AN AUTOMATED MODELING, MESHING, AND ADAPTIVE FRAMEWORK FOR TOKAMAK PLASMA SIMULATIONS

Usman Riaz

Submitted in Partial Fullfillment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

Approved by:
Dr. Mark S. Shephard, Chair
Dr. Onkar Sahni
Dr. Lucy T. Zhang
Dr. Fengyan Li

Department of Mechanical, Aerospace and Nuclear Engineering
Rensselaer Polytechnic Institute
Troy, New York

[May 2024]

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

First, I would like to extend my deepest gratitude to my advisor, Prof. Mark S. Shephard, for his continuous support and insightful guidance throughout my doctoral journey. I will always be grateful to him for placing his confidence in me six years ago when I lacked experience in computational engineering. His patience and trust were invaluable during the early days of my PhD, as I was mastering the fundamentals of the subject and other essential skills. Prof. Shephard's extensive knowledge and experience have consistently encouraged me to view problems from different perspectives and find innovative solutions. I greatly appreciate his timely and thorough feedback on my work, which significantly simplified the compilation of this dissertation. Additionally, I would also like to acknowledge my thesis committee members, Prof. Onkar Sahni, Prof. Lucy T. Zhang, and Prof. Fengyan Li for their valuable suggestions that helped shaped this dissertation.

I am grateful to all the members of the Scientific Computation Research Center (SCOREC). I would especially like to extend thanks to Seegyoung Seol for her invaluable support and guidance throughout my time at SCOREC. Her guidance was crucial at every stage of my PhD journey, from compiling codes on different machines to writing technical documents, debugging, and developing new codes. I also wish to acknowledge Morteza Hakimi for his significant contributions to the M3D-C^1 mesh adaptation work. My thanks also go to Cameron Smith, who was consistently helpful. Whenever I encountered technical issues, reaching out to him always resulted in immediate and effective help. Furthermore, I appreciate the support of other members of SCOREC who have contributed to my research in various ways: Chonglin Zhang, Gopan Perumpilly, Jacob Merson, Samiullah Malik, Avinash Moharana, Osama Raisuddin, Dhyanjyoti Nath, and Aditya Joshi.

I am grateful to Prof. Eisung Yoon from Ulsan National Institute of Science and Technology (UNIST), Korea for his guidance on Tokamak Modeling and Meshing Software during the initial phase of my PhD.

I extend my thanks to our collaborators in XGC and M3D-C^1 teams, with special appreciation for Robert Hager and Seung-Hoe Ku from XGC team, and Nathaniel Ferraro, Brendan Lyon, and Stephen Jardin from M3D-C^1 team. Additionally, I would like to acknowledge the computing facilities at the RPI Center for Computational Innovations (CCI), Princeton Research Computing, PPPL, and National Energy Research Scientific Comput-

ing Center (NERSC). I also wish to express my gratitude towards the institutions I have been associated with over the years, including FASTMath SciDAC Institute, Center for High-Fidelity Boundary Plasma Simulation (HBPS), and Center for Tokamak Transient Simulation (CTTS).

This journey would have been impossible without the unwavering support and love of my friends. Over the last six years, I have had the privilege of meeting some exceptional individuals who have made this experience incredibly rewarding. I am profoundly thankful to Ahsan Raza, Asadullah Amin Shah, Atharwa Thigale, Dazy Singh, Kaushik Nallan, Kevin Bhimani, Khurram Malik, Lisa Balkissoon, Mahwish Bhabhi, Noor Bhabhi, Rohail Hassan, Sahil Chaudhary, Surya Karla, Waleed Mansha, and Zaid Bin Tariq for their support throughout my PhD journey. A special mention goes to Hassnain Asgar and Umair Hussain Shah, for their enduring friendship and support in all these years.

Lastly and most importantly, I am extremely grateful for the unwavering support and encouragement from my family throughout my life. I am appreciative of my parents who have always supported my academic ambitions, while the love and support from my brother, sister, and grandmother have been indispensable. This journey would not have been possible without my family being at my side.

ABSTRACT

The accurate representation of a problem domain in terms of a geometric model and its effective discretization into a mesh plays an important role in achieving higher-quality simulation results and better computational efficiency. In this work, an automated modeling and meshing infrastructure with adaptive mesh control is developed to support large-scale fusion plasma simulations. The goal of the presented work is to develop a framework that accurately constructs the tokamak geometric models and discretize these model such that they meet the simulation needs of specific simulation codes. The modeling and meshing procedures for two fusion plasma codes, XGC and M3D-C^1, are presented.

An automated modeling and meshing framework, TOMMS, was developed to support the magnetic field-aligned and one-element deep meshing requirements of XGC. Recent developments to extend this infrastructure to better support the near flux field following meshing requirements of the XGC are presented. The first extension is a procedure to effectively represent a general set of O-point and X-points configurations. The second extension is a procedure to decompose the geometric model that includes the tokamak wall, selected flux curves and the separatrix curves in a manner such that an appropriate set of mesh generation procedures can be applied to each sub-region. The third extension is the application of appropriate mesh generation procedures in each sub-region to create the desired final mesh. Also, the procedures to provide the mesh information needed for the efficient execution of XGC computational operations are presented.

The M3D-C^1 code requires unstructured high-order triangular elements on the 2D poloidal plane for 2D simulations and 3D wedge elements which are constructed from the extrusion of 2D elements in the toroidal direction for 3D simulations. As part of work to extend M3D-C^1 to a wide range of tokamak geometries, a generalized modeling and meshing infrastructure was developed to support the modeling of tokamaks with any configuration of physical features. The goal of this development is to support construction of model geometries with an arbitrary number of model loops and better control on mesh property settings on individual model entities. Moreover, a set of procedures to support a-priori mesh modifications to generate the initial M3D-C^1 meshes with desired resolution are presented. In M3D-C^1 simulations, the plasma equilibrium evolves over time. To effectively simulate those changes in plasma during a simulation, the mesh needs to be adapted dynamically.

A mesh adaptation approach driven by an SPR error estimator is developed. 2.5D mesh adaptation procedure to extend 2D error-based mesh adaptation approach to 3D is also presented.

CHAPTER 1
INTRODUCTION AND ORGANIZATION

1.1 Background and Motivation

The idea of fusion providing a solution to the world's energy problems never seemed as realistic as it does now. The recent achieving of net-positive energy from a fusion ignition at the National Ignition Facility (NIF) is a major scientific breakthrough in fusion research [1]. However, many challenges related to fusion plasma sustainability still need to be addressed before fusion energy becomes practical. Traditionally, sustainable plasma is achieved through magnetic confinement devices like tokamaks and stellarators [2]. The well-established research in tokamaks has led to building the world's largest tokamak, ITER [3], through a multi-nation collaboration. In addition, the commercial fusion energy sector is growing quickly, with over six billion dollars invested as of July 2023 [4].

Key tokamak modeling issues are related to the modeling of possible plasma instabilities. These include the instabilities in the edge plasma region due to the interaction of the hot plasma to the plasma facing components of tokamak and large-scale MHD instabilities. Any such instability in the plasma leads to the deterioration of plasma confinement thus damaging tokamak geometry and causing energy loss. Hence, it is important to understand the physics that leads to such plasma instabilities. Studies of these instabilities are carried out through the application of numerical simulations that require the use of large-scale supercomputers. The advancements in high-performance computing provide an excellent platform to physicists to perform the needed plasma simulations. There is a wide range of codes for the numerical simulations of fusion plasma-related issues. This thesis focuses on two such simulation codes; first is X-point Gyro-kinetic Code (XGC) [5] and second is M3D-C^1 [6] which is an extended magneto-hydrodynamics code. Both these codes use mesh-based numerical methods, and the target of the work presented in this thesis is to provide effective meshing technologies that meet the need of these codes.

XGC is an edge plasma particle-in-cell (PIC) code that is capable of performing turbulence and neoclassical transport simulations. XGC uses a field-aligned mesh that is identical on all the poloidal planes. The use of field-aligned meshes improves the particle tracking along the field line with relatively coarser resolution in the toroidal direction (fewer number of poloidal planes). A non-field aligned mesh would require a very high resolution in the

toroidal direction for a similar level of accuracy [7]. The requirement of field-aligned mesh in XGC led to the development of Tokamak Modeling and Meshing Software (TOMMS) [8] at the Scientific Computation Research Center (SCOREC). The initial set of tools and related results were presented in reference [9]. The work presented in the thesis is the extension of that mesh generation tool to meet the full set of requirements the XGC users have for large-scale simulations addressing new physics questions. The work presented in this thesis is focused on the implementation of methods for the optimized definition of the physics components (critical points, flux curves) and physical components (tokamak wall). Also, the work describes the methods to update the model topology to produce meshes of higher quality in regions with unsatisfactory meshes. Furthermore, the methods to provide the mesh information in a structured and efficient way for effective XGC simulations are discussed in detail.

The M3D-C^1 code is designed to study the large-scale MHD instabilities in tokamaks. The M3D-C^1 uses unstructured higher-order elements on the 2D poloidal plane, which are extruded in a toroidal direction for the 3D mesh. The initial set of tools to support the meshing needs of M3D-C^1 were presented in reference [9]. This initial tool was limited to a predefined number of configurations in terms of physical components. The recent advances in M3D-C^1 enabled the use of more complex physical tokamak components in the simulations that posed a need of a generalized and more robust modeling and meshing tool. One of the focuses of the work presented in this thesis is the development of a generalized and fully automated modeling and meshing framework to support growing meshing needs of M3D-C^1. The unstructured meshes, if defined effectively, perform accurate simulations up to a certain number of time steps. However, with the plasma equilibrium profiles changing with time, the meshes need to be modified for accurate simulations. Adaptive mesh control based on a-posteriori error estimates can effectively produce meshes that evolve with time. The other focus of the presented work is the implementation of error-based adaptation methods both in 2D and 3D simulation workflows in M3D-C^1.

The work presented in this thesis targets the following outcomes:

- Accurate geometric representation of the tokamaks models through the implementation of methods that identify and model both physics and physical features of interest to the simulation procedures.

- Discretizing tokamaks computational domain with high-quality meshes while still satisfying the requirements placed on the mesh based on the specific simulation codes. This requires producing high-quality field-aligned meshes in TOMMS and adapting the meshes for accurate solutions in M3D-C^1

- Define an automated modeling, meshing, and adaptive framework for the two fusion plasma codes of interest that is unaffected by the changes in tokamak designs, meshing resolutions, mesh modifications, and simulation parameters.

The key research contributions include:

- Implementation of generalized and robust geometry construction methods in TOMMS that extended the support of TOMMS to new set of tokamak configurations.

- Development of methods for smooth transitions between different mesh sizes in key regions of tokamak geometry. This led to well-defined field-aligned production meshes for XGC.

- Implementation of a-posteriori mesh adaptation procedures based on error estimation in M3D-C^1. Such mesh adaptation procedures were not tried in fusion simulation codes before.

1.2 Thesis Organization

This organization of this dissertation is provided in the following paragraphs.

Chapter 2 provides an overview of the XGC and its meshing requirements. A brief introduction of the coordinate systems used in this work is also provided in this chapter.

Chapter 3 presents the technical details and workflow of TOMMS. This chapter discusses 2D poloidal model generation of a tokamak geometry from a given input plasma equilibrium file and its meshing based on the specific meshing requirements in different areas of model. This chapter also summarizes developments carried out to optimize, automate, and generalize the modeling and mesh generation procedures in TOMMS. A discussion of the mesh data structure for XGC and effective mesh entity ordering scheme is also included. Furthermore, this chapter briefly introduces the generation of meshes using the TOMMS graphical user interface.

Chapter 4 discusses the background of M3D-C^1 code and its meshing needs along with the introduction of 2D and 3D model and mesh generation workflows. A summary of recent developments towards an automated and generalized modeling and meshing infrastructure for M3D-C^1 is presented. This chapter also discusses the a-priori mesh modification options available for M3D-C^1 meshes.

Chapter 5 presents the mesh adaptation based on the the estimated errors. Results of the 2D and 3D adaptive M3D-C^1 simulations are presented.

Chapter 6 summarizes the work carried out to support automated modeling and meshing infrastructures for two tokamak plasma simulation codes. Chapter 6 also discusses the direction for future developments in TOMMS and M3D-C^1 modeling and meshing.

CHAPTER 2
FUSION EDGE PLASMA SIMULATION AND XGC

2.1 Introduction

The goal of producing clean energy from the fusion plasma in a magnetic confinement device is a challenging scientific problem. One of the key challenges is to understand the physics of the edge plasma region outside of the hot plasma core that interacts with the wall and other physical components of the tokamak. Topologically, the region with the closed flux curves is the core plasma region, and the region with the open flux curves (including separatrix) defines the plasma edge region [10]. Figure 2.1 demonstrates the classification of the plasma into core and edge plasma regions. The plasma inside the last closed curve, as shown in green in figure 2.1 is the core plasma, and the plasma outside the last closed curve, as shown in red in figure 2.1 is the edge plasma.

In the plasma edge region, the plasma interacts with the reactor wall and other plasma facing components (PFC). These interactions lead to issues related to the performance of the confined plasma and the lifetime of the reactor and its components. Firstly, the erosion of the material from plasma-facing components damages the reactor and compromises its lifetime, and secondly, the eroded particles contaminate the core plasma, affecting the fusion reaction's performance by the loss of the heat [11]. To suppress the contamination issue, a divertor is included to the system [12]. The idea is to minimize the contamination by extracting the eroded particles and other impurities through the divertor. This is achieved by the use of divertor magnets, which results in the diverting magnetic lines (separatrix). The other approach to this problem is to minimize the erosion of the materials from the plasma-facing components by injecting gas to cool down the plasma before it interacts with the wall and other components [13].

The physics of the edge plasma region is extremely complex due to a number of reasons, including the interaction of plasma with the plasma-facing components, the presence of charged particles, injected gas particles, and eroded materials particles in the edge region, and complications in the plasma transport due to the presence of the X-point [11]. An accurate understanding of this complex physics of the plasma in the edge region is required

Portions of this chapter previously appeared as:
U. Riaz, E. Seegyoung Seol, R. Hager, and M. S. Shephard, "Modeling and meshing for tokamak edge plasma simulations," *Comput. Phys. Commun.*, vol. 295, Feb. 2024, Art no. 108982.

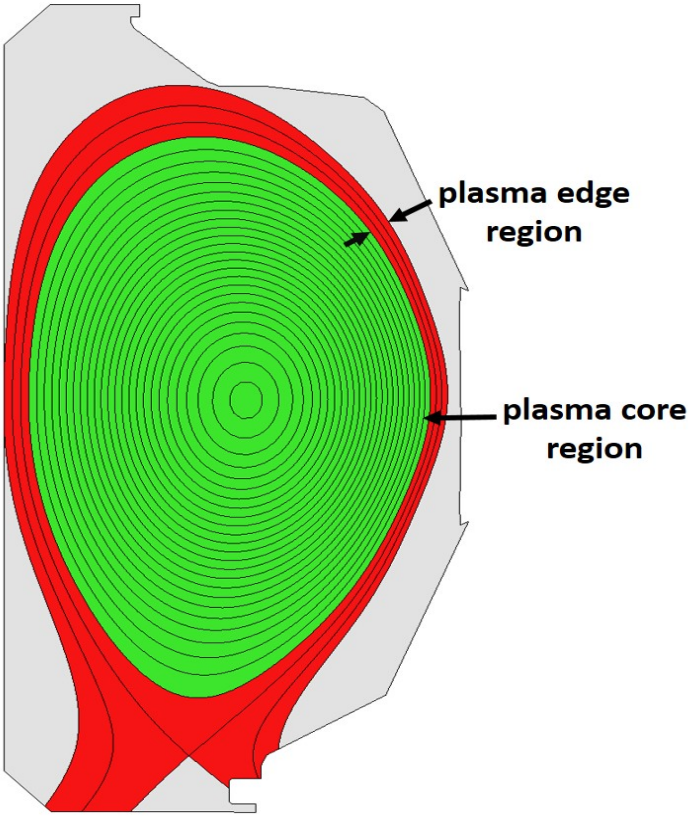

Figure 2.1: The classification of plasma regions in the tokamaks.

before large tokamaks like ITER are put in operation. The turbulent nature of plasma in the edge plasma is a major focus of a number of codes designed to study the edge regions. XGC [5] is an edge plasma PIC code that is capable of performing electromagnetic turbulence and neoclassical transport simulations. SOLPS [14] is a 2D transport code to study the multi-fluid plasma transport in the edge region [14]. SOLedge[15] is a high-order code that uses Discontinuous Galerkin discretizations over unstructured meshes to simulate 2D transport phenomenon in the scrape-off layer. UEDGE [16] is a suite of plasma codes that are focused on the 2D transport of plasma in the edge region. TOKAM2D [17] is used to study the turbulence of the plasma in the edge region in 2D. A number of codes are available for the 3D turbulence simulations of the edge plasma, including TOKAM3D [18], Hermes [19], and GRILLIX [20]. The focus of this chapter is the XGC code. A brief introduction of XGC and its meshing requirements are provided in sections 2.2 and 2.4 respectively. Section 2.3 provides an overview of commonly used coordinate systems in gyrokinetic codes.

2.2 X-point Gyro-kinetic Code (XGC)

X-point included gyrokinetic code (XGC) [21], [22], [23], [24] is a suite of PIC codes (XGC0, XGC1, XGCa) developed at Princeton Physics Plasma Lab (PPPL) to study the transport mechanism in the edge region of the tokamaks. Although XGC specializes in the edge plasma region, it is capable of simulating the full plasma domain, including all of the core plasma region.

In the PIC method, the plasma is expressed by the motion of collections of the particles as they flow in a tokamak. Although the understanding of the transport of a single particle under the magnetic field is well established, the physics as a result of collection of large number of charged particles in motion is complex due to the interaction of these particles with each other. To understand the transport mechanism in a setting where instabilities due to turbulence are present, the gyrokinetic equations can be used to describe the relevant physics. The gyrokinetic equation derived from the Vlasov equation for the particle distribution function f($\boldsymbol{x}, \boldsymbol{v}, t$) describes the particle behavior in 6D. The Vlasov equation for f($\boldsymbol{x}, \boldsymbol{v}, t$) is:

$$\frac{\partial f}{\partial t} + \frac{\boldsymbol{dx}}{dt} \cdot \frac{\partial f}{\partial \boldsymbol{x}} + \frac{d\boldsymbol{v}}{dt} \cdot \frac{\partial f}{\partial \boldsymbol{v}} = C \tag{2.1}$$

where $\boldsymbol{x}$ and $\boldsymbol{v}$ are the spatial and velocity vectors, and f describes the total particle distribution function. The particle methods that approximate the total f are known as full-f methods. To reduce the computational costs, equation 2.1 is simplified by the gyrokinetic assumption and by the use of a delta-f particle scheme as follows:

1. Under the gyrokinetic assumption, the physics of the particle can be explained by using the time scales that are much slower than the fast scale gyro-motion. This allows the description of the particle distribution function using 5D variables. Using this assumption, the f is defined by the variables $\boldsymbol{X}, \mu, v_{||}$ [25], where $\boldsymbol{X}$ is guiding center position of the particle, μ is the magnetic moment, and $v_{||}$ is the component of particle velocity parallel to the magnetic field $\boldsymbol{B}$. The magnetic moment is defined as $\mu = \frac{mv_{\perp}^2}{2B}$, where m is the particle mass, $v_{\perp}$ is the component of particle velocity perpendicular to the magnetic field $\boldsymbol{B}$, and B is the magnitude of the magnetic field $\boldsymbol{B}$. The evolution of guiding center position $\boldsymbol{X}$ and parallel velocity $v_{||}$ with time describes the motion of each particle. The governing equations are given as:

$$\frac{d\boldsymbol{X}}{dt} = \frac{1}{1+\frac{v_{||}}{B}\boldsymbol{b}\cdot(\nabla\times\boldsymbol{b})}\left[v_{||}\boldsymbol{b} + \frac{v_{||}^2\nabla B\times\boldsymbol{b}}{B} + \frac{\boldsymbol{B}\times(\mu\nabla B-\boldsymbol{E})}{B^2}\right] \tag{2.2}$$

$$\frac{dv_{||}}{dt} = \frac{1}{1+\frac{v_{||}}{B}\boldsymbol{b}\cdot(\nabla\times\boldsymbol{b})}(\boldsymbol{B}+v_{||}\nabla B\times\boldsymbol{b})\cdot(\mu\nabla B-\boldsymbol{E}) \tag{2.3}$$

where $\boldsymbol{B}$ is the magnetic field vector, $\boldsymbol{b}$ is the unit vector along the magnetic field vector $\boldsymbol{B}$, and $\boldsymbol{E}$ is the electric field vector [25]. The particle trajectory and velocity are evaluated using the equations 2.2 and 2.3.

2. Under the gyrokinetic assumption, the 5D gyrokinetic Boltzmann equation is solved to describe the change in particle distribution function $f(\boldsymbol{X},\mu,v_{||})$ due to external source and particle collisions. However, solving the full distribution function requires a large number of particles to resolve the physics accurately. To avoid the high computational costs associated with the use of a large number of particles, XGC uses the delta-f (δf) method, in which the velocity distribution function $f(\boldsymbol{X},\mu,v_{||})$ can be described as the sum of the two different contributions (f_0 and δf).

$$f = f_0 + \delta f \tag{2.4}$$

where f_0 is the time-independent distribution function also known as the equilibrium Maxwellian-Boltzmann distribution function and δf is the time-dependent perturbation. The δf method assumes that the background plasma is defined with an initial equilibrium state (represented by f_0 distribution function) that does not change during the simulation. Since only the perturbations (δf) evolve with time, using this assumption reduces the computational costs than the full-f methods.

In the XGC workflow, the particle charge density is calculated using the particle trajectory, and velocity information and then collected on the background mesh. Given the calculated charge density from the first step, the electrostatic potential field generated by the charged particles is calculated using the Poisson equation [23], [25] and the corresponding electric field is calculated on the mesh. The field information from the background mesh is gathered at the particles and in the last step particle positions and velocities are updated [23], [25].

Most of the traditional gyrokinetic codes use the flux-coordinates system to simplify the domain by not allowing the domain to approach the magnetic separatrix and magnetic axis [22]. In order to simulate a realistic domain with the actual complexities of a tokamak, the XGC code uses a cylindrical coordinate system to avoid the singularities at the magnetic separatrix and magnetic axis [24]. The next section will provide an overview of both the coordinate systems.

2.3 Coordinate Systems

The two coordinate systems that are frequently used in the gyrokinetic codes are magnetic flux coordinates and cylindrical coordinates. This section will provide a brief introduction to both the coordinate systems.

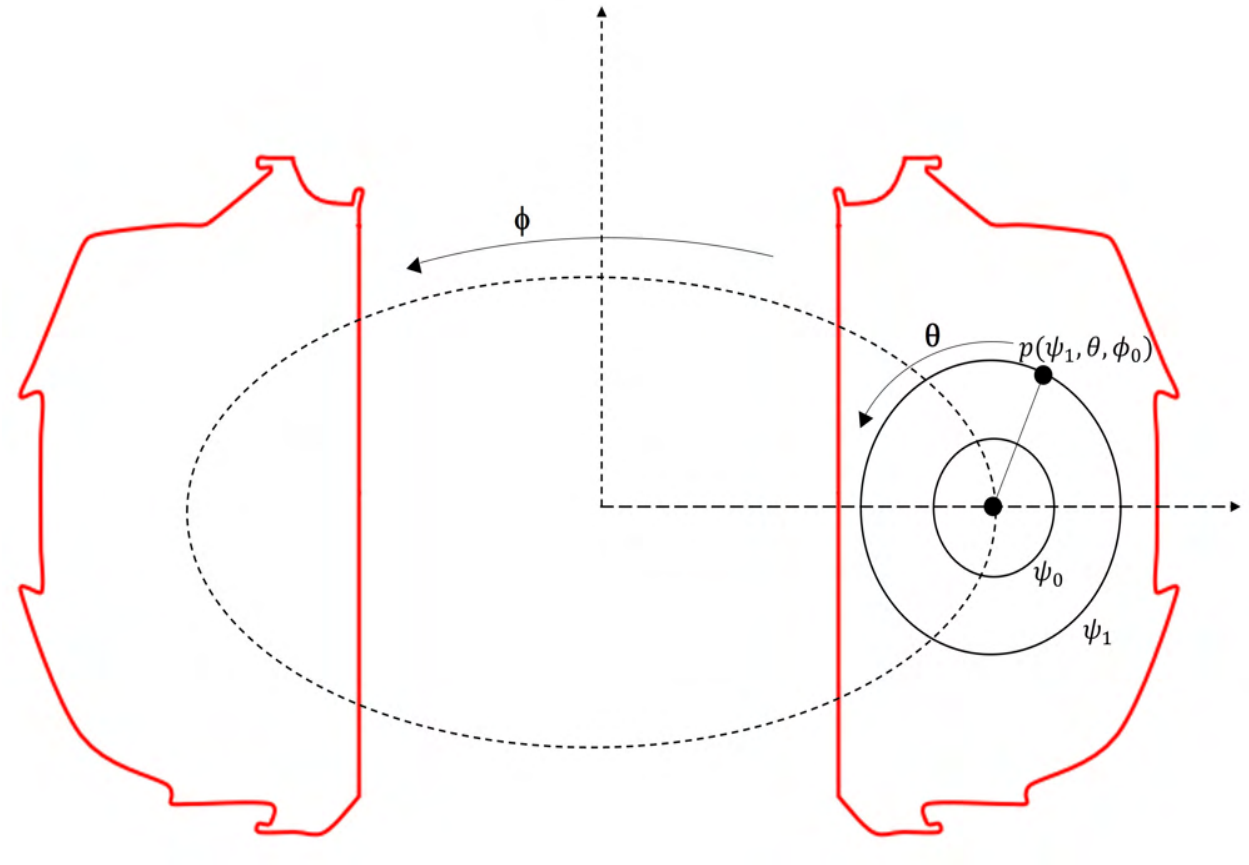

Figure 2.2: Magnetic flux coordinate system (ψ, θ, ϕ), where ψ is the flux label of the flux curve and is constant along a flux curve, θ is the poloidal angle, and ϕ is the toroidal angle.

2.3.1 Magnetic Flux Coordinates

The magnetic flux coordinate system (ψ, θ, ϕ) is a popular choice for plasma-based simulation codes such as GENE [26] and ORB5 [27]. In such plasma simulation codes, meshes aligned along the magnetic field lines are desirable which could be achieved easily by using the magnetic flux coordinates since one of the coordinates (ψ) is aligned with the magnetic field lines. In magnetic flux coordinates, as demonstrated in figure 2.2, a point on a flux surface is defined using a flux label ψ that is constant on the flux curve and does not change along the magnetic field line, position along the poloidal direction defined by the poloidal angle θ, and position along the toroidal direction defined by the toroidal angle ϕ. In figure 2.2, a point p on the poloidal plane with $\phi = \phi_0$ is defined by its flux label ψ_1 and the poloidal angle θ formed between the point and horizontal line passing through the magnetic axis on the poloidal plane.

Although magnetic field-aligned meshes can be obtained using magnetic flux coordinates in the core region, the presence of singularities at the magnetic axis and separatrix with such a system makes it impossible to accurately model the separatrix. The separatrix is one of the most critical parts of the edge plasma simulations, hence a meshing strategy with an alternate coordinate system is desirable, that allows field-aligned meshes without running into coordinate singularities. This is achieved using a cylindrical coordinate system.

2.3.2 Cylindrical Coordinates

In a right-handed cylindrical coordinate system (R,ϕ,Z), a point is defined spatially using two coordinates R and Z on a poloidal plane and one coordinate ϕ for the toroidal direction. In cylindrical coordinates, R defines the major radius of the torus, Z is the vertical axis, and ϕ is the toroidal angles as shown in figure 2.3. The right-handed cylindrical coordinates are used in XGC [23].

