinate (Figure 2.3 in Section 2.3.1). The mapping from the local (ξ, η) coordinate to the global (R, Z) coordinate is defined as (also see Equation 2.44)

$$\left[\begin{array}{c} R \\ Z \end{array} \right] = \left[\begin{array}{cc} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{array} \right] \left[\begin{array}{c} \xi \\ \eta \end{array} \right] + \left[\begin{array}{c} R_0 \\ Z_0 \end{array} \right], \tag{3.27}$$

where (R_0, Z_0) is the origin point of the (ξ, η) coordinate system.

In order to obtain the terms defined by Equation 3.26, the derivatives of the

shape functions up to fourth order need to be evaluated in the global (R, Z) coordinate. It is straightforward to evaluate the derivatives in the local (ξ, η) coordinate, given that the polynomials are written explicitly in terms of ξ and η. The inverse of the matrix in Equation 2.48 gives the transformation from the derivatives with respect to (ξ, η) to the derivatives with respect to (R, Z) up to the second order by replacing θ with $-\theta$. In order to keep the simple notation, define $c = cos(\theta)$ and $s = sin(\theta)$. By applying the chain rule recursively, we get the transformation for the third-order derivatives as

$$\begin{bmatrix} F_{,RRR} \\ F_{,RRZ} \\ F_{,RZZ} \\ F_{,ZZZ} \end{bmatrix} = \begin{bmatrix} c^3 & -3c^2s & 3cs^2 & -s^3 \\ c^2s & c^3 - 2s^2c & s^3 - 2sc^2 & s^2c \\ s^2c & -s^3 + 2sc^2 & c^3 - 2s^2c & -sc^2 \\ s^3 & 3s^2c & 3sc^2 & c^3 \end{bmatrix} \begin{bmatrix} F_{,\xi\xi\xi} \\ F_{,\xi\xi\eta} \\ F_{,\xi\eta\eta} \\ F_{,\eta\eta\eta} \end{bmatrix}, \tag{3.28}$$

and the transformation matrix for the fourth-order derivatives is

$$\begin{bmatrix} c^4 & -4c^3s & 6c^2s^2 & -4cs^3 & s^4 \\ c^3s & c^4 - 3s^2c^2 & -3c^3s + 3cs^3 & 3c^2s^2 - s^4 & -s^3c \\ c^2s^2 & 2sc^3 - 2s^3c & c^4 + s^4 - 4s^2c^2 & 2s^3c - 2sc^3 & s^2c^2 \\ s^3c & -s^4 + 3s^2c^2 & -3s^3c + 3sc^3 & -3s^2c^2 + c^4 & -sc^3 \\ s^4 & 4s^3c & 6s^2c^2 & 4sc^3 & c^4 \end{bmatrix}. \tag{3.29}$$

3.4 Effectiveness of Error Estimator

This section demonstrates the effectiveness of the error estimator defined in Section 3.2.

Define the global error estimator for U and ψ as [37]

$$\epsilon_U^2 = \sum_{\mathcal{K} \in \mathcal{P}} \epsilon_{U\mathcal{K}}^2, \tag{3.30a}$$

$$\epsilon_\psi^2 = \sum_{\mathcal{K} \in \mathcal{P}} \epsilon_{\psi\mathcal{K}}^2. \tag{3.30b}$$

The global effectivity indices are defined as [30]

$$\beta_U = \frac{\epsilon_U}{|||e^{n+1}|||_U}, \tag{3.31a}$$

$$\beta_\psi = \frac{\epsilon_U}{|||E^{n+1}|||_\psi}, \tag{3.31b}$$

where $(e^{n+1}, E^{n+1}) = (U^{n+1} - U_h^{n+1}, \psi^{n+1} - \psi_h^{n+1})$, and $|||\cdot|||_U$ and $|||\cdot|||_\psi$ are the energy norms defined by Equation 3.24.

3.4.1 Steady-state Solution: Hartmann Boundary Layer

The effectiveness of the error estimator is studied on the problem of the Hartmann boundary layers between two planes. It is a steady state of the conductive fluid under low Reynold numbers [41]. Assume that two planes are placed at $Z = -w$ and $Z = w$, there is a background magnetic field along the Z axis with the magnitude as B, and the characteristic magnitude of the velocity is one. The laminar flow through two planes takes the pattern as [41]

$$\mathbf{V} \cdot \hat{R} = 1 - \frac{1}{cos(Ha)} cosh\left(\frac{Ha}{w} Z\right), \tag{3.32a}$$

$$\mathbf{V} \cdot \hat{Z} = 0, \tag{3.32b}$$

where Ha is the Hartmann number defined as

$$Ha = \frac{Bw}{\sqrt{\mu\eta}}. \tag{3.33}$$

The solution of the laminar flow indicates that the boundary layers with the thickness characterized as $\delta = \frac{w}{Ha}$ are developed at $Z = \pm w$.

Given the definition of the U and ψ by Equation 2.17, we seek the analytic solution that takes form as

$$U = Z - \frac{w}{Ha} \frac{1}{cos(Ha)} sinh\left(\frac{Ha}{w} Z\right), \tag{3.34a}$$

$$\psi = C_1 R + C_2 cosh\left(\frac{Ha}{w} Z\right). \tag{3.34b}$$

Note that the steady-state solution of Equation 2.31 satisfies ($\rho \equiv 1$)

$$< \nabla^2 U, U > - < \nabla^2 \psi, \psi > + \mu \nabla^4 U = 0, \tag{3.35a}$$

$$\nabla^2 < \psi, U > + \eta \nabla^4 \psi = 0. \tag{3.35b}$$

Substituting Equation 3.34 into Equation 3.35, we obtain

$$C_1 = B, \tag{3.36a}$$

$$C_2 = \frac{\mu}{cosh(Ha)B}. \tag{3.36b}$$

Substituting C_1 and C_2 into Equation 3.34, we obtain the analytic solution of the steady state as

$$U = Z - \frac{w}{Ha} \frac{1}{cos(Ha)} sinh\left(\frac{Ha}{w} Z\right), \tag{3.37a}$$

$$\psi = BR + \frac{\mu}{cosh(Ha)B} cosh\left(\frac{Ha}{w} Z\right). \tag{3.37b}$$

Figure 3.2 plots the analytic solution defined by Equation 3.37 with two planes placed at $Z = \pm 2$. Ha is set to be 6.3 ($B = 1.0$, $\mu = 1.0$, $\eta = 0.1$) and 20.0 ($B = 1.0$, $\mu = 0.1$, $\eta = 0.1$), respectively. It shows that the current also exhibits the structure of the boundary layers in addition to the velocity profile. Also note that the characteristic magnitude of the velocity defined analytically is minus one.

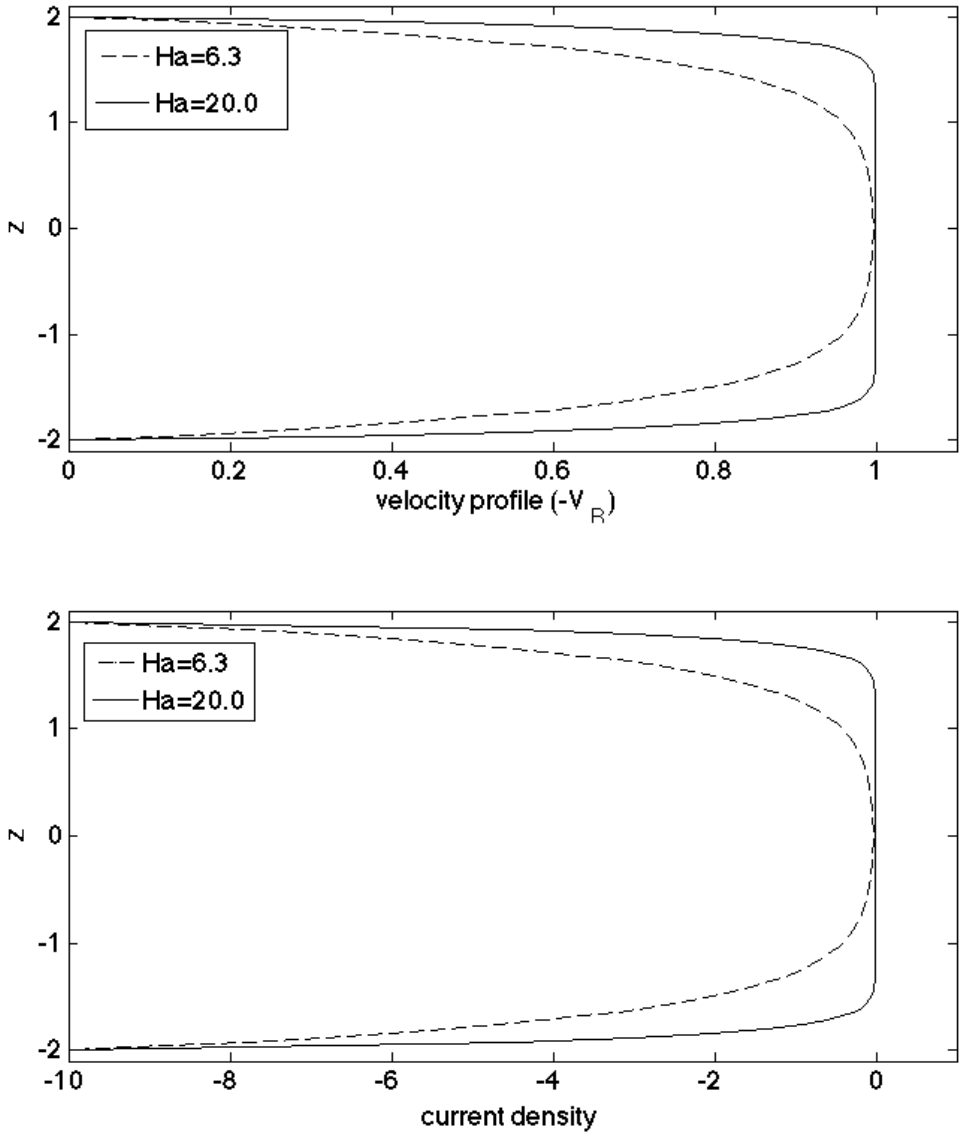


Figure 3.2: Profiles of the Velocity ($\frac{\partial U}{\partial Z} = -\mathbf{V} \cdot \hat{R}$) and current density ($j = -\nabla^2 \psi$) defined by Equation 3.37.

3.4.2 Error Estimation on Hartmann Boundary Layer Problem

In order to obtain the steady-state solution, the Dirichlet boundary conditions are set to satisfy the analytic solution by Equation 3.37 at the domain boundary. The initial condition is set as the perturbed solution of the analytic solution. The simulation domain is a $[-2, 2] \times [-2, 2]$ rectangular. Two sets of tests are performed that use $Ha = 6.3$ ($B = 1.0$, $\mu = 1.0$, $\eta = 0.1$) and $Ha = 20.0$ ($B = 1.0$, $\mu = 0.1$, $\eta = 0.1$), respectively. For each set of the tests, a coarse initial mesh is used. The mesh is adapted when the solution reaches the steady state. The loop continues with calculating the steady state on the adapted mesh until the estimated local error falls below the given criterion. The global effectivity indices are calculated during each iteration.

The energy norm defined by Equation 3.24 depends on the size of time step, δt. Since the solution goes to steady state, the energy norm should be independent of δt. Re-wring Equation 3.6 by dropping the time derivative terms, using the technique that defines the error estimator for the steady incompressible Navier-Stokes equation in [30, 42], and following the procedure in Section 3.2, we obtain the same error estimator as in Equation 3.26 (dropping the δt terms and the superscript n for $(\cdot)^n$ terms), and the energy norm for the steady-state problem is

$$|||e|||_U = \sqrt{\mu a_2(e, e)}, \tag{3.38a}$$

$$|||E|||_\psi = \sqrt{\eta a_2(E, E)}. \tag{3.38b}$$

Figure 3.3 and 3.4 plot the initial and adapted meshes for the cases of $Ha = 6.3$ and $Ha = 20.0$. It can be seen that the mesh is refined at $Z = \pm 2$ where the boundary layers are developed (see Figure 3.2). Figure 3.7 illustrates the steady state solution of ψ on the final adapted mesh.

Table 3.1 and 3.2 calculate the global effectivity indices on the two sets of meshes in Figure 3.3 and Figure 3.4. It can be seen the global effectivity indices for a given problem over estimate the true error. Although the degree of the over estimation is relatively large, they are reasonably steady and the distribution of the element level values is consistent with that of the true error (Figure 3.5 and Fig-

Table 3.1: Global effectivity indices calculated on the meshes in Figure 3.3.

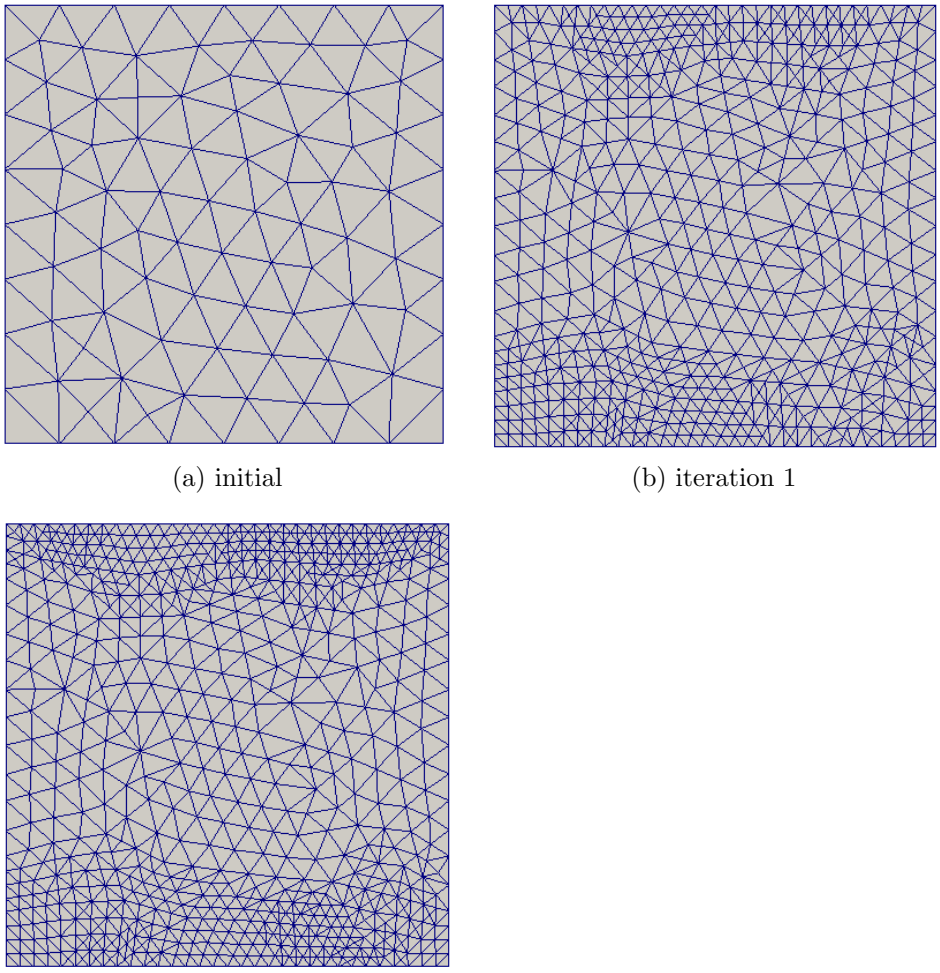
mesh	initial	iteration 1	iteration 2
β_U	25.4	33.7	27.3
β_ψ	10.8	10.4	8.8

Table 3.2: Global effectivity indices calculated on the meshes in Figure 3.4.

mesh	initial	iteration 1	iteration 2	iteration 3	iteration 4
β_U	10.3	7.0	10.5	10.7	10.7
β_ψ	12.1	7.1	10.4	10.7	10.7

ure 3.6). Although additional improvements to this error estimator are possible, the application to the fusion plasma problems of interest (see Chapter 7) would need to be taken into account and it would greatly increase the complexity of the derivation. Consideration of such improvements, along with more explicit consideration of temporal error control, remains an area of future investigation.

Figure 3.8 plots the velocity profile and current density on the initial and final adapted mesh (Figure 3.4) with $Ha = 20.0$. It can be seen the solution on the adapted mesh agrees with the analytic solution.

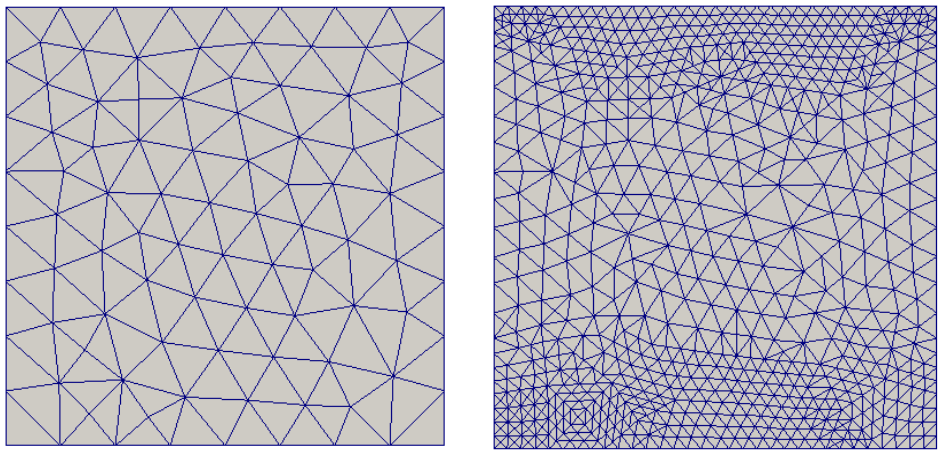


(a) initial

(b) iteration 1

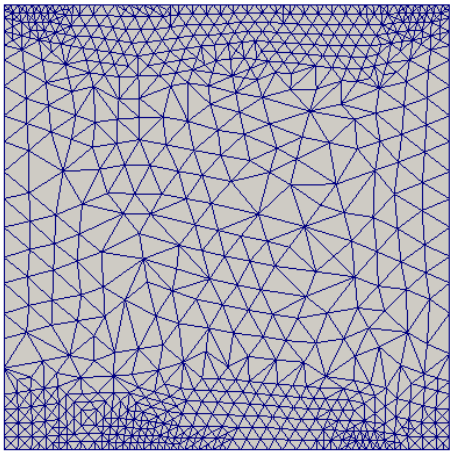
(c) iteration 2

Figure 3.3: Initial and adapted meshes for $Ha = 6.3$.

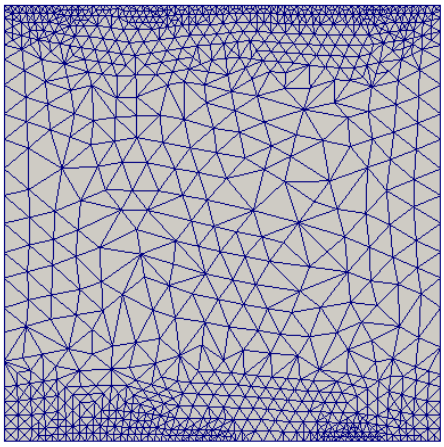


(a) initial

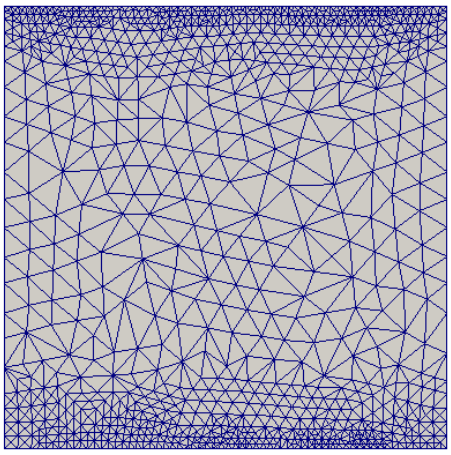
(b) iteration 1



(c) iteration 2

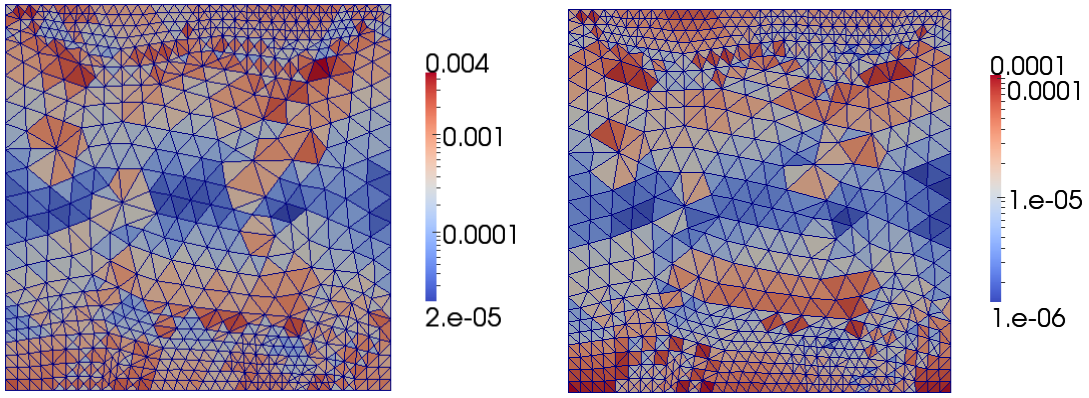


(d) iteration 3

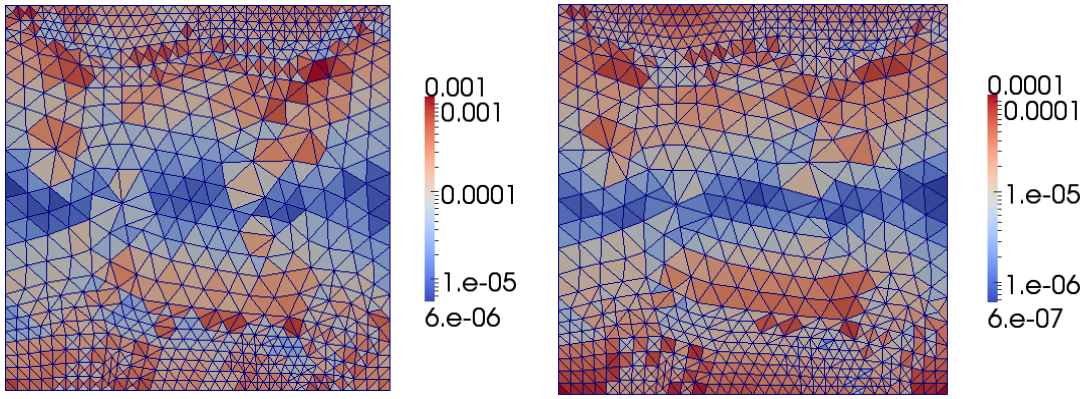


(e) iteration 4

Figure 3.4: Initial and adapted meshes for $Ha = 20.0$.

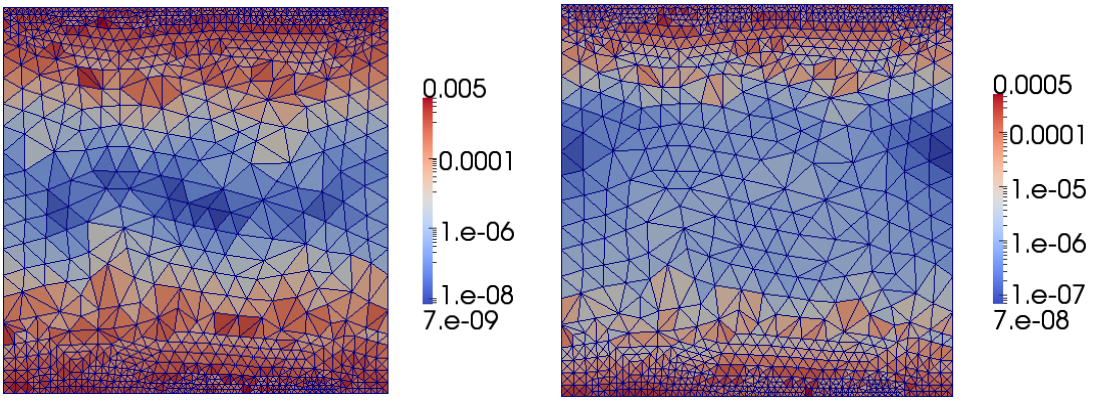


(a) estimated error of U (b) true error of U



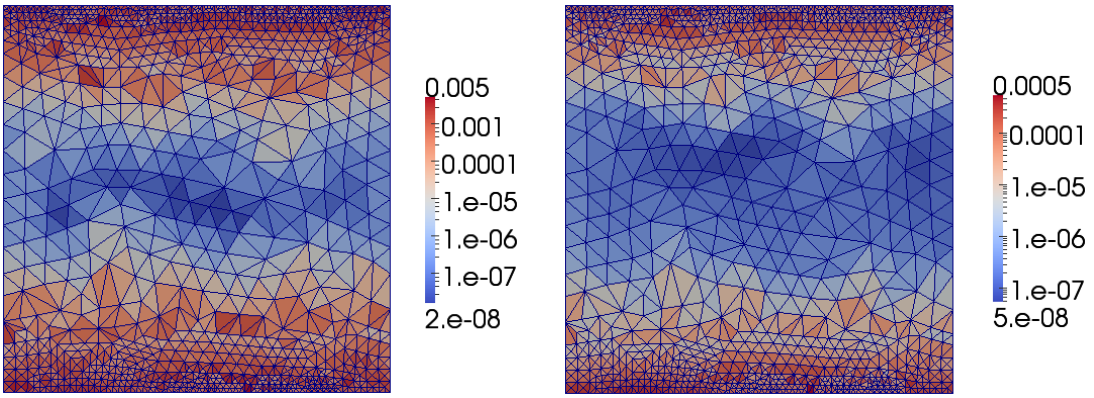
(c) estimated error of ψ (d) true error of ψ

Figure 3.5: Estimated and true element errors on the final adapted mesh in Figure 3.3 ($Ha = 6.3$).



(a) estimated error of U

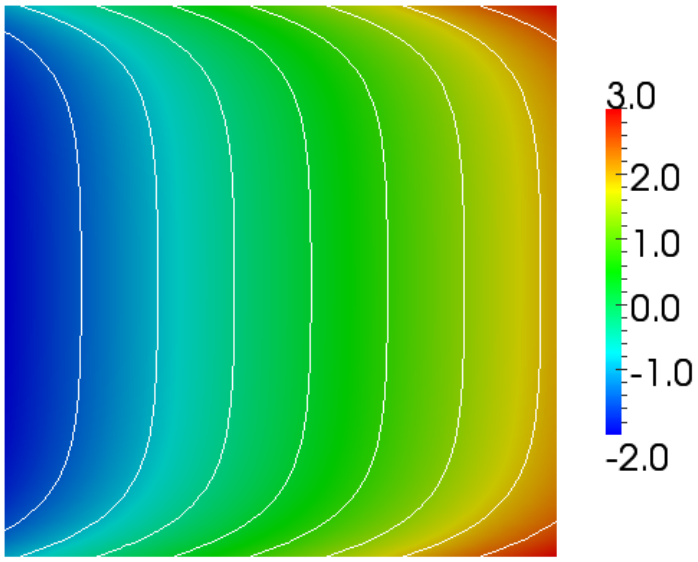
(b) true error of U



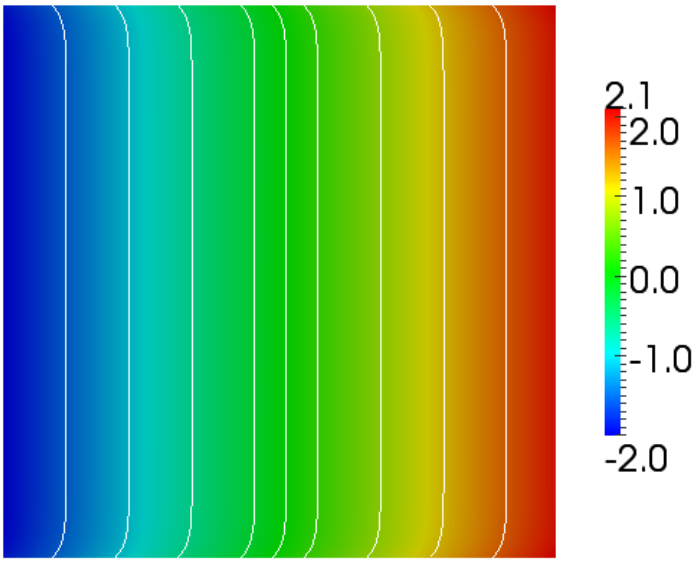
(c) estimated error of ψ

(d) true error of ψ

Figure 3.6: Estimated and true element errors on the final adapted mesh in Figure 3.4 ($Ha = 20.0$).



(a) $Ha = 6.3$



(b) $Ha = 20.0$

Figure 3.7: Steady state solution of ψ on the final adapted mesh in Figure 3.3 and 3.4.