2.4 Magnetic Field Aligned Meshing

The use of cylindrical coordinates in XGC means that the metric tensor is free of singularities in the simulation domain and that the whole volume inside the inner reactor wall can be included without the complications arising for simulation codes based on flux-coordinates [28]. Although XGC meshes are in principle unstructured, the placement of vertices in a large part of the simulation domain is still guided by the topological entities of

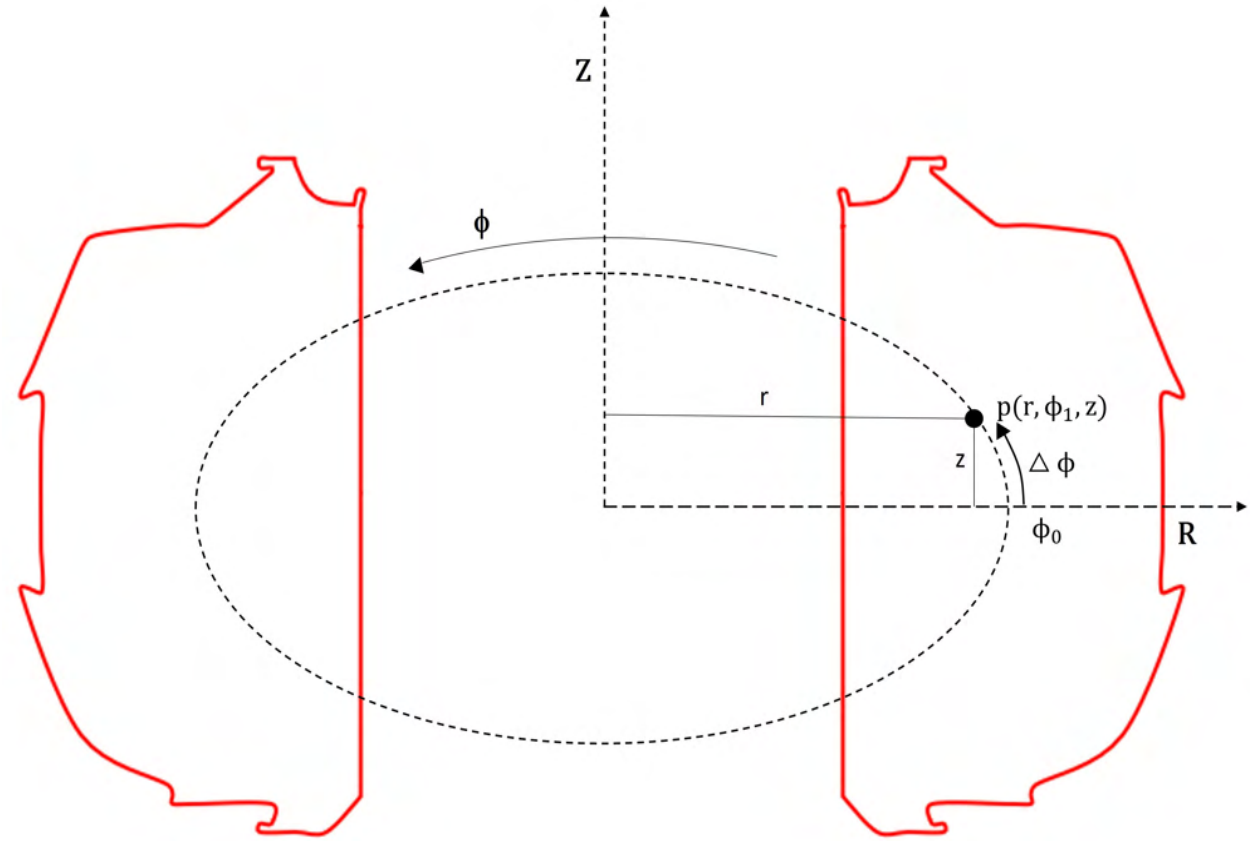

Figure 2.3: The (R,ϕ,Z) coordinate system where R is the major radius of the torus, Z is the vertical axis, and ϕ is the toroidal angle.

the magnetic field, i.e., the critical points, flux surfaces, and magnetic field lines as explained in references [8], [9], [23], [24]. This choice is rooted in the specific physics XGC is designed to model as well as in the context of the entirety of numerical methods utilized in XGC.

In flux coordinates, for which one basis vector is parallel to the magnetic field, one can easily exploit this anisotropy by drastically reducing the resolution of a the numerical model in the parallel direction. A similar benefit can be achieved using non-flux-coordinates by discretizing the model equations along the magnetic field via a magnetic field-following map of mesh vertices to mesh elements on adjacent poloidal planes (compare the discussions in references [24], [29]). In the case of particle-in-cell codes like XGC, this means choosing shape functions that generate Voronoi volumes aligned with the magnetic field (as opposed to the toroidal direction), and discretizing differential operators as a combination of derivatives along and perpendicular to the magnetic field. This approach has also been dubbed "Flux Coordinate Independent" (FCI) approach [30].

Since not every mesh vertex can be mapped exactly to mesh vertices on the adjacent poloidal planes due to critical points or constraints on the mesh size, this special numerical discretization requires interpolation (e.g. reference [29]) and introduces interpolation error and numerical dissipation. Since XGC to a large degree relies on linear shape functions, keeping the mesh closely aligned with the structure of the magnetic field minimizes numerical dissipation. GENE-X [31] is an example of an Eulerian global gyrokinetic code that utilizes the FCI approach with regular cartesian meshes but higher order interpolation.

The approximately field-aligned mesh model used by XGC is also beneficial for solving the electrostatic gyrokinetic Poisson equation (equation (8) in reference [21]), which requires the accurate evaluation of the flux-surface averaged electrostatic potential $\langle\phi\rangle$ in $\delta\phi = \phi - \langle\phi\rangle$, and it enables the use of Fourier filtering in flux-coordinates [24]. When only axisymmetric physics are modeled with XGC [32], [33], [34], the mesh model is simplified by dropping the approximately field-aligned vertex placement while retaining the flux-surface alignment.

2.4.1 Parametric Form of Magnetic Field Line

In order to generate a field following mesh for XGC, the locations of mesh vertices along the flux curves on the poloidal cross-section are such that the magnetic field line passes through these points at each poloidal plane. Figure 2.4 illustrates a magnetic field aligned mesh such that the magnetic field line starts at a mesh vertex on the poloidal plane with $\phi = \phi_i$ and intersects the poloidal plane with $\phi = \phi_{i+1}$ at one of the mesh vertices on the flux curve [24]. In order to trace points along the magnetic field line and project them on the poloidal plane, a parametric form for 3D field line is defined.

The magnetic field $\boldsymbol{B}$ in terms of its components B_R, B_Z, and B_ϕ along R,Z, and ϕ directions respectively as shown in figure 2.5 is defined as:

$$\boldsymbol{B} = B_R\hat{R} + B_Z\hat{Z} + B_\phi\hat{\phi} \tag{2.5}$$

where $\hat{R}, \hat{Z}, \hat{\phi}$ represent the directions along major radius R, vertical axis Z, and toroidal axis ϕ respectively in the (R,ϕ,Z) coordinate system as shown in figure 2.5. The magnetic field $\boldsymbol{B}$ for a given ψ and poloidal current density $I(\psi)$ [9] is expressed as:

$$\boldsymbol{B} = -\frac{1}{R}\frac{\partial\psi}{\partial Z}\hat{R} + \frac{1}{R}\frac{\partial\psi}{\partial R}\hat{Z} + \frac{I(\psi)}{R}\hat{\phi} \tag{2.6}$$

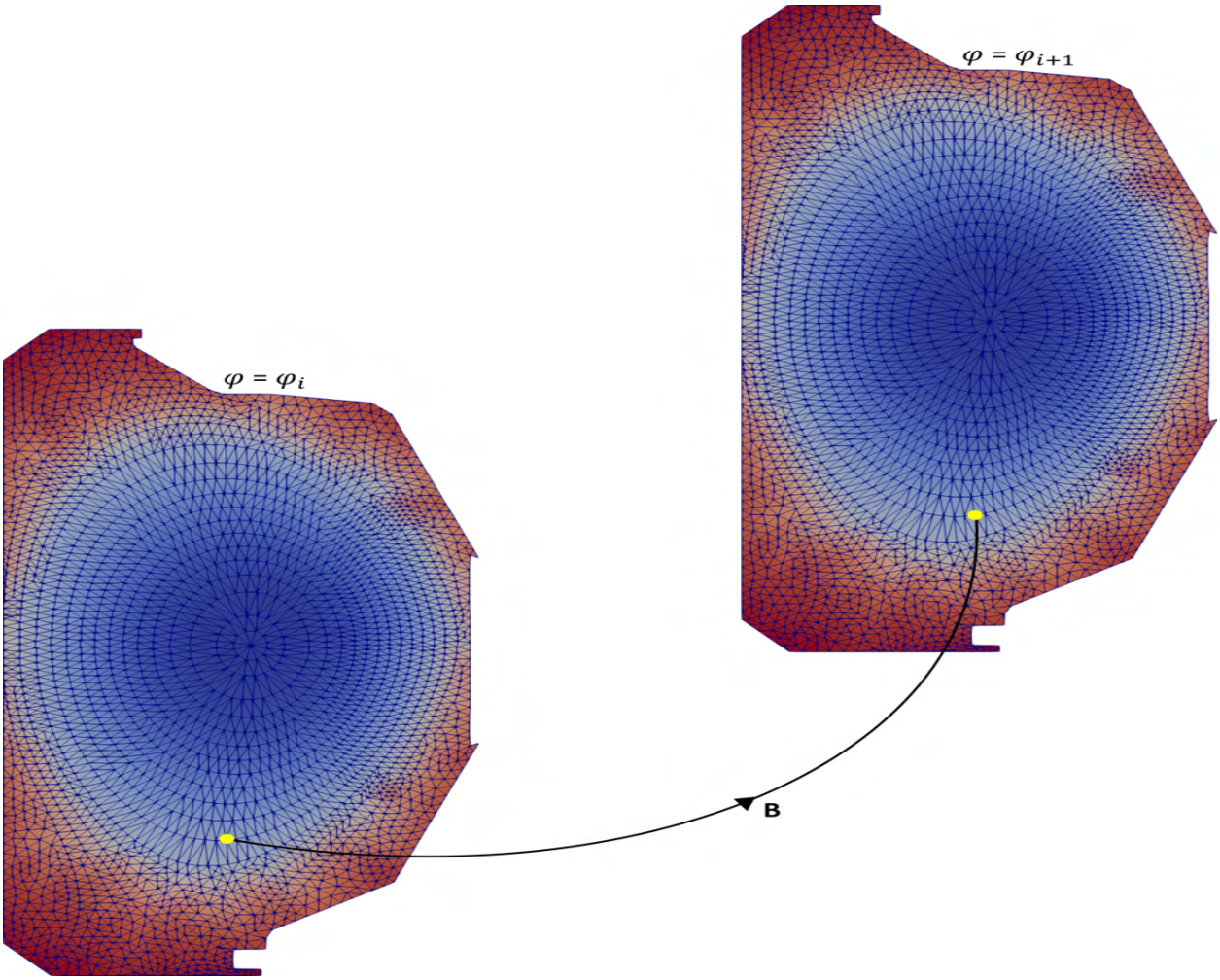

Figure 2.4: Schematic representation of field-aligned meshing in XGC.

The position vector of any point on the 3D field line can be expressed in a parametric form $\boldsymbol{L}(t)$ with respect to parameter t as shown in equation 2.7.

$$\boldsymbol{L}(t) = [L_R(t), L_Z(t), L_\phi(t)] \tag{2.7}$$

where its components are expressed in (R,ϕ,Z) coordinates.

If the trajectory of $\boldsymbol{L}(t)$ follows the field line, then $d\boldsymbol{L}(t)/dt$ is always parallel to the $\boldsymbol{B}$. By definition, the cross product of two parallel vectors is zero:

$$\frac{d\boldsymbol{L}(t)}{dt} \times \boldsymbol{B} = 0 \tag{2.8}$$

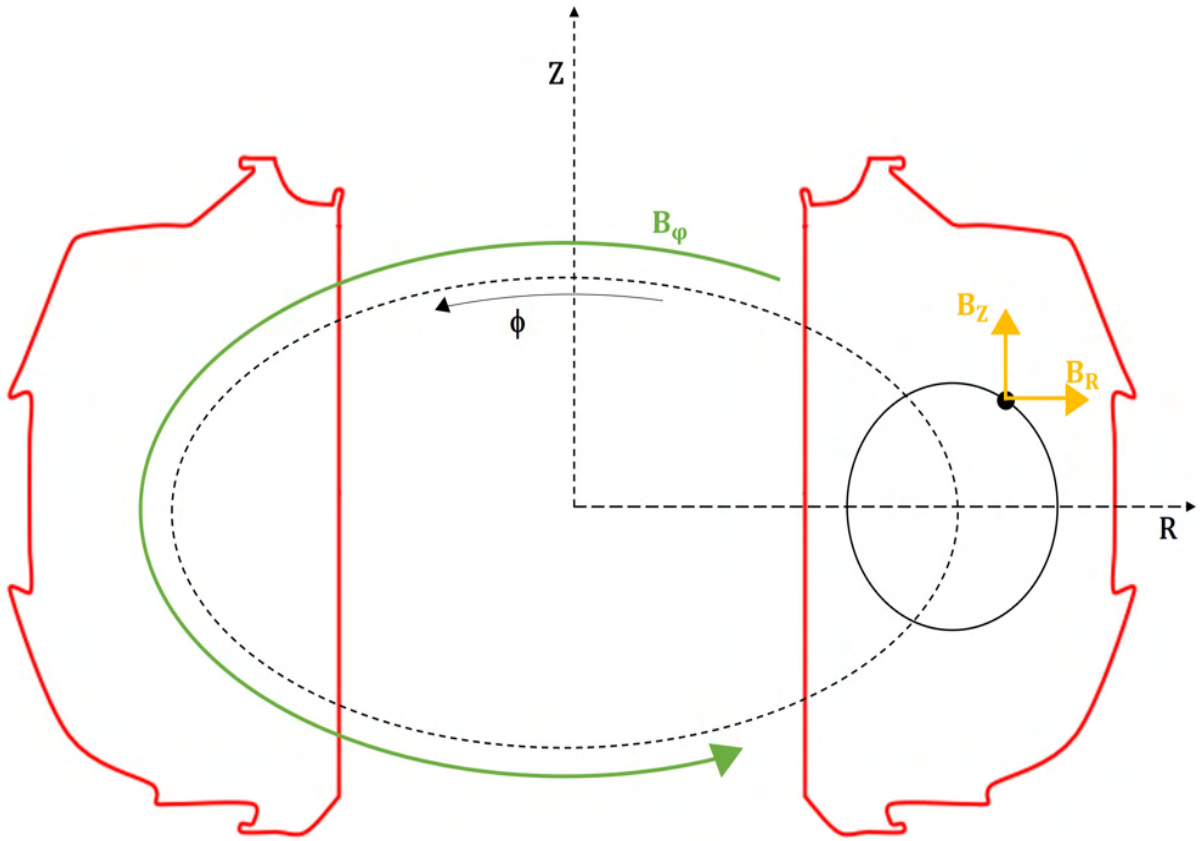

Figure 2.5: Demonstration of components of magnetic field $\boldsymbol{B}$ in R, Z, and ϕ directions in a cylindrical coordinates system.

$$\left(\frac{dL_R}{dt}, \frac{dL_Z}{dt}, R\frac{dL_\phi}{dt}\right) \times \left(B_R, B_Z, B_\phi\right) = 0 \tag{2.9}$$

In the cylindrical coordinate system, the expansion of the above cross product will result in the following form along R,Z, and ϕ directions respectively:

$$(B_\phi \frac{dL_Z}{dt} - B_Z R\frac{dL_\phi}{dt})\hat{R} = 0 \tag{2.10}$$

$$(B_\phi \frac{dL_R}{dt} - B_R R\frac{dL_\phi}{dt})\hat{Z} = 0 \tag{2.11}$$

$$(B_Z \frac{dL_R}{dt} - B_R \frac{dL_Z}{dt})\hat{\phi} = 0 \tag{2.12}$$

Rearranging the equations 2.10, 2.11, 2.12 and representing the magnetic field components B_R, B_Z, and B_ϕ in terms of ψ and $I(\psi)$ as shown in equations 2.6 results in equations

2.13, 2.14, 2.15 that represents the change in the poloidal magnetic field with the unit change of ϕ.

$$\frac{dL_R}{dt} = \frac{RB_R}{B_\phi} = -\frac{\partial\psi}{\partial Z}\frac{R}{I(\psi)} \tag{2.13}$$

$$\frac{dL_Z}{dt} = \frac{RB_Z}{B_\phi} = \frac{\partial\psi}{\partial R}\frac{R}{I(\psi)} \tag{2.14}$$

$$\frac{dL_\phi}{dt} = 1 \tag{2.15}$$

The numerical integration of the magnetic field line in equations 2.13, 2.14, and 2.15 using Runge-Kutta 4th Order (RK4) method results in points $[L_R(t), L_Z(t), \phi]$ on the flux curves that are approximately field-following. For every value of ϕ, a point $[L_R(t), L_Z(t)]$ is projected to the flux curve on the original poloidal cross-section.

CHAPTER 3
MODELING AND MESHING FOR TOKAMAK EDGE PLASMA SIMULATIONS

3.1 Introduction

The development of tokamak fusion systems capable of delivering power to the electric grid requires an ability to effectively model the system's plasma physics. A large percentage of the current state-of-the-art plasma physics analysis codes employ a discretization of the domain in terms of a mesh. In many cases, the meshes used are field-following structured meshes that are mapped based on specific physics considerations to effectively account for the anisotropy of the physics being modeled. Although field-following structured meshes support highly efficient calculations that accurately solve the specific physics equations, they require the development of specific mapping methods to distort a regular grid to the field-following coordinates, often contain mapping singularities (e.g., at the O-point) and cannot deal with the geometric complexity of real tokamak systems. On the other hand, unstructured mesh generation techniques are capable of generating graded anisotropic meshes for fully general 3D domains. Specific tokamak physics codes directly use unstructured meshes to effectively address tokamak physics cases. The M3D-C^1 code uses unstructured higher-order finite elements in the poloidal plane for 2D simulations and extrudes those meshes in the toroidal direction to create higher-order wedge elements for 3D simulations [35]. The SolEdge3X-HDG code [15], [36] employs high-order Discontinuous Galerkin discretizations over unstructured meshes to address plasma physics problems. The PetraM/MFEM code uses graded unstructured tetrahedral meshes to perform tokamak radio frequency (RF) simulations, including detailed antenna geometries [37]. The GITRm code uses graded, anisotropic unstructured tetrahedral meshes to perform impurity transport simulations [38]. In the case of the XGC edge plasma particle-in-cell (PIC) code [22],[39], unstructured meshes capable of resolving fully detailed poloidal plane geometries while maintaining a nearly field-following representation in the toroidal direction, are being used. This paper is a follow-up to reference [9], which presented an unstructured mesh generation tool that met an initial set of meshing

Portions of this chapter previously appeared as:
U. Riaz, E. Seegyoung Seol, R. Hager, and M. S. Shephard, "Modeling and meshing for tokamak edge plasma simulations," *Comput. Phys. Commun.*, vol. 295, Feb. 2024, Art no. 108982.

requirements for M3D-C^1 and XGC. The current paper is focused on recent developments carried out to extend the Tokamak Modeling and Meshing Software (TOMMS) [40] to meet additional meshing requirements of the current XGC gyrokinetic PIC code.

To model plasma physics in the entire tokamak domain XGC employs a 2D triangular mesh defined on a poloidal plane which is then repeated on a set of poloidal planes [22], [39]. Unlike the 2D SolEdge3X-HDG [15], [36] and M3D-C^1 [35] meshes, the XGC meshes are not fully unstructured with respect to the placement of mesh nodes. By using approximately field-following meshes, XGC can exploit the scale anisotropy of turbulence (Larmor radius scale perpendicular to the magnetic field but device-scale along the magnetic field) and model turbulence with high toroidal periodicity with relatively low toroidal resolution [23], [30] and low-order numerical methods. To gain the advantage of a field following mesh [22],[39] in the critical portions of the domain, a specific mesh generation process is applied in which the mesh vertices are placed at critical points and on a selected set of flux curves. The placement of the mesh vertices is such that as one traverse from one poloidal plane to the next, the field following nature of the mesh is maintained. Further, the mesh defined between pairs of closed flux curves is such that no additional mesh vertices are introduced between flux curves until the region between the last closed curve and the separatrix, where additional mesh vertices are introduced to control the mesh quality near the X-point. Similarly, there are additional mesh vertices introduced between pairs of open flux curves, but only in areas close to where those curves intersect the tokamak wall and in regions bounded by a single flux curve and the tokamak wall. General unstructured meshes are generated in such areas where mesh vertices are desired between pairs of flux curves or areas bounded by a single flux curve and the tokamak wall. Section 3.2 gives an overview of the workflow in TOMMS.

To support the ability to automatically generate the desired poloidal plane meshes, the TOMMS mesh generator requires a geometric model that includes a description of not only the tokamak wall but the physics features of the O-point, X-point(s), separatrix curves and selected flux curves. Section 3.3 presents the procedures used to construct the plasma geometric model that consists of a set of model vertices, edges, loops, and faces that properly represent the tokamak wall and the desired physics features. The procedures presented in this paper are generalized to support a full range of O-point and X-point combinations, while the previous procedures [9] were limited to specific predefined X-point configurations.

Section 3.4 briefly summarizes the mesh generation procedures originally developed

to construct XGC meshes, the details of which are provided in reference [9]. The resulting procedure is capable of producing good-quality meshes as long as the aspect ratios of elements between pairs of flux curves are not too high. As the XGC code continued to be applied to perform more accurate physics simulations, it became necessary to support mesh elements with high aspect ratios between flux curves. As indicated in section 3.5, the generation of satisfactory meshes for these cases required further developments with respect to refining the topology of the plasma geometric model being meshed and the application of specific mesh generation procedures on the resulting model faces.

The ability to effectively execute the various XGC computational operations requires providing specifically structured sets of mesh information. Section 3.6 discusses providing this data and having it ordered for efficient execution. Section 3.7 provides an overview of the graphical user interface (GUI) of TOMMS supported through the Simmodeler.

3.2 TOMMS Workflow

Figure 3.1 illustrates the TOMMS workflow for the generation of flux-aligned meshes. The plasma equilibrium file (GEQDSK) provided as an input to TOMMS contains the poloidal flux function (ψ) on the background grid and a description of the tokamak wall in terms of discrete edges. The discrete ψ field is used to construct C^2 continuous field using the PSPLINE library [41]. The discrete wall information is processed to generate a wall curve based on a set of straight and curved edges using the procedures described in section 3.3.1. The continuous ψ field is used to search the critical points in the computational domain. These critical points are filtered by processing the critical points that lie inside the wall curve. The details of the critical points search are presented in section 3.3.2.2. After finding the critical points, the next step is to use the flux curves definitions provided in the input parameters file and the continuous ψ field to find the starting points of the curves. The flux curves are traced using these starting points and magnetic field information (evaluated from continuous ψ field) as explained in section 3.3.2.3. Once the physics entities are defined (critical points, flux curves), the model is defined by adding the model entities to these physics entities. This includes defining model vertices on the critical points, start points of the flux curve, and endpoints (if not periodic) of the flux curve, and also model edges to the flux curves. After the model definition, the model areas are divided into regions driven by the physics (plasma core, scrape-off layer, etc). The model just defined is suitable for the

generation of high-quality meshes as long as the aspect ratios of elements between flux curves are not too high. In such cases where it is high, the model topology needs to be updated based on the complexity of the geometry using the procedures described in section 3.5. The model is ready for meshing at this stage. The model is meshed using an one-element-deep meshing procedure and a general unstructured meshing procedure based on specific requirements of different physics regions. The resulting mesh information is provided in terms of a set of output files (node file, element file, and flux curves file) for use in XGC. The details of the specific output requirements for the XGC simulations are presented in section 3.6.

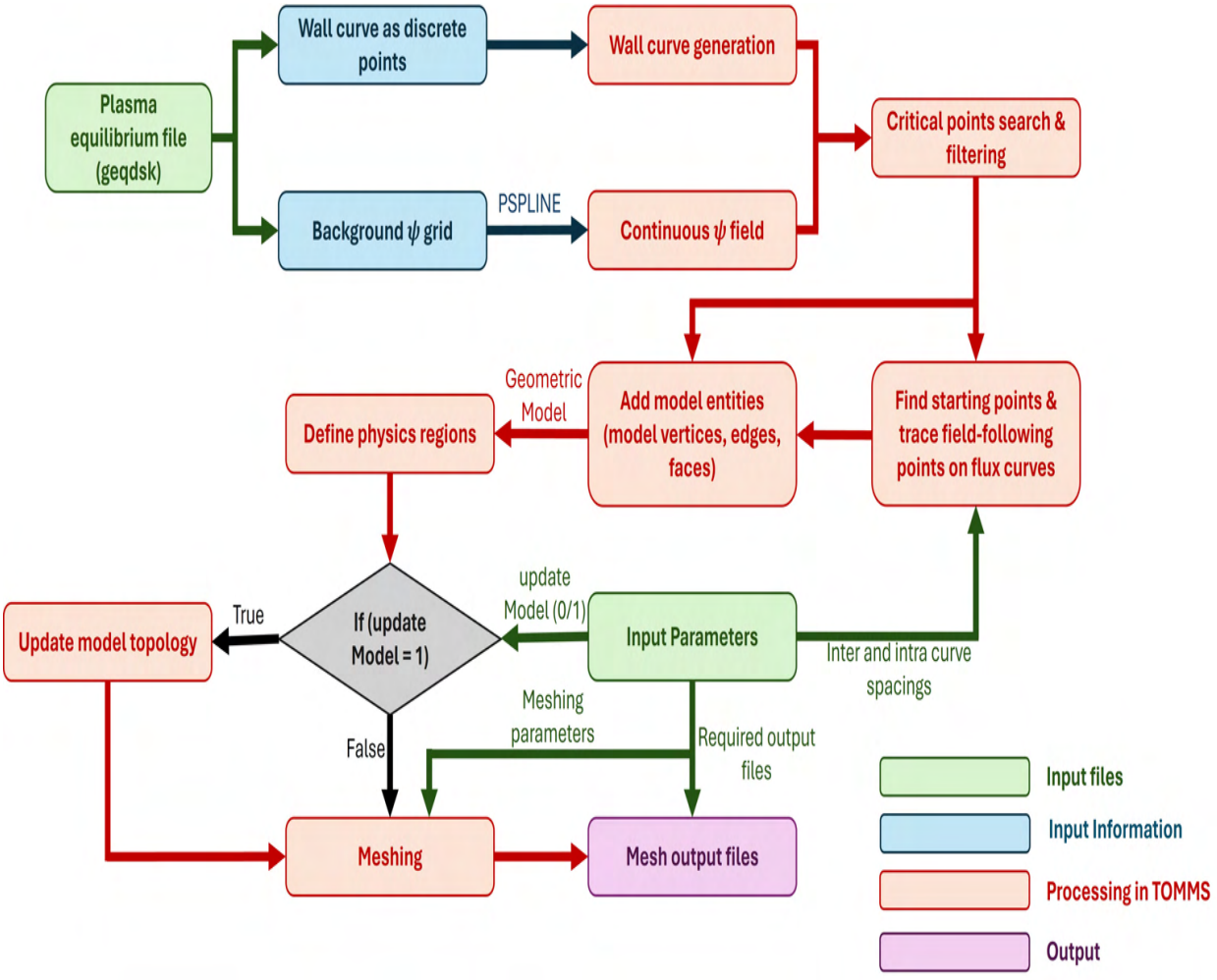

Figure 3.1: TOMMS workflow for the generation of field-aligned tokamak meshes.

3.3 Plasma Geometric Model Construction

The most general input to an automatic mesh generator is a boundary representation of the domain to be meshed in terms of a topological description of the domain in which the shape information is associated with the topological entities. In the case of a poloidal plane cross-section of a tokamak, the topological model is a set of faces where each face is bounded by loops of edges that are bounded by vertices.

To support the TOMMS user community, the plasma geometric model is constructed from two commonly used community inputs. The first input is the discrete description of the tokamak wall. The second is the poloidal flux function (ψ) field provided as the ψ values at the vertices of a background grid that encloses the tokamak. The ψ field provided in the GEQDSK equilibrium file is used to construct the physics geometric model entities of O-points, X-points, separatrix curves and selected flux curves.

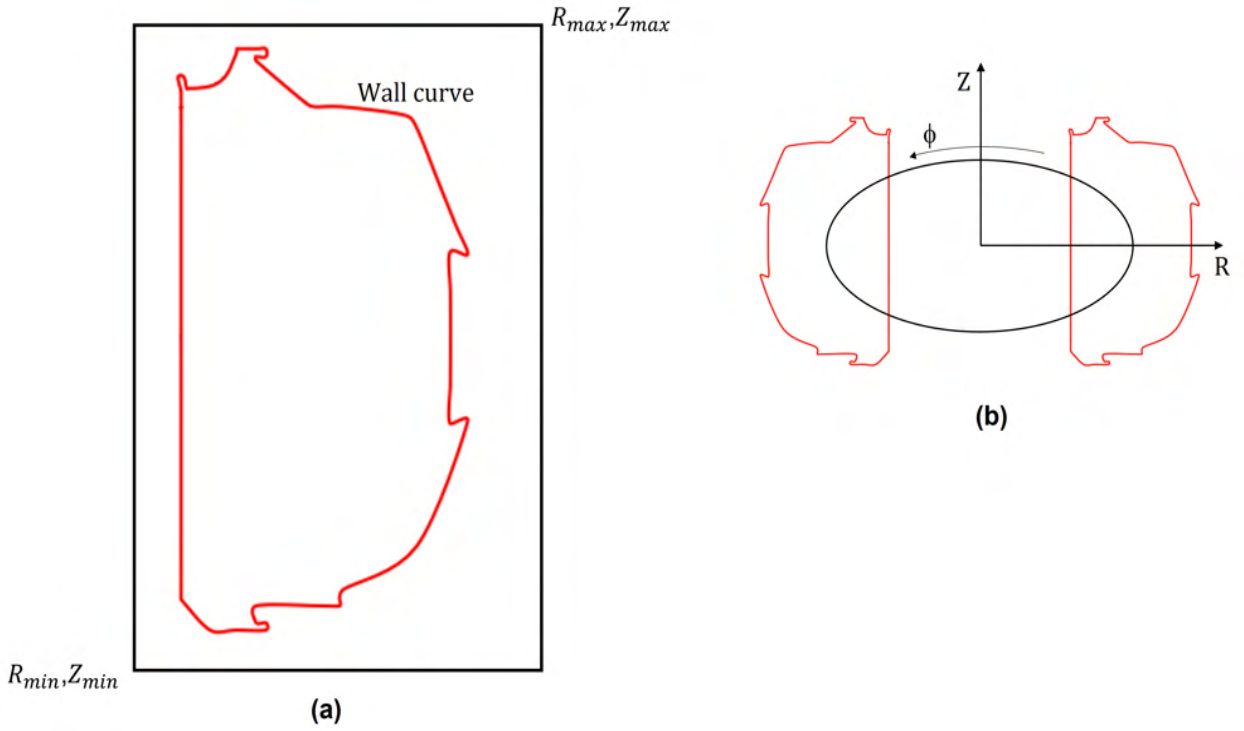

Figure 3.2: (a) A sample computational grid boundary (black) and actual physical wall curve (red) of a DIII-D tokamak, (b) the (R,ϕ,Z) coordinate system where R is the major radius of the torus, Z is the vertical axis, and ϕ is the toroidal angle.

A sample computational grid boundary and physical wall curve of the DIII-D tokamak [42] are shown in figure 3.2a. Figure 3.2b illustrates the (R,ϕ,Z) tokamak coordinate system

used in this work.

3.3.1 Modeling Tokamak Wall Boundary

The input received for the wall geometry is a set of line segments defined between consecutive points which could be directly used to define the loop associated with the wall either as a single piecewise linear edge with a single model vertex at the starting point or as a set of model edges where each input segment is a model edge bounded by two model vertices. Neither of these options is ideal in that in the first case geometric features like model corners would be lost and in the second case, the mesh generator would be constrained to place mesh vertices at the locations of all of the large set of model vertices. The desired tokamak wall loop would have model vertices only at the appropriate locations that represent "corners" with edges of the appropriate shape defined between those model vertices. Since such information is not provided, the input set of line segments is processed to extract the desired set of model edges as follows:

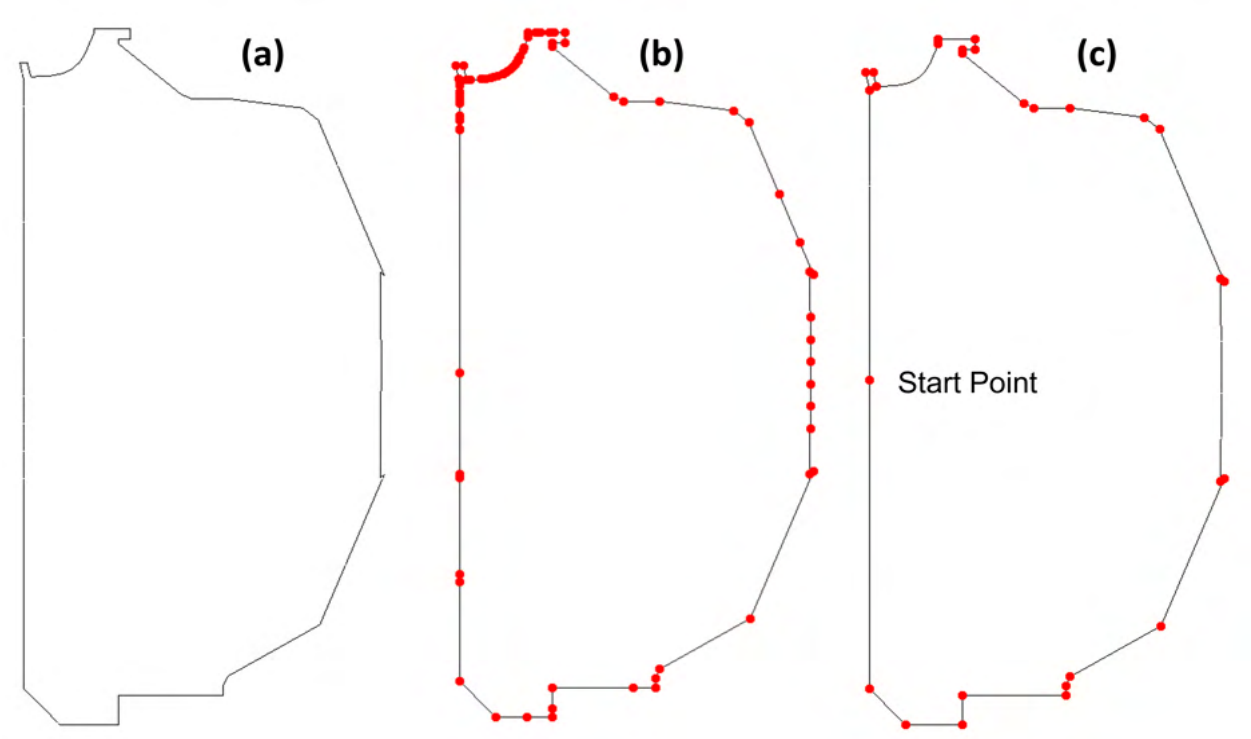

Figure 3.3: (a) Actual wall geometry, (b) set of points before processing, (c) final points used in the model.

The procedure begins by placing a model vertex at the start of the first line segment.

The next line segment is then examined. If the angle between the two line segments is 180° to within a tight tolerance, the line segment is concatenated to the current edge that is identified as a linear edge. The procedure examines line segments until one is found where the angle is no longer close enough to 180°. A model vertex is placed at the end of the previous segment, and the current segment is considered to be the start of the next edge. When the angle between two line segments is not 180°, a decision needs to be made to determine if there will be a model vertex placed between the two line segments or if the segment should be concatenated to the previous segment and considered as part of a curved edge. The procedure for making this decision is based on the algorithm presented in reference [43]. Additional segments are considered part of the curved edge until either consecutive line segments again meet the linear edge criteria, at which time they form the beginning of a straight edge, or the angle change is so large that a model vertex is required, and then we begin processing the next two line segments to determine if the next model edge is a straight line or a curved edge.

Algorithm 1 provides a pseudo code for the method described above. The results demonstrating the use of the above method are presented in figure 3.3. The set of line segments defined in terms of their endpoints can be seen in figure 3.3b. The set of model edges with their endpoints (red) after processing the points is presented in figure 3.3c.

3.3.2 Processing Magnetic Flux Function Field to Define the Physics Geometry

The poloidal flux function (ψ) field drives the construction of the physics components of the plasma geometric model that consist of O-points, X-points, separatrix curves, and selected flux curves. Figure 3.4a depicts a set of representative physics model entities generated on the rectangular grid that the flux field is defined over. Figure 3.4b represents a completed plasma geometric model that can be given to mesh generation procedures.

The minimum point of the ψ field makes up the magnetic axis (O-point) of the domain shown as $V1$ in figure 3.4a. Topologically, the O-point is the model vertex with no adjacent model edges. The X-points shown as $V2$ and $V3$ are the saddle points of the ψ field where the magnetic flux curve diverges. The most common plasma geometric model configurations contain one or two X-points. There are three types of physics curves in the domain. The closed magnetic flux curves ($C1$) are the non-interesting curves constructed along the ψ field in the core region ($R1$) of the tokamak. The separatix curves ($C2$ and $C4$) intersecting at X-

Algorithm 1 Tokamak Wall Modeling

```
1: procedure BuildWallCurve(P(1:N))                    ▷ N number of given points
2:     P(1) ← End Point(E)                              ▷ set first point as end point
3:     for i = 2 to N do
4:         Evaluate if P(i) is an End Point(E), Curve Point(C) or point on straight line(S)
5:         M ← 1                                        ▷ Initiate M with 1
6:         if P(i) = E and P(i − 1) = E then
7:             Connect P(i) and P(i − 1) with a straight line
8:         end if
9:         if P(i) = C then
10:            if P(i − 1) = E then
11:                Save P(i − 1) and P(i) in Curve
12:            else
13:                Save P(i) in Curve
14:            end if
15:            M ← M + 1
16:        end if
17:        if P(i) = E and P(i − 1) = C then
18:            Save P(i) in Curve
19:            M ← M + 1
20:            Fit a spline for M number of points in Curve
21:            Clear the Curve for the next curve
22:            M ← 1
23:        end if
24:        if P(i) = S then
25:            if P(i − 1) = E then
26:                S1 ← P(i − 1)
27:            else
28:                Ignore the point
29:            end if
30:        end if
31:        if P(i) = E and P(i − 1) = S then
32:            S2 ← P(i)
33:            Connect S1 and S2 with a straight line
34:            Clear the points S1 and S2 for next straight line points
35:        end if
36:    end for
37: end procedure
```

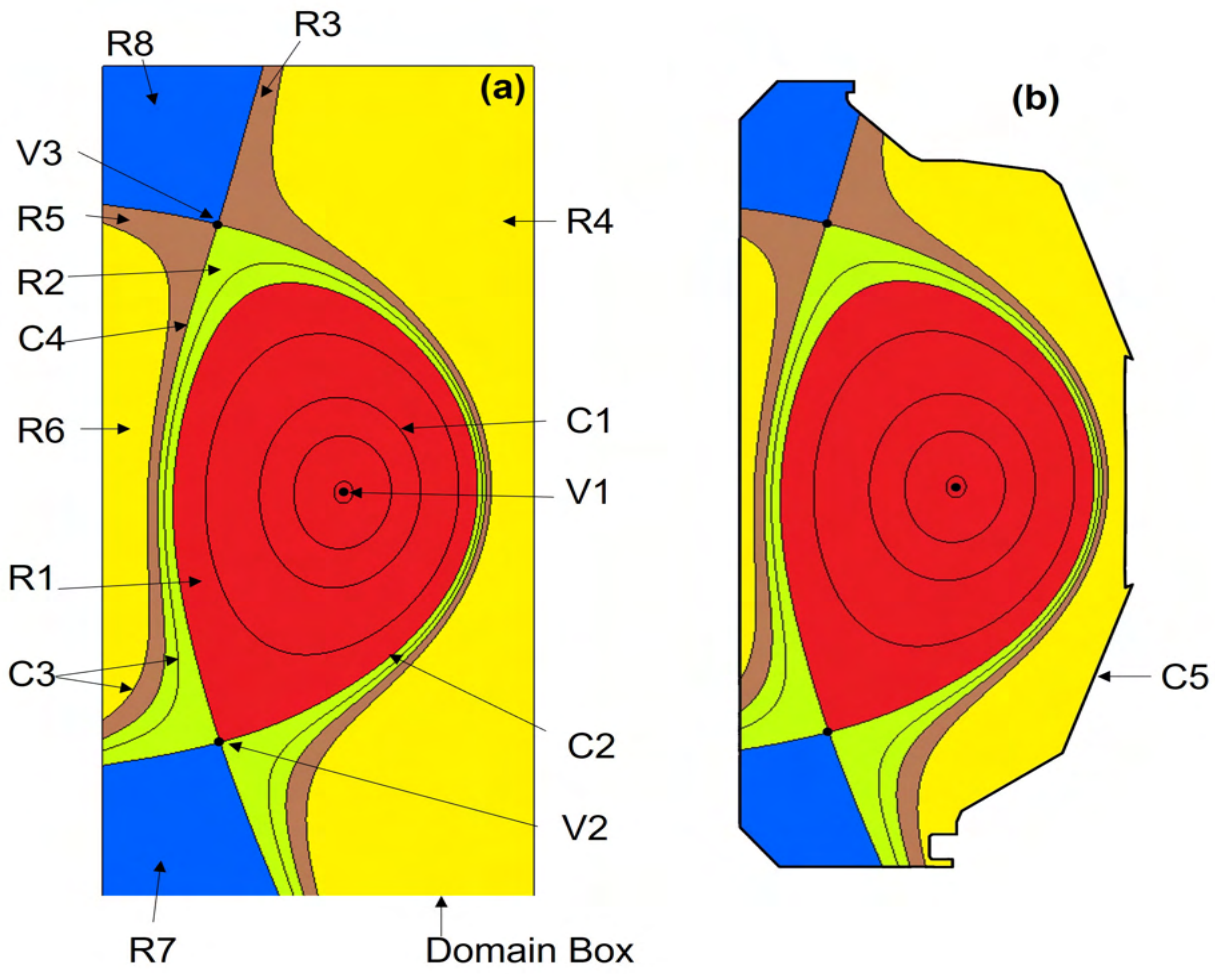

Figure 3.4: (a) Physics components of plasma geometric model. V represents model vertices, C represents curves, and R represents physics regions.V1 = magnetic axis, V2, V3 = X-points, C1 = closed magnetic flux curves, C2, C4 = separatrix curves, C3= open magnetic flux curves, R1 = plasma core region, R2 = scrape-off layer, R3 = low field side edge region, R4 = near-vacuum region, R5 = high field side edge region, R6 = near-vacuum region, R7 = lower private region, R8 = upper private region, (b) physics model entities bounded by a physical boundary (wall curve C5).

points split the domain into different physics regions. There is a separatrix curve associated with each X-point in the domain. Topologically a separatrix curve consists of a set of model edges in such a way that every X-point has three (two legs intersecting the wall and two legs forming one closed model edge) or four (all four legs intersecting the wall edges) model edges adjacent to it. The open flux curves are defined as the flux curves outside the outer most separatrix curve and flux curves between two separatrix curves. The open flux curves

($C3$) intersect with the wall curves at their starting and ending points along the ψ field, and they do not self-intersect at any point in the domain. The plasma core region ($R1$), which is enclosed by the inner separatrix contains all the closed curves and magnetic axis. The scrape-off layer between the inner and outer separatrix curves marked as $R2$ contains a set of open curves. The magnetic field at the inboard plane (left of the magnetic axis) is stronger than the magnetic field at the outboard plane (right of the magnetic axis) and plays an important role in the definition of physics regions. The physics regions at the outboard are low-field regions, and regions at the inboard planes are high-field regions. The low field side edge region ($R3$) and high field side edge region ($R5$) are the regions containing the open curves outside of the outer separatrix. These regions end with the last open flux curves. The regions between the last open curves and wall curves are defined as near-vacuum regions ($R4$) and ($R6$). The regions bounded by the two legs of separatix and a wall curve are defined as private regions. Depending upon the Z-coordinates of X-points, one region is defined as a lower private region ($R7$) and the other one as an upper private region ($R8$).

There are three steps involved in constructing the physics geometric model entities from the flux field input of the ψ values at the vertices of the background. The first step constructs a ψ field over the grid that is C^2 continuous to support the operations of the second step, which is to determine the locations of the critical points which are used in the third step of defining the desired set of flux curves.

3.3.2.1 Discrete Field to Continuous Field

The ψ field is defined in terms of discrete values at the vertices of a background grid. A C^2 continuous bi-cubic spline ψ field is constructed using the PSPLINE routine [41]. Having a C^2 representation effectively supports the procedures used to determine the locations of the O-points and X-points since they employ first and second-derivative evaluations. The bi-cubic spline of ψ is the fundamental shape information defining the geometric shape of the flux curves used as physics geometry in the plasma geometric model.

3.3.2.2 Critical Point Search Method

The critical points are needed to identify the physics regions in the domain and in the construction of the desired set of flux curves. The O-point, also known as the magnetic axis point, is defined as the minimum of the poloidal flux function ψ. The O-point represents

the "center" of the core region that will be surrounded by a selected number of closed flux curves in the construction of the plasma geometric model. The X-points, also known as the magnetic null point, are saddle points of the ψ field. Since there can be multiple minima (only one of which is at the O-point of the core) and multiple saddle points (X-points), an accurate search method capable of identifying all critical points over the domain of the background grid is required. The search method employed is the Downhill Simplex method [44]. The method uses simplex cells, in 2D triangles. References [44], [45] provide a full description of the method. To employ the Downhill Simplex method for extreme point search, the background grid is subdivided into a fine search grid. The local extreme points are found using Newton-Raphson root-finding [46] with the points evaluated by the Downhill Simplex method as initial guesses for the Newton method. To find out if the point is a minimum point (O-point) or saddle point (X-point), the Second Partial Derivative (Hessian (H)) test [47] of the ψ field is used. The determinant of H is defined using:

$$\det H = (\frac{\partial^2 \psi}{\partial R^2} \times \frac{\partial^2 \psi}{\partial Z^2}) - (\frac{\partial^2 \psi}{\partial R \partial Z})^2 \tag{3.1}$$

- If $\det H < 0$, it's a saddle point (X-point)
- If $\det H > 0$ and $\frac{\partial^2 \psi}{\partial R^2} > 0$, it's a minimum point
- If $\det H > 0$ and $\frac{\partial^2 \psi}{\partial R^2} < 0$, it's a maximum point

The search method returns the number and type of each critical point along with the location of critical points and ψ values at these points.

3.3.2.3 Flux Curves

The next step is to form the base definition of the flux curves in the domain. Given the fact that we have a continuous definition of the ψ field over the background grid, the base definition of the flux curves will consist of the determination of the ψ value of each flux and separatrix, the topology of the model edge(s) that will represent the flux curves, and the location of model vertices that must be found as part of determining the topology of the edge(s) that represent the flux curves.

The input required for this process is the number of desired flux curves N and a "normalized" $\tilde{\psi}_i$ value for each flux curve where $\tilde{\psi}_i > 0$. The normalization is performed

such that $\tilde{\psi}_i = 1$ will correspond to the first separatrix. Given that, $0 < \tilde{\psi}_i < 1$ will correspond to closed flux curves inside the first separatrix and $\tilde{\psi}_i > 1$ correspond to open flux curves outside the first separatrix. Since a flux curve will be generated for each separatrix the total number of physics curves to be defined is $N_{total} \leq N + N_X$ where N_X is the number of separatrix points found. In the common case of when there is a single separatrix with $\tilde{\psi}_i = 1$ the number of flux curves generated is N. The flux value associated with each of the requested N flux curves is defined as

$$\psi_i = \tilde{\psi}_i(\psi_{sep} - \psi_0) + \psi_0, \quad i = 0, 1, 2, \cdots, N-1 \tag{3.2}$$

Where ψ_0 and ψ_{sep} are the ψ values at the O-point, and first separatrix curve (X-point), respectively.

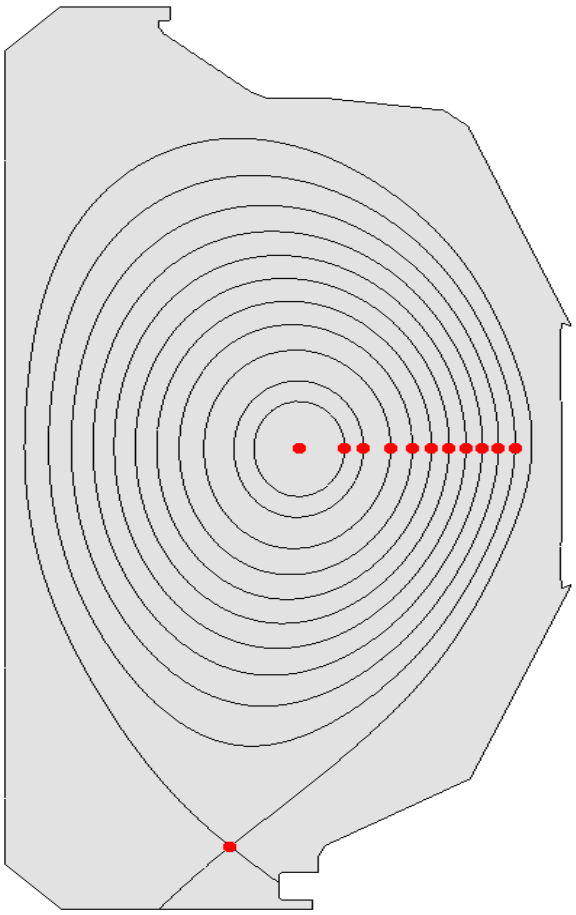

Figure 3.5: Demonstration of starting points for the definition of closed curves for DIII-D case.

After ψ values for the closed flux curves are determined from the input $\tilde{\psi}_i$ values, the next step is to find the starting points of each closed flux curve along the outboard midplane as shown in figure 3.5. The starting point of each closed flux curve is found by a bisection procedure that iterates until a point on the outboard plane with the required ψ value is found. The model edge defining the closed flux curve is bounded by a single model vertex that is placed on the starting point on the outboard plane as shown in figure 3.5.

Separatrix curves are the flux curves that are self-intersecting at the X-point and also intersect at the wall curve. There can be multiple X-points in the domain, and each X-point has a ψ value already evaluated from the procedures in section 3.3.2.2. Note that each separatrix curve is associated with one or more X points. Since the locations of the X-points are already known, the topology of the separatrix curves can be defined by evaluating the intersection of these curves with the wall shown as points i, ii, iii, iv in figure 3.6. To find these intersections, all the possible points on the wall boundary with the ψ value of the given separatrix curves are evaluated by a traversal of the wall geometry determining locations where the ψ value is that of the flux curve of interest. The model edges defining the separatrix curves are of three types, (a) model edge bounded by a single X-point model vertex as shown in figure 3.6a, (b) model edge bounded by two model vertices, one at X-point and the other one at the intersection of separatrix and wall, and (c) model edge bounded by two X-points model vertices as shown in figure 3.6c.

Open flux curves are the flux curves that exist outside of the core region ($\tilde{\psi} > 1$) and intersect with the wall boundary. A single flux curve can have multiple intersections with the wall boundary; therefore, an open flux curve can require the use of multiple model edges to represent, as can be seen in figure 3.7, where one flux curve intersects the wall at six locations, thus defining three model edges bounded by model vertices at the locations where the flux curve intersects the wall. To define the topology and model vertex locations, a method similar to that used for the separatrix-wall intersections is applied.

3.3.3 Example Models

The physics and physical model entities created using the procedure explained in the preceding subsections produce the plasma geometric model.

Figure 3.8 illustrates the plasma geometric models for different tokamak geometries created using procedures explained in this section. The figures 3.8a and 3.8b present base

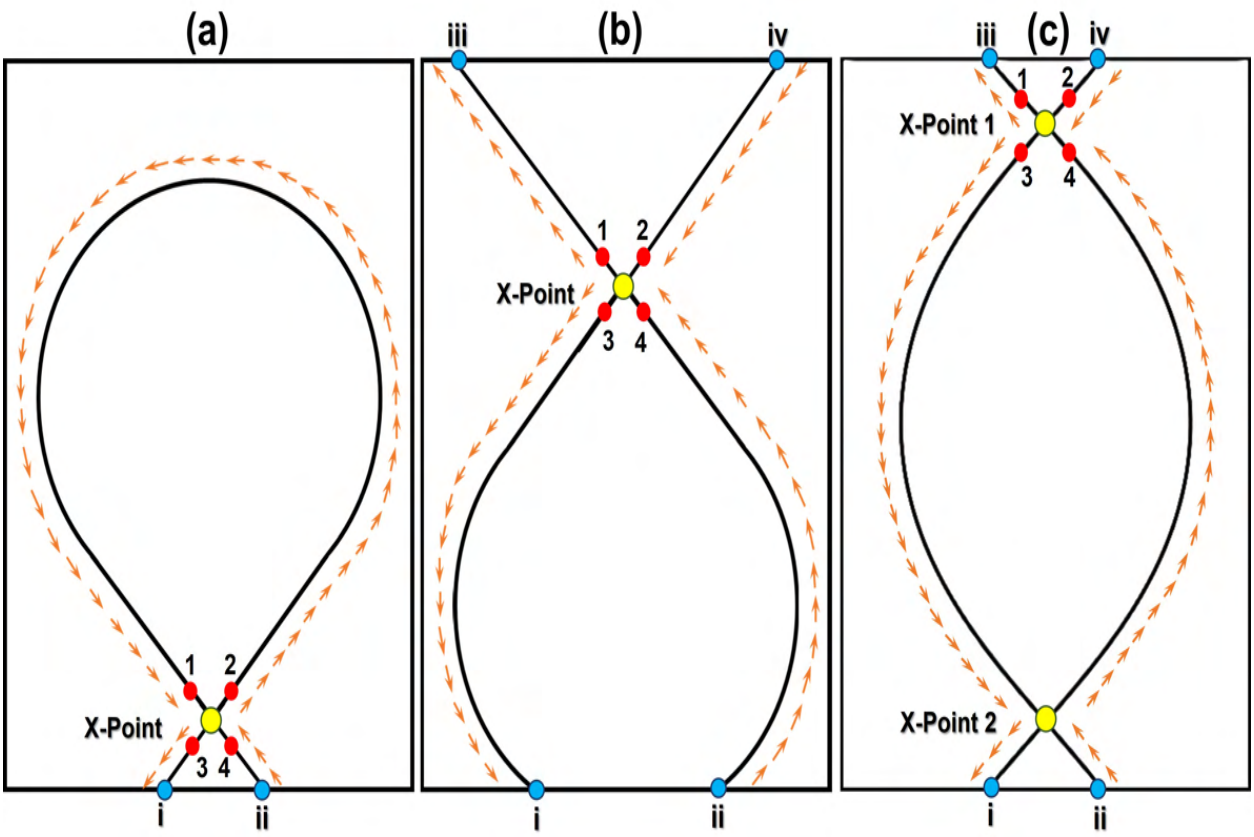

Figure 3.6: Construction of separatrix curves (a) inner separatrix curve, (b) outer separatrix curve, (c) separatrix curve with two X-points. Note that in this example the wall geometry is a simple rectangle.

models with one X-point and two X-points, respectively, where X-point is shown in the solid black circle. Each color in the figure shows a different physics region with specific meshing requirements. Figure 3.8c demonstrates a special case of two X-point model, where both the X-points are on the same flux curve defined by the same ψ value.

The procedure for model generation explained in section 3.3, and the models presented in figure 3.8 are in 2D. The 3D tokamak model, as required in some fusion simulation codes, can be created by rotating the 2D plasma geometric model for the required toroidal angle. Figure 3.9 presents the cross-sectional view of a 3D model created using 180° rotation of a 2D plasma geometric model with one X-point.

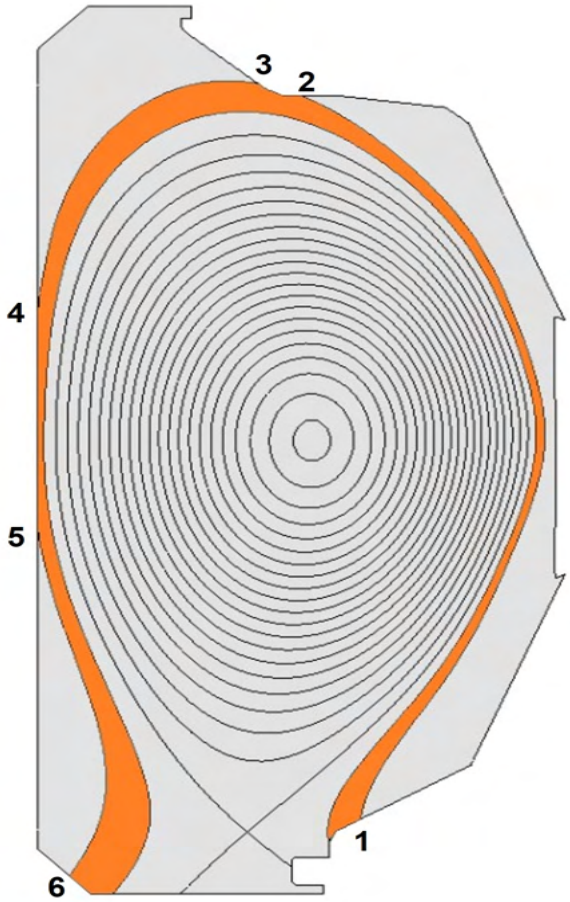

Figure 3.7: A face on open curves of tokamak cross-section of DIII-D. The inner loop consists of a flux curve with a single edge, and the outer loop consists of a flux curve with multiple edges.