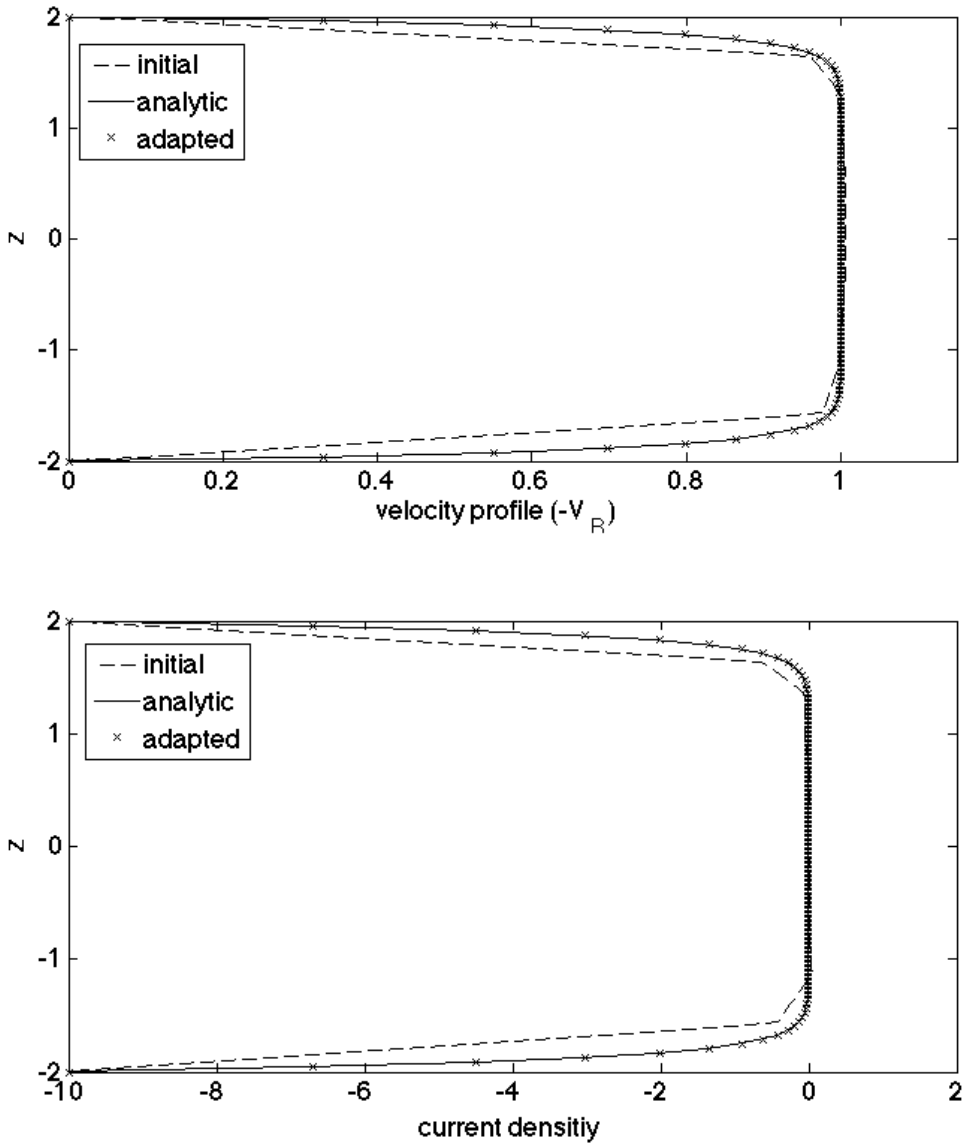


Figure 3.8: Velocity profile and current density on the initial and final adapted mesh at $R = 0$ (Figure 3.4) with $Ha = 20.0$.

CHAPTER 4
M3D-C^1 Simulation Loop and Extensions towards Automated Version

This chapter reviews the parallel adaptive simulation loop in M3D-C^1 and the specific developments made for M3D-C^1. Section 4.1 presents the software tools in the parallel adaptive loop. Section 4.2 discusses the mesh entity and DOF ordering on the partitioned mesh. Section 4.3 gives an overview of the parallel sparse matrix generated by the finite element procedure in M3D-C^1. A matrix assembly procedure based on the mesh adjacency information is discussed in Section 4.4. Section 4.5 introduces the solution mapping method during the mesh modifications.

4.1 M3D-C^1 Simulation Loop

4.1.1 Simulation Loop on 2D Mesh

Figure 4.1 illustrates the workflow of the adaptive loop for M3D-C^1 on a 2D mesh. Specific software tools used are placed within the boxes of the functional components illustrated by Figure 1.1 in Section 1.1. The workflow and the software components include:

- [A in Figure 4.1] The mathematical model defines the governing PDEs of the problem (see Section 2.1), the initial condition and boundary conditions over the problem domain.

- [B in Figure 4.1] The geometric model of the tokamak cross-section is constructed with the boundaries represented by the analytic functions or splines fitted to an ordered set of points. The 2D mesh on the toroidal plane (tokamak cross-section) is generated with the Simmetrix mesh generation tool [43] and partitioned by the graph-based partitioning tool in the Parallel Unstructured Mesh Infrastructure (PUMI) [44]. The geometric model and the distributed mesh are loaded in PUMI that provides the interface to the geometric model

and manages the distributed mesh. For the detailed discussion on the geometric model and mesh definition, see Section 5.4.2.

- [C in Figure 4.1] The fourth-order PDEs derived from the MHD equations are discretized with the C^1 element [3, 8, 6] and the element contributions to the stiffness matrix and the force vector are calculated. The geometric entity that a mesh entity is classified on identifies the set of the governing equation applied in the region with the three region types being plasma, material wall and vacuum. The geometric shape information such as the normal direction and curvature provides the necessary boundary information during the PDE discretization process [2]. The input fields needed to define the element contributions include the equilibrium of the plasmas and/or the solution field from the previous time step. For the detailed discussion on the PDE discretization, refer to Section 2.2 and 2.3.

- [D in Figure 4.1] The element contributions to the stiffness matrix and the force vector are assembled to form the global discrete equation. A global DOF ordering that assigns integer labels to the DOFs associated with the mesh vertex is calculated. The integer corresponds to the equation number in the global discrete system.

- [E in Figure 4.1] The global discrete system is solved to get the solution fields (for instance, the velocity and magnetic fields) at the current time step. Direct solvers such as SuperLU [45] are used to solve problems on the 2D mesh.

- [F in Figure 4.1] A mesh size field is defined either from the priori knowledge of the solution or the posteriori error estimation (see Chapter 3) if the mesh on the cross-section needs to be improved.

- [G in Figure 4.1] The mesh is adapted through MeshAdapt [46, 47]. The fields on the original mesh are mapped onto the new mesh.

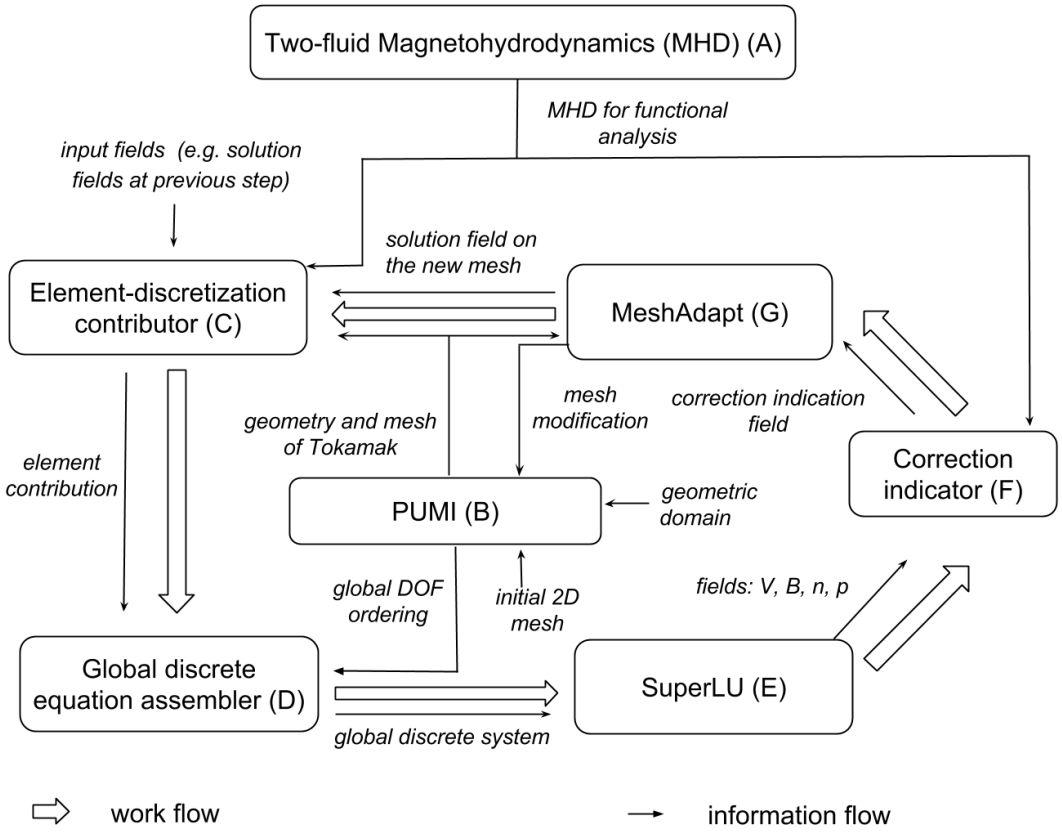


Figure 4.1: Simulation loop of M3D-C^1 on a 2D mesh.

4.1.2 Simulation Loop on 3D Mesh

Figure 4.2 illustrates the workflow of a 3D M3D-C^1 simulation. Similar to the simulation loop on the 2D mesh (Figure 4.1), the specific software tools used are placed within the boxes of the functional components. Note that the components in the dashed boxes (F and G in Figure 4.2) are to be put in place for the 3D simulation loop in the future.

The application of the C^1 wedge element (Section 2.3.3) requires that the 3D mesh in M3D-C^1 is constructed by connecting the matched pairs of the 2D meshes on the planes that correspond to the cross-sections of the torus in the toroidal direction. The 2D mesh is either generated by Simmetrix [43] or adapted from the simulation loop on the 2D mesh (Figure 4.1). The processes used for the 3D simulation are grouped and the number of the process groups is equal to the number of the planes used in the 3D simulation (input information of B in Figure 4.2). Each process

group loads the same distributed 2D mesh, and the wedge elements are created between the mesh parts that correspond to the matched pairs of the toroidal planes (see Section 5.3.4 for the detailed description).

In addition to the mesh setup, iterative solving methods are used to solve the problems on the 3D mesh compared with the simulation loop on the 2D mesh. The global discrete system is solved through PETSc [48] (E in Figure 4.2).

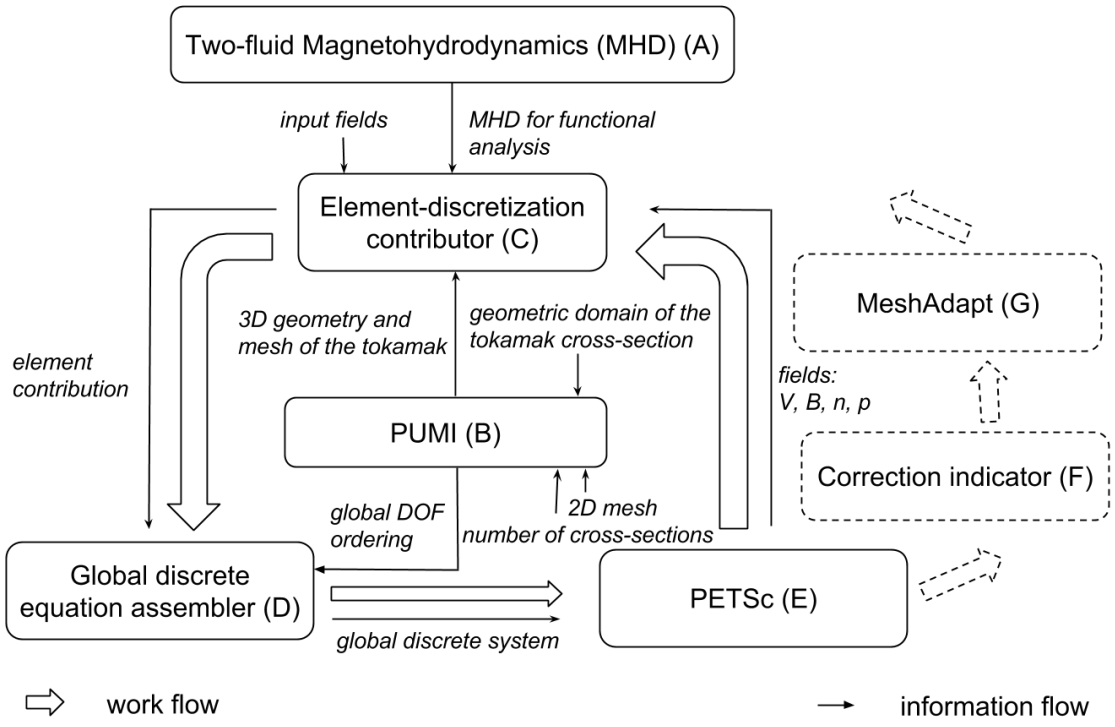


Figure 4.2: Simulation loop of M3D-C^1 on a 3D mesh.

4.1.3 Simulation Loop from 2D Mesh to 3D Mesh

Figure 4.3 illustrates the simulation loop from the 2D mesh to the 3D mesh. The loop starts with the simulation on the 2D mesh and the axis-symmetric instabilities are developed on the tokamak cross-section (C in Figure 4.3). The mesh on the cross-section is adapted by evaluating the solution quality if needed (E in Figure 4.3). If the instabilities in the toroidal direction of the tokamak also need to be developed (F in Figure 4.3), the simulation is switched to the 3D mesh (G in Figure 4.3). See Section 5.3.4 for the detailed description on the 3D mesh construc-

tion from the 2D mesh. The 3D simulation represents a full set of instabilities in all the spatial directions of the tokamak (D in Figure 4.3).

The Attached Parallel Field and Mesh Interface Library (APF) in PUMI [44] is used to store and manage the fields in the adaptive simulation loop (H in Figure 4.3). It provides the array-based field data structure for the mesh-based analysis, the automatic data migration during the dynamic mesh load balancing [44], and the interface to define the field mapping method associated with the mesh modifications.

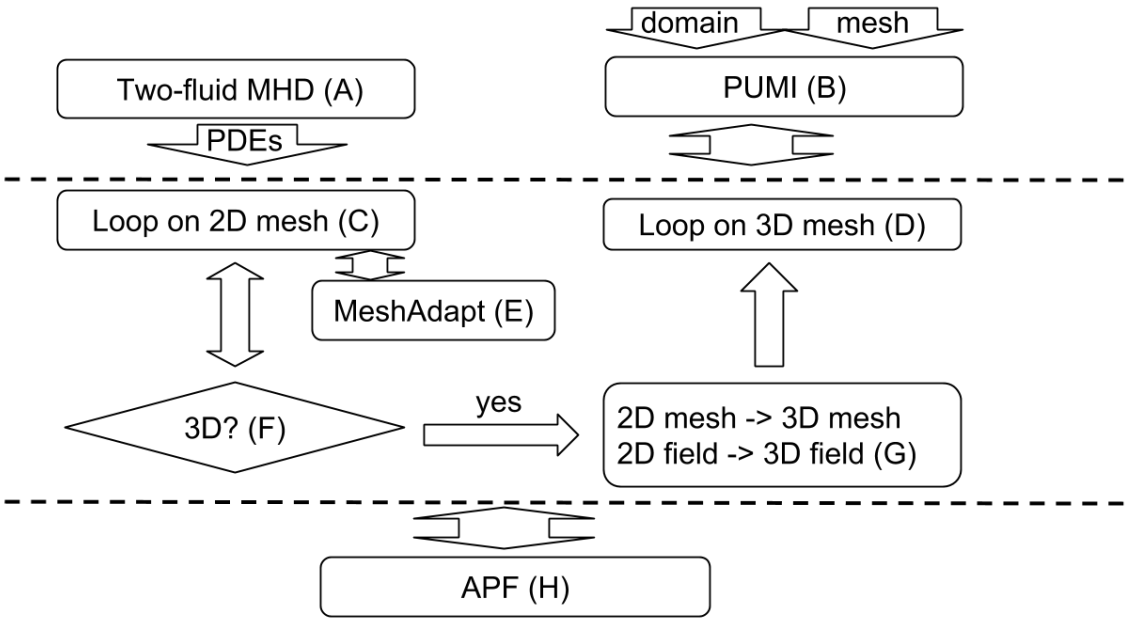


Figure 4.3: Simulation loop from the 2D mesh to the 3D mesh.

4.2 Mesh Entity and DOF Ordering

4.2.1 Mesh Entity Ordering and Ownership on Partitioned Mesh

The mesh entities, specifically the mesh vertices and the mesh elements (the mesh faces in 2D and the mesh regions in 3D) for M3D-C^1, are ordered based on the adjacency information [49] to improve the data locality. Enhanced data locality leads to an increased cache hit rate of the program and thus improve the efficiency of the computation [49].

On the partitioned mesh, there are copies of the same mesh entities at the

mesh part boundary and one copy of the mesh entity is set as the "owner" [44]. The ownership is used to decide the inter-part message passing [50] and to perform the consistent mesh modifications [44] on the partitioned mesh.

For the details on the mesh-adjacency based ordering and the ownership rule on the partitioned mesh, refer to [49, 44].

4.2.2 Global DOF Ordering

In order to assemble the element contribution to the stiffness matrix and the force vector in the global discrete system, the DOFs are labeled with the integers that start with one and increase continuously. The integers give the equation numbers in the global discrete system (D in Figure 4.1 and 4.2).

Given that the DOFs are associated with the mesh vertices in M3D-C^1 (see Section 2.3), the DOFs are labeled from the ordering and the ownership of the mesh vertices (Section 4.2.1). The algorithm that orders the DOFs on the partitioned

mesh is summarized as follows:

```
procedure ORDER_DOF
    var DOFVERTEX = {number of DOFs associated with a vertex}
    var VERTEXSTART={start vertex number}
    var DOFSTART={start DOF number}
    get the number of owned mesh vertices in the mesh part
    get VERTEXSTART by MPI_Reduce
        (VERTEXSTART=1 on the first mesh part)
    DOFSTART = (VERTEXSTART-1) * DOFVERTEX + 1
    for P = 1 to the number of the vertices in the mesh part
        if vertex P is owned by the mesh part
            label DOFs of P from DOFSTART
                 to DOFSTART + DOFVERTEX -1
            DOFSTART = DOFSTART + DOFVERTEX
        end if
    end for
    for each mesh vertex on the part boundary
        if the vertex  is owned by the mesh part
            send the DOF numbers to the non-owner copies
        else
            receive the DOF numbers from the owner copy
        end if
    end for
end procedure
```

4.3 Parallel Sparse Matrix

4.3.1 Matrix Sparsity Pattern

The global DOF ordering on the partitioned mesh discussed in Section 4.2 also provides the row and column numbers to the stiffness matrix in the finite element procedure of M3D-C^1. Since the matrix in the finite element analysis is obtained by assembling the contribution from the individual element, there is a non-zero value

in the matrix at the location specified by the row and the column that correspond to the same mesh vertex or two mesh vertices in the same element. As a result, the sparsity of the matrix is decided by the global DOF ordering and the element connectivity.

For example, given the DOF ordering on the mesh in Figure 4.6, the contributions of element a and element b to the global matrix are

$$\mathbf{K}^a = \begin{bmatrix} k^a_{11} & k^a_{12} & k^a_{13} \\ k^a_{21} & k^a_{22} & k^a_{23} \\ k^a_{31} & k^a_{32} & k^a_{33} \end{bmatrix}, \tag{4.1}$$

and

$$\mathbf{K}^b = \begin{bmatrix} k^b_{11} & k^b_{13} & k^b_{14} \\ k^b_{31} & k^b_{33} & k^b_{34} \\ k^b_{41} & k^b_{43} & k^b_{44} \end{bmatrix}, \tag{4.2}$$

respectively. The resulted global matrix by assembling $\mathbf{K}^a$ and $\mathbf{K}^b$ is

$$\mathbf{K} = \begin{bmatrix} k^a_{11} + k^b_{11} & k^a_{12} & k^a_{13} + k^b_{13} & k^b_{14} \\ k^a_{21} & k^a_{22} & k^a_{23} & \\ k^a_{31} + k^b 31 & k^a_{32} & k^a_{33} + k^b_{33} & k^b_{34} \\ k^b_{41} & & k^b_{43} & k^b_{44} \end{bmatrix}, \tag{4.3}$$

The matrices in M3D-C^1 also exhibit the block structures in a hierarchical way. The C^1 finite elements assign multiple DOFs to each mesh vertex. Each mesh vertex is associated with six DOFs on the 2D mesh or twelve DOFs on the 3D mesh (Section 2.3). In addition to the multiple DOFs at the element level, the vector fields are represented by the multiple scalar components and the number of the scalar components is decided by the M3D-C^1 option. For example, the velocity field in the full MHD model is represented by the three scalar components, (U, ω, χ), in the cylindrical coordinate (R, φ, z) as (also see Equation 2.12 in Section 2.2)

$$\mathbf{V} = R^2 \nabla U \times \nabla \varphi + R^2 \omega \nabla \varphi + \frac{1}{R^2} \nabla_\perp \chi, \tag{4.4}$$

and the one-scalar representation of the velocity field in the reduced MHD model is

$$\mathbf{V} = R^2 \nabla U \times \nabla \varphi. \tag{4.5}$$

The block structure of the matrices by the definition of the finite element and the vector field exist in both the 2D and 3D simulations. In 3D simulation, the construction of the 3D mesh by connecting the meshes on the pair of the adjacent planes leads to the matrix with the cyclic tri-diagonal block structure (Figure 4.4). The diagonal blocks of the 3D matrix (Figure 4.4) are used to form the preconditioning method for the iterative solving process (E in Figure 4.2) [6].

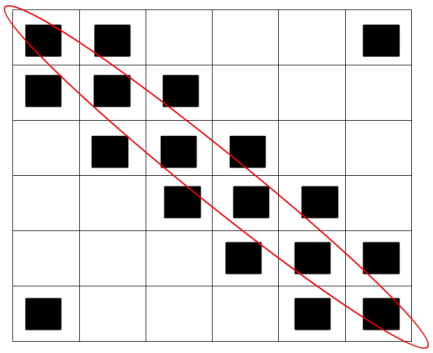


Figure 4.4: 3D Matrix structure with six cross-sections in the torus geometry. Each diagonal block corresponds to a 2D cross-section.

4.3.2 Matrix Partition

The sparse matrix is partitioned to the processes row-wisely in PETSc [51]. Each process "owns" a range of the rows labeled by continuous numbers. The matrix layout in PETSc is illustrated in Figure 4.5. The matrix with the number of rows as n2 is distributed on two processes. The first process owns the rows labeled from 1 to $n1$, and the second process owns the rows labeled from $n1 + 1$ to $n2$.

In M3D-C^1, the ownership of the rows inherits from the ownership of the DOFs that is decided by the mesh vertex ownership (see Section 4.2). Figure 4.6 illustrates the global DOF ordering and the layout of the distribution matrix on a partitioned mesh. Assume the DOF or the vertex labeled by 1 is owned by process 1 and the

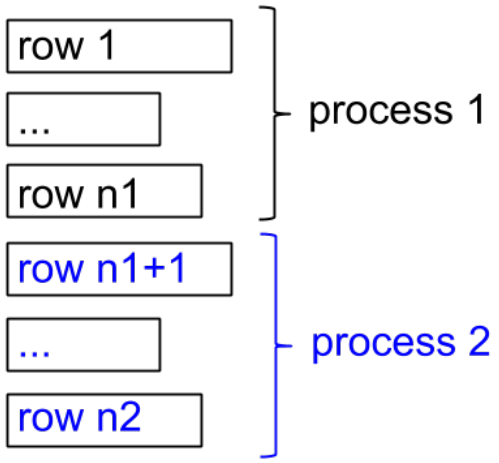


Figure 4.5: PETSc matrix layout.

DOF or the vertex labeled by 3 is owned by process 2. Process 1 owns row 1 and row 2 and process 2 owns row 3 and row 4 in the resulted matrix.

As is discussed in Section 4.3.1, the locations that the element contributions are assembled to the matrix are decided by the global DOF numbers associated with the element. Matrix values in the rows owned by the process are "on-part" values and matrix values in the rows owned by other processes are "off-part" values. For example, element a on process 1 in Figure 4.6 contributes the values of the rows and columns indexed by 1, 2, 3 in the global matrix (see Equation 4.1). The values in row 1 and row 2 are on-part values on process 1. The values in row 3 (k^a_{31}, k^a_{32}, and k^a_{33} in Equation 4.1) are off-part values on process 1.

4.4 Mesh-adjacency Based Matrix Assembly Procedure

As is discussed in Section 4.3.2, there are off-part matrix values generated by the finite element procedure on the partitioned mesh. These off-part values need to be sent to the owner processes to assemble the global matrix. PETSc is used as the matrix library in M3D-C^1. Extra memory is needed to store the off-part values in PETSc, compared with the on-part values during matrix assembly. The curve marked by the circles in Figure 4.7 plots the memory usage to assemble the 3D matrix with the structure illustrated by Figure 4.4. The memory usage is profiled by the valgrind massif tool [52] on a 64 bit machine (piglet at SCOREC) with

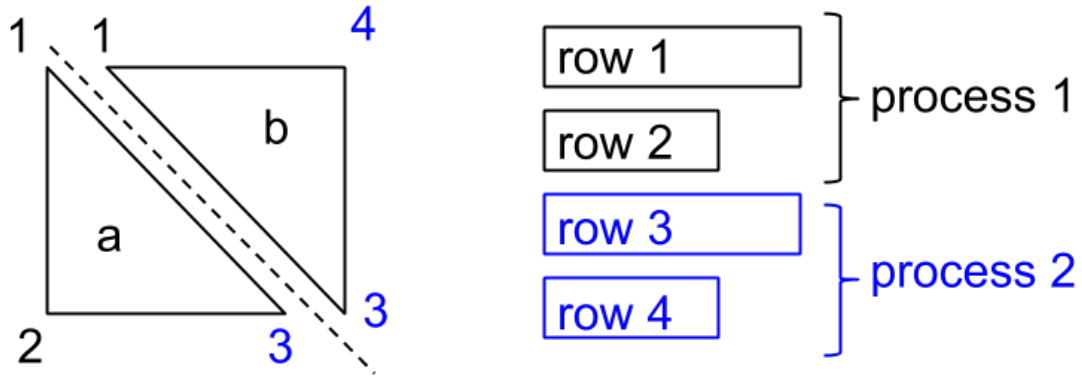


Figure 4.6: Global DOF ordering and the matrix layout on a mesh partitioned and loaded by two processes. Vertex 1 is owned by process 1 and vertex 3 is owned by process 2.

the PETSc version as 3.5.1. The 3D mesh is constructed following the procedure in Section 4.1.2. There are 1034 vertices on the input 2D mesh. Six processes are divided into six groups, and each process loads the same 2D mesh. 3D wedge elements are created between the matched pair of the processes. The contribution of a single wedge element are assembled to the rows of the global matrix that are owned by the two processes (see Figure 2.2). At stage 0-1 in Figure 4.7, the sparsity of the global matrix is set up and the memory needed by the global matrix is preallocated. At stage 1-2, the element contribution is calculated and passed to the matrix structure in PETSc. The additional memory compared with stage 0-1 is to store the off-part values generated by the element procedure. At stage 2-3, the off-part values are passed to the owner processes and the assembled global matrix is obtained at stage 3-4. The additional memory used to store and pass the off-part values (see the peak memory at stage 2-3) is about seven times of the memory needed by the assembled global matrix (see the memory usage at stage 0-1 or stage 3-4) for the 3D matrix studied.

In order to reduce the memory footprint during the matrix assembly, a procedure that takes advantage of the mesh-adjacency information is developed to perform the parallel matrix assembly. The off-part values generated by the element procedure are not directly assembled to the matrix that is to form the global discrete system. Instead, an auxiliary on-part matrix is created to store these off-part values. The

values in the auxiliary matrix are sent to the owner processes by the point-to-point communication pattern on the partitioned mesh [50, 53] and assembled on the owner processes to the matrix in the global discrete system.

The following data structures used by the matrix assembly procedure are defined:

- `NVERTEX`: the number of the vertices in a mesh element (equal to 3 for the 2D triangle element and 6 for the 3D wedge element),
- `DOFVERTEX`: the number of DOFs associated with a mesh vertex,
- `ELMMAT`: the element matrix ($\mathbf{K}^a$ in Equation 4.1 or $\mathbf{K}^b$ in Equation 4.2 if `DOFVERTEX` equals one),
- `PARAMAT`: the parallel matrix partitioned row-wisely in the global discrete system ($\mathbf{K}$ in Equation 4.3),
- `SEQMAT`: the auxiliary on-part matrix to store the element contributions that are off-part with respect to `PARAMAT`,
- `GLBIDX`: the array of the global DOF numbers attached to a vertex (see Section 4.2),
- `LOCIDX`: the array of indices corresponding to the locations of the DOF values in the array-based field data structure (local DOF numbers) [44].

The details of the procedure is discussed by the following sub-sections.

4.4.1 Preallocation Stage

The *preallocation* stage calculates the number of the non-zero values in each row of the matrix and provides the information to the matrix library. Preallocation is performed for both the parallel matrix in the global discrete system, `PARAMAT` and the auxiliary on-part matrix, `SEQMAT`. The algorithm is summarized as the follows:

```
procedure PREALLOCATE_MATRIX
    for each mesh vertex in the local mesh part
        if the vertex is owned by other processes
            get the number of the adjacent vertices
                in the local mesh part as NUMADJ
            set the number of non-zero values in the rows of SEQMAT
                associated with the vertex to be (1+NUMADJ)*DOFVERTEX
        else
            get the number of the adjacent vertices
                in the global complete mesh as NUMADJ
            set the number of non-zero values in the rows of PARAMAT
                associated with the vertex to be (1+NUMADJ)*DOFVERTEX
        end if
    end for
end procedure
```

4.4.2 Matrix Set-up Stage

During the matrix set-up stage, the element contribution is calculated and assembled to `PARAMAT` or `SEQMAT`, depending on whether the vertex in the element is owned by the process. Note that the block structure of the matrix due to the multiple DOFs at the element level (Section 4.3.2) is used to assemble the element contribution. A block of the matrix values instead of a single matrix value is assem-

bled to the corresponding matrix at a time.

```
procedure SETUP_MATRIX
    for each element in the local mesh part
        calculate ELMMAT
        for each vertex in the element
            if the vertex is owned by the process
                get GLBIDX of the vertex
                assemble the rows of ELMMAT
                    associated with the vertex to PARAMAT by GLBIDX
            else
                get LOCIDX of the vertex
                assemble the rows of ELMMAT
                    associated with the vertex to SEQMAT by LOCIDX
            end if
        end for
    end for
end procedure
```

4.4.3 Parallel Assembly Stage

During parallel assembly stage, matrix values stored in the auxiliary matrix, `SEQMAT`, are sent to the owner processes and assembled to the global matrix, `PARAMAT`, on the owner processes. The algorithm is described as follows [50, 53]:

```
procedure ASSEMBLE_PARALLEL_MATRIX
    for each vertex in the local mesh part
        if the vertex is owned by other processes
            get the list of the adjacent vertices
                on the local mesh part as VERTEXADJ
            add the vertex to VERTEXADJ
            get global DOF numbers of VERTEXADJ in GLBIDX
            get local DOF numbers of VERTEXADJ in LOCIDX
            get the rows in SEQMAT by LOCIDX
            pack GLBIDX and the rows into data buffer
                to be sent to the owner process
        end if
    end for
    destroy SEQMAT
    send the data to the owner processes
    receive the data from non-owner processes
    assemble the rows received to PARAMAT
end procedure
```

Note that the block structure of the matrix (Section 4.3.2) is further used to reduce the data size during parallel communication. The indices of the blocks are sent instead of the indices of individual values (`GLBIDX` in the algorithm).

4.4.4 Memory Usage Improvement

Figure 4.7 compares the memory usage of the 3D matrix assembly by the mesh-adjacency based procedure and the procedure using PETSc directly. At stage 0-1, the memory needed to store the matrix/matrices is preallocated. The parallel assembly by PETSc only preallocates the amount of the memory for the rows owned

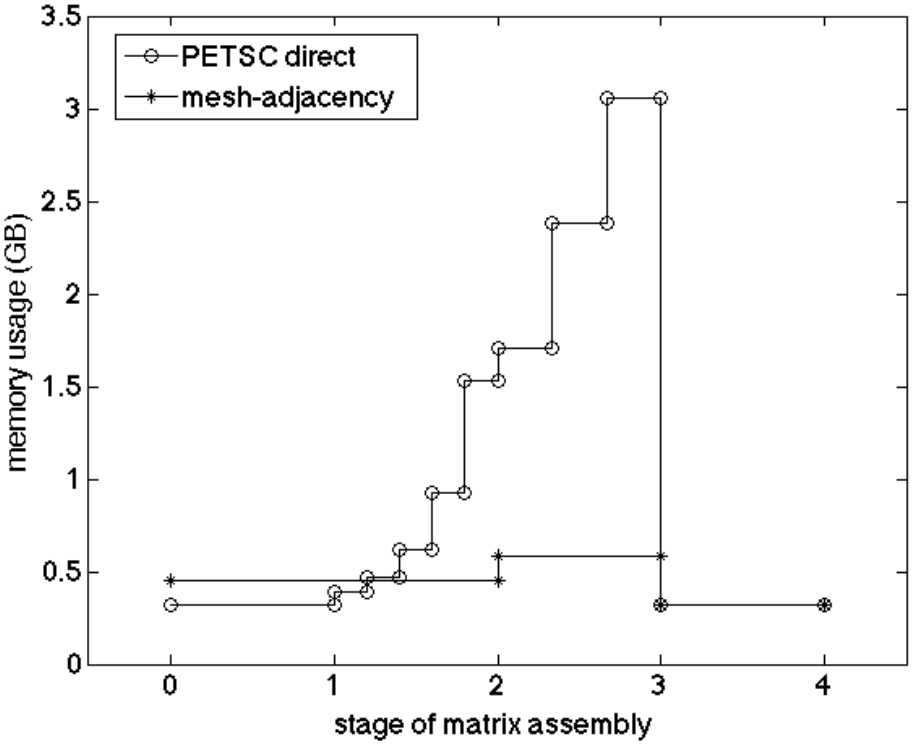


Figure 4.7: Memory usage by using PETSc and the mesh-adjacency based procedure. Each mesh vertex is attached with 36 DOFs. On each plane, there are 1034 mesh vertices and 2.7×10^7 non-zeros values in the assembled matrix.

by the calling process. The mesh-adjacency based parallel assembly also preallocates the auxiliary on-part matrix (see the additional amount of memory by the mesh-adjacency based parallel assembly at stage 0-1). At stage 1-2, the element matrix values are generated and passed to the matrix structure. The parallel assembly by PETSc stores off-part values in a separate data structure (referred as "stash" [51]) and new memory for the off-part values is allocated incrementally as the element procedure goes on. The mesh-adjacency based parallel assembly stores the off-part values to the auxiliary on-part matrix of which the memory is preallocated at stage 0-1. At stage 2-3, off-part matrix values owned by the other processes are sent. An additional data buffer is allocated to send/receive the values for both methods. At stage 3-4, the final assembled global parallel matrix is obtained. Both methods use the same amount of memory at this stage. Also notice that the storage format of

Table 4.1: Running time of the parallel matrix assembly by using PETSc directly and by using the mesh-adjacency based procedure. There are 13,113 vertices on each plane and 36 DOFs associated with each vertex. The number of the processes in each process group is 312.

number of planes (process groups)	6	12	24	48
number of processes	1,872	3,744	7,488	14,976
number of DOFs	2.8E6	5.7E6	11.3E6	22.7E6
number of non-zero values	2.1E9	4.2E9	8.5E9	17.0E9
mesh-adjacency-assembly time	0.41s	0.43 s	0.44 s	0.45 s
PETSc-assembly time	0.84 s	0.88 s	0.96 s	0.97 s

the auxiliary matrix (`SEQMAT`) and the data buffer for the message passing at stage 2-3 in the mesh-adjacency based assembly take advantage of the block structure of the sparse matrix. Only the indices of the blocks are stored instead of the individual values. As for the matrix to form the global discrete system (`PARAMAT`), the storage format does not use the block structure due to the limitation of the current PETSc implementation (see [48] for further information).

It shows that the peak memory is reduced to $\sim 25\%$ by the mesh-adjacency based parallel matrix assembly procedure. The matrix assembly procedure that uses the mesh-adjacency information improves the efficiency of the matrix assembly in M3D-C^1 regarding to the memory cost.

Table 4.1 compares the running time to perform the parallel matrix assembly (Section 4.4.3) on NERSC Edison [54]. It can be seen that the running time by the mesh-adjacency based procedure is reduced by more than 50% compared with the procedure using PETSc directly due to the smaller size of the inter-process message (see stage 2-3 in Figure 4.7). The mesh-adjacency based assembly procedure also shows scalability on the weak scaling study in Table 4.1 by applying the point-to-point communication pattern [50, 53].

4.5 Solution Mapping during Mesh Modification

4.5.1 Solution Mapping from 2D Mesh to 3D Mesh

When the simulation loop is switched from the 2D to the full 3D (Figure 4.3), the solution fields on the 2D mesh are mapped to the 3D mesh. Assume U is an axis-symmetric field defined on the 2D mesh. To map U from the 2D mesh to the 3D mesh, the DOFs of the C^1 wedge elements are specified (see Section 2.3.3). For each vertex, the number of DOFs needed by U is changed from six to twelve. The DOFs corresponding to U, $U_{,R}$, $U_{,Z}$, $U_{,RR}$, $U_{,RZ}$, $U_{,ZZ}$ on the 3D mesh are assigned with the same values as on the original 2D mesh. The DOFs corresponding to $U_{,\varphi}$, $U_{,R\varphi}$, $U_{,Z\varphi}$, $U_{,RR\varphi}$, $U_{,RZ\varphi}$, $U_{,ZZ\varphi}$ are assigned to zero since the field on the 3D mesh is still axis-symmetric and there is no variation in the φ direction ($\partial U/\partial\varphi = 0$).

4.5.2 Solution Transfer with Local Modification

In the adaptive loop illustrated by Figure 4.1, the solutions are mapped to the new mesh when the mesh is adapted by applying the local modifications. We focus on the case that a edge is split on the toroidal 2D mesh in M3D-C^1. Figure 4.8 illustrates the edge split operation during the local mesh modifications [46]. The edge split operation creates a new vertex, P. The DOF values associated with the new vertex need to be specified. Assume the original finite element solution field on elements J and K is

$$U^J(\xi,\eta) = \sum_{i=1}^{18} \lambda_i^J \mu_i(\xi,\eta), \tag{4.6a}$$

$$U^K(\xi,\eta) = \sum_{i=1}^{18} \lambda_i^K \mu_i(\xi,\eta), \tag{4.6b}$$

respectively, where μ_i is the i^{th} reduced quintic shape function (Section 2.3.1), and λ_i is the multiplier of μ_i calculated from the DOFs associated with the triangle by Equation 2.48.

The coordinate of $P(R,Z)$ is converted to the local coordinate in elements J and K by Equation 2.44 as $P(\xi^J,\eta^J)$, and $P(\xi^K,\eta^K)$, respectively. The function value, first and second order derivatives of the field in Equation 4.6 are evaluated at

P and converted to the DOF values by the inverse mapping defined by Equation 2.48 as $\{d_i^J\}$ and $\{d_i^K\}$, respectively.

The DOFs associated with P are specified as

$$d_i = \frac{S^J}{S^K + S^J} d_i^J + \frac{S^K}{S^K + S^J} d_i^K, \tag{4.7}$$

where $i = 1, 2, ..., 6$, and S^J and S^K are areas of the elements J and K, respectively.

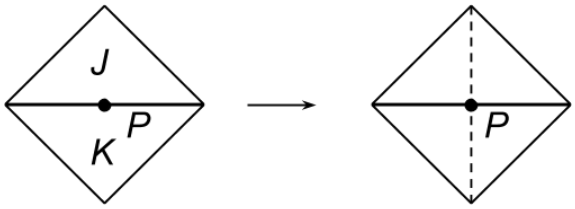


Figure 4.8: Edge split.

Given the property of the C^1 field, it can be seen:

- DOFs corresponding to the function value and the first-order derivatives are interpolated from the original field directly.
- DOFs corresponding to the second-order derivatives are obtained by averaging the values evaluated at the new vertex in the elements adjacent to the edge to be split.

Solution transfer associated with the other types of local modifications, such as the swap and the collapse operations, is performed either by keeping the DOF values associated with the mesh vertices unchanged (swap operation) or dropping the DOFs along with the mesh vertex to be collapsed. More sophisticated solution transfer methods such as the superconvergent patch recovery (SPR) [55] is to be investigated in the future.

CHAPTER 5
Mesh Generation and Adaptation for Confined Fusion Plasma Simulation

5.1 Introduction

Mesh-based methods are extensively applied to study the behaviors of the plasmas in tokamak geometries [11, 2, 1]. The challenges of meshing for fusion reactor simulations lie in satisfying the specific requirements from the combination of physics of interest and the computational methods applied to model the physics. A fully automatic meshing procedure is needed to most effectively meet specific constraints from different fusion plasma simulation codes.

XGC1 [56] applies the particle-in-cell method (PIC) to solve the gyrokinetic Vlasov-Maxwell system and it focuses on the physics phenomena at the plasma edge. The motion of particles is tracked and the fields that provide the driving force are computed on the mesh [11, 57, 13]. In case of meshes for XGC1, the magnetic flux surfaces form a set of curves on the 2D cross-section. Based on the combination of the complex physics and numerical methods used in XGC1 [11, 57, 13], the meshing requirements include: *(i)* mesh edges must align with the magnetic flux surfaces, *(ii)* mesh vertices on the magnetic flux surfaces must be placed in a specific manner [13] to follow the motion of particles through the magnetic field, *(iii)* the mesh should be one-element deep between adjacent magnetic flux surfaces, *(iv)* the layered mesh between surfaces needs to be improved at particular areas such as the X-point [4, 58], and *(v)* the mesh should be generated on the real geometries of fusion reactors.

M3D-C^1 [3, 7, 8, 9, 6, 10, 59] studies non-linear magnetohydrodynamic (MHD) instabilities of the plasmas in the tokamak. C^1 finite elements are applied to solve the fourth order PDEs that arise when a stream function/potential representation

---

This chapter is in Press: F. Zhang *et al.*, "Mesh generation for confined fusion plasma simulation," *Eng. with Comput.*

for the velocity and magnetic potential vector fields are combined with the MHD equations. The meshing requirements are *(i)* initial 2D mesh generation on the toroidal cross-section geometry, *(ii)* mesh adaptation on the toroidal cross-section, and *(iii)* 3D geometric model and distributed mesh construction out of multiple 2D models and meshes. An initial unstructured mesh on the cross-section is generated with the controlled mesh size, and then improved by anisotropic mesh adaptation. The 3D mesh with wedge elements is created by connecting triangular mesh faces on 2D cross-sections.

This paper discusses a set of procedures to meet the particular meshing requirements of the XGC1 and M3D-C^1 plasma physics codes as follows: Section 5.2 describes the geometry definition in the fusion plasma codes. Section 5.3 discusses software design and algorithms applied to generate meshes in a controlled manner. Section 5.4 and Section 5.5 present resulting meshes and the closing remarks, respectively.

5.2 Geometric Model Definition

In a geometry-based analysis environment, an effective representation of the analysis domain is a non-manifold boundary topology with associated shape information [60]. A geometric model boundary representation is a hierarchy of regions, shells, faces, loops, edges, vertices, (Figure 5.1) and use entities for vertices, edges, loops, and faces [61] that can effectively define the adjacencies seen in analysis model idealizations of physical domains. The geometric model is a necessary input for the reliable automatic meshing [62].

The mesh is a discretized representation of the geometric model. It consists of the four types of topological entities, which are regions (3D), faces (2D), edges (1D) and vertices (0D), with controlled size, shape, and distribution [63, 64, 65]. A mesh that does not bound any higher dimensional entities is termed an *element* with respect to the simulation.

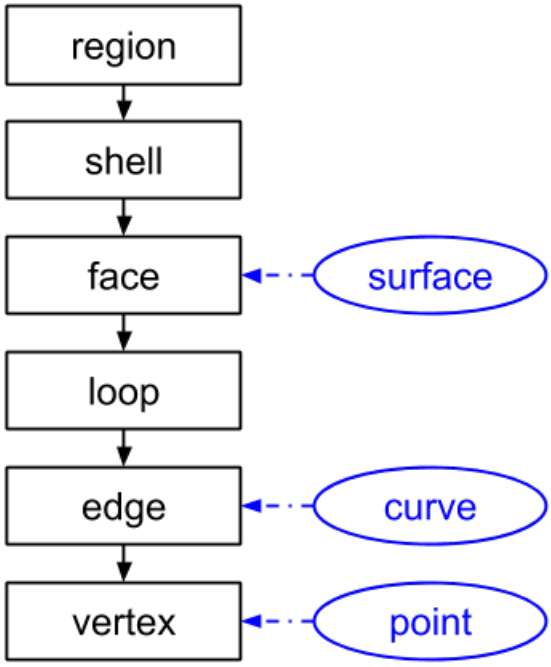


Figure 5.1: Topological entities (rectangles) and associated shape information (ellipses) in the geometric model.

5.2.1 Geometric Description

Tokamak devices use a magnetic field to confine the plasmas for sustainable fusion reactions. The basic tokamak geometry is a torus that is symmetric along the toroidal direction. Magnetic field lines wind around the torus and form magnetic flux surfaces [4]. The meshing procedure first generates the 2D geometric model and then generates a 2D mesh based on the geometry.

Figure 5.2 illustrates the 2D geometric model of the toroidal cross-section. In order to comply with different meshing requirements of the plasma simulation codes, the geometry of the tokamak contains the combination of the physics and physical components. The physics and physical components are depicted on the left and right of Figure 5.2, respectively.

A separatrix (3 in Figure 5.2) adjacent to the magnetic axis (0 in Figure 5.2) splits the face into two distinct areas, the scrape-off layer (4 in Figure 5.2) and the plasma core (5 in Figure 5.2). The scrape-off layer is the area between the separatrix and the wall boundary. In axisymmetric equilibria of the plasmas, the

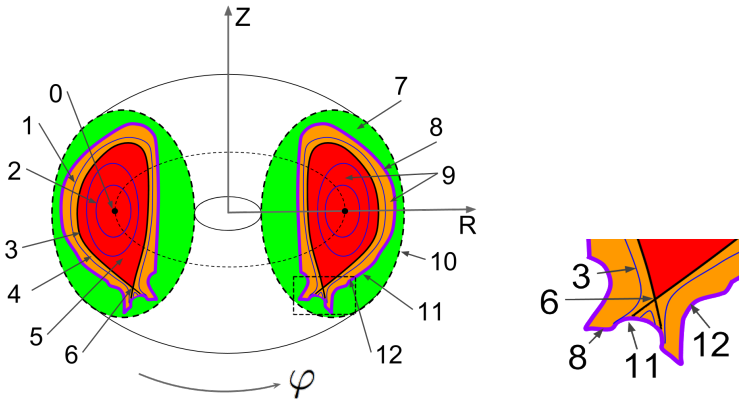


Figure 5.2: Geometric components of the fusion reactor [4]. The coordinate system is (R, Z, φ) or (r, θ, φ), where R, r, φ and θ are major radius, minor radius, toroidal angle and poloidal angle, respectively. The model components include the (0) magnetic axis, (1) open magnetic flux surfaces, (2) closed magnetic flux surfaces, (3) separatrices, (4) scrape-off layer, (5) plasma core, (6) X-points, (7) vacuum vessel, (8) wall area, (9) plasma area, (10) vacuum boundary, (11) outer wall boundary, and (12) inner wall boundary.

magnetic flux surfaces (2 and 3 in Figure 5.2) in the plasma core are nested toroids, that form non-intersecting loops on the 2-dimensional toroidal cross-section between the magnetic axis and the separatrix. The magnetic surfaces diverge at the X-point (6 in Figure 5.2) that is a saddle point of the magnetic flux field on the separatrix. The flux surfaces intersect the wall in the area of the scrape-off layer. In addition to the plasma area, the geometry can include a vacuum vessel (7 in Figure 5.2) and a finite thickness wall (8 in Figure 5.2). The vacuum vessel is the outermost area of the fusion device surrounding the wall. The plasma area (9 in Figure 5.2) is the interior area bounded by the limiter or the boundary of the material wall [4]. There may exist one or more separatrices in the plasma area.

The geometry of XGC1 contains the plasma area with the scrape-off layer (4 in Figure 5.2) separated by the separatrix (3 in Figure 5.2) and the plasma core (5 in Figure 5.2). It also includes magnetic flux surfaces (1 and 2 in Figure 5.2) in the plasma area and the inner wall boundary to meet the requirement of placing mesh vertices. Multiple identical planes are placed around the torus axis.

The geometry of M3D-C^1 is a 3D torus made up of the vacuum vessel (7 in

Figure 5.2), the material wall with or without a finite thickness, δ_h, (8 in Figure 5.2), and the plasma area (9 in Figure 5.2). Each plane forms a cross-section of the tokamak, so multiple planes are placed around the torus axis, each with an identical mesh as is also the case with XGC1. The outer boundary of the vacuum vessel is a simple loop (10 in Figure 5.2) that encloses the wall area.

5.2.2 Topological Representation

Figure 5.3 illustrates the topological structure of model faces in XGC1 geometry. Faces interior to the separatrix are bounded by two loops that correspond to a pair of adjacent closed magnetic flux surfaces (faces B and E) or a pair defined by the magnetic axis and the adjacent closed magnetic flux surface (face A). Faces exterior to the separatrix are bounded by one loop that corresponds to a collection of open magnetic flux surfaces and a portion of the wall boundary (faces D, F, G and H).

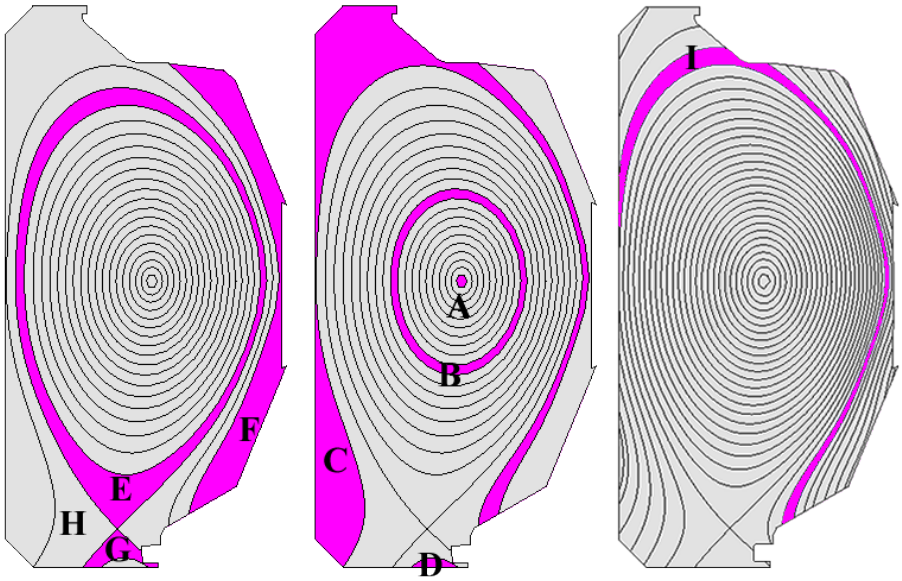
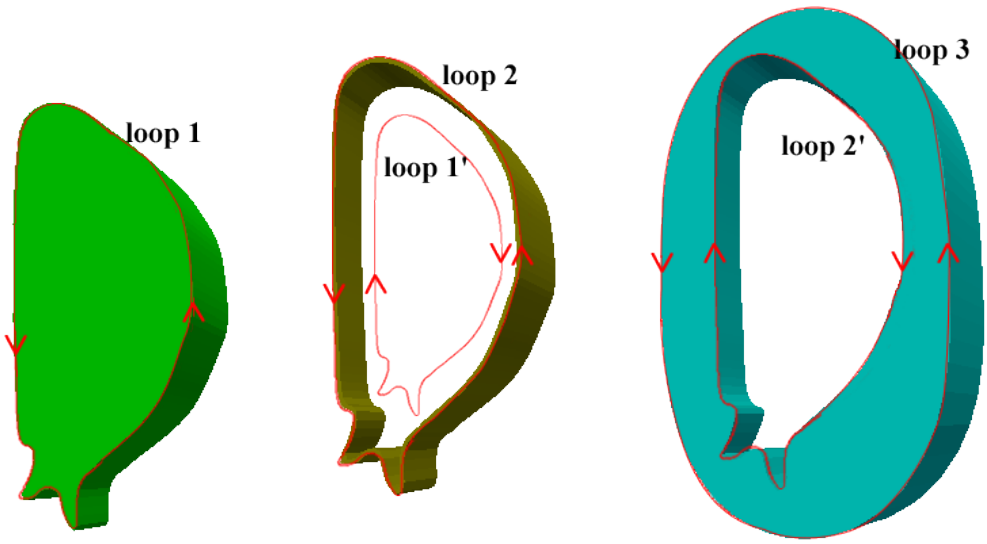


Figure 5.3: Geometric faces on the toroidal cross-section in XGC1.

The geometry of M3D-C^1 is a torus composed of multiple wedge segments separated by the planes placed around the major axis of the torus. There are up to three model regions between two adjacent planes, which represent the plasma (Figure 5.4a), wall (Figure 5.4b) and vacuum (Figure 5.4c) areas. Note that the

simulation domain may consist of the plasma area only or the three areas depending on the input options of M3D-C^1. Each model region is bounded by a shell that consists of two faces on the two planes and faces joining two planes. There are up to three model faces on any given plane or cross-section, which are bounded by the loops on the boundary of the material wall and the vacuum (Figure 5.4).



(a) plasma face (b) wall face (c) vacuum face

Figure 5.4: Geometric faces on the plane (loop 1' of the wall face is offset to be distinct from loop 2).

5.2.3 Shape Definition

On the toroidal cross-section, the key shape information is the geometry of curves. There are two kinds of curves in the geometric model. The first are physical domain curves that define the inner/outer walls of the reactor and the vacuum boundary. The second are physics curves that define the features interior to the reactor. A physics curve corresponds to a specific magnetic flux surface.

5.2.3.1 Physical Curves

The geometry of the reactor wall curves is defined by either CAD model input, analytic functions, or splines fitted to an ordered set of points. Figure 5.5 illustrates

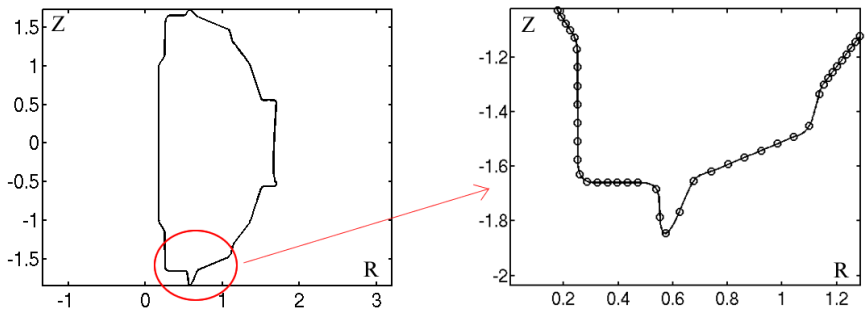


Figure 5.5: Wall curve of NSTX [5] by the cubic spline interpolation with C^2 continuity.

a wall curve with C^2 continuity by interpolating an ordered set of points with the cubic B-splines [66] in M3D-C^1.

5.2.3.2 Physics Curves

The intersection of the magnetic flux surfaces and a plane with a constant φ value form the physics curves, which are flux surfaces composed of common magnetic field lines. Magnetic fields in axisymmetric equilibria can be described by the poloidal magnetic flux field, $\psi = \psi(R, Z)$, and the field related to the poloidal current density, $I(\psi)$ [4]. Given $\psi = \psi(R, Z)$ and $I(\psi)$, the magnetic field $\mathbf{B}$ is defined as [4]

$$\mathbf{B} = -\frac{1}{R}\frac{\partial \psi}{\partial Z}\hat{R} + \frac{1}{R}\frac{\partial \psi}{\partial R}\hat{Z} + \frac{I(\psi)}{R}\hat{\varphi} \tag{5.1}$$

The value of ψ does not change along the direction of $\mathbf{B}$ from the definition. Therefore, each magnetic flux surface associates with a constant ψ.

The 3D field lines along the constant magnetic flux surfaces are defined in the parametric form as $\mathbf{L}(t) = [L_R(t), L_Z(t), L_\varphi(t)]$. Given a set of the constant magnetic flux surfaces, $\{\psi_i\}$, the field lines are as follows:

$$\begin{aligned} \frac{dL_R}{dt} &= \frac{RB_R}{B_\varphi} = -\frac{\partial \psi}{\partial Z}\frac{R}{I(\psi_i)} \\ \frac{dL_Z}{dt} &= \frac{RB_Z}{B_\varphi} = \frac{\partial \psi}{\partial R}\frac{R}{I(\psi_i)} \\ \frac{dL_\varphi}{dt} &= 1 \end{aligned} \tag{5.2}$$

Equation 5.2 describes how the poloidal component of **B** changes according to a unit change of φ. Figure 5.6 illustrates the 3D field line on a closed magnetic flux surface.

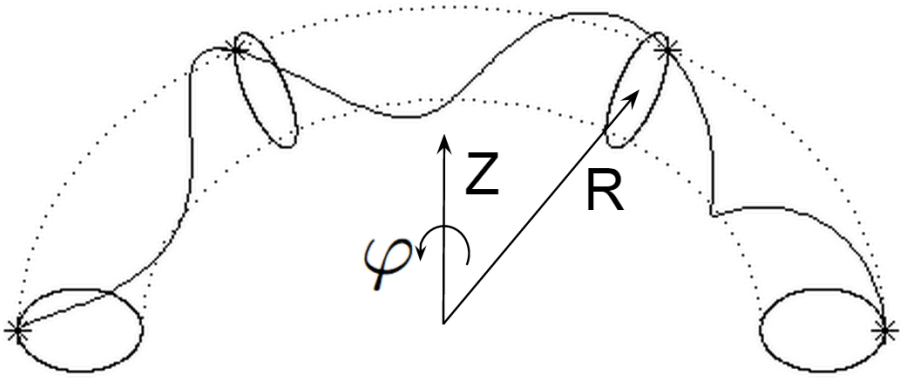


Figure 5.6: Magnetic field line on a closed magnetic flux surface.

The ordinary differential equation defined by Equation 5.2 is integrated by the Runge-Kutta method on a uniform grid. The two flux fields, $\psi(R, Z)$ and $I(\psi)$, are defined by fitting the experimental data with splines [67, 68].

Given the definition of magnetic field line in Equation 5.2, the physics curves are obtained by replacing $L_\varphi(t)$ with a constant value.

5.2.4 Geometric Model Construction

Figure 5.7 illustrates two basic topological splits that are applied iteratively to construct the geometric model on the toroidal cross-section. A model face is either split by the model edges that connect two model vertices on the boundary of the model face or by the model edges that form a loop interior to the model face.

If the model face is to be split by a list of model edges associated with a magnetic flux surface, the value of ψ is specified to create the curves associated with the model edges. The value of ψ between adjacent magnetic flux surfaces is changed by $\delta\psi$ that is specified by the user. The exceptional case is at the magnetic axis point where $\frac{d\psi}{dr} \sim 0$. The new value of ψ near the magnetic axis point is determined based on the physical distance between the adjacent flux surfaces.