3.4 Original Mesh Generation Procedure

The original mesh generation procedures employed the plasma geometric model in combination with a near field-following mesh vertex placement procedure and three specific meshing tools [9].

The first step in this process is to place the desired field following mesh vertices on the physics geometry of open and closed flux curves and separatrix curve(s). Each of the model edges representing either all or part of a physics curve has at least one starting model vertex. The model vertex will represent a mesh vertex on that physics curve and is assumed to be the starting point of a field-following curve on the poloidal plane mesh. From there, the field-following procedure given below will determine the location where a field-following point starting at that mesh vertex will intersect the next poloidal plane. This intersection

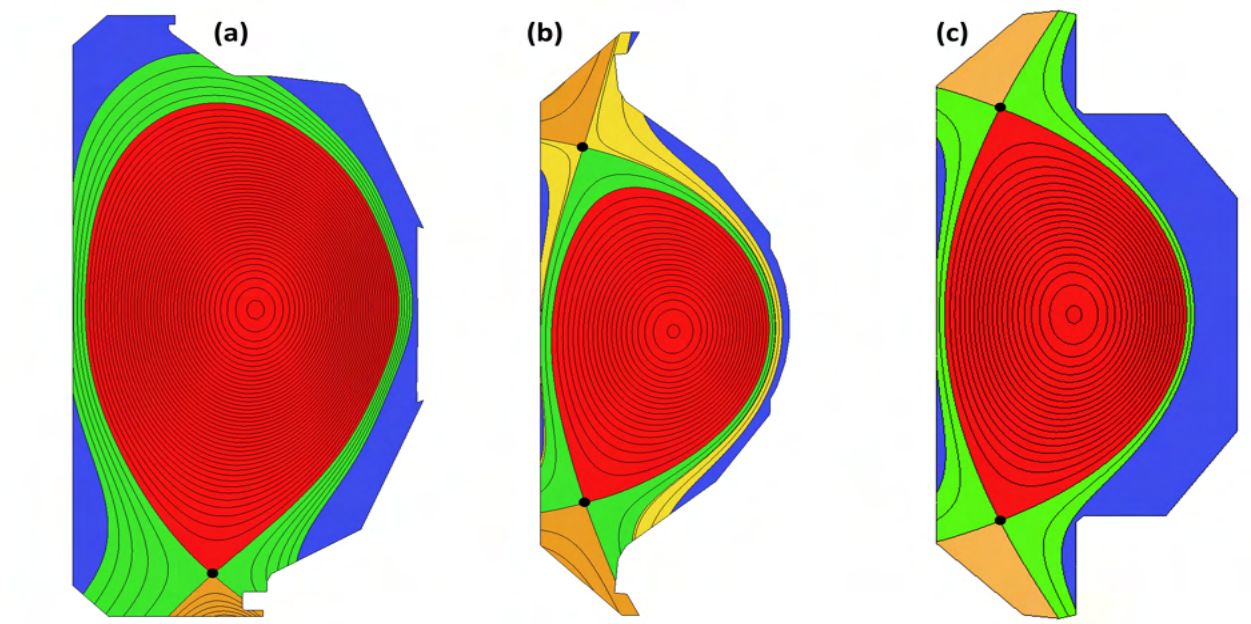

Figure 3.8: Plasma geometric model examples (a) DIII-D with one X-point, (b) Korea Superconducting tokamak Advanced Research (KSTAR) with two X-points, (c) SPARC with two X-points on the same flux curve.

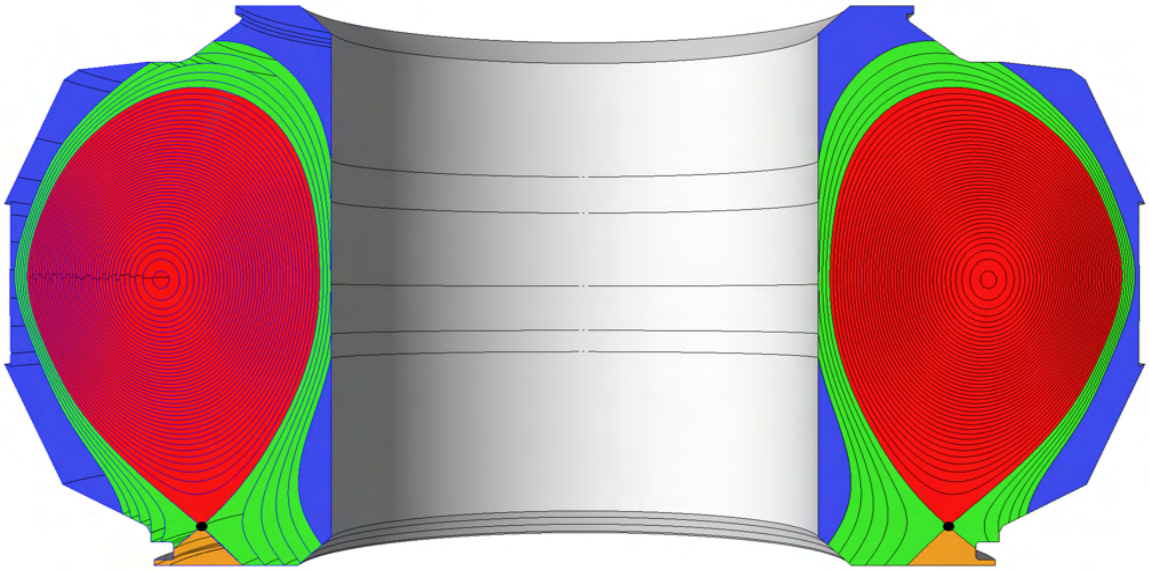

Figure 3.9: The cross-sectional view of a 3D model generated from a 2D plasma geometric model with one X-point.

will define the location of a new mesh vertex on the poloidal plane mesh. This process is continued traversing all the poloidal planes until reaching the original poloidal plane. Since the location of the field-following intersection on the original poloidal plane does not match

the location of the original vertex, this process continues for a number of toroidal traversals until the intersection on the original toroidal plane is sufficiently close to a mesh vertex that has been defined on that poloidal plane mesh.

The field-following locations of mesh vertices are traced by placing the points on the poloidal cross-section such that a magnetic field line passes through these points at each poloidal plane. The magnetic field $\boldsymbol{B}$ in terms of its components B_R, B_Z, and B_ϕ along R,Z, and ϕ directions respectively is defined as:

$$\boldsymbol{B} = B_R\hat{R} + B_Z\hat{Z} + B_\phi\hat{\phi} \tag{3.3}$$

where $\hat{R}, \hat{Z}, \hat{\phi}$ represent the directions along major radius R, vertical axis Z, and toroidal axis ϕ respectively in the (R,ϕ,Z) coordinate system as shown in figure 3.2b. $\boldsymbol{B}$ for a given ψ and poloidal current density $I(\psi)$ is expressed as:

$$\boldsymbol{B} = -\frac{1}{R}\frac{\partial\psi}{\partial Z}\hat{R} + \frac{1}{R}\frac{\partial\psi}{\partial R}\hat{Z} + \frac{I(\psi)}{R}\hat{\phi} \tag{3.4}$$

The position vector of any point on the 3D field line can be expressed in a parametric form $\boldsymbol{L}(t)$ with respect to parameter t as shown in equation 3.5.

$$\boldsymbol{L}(t) = [L_R(t), L_Z(t), L_\phi(t)] \tag{3.5}$$

where its components are expressed in (R,ϕ,Z) coordinates.

If the trajectory of $\boldsymbol{L}(t)$ follows the field line, then $d\boldsymbol{L}(t)/dt$ is always parallel to the $\boldsymbol{B}$ leading to the equations 3.6, 3.7, 3.8 that represents the change in the poloidal magnetic field with the unit change of ϕ.

$$\frac{dL_R}{dt} = \frac{RB_R}{B_\phi} = -\frac{\partial\psi}{\partial Z}\frac{R}{I(\psi)} \tag{3.6}$$

$$\frac{dL_Z}{dt} = \frac{RB_Z}{B_\phi} = \frac{\partial\psi}{\partial R}\frac{R}{I(\psi)} \tag{3.7}$$

$$\frac{dL_\phi}{dt} = 1 \tag{3.8}$$

The numerical integration of the magnetic field line in equations 3.6, 3.7, 3.8 using Runge-Kutta 4th Order (RK4) method results in points $[L_R(t), L_Z(t), \phi]$ on the flux curves

that are approximately field-following. For every value of ϕ, a point $[L_R(t), L_Z(t)]$ is projected to the flux curve on the original poloidal cross-section.

The mesh vertex placement on the tokamak wall edges is based on the distance between mesh vertices on the flux curves at the model vertex at the intersection of the flux curve with the wall and an input wall maximum spacing. When the spacing on flux curves at the model vertices at the wall intersection is finer than the maximum wall spacing, the distance between mesh vertices on the wall edge is graded smoothly from the model vertex until reaching the maximum wall spacing.

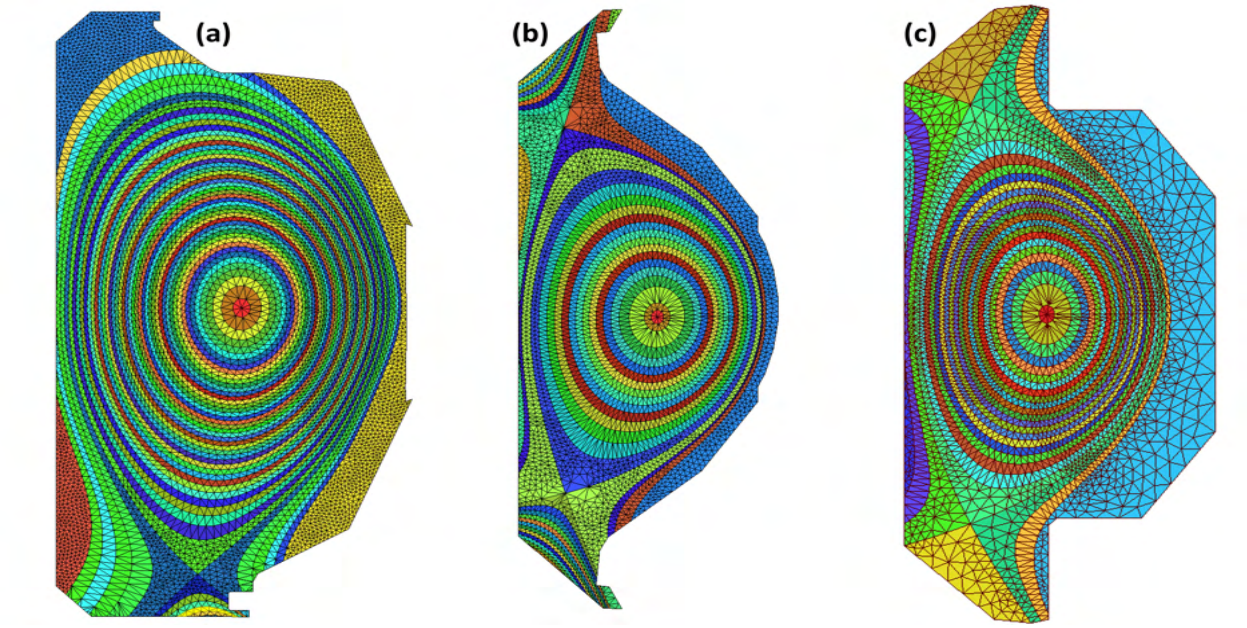

Figure 3.10: Mesh examples created with the original meshing procedures (a) DIII-D with one X-point, (b) Korea Superconducting tokamak Advanced Research (KSTAR) with two X-points, (c) SPARC with two X-points on the same flux curve.

The meshes produced created with the original meshing procedures [9] at the level of mesh coarseness shown in figure 3.10 are generally satisfactory. However, the meshes required for the recent XGC production simulations demand a much tighter spacing of mesh vertices on the flux curves relative to the spacing between flux curves. This produces elements with a much higher aspect ratio between the flux curves. Although the base meshing procedure produces the desired mesh in most of the areas between flux curves, the one-element deep meshing procedure produces unacceptable elements near X-points and in areas where open flux curves approach the tokamak walls (see figure 3.11). In particular, the application of

the one-element deep procedure results in the creation of a poor quality meshed in those areas.

Even the application of the local edge split and swap mesh modifications (previously used around the X-points) did not provide a satisfactory mesh, given the high aspect ratio elements between flux curves. To provide the level of mesh resolution and element shape quality desired in those areas of the domain, procedures that modify the plasma geometric model to provide additional model faces for the application of appropriate meshing operations are required. The procedures to update the plasma geometric model and additional mesh generation operations are described in the next section.

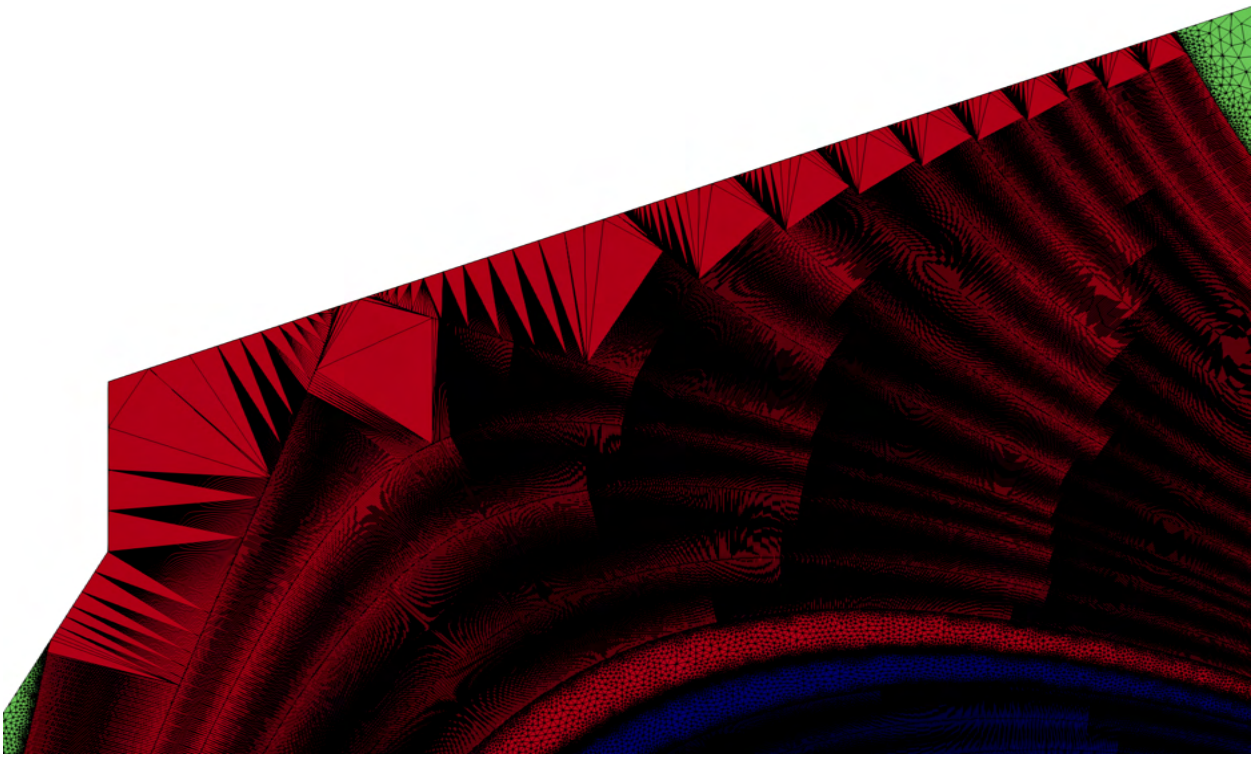

Figure 3.11: National Spherical Torus Experiment (NSTX) mesh areas with flux curves interacting with the wall curve producing unsatisfactory meshes when applying the original meshing procedures.

3.5 Model Topology Update and Mesh Generation

Although maintaining the one-element deep field-following mesh between flux curves is important for the area between the closed flux curves and most of the area around the separatrix curve and between open flux curves, the XGC models of the physics behavior

near X-points and tokamak walls allows the use of more general mesh configurations. This provides the opportunity to employ alternative mesh generation procedures in those areas that yield higher quality meshes as desired to ensure accurate and stable solution results of the mesh-based PDE solution.

Conceptually, the approach to produce the desired meshes in the problem areas is to decompose the model faces between open flux curves and around X-points into multiple faces. One face, which would consist of most of the original plasma geometric model face, would continue to employ the one-element-deep meshing procedure. The remaining portions of the original plasma geometric model face employ a general-purpose unstructured mesh generator that is suited to producing well-controlled meshes, such as the one that is already applied to model faces bounded by a single flux curve and a portion of the tokamak wall. The application of this procedure alone is not satisfactory in that it still has the problem of an immediate transition from a high aspect ratio mesh configuration to a more general low aspect ratio element configuration as shown in figure 3.12. This problem is eliminated by the introduction of a doubling transition procedure that is applied to transition from the high aspect ratio one-element deep elements to a lower aspect ratio unstructured mesh.

3.5.1 Splitting Model Face Between Flux Curves

Given the plasma geometric model as constructed in section 3.3, the set of model faces between two flux curves that include portions of the tokamak wall (figure 3.13a) and/or X-points (figure 3.13b) are identified as the model faces that need to be decomposed into a set of model faces in which the one-element deep procedure, the doubling transition procedure (described below), and general unstructured mesh generation are applied. The goal is to maintain the one-element deep structure for as much of the area between flux curves as possible while being sure to transition to lower aspect ratio elements in the specific areas where the mesh would not be satisfactory with the application of only the one-element deep procedure. The key factors in determining the portions of the base model face between two flux curves to which the doubling transition and unstructured mesh procedures need to be applied are a function of the portion of the tokamak wall geometry that is part of the base model face between two flux curves, or how much wider the base model face between flux curves is in the vicinity of an X-point.

The model faces are decomposed into a set of new model faces by introducing model

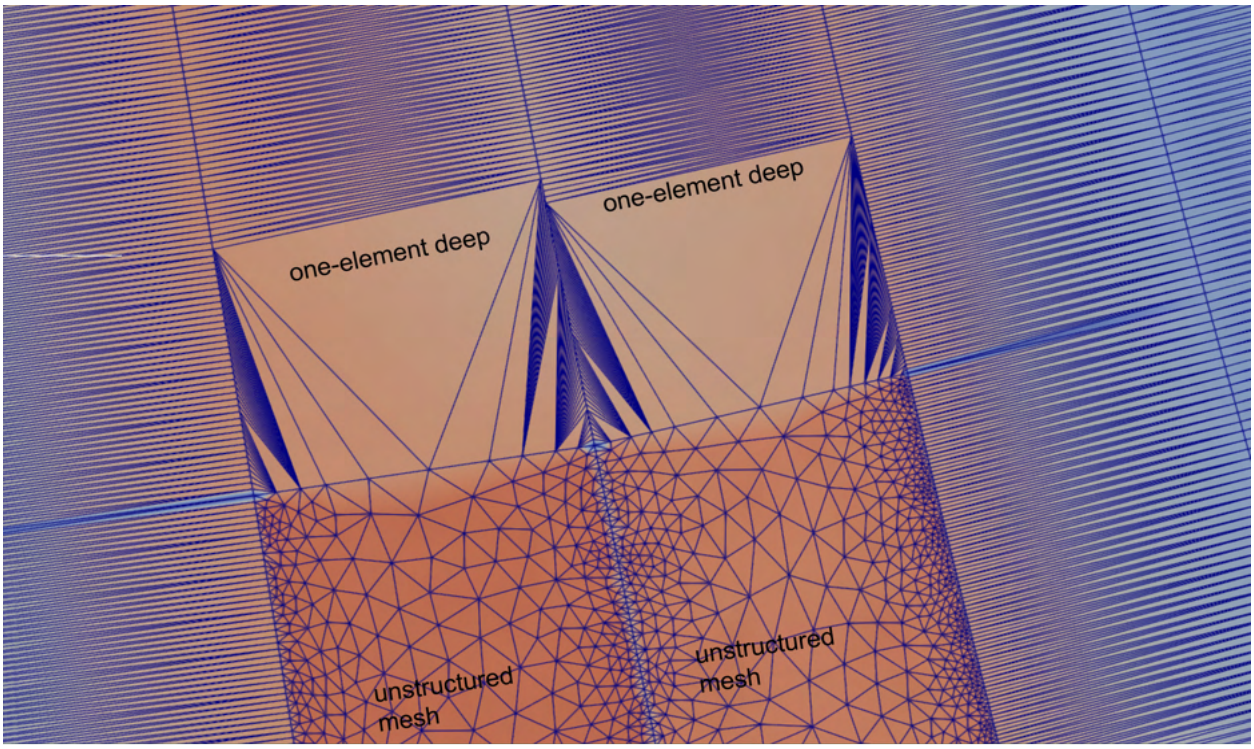

Figure 3.12: The immediate transition from one-element between flux curves to an unstructured mesh using the given flux curve mesh vertex spacing will yield unsatisfactory meshes, as shown here.

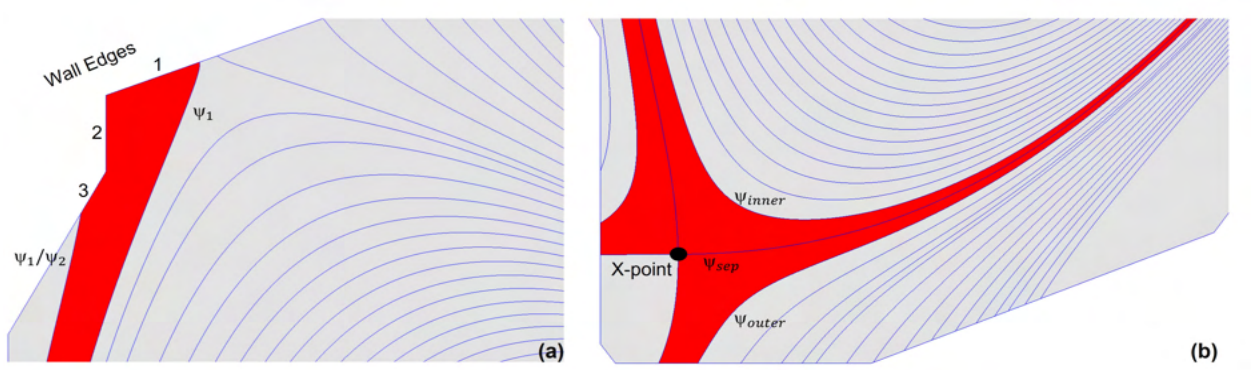

Figure 3.13: Types of model faces identified for doubling transitioning meshes, (a) a model face bounded by two flux curves ψ_1 and ψ_2 and is adjacent to the wall edge, (b) model faces bounded by two flux curves ψ_{sep}, ψ_{inner}, and ψ_{sep},ψ_{outer} respectively and are adjacent to the X-point.

edges across the face between two flux curves. Two new model edges are inserted to split the face into three model face types; one-element-deep mesh face, transition mesh face(s), and unstructured mesh face(s). The first model edge is introduced on the face at a distance 'd' from the tokamak wall or X-points. The distance d is a function of the length of the tokamak

wall on the base model face or the width of the base model face in the vicinity of the X-point for the model faces that are adjacent to X-point. The model edge is introduced on the face by inserting two model vertices v_1 and v_2 on the flux curves ψ_1 and ψ_2 respectively such that the new model edge is bounded by both vertices. This model edge splits the face into two model faces, an unstructured mesh face and a one-element-deep mesh face as shown in figure 3.14a. In case of faces adjacent to the X-point (except the ones in private region), a vertex v_s is inserted at a distance d (function of d_{min} in figure 3.15) on the two legs of X-points which are not bounding private region. The model vertices v_i and v_o are inserted on the inner flux curve (ψ_i) and outer flux curve (ψ_o), respectively, by finding the nearest field following point on these flux curves to v_s. The two model edges (one bounded by v_i and v_s, and the other by v_s and v_o) split the corresponding faces into one-element-deep mesh faces and an unstructured mesh face, as shown in figure 3.15. From this point, the newly created faces are meshed with the doubling transition and unstructured mesh procedures.

The second model edge is introduced at a distance 'd_l' from the first model edge, where d_l depends on the number of layers N required for the transition from the highly anisotropic one-element-deep mesh to a more isotropic mesh. The number of transition layers N depends on the approximate aspect ratio (A_r) of mesh elements on the one-element-deep mesh face before transitioning starts and is defined as $N = log_2(A_r)$. The distance d_l is evaluated by moving N number of field-following points on the flux curve from the starting point (v_1 in the figure 3.14b). This second model edge is bounded by new model vertices v_a and v_b as shown in figure 3.14b.

To ensure the field-following property of the flux curves, every new model vertex added to the flux curves is placed at one of the field-following points (already specified on flux curves in the base model) on the flux curve.

3.5.2 Doubling Transition Procedure

The smooth transitioning of the mesh from the high aspect ratio elements produced by the one-element deep meshing procedure to unstructured mesh is executed by applying a doubling transition. The doubling transition is applied as many times as needed to produce a number of mesh vertices between the flux curves that will allow the effective application of general unstructured mesh generation for the remainder of the face between the two flux curves. Figure 3.14c-e indicates the process of defining a set of mesh edges and vertices that

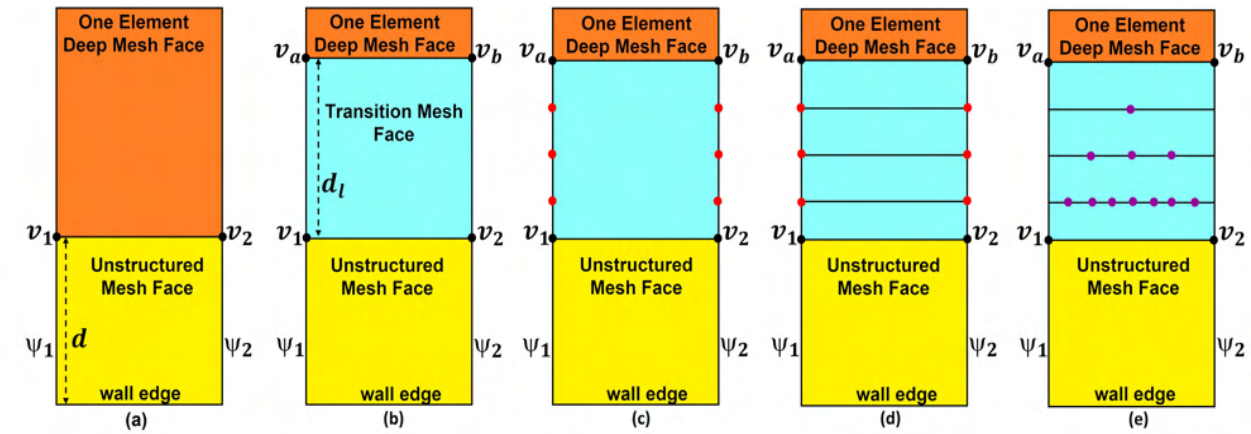

Figure 3.14: Schematic view of 5 steps of decomposing a based model face into three model faces to be meshed by the one-element-deep, doubling transition, and unstructured meshing procedures. (a) Introduction of the face (yellow) for unstructured mesh meshing, (b) introduction of the face (blue) needed for the doubling transition mesh, (c) vertices (red) added to the flux curves, (d) transition layers added, (e) vertices (purple) specified on the layers used by the doubling transition procedure.

supports going from the high aspect ratio elements between the flux curves to a number of mesh edges across the face yielding elements with a more acceptable aspect ratio for use by a general triangulation procedure.

Figure 3.16 shows an example of the process of going from the one-element-deep mesh (area a) to a satisfactory unstructured mesh (area c) by the application of four layers of the doubling transition operator (area b). The number of doubling transitioning layers required for smooth transition depends on the ratio of the distance between the field-following mesh vertices on the flux curves to the distance between two flux curves at the location where the doubling transition operator is applied. The goal is to add the transition layers until the spacing of mesh vertices on the last transition layer is roughly equal to the spacing between the field-following mesh vertices on the flux curves.

3.5.3 Comparison Meshes

The splitting of model faces between flux curves, followed by the application of the doubling transition operator, aims to produce satisfactory meshes for a full range of desired tokamak geometries and mesh configurations. The base mesh generation procedures presented in reference [9] have limitations in producing meshes of satisfactory quality due to the need for high aspect ratio elements between flux curves for effective XGC simulations.

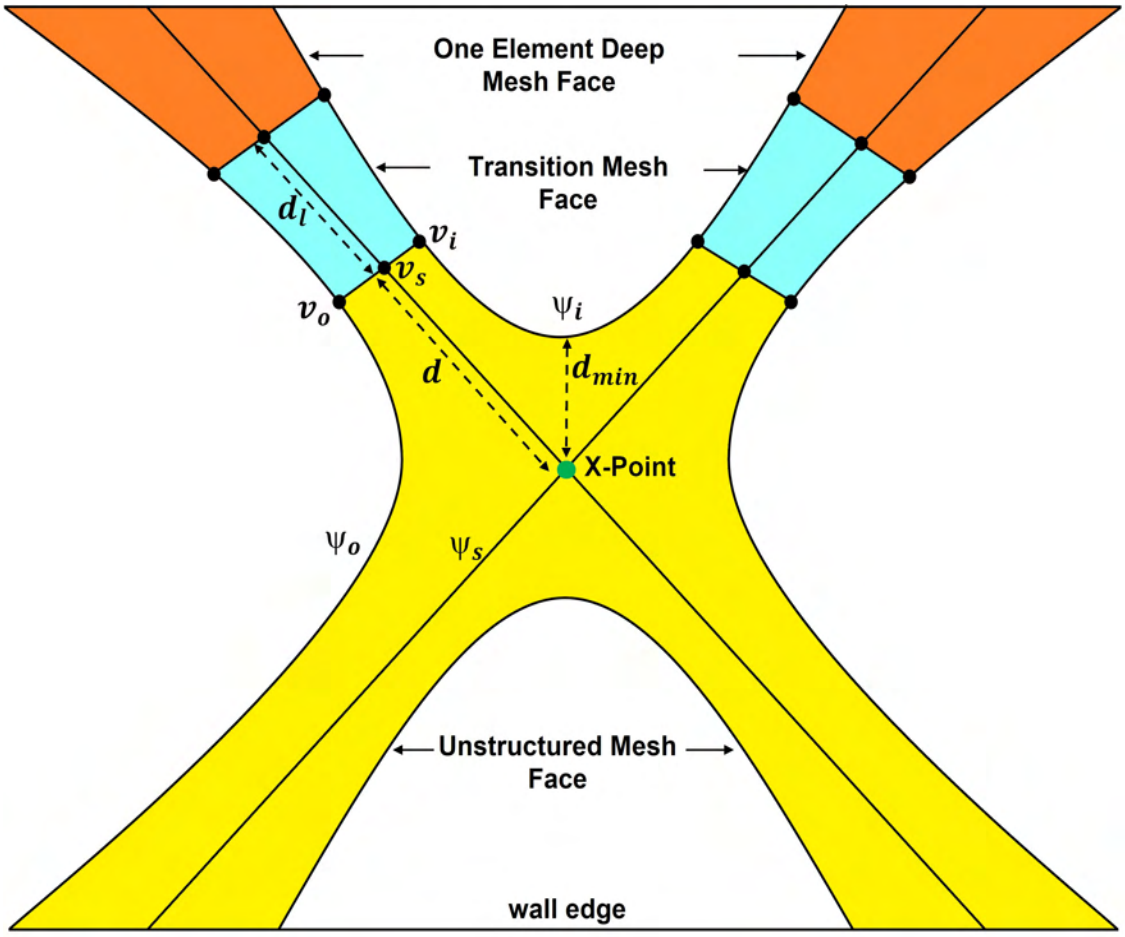

Figure 3.15: Splitting of model faces adjacent to the X-point into a set of model faces to be meshed by the one-element-deep, doubling transition, and unstructured meshing procedures.

Figure 3.17 shows a close-up of a portion of a production simulation mesh produced by the base mesh generation procedure compared to the mesh generated using the modifications and extensions indicated in this section. Figure 3.18 shows a comparison of portion of a mesh for a physics simulation in the areas adjacent to the X-point meshed by the original mesh generation procedure with the mesh generated using the procedures described above.

Figure 3.19 shows a comparison of ratios of the area of the largest elements to the area of the smallest element in its neighborhood (elements sharing an edge) on the flux faces using original and current meshing procedures. The ratios were consistent in the core region inside the last closed curve. The black curves in figure 3.19 represent the ratios with the updated meshing procedures where transition layers were used. It can be seen that the peak value of the ratio is 2.03 and the ratios are consistent throughout the domain. On the other hand, it

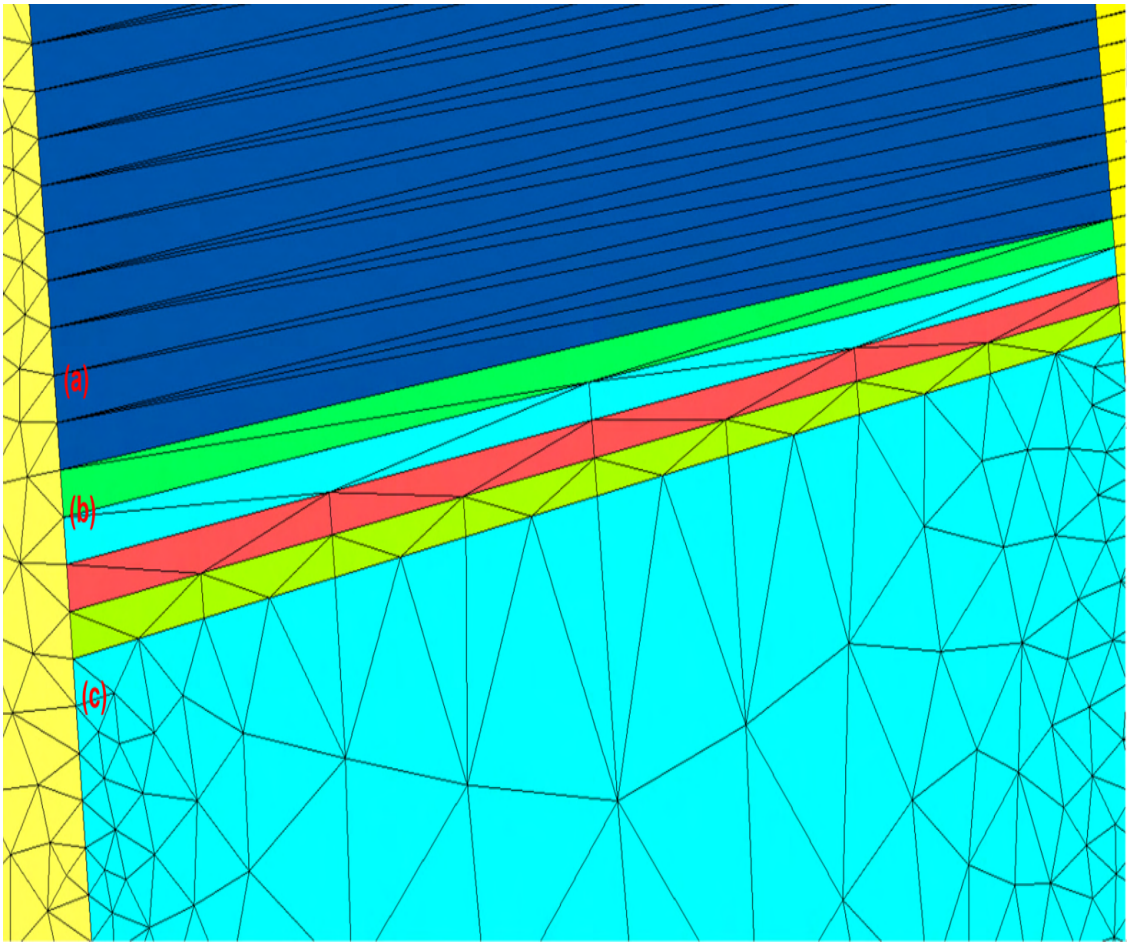

Figure 3.16: Close-up of the use of the doubling transition mesh (area *b*), to smoothly transition the mesh from the anisotropic elements in the one-element between flux curves (area *a*) to the unstructured mesh (area *c*).

can be seen from the red curve that the ratios are inconsistent throughout the plasma edge region with very sharp jumps in values. The peak value of the ratio is 381.59 for the element highlighted in the figure 3.20. This peak value is much higher than the peak value of 2.03 achieved from the updated meshing procedure.

Figure 3.21 demonstrates the benefit of the new meshing procedures in XGC simulations. For this demonstration, we exercised an axisymmetric (i.e., without micro-turbulence), electrostatic XGC simulation using the geometry of the National Spherical Tokamak Experiment (NSTX). The axisymmetric electrostatic potential in this situation is expected to vary slowly (> 0.1m) along the flux curves away from the material wall, but very sharply (ion Larmor radius scale) in close proximity to the wall. Note that XGC does not resolve the Debye sheath, but instead uses a logical sheath boundary condition [48] for the marker par-

ticles. The mesh produced with the new meshing procedures figure 3.21b reproduces this expected behavior much better than the mesh produced with the original meshing procedure figure 3.21a. The improved solution of the electrostatic potential, in turn, improves the accuracy of the marker trajectories and affects the accuracy of physics observables such as the divertor heat load. A reduction of the ion particle noise is observed as well. Figures 3.21c and 3.21d show the standard deviation of the marker particle weights divided by the mesh vertex volume (a measure for the sampling error of the density) for the original (c) and new (d) meshing procedures. Reduced marker noise near the wall improves the accuracy of physics observables on the wall and of the neutral particle recycling model [49].

The close-up mesh images shown in figures 3.17 and 3.18 show a local increase in the total number of elements in those areas of the mesh. It should be pointed out that the new meshing procedures only increase the total number of elements in the mesh no more than a couple of percent. Thus, there is only a small increase in the total XGC execution times between the two meshing procedures.

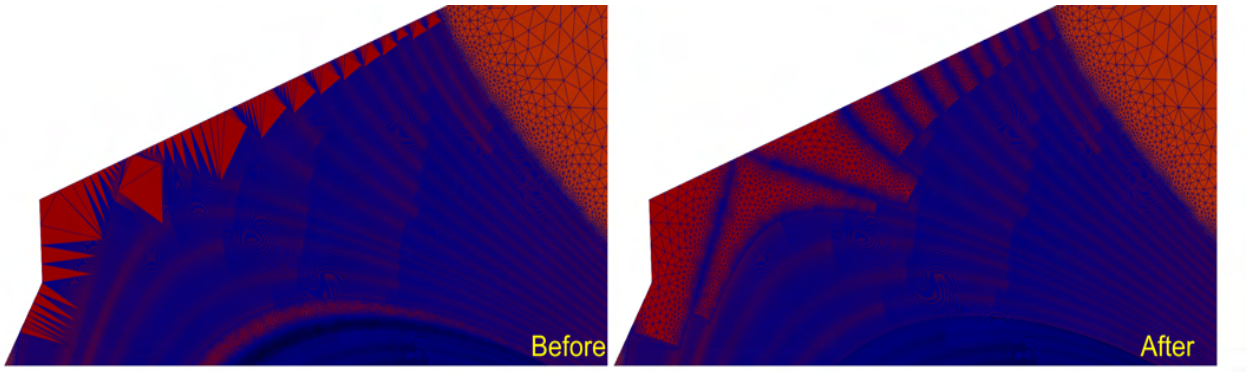

Figure 3.17: Comparison of previous and current meshing procedures on a National Spherical Torus Experiment (NSTX) production simulation mesh. (Before) re-shows the portion of the mesh from figure 3.11 meshed with the base meshing procedure, while (After) shows that portion of the domain meshed with the updated meshing procedure presented in this paper.

3.5.4 Executing the Mesh Generator

The geometric model provided to the mesh generation procedures consists of the physical geometry of the wall curves and the physics geometry of the core O-point, the X-points within the wall geometry, the separatrix curves, and the selected flux curves. Given the appropriate GEQDSK equilibrium file with discrete wall geometry and a set of input pa-

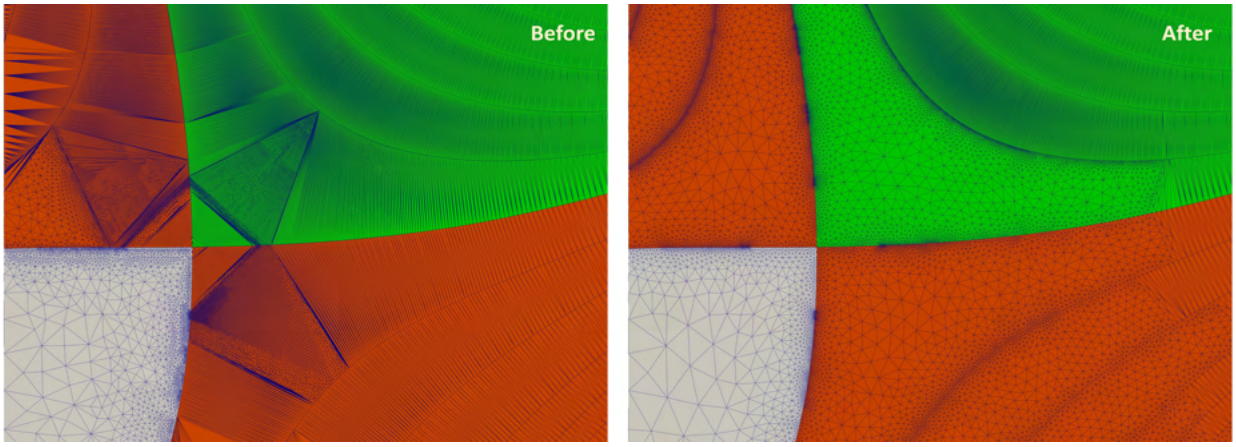

Figure 3.18: Comparison of original and current meshing procedures on a National Spherical Torus Experiment (NSTX) production simulation mesh. (Before) shows the portion of mesh on faces adjacent to the separatrix curve meshed with the original meshing procedure, while (After) shows the same area meshed with the meshing procedure presented in this paper.

rameters that define the desired set of flux curves, the geometric model is automatically constructed in a matter of seconds in serial using the procedures described in section 3.3 in combination with the procedures presented in section 3.5.1. The set of flux curves in the geometric model can be defined in a number of ways, including (a) uniform spacing in terms of normalized ψ value, (b) uniform spacing in terms of real distance in meters, (c) spacing in terms of local ion gyro-radius, and (d) list of normalized ψ values for desired flux curves.

During the process of constructing the geometric model, the various geometric model regions are attributed to indicate the specific mesh generation tool to be used to mesh that area. The geometric model entities are also attributed with mesh control parameters indicating node point placement and mesh gradation parameters based on a set of user-provided input parameters. The user-provided mesh control parameters consist of spacing between nodes on flux curves, node spacing control near X-points, and mesh size in unstructured mesh regions. The node spacing on the flux curves can be defined in two ways: (a) constant uniform spacing between nodes on all flux curves defined by a single value and (b) a list indicating node spacing on each flux curve for better control. The actual mesh generation process is fully automatic and takes a manner of seconds in serial.

The two options available for the execution of TOMMS are terminal execution, where the inputs are provided in an editable ASCII file, and graphical user interface (GUI), where input parameters are set interactively. Expert XGC users prefer the former method, whereas

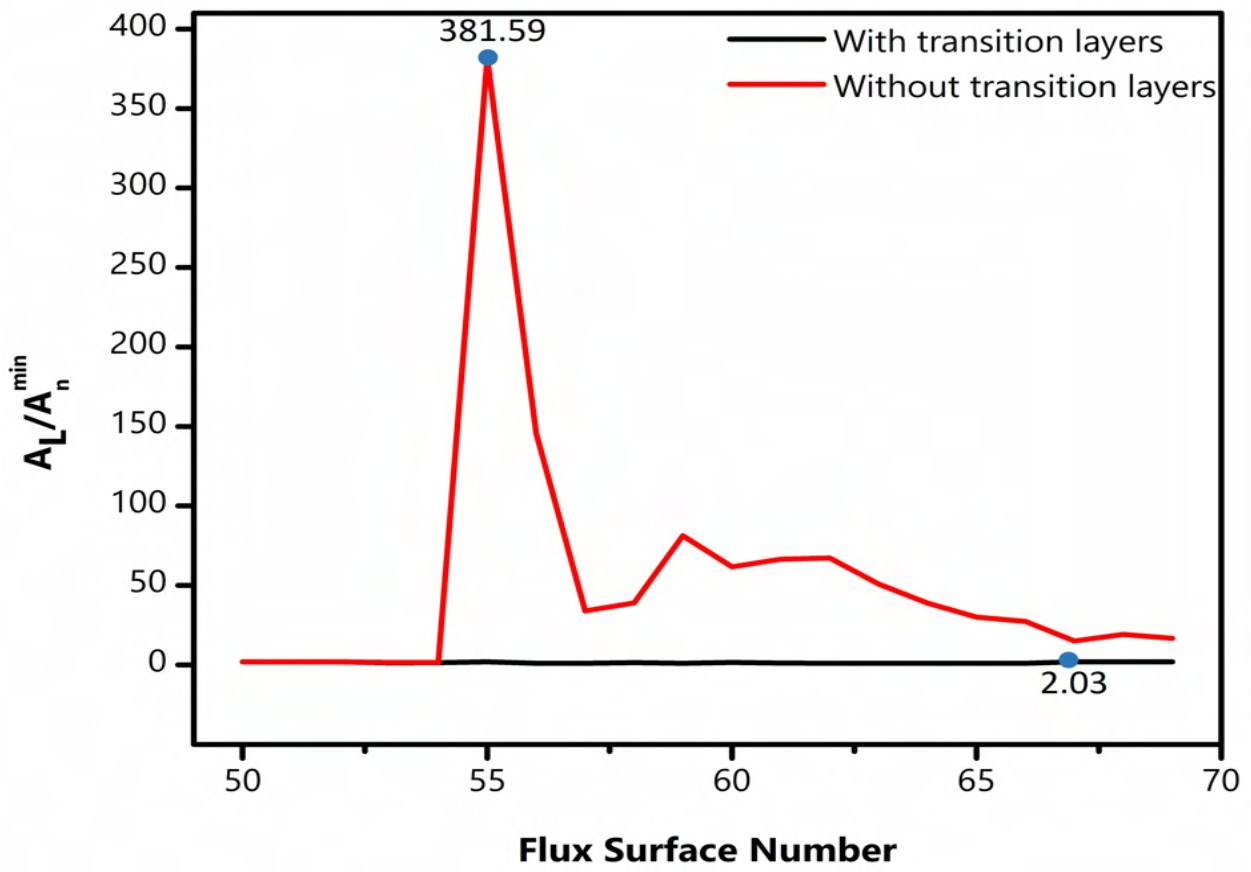

Figure 3.19: Comparison of the ratio of the area of an element with largest area (A_L) to the area of an element with minimum area (A_n^{min}) of n neighboring elements using original and current meshing procedures on a National Spherical Torus Experiment (NSTX) production simulation mesh. The number of flux surfaces increases from inner most surface (bounding magnetic axis) to the outermost flux surface.

new users tend to prefer GUI since it provides quick feedback on the mesh. The time required to set up the input parameters ranges from one minute to a few, depending on the desired control of flux curves and node spacing on these flux curves. To define flux curves using a uniform spacing between flux curves and uniform node spacing requires modifying only two parameters. On the other hand, for advanced control of flux curves (preferred by expert XGC users), a list of flux curves and the desired node spacing on each flux curve which requires manually adding each flux curve value in the input file, is provided to TOMMS. Once the input parameters are set, executing TOMMS to generate a model and mesh takes a few seconds in serial. The details on the use of TOMMS, both from terminal and GUI, along with the details on input parameters, can be found in reference [40].

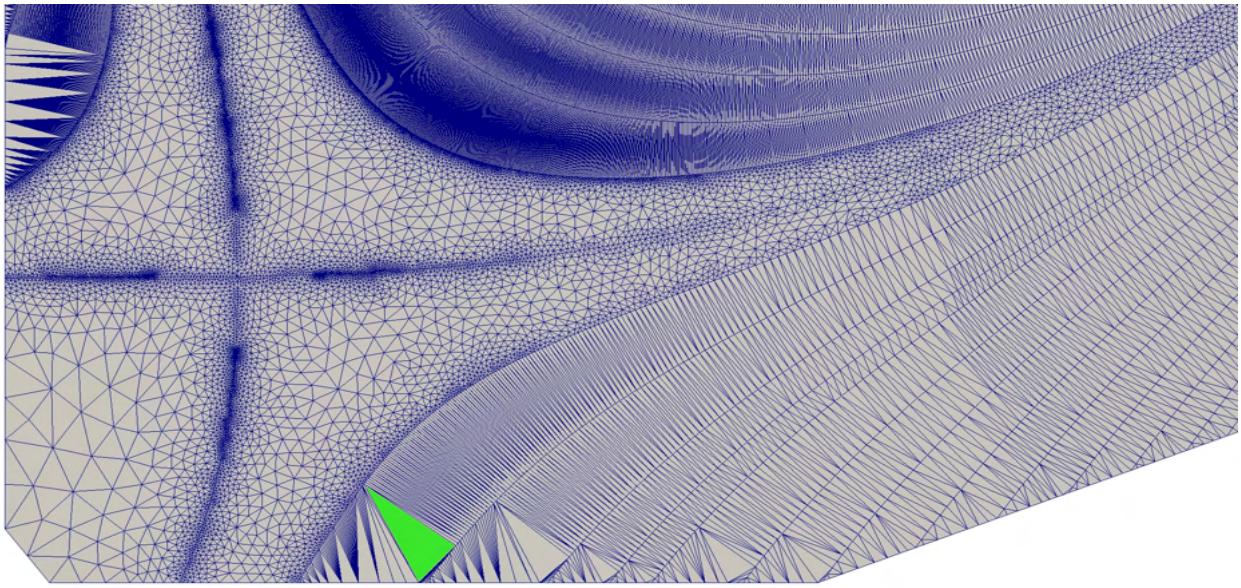

Figure 3.20: The element highlighted in green with the peak value of ratio A_L/A_n^{min}.

3.6 Providing Mesh Information for Efficient XGC Execution

The execution of operations within an XGC simulation requires ordering and structuring the mesh information as needed to support the effective execution of the XGC computations.

3.6.1 Mesh Data for XGC Mesh Based Operations

The ability of XGC to execute the field-following related calculations, such as flux surface averaging and field-following Fourier filtering, requires specific ordered data related to the set of mesh vertices on each of the flux curves. In addition, other operations take advantage of the one-element between flux curves property of the mesh faces placed between flux curves. As discussed in section 3.5, the one-element deep property is lost for portions of the mesh between flux curves adjacent to X-points and for open flux curves as they approach the tokamak walls. In these areas, specific knowledge of the mesh vertices that have been placed between the flux curves to support the generation of acceptable meshes is needed.

The TOMMS mesh generator employs a mesh data structure that maintains explicit knowledge of the association of each mesh entity (mesh vertex, edge, and face for a 2D mesh) with the appropriate model entity (model vertex, edge, and face for a 2D model) [50]. This

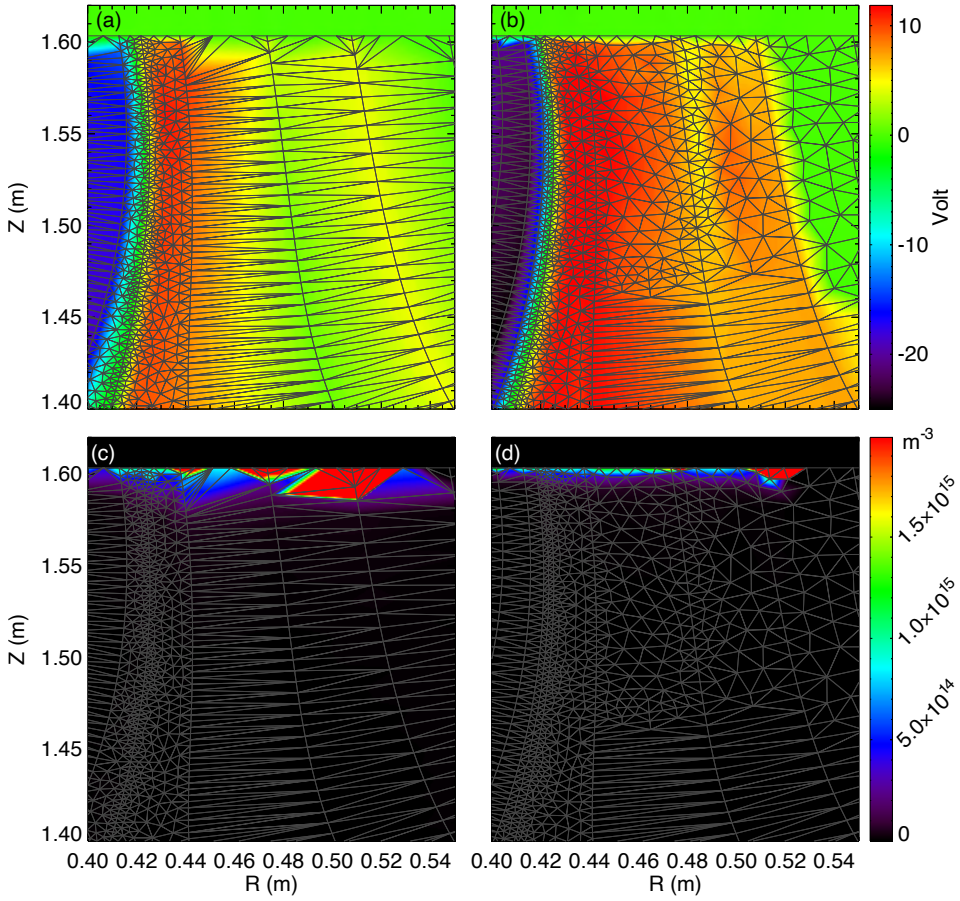

Figure 3.21: Electrostatic potential from an axisymmetric, electrostatic XGC simulation using (a) the original and (b) current meshing procedures. Using the mesh produced with the new meshing procedures better reproduces the expected transition from slow variation along the flux-surfaces away from the material wall to very short (ion Larmor radius) variation close to the wall. In addition, the new meshing procedures reduce particle noise near the material wall as shown by the standard deviation of the ion marker weights divided by the mesh vertex volume for (c) the original and (d) new meshing procedures.

association information is referred to as mesh classification. The mesh data structure also contains a complete mesh topology so it can efficiently provide information of any mesh entity adjacency (e.g., the mesh vertices bounding a mesh edge, the edges bounded by a mesh vertex, the mesh edges bounding a mesh face, etc.). This information supports the efficient determination of the specific mesh information needed by XGC, which includes:

1. Mesh vertices bounding each mesh face (the standard mesh element connectivity information).

2. Ordered list of mesh vertices on each flux curve as shown in red in figure 3.22.

3. Unstructured mesh vertices classified on the model faces that are bounded by two flux curves and also on the model faces that are bounded by a single flux curve and wall curve as shown in blue and green respectively in figure 3.22.