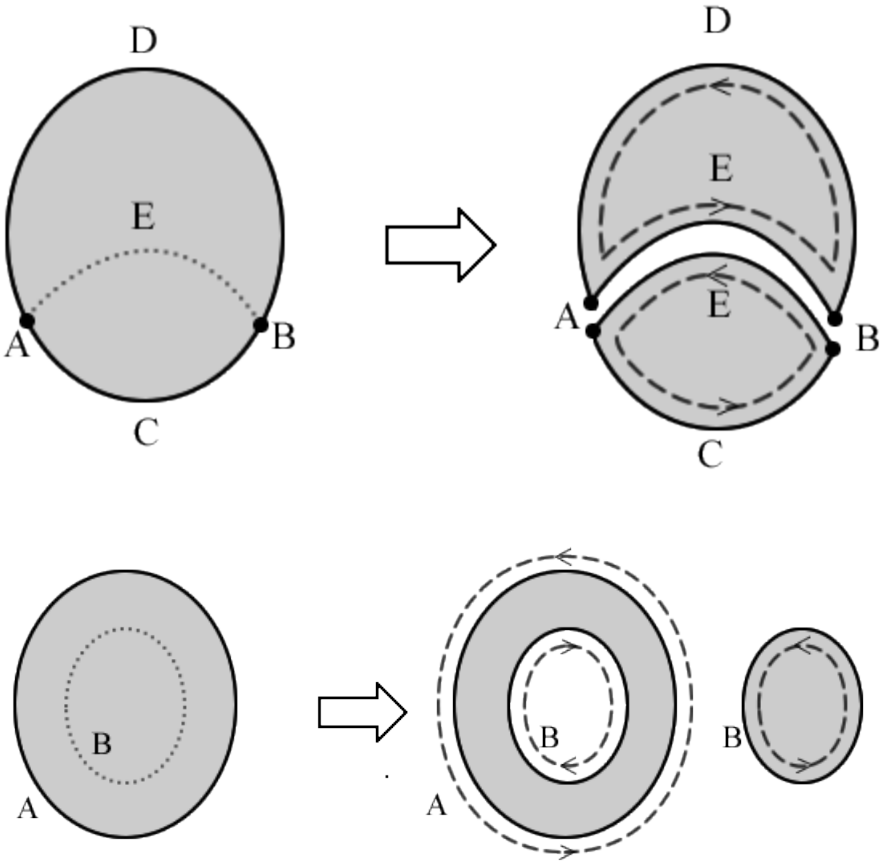


Figure 5.7: Basic topological splits (the loops on the right side are shown with an offset).

5.3 Meshing Procedure

In the automatic mesh generation procedure, mesh control parameters are specified onto the entities of the 2D geometric model defining a cross-section. The purposes of mesh control parameters are *(i)* meeting the mesh layout constraints, *(ii)* having the desired mesh gradation, *(iii)* controlling mesh quality, and *(iv)* meeting the needs of the simulation procedure. The full set of mesh control parameters includes:

- d_i to control spacing mesh vertices on a magnetic flux surface.
- $\delta\psi$ to control spacing of the magnetic flux surfaces. $\delta\psi$ specifies the change in the values of ψ between adjacent magnetic flux surfaces. The number of

magnetic flux surfaces used is proportional to $\frac{\psi_{max}-\psi_{min}}{\delta\psi}$, where ψ_{max} and ψ_{min} are the maximum and minimum values of ψ in the domain.

- Mesh size control on the model entities.
- Element shape control that determines the desired shape of mesh elements.
- Ansiotropic mesh size field that drives the mesh adaption.

The mesh generation procedure consists of four software components: *(i)* unstructured triangulation, *(ii)* layered mesh generation, *(iii)* general mesh modification, and *(iv)* toroidal mesh extrusion of the 2D mesh to create a 3D mesh of the full reactor.

5.3.1 Unstructured Triangulation

Unstructured triangulation creates a graded mesh in the portion of domain where there is no need to generate an one-element-deep mesh. A number of tools are available to generate unstructured triangular meshes. The Simmetrix $MeshSim$ [43] is used in mesh generation for XGC1 and M3D-C^1 due to its benefits of well controlled graded meshes based on a geometry based specification, and the ability to be incorporated as a component of an overall meshing procedure responsible for meshing selected portion of the domain. In case of XGC1, $MeshSim$ is used to mesh the geometric faces between the open magnetic flux surfaces and the wall boundary (faces D and F, and part of face C in Figure 5.3). In case of M3D-C^1, it is used to create an initial mesh of the entire 2D domain.

5.3.2 Layered Mesh Generation

In XGC1, the motion of particles is driven by the electromagnetic field and the particle orbits mainly follow the magnetic flux surfaces in axisymmetric equilibria of the plasmas since the magnetic field pertubation is small. The motion of particles changes the charge and current density that determines the field by the gyrokinetic Possion and Ampere's equations [11, 13]. Meshes following the equilibrium field lines improve the efficiency of parallel particle tracking, and the accuracy of derivative calculation along the field line. Therefore, in case of the mesh in XGC1, mesh

vertices and edges on the model edge and interior area need to follow the field lines associated with the magnetic flux surfaces. The constant electromagnetic turbulence contours also follow the equilibrium magnetic field closely. Thus, the meshes following magnetic field line also improve the efficiency of the field solvers.

The vertex placement on the toroidal cross-section is as follows. Assume the number of planes placed around the major axis of the torus is n. For the magnetic flux surface with $\psi = \psi_0$, an initial vertex, $\mathbf{L}(0) = [R_0, Z_0, 0]$, is picked up on the plane with $\varphi = 0$ and it satisfies $\psi(R_0, Z_0) = \psi_0$. The vertices on the curve (Equation 5.2) are placed at the sequence of the parameters defined as

$$t_i = \frac{i}{2n\pi},\ i = 0, 1, ..., N-1 \tag{5.3}$$

The sequence is terminated and an approximately closed curve is formed based on the mesh vertex spacing requirement, d_i, or it reaches the wall boundary. Note that φ of each vertex is replaced by a constant value and the vertices are projected to the same cross-section.

The parameter, d_i, is specified to set a tolerance of vertex spacing. The sequence of the vertices defined by Equation 5.3 is refined by bisecting the interval of parameters if the distance between vertices is greater than d_i. For example, bisecting the interval $[t_i, t_{i+1}]$ places an additional point $\mathbf{L}(\frac{t_i+t_{i+1}}{2})$ between points, $\mathbf{L}(t_i)$ and $\mathbf{L}(t_{i+1})$. The sequence of the vertices is coarsened if the distance between vertices falls below d_i.

Given the placement of the curves and vertices, triangular elements on the geometric faces between adjacent curves are created by an *one-element-deep marching procedure*. The procedure starts with an edge designated as an initial working edge. The two marching options of creating new elements in the marching direction are evaluated based on the validity of the element and the shape indicator. The validity requires that the new edge must fall between the working edge and the edge on the surface in the marching direction. The marching option with a better element shape is chosen to form a new element. The edge created between the layers becomes the new working edge. The procedure continues until the last element is created.

Figure 5.8 illustrates the one-element-deep marching procedure. Assume the

working edge is (i, j). The new edge created by the marching procedure can be either $(i, j+1)$ or $(j, i+1)$ and the corresponding new element is $(i, j, i+1)$ or $(j, i, j+1)$. According to the validity requirement, edges $\{(i, j), (i, j+1), (i, i+1)\}$ or $\{(j, i), (j, i+1), (j, j+1)\}$ must be placed clockwise or counter-clockwise. Therefore, element $(i, j, i+1)$ or $(j, i, j+1)$ must satisfy $(\vec{v}_{i,j} \times \vec{v}_{i,j+1}) \cdot (\vec{v}_{i,j+1} \times \vec{v}_{i,i+1}) > 0$ or $(\vec{v}_{j,i} \times \vec{v}_{j,i+1}) \cdot (\vec{v}_{j,i+1} \times \vec{v}_{j,j+1}) > 0$ to be valid, where $\vec{v}_{j,i}$ defines the vector from point i to point j. Element $(j, i, j+1)$ is invalid although it gives a better shape indicator, thus it won't be chosen as the new element in the marching procedure. Note that the current point placement controlled by d_i leads to the similar local mesh size on the two adjacent curves and the situation that neither element is valid does not happen.

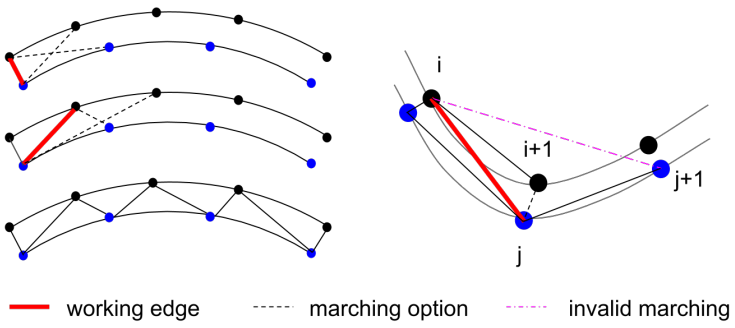


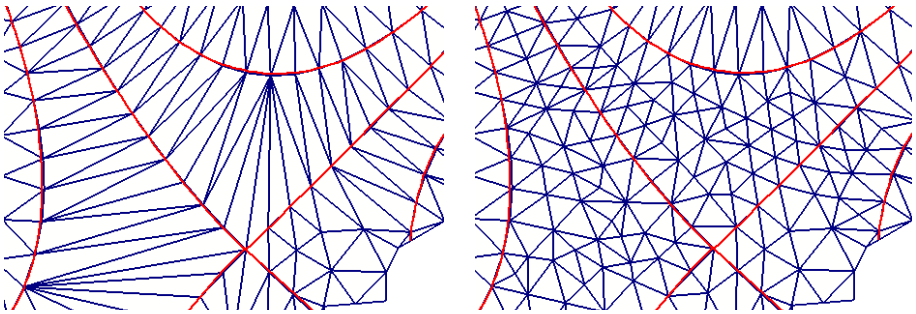
Figure 5.8: One-element-deep marching procedure to generate triangular mesh faces between curves.

5.3.3 Mesh Modification

General mesh modifications allow the users to adapt the unstructured mesh to match an anisotropic mesh size field defined over the initial mesh [69, 46, 70]. In order to evaluate the mesh quality, the mesh element is transformed by the metric tensor that defines the desired mesh size field [71] such that the mesh modification is controlled by the modification criterion and the desired mesh size field. The quality of the element size and shape are evaluated in the metric space. The type of the local mesh modification is chosen based on the evaluation of different operations [46].

In this paper, the mesh adaptation procedure is used to improve both the

initial unstructured mesh and the layered mesh in the selected areas, and/or to adapt them to control mesh discretization errors. In case of XGC1, the layered mesh generation controlled by $\delta\psi$ and d_i results in elements with poor shapes near the X-point. Therefore, mesh modification is used to improve the mesh quality and element shapes near the X-point. Figure 5.9 illustrates the layered mesh before and after mesh modification near the X-point. In M3D-C^1, an initial mesh is obtained by the unstructured triangulation controlled by the mesh size parameters specified on the model entities and mesh adaption is performed during the analysis. In this case, error indicators, or given functions, are used to define a new anisotropic mesh size field and a combined set of mesh modification operations are applied to convert the current mesh into one that satisfies the new mesh size field [14].



(a) initial (b) final

Figure 5.9: Improved mesh near the X-point.

5.3.4 3D Mesh Construction

In parallel simulations, a mesh is split into multiple *parts* for the purpose of distribution to processes. Therefore, each part consists of the set of mesh entities that is assigned to a process. For efficient manipulation, a part is uniquely identified within an entire mesh by a global part ID. Based on the adjacency relations, mesh entities on inter-part boundaries are duplicated to connect entities across parts such that they describe *part boundaries*. With the addition of part boundaries, each part is treated as a serial mesh. Each mesh entity duplicated on a inter-part boundary maintains a set of *remote copies* that is the memory location of mesh entity duplicated in the other part. The remote copy information is updated as the mesh

partitioning changes dynamically, which is required by mesh modification or load balancing [72, 44].

In a 2D M3D-C^1 analysis with P processes, the mesh is distributed into P parts. In a 3D M3D-C^1 simulation, N copies of the same 2D mesh are loaded into $N*P$ processes, where $N*P$ processes are divided into N process groups such that each process group loads the 2D mesh onto a set of P processes. Within a process group, each process is assigned with a rank p, $0 \leq p < P$, and a process group is uniquely identified as plane i, $0 \leq i < N$. For each plane i, the backward plane is the plane i-1, and the forward plane is the plane i+1. For plane *0*, the backward plane is plane N-1. For the plane N-1, the forward plane is the plane *0*. In order to switch the mesh from 2D to 3D, on each plane i, a remote copy of the forward plane is created and then quadrilateral mesh faces and wedge elements are created using the entities on plane i and the remote copy of the forward plane. The total number of 3D elements created is the number of triagular faces in the 2D mesh times the number of planes. Figure 5.10 illustrates a 3D mesh constructed with 8 planes.

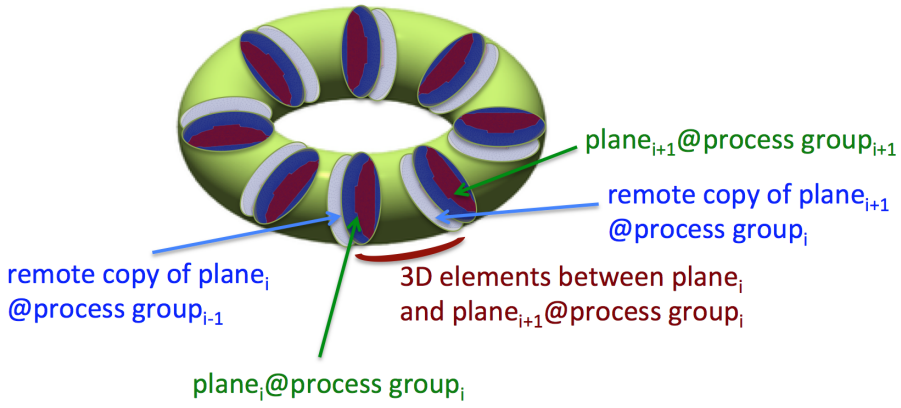


Figure 5.10: 3D mesh constructed on 8 process groups.

5.4 Examples

5.4.1 XGC1

Using the meshing control parameters discussed in Section 5.3, the steps for mesh generation are as follows: *(i)* the geometric model with the material wall boundary and magnetic flux surfaces interior to the domain is created, *(ii)* triangular

elements are created on the model face with matched ψ curves that are oriented in the opposite directions, *(iii)* mesh modification is applied to improve the elements in poor shape near the X-point, and *(iv)* the rest of area is filled with unstructured elements.

Figure 5.11 and 5.12 illustrate the mesh examples with different numbers of X-point in the simulation domain. Figure 5.13 depicts how $\delta\psi$ and d_i control the spacing of ψ curves and mesh vertices on each curve. Figure 5.14 illustrates how the targeted mesh size field affects mesh improvement in the X-point area.

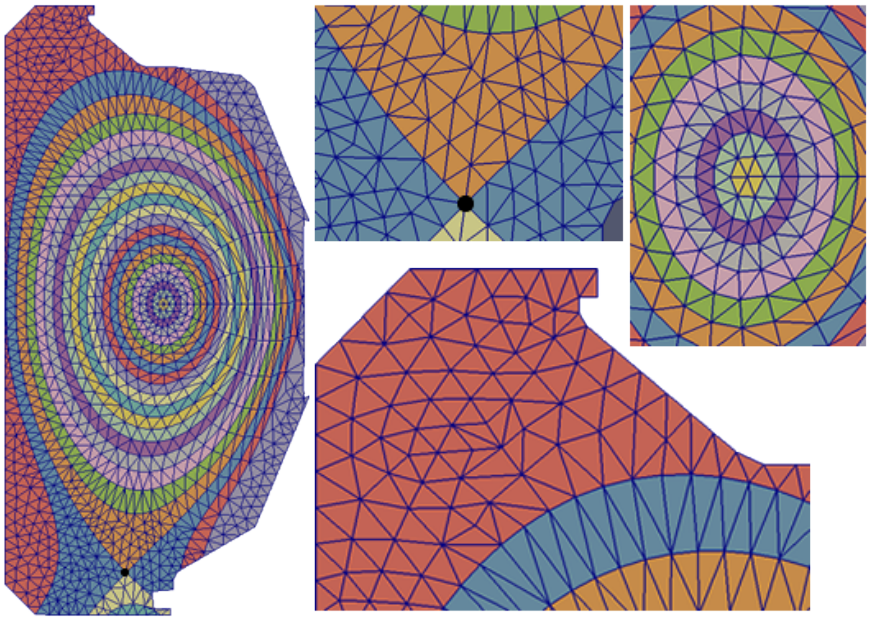


Figure 5.11: Mesh example with one X-point (labeled by the solid circle).

5.4.2 M3D-C^1

The simulation domain for M3D-C^1 consists of at most three areas that correspond to the plasma area, the finite-thickness material wall and the vacuum vessel. In a 2D mesh, unstructured mesh elements are created and then adapted by an ansiotropic size field. In a 3D mesh, the full torus is created by extrusion of the toroidal 2D meshes on toroidal cross-sections.

Figure 5.15 depicts an example of the initial mesh on the NSTX model with a finite-thickness wall. The mesh size is specified at the model faces of the plasma,

material wall and the vacuum vessel to control the initial mesh. Figure 5.16 illustrates an example of parallel mesh adaptation in four processes with the boundary curve defined as

$$\begin{aligned} R(t) &= c_1 + c_2 cos\left(t + c_3 sin(t)\right) \\ Z(t) &= c_4 + c_5 sin(t), 0 \leq t \leq 2\pi \end{aligned} \tag{5.4}$$

The mesh size field is calculated from the poloidal magnetic flux field ψ. A normalized field is defined as $\tilde{\psi} = \frac{\psi - \psi_0}{\psi_l - \psi_0}$, where ψ_l and ψ_0 are the field values at the plasma boundary and the magnetic axis, respectively. The mesh size normal to the surface is h_1 and the mesh size tangent to the surface is h_2 that are defined as

$$h_i^{-1} = \tilde{h}_i^{-1} + \frac{1}{l_{ci}\left(1 + \frac{\tilde{\psi} - \psi_c}{W_c}\right)^2}, \quad i = 1, 2 \tag{5.5}$$

where l_{ci}, ψ_c and W_c are constants. $\tilde{h}_i$ is defined as

$$\begin{aligned} \tilde{h}_i &= b_i[1 - e^{-|\frac{\tilde{\psi}}{a_1} - 1|^{a_2}}] + c_i, \quad \tilde{\psi} < a_1 \\ \tilde{h}_i &= d_i[1 - e^{-|\frac{\tilde{\psi}}{a_1} - 1|^{a_2}}] + c_i, \quad \tilde{\psi} > a_1 \end{aligned} \tag{5.6}$$

where b_i, d_i, a_i and c_i are constants. The constant parameters of the equations are determined such that the adapted mesh has finer size at the magnetic flux surface $\tilde{\psi} = a_1$. Since the solution to the physical equations varies more rapidly in the direction normal to the magnetic surfaces than within the surfaces, the directional mesh size fields are defined to represent this property. Figure 5.17 illustrates an example of a 3D mesh constructed with 64 planes.

5.5 Closing Remarks

This paper has presented a set of procedures for the automatic mesh generation with well-defined control parameters to satisfy the needs of two fusion plasma simulations codes, XGC1 and M3D-C^1. Core capabilities include:

- Employing a geometric model definition of the domain that represents physical

and physics components that must be reflected in the resulting mesh.

- Straightforward specification of the needed mesh control information in terms of the geometric model.
- A component-based mesh generation procedure that satisfies the constraints of the simulation procedures while creating well controlled graded meshes.

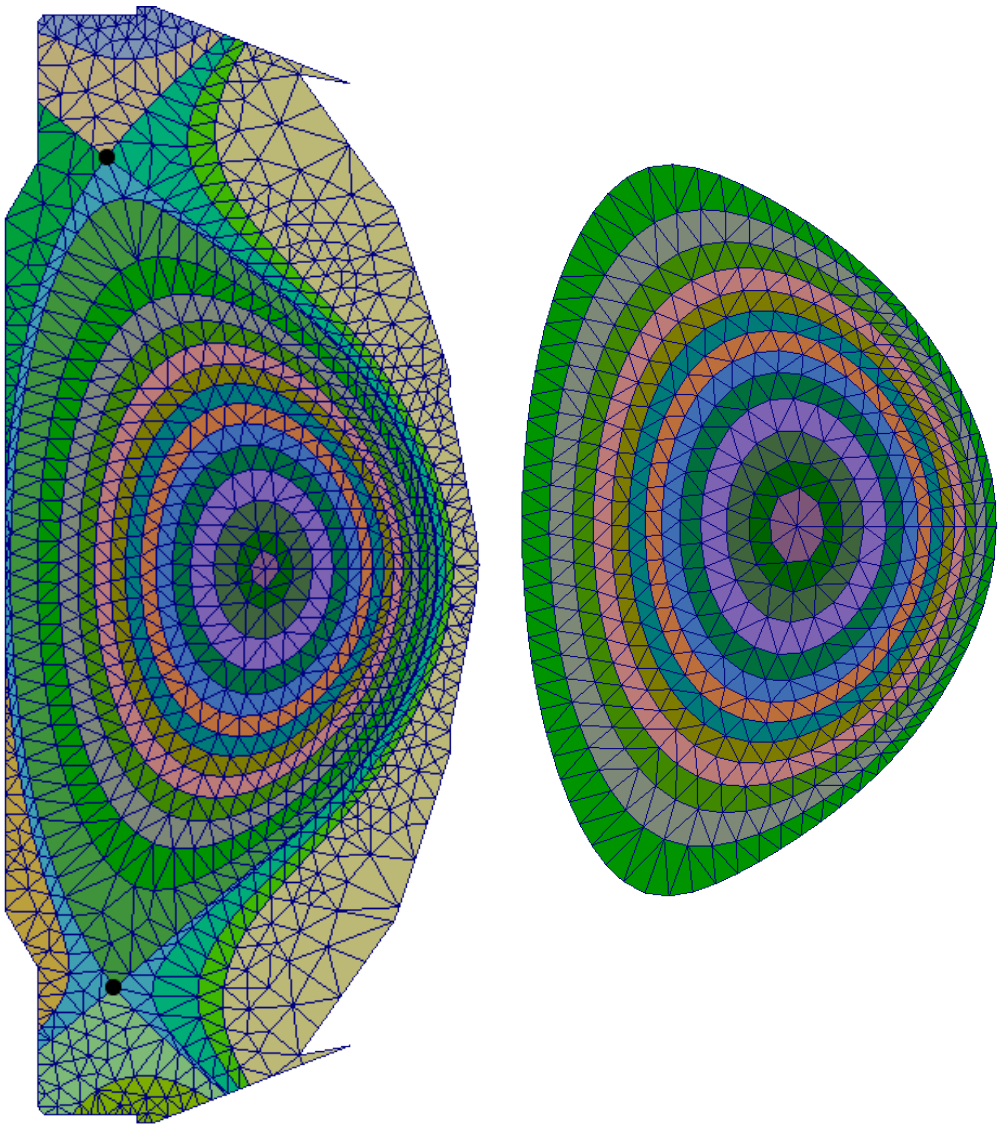
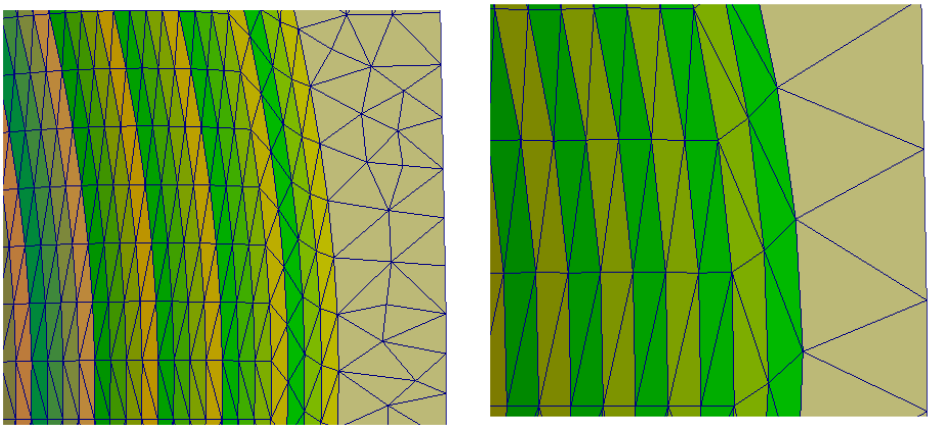
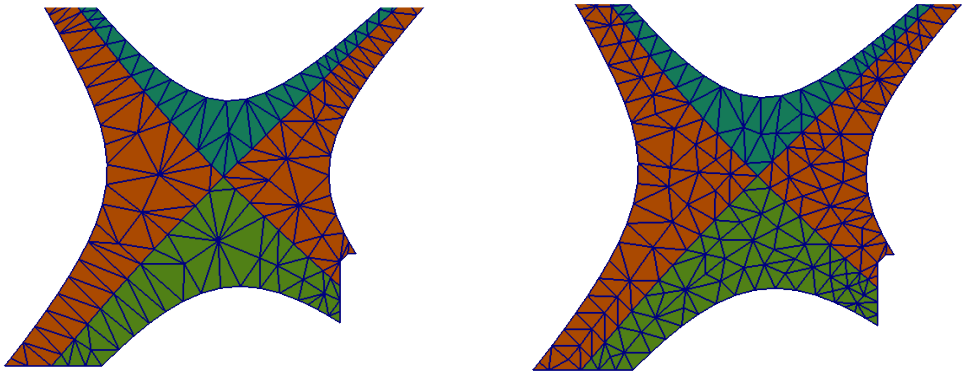


Figure 5.12: Mesh example with two X-points (left) and no X-point (right).



(a) $\delta\psi = 0.006$, $d_i = 3cm$ (b) $\delta\psi = 0.012$, $d_i = 6cm$

Figure 5.13: Two meshes with different field line placement ($\delta\psi$) and vertex spacing (d_i) parameters.



(a) $h = 2d_i$ (b) $h = d_i$

Figure 5.14: Improved X-point area by different targeted mesh sizes (h).

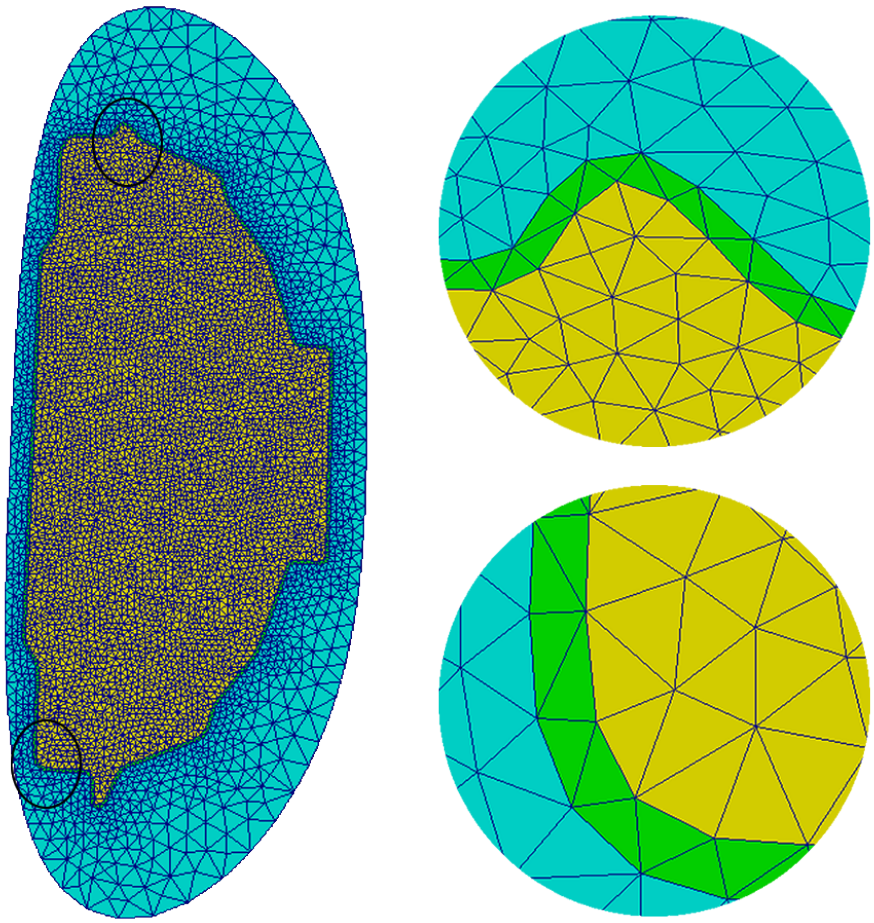
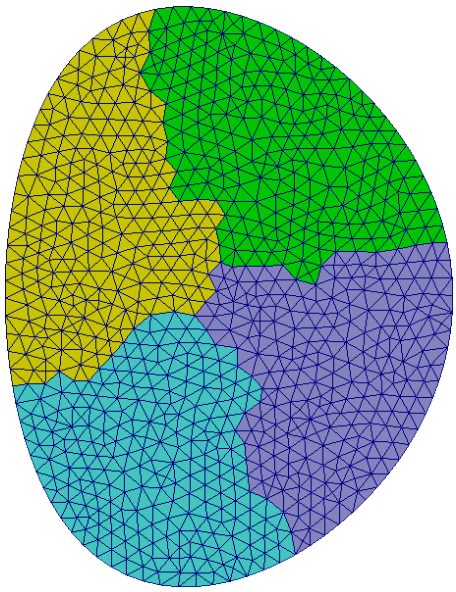
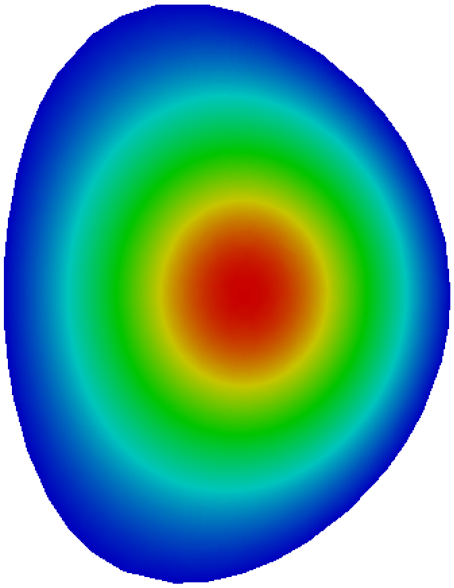


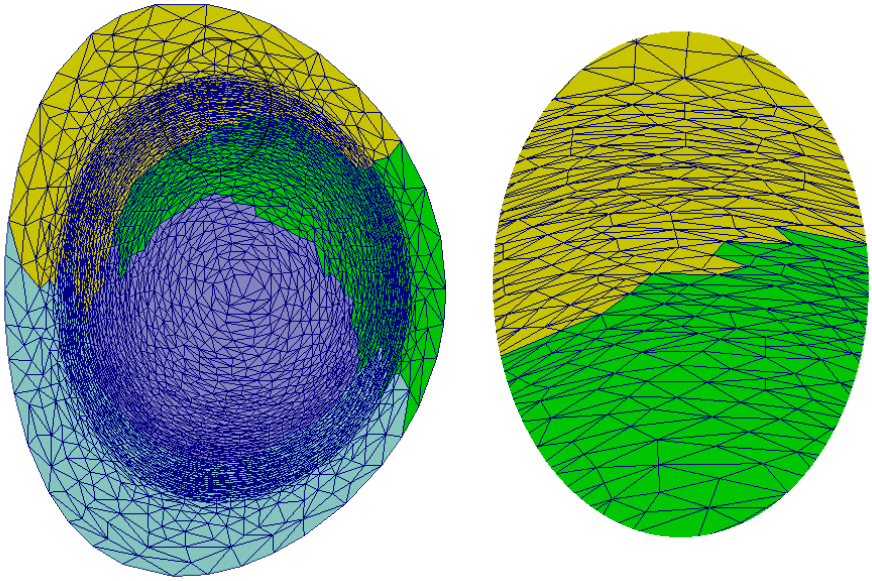
Figure 5.15: Initial mesh on the NSTX model with a finite-thickness wall.



(a) initial mesh (1678 elements) colored by process rank



(b) solution field on the initial mesh



(c) adapted mesh (5746 elements) and its close-up (colored by process rank)

Figure 5.16: Anisotropically-adapted mesh in M3D-C^1.

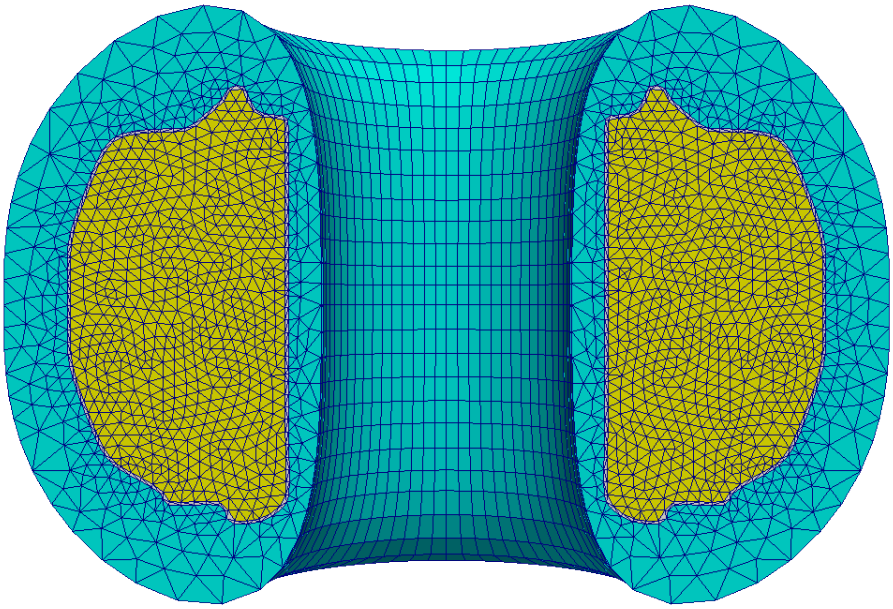


Figure 5.17: Cross-cut view of a 3D mesh with 64 planes in M3D-C^1.

CHAPTER 6
Improvement of Numerical Conditioning

The systems of the equations in M3D-C^1 are obtained by discretizing the fourth-order PDEs derived from the extended MHD (see Chapter 2). The systems of the equations are hard to solve due to the poor conditioning (Figure 6.1). Direct solvers such as SuperLu [45] are applied for problems on the 2D mesh and GMRES based iterative methods [73] with the 2D direct solver as the preconditioner are used to solve problems on the 3D mesh [6]. The usage of direct solvers becomes an issue due to loss of good scaling as the size of the problem simulated and the number of computing cores used increase. The effective usage of iteratively solvers requires the numerical conditioning of the global matrix improved.

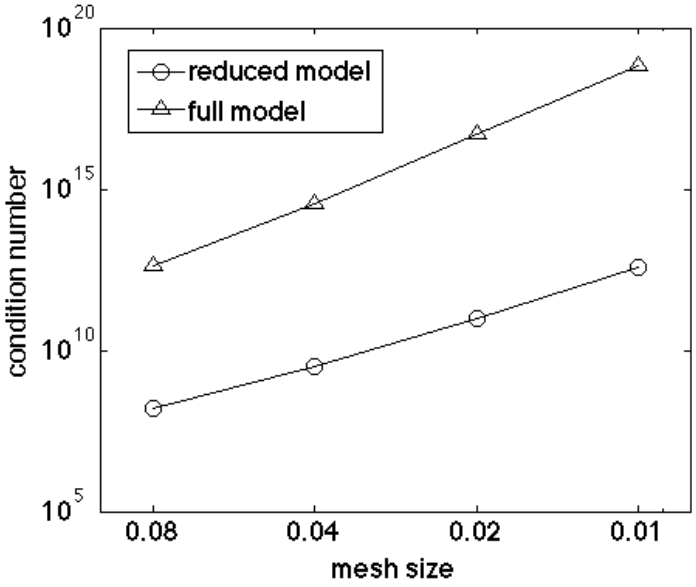


Figure 6.1: Condition number of the velocity matrix on a sequence of uniform mesh. The reduced model uses one-scalar representation and the full model uses three-scalar representation [6].

The ill-conditioning of the linear system is dictated by the complex physics that couple the multi-scale components [6] and the fourth-order PDE solved that requires the usage of the C^1 finite elements [21]. The procedure of defining shape functions

and obtaining the elemental contribution of the matrix, together with the spectrum analysis of the physics waves [6], were examined. A method by redefining the shape functions is introduced and a diagonal form is applied to improve the numerical conditioning. Three terms are introduced in the diagonal form. The first term is to account for the "element size" and "element size squared" introduced by using the first-order and second-order derivatives as degrees of freedom in the finite element discretization. The second is the physics-based scaling of the contributions to the multicomponent set of equations based on the grouping of eigenvalues associated with the components. The third accounts for integration over elements if the mesh size varies over the domain. The new linear system has both the diagonal and off-diagonal values in a closer range. The finite element is regularized to result in a linear system with a smaller condition number. The procedure of shape function modification can be viewed as a symmetric preconditioning method applied to the original system algebraically.

Section 6.1 defines the method of redefining the shape functions and its relation with the symmetric preconditioning. Section 6.2 discusses the three scaling factors in the diagonal form. The results of applying the scaling factors are discussed in Section 6.3.

The condition number of the sparse matrix is calculated by the matlab routine, *condest*, which estimates 1-norm condition number of a square matrix [74].

6.1 Elemental Regularization and Symmetric Preconditioning

The linear system derived from the finite element discretization of PDEs takes the form

$$\mathbf{K}\mathbf{d} = \mathbf{f}, \tag{6.1}$$

where $\mathbf{K}$ is the stiffness matrix, $\mathbf{d}$ is the displacement vector, and $\mathbf{f}$ is the load vector.

To alter the numerical conditions of the system, we define the new set of DOFs and associated shape functions $(\tilde{\mathbf{d}}, \tilde{\mathbf{N}})$ by applying a transformation to the

original set of DOFs and associated shape functions $(\mathbf{d}, \mathbf{N})$ through a non-singular transformation matrix $(\mathbf{S})$ that takes the form as

$$\tilde{\mathbf{d}} = \mathbf{S}^T\mathbf{d}, \tag{6.2a}$$

$$\tilde{\mathbf{N}} = \mathbf{S}^{-1}\mathbf{N}. \tag{6.2b}$$

The new linear system by the new DOFs and shape functions is

$$\tilde{\mathbf{K}}\tilde{\mathbf{d}} = \tilde{\mathbf{f}}, \tag{6.3}$$

where

$$\tilde{\mathbf{K}} = \mathbf{S}^{-1}\mathbf{K}\mathbf{S}^{-T}, \tag{6.4a}$$

$$\tilde{\mathbf{f}} = \mathbf{S}^{-1}\mathbf{f}. \tag{6.4b}$$

The process of defining the new DOFs and shape functions can be viewed as a split preconditioning applied to the original linear system [73]. The split preconditioning is symmetric in the sense that the right preconditioning matrix $(\mathbf{S}^{-T})$ is the transpose of the left preconditioning matrix $(\mathbf{S}^{-1})$.

The matrix $\mathbf{S}$ is the regulation matrix in order to transform the element to generate an algebraically equivalent but numerically better linear system. The matrix $\mathbf{S}$ satisfying the following properties are desired for numerical efficiency:

- $\tilde{\mathbf{K}} = \mathbf{S}^{-1}\mathbf{K}\mathbf{S}^{-T}$ is a better conditioned system.
- $\mathbf{S}$ and $\mathbf{S}^{-1}$ is easy to get.
- $\tilde{\mathbf{N}} = \mathbf{S}^{-1}\mathbf{N}$ still maintains the localized property of the shape functions that does not harm the efficiency of the finite element. The new shape functions associated with the mesh vertex are bounded by the mesh patch that are formed by the elements sharing the same vertex.

The simplest form of $\mathbf{S}$ is the diagonal matrix. Define a diagonal matrix $\mathbf{D}$ and the

new pair of DOFs ($\tilde{\mathbf{d}}$) and associated shape functions ($\tilde{\mathbf{N}}$) is

$$\tilde{\mathbf{d}} = \mathbf{D}\mathbf{d}, \tag{6.5a}$$

$$\tilde{\mathbf{N}} = \mathbf{D}^{-1}\mathbf{N}. \tag{6.5b}$$

It can be viewed as applying a symmetric diagonal preconditioning which takes the form of $\tilde{\mathbf{K}} = \mathbf{D}^{-1}\mathbf{K}\mathbf{D}^{-1}$ (see Equation 6.4a, given that $\mathbf{D}^{-1} = \mathbf{D}^{-T}$).

6.2 Definition of Scaling Factors

This section defines three terms that contribute to the diagonal of the transformation matrix, $\mathbf{D}$.

6.2.1 Scaling Factor due to Applying Derivatives as DOFs

Recall the DOFs of the C^1 triangle element applied by M3D-C^1 correspond to the function value, the first and second derivatives ([3], also see Section 2.3.1). The corresponding shape functions vary at the different rates of the mesh size and they can be scaled by a factor that is a function of mesh size.

[75] gives the definition of the same C^1 triangle element in an explicit form. The normalized multipliers ($\lambda \sim O(1)$) in the area coordinate (ξ, η) and the DOFs in the global coordinate (R, Z) are related by [75]

$$\begin{bmatrix} \lambda \\ \lambda_{,\xi} \\ \lambda_{,\eta} \\ \lambda_{,\xi\xi} \\ \lambda_{,\xi\eta} \\ \lambda_{,\eta\eta} \end{bmatrix} = \begin{bmatrix} 1 & & & & & \\ & O(h_R) & O(h_Z) & & & \\ & O(h_Z) & O(h_Z) & & & \\ & & & O(h_R^2) & O(h_R h_Z) & O(h_Z^2) \\ & & & O(h_R^2) & O(h_R h_Z) & O(h_Z^2) \\ & & & O(h_R^2) & O(h_R h_Z) & O(h_Z^2) \end{bmatrix} \begin{bmatrix} d \\ d_{,R} \\ d_{,Z} \\ d_{,RR} \\ d_{,RZ} \\ d_{,ZZ} \end{bmatrix}. \tag{6.6}$$

Note that "," means taking the derivative to keep the consistent notation with the previous chapters. The associated shape functions are transformed by the inverse transpose of the matrix and possess similar mesh size related scales [75].

The transformation matrix in Equation 6.6 illustrates how the shape functions

and the corresponding DOFs depend on the mesh size. The scaling factor of the i^{th} DOF can therefore be defined as h_i^d, where $d = 0, 1, 2$ depending the order of derivative of the i^{th} DOF, and h_i is the nodal element size in the direction of derivative that the i^{th} DOF attached to. Specially, $h_i = (h_R h_Z)^{\frac{1}{2}}$ for the DOF corresponding to the twisted derivative ($d_{,RZ}$).

A simple example can illustrate how the scaling factor of the DOFs can rescale both the diagonal and off-diagonal terms of the matrix. Assume the original linear system takes the form

$$\mathbf{K} = \begin{bmatrix} 1 & h_R & h_Z & h_R^2 & h_R h_Z & h_Z^2 \\ h_R & h_R^2 & h_R h_Z & h_R^3 & h_R^2 h_Z & h_R h_Z^2 \\ h_Z & h_R h_Z & h_Z^2 & h_R^2 h_Z & h_R h_Z^2 & h_Z^3 \\ h_R^2 & h_R^3 & h_R^2 h_Z & h_Z^4 & h_R^3 h_Z & h_R^2 h_Z^2 \\ h_R h_Z & h_R^2 h_Z & h_R h_Z^2 & h_R^3 h_Z & h_R^2 h_Z^2 & h_R h_Z^3 \\ h_Z^2 & h_R h_Z^2 & h_Z^3 & h_R^2 h_Z^2 & h_R h_Z^3 & h_Z^4 \end{bmatrix}. \tag{6.7}$$

Apply the diagonal rescaling matrix as

$$\mathbf{D}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{h_R} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{h_Z} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{h_R^2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{h_R h_Z} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{h_Z^2} \end{bmatrix}, \tag{6.8}$$

and it results in a system

$$\tilde{\mathbf{K}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{6.9}$$

6.2.2 Scaling Factor due to Multicomponent of the Velocity

The velocity of the full MHD model has three scalar components and these components are weakly coupled after applying the annihilation operators ([6], also see Equation 2.14 in Section 2.2). The velocity in the full MHD model is represented by the scalar components, (U, ω, χ), in the cylindrical coordinate (R, φ, z) as (also see Equation 2.12 in Section 2.2)

$$\mathbf{V} = R^2 \nabla U \times \nabla \varphi + R^2 \omega \nabla \varphi + \frac{1}{R^2} \nabla_\perp \chi. \tag{6.10}$$

The reduced models of two-scalar representation is

$$\mathbf{V} = R^2 \nabla U \times \nabla \varphi + R^2 \omega \nabla \varphi, \tag{6.11}$$

and the one-scalar representation is

$$\mathbf{V} = R^2 \nabla U \times \nabla \varphi. \tag{6.12}$$

The analysis of eigenvalues of the stiffness matrix [6] indicates that different scalar component represent different groups of waves in the plasmas. Three factors (α_1, α_2, and α_3) are introduced to rescale the shape functions such that different groups of eigenvalues are in a closer range. This is equivalent to rewrite Equation 6.10 as

$$\mathbf{V} = \alpha_1^2 R^2 \nabla U \times \nabla \varphi + \alpha_2^2 \cdot R^2 \omega \nabla \varphi + \alpha_3^2 \cdot \frac{1}{R^2} \nabla_\perp \chi. \tag{6.13}$$

6.2.3 Scaling Factor due to Integration over Element

Elemental contribution of the matrix is also a function of the element size. The variation of the magnitude due to integration over elements with quite different sizes can worsen the numerical conditioning of the system. A scaling factor is introduce to account for this effect.

Define the physical coordinate $\mathbf{x}$, the parametric coordinate ξ and the mapping $\mathbf{x} = \mathbf{x}(\xi)$. The Jacobian matrix and its determinant are $\mathbf{J} = \frac{d\mathbf{x}}{d\xi}$ and $J = det(\mathbf{J})$.

Some common terms in the integration of the weak form is listed as follows:

$$
\begin{aligned}
d\Omega_x &= J d\Omega_\xi, && (6.14a)\\
\nabla_x \mu &= \mathbf{J}^{-T} \nabla_\xi \mu, && (6.14b)\\
\nabla_x \mu \cdot \nabla_x \nu &= \mathbf{J}^{-1}\mathbf{J}^{-T} : \nabla_\xi \mu \nabla_\xi \nu, && (6.14c)\\
< \mu, \nu >_x \equiv \mu_{,x}\nu_{,y} - \mu_{,y}\nu_{,x} &= \frac{1}{J}\left[\mu, \nu\right]_\xi . && (6.14d)
\end{aligned}
$$

If $\mathbf{J}$ is constant in the domain of the element (e.g., area coordinate), the magnitude of $\mathbf{J}$ and its determinate J is related to the element size as

$$
\begin{aligned}
\mathbf{J} &\sim h, && (6.15a)\\
J &\sim h^{dim}. && (6.15b)
\end{aligned}
$$

where *dim* is the dimension of the domain. Considering that the integration in the parametric coordinate without the Jacobian is usually normalized, we have an estimation of the order of integration terms as,

$$
\begin{aligned}
\int d\Omega_x &\sim h^{dim}, && (6.16a)\\
\int \mu(\mathbf{x})\nu(\mathbf{x}) d\Omega_x &\sim h^{dim}, && (6.16b)\\
\int \nabla_x \mu \cdot \nabla_x \nu d\Omega_x &\sim h^{dim-2}, && (6.16c)\\
\int \nabla_x^2 \mu \cdot \nabla_x^2 \nu d\Omega_x &\sim h^{dim-4}. && (6.16d)
\end{aligned}
$$

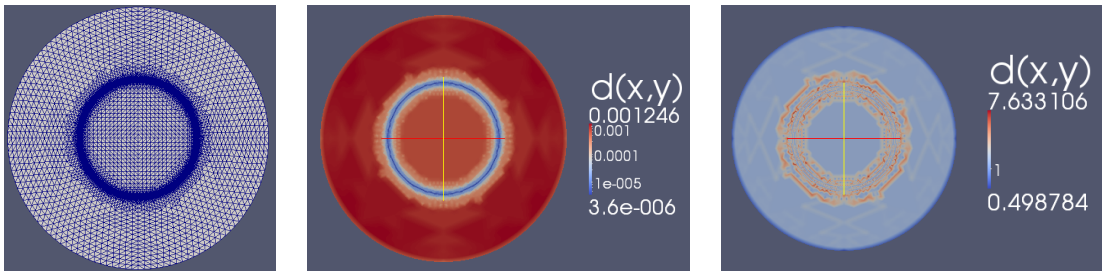
Therefore the magnitude of the result can be scaled by h^{dim-n} which depends on the total order of the derivative n appearing in the integration. The scaling factor $f(J)$ due to integration can be defined as

$$
f(J) = \sqrt{h^{dim-n}} = h^{(dim-n)/2}, \tag{6.17}
$$

and specifically

$$
f(J) = h^{1-n/2} \tag{6.18}
$$

for the 2D case. Figure 6.2 is an example that illustrates how the magnitude of



(a) mesh (b) original (c) rescaled by $f(J) = h$

Figure 6.2: Magnitude of diagonal entries that correspond to the function value in the mass matrix on a graded mesh before and after rescaling by $f(J) = h$.

diagonal entries varies due to integration and they are in closer range after applying the rescaling factor. A graded mesh on a circular domain with the radius as r_0 is used in Figure 6.2. The mesh size is 0.002 at $r = 0.5r_0$, 0.03 at $r = 0$ and 0.04 at $r = r_0$. Mass matrix is obtained by integrating product of the shape functions as $\int \mu_i \nu_j d\Omega \sim h^2$. The diagonal entries which correspond to the function value vary from $4E-6$ to $1E-3$ while the mesh size varies from $0.002 - 0.03$. After applying the scaling factor $f(J) = h$, the magnitude of the diagonals varies from 0.5 to 8.

6.2.4 Overview

The three scaling factors in the elemental equilibration matrix are put together and the final diagonal transform matrix takes the form as

$$D_{ij} = \alpha_i h_i^d \tilde{h}_i^{1-n/2} \delta_{ij}, \tag{6.19}$$

where $d = 0, 1, 2$ depending the order of derivative of the i^{th} DOF, h_i is the nodal element size in the direction of derivative that the i^{th} DOF attached to, $\tilde{h}_i$ is the nodal element size which evaluates the area of the integration domain, and n is the total order of derivatives in the dominant part of the integration.

Table 6.1: Condition numbers of the original velocity matrix and the regularized matrix by applying the diagonal rescaling factors on a sequence of uniform meshes. The number of DOFs of the largest matrix is 140,994.

mesh size		0.08	0.04	0.02	0.01
original	reduced model	1.5E08	3.0E09	9.8E10	3.8E12
	full model	4.0E12	3.5E14	5.0E16	6.9E18
regularized	reduced model	2.2E04	4.4E04	1.7E05	1.1E06
	full model	1.1E08	1.5E07	2.4E07	3.6E08

6.2.5 Extension to 3D Element

The 3D shape functions are defined by taking tensor product of the C^1 reduced quintic shape functions and the Hermite cubic polynomials [6]. The DOFs are listed in Table 2.6 in Section 2.3.3. The new DOFs correspond to field values that takes the derivative in the φ direction. It can be rescaled following the same procedure that rescales the derivatives in the 2D plane. Define the mesh size in the φ direction as h_φ, and the scaling factor by Equation 6.19 is extended to the 3D element,

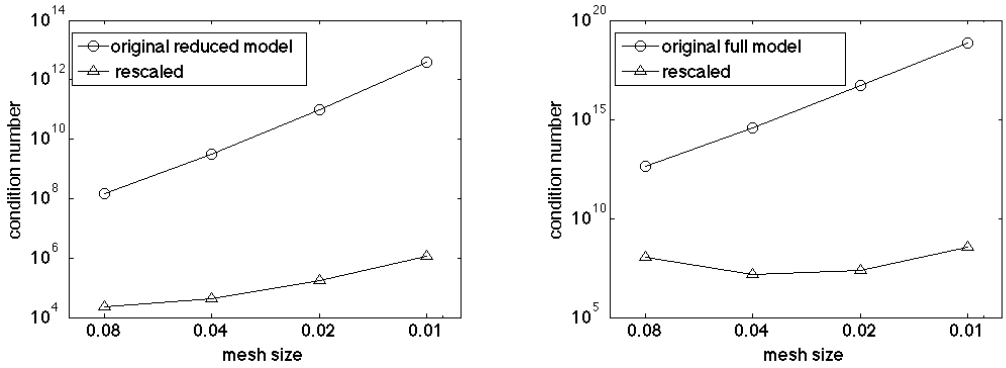
$$D_{ij} = \alpha_i h_i^d h_\varphi^{d'} \tilde{h}_i^{1-n/2} \delta_{ij}, \tag{6.20}$$

where d' is the order of derivative in the φ direction ($d' = 0, 1$) and the other symbols have the same meaning as Equation 6.19.

6.3 Results

The result on a sequence of uniformly refined meshes is presented first (Figure 6.3 and Table 6.1). The condition number of the regularized system is reduced by $10^4 \sim 10^{10}$ for the full model with three components of the velocity. Despite of the simple form of elemental regulation, the condition number is substantially reduced.

Table 6.2 and Table 6.3 illustrate how the scaling factor, $f(J)$, due to integration can improve the numerical conditioning on an anisotropic mesh (Figure 6.4). The highest total order of derivatives is fourth to obtain the elemental contribution of the matrix. $f(J)$ is set to be $\tilde{h}^{-1}$ (Equation 6.18 by setting $n = 4$). Table 6.2 lists



(a) reduced model (one-scalar representation)

(b) full model (three-scalar representation)

Figure 6.3: Condition number of the original velocity matrix and the regularized matrix by applying the diagonal rescaling factors.

Table 6.2: Magnitude of the diagonal entries of the linear system.

	h_R	h_Z	$\frac{h_Z}{h_R}$	$d(R,Z)$	$d_{,R}$	$d_{,Z}$	$d_{,RR}$	$d_{,RZ}$	$d_{,ZZ}$
original	0.03	0.03	1	1.6E+03	3.6E-02	7.8E-02	1.0E-06	6.1E-06	5.5E-06
	0.004	0.04	10	3.4E+03	9.5E-02	1.1E-00	3.0E-07	1.1E-05	7.8E-05
$f(J)=1$	0.03	0.03	1	1.6E+02	3.5E+01	8.0E+01	1.0E+00	6.0E+00	5.6E+00
	0.004	0.04	10	3.4E+03	6.0E+03	7.1E+02	1.2E+03	4.9E+02	3.6E+01
$f(J)=\tilde{h}^{-1}$	0.03	0.03	1	1.5E-01	3.2E-02	7.2E-02	9.0E-04	5.5E-03	5.1E-03
	0.004	0.04	10	7.1E-01	1.3E-00	1.5E-01	2.5E-01	1.0E-01	7.6E-03

the typical values of diagonal entries of the original linear system on the anisotropic mesh. The order of difference is reduced from 10^9 to 10^3 after rescaling the DOFs due to derivatives, and is further reduced to 10^2 after applying $f(J) = h^{-1}$. The condition number is reduced from $8.2E+12$ to $6.6E+06$ after applying the scaling factor due to DOfs of derivatives and further reduced to $1.2E+06$ by applying $f(J) = \tilde{h}^{-1}$ (Table 6.3).

A 3D test problem with up to 55,296 dof (4 planes and 384 nodes each plane) is tested. Table 6.4 shows the difference of the condition number and the number of iterations to convergence when applying the block Jacobi preconditioner with

Table 6.3: Condition number of the linear system on the anisotropic mesh.

original	$f(J)=1$	$f(J)=h^{-1}$
8.2E+12	6.6E+06	1.2E+06

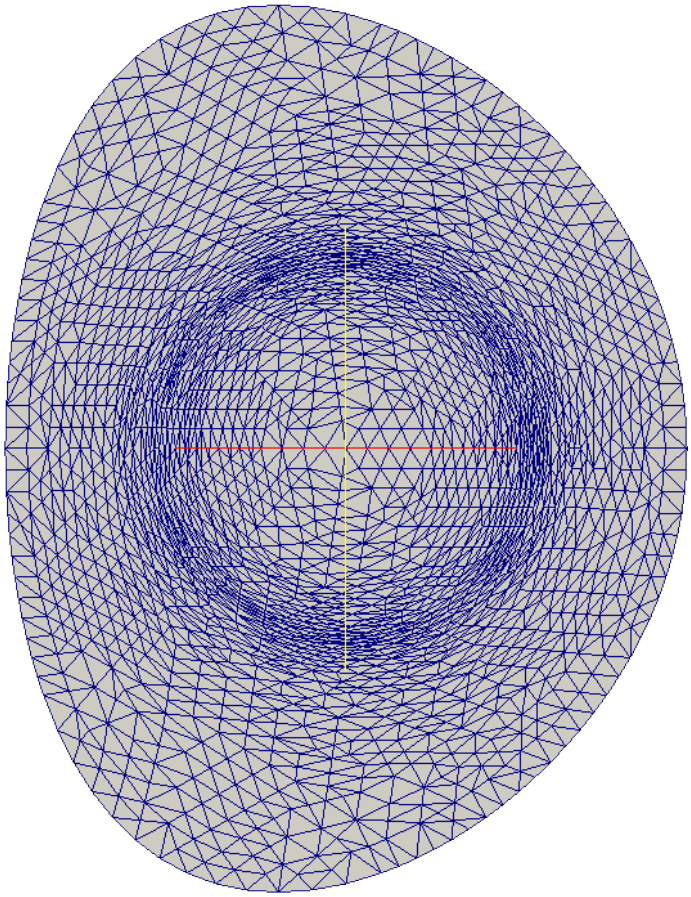


Figure 6.4: Mesh with the ratio of anisotropic mesh sizes in the two directions up to 10.

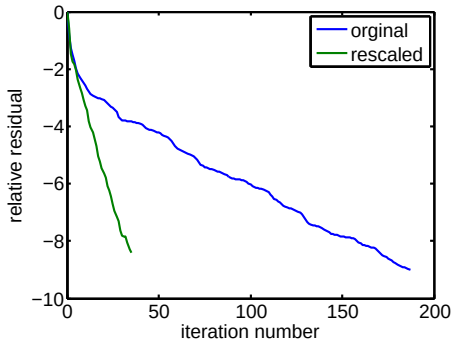
each block factorized by incomplete LU. The condition number of the matrices is reduced by the order of $\sim 10^5$. The velocity matrix ($s1\_mat$) applies rescale factor that takes both mesh size and different scalar variable into account and thus the condition number is reduced by a larger amount than the other linear systems. Figure 6.5 shows the change of convergence behaviors. The regularized matrices by applying the diagonal rescaling factors give smaller number of iteration for the linear systems that advance the density and magnetic field. The velocity equation fails to converge due to the fact the fourth-oder differential operators dominate (see Section 2.2).