4. Mesh vertices on the wall curve as shown in maroon in figure 3.22.

5. Mesh vertex classified on the X-point model vertex as shown in orange in figure 3.22.

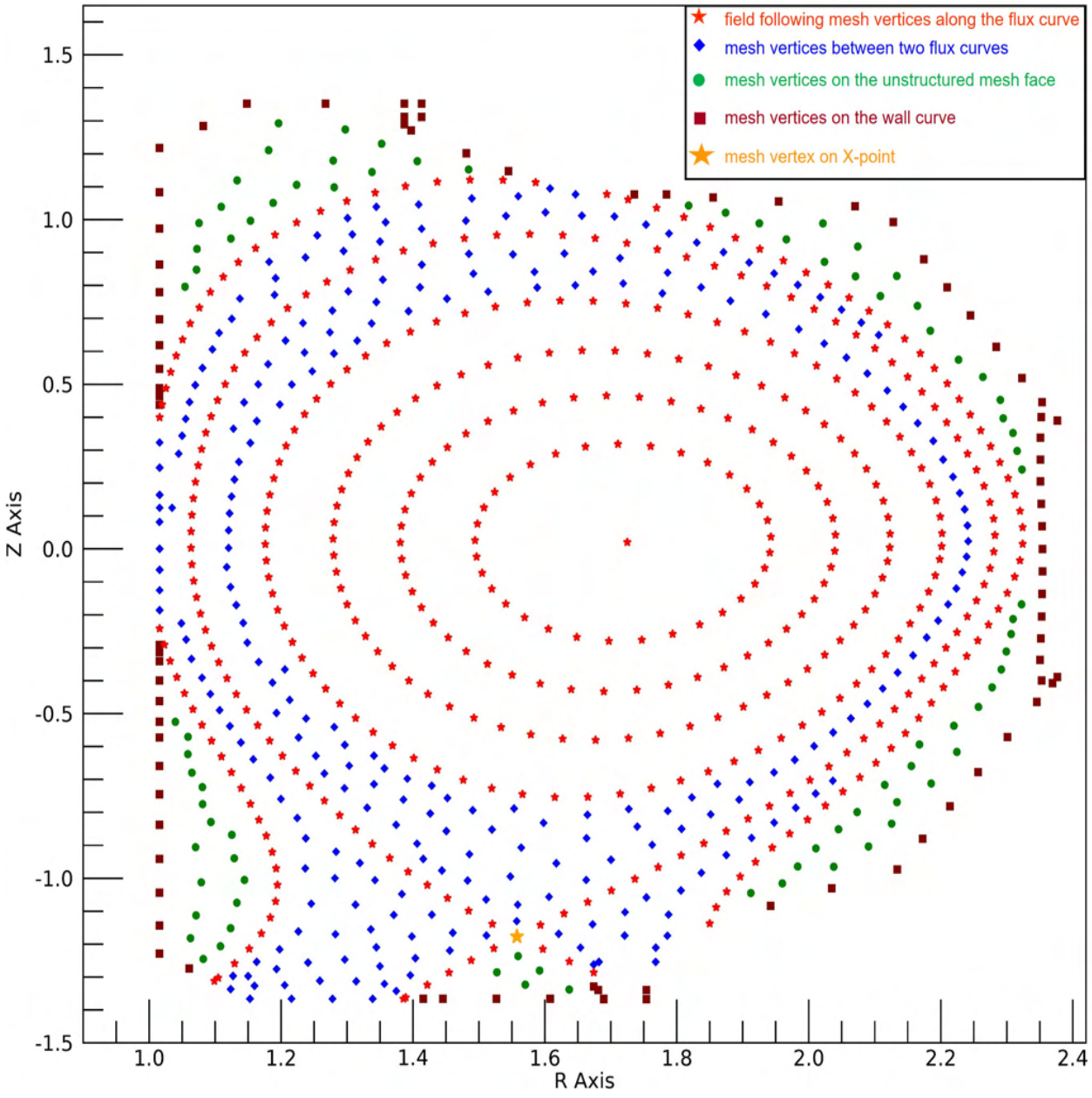

Figure 3.22: Classification of mesh vertices in XGC.

The ordered list of mesh vertices on each flux curve segment is provided using the mesh classification information. The procedure uses a traversal of the mesh edges and collects the set of mesh edges and their bounding vertices on the closure of the model edges that are part of the flux curves. In the case of closed flux curves, the flux curve is represented by a single closed model edge. In this case, the ordered list of mesh vertices is determined by executing a traversal starting at the mesh vertex classified on the single model vertex for that closed model edge. Using mesh vertex to edge mesh adjacency, the mesh edge classified on that model edge using the mesh vertex is found. The other mesh vertex bounding that mesh edge is then the next mesh vertex. The process is continued until the starting mesh vertex is reached. In the case of an open flux curve, the starting point is a mesh vertex classified on a model vertex that is classified on a model edge that is a wall edge. In this case, the traversal terminates when the next vertex found is classified on a wall curve.

The mesh vertices that are classified on the model faces bounded by two flux curves must be collected and associated with the appropriate flux curve. To determine if a model face is between two flux curves, the model edges bounding that model face are traversed, and if two different ψ values are found for the model edges that are part of flux curves in the face loop, then the model face is between those two curves. The mesh vertices classified on the model face between two flux curves are collected and evaluated to determine their association with the flux curve it is closest to. The algorithm for the above procedure is presented in algorithm 2.

3.6.2 Ordering Mesh Data for Efficient Cache Performance

The computational efficiency of numerical operations on GPU-accelerated parallel computers is strongly dictated by the time spent accessing the data required for the execution of stimulation. The "closer" the data is, the more efficient the access will be, with access to data in the various levels of cache memory being orders of magnitude faster than access to data in main memory. Within XGC, the placement of mesh data in memory is based on the labeling that data is given by the mesh generation procedure. Thus, it is desirable to assign labels to the mesh vertices and faces that will optimize memory access.

Due to the field-following aspects of the XGC operations, a potentially advantageous labeling of the mesh vertices and faces is to label the vertex at the core O-point first, and then selecting a mesh vertex on the first flux curve and labeling it followed by labeling the

Algorithm 2 Mesh Vertices Classified on Face

```
1: procedure MeshVerticesOnFace(model)
2:     N ← number of model faces
3:     for i = 1 to N do
4:         NumEdges = Number of Edges on the face F(i)
5:         for j = 1 to NumEdges do
6:             if E(j) = OnWall then
7:                 continue
8:             end if
9:             ψ ← psi value on E(j)
10:            push ψ to setPsi                    ▷ set will only save unique values
11:        end for
12:        S ← size of setPsi
13:        if S = 2 then                           ▷ face bounded by two flux curves
14:            NumVert = Number of Vertices on F(i)
15:            for k = 1 to NumVert do
16:                if V(k) = OnFluxCurve then
17:                    continue
18:                else if V(k) = OnFace or V(k) = OnWallEdge or V(k) = OnModelVer-
   texOnWall then
19:                    Indx ← Index of V(k)
20:                    Flux Curve 1 ← closest flux curve to V(k) with ψ = setPsi(1)
21:                    Flux Curve 2 ← closest flux curve to V(k) with ψ = setPsi(2)
22:                    Report Indx along with flux Curve 1 and flux Curve 2
23:                end if
24:            end for
25:        end if
26:        if S = 1 then                ▷ face bounded by single flux curve (wall face)
27:            NumVert = Number of Vertices on F(i)
28:            for k = 1 to NumVert do
29:                if V(k) = OnFluxCurve then
30:                    continue
31:                else if V(k) = OnFace or V(k) = OnWallEdge or V(k) = OnModelVer-
   texOnWall then
32:                    Indx ← Index of V(k)
33:                end if
34:            end for
35:        end if
36:    end for
37: end procedure
```

other vertices on that flux curve in order, based on the edge adjacent to the last labeled vertex, until all mesh vertices on that flux curve are labeled. As the mesh vertices on the flux curve are being labeled, the mesh faces that are bounded by the mesh vertices on the flux curve are also labeled. Afterward, the mesh vertices on the next flux curve and the mesh faces between the two flux curves are labeled, starting with a mesh vertex that bounds an edge with the first mesh vertex labeled on the previous flux curve bounds. This process is easily supported for the closed flux curves up to the last closed flux curve before the first separatrix using the classification and mesh adjacency information stored in the mesh data structure used in TOMMS. After that point, the loss of the one-element deep between flux curves property, as well as having general mesh between flux curves and wall, means that an alternative method must be applied to label the mesh vertices and faces outside the last closed flux curve before the first separatrix.

The desired labeling of the remaining mesh vertices and faces would continue to follow the strategy of labeling them "spiraling out" from the core O-point. The procedure implemented to do this is built on the adjacency-based ordering algorithm presented in reference [51]. That adjacency-based algorithm employs a queue of the mesh vertices such that the unlabeled mesh vertices on the other end of the edge bounding the recently labeled vertices are put in the queue.

In the application cases given in reference [51], that queue was initiated at the start of the process with a single mesh vertex on the outer boundary of the domain. To have that algorithm produce the "spiraling out" labeling desired for the XGC meshes, that queue is initiated with the ordered set of mesh vertices on the last closed flux curve before the first separatrix. Since the procedures will label the entities in the queue in order and add unlabeled edge adjacent vertices to the queue, this ordering process will continue to label in a spiraling manner.

Figure 3.23 shows a coarse mesh example of the resulting reordering of the mesh vertices. Initial testing indicates that this ordering does improve memory access time thus improving particle push execution time by 10% on average. However, it had only a marginal effect on total XGC solution time over using the previous less optimized ordering where the ordering of the mesh entities outside the last closed flux curve before the first separatrix was more random. It was determined that was a small increase in load imbalance countered the memory access improvements. Additional investigation is underway to consider poten-

tial changes to the load balancing scheme so take better advantage of the memory access benefits.

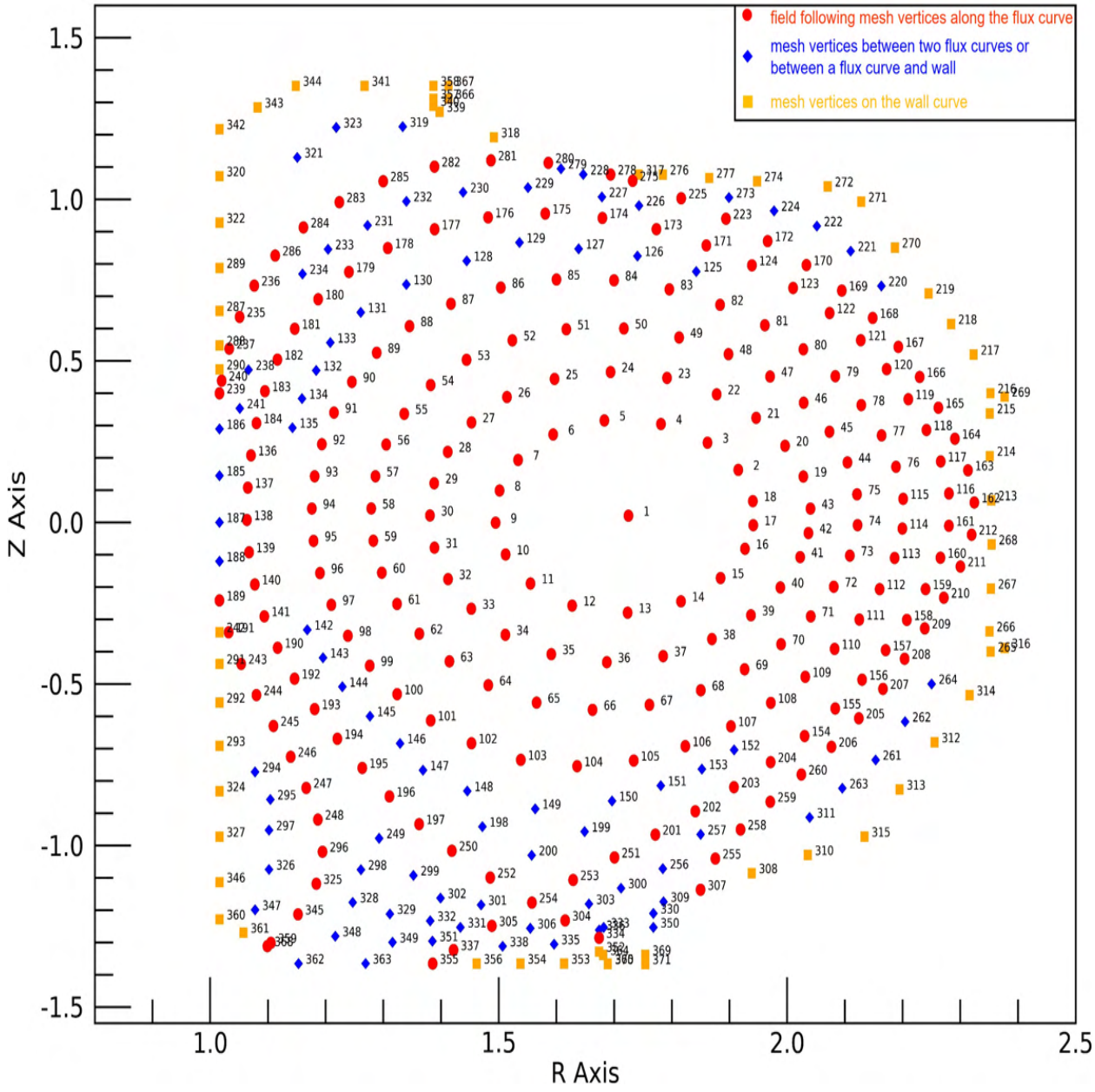

Figure 3.23: "Spiral ordering" of mesh vertices for a very coarse mesh example.

3.7 TOMMS Graphical User Interface

The TOMMS graphical user interface (GUI) is supported through Simmetrix Simmodeler. The TOMMS GUI can be accessed through the Fusion tab in Simmodeler as shown in

figure 3.24. The Fusion tab in the Simmodeler allows the users to interactively provide the inputs, set the modeling and meshing parameters, and control the outputs. The TOMMS GUI requires two inputs at minimum, the first one is the TOMMS binary executable, and the second is the plasma equilibrium file (geqdsk). With these two files provided, TOMMS GUI can generate a mesh with the default modeling and meshing parameters. The users can modify these modeling and meshing parameters using the "Meshing Parameters" tab as shown in figure 3.25.

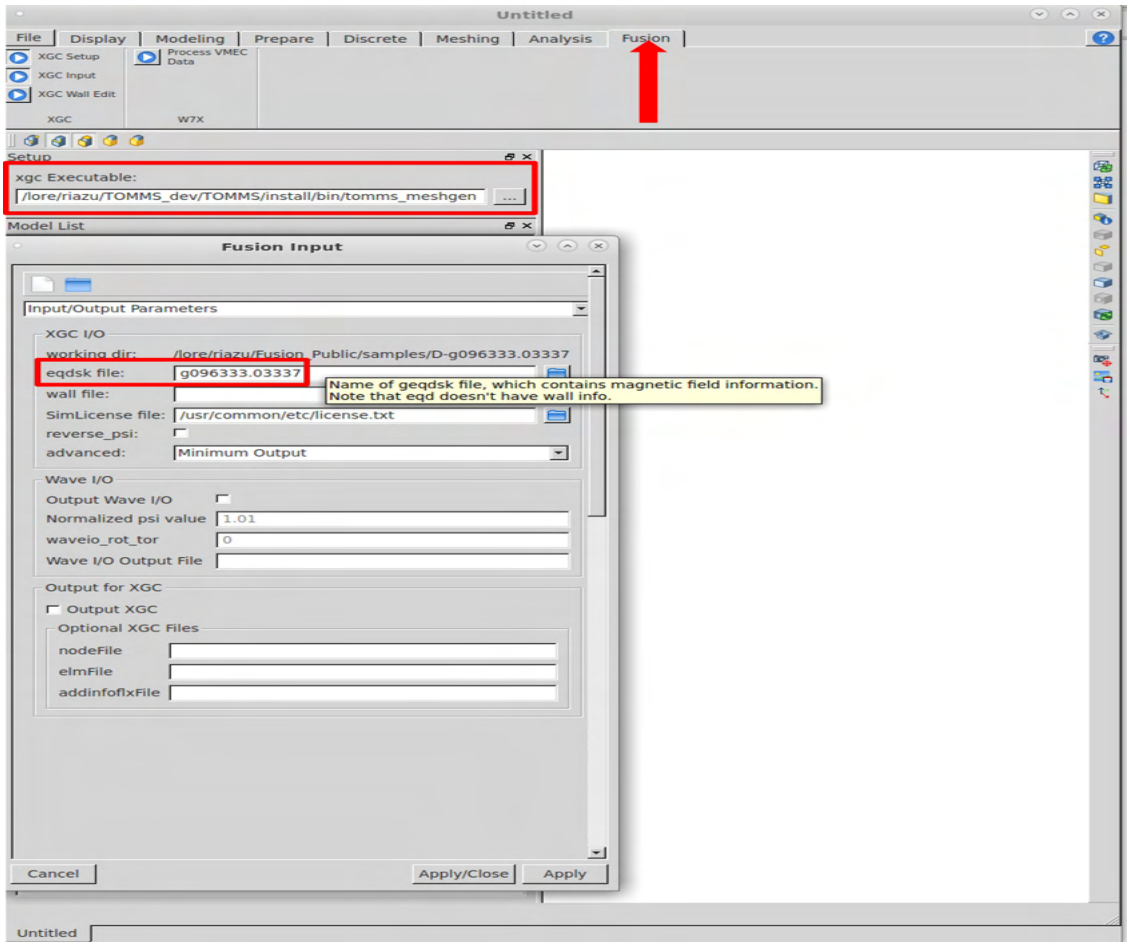

Figure 3.24: Fusion tab in Simmetrix Simmodeler with input and output options.

The TOMMS GUI is a convenient way for new users to generate meshes since it provides quick feedback on the model and mesh, and also provides a way to interactively modify it by changing different input parameters. A sample DIII-D model and corresponding relatively coarser mesh using TOMMS GUI are presented in figure 3.26. Moreover, the GUI gives the model adjacency information along with the modeling attributes and geometric properties.

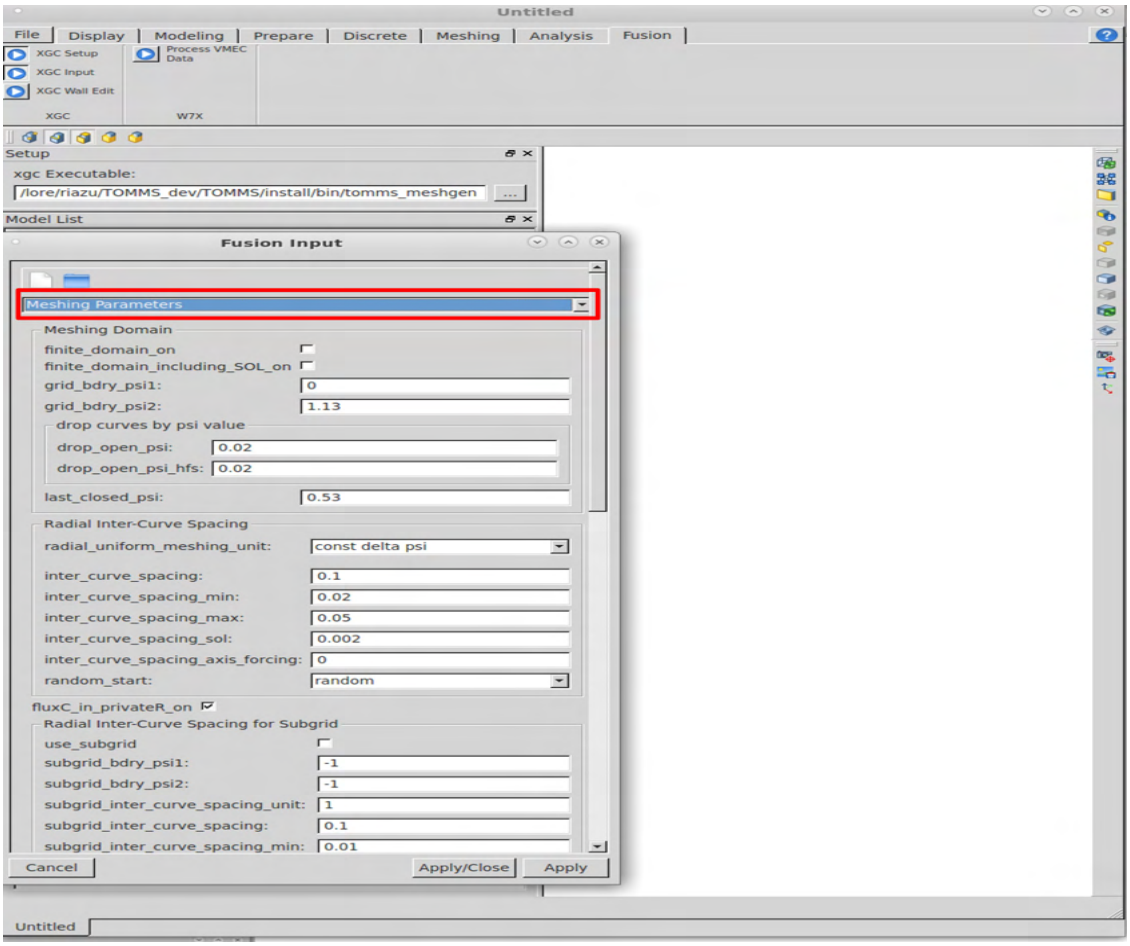

Figure 3.25: An interface to set the modeling and meshing parameters in TOMMS GUI.

This is a helpful feature in debugging the issues both for developers and users. The details of TOMMS GUI with step-by-step instructions to use it can be found in TOMMS user document [40].

3.8 Closing Remarks

This paper has presented a set of developments required to advance TOMMS to meet the mesh generation needs of the XGC gyrokinetic particle-in-cell code. The developments carried out include:

1. Extending the tokamak model construction process to support general combinations of O-points and X-points. This includes the development of methods for effective critical point search, modeling of the tokamak wall curve, and the construction of flux

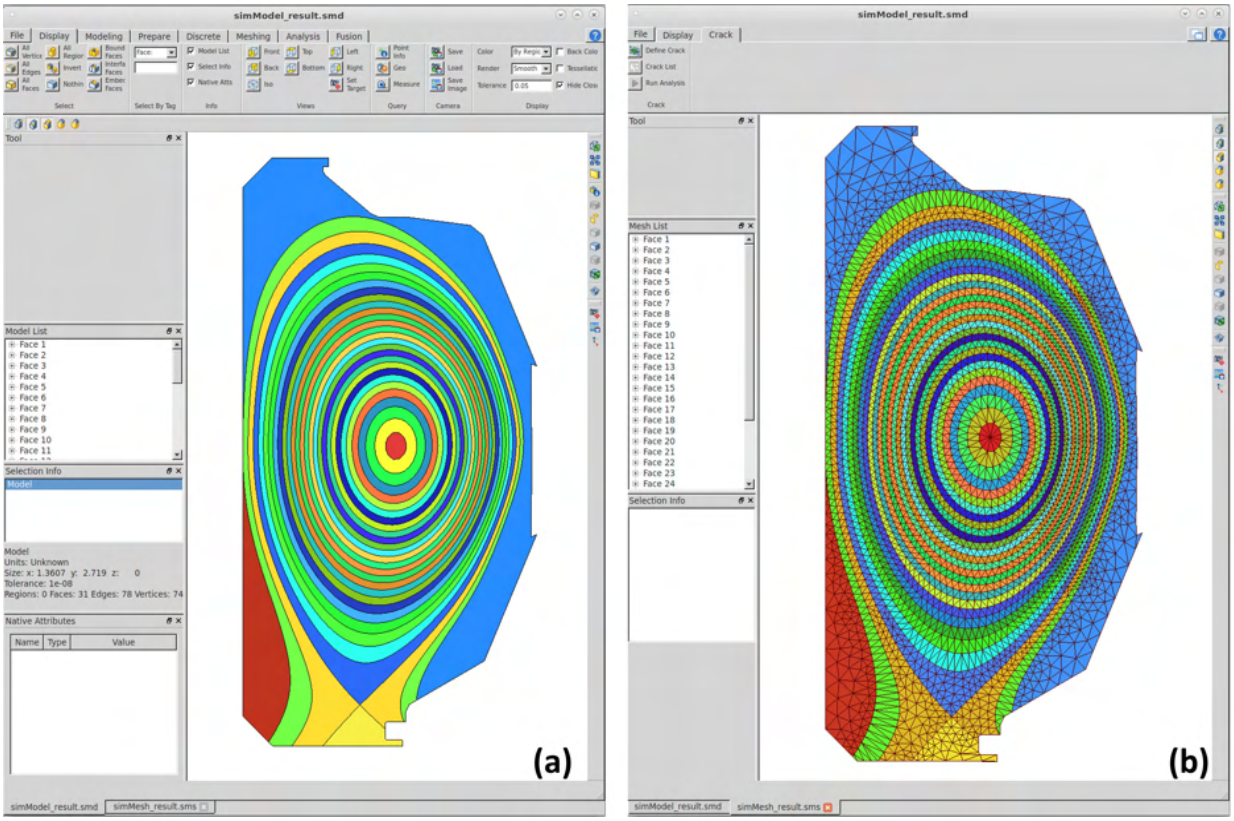

Figure 3.26: A view of (a) DIII-D model, (b) corresponding mesh on TOMMS GUI.

curves for a number of tokamak configurations. The critical point search method uses a combination of the Downhill Simplex method and Newton-Raphson method of root finding to ensure the full set of critical points. The wall modeling procedures employ a set of geometric properties to identify the key features of model vertices and model edges that are either straight lines or curves. The flux curve construction methods have been updated to support a general configuration of tokamak geometric features.

2. The mesh generation procedures were extended to support the creation of better-conditioned XGC meshes. These extensions first modified the model topology to split the faces between flux curves that included X-points or portions of the tokamak wall on their boundary into multiple faces where the additional faces were created near the X-point or where the flux curves approached the tokamak wall. As the X-point or wall face is approached, one of the newly created faces is meshed using a doubling transition procedure to go from high aspect ratio elements to low aspect ratio elements. This allowed the effective application of a general triangulation procedure to create a quality

mesh in the remaining face that included the X-point or portion of the wall curve on its boundary.

3. Providing the mesh data in a form, and with an ordering, as needed for the efficient XGC PIC-based simulations.

The resulting modeling and meshing tool, TOMMS, is being applied by XGC users to produce the meshes required for critical fusion plasma physics studies.

It should be pointed out that XGC is under continuous development to extend its ability to effectively address additional physics behavior. Some of these development will require new capabilities be added to the TOMMS mesh generation tool to provide the required meshes. A development that is in the planning stage is the selective application of mesh vertex motion and local mesh modification to account for large local gradients and evolving field. These procedures will be similar to those presented in reference [52].

CHAPTER 4
AN AUTOMATED MODELING AND MESHING FRAMEWORK FOR M3D-C^1

4.1 Background

M3D-C^1 [53] is a fusion plasma code developed at Princeton Physics Plasma Lab (PPPL) that solves the extended magneto-hydrodynamics (MHD) equations to study the large-scale plasma instabilities in tokamaks. These instabilities can degrade the magnetic confinement of the plasma, leading to loss of energy and/or damage to tokamak components [54]. The M3D-C^1 code is used to study a wide range of instabilities and their mitigation techniques [54], [55], [56], [57], [58]. Some of the instabilities and mitigation techniques that can be studied in M3D-C^1 are:

- Edge localized mode (ELM) is an MHD instability that occurs at the edge of the plasma. The ELMs cause the discharge of particles and energy from the plasma at the edge [59].

- Vertical displacement events (VDEs) are the MHD disruptions that occurs when plasma loses vertical stability. In such events, plasma moves upward or downward vertically and interacts with the walls of the tokamak leading to the abrupt release of the stored energies [35], [56].

- Tokamak sawtooth cycle is a type of MHD instability in which the plasma current increases to a limit where it becomes unstable to an internal kink mode. In such disruptions, the plasma current density and temperature drop in the center of the core [59], [60]. The process repeats again after the current density and temperature drop and is cyclic in nature.

- Pellet injection to the plasma is a disruption mitigation technique in tokamaks. In this method, impurities are added to the plasma as soon as disruptions are detected [54]. The study of the stabilization of plasma using the pellet injection is supported in M3D-C^1.

To solve the fourth-order MHD partial differential equation (PDE's), M3D-C^1 uses reduced quintic elements with an inter-element continuity of C^1 [53] on the poloidal cross-

section. The 3D elements between two poloidal planes are constructed using C^1 cubic Hermite polynomials in the toroidal direction [60]. The meshing data structure for the M3D-C^1 meshing needs is provided by Parallel Unstructured Mesh Infrastructure (PUMI) [61], [62] developed and maintained at Scientific Computation Research Center (SCOREC) at Rensselaer Polytechnic Institute (RPI). The initial unstructured mesh generation is done in Simmetrix [63]. PUMI provides the parallel mesh infrastructure and an adaptive mesh framework to the M3D-C^1 which performs three types of simulations [59] as the following.

- The 2D linear simulation considers a single poloidal plane and assumes that the model equations are linearized in such a way that a set of field equations independent of the toroidal direction is obtained.
- The 2D nonlinear simulation considers a single poloidal plane and assumes that the solution is symmetric in the toroidal direction.
- The 3D nonlinear simulation considers a full 3D domain discretized with wedge elements. The wedge element is constructed by using cubic Hermite polynomials between two 2D reduced quintic triangular elements.

This chapter provides an overview of M3D-C^1 and discusses recent developments in model and mesh generation in M3D-C^1. Section 4.2 presents an overview of the set of equations used in the extended MHD model. An overview of finite element discretization are presented in section 4.3. Sections 4.4 and 4.5 review the details of fully automated simulation workflows for M3D-C^1 in 2D and 3D respectively. The recent developments in terms of model and mesh generation for M3D-C^1 are summarized in section 4.6. The details of a-priori mesh modifications in M3D-C^1 are presented in section 4.7. Section 4.8 discusses mesh modifications on the model faces of interest and mesh size transitions in neighboring faces.