It can been seen from the numerical tests that the elemental regulation by the diagonal scaling factors greatly reduces the condition number of the linear systems of equations in M3D-C^1. Although the method alone is likely not enough to achieve

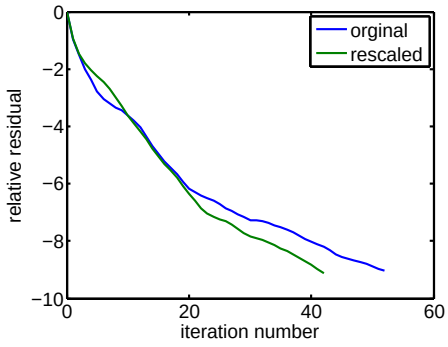
Table 6.4: Condition numbers of the original systems ($cond_{org}$) and regularized systems ($cond_{reg}$).

matrix	DOF/node	# unknowns	$cond_{org}$	$cond_{reg}$	$Iter_{org}$	$Iter_{reg}$
$mass\_mat$	12	18,432	4.4E14	6.1E09	187	35
$s8\_mat$	12	18,432	2.4E14	1.78E09	52	42
$s1\_mat$	36	55,296	8.0E18	1.8E10	fail to converge	
$s9\_mat$	24	36,864	1.2E14	1.8E09	593	606
$s2\_mat$	36	55,296	1.0E16	1.4E11	120	77

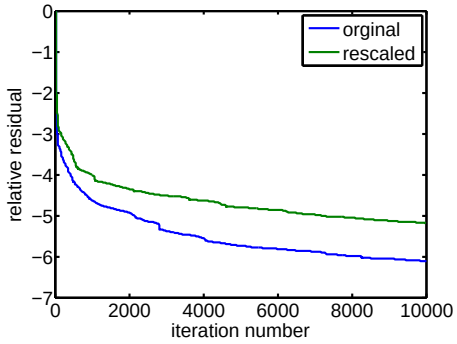
the most efficient iterative solving process, it provides a better starting point to investigate more sophisticated solving methods [76, 77] in the future.



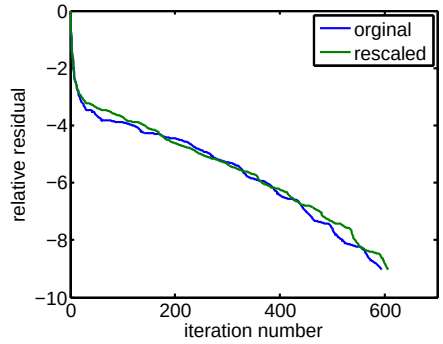
(a) *mass_mat*: mass matrix



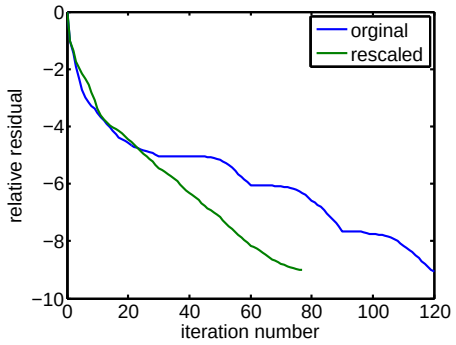
(b) *s8_mat*: advance density



(c) *s1_mat*: advance velocity



(d) *s9_mat*: advance pressure



(e) *s2_mat*: advance magnetic field

Figure 6.5: Convergence behavior of the original and the regularized linear systems by applying the diagonal rescaling factors. The linear systems are solved by block Jacobi preconditioned GMRES. Each block is factorized by the incomplete LU (ILU) which is cheaper than the complete LU.

CHAPTER 7
Examples of Parallel Adaptive Simulations with M3D-C^1

This chapter demonstrates examples of the parallel adaptive simulations with M3D-C^1. Section 7.1 discusses the adaptive error control used by the examples. Section 7.2 presents the adaptive non-linear simulation of the tilt mode. Section 7.3 and Section 7.4 study the linear instability of the double tearing mode and the edge localized mode on the adapted meshes, respectively. Section 7.5 demonstrates the capability of switching the simulation loop from the 2D mesh to the 3D mesh.

7.1 Adaptive Error Control

The explicit error estimator defined by Section 3.2 is used to drive the adaptive simulation loop (Section 4.1) on the 2D mesh. $\epsilon_{U\mathcal{K}}$ and $\epsilon_{\psi\mathcal{K}}$ estimate the spatial discretization error for the U and ψ equations (Section 2.2.4), respectively. In order to decide the desired mesh size, a single element error indicator is defined by [42]

$$\epsilon_{\mathcal{K}} = \sqrt{\epsilon_{U\mathcal{K}}^2 + \epsilon_{\psi\mathcal{K}}^2}. \tag{7.1}$$

Given $\epsilon_{\mathcal{K}}$, the desired mesh size field is specified to control the element error by [55, 78]

$$\frac{h_{new}}{h_{old}} = \left(\frac{\tau}{\epsilon_{\mathcal{K}}}\right)^{\frac{2}{2p+d}}, \tag{7.2}$$

where h_{new} is the targeted mesh size, h_{old} is the original mesh size, $\epsilon_{\mathcal{K}}$ is the estimated element contribution to the error, τ is the targeted element error, d is the spatial dimension ($d = 2$ here), and p is the convergence rate of the solution depending on the smoothness of the solution and the order of polynomial completeness in the finite element [24]. The order of polynomial completeness for the C^1 triangle element used by M3D-C^1 is fourth (Section 2.3.1). The convergence rate in H^2 norm is equal to 3 if the solution is smooth enough in the sense that $U \in H^4$ [21]. In practice, $p \leq 3$ is used. The bounds of the relative mesh size ($\frac{h_{new}}{h_{old}}$) and the absolute mesh size (h_{new}), and the maximum number of nodes in the adapted mesh are also specified

to control the adaptive result.

For the non-linear adaptive simulation, such as the tilt mode in Section 7.2, the solution is transferred during the local mesh modification (Section 4.5). For the linear simulation that studies the instability of the eigen-modes in the perturbed plasma equilibria, such as the double tearing mode in Section 7.3, the simulation goes back the stage of setting the initial condition (the perturbed plasma equilibrium) on the new mesh after the mesh is adapted.

7.2 Tilt Mode

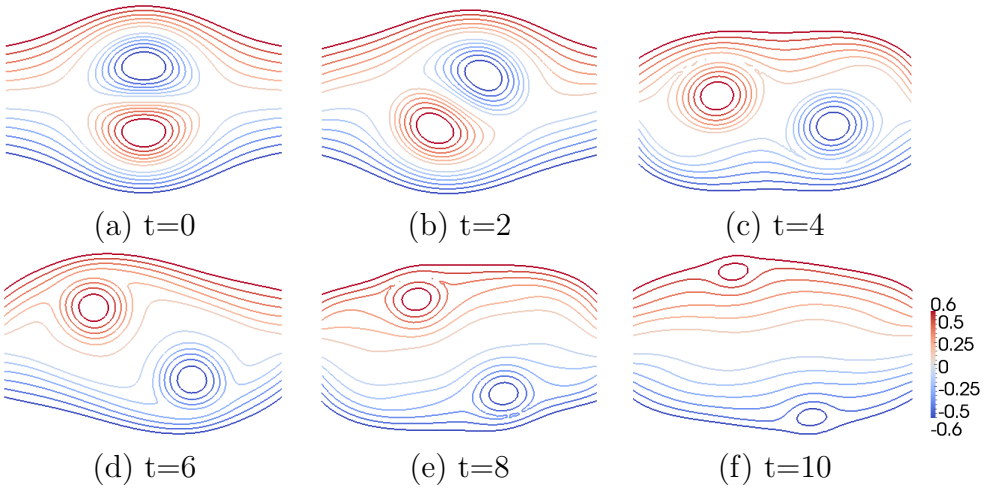
The error estimator defined by Equation 3.26 in Section 3.2 is applied to the simulation of the tilt mode [3, 25, 79]. The problem is described by Equation 3.4. The initial condition is set as

$$\psi^0 = \begin{cases} 2/kJ_0(k)J_1(kr)cos\theta, & r > 1, \\ (r - 1/r)cos\theta, & r < 1, \end{cases} \tag{7.3a}$$

$$U^0 = 0.1e^{-r^2}cos(z). \tag{7.3b}$$

ψ^0 defines the magnetic field of the bipolar vortexes (Figure 7.1) and they correspond two anti-parallel toroidal currents. U^0 defines a clockwise rotation. The bipolar vortexes rotate around the geometric center. When aligned horizontally, the vortexes start compelling each other until they vanish at the wall boundary. The contour plots of the ψ field and the toroidal current density are illustrated in Figure 7.1 and 7.2. The results are obtained by M3D-C^1 on a $[0, 4] \times [0, 4]$ domain with 6219 nodes. The fluid viscosity and electrical resistivity are set as $\mu = 0.005$ and $\eta = 0.001$, respectively. The time step, δt, is set as 0.05. It can be seen that the toroidal current localizes at the leading edges of the vortexes and forms the current sheets during the process (Figure 7.2).

The simulation result on the adapted mesh and the uniformly refined meshes are compared. Three uniformly refined meshes with the numbers of mesh nodes as 409, 1544 and 6219 nodes are used. The minimum mesh size in the adapted meshes is set to be the same as the mesh with 6219 nodes. Figure 7.3 illustrates the toroidal



(a) t=0 (b) t=2 (c) t=4

(d) t=6 (e) t=8 (f) t=10

Figure 7.1: Contour plots of the ψ field in the tilt mode.

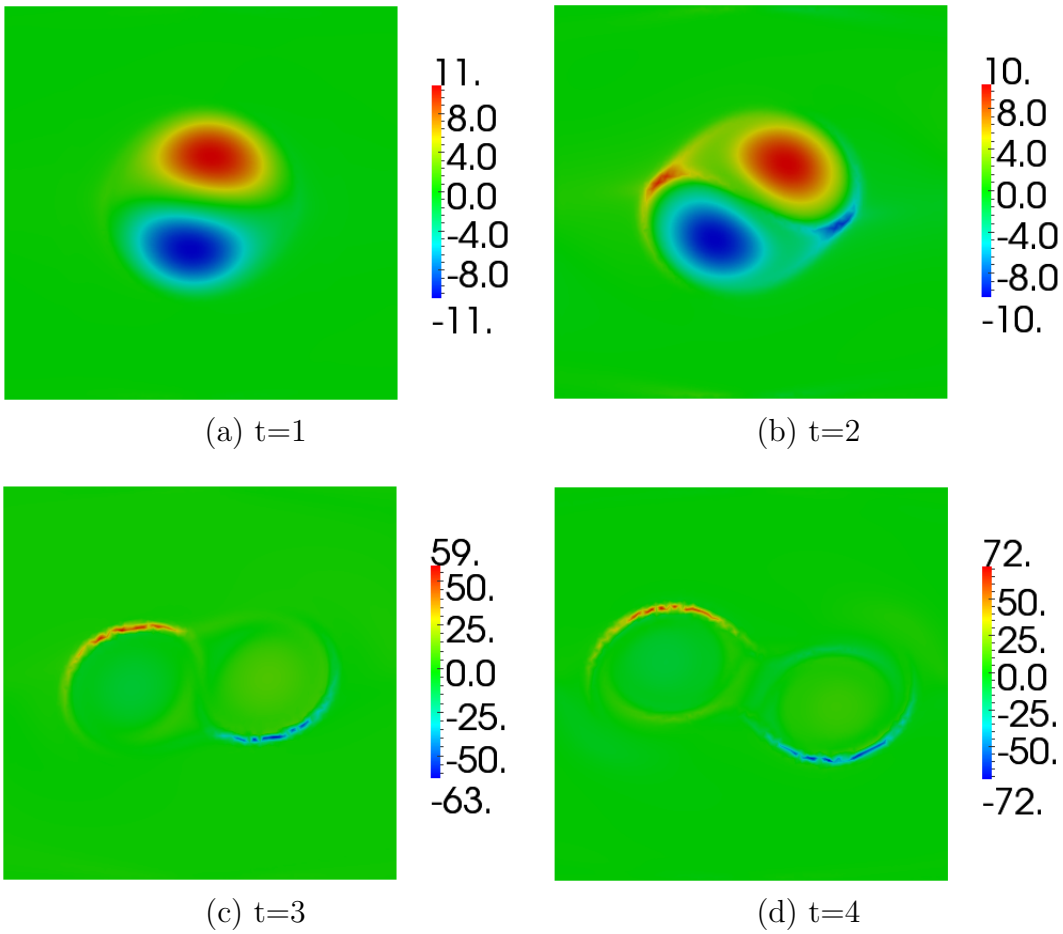
current density on the adapted meshes at different time steps. It can be seen that the mesh is adaptively refined at the leading edges of the vortexes where the current sheets locate.

Figure 7.4 plots the U field at t=6 on the adapted mesh and uniformly refined meshes. The contours of U in the figure show the structures of the streamline. It shows that the structure of the streamline in the fluid on the adapted mesh matches with the result on the finest uniform mesh (b in Figure 7.4). Figure 7.5 illustrates the U field at t=8 on the adapted mesh and uniformly refined meshes. It can be seen that the adapted mesh with 674 nodes shows a more clearly better solution of the fluid streamline (U field) than the uniform meshes with 409 and 1544 nodes as the simulation time accumulates.

The kinetic energy of the fluid is defined as

$$E_k = \int_\Omega \frac{1}{2} V^2 d\Omega = \int_\Omega \frac{1}{2} \nabla U \cdot \nabla U d\Omega. \tag{7.4}$$

Note that $\rho = 1$ and the velocity is represented by U only for the tests. Figure 7.6 compares the kinetic energy by the simulations on the uniformly refined meshes and the adaptive mesh. In the adaptive simulations, a coarse mesh is maintained when the simulation is in the linear region ($t < 3$), and the mesh is refined at the place that requires higher resolution when the simulation goes to the non-linear region.

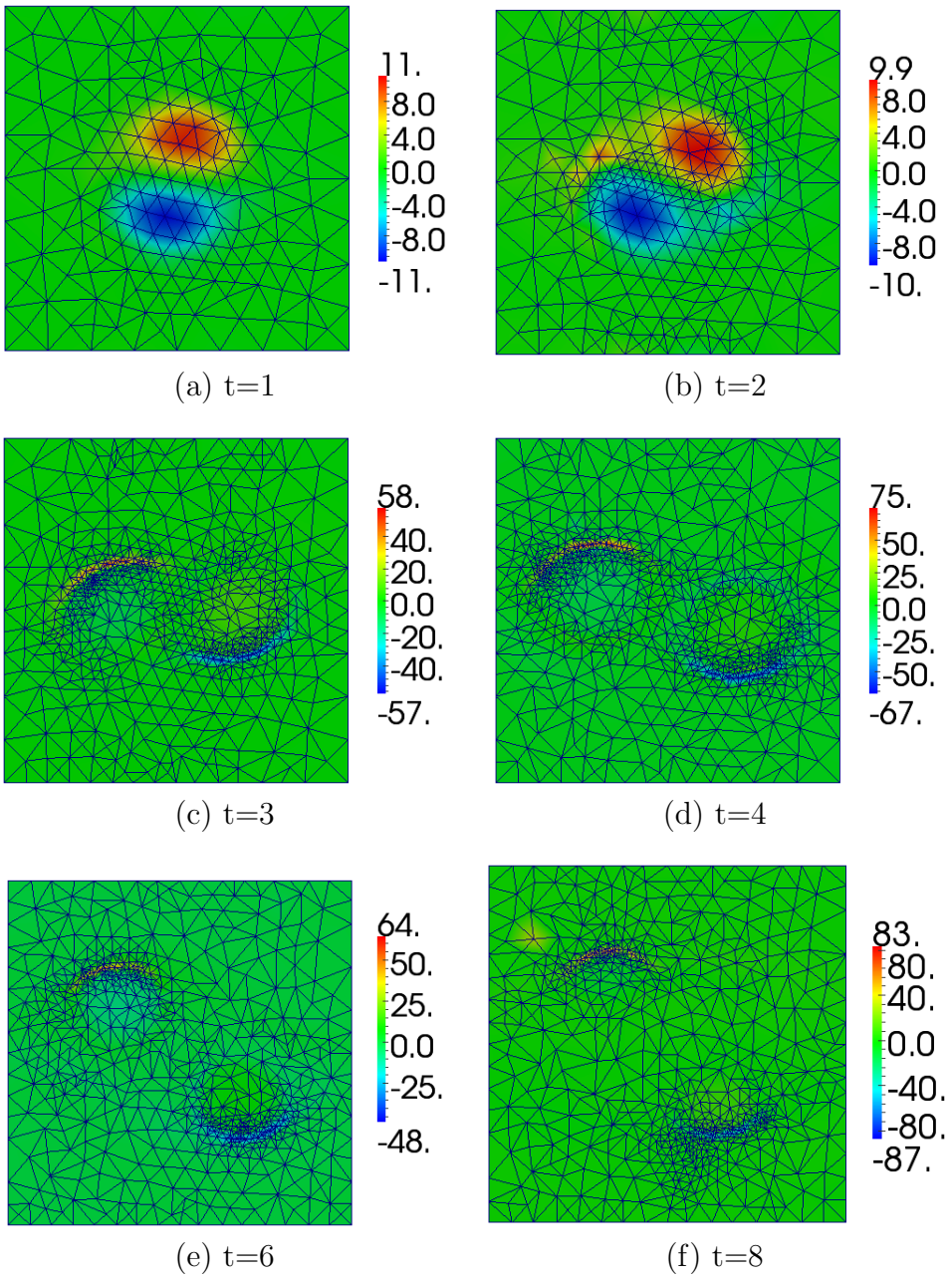


(a) t=1 (b) t=2

(c) t=3 (d) t=4

Figure 7.2: Toroidal current density in the tilt instability on the uniform mesh with 6219 nodes.

It can be seen that the adapted mesh captures the non-linear behavior in terms of the kinetic energy change with the mesh size smaller than 800 nodes. The adaptive mesh shows the advantage in simulating the non-linear behavior of the tilt mode.

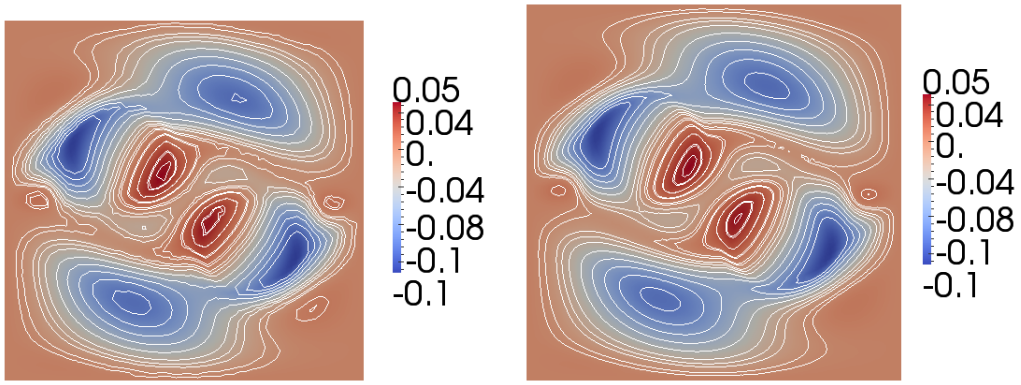


(a) t=1 (b) t=2

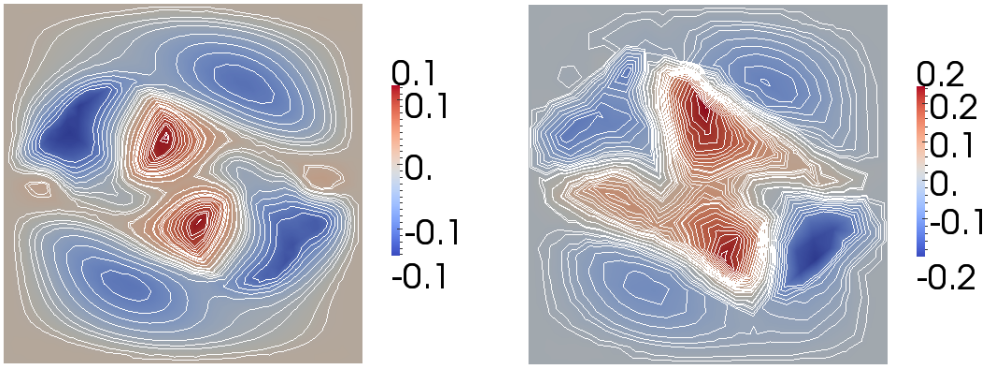
(c) t=3 (d) t=4

(e) t=6 (f) t=8

Figure 7.3: Toroidal current density on the adapted meshes at t=1, 2, 3, 4, 6 and 8. The number of mesh nodes are 200, 326, 664, 737, 665 and 675 respectively (also see the second row of Figure 7.6).

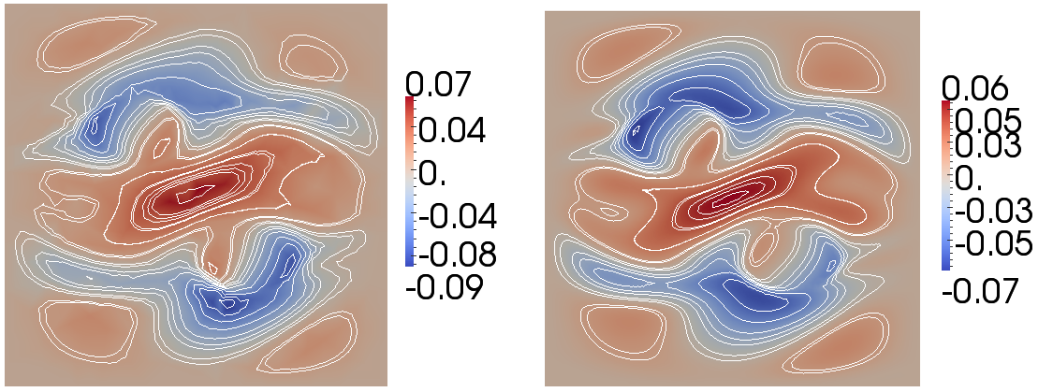


(a) adapted 665 nodes (b) uniform 6219 nodes



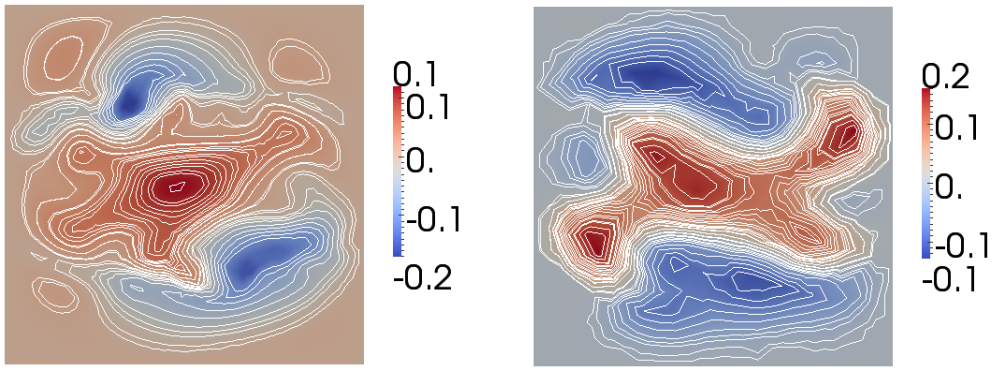
(c) uniform 1544 nodes (d) uniform 409 nodes

Figure 7.4: *U* **field at t=6 on the adapted mesh (mesh e in Figure 7.3) and uniformly refined meshes (the contour plot shows the structure of the streamline).**



(a) adapted 674 nodes

(b) uniform 6219 nodes



(c) uniform 1544 nodes

(d) uniform 409 nodes

Figure 7.5: *U* **field at t=8 on the adapted mesh (mesh f in Figure 7.3) and uniformly refined meshes (the contour plot shows the structure of the streamline).**

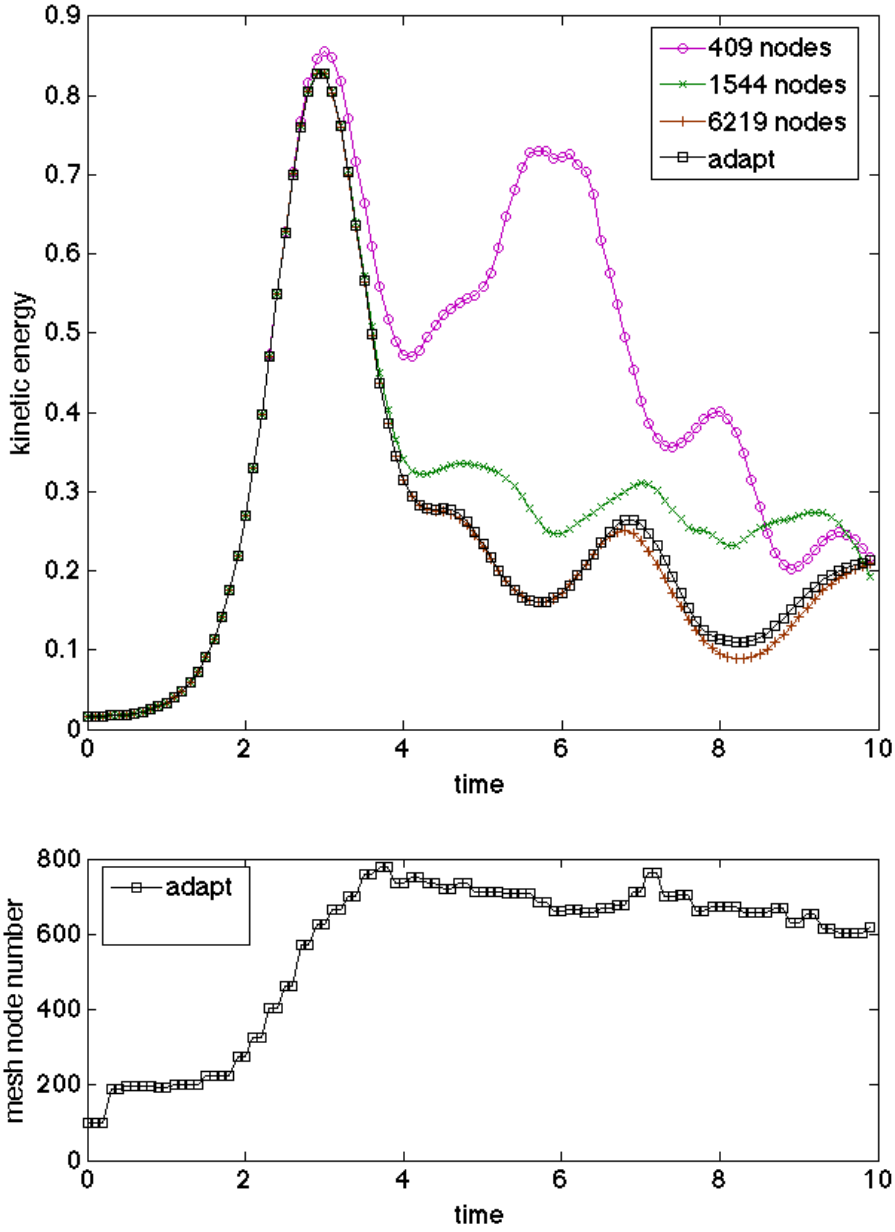


Figure 7.6: Top: kinetic energy on the adapted mesh (Figure 7.3) and the uniform meshes with 409, 1544, 6219 nodes; bottom: the number of mesh nodes in the adapted mesh at each step. The minimum mesh size in the adapted meshes are the same as the mesh with 6219 nodes. The plots are marked every two steps. The mesh is adapted every four steps.

7.3 Double Tearing Mode

The tearing mode [80] is a type of MHD instability in the plasmas due to the radical gradient of the toroidal current density in the equilibrium and the finite electrical resistivity [4]. Resistive layers in the form of the localized physics properties, such as the perturbed current (Figure 7.9.a), are developed at the resonant surfaces in the plasmas [4].

The physics of the tearing mode over the domain are characterized by two sets of equations. The ideal MHD [81] is adequate to describe the plasma behaviors outside the resistive layers. The plasmas in this area are equilibrated by the Lorentz force and the pressure gradient [4] (also see Equation 2.10 by dropping the inertial and η terms). Assume the perturbation takes the form as $e^{i(m\theta - n\varphi)}$, where θ is the poloidal angle, φ is the toroidal angle, i is the imaginary unit, and m and n are the poloidal and toroidal mode numbers, respectively. It is shown that the destabilizing effect from the equilibrium governed by the ideal MHD is proportional to [4]

$$\frac{1}{1-\frac{nq}{m}}, \tag{7.5}$$

where q is the safety factor [4]. It can be seen that the destabilizing term goes to infinity at the places where $\frac{1}{1-\frac{nq}{m}} = 0$ and the MHD model including the electrical resistivity must be used at these places.

The linear instability of the tearing mode with two resistive layers is studied by M3D-C^1. The extended MHD equation with the finite electrical resistivity (see Ohm's law defined by Equation 2.7) is simulated over the whole domain.

The initial condition is obtained by solving the Grad-Shafranov equation which defines an axis-symmetric equilibrium of the ideal MHD [81] by

$$\nabla^2_{GS}\psi = -R^2\frac{dp(\psi)}{d\psi} - \frac{1}{2}\frac{dg^2(\psi)}{d\psi}, \tag{7.6}$$

where ψ is the poloidal magnetic flux field in the equilibrium and $\nabla^2_{GS}\psi = R^2\nabla\cdot\left(\frac{1}{R^2}\nabla\psi\right)$. $\frac{dp(\psi)}{d\psi}$ and $\frac{dg^2(\psi)}{d\psi}$ are defined analytically [3] such that there are two $q = 2$ surfaces in the equilibrium (Figure 7.7) and the profile of the current density is in the form illustrated in Figure 7.8.

The resistive layers are localized at the two $q = 2$ surfaces for the test simulated. From Equation 7.5, the instability is due to the growth of the $(n, m) = (1, 2)$ mode. Without losing the physics feature, the reduced MHD model is applied for the double tearing mode. The perturbed magnetic field and the velocity field are represented by

$$\tilde{\mathbf{B}} = \nabla \times \tilde{\psi} \nabla \varphi, \tag{7.7a}$$

$$\tilde{\mathbf{V}} = R^2 \nabla \tilde{U} \times \nabla \varphi, \tag{7.7b}$$

respectively. Note that $\tilde{\psi}$ and $\tilde{U}$ depend on φ as $\sim e^{in\varphi}$, where $n = 1$ is used in the test studied. Since the solution is also a function of φ, the error indicator needs to be extended to the 3D case. However, given that $\tilde{\psi}$ and $\tilde{U}$ vary smoothly with regard to φ ($\sim e^{i\varphi}$) and the major physics feature happens on the RZ plane, the error indicator that only considers the differential operators on the RZ plane still provides useful information on the spatial discretization error.

The 2D differential operators in the cylindrical coordinate are defined on the RZ plane with the constant φ as:

$\nabla F \equiv \frac{\partial F}{\partial R}\hat{R} + \frac{\partial F}{\partial Z}\hat{Z}$,

$\nabla^2 F \equiv \frac{1}{R}\frac{\partial}{\partial R}\left(R\frac{\partial F}{\partial R}\right) + \frac{\partial^2 F}{\partial Z^2}$,

$\nabla^2_{GS} F \equiv R^2 \nabla \cdot \left(\frac{1}{R^2}\nabla F\right) \equiv \nabla^2 F - \frac{2}{R}\frac{\partial F}{\partial R}$,

$< F, G > \equiv -(\nabla F \times \nabla G) \cdot \hat{\varphi}$.

The error indicator for the linearized reduced MHD model (only consider $\tilde{U}$ and $\tilde{\psi}$, also see Section 3.2) in the cylindrical coordinate is defined as:

$$\epsilon^2_{U\mathcal{K}} = h^4_{\mathcal{K}} ||\mathcal{R}^U_1||^2_{L_2(\mathcal{K})} + \frac{1}{2}\sum_{\Gamma \in \partial\mathcal{K}\setminus\partial\Omega}\left(h^3_{\mathcal{K}} \left\| \left[\mathcal{R}^U_2\right]_\Gamma \right\|^2_{L_2(\Gamma)} + h_{\mathcal{K}} \left\| \left[\mathcal{R}^U_3\right]_\Gamma \right\|^2_{L_2(\Gamma)}\right), \tag{7.8a}$$

$$\epsilon^2_{\psi\mathcal{K}} = h^4_{\mathcal{K}} ||\mathcal{R}^\psi_1||^2_{L_2(\mathcal{K})} + \frac{1}{2}\sum_{\Gamma \in \partial\mathcal{K}\setminus\partial\Omega}\left(h^3_{\mathcal{K}} \left\| \left[\mathcal{R}^\psi_2\right]_\Gamma \right\|^2_{L_2(\Gamma)} + h_{\mathcal{K}} \left\| \left[\mathcal{R}^\psi_3\right]_\Gamma \right\|^2_{L_2(\Gamma)}\right), \tag{7.8b}$$

where

$$\mathcal{R}_1^U = \frac{1}{\delta t}R^2\nabla^2(\tilde{U}_h^{n+1}-\tilde{U}_h^n)+\frac{1}{2}\left\{\begin{array}{c}-R^4<\nabla^2\tilde{U}_h^n,U_h^0>-R^2<\nabla^2U_h^0,\tilde{U}_h^n>\\ +<\nabla^2\tilde{\psi}_h^n,\psi_h^0>+<\nabla^2\psi_h^0,\tilde{\psi}_h^n>\end{array}\right\}$$
$$-\mu R^2\nabla^4\tilde{U}_h^{n+1}, \tag{7.9a}$$

$$\mathcal{R}_1^\psi = \frac{1}{\delta t}\nabla^2(\tilde{\psi}_h^{n+1}-\tilde{\psi}_h^n)-\frac{1}{2}R^2\left\{\nabla^2<\tilde{\psi}_h^n,U_h^0>+\nabla^2<\psi_h^0,\tilde{U}_h^n>\right\}$$
$$-\eta\nabla^4\tilde{\psi}_h^{n+1}, \tag{7.9b}$$

$$\mathcal{R}_2^U = \frac{1}{2}\left\{\begin{array}{l}R^4\nabla^2\tilde{U}_h^n\left(\frac{\partial U_h^0}{\partial R}cos(\gamma)+\frac{\partial U_h^0}{\partial Z}sin(\gamma)\right)\\ +R^4\nabla^2U_h^0\left(\frac{\partial\tilde{U}_h^n}{\partial R}cos(\gamma)+\frac{\partial\tilde{U}_h^n}{\partial Z}sin(\gamma)\right)\\ -\nabla^2_{GS}\tilde{\psi}_h^n\left(\frac{\partial\psi_h^0}{\partial R}cos(\gamma)+\frac{\partial\psi_h^0}{\partial Z}sin(\gamma)\right)\\ -\nabla^2_{GS}\psi_h^0\left(\frac{\partial\tilde{\psi}_h^n}{\partial R}cos(\gamma)+\frac{\partial\tilde{\psi}_h^n}{\partial Z}sin(\gamma)\right)\end{array}\right\}+\mu R^2\frac{\partial\nabla^2\tilde{U}_h^{n+1}}{\partial n}, \tag{7.9c}$$

$$\mathcal{R}_2^\psi = \frac{1}{2}R^2\left\{\frac{\partial}{\partial n}<\tilde{\psi}_h^n,U_h^0>+\frac{\partial}{\partial n}<\psi_h^0,\tilde{U}_h^n>\right\}+\eta\frac{\partial\nabla^2_{GS}\tilde{\psi}_h^{n+1}}{\partial n}, \tag{7.9d}$$

$$\mathcal{R}_3^U = -\mu R^2\nabla^2\tilde{U}_h^{n+1}, \tag{7.9e}$$

$$\mathcal{R}_3^\psi = -\eta\nabla^2_{GS}\tilde{\psi}_h^{n+1}. \tag{7.9f}$$

Note that $(\cdot)^0$ is the equilibrium solution, and $\tilde{(\cdot)}$ is the perturbation from the equilibrium. Assume that the equilibrium solution is smooth compared with the perturbed solution, terms of $(\cdot)^0$ is further eliminated. Also consider that $\mu = 10^{-6} \ll 1$ and $\eta = 10^{-7} \ll 1$ (see the energy norms defined by Equation 3.24), the approximation theory in H^1 space is more suitable for the problem studied (setting $s = r = 2$, $l = 2$ and $m = 0$ for Theorem 1.1 in [30]). The revised error indicator for the linear instability study of the double tearing problem is

$$\epsilon^2_{U\mathcal{K}} = h^2_{\mathcal{K}}||\mathcal{R}_1^U||^2_{L_2(\mathcal{K})}+\frac{1}{2}\sum_{\Gamma\in\partial\mathcal{K}\backslash\partial\Omega}\left(h_{\mathcal{K}}\left\|\left[\mathcal{R}_2^U\right]_\Gamma\right\|^2_{L_2(\Gamma)}+h_{\mathcal{K}}^{-1}\left\|\left[\mathcal{R}_3^U\right]_\Gamma\right\|^2_{L_2(\Gamma)}\right), \tag{7.10a}$$

$$\epsilon^2_{\psi\mathcal{K}} = h^2_{\mathcal{K}}||\mathcal{R}_1^\psi||^2_{L_2(\mathcal{K})}+\frac{1}{2}\sum_{\Gamma\in\partial\mathcal{K}\backslash\partial\Omega}\left(h_{\mathcal{K}}\left\|\left[\mathcal{R}_2^\psi\right]_\Gamma\right\|^2_{L_2(\Gamma)}+h_{\mathcal{K}}^{-1}\left\|\left[\mathcal{R}_3^\psi\right]_\Gamma\right\|^2_{L_2(\Gamma)}\right), \tag{7.10b}$$

where

$$\mathcal{R}_1^U = \frac{1}{\delta t} R^2 \nabla^2 (\tilde{U}_h^{n+1} - \tilde{U}_h^n) - \mu R^2 \nabla^4 \tilde{U}_h^{n+1}, \tag{7.11a}$$

$$\mathcal{R}_1^\psi = \frac{1}{\delta t} \nabla^2 (\tilde{\psi}_h^{n+1} - \tilde{\psi}_h^n) - \eta \nabla^4 \tilde{\psi}_h^{n+1}, \tag{7.11b}$$

$$\mathcal{R}_2^U = \mu R^2 \frac{\partial \nabla^2 \tilde{U}_h^{n+1}}{\partial n}, \tag{7.11c}$$

$$\mathcal{R}_2^\psi = \eta \frac{\partial \nabla_{GS}^2 \tilde{\psi}_h^{n+1}}{\partial n}, \tag{7.11d}$$

$$\mathcal{R}_3^U = -\mu R^2 \nabla^2 \tilde{U}_h^{n+1}, \tag{7.11e}$$

$$\mathcal{R}_3^\psi = -\eta \nabla_{GS}^2 \tilde{\psi}_h^{n+1}. \tag{7.11f}$$

Figure 7.10 illustrates the initial and adapted meshes. The mesh is iteratively adapted every 300 time steps ($dt = 20$, $t = 6000$), and the simulation loop goes back to the step of calculating the equilibrium on the new mesh (see Equation 7.6). It can be seen that the adapted mesh has the finest mesh elements at the two resistive layers in the double tearing mode. Figure 7.11 illustrates the change of the kinetic energy (Equation 7.4) on the uniformly refined meshes and the adapted mesh. The smallest element in the adapted mesh is set to be the same as that in the uniform mesh with 96,702 nodes. It shows that the adapted mesh captures the growth of the eigen modes.

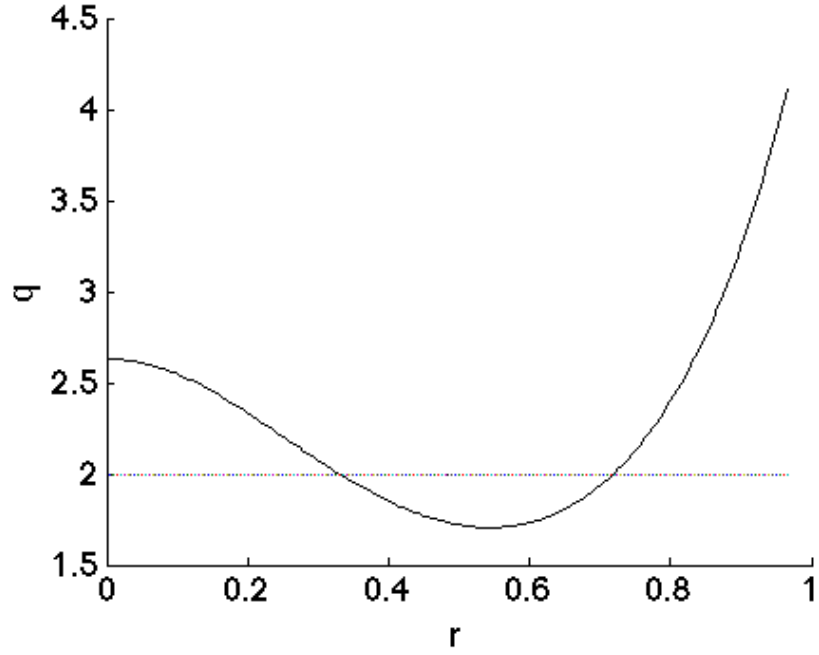


Figure 7.7: Profile of the safety factor (q) in the equilibrium. r is the minor radius (see Figure 2.1).

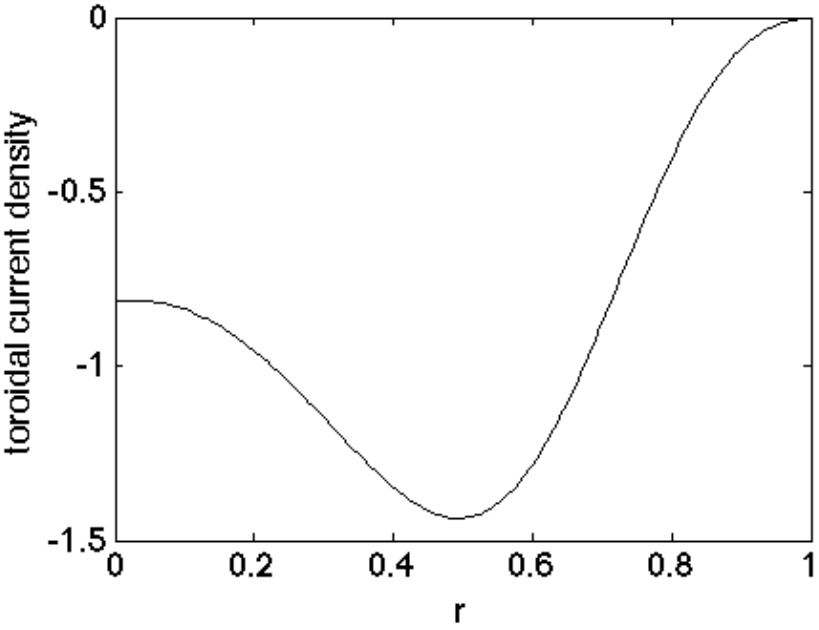
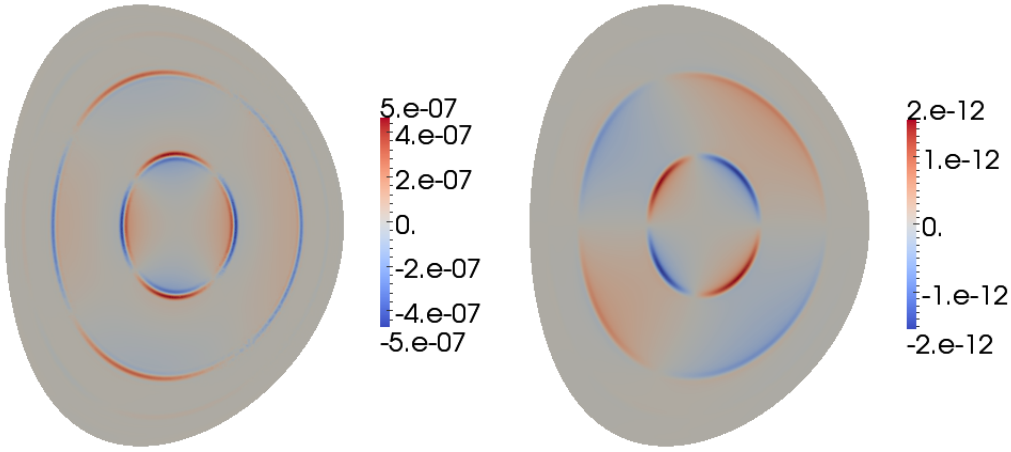
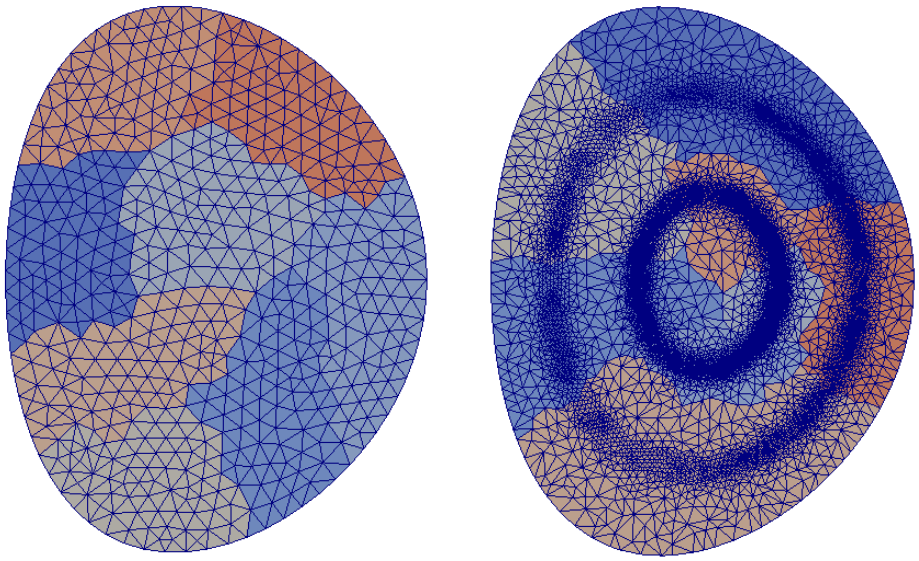


Figure 7.8: Profile of the toroidal current density in the equilibrium over the minor radius, r.



(a) $\tilde{j}$ (b) $\tilde{U}$

Figure 7.9: Perturbed toroidal current density ($\tilde{j}$) and the perturbed U component of the velocity field ($\tilde{U}$) developed on a uniform mesh with 96,703 nodes in the double tearing mode ($\eta = 10^{-7}$).



(a) initial (579 nodes) (b) adapted mesh (10,396 nodes)

Figure 7.10: Initial mesh and the adapted mesh by eight processes for the double tearing mode. The major radius, R (see Figure 2.1), ranges in $[2.2, 4.2]$. The meshes are colored by the process ranks.

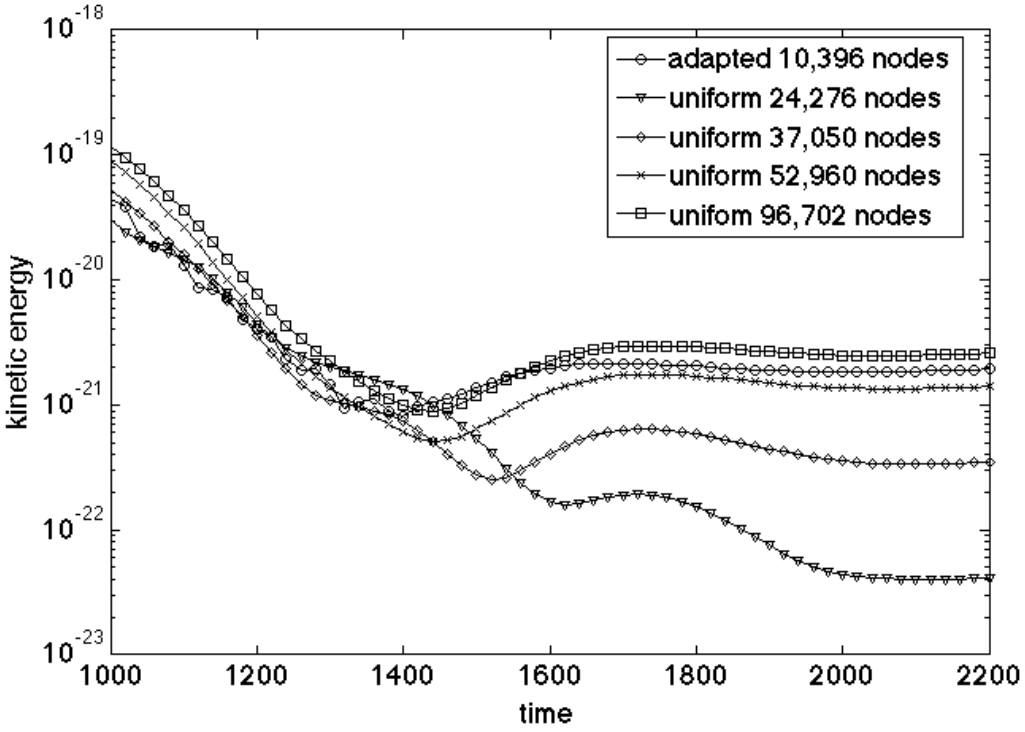
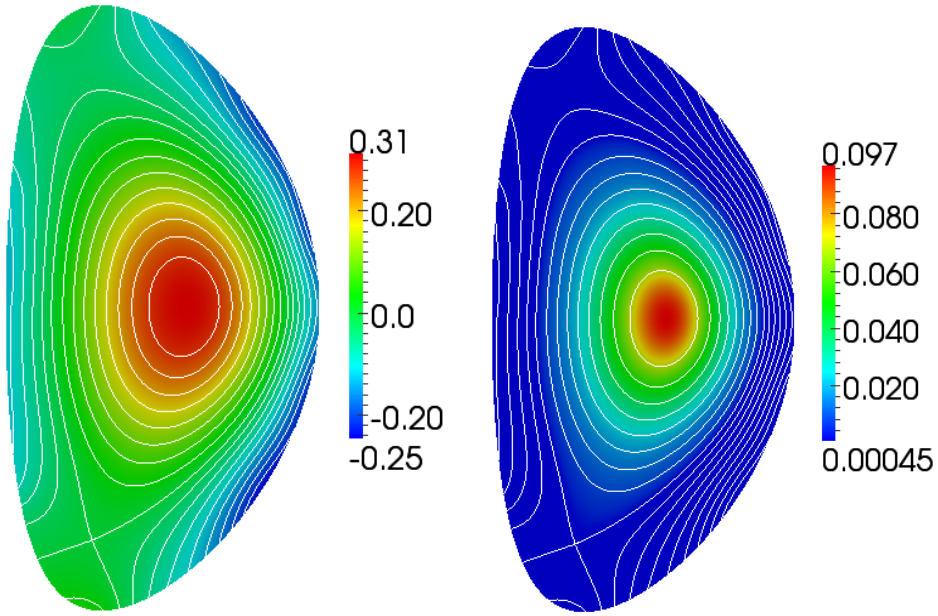


Figure 7.11: Change of kinetic energy (E_k in Equation 7.4) on the adapted mesh with 10,396 nodes (Figure 7.10) and the uniform meshes with 24,276, 37,050, 52,960, and 96,703 nodes.

7.4 Edge Localized Modes

The edge localized mode (ELM) is a type of MHD instability happening at the plasma edge [82, 4]. An example on the linear instability of the ELM is studied by M3D-C^1 on the adaptive mesh.

Similar to the process of the linear instability example discussed by Section 7.3, the initial equilibrium of the plasma is obtained by solving the the Grad-Shafranov equation defined by Equation 7.6. The profiles of $p(\psi)$ and $g(\psi)$ are defined by the spline-fitted data from [67]. Figure 7.12 illustrates the ψ and pressure fields in the equilibrium state.



(a) ψ (b) pressure

Figure 7.12: Initial equilibrium of the ψ and pressure fields. The lines show the structure of the magnetic flux surfaces (also see Section 5.2.3.2).

In the linear instability simulation of ELM, the perturbed velocity and magnetic field are represented by the full model defined by Equation 2.12 [6]. Similar to Section 7.3, the perturbation depends on φ in the form of $e^{in\varphi}$, where $n = 8$ for the test studied. The kinetic energy of the system ($\frac{1}{2}\rho V^2$) increases at the rate $\sim e^{\gamma t}$ after the equilibrium is perturbed, where γ is the growth rate.

Compared with the reduced model applied in Section 7.3, the full model includes formulation of the variables, $[\omega, \chi, f]$, for the velocity and magnetic fields in addition to ψ and U. The estimated error contributed by the jump discontinuity derived for the reduced model (Equation 7.11) is used as the indicator of the mesh adaptation.

Figure 7.13 illustrates the initial mesh and the adapted mesh by eight processes. The mesh is iteratively adapted when the kinetic energy reaches to 0.05. Figure 7.14 and Figure 7.15 plot the toroidal current and poloidal magnetic flux fields on the adapted mesh.

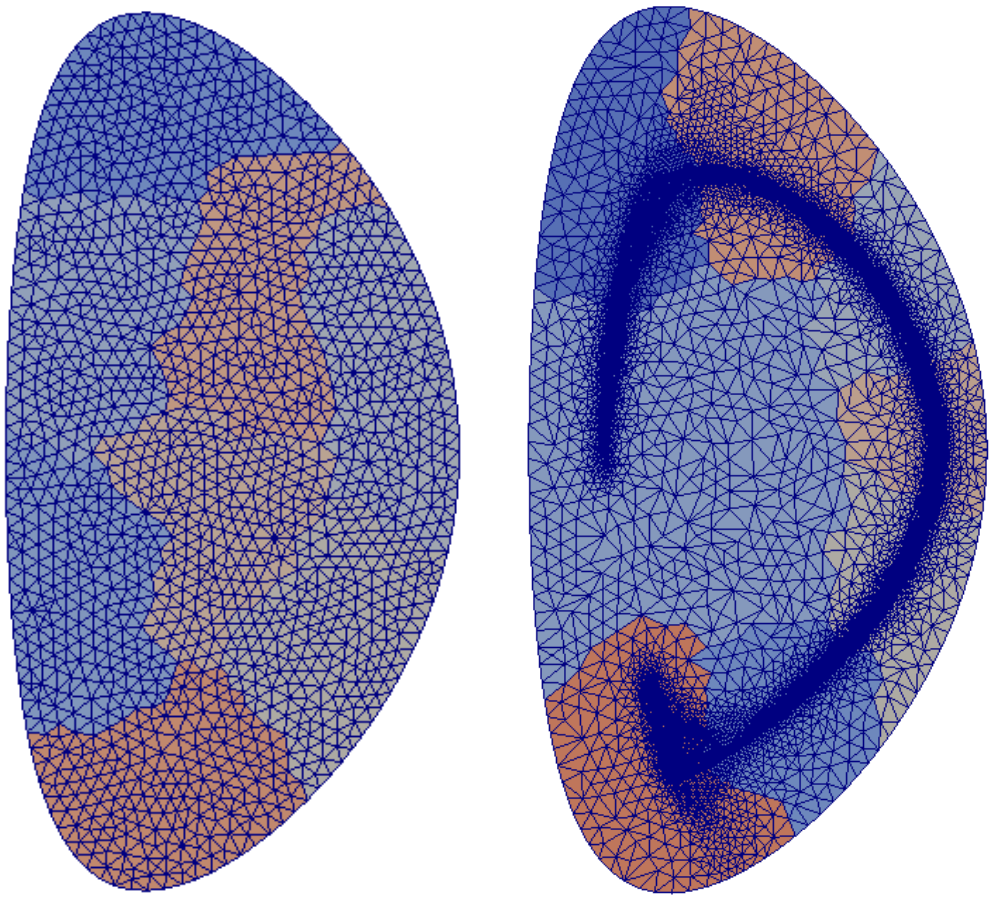
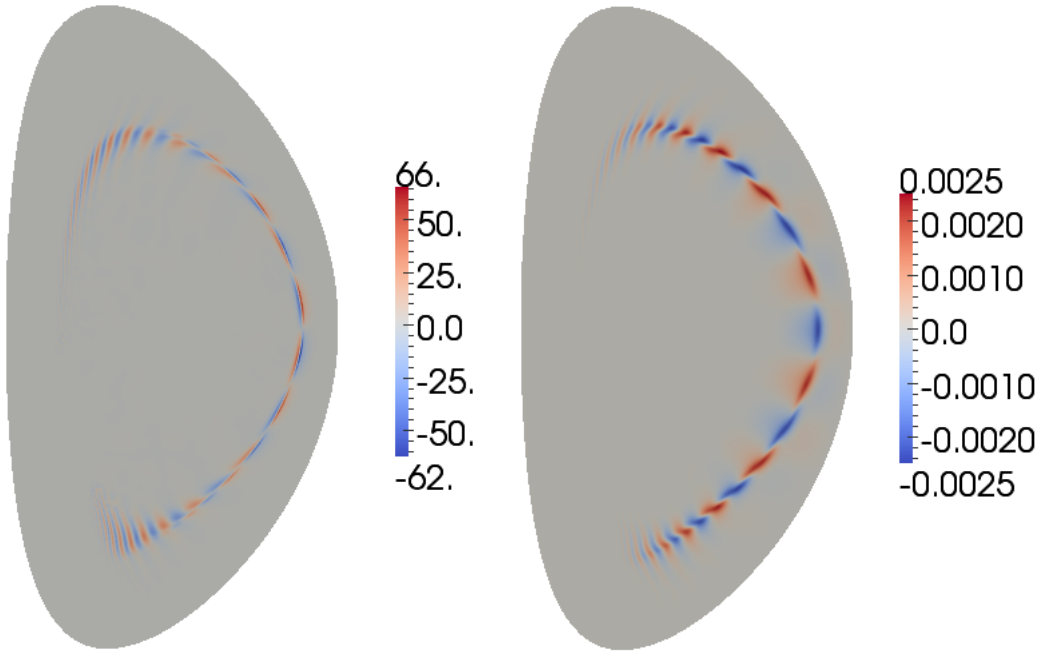


Figure 7.13: Initial mesh (1,469 nodes) and the adapted mesh (11,941 nodes) for edge localized mode by eight-process run.

Table 7.1 compares the growth rates of the kinetic energy on the uniform meshes and the adapted mesh. It shows that the result on the adapted mesh with 11,941 nodes agrees with the result on the uniform mesh with 53,564 nodes.

Table 7.1: Growth rate of the kinetic energy ($\sim e^{\gamma t}$) on the uniform meshes with 13,437, 23,399, and 53,564 nodes and the adapted mesh with 11,941 nodes.

mesh	uniform	uniform	uniform	adapted
number of nodes	13,437	23,399	53,564	11,941
growth rate (γ)	0.1363	0.1425	0.1473	0.1474



(a) $\tilde{j}$ (b) $\tilde{U}$

Figure 7.14: Toroidal current ($\tilde{j}$) and $\tilde{U}$ fields on the adapted mesh in Figure 7.13.