4.2 MHD Governing Equations

M3D-C^1 solves fourth-order PDE's derived from the extended MHD equations that describe plasma as an electricity-conducting fluid of ions and electrons. This section provides an overview of the PDE's solved in M3D-C^1 and their weak form formulation. For simplicity, the representation given here does not include two-fluid terms (separate representation of

ions and electrons), and particle source. The strong form of fourth-order PDE for 2D stream function $U = U(R, Z)$ and poloidal magnetic flux $\psi = \psi(R, Z)$ solved for M3D-C^1 is stated as

$$-\rho\nabla^2\frac{\partial U}{\partial t} + \rho < \nabla^2 U, U > - < \nabla^2\psi, \psi > +\mu\nabla^4 U = 0 \tag{4.1a}$$

$$-\nabla^2\frac{\partial \psi}{\partial t} + \nabla^2 < \psi, U > \eta\nabla^4\psi = 0 \tag{4.1b}$$

where ρ is the ion mass density, μ is the dynamic viscosity, η is the electrical resistivity, and the Poisson bracket [64] $< F, G >$ is defined as

$$< F, G > \quad \equiv \quad -(\nabla F \times \nabla G) \cdot \hat{\phi} \quad \equiv \quad \frac{\partial F}{\partial R}\frac{\partial G}{\partial Z} - \frac{\partial F}{\partial Z}\frac{\partial G}{\partial R} \tag{4.2}$$

The formulation of the weak form from the strong form of fourth-order PDE's presented in equation 4.1 assumes homogeneous Dirichlet boundary conditions for U and ψ such that $U = \nabla U \cdot \hat{n} = 0$ and $\psi = \nabla\psi \cdot \hat{n} = 0$ at the boundary. The weighting function space is defined as:

$$\mathcal{V} = \{\nu | \nu \in H^2, \nu = \nabla\nu \cdot \hat{n} = 0 \quad at \quad d\Omega\} \tag{4.3}$$

The solution space is defined as:

$$\mathcal{W} = \{\omega | \omega \in H^2, \omega = \nabla\omega \cdot \hat{n} = 0 \quad at \quad d\Omega\} \tag{4.4}$$

The weak form of the fourth-order PDE in equation 4.1 is defined as:
Find $(U, \psi) \in \mathcal{W} \times \mathcal{W}, \quad \ni \forall \quad (\nu, q) \in \mathcal{V} \times \mathcal{V}$.

$$\int_\Omega \left\{ \rho\nabla\frac{\partial U}{\partial t} \cdot \nabla\nu + \rho\nabla^2 U < U, \nu > -\nabla^2\psi < \psi, \nu > +\mu\nabla^2 U\nabla^2\nu \right\} d\Omega = 0 \tag{4.5a}$$

$$\int_\Omega \left\{ \nabla\frac{\partial \psi}{\partial t} \cdot \nabla q - \nabla < \psi, U > \cdot\nabla q + \eta\nabla^2\psi\nabla^2 q \right\} d\Omega = 0 \tag{4.5b}$$

where the initial condition is defined as $(U, \psi) = (U^0, \psi^0) \quad at \quad t = 0$. See reference [64] for the integration identities used in the formulation of weak form from the strong form

of the PDE.

4.3 Finite Elements in M3D-C1

The fourth-order MHD PDE's are discretized using the C^1 finite elements. M3D-C^1 uses a C^1 reduced quintic triangle elements on the 2D cross-section of the poloidal plane [53]. The 3D wedge elements are constructed between two consecutive poloidal planes using C^1 cubic Hermite polynomials in the toroidal direction [60]. This section overviews the properties of 2D reduced quintic triangle elements and 3D wedge elements with C^1 continuity.

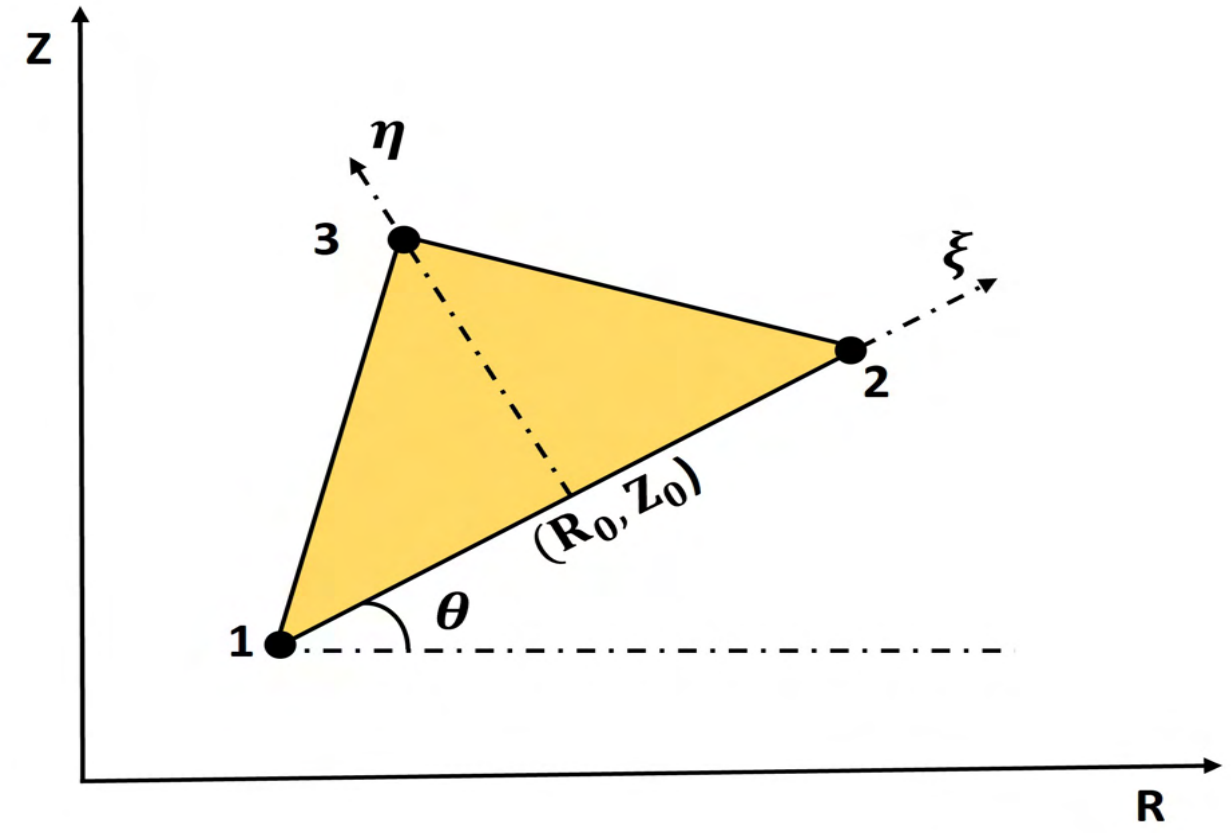

Figure 4.1: 2D reduced quintic triangle element.

4.3.1 Reduced Quintic Triangle Element

Consider the reduced quintic triangle element on a poloidal R-Z plane as shown in the figure 4.1. The origin of the local (ξ, η) coordinates is (R_0, Z_0) as shown in the figure. The mapping between the local and global coordinates is defined as

$$\begin{bmatrix} R \\ Z \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} \xi \\ \eta \end{bmatrix} + \begin{bmatrix} R_0 \\ Z_0 \end{bmatrix} \tag{4.6}$$

There are six degrees of freedom (DOF) defined on each of the three vertices of the traingle. These DOF's include the field values, first derivatives, and second derivatives. The six DOF's (U, $U_{,\xi}$, $U_{,\eta}$, $U_{,\xi\xi}$,$U_{,\xi\eta}$, $U_{,\eta\eta}$) evaluated at each of the vertex are in local (ξ, η) coordinate system. The desired DOF's $(U, U_R, U_Z, U_{RR}, U_{RZ}, U_{ZZ})$ are in global coordinate system. Hence, the local DOF's are required to map to global coordinates according to the expression given in equation 4.6. The details of the construction of the reduced quintic basis functions such that the element satisfies C^1 inter-element continuity is covered in references [53], [65].

4.3.2 3D Wedge Element

Consider the 3D wedge element in (R, Z, ϕ) coordinates as shown in figure 4.2. A cubic Hermite polynomial in the ϕ direction along with the reduced quintic in the R-Z plane are used to construct a 3D wedge element. The mapping between local toroidal coordinate $(\tilde{\phi})$ and global toroidal coordinate (ϕ) between two poloidal planes at the toroidal angles of ϕ_1 and ϕ_{i+1} is defined as:

$$\phi = \phi_i + (\phi_{i+1} - \phi_i)\tilde{\phi} \tag{4.7}$$

The overall mapping between the local $(\xi, \eta, \tilde{\phi})$ and global coordinates $(R, Z, \phi,)$is defined as

$$\begin{bmatrix} R \\ Z \\ \phi \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & \phi_{i+1} - \phi_i \end{bmatrix} \begin{bmatrix} \xi \\ \eta \\ \tilde{\phi} \end{bmatrix} + \begin{bmatrix} R_0 \\ Z_0 \\ \phi_i \end{bmatrix} \tag{4.8}$$

There are twelve DOF's defined on each of the six vertices of the wedge elements. The field values corresponding to twelve DOF's in local and global coordinates evaluated at each of the six vertices are given in table 4.1. The mapping between the local DOF's and global DOF's is done using the expression given in equation 4.8.

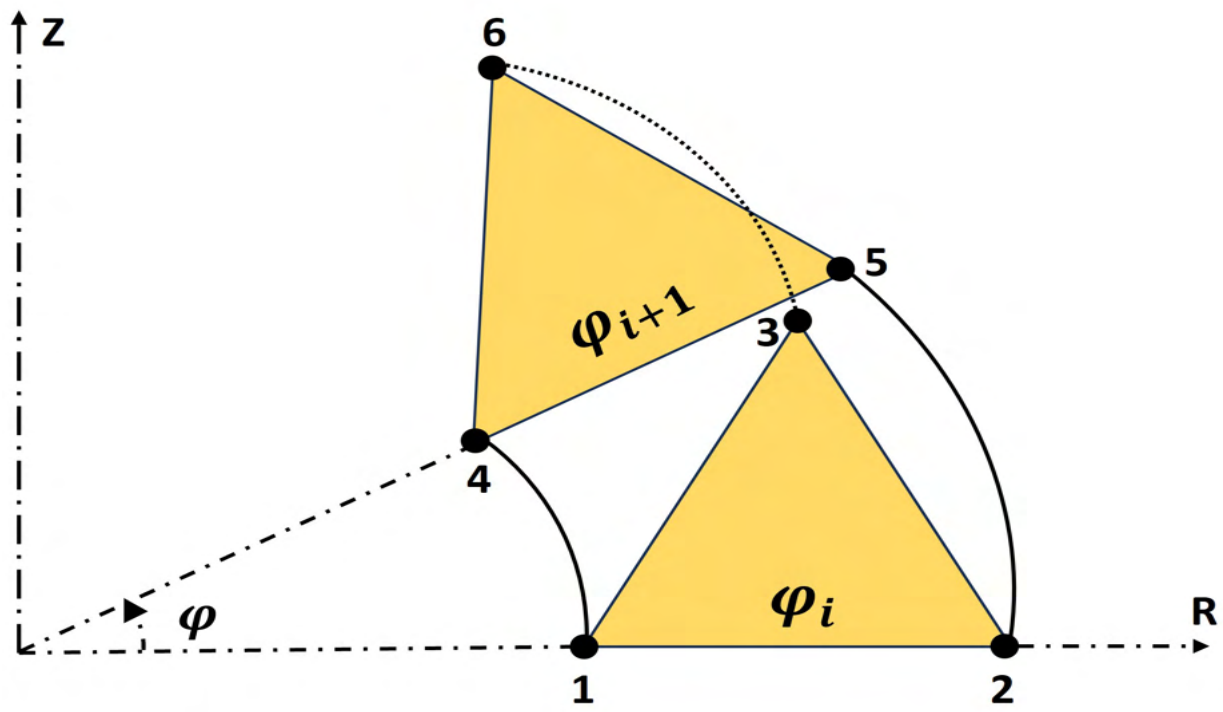

Figure 4.2: 3D wedge elements in a right-handed cylindrical coordinate system.

Table 4.1: The field values corresponding to twelve DOF's in local and global coordinates.

Local DOF	U	$U_{,\xi}$	$U_{,\eta}$	$U_{,\xi\xi}$	$U_{,\xi\eta}$	$U_{,\eta\eta}$
Global DOF	U	$U_{,R}$	$U_{,Z}$	$U_{,RR}$	$U_{,RZ}$	$U_{,ZZ}$
Local DOF	$U_{,\tilde{\phi}}$	$U_{,\xi\tilde{\phi}}$	$U_{,\eta\tilde{\phi}}$	$U_{,\xi\xi\tilde{\phi}}$	$U_{,\xi\eta\tilde{\phi}}$	$U_{,\eta\eta\tilde{\phi}}$
Global DOF	$U_{,\phi}$	$U_{,R\phi}$	$U_{,Z\phi}$	$U_{,RR\phi}$	$U_{,RZ\phi}$	$U_{,ZZ\phi}$

4.4 2D Simulation Workflow

The M3D-C^1 uses a set of specific software tools for a complete simulation loop as shown in figure 4.3. This section provides an overview of all the software tools and necessary steps required to support the meshing needs in M3D-C^1. The three software components in the workflow are:

- Simmetrix: The initial 2D model and mesh generation is done in Simmetrix [63]. The definition of the model in Simmetrix requires a set of model loop geometries and a set of model faces with the information of bounding loops. The mesh generation requires the mesh size information on the model entities and global mesh control parameters. A detailed discussion of model and mesh generation is provided in section 4.6.

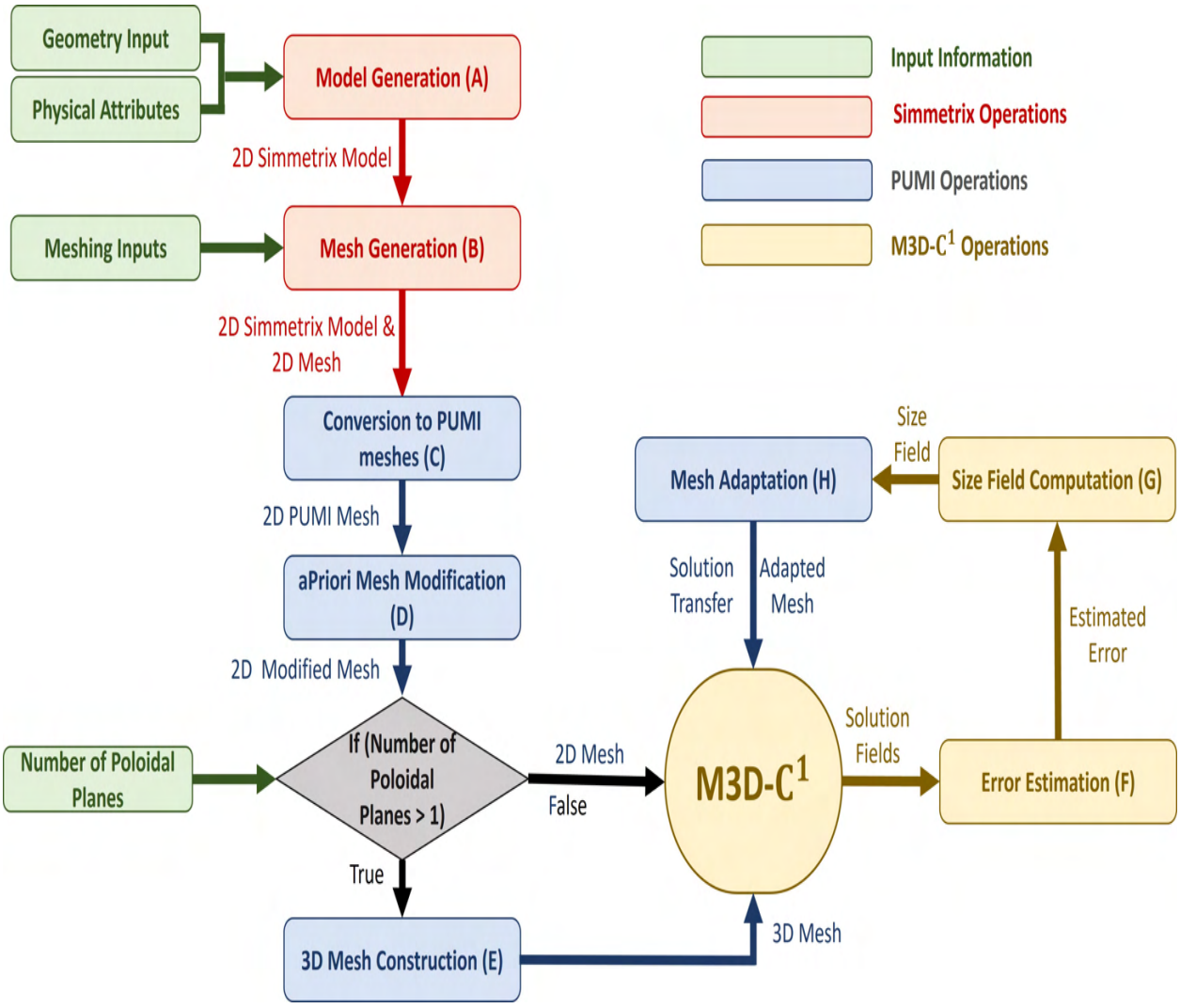

Figure 4.3: M3D-C^1 workflow.

- PUMI: The geometric modeling interface and mesh data structure for the M3D-C^1 are provided by PUMI. The geometric model information provided to a M3D-C^1 simulation is the number of loops with the geometry of loops defined by a set of points on the 2D plane. The loops in the model are represented by the splines fitted to a set of input points. The Geometric Model Interface (GMI) in PUMI supports the model queries for the M3D-C^1 simulations such as the calculation of normal vectors and curvatures on the boundaries etc.. The initial mesh generated in Simmetrix is loaded to PUMI to support the full range of meshing capabilities (mesh queries, mesh partitioning, mesh adaptation, etc.).

- M3D-C^1: The set of MHD equations in the weak form, the boundary conditions over

the domain, and the initial conditions are provided for problem definition. Once the mathematical model is defined and the mesh is loaded, the next step is the discretization of the governing PDE's with C^1 elements. The contributions to the stiffness matrix and force vector from the elements are calculated. The plasma equilibrium file provides the input field information for elemental level contribution in the initial time step. After the first time step, the field information at each time step is collected from the previous time step. The elemental level contributions from all the elements in the mesh are assembled in the global stiffness matrix and force vector. The global system of equations is solved using the PETSc solver [66] to get the desired solution fields at each time step.

The steps in the workflow are:

- Model Generation (A in figure 4.3): The geometry input in the form of loops and faces is provided to the Simmetrix for the generation of 2D poloidal geometric model. Furthermore, the physical attributes for the model entities are provided to the model generation process for the definition of physics regions in the model (wall, vacuum etc.).

- Mesh Generation (B in figure 4.3): The 2D geometric model is meshed in Simmetrix using the mesh size information on the model entities and other global mesh parameters.

- Conversion to PUMI meshes (C in figure 4.3): The mesh data structure for the M3D-C^1 is supported through PUMI. All the mesh queries, mesh distribution, and mesh adaptation in M3D-C^1 are done in PUMI. Therefore, the initial model and mesh generated in Simmetrix need to be converted to PUMI readable model and mesh.

- a-priori Mesh Modification (D in figure 4.3): In certain cases, the initial mesh needs to be modified based on the needs of a particular problem. This is done by setting the a-priori size field information for the mesh modification. This information can come from the input plasma equilibrium file, the location of impurities (pellets) in the domain, or the position of the tokamak wall in the domain. The desired size field information is communicated to PUMI for the mesh modification operations. A detailed discussion of a-priori mesh modification is provided in section 4.7.

- Error Estimation (F in figure 4.3): After the computation of solution fields, a decision to improve the meshes is made based on an estimation of the solution error field. The error from the a-posteriori error estimation method is used to compute the error on the mesh vertices.

- Size-field Computation (G in figure 4.3): Once the error is evaluated on the nodes, the scaler elemental size field is evaluated for every mesh element in the domain. The details of size field evaluation are presented in section 5.2.

- Mesh Adaptation (H in figure 4.3): The size field information is used by the mesh modification procedures to adapt the mesh at the desired time steps. A comprehensive discussion of error estimation with adaptive results is presented in chapter 5. The solution fields are mapped onto the new mesh after the mesh is adapted.

4.5 3D Simulation Workflow

The workflow for a 3D automated simulation loop is presented in figure 4.3. The basic set of software components required for a 3D simulation are those of the 2D simulation (see section 4.4 for the description of these components) with the addition of 3D element construction. Aside from the mathematical model and an initial 2D model and mesh, an additional input defining the desired number of poloidal planes is required. The 2D poloidal plane mesh, either from Simmetrix or modified using a-priori mesh modification, is extruded for the desired number of planes to get a 3D mesh. The resulting mesh contains C^1 wedge elements, such that a wedge is defined between the matching 2D elements on two consecutive poloidal planes. If the number of elements on the 2D poloidal plane mesh is n, and the total number of planes is P, the number of 3D wedge elements in the mesh is $n \times P$.

In terms of a-posteriori mesh adaptation in a 3D simulation loop, the desired size field from all the poloidal planes is collected at the master plane, and the smallest size for each node is selected for adapting the 2D master poloidal plane. After the adaptation, the 3D mesh is reconstructed and the solution fields are mapped onto the new 3D mesh. The details of mesh adaptation and solution transfer for the 3D simulations are presented in section 5.4.

4.6 Model and Mesh Generation

The model and mesh generation of the tokamak geometries for M3D-C^1 is supported through a combination of Simmetrix and PUMI tools. The initial model and mesh generation are done using the Simmetrix tools, and mesh parallelization and mesh information management in M3D-C^1 are supported by PUMI. Two options are available for defining the M3D-C^1 geometry and associated mesh control information.

- Option 1: Five predefined model configurations with a maximum of three model faces and a minimal set of mesh control parameters are available.
  - Configuration 1: A single-face model with the vacuum boundary parameterized using an analytic expression.
  - Configuration 2: A single-face model with the vacuum boundary defined by piecewise linear points.
  - Configuration 3: A single-face model with the vacuum boundary defined by piecewise polynomials.
  - Configuration 4: A three-face model with three curves (inner wall, outer wall, and vacuum) defined from the spline fitting of a set of points.
  - Configuration 5: A three-face model with three curves (inner wall, outer wall, and vacuum). The set of points are provided for the inner wall and an offset distance from the inner wall is provided for the outer wall. The points on the outer wall are evaluated using the offset distance and inner wall points. The resulting three curves are obtained by the spline fitting of the set of the points.
- Option 2: A more generalized model and mesh generation procedure supports an arbitrary number of model faces. The model faces are defined by providing the information on the bounding loops for each face. The mesh control parameters [67] are defined on each loop and face for an effective starting mesh.

The original geometry and mesh control tools were limited to a standard three-region configuration. As part of this work, the generalized model generation procedure was developed that can handle any number of model faces representing different physics regions.

4.6.1 Generalized Model Generation for Multi-face Geometries

The following information is required to generate a model with an arbitrary number of model loops and faces in M3D-C^1.

- A set of model loop geometries.
- A set of model faces defined in terms of the set of bounding loops.

The 2D model is defined in terms of a set of model faces.

4.6.1.1 Model Vertices

A model vertex is defined at the starting point of each model loop. If the loop is defined by a set of discrete line segments, a model vertex is placed at the starting point of the first segment. If the loop is parameterized, a model vertex is placed at the point defined by the parameter $t = 0$. Since all the loops are periodic, only a single model vertex will bound each loop. Figure 4.4 demonstrates the model vertices defined on each model loop, where each loop represents a physical component in the SPARC tokamak [68].

- $L1$: First wall that bounds the region containing the plasma.
- $L2$, $L3$: Outer boundaries of lower and upper passive plates respectively. The passive plates are conducting plates (usually copper) that are placed near the plasma to stabilize it.
- $L4$, $L5$: Inner and outer boundaries of the first vessel wall.
- $L6$, $L7$: Inner and outer boundaries of the second vessel wall.
- $L8$: Outer boundary of vacuum region.

4.6.1.2 Model Loops

A model loop can be defined in three different ways.

- A set of discrete line segments.

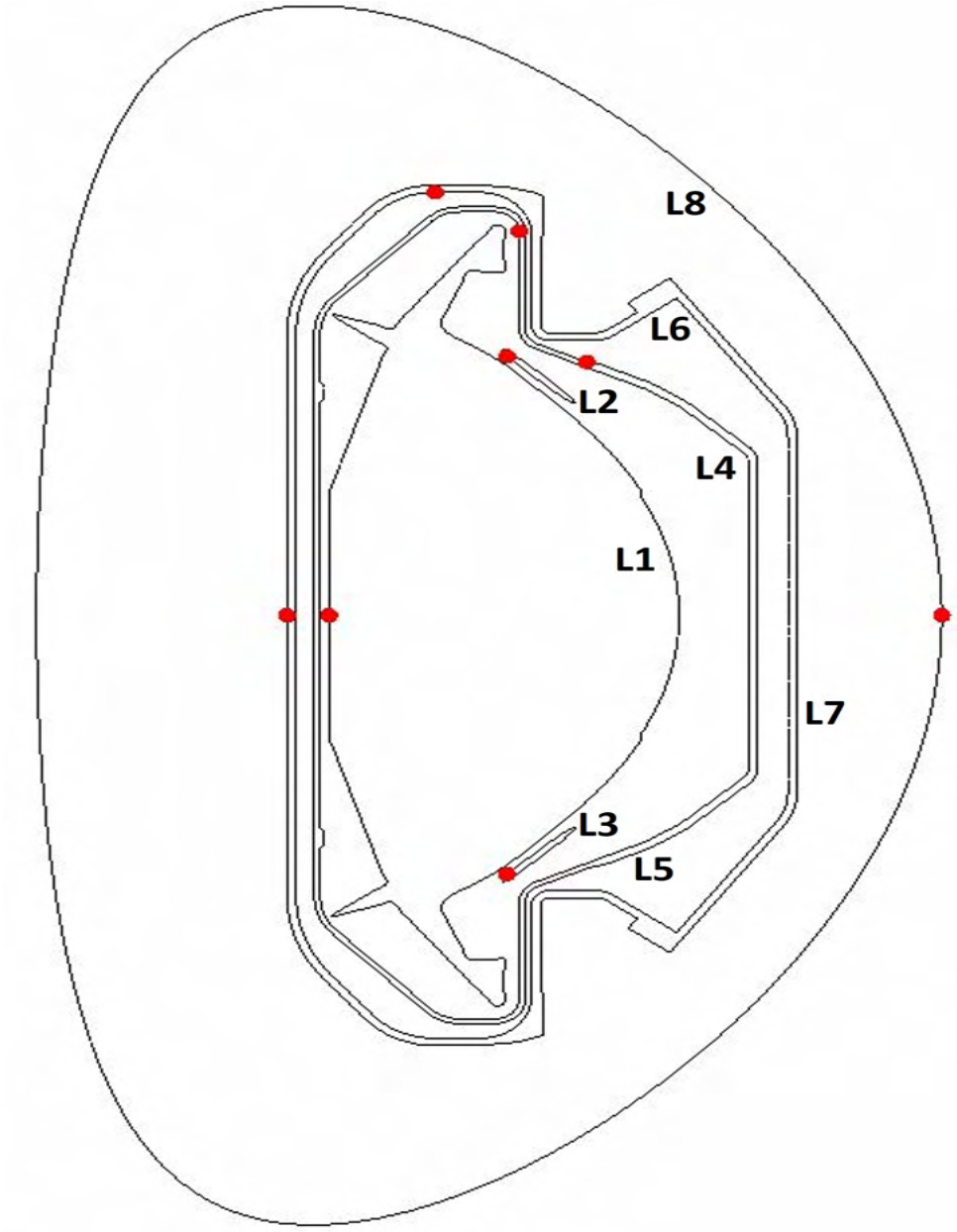

Figure 4.4: Demonstration of model vertices defined on the set of model loops.