7.5 2D Simulation to 3D Simulation

An example demonstrating the capability of switching the simulation loop from the 2D mesh to the 3D mesh (see Section 4.1.3) is presented. The initial axisymmetric solution is calculated on the 2D mesh by solving the Grad-Shafranov equation defined by Equation 7.6. Figure 7.16 illustrates the profiles of the safety factor (q), the toroidal current density (j), and the pressure (p) over the minor radius (r) in the axisymmetric equilibrium by the 2D simulation. The axisymmetric solution on the 2D mesh is mapped to the 3D mesh by the method defined by Section 4.5.1 (Figure 7.17). The non-axisymmetric instability is developed on the

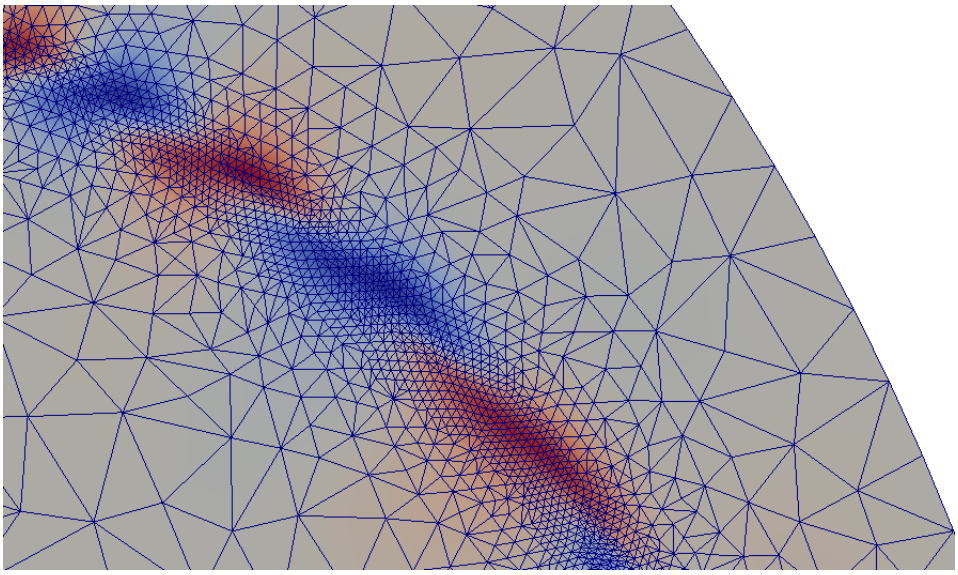
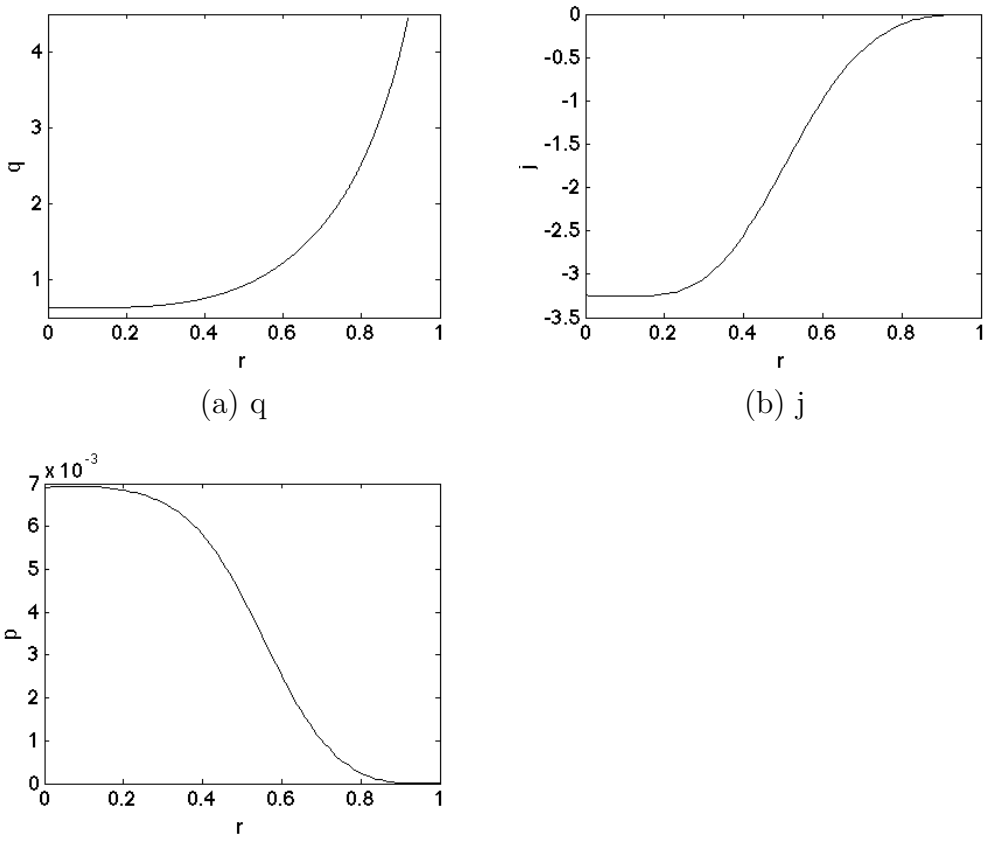


Figure 7.15: Close-up view of $\tilde{U}$ on the adapted mesh in Figure 7.13.

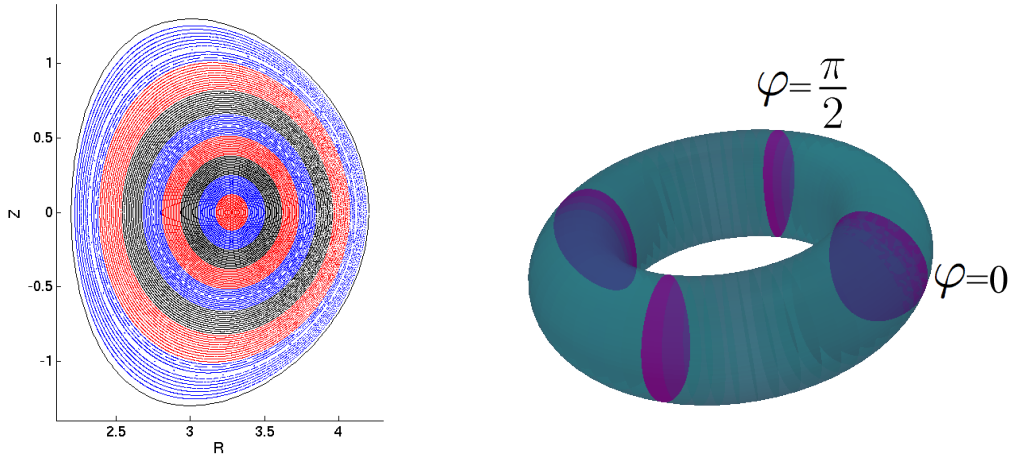
3D mesh. Figure 7.18 shows the structure of the magnetic islands on the planes at $\varphi = 0$ and $\frac{1}{2}\pi$ obtained from the 3D simulation, respectively.



(a) q (b) j

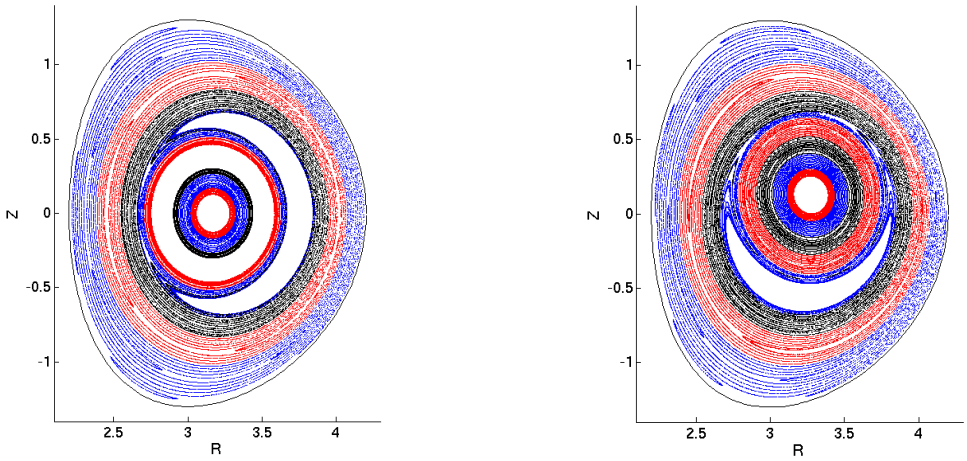
(c) p

Figure 7.16: Profiles of the safety factor (q), the toroidal current density (j), and the pressure (p) over the minor radius (r) in the axisymmetric equilibrium by the 2D simulation.



(a) axisymmetric solution (b) 3D mesh

Figure 7.17: 3D simulation that uses the axisymmetric solution calculated on the 2D mesh. The 2D mesh is a uniform mesh with 1159 nodes. There are 8 planes in the 3D mesh. Sub-figure a is the Poincare plot of the magnetic flux surfaces.



(a) $\varphi = 0$ (b) $\varphi = \frac{1}{2}\pi$

Figure 7.18: Poincare plot of the magnetic flux surfaces on the planes with $\varphi = 0, \frac{1}{2}\pi$.

CHAPTER 8
Conclusion and Future Work

8.1 Conclusion

This thesis studies the methods and associated software for the automatic and adaptive simulations of plasmas physics on high-performance parallel computers. A parallel adaptive infrastructure for the magnetically confined fusion plasma simulations is developed and applied to M3D-C^1.

In Chapter 2, the extended MHD equations and the finite element formulation were reviewed. The fourth-order PDE from the reduced MHD model under large-aspect-ratio approximation in the tokamak and its weak formulation were discussed specifically.

In Chapter 3, an explicit *a posteriori* error estimator was derived for the model problem discussed in Chapter 2. The error estimator provides an indication of the numerical error from the spatial discretization (assuming the temporal discretization error is controlled to be small). The mesh-dependent norm of the residual in the strong form is calculated and it is shown to provide an upper-bound of the spatial discretization error in the energy norm. The error estimator is applied to the problem of the Hartman boundary layer. Although the global effectivity indices calculated show that the true error is over estimated, the global effectivity indices are relatively steady as the mesh is adaptively refined, and the distribution of the element level estimated error is consistent with the true error measured by the energy norm.

In Chapter 4, an overview of the parallel adaptive simulation loop in M3D-C^1 and the corresponding software tools were presented. The simulation loops on the 2D mesh and the 3D mesh, and the loop from 2D mesh to 3D mesh were discussed, respectively. The initial mesh on the tokamak cross-section is generated by the Simmetrix mesh generation tool [43]. PUMI [44] is applied as the unstructured mesh infrastructure that provides the interface to the geometric model of the tokamak and manages the distributed mesh. MeshAdapt [14] is used as the tool to adaptively modify the mesh. SuperLU [45] and PETSc [48] are used by M3D-C^1 to solve the

algebraic systems by the direct and iterative methods, respectively. Mesh adjacency information is used to reduce the memory usage during setting up the global discrete system.

In Chapter 5, a tokamak mesh definition procedure for the fusion plasma codes, M3D-C^1 and XGC1 was presented. The procedure employs a geometric model definition of the domain that represents both the physical and physics components that must be reflected in the resulting mesh. The needed mesh control information is easily specified in terms of the geometric model. The mesh generation procedure is component-based and it satisfies the constraints of the simulation procedures and creates well controlled graded meshes at the same time.

In Chapter 6, a procedure to improve the numerical conditioning of the resulting matrix systems in M3D-C^1 was discussed. The physics that couple the multi-scale components [6] and the fourth-order PDE solved that requires the usage of the C^1 finite elements lead to the ill-conditioning of the linear systems in M3D-C^1. A method that defines a new set of the shape functions is introduced. Three terms are introduced to rescale the shape functions. The first term is to account for the "element size" and "element size squared" introduced by using the first-order and second-order derivatives as degrees of freedom in the finite element discretization. The second is the physics-based scaling of the contributions to the multicomponent set of equations based on the grouping of eigenvalues associated with the components. The third accounts for integration over elements if the mesh size varies over the domain. The C^1 finite element is regularized and the procedure results in a linear system with a smaller condition number.

In Chapter 7, examples of the parallel adaptive simulation in M3D-C^1 were presented.

8.2 Future Work

The explicit error estimator derived in the present work overestimates the error with respect to the global effectivity indices. The effectiveness can be improved by approximating the magnitude of the multiplicative constant used in the error estimator (C in Equation 3.25). The constant depends on the specific set of the

shape functions, the shape of the mesh element and the parameters in the energy norm (see Equation 3.24). The alternative way, perhaps more desired with respect to the global effectivity indices, is applying the implicit error estimator that solves the appropriate sub-domain residual problem [30, 42]. One challenge of defining such an implicit error estimator lies in constructing a proper subspace to approximate the sub-domain residual problem, given the fact that the original problem is solved by the shape functions of fifth-order polynomials.

It is possible to extend the current error estimator for the anisotropic mesh adaptation, if the directional property of the solutions from the problems with the anisotropic transport coefficients, such as the gyro-viscosity [8], is considered.

It is also desired to estimate the temporal discretization error explicitly such that both the temporal and the spatial discretization errors can be adaptively controlled.

The present work applies a simple method to transfer the solutions associated with local mesh modifications. The field values and its derivatives up to second order are interpolated on the original mesh during the split operation and the DOF values associated with the new vertex are assigned to be the area-weighted average values interpolated over the adjacent elements. The DOF values are kept unchanged during the swap and collapse operations. It is desired to investigate more sophisticated solution transfer methods such as the superconvergent patch recovery (SPR) [55] and compare with the currently applied method to see how the solution transfer can affect the accuracy of the adaptive simulation.

A more scalable and efficient iterative solving method for the 3D problem in M3D-C^1 that combines the general preconditioning method [76, 77] with the fast element routines developed in Chapter 6 is still to be investigated in the future.

REFERENCES

[1] W. Park *et al.*, "Plasma simulation studies using multilevel physics models," *Phys. Plasmas*, vol. 6, no. 5, pp. 1796–1803, Jan. 1999.

[2] S. C. Jardin, *Computational Methods in Plasma Physics*. Boca Raton, FL, USA: CRC Press, 2010.

[3] S. C. Jardin, "A triangular finite element with first-derivative continuity applied to fusion MHD applications," *J. Comput. Phys.*, vol. 200, no. 1, pp. 133–152, Oct. 2004.

[4] J. Wesson, *Tokamaks*. Oxford, UK: Oxford Univ. Press, 2011.

[5] Princeton Plasma Physics Lab, "National Spherical Torus Experiment (NSTX)," [Online]. Available: http://www.pppl.gov/nstx. Accessed on: 17 Aug. 2015.

[6] S. C. Jardin, N. M. Ferraro, J. Breslau, and J. Chen, "Multiple timescale calculations of sawteeth and other global macroscopic dynamics of tokamak plasmas," *Comput. Sci. Discov.*, vol. 5, no. 1, May 2012, Art.ID. 014002.

[7] S. C. Jardin, J. Breslau, and N. M. Ferraro, "A high-order implicit finite element method for integrating the two-fluid magnetohydrodynamic equations in two dimensions," *J. Comput. Phys.*, vol. 226, no. 2, pp. 2146–2174, Oct. 2007.

[8] N. M. Ferraro and S. C. Jardin, "Calculations of two-fluid magnetohydrodynamic axisymmetric steady-states," *J. Comput. Phys.*, vol. 228, no. 20, pp. 7742–7770, Nov. 2009.

[9] S. C. Jardin, "Review of implicit methods for the magnetohydrodynamic description of magnetically confined plasmas," *J. Comput. Phys.*, vol. 231, no. 3, pp. 822–838, Feb. 2012.

[10] N. M. Ferraro *et al.*, "Fluid modeling of fusion plasmas with M3D-C1," Sci. Discov. through Advanced Comput., Tech. Rep., Jul. 2011, [Online]. Available: http://www.mcs.anl.gov/uploads/cels/papers/scidac11/final/ferraro_nathaniel.pdf. Accessed on: 17 Aug. 2015.

[11] C. Chang *et al.*, "Compressed ion temperature gradient turbulence in diverted tokamak edge," *Phys. Plasmas*, vol. 16, no. 5, Feb. 2009, Art.ID. 056108.

[12] C. Chang, S. Ku, and H. Weitzner, "Numerical study of neoclassical plasma pedestal in a tokamak geometry," *Phys. Plasmas*, vol. 11, no. 5, pp. 2649–2667, Apr. 2004.

[13] M. F. Adams *et al.*, "Scaling to 150k cores: Recent algorithm and performance engineering developments enabling XGC1 to run at scale," *J. Phys. Conf. Ser.*, vol. 180, no. 1, Jun. 2009, Art.ID. 012036.

[14] M. S. Shephard, C. W. Smith, E. S. Seol, and O. Sahni, "Methods and tools for parallel anisotropic mesh adaptation and analysis," *Proc. ADMOS*, pp. 619–631, Jun. 2013.

[15] K. J. Weiler, "Topological structures for geometric modeling," Ph.D. dissertation, Dept. Comput. and Syst. Eng., Rensselaer Polytechnic Inst., Troy, NY, 1986.

[16] M. S. Shephard, "Meshing environment for geometry-based analysis," *Int. J. Num. Methods Eng.*, vol. 47, no. 1-3, pp. 169–190, Jan. 2000.

[17] F. Zhang *et al.*, "Mesh generation for confined fusion plasma simulation," *Eng. with Comput.*, to be published.

[18] R. L. Panton, *Incompressible Flow*. Hoboken, NJ, USA: Wiley, 2006.

[19] J. D. Jackson, *Classical Electrodynamics*. Hoboken, NJ, USA: Wiley, 1962.

[20] J. A. Bittencourt, *Fundamentals of Plasma Physics*. New York, NY, USA: Springer, 2013.

[21] T. J. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Mineola, NY, USA: Dover Publications, 2012.

[22] R. Verfürth, *A Review of a Posteriori Error Estimation and Adaptive Mesh-refinement Techniques*. Hoboken, NJ, USA: Wiley, 1996.

[23] R. Becker and R. Rannacher, "An optimal control approach to a posteriori error estimation in finite element methods," *Acta Numerica*, vol. 10, no. 1, pp. 1–102, May 2001.

[24] O. C. Zienkiewicz and J. Z. Zhu, "The superconvergent patch recovery and a posteriori error estimates. part 2: Error estimates and adaptivity," *Int. J. Num. Methods Eng.*, vol. 33, no. 7, pp. 1365–1382, May 1992.

[25] H. Strauss and D. Longcope, "An adaptive finite element method for magnetohydrodynamics," *J. Comput. Phys.*, vol. 147, no. 2, pp. 318–336, Dec. 1998.

[26] K. G. Powell, "A tree-based adaptive scheme for solution of the equations of gas dynamics and magnetohydrodynamics," *Appl. Num. Math.*, vol. 14, no. 1, pp. 327–352, Apr. 1994.

[27] K. G. Powell, P. L. Roe, T. J. Linde, T. I. Gombosi, and D. L. De Zeeuw, "A solution-adaptive upwind scheme for ideal magnetohydrodynamics," *J. Comput. Phys.*, vol. 154, no. 2, pp. 284–309, Sep. 1999.

[28] B. Philip, M. Pernice, and L. Chacón, "Solution of reduced resistive magnetohydrodynamics using implicit adaptive mesh refinement," in *Domain Decomposition Methods in Science and Engineering XVI.* New York, NY, USA: Springer, 2007, pp. 723–729.

[29] S. C. Jardin, "The nonlinear M3D-C1 code with application to disruptive beta limits in NSTX," 2013, [Online]. Available: http://w3.pppl.gov/cemm/M3D-C1.pdf. Accessed on: 17 Aug. 2015.

[30] M. Ainsworth and J. T. Oden, "A posteriori error estimation in finite element analysis," *Comput. Methods Appl. Mechanics Eng.*, vol. 142, no. 1, pp. 1–88, Mar. 1997.

[31] O. C. Zienkiewicz and J. Z. Zhu, "The superconvergent patch recovery and a posteriori error estimates. part 1: The recovery technique," *Int. J. Num. Methods Eng.*, vol. 33, no. 7, pp. 1331–1364, May 1992.

[32] R. Keppens, M. Nool, G. Tóth, and J. Goedbloed, "Adaptive mesh refinement for conservative systems: multi-dimensional efficiency evaluation," *Comput. Phys. Comm.*, vol. 153, no. 3, pp. 317–339, Jul. 2003.

[33] A. Van Dam and P. Zegeling, "A robust moving mesh finite volume method applied to 1D hyperbolic conservation laws from magnetohydrodynamics," *J. Comput. Phys.*, vol. 216, no. 2, pp. 526–546, Aug. 2006.

[34] R. Verfürth, "A posteriori error estimators for the stokes equations," *Numerische Mathematik*, vol. 55, no. 3, pp. 309–325, May 1989.

[35] R. Verfürth, "A posteriori error estimators and adaptive mesh-refinement for a mixed finite element discretization of the Navier-Stokes equations," in *Numerical Treatment of the Navier-Stokes Equations.* New York, NY, USA: Springer, 1990, pp. 145–152.

[36] R. Verfürth, "A posteriori error estimation and adaptive mesh-refinement techniques," *J. Comput. Appl. Math.*, vol. 50, no. 1, pp. 67–83, May 1994.

[37] S. Prudhomme and J. Oden, "A posteriori error estimation and error control for finite element approximations of the time-dependent Navier-Stokes equations," *Finite Elements Analy. Des.*, vol. 33, no. 4, pp. 247–262, Nov. 1999.

[38] K. Segeth, "A comparison of a posteriori error estimates for biharmonic problems solved by the FEM," *J. Comput. Appl. Math.*, vol. 236, no. 18, pp. 4788–4797, Dec. 2012.

[39] R. Verfürth, "A posteriori error estimators for convection-diffusion equations," *Numerische Mathematik*, vol. 80, no. 4, pp. 641–663, Oct. 1998.

[40] P. Clément, "Approximation by finite element functions using local regularization," *RAIRO Anal. Num.*, vol. 9, no. R2, pp. 77–84, Aug. 1975.

[41] P. A. Davidson, *An Introduction to Magnetohydrodynamics*. Cambridge, UK: Cambridge Univ. Press, 2001, vol. 25.

[42] M. Ainsworth and J. T. Oden, *A Posteriori Error Estimation in Finite Element Analysis*. Hoboken, NJ, USA: Wiley, 2011.

[43] Simmetrix Inc., "Simmetrix Inc. - Mesh Generation, Geometry Access," [Online]. Available: http://www.simmetrix.com/. Accessed on: 17 Aug. 2015.

[44] D. A. Ibanez, E. S. Seol, C. W. Smith, and M. S. Shephard, "PUMI: Parallel unstructured mesh infrastructure," *ACM Trans. Math. Softw.*, submitted for publication.

[45] X. Li *et al.*, "SuperLU Users' Guide," Lawrence Berkeley National Laboratory, Tech. Rep. LBNL-44289, Sep. 1999, [Online]. Available: http://crd.lbl.gov/~xiaoye/SuperLU/. Accessed on: 17 Aug. 2015.

[46] X. Li, M. S. Shephard, and M. W. Beall, "3D anisotropic mesh adaptation by mesh modification," *Comput. Methods Appl. Mechanics Eng.*, vol. 194, no. 48, pp. 4915–4950, Nov. 2005.

[47] O. Sahni, "Automated adaptive viscous flow simulations," Ph.D. dissertation, Dept. Mech. Aerospace and Nucl. Eng., Rensselaer Polytechnic Inst., Troy, NY, 2007.

[48] S. Balay *et al.*, "PETSc Web page," 2013, [Online]. Available: http://www.mcs.anl.gov/petsc. Accessed on: 17 Aug. 2015.

[49] M. Zhou, O. Sahni, M. S. Shephard, C. D. Carothers, and K. E. Jansen, "Adjacency-based data reordering algorithm for acceleration of finite element computations," *Sci. Program.*, vol. 18, no. 2, pp. 107–123, Apr. 2010.

[50] O. Sahni, C. D. Carothers, M. S. Shephard, and K. E. Jansen, "Strong scaling analysis of a parallel, unstructured, implicit solver and the influence of the operating system interference," *Sci. Program.*, vol. 17, no. 3, pp. 261–274, Dec. 2009.

[51] S. Balay *et al.*, "PETSc users manual," Argonne National Laboratory, Tech. Rep. ANL-95/11 - Revision 3.5, 2014, [Online]. Available: http://www.mcs.anl.gov/petsc, Accessed on: 17 Aug. 2015.

[52] Valgrind, "Massif: a heap profiler," [Online]. Available: http://valgrind.org/docs/manual/ms-manual.html. Accessed on: 17 Aug. 2015.

[53] A. Ovcharenko *et al.*, "Neighborhood communication paradigm to increase scalability in large-scale dynamic scientific applications," *Parallel Comput.*, vol. 38, no. 3, pp. 140–156, Mar. 2012.

[54] National Energy Research Scientific Computing Center, "NERSC computational systems," [Online]. Available: http://www.nersc.gov/users/computational-systems/. Accessed on: 17 Aug. 2015.

[55] J. Wan, "An automated adaptive procedure for 3D metal forming simulations," Ph.D. dissertation, Dept. Mech. Eng., Rensselaer Polytechnic Inst., Troy, NY, 2006.

[56] Princeton Plasma Physics Lab, "Center for Edge Physics Simulation," [Online]. Available: http://epsi.pppl.gov/. Accessed on: 17 Aug. 2015.

[57] S. Ku, C. Chang, and P. Diamond, "Full-f gyrokinetic particle simulation of centrally heated global ITG turbulence from magnetic axis to edge pedestal top in a realistic tokamak geometry," *Nucl. Fusion*, vol. 49, Sep. 2009, Art.ID. 115021.

[58] F. Chen, *Introduction to Plasma Physics and Controlled Fusion. Volume 1: Plasma Physics*. New York, NY, USA: Springer, 1984.

[59] N. M. Ferraro *et al.*, "Resistive wall model in M3D-C1," presented at the Int. Sherwood Fusion Theory Conf., San Diego, CA, USA, Mar. 24-26, 2014.

[60] M. W. Beall, J. Walsh, and M. S. Shephard, "A comparison of techniques for geometry access related to mesh generation," *Eng. with Comput.*, vol. 20, no. 3, pp. 210–221, Sep. 2004.

[61] C. M. Hoffmann, *Geometric and Solid Modeling*. Burlington, MI, USA: Morgan Kaufmann, 1989.

[62] M. S. Shephard and M. K. Georges, "Reliability of automatic 3D mesh generation," *Comput. Methods Appl. Mechanics Eng.*, vol. 101, no. 1, pp. 443–462, Dec. 1992.

[63] M. W. Beall and M. S. Shephard, "A general topology-based mesh data structure," *Int. J. Num. Methods Eng.*, vol. 40, no. 9, pp. 1573–1596, Aug. 1997.

[64] R. Garimella, "Mesh data structure selection for mesh generation and FEA applications," *Int. J. Num. Methods Eng.*, vol. 55, no. 4, pp. 451–478, Jul. 2002.

[65] W. Celes, G. Paulino, and R. Espinha, "A compact adjacency-based topological data structure for finite element mesh representation," *Int. J. Num. Methods Eng.*, vol. 64, no. 11, pp. 1529–1556, Sep. 2005. .

[66] H. Prautzsch, W. Boehm, and M. Paluszny, *Bézier and B-spline Techniques*. New York, NY, USA: Springer, 2002.

[67] General Atomics. EFIT Equilibrium and Reconstruction Fitting Code. [Online]. Available: https://fusion.gat.com/theory/Efit. Accessed on: 17 Aug. 2015.

[68] Princeton Plasma Physics Lab, "PSPLINE help," [Online]. Available: http://w3.pppl.gov/~pshare/help/pspline.htm. Accessed on: 17 Aug. 2015.

[69] X. Li, M. S. Shephard, and M. W. Beall, "Accounting for curved domains in mesh adaptation," *Int. J. Num. Methods Eng.*, vol. 58, no. 2, pp. 247–276, Jul. 2003.

[70] F. Alauzet, X. Li, E. S. Seol, and M. S. Shephard, "Parallel anisotropic 3D mesh adaptation by mesh modification," *Eng. with Comput.*, vol. 21, no. 3, pp. 247–258, Jan. 2006.

[71] C. L. Bottasso, "Anisotropic mesh adaption by metric-driven optimization," *Int. J. Num. Methods Eng.*, vol. 60, no. 3, pp. 597–639, May 2004.

[72] E. S. Seol and M. S. Shephard, "Efficient distributed mesh data structure for parallel automated adaptive analysis," *Eng. with Comput.*, vol. 22, no. 3-4, pp. 197–213, Nov. 2006.

[73] Y. Saad, *Iterative Methods for Sparse Linear Systems*. Philadelphia, PA, USA: SIAM, 2003.

[74] W. W. Hager, "Condition estimates," *SIAM J. Sci. Stat. Comput.*, vol. 5, no. 2, pp. 311–316, Mar. 1984.

[75] R. E. Barnhill and G. Farin, "C1 quintic interpolation over triangles: two explicit representations," *Int. J. Num. Methods Eng.*, vol. 17, no. 12, pp. 1763–1778, Dec. 1981.

[76] J. H. Bramble and X. Zhang, "Multigrid methods for the biharmonic problem discretized by conforming C1 finite elements on nonnested meshes," *Num. Functional Anal. Optimization*, vol. 16, no. 7-8, pp. 835–846, May 1995.

[77] S. Zhang and J. Xu, "Optimal solvers for fourth-order PDEs discretized on unstructured grids," *SIAM J. Num. Anal.*, vol. 52, no. 1, pp. 282–307, Feb. 2014.

[78] E. Oñate and G. Bugeda, "A study of mesh optimality criteria in adaptive finite element analysis," *Eng. Comput.*, vol. 10, no. 4, pp. 307–321, Dec. 1993.

[79] S. Lankalapalli, J. E. Flaherty, M. S. Shephard, and H. Strauss, "An adaptive finite element method for magnetohydrodynamics," *J. Comput. Phys.*, vol. 225, no. 1, pp. 363–381, Jul. 2007.

[80] H. Furth, P. Rutherford, and H. Selberg, "Tearing mode in the cylindrical tokamak," *Phys. Fluids*, vol. 16, no. 7, pp. 1054–1063, May 1973.

[81] J. P. Goedbloed and S. Poedts, *Principles of Magnetohydrodynamics: with Applications to Laboratory and Astrophysical Plasmas*. Cambridge, UK: Cambridge Univ. Press, 2004.

[82] H. Zohm, "Edge localized modes (ELMs)," *Plasma Phys. Controlled Fusion*, vol. 38, no. 2, Feb. 1996, Art.ID. 105.