- A parameterized vacuum loop that is defined from a set of five input parameters. For given five parameters (R_0, R_1, R_2, Z_0, Z_1), the vacuum boundary with respect to parameter t is defined as:

$$R(t) = R_0 + R_1 \cos\left(t + R_2 \sin\left(t\right)\right) \tag{4.9a}$$

$$Z(t) = Z_0 + Z_1 \sin\left(t\right) \tag{4.9b}$$

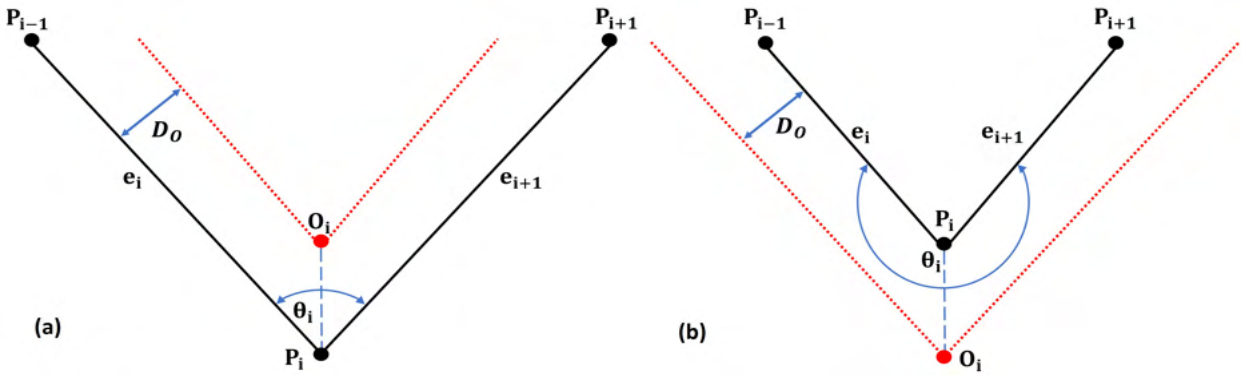

Figure 4.5: (a) Internal angle between the two incident line segments is less than 180°, (b) internal angle between the two incident line segments is greater than 180°.

- A loop created by the offset of a loop already defined in the geometry. The inputs required for the offset loop are the original input loop and the offset distance (D_O). This option is used to create a finite-thickness wall in the model, however, it can be used to create any offset loop for any given loop. Given the offset distance (D_O), an offset point O_i on the offset curve from the point P_i (on the intersection of line segments (P_{i-1}, P_i) and (P_i, P_{i+1})) on the input curve can be evaluated using the equations [69],[70] provided below.

$$O_i = P_i + \frac{e_{i+1} - e_i}{|e_{i+1} - e_i|} \cdot \frac{D_O}{sin(\frac{\theta_i}{2})} \quad \text{if} \quad (0 < \theta < 180) \tag{4.10a}$$

$$O_i = P_i + \frac{e_i - e_{i+1}}{|e_i - e_{i+1}|} \cdot \frac{D_O}{cos(\frac{\pi - \theta_i}{2})} \quad \text{if} \quad (\theta > 180) \tag{4.10b}$$

where $e_i = \frac{P_i - P_{i-1}}{|P_i - P_{i-1}|}$, $e_{i+1} = \frac{P_{i+1} - P_i}{|P_{i+1} - P_i|}$, and θ_i is the angle between the two line segments. If the point is on a straight line ($\theta = 0, 180$), the point is ignored and no offset point is evaluated. Once the raw offset points are found, the next step is to find the invalid offset line segments. The two types of invalidity that can arise are:

  - Invalid direction where the direction of an offset line segment is opposite to the direction of the parent line segment in the original curve.
  - Invalid position where the offset line segment lies completely in the offset area, i.e., the shortest distances from both ends of offset line segments to parent curve

are less than the offset distance.

The corrective measures taken to eliminate any invalidity depend on the type of invalidity and the number of invalid line segments between each pair of valid line segments. The details of which can be found in reference [69]. Figures 4.6, 4.7, and 4.8 demonstrate the results of the implementation of the curve offset algorithm for offset distances of 0.04m, 0.1m, and 0.2m respectively. For the sharp corners as demonstrated in zoomed-in regions of three figures, the invalid segments increase with increasing the offset distance and the resulting raw offset curve can have multiple self-intersecting segments between a pair of valid segments as shown in figure 4.8.

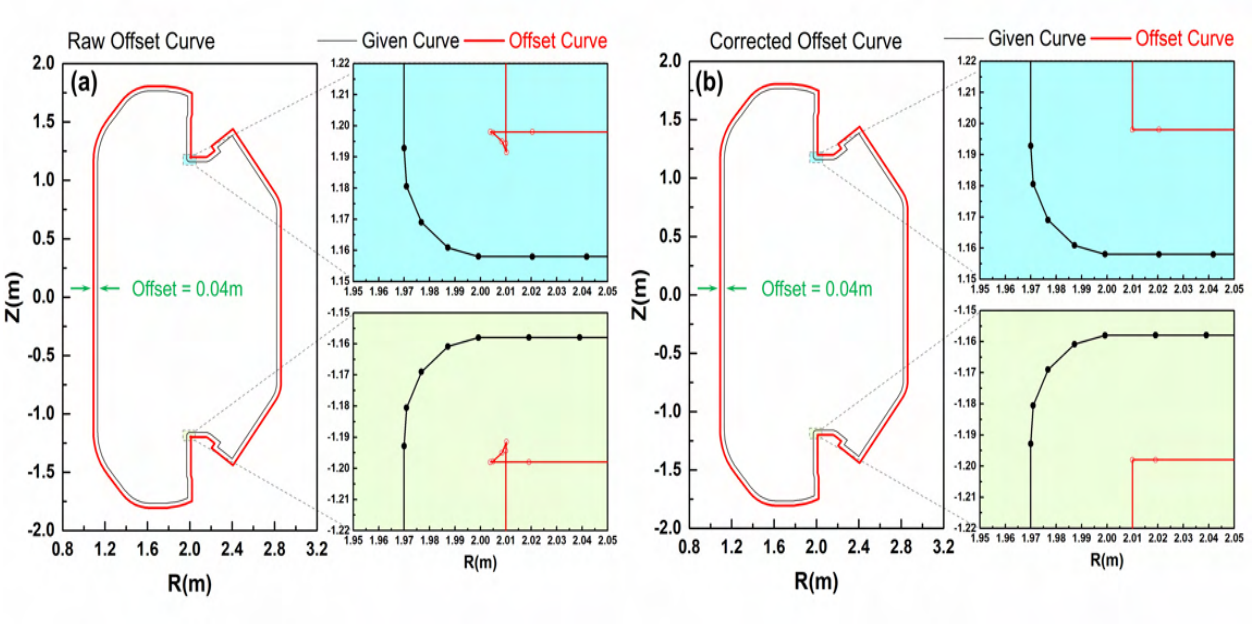

Figure 4.6: The parent and offset curves with an offset of 0.04m (a) raw offset curve with invalid offset segments, (b) corrected offset curve with no invalidity.

The curve points evaluated from any of the above methods are processed by fitting the cubic B-splines. Since the degree of the splines is 3, the resultant curve ensures C^2 continuity between model vertices.

4.6.1.3 Model Faces

A model face is defined by providing the information of bounding model loops. The convention for the generation of the model face in Simmetrix is to define the outermost

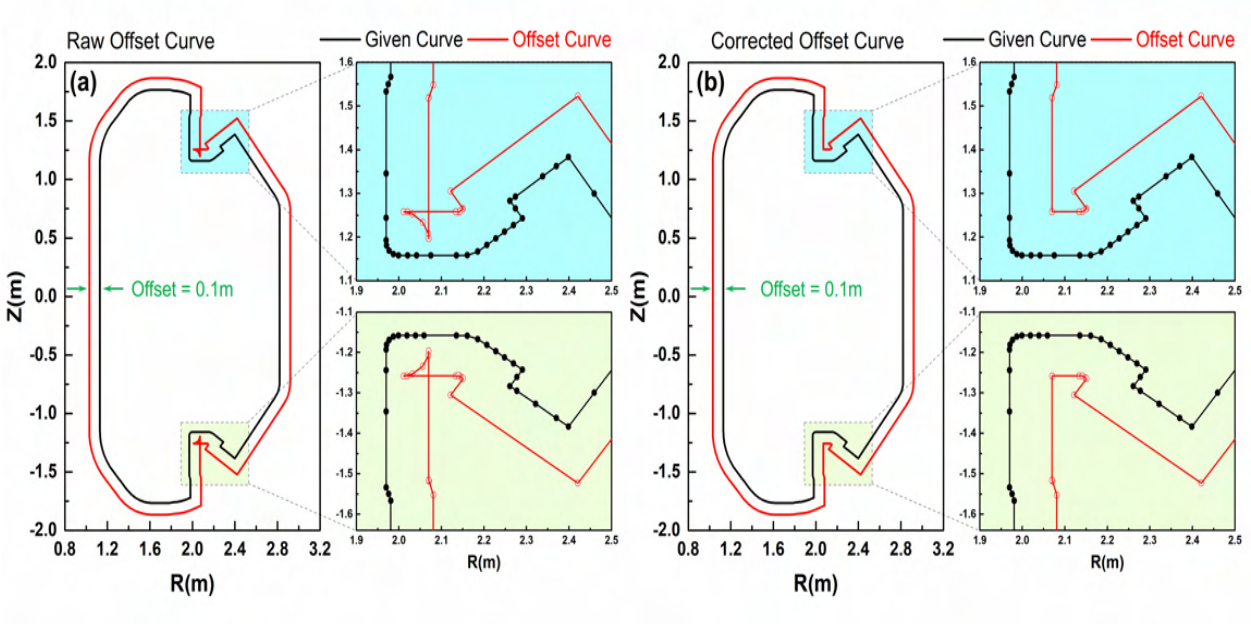

Figure 4.7: The parent and offset curves with an offset of 0.1m (a) raw offset curve with invalid offset segments, (b) corrected offset curve with no invalidity.

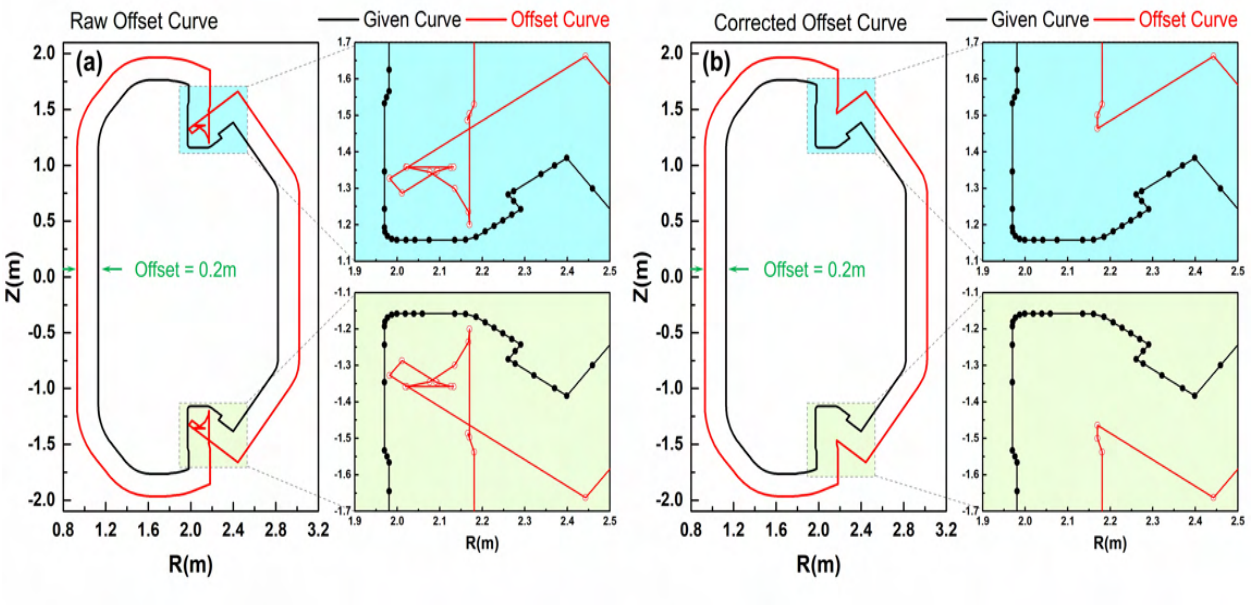

Figure 4.8: The parent and offset curves with an offset of 0.2m (a) raw offset curve with invalid offset segments, (b) corrected offset curve with no invalidity.

loop in the counter-clockwise direction and any inner loops in the clockwise direction. The input loops may be given in either counter-clockwise or clockwise directions, and hence it is necessary to find the direction of each input loop before defining a model face. The direction

of an arbitrary curve can be found using algorithm 3. An example model generated from eight loops and eight resulting model faces is presented in figure 4.9. The list of model faces (labeled with F in figure 4.9) with the information of bounding loops (labeled with L in figure 4.9) is provided.

- $F1$: $L1$
- $F2$: $L2$
- $F3$: $L3$
- $F4$: $L1$, $L2$, $L3$, and $L4$
- $F5$: $L4$, and $L5$
- $F6$: $L5$, and $L6$
- $F7$: $L6$, and $L7$
- $F8$: $L7$, and $L8$

Algorithm 3 Curve Direction

```
1: procedure CURVEORIENTATION(P(1:N))          ▷ N number of given points
2:     total area ← 0
3:     for i = 2 to N do
4:         area under line segment ← (x_i − x_{i−1}) × (y_i + y_{i+1})
5:         total area ← total area + area under line segment
6:     end for
7:     if total area > 0 then
8:         curve is clockwise
9:     else
10:        curve is counter-clockwise
11:    end if
12: end procedure
```

4.6.2 Mesh Generation and Examples

The mesh size on each loop and each face is defined for effective mesh resolution. Moreover, the global mesh control parameters such as gradation rate can be defined for better control of mesh sizes. These mesh control parameters [67] can be directly provided

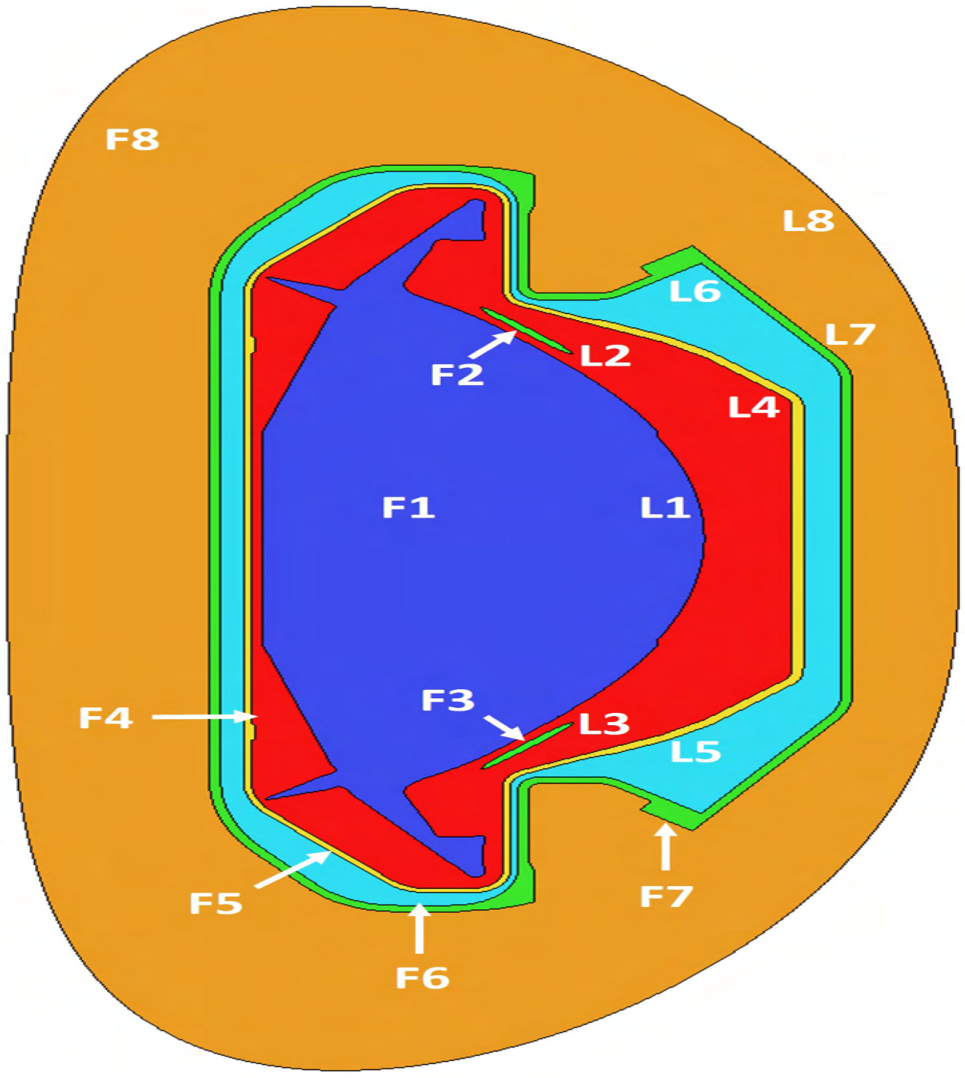

Figure 4.9: A model defined from a set of model loops and model faces including vacuum region.

with the input file or could be interactively modified by using Simmetrix Simmodeler GUI. Once the model is defined and mesh control parameters are set, Simmetrix generates the mesh.

Figure 4.10 demonstrates two meshes with three model faces each. Figure 4.10a shows a mesh with no vacuum region and figure 4.10b presents a mesh with a vacuum region.

A mesh with eight model faces is demonstrated in figure 4.11a, along with a zoomed-in view of the inner tokamak region (minus vacuum face) in figure 4.11b and a closeup of the top quarter of the mesh in figure 4.11c.

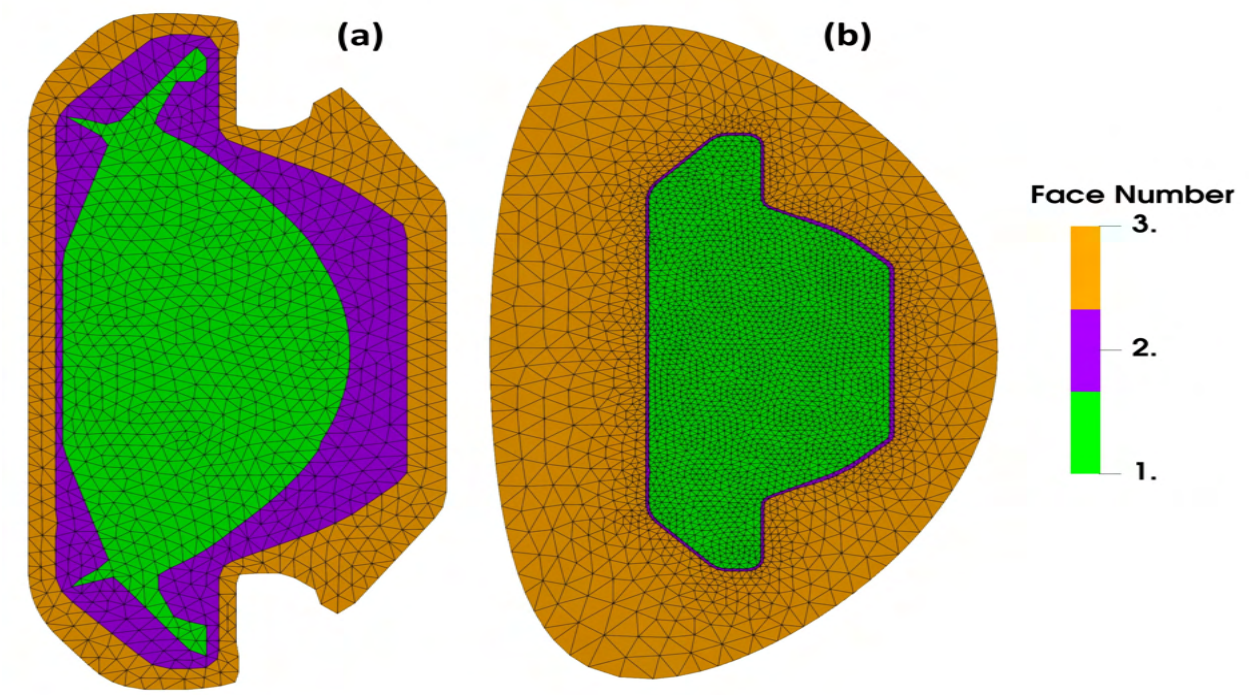

Figure 4.10: (a) A mesh with three loops and three model faces where all three loops are defined as a set of discrete points, (b) a mesh with three loops and three model faces where two loops are defined as a set of discrete points and vacuum loop is parameterized using an analytical expression.

Along with the generalized framework for mesh generation, control of mesh generation and mesh gradation is also needed to generate a mesh with desired resolution. One of the critical areas in M3D-C^1 is the finite-thickness wall area. A high resolution is desired through the wall thickness but defining an isotropic mesh size results in meshes with a very high number of undesired elements in, and near, the wall. An anisotropic layered mesh procedure was introduced to allow the desired refined mesh resolution through the wall thickness without compromising the coarser mesh resolution in the rest of the domain. Figure 4.12 demonstrates a comparison of two meshes, one without layered mesh on the wall face (figure 4.12a) and one with an anisotropic layer mesh on the wall face (figure 4.12b). The resulting layered mesh has 4.9 thousand elements as compared to 19 thousand elements on the original mesh with isotropic mesh size on the wall face.

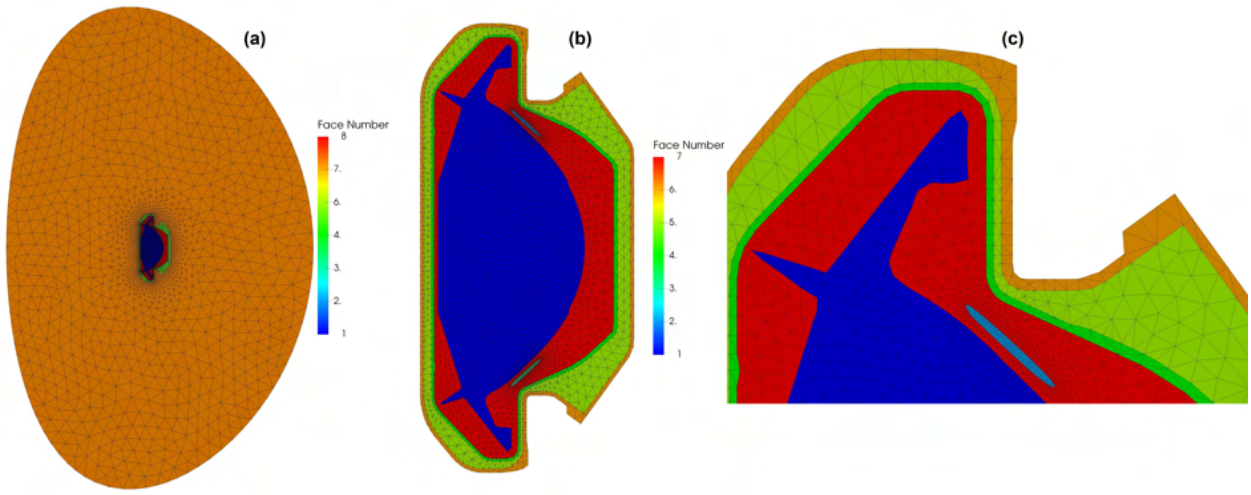

Figure 4.11: (a) A mesh on a model which is defined from a set of model faces including vacuum region, (b) a magnified view of the region bounded by inner vacuum loop, (c) zoomed-in view of the top quarter of the mesh in (b).

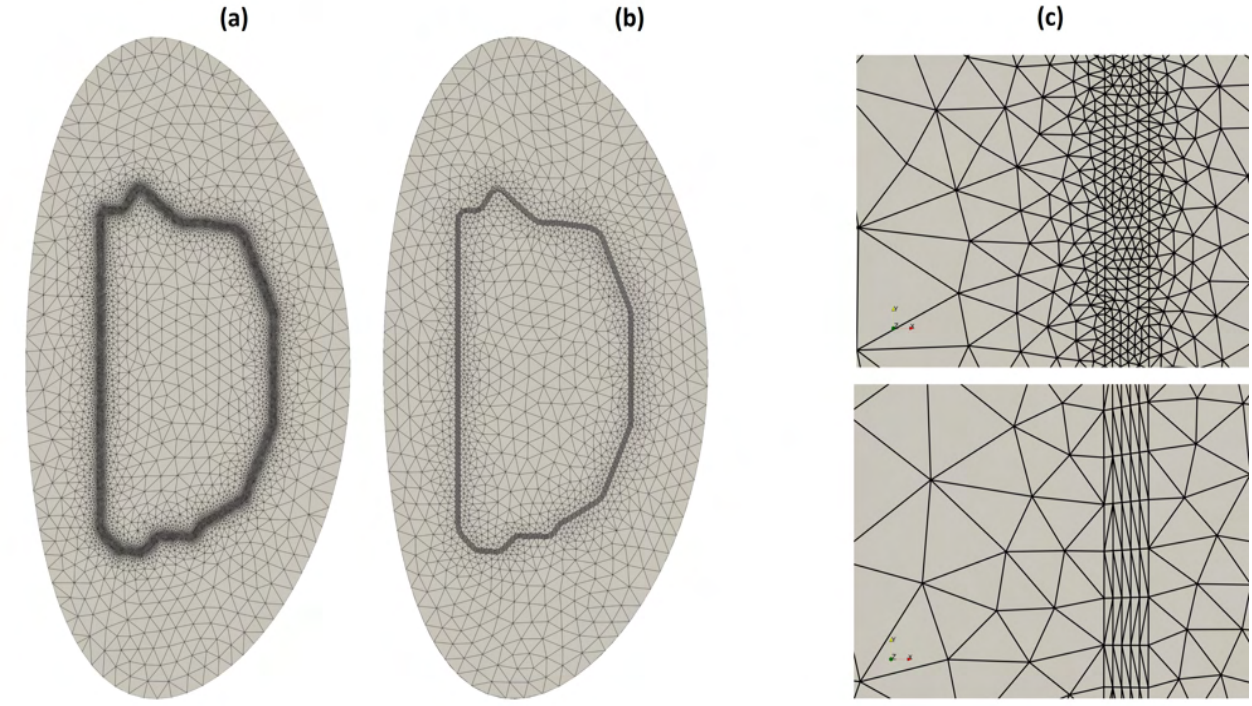

Figure 4.12: (a) A mesh with an isotropic mesh size on the wall face, (b) a mesh with an anisotropic mesh size on the wall face, (c) a close-up of the wall face for two meshes.

4.7 A-priori Mesh Modifications

The initial mesh generation procedure requires mesh size definition on the model entities. The other way to get desired mesh sizes is to modify a given mesh by defining a new size field on that mesh and executing mesh modification. A-priori procedures allow the users to modify the meshes using a mesh size field based on the knowledge of plasma equilibrium, pellet locations, wall location, or an analytic expression before the actual simulation loop. The most frequently used a-priori mesh modification procedure is the use of a magnetic flux field (ψ) from the plasma equilibrium to specify the mesh size field. A normalized field $\tilde{\psi}$ is defined as

$$\tilde{\psi} = \frac{\psi - \psi_0}{\psi_1 - \psi_0} \tag{4.11}$$

where ψ_0 and ψ_1 are the flux field values at the magnetic axis and plasma boundary. An expression based on the normalized flux value evaluates the mesh size field in different regions based on $\tilde{\psi}$. The normal mesh size h_1 and tangent mesh size h_2 on every node are defined as:

$$h_i^{-1} = \tilde{h}_i^{-1} + \frac{1}{l_{ci}\left(1 + \frac{\tilde{\psi}-\psi_c}{W_c}\right)^2}, \quad i = 1, 2 \tag{4.12}$$

where l_{ci}, W_c, ψ_c are constants and $\tilde{h}_i$ is defined based on what region the node lies [64]. If the node exists in the region bounded by the flux curve defined by a_1, then $\tilde{h}_i$ is defined as:

$$\tilde{h}_i = b_i[1 - e^{-|\frac{\tilde{\psi}}{a_1}-1|^{a2}}] + d_i, \quad \tilde{\psi} < a_1 \tag{4.13}$$

If the mesh node is in the region outside of the flux curve defined by a_1, then $\tilde{h}_i$ is defined as:

$$\tilde{h}_i = c_i[1 - e^{-|\frac{\tilde{\psi}}{a_1}-1|^{a3}}] + d_i, \quad \tilde{\psi} > a_1 \tag{4.14}$$

where the constant a_1 splits the mesh in two physics regions (mostly $a_1 = 1$ and signifies the separatrix curve). The constants b_i and a_2 are used to modify the mesh sizes in the region defined by $\tilde{\psi} < a_1$ and the constants c_i and a_3 do the same for the region defined

by $\tilde{\psi} > a_1$. The constants d_i controls the aspect ratio of the modified elements. Figure 4.13 demonstrates the a-priori modification of a mesh by defining a size field from the $\tilde{\psi}$ field. The demonstration shows the meshes for $a_1 = 1.0$ and $a_1 = 0.5$ by keeping all the other constants the same.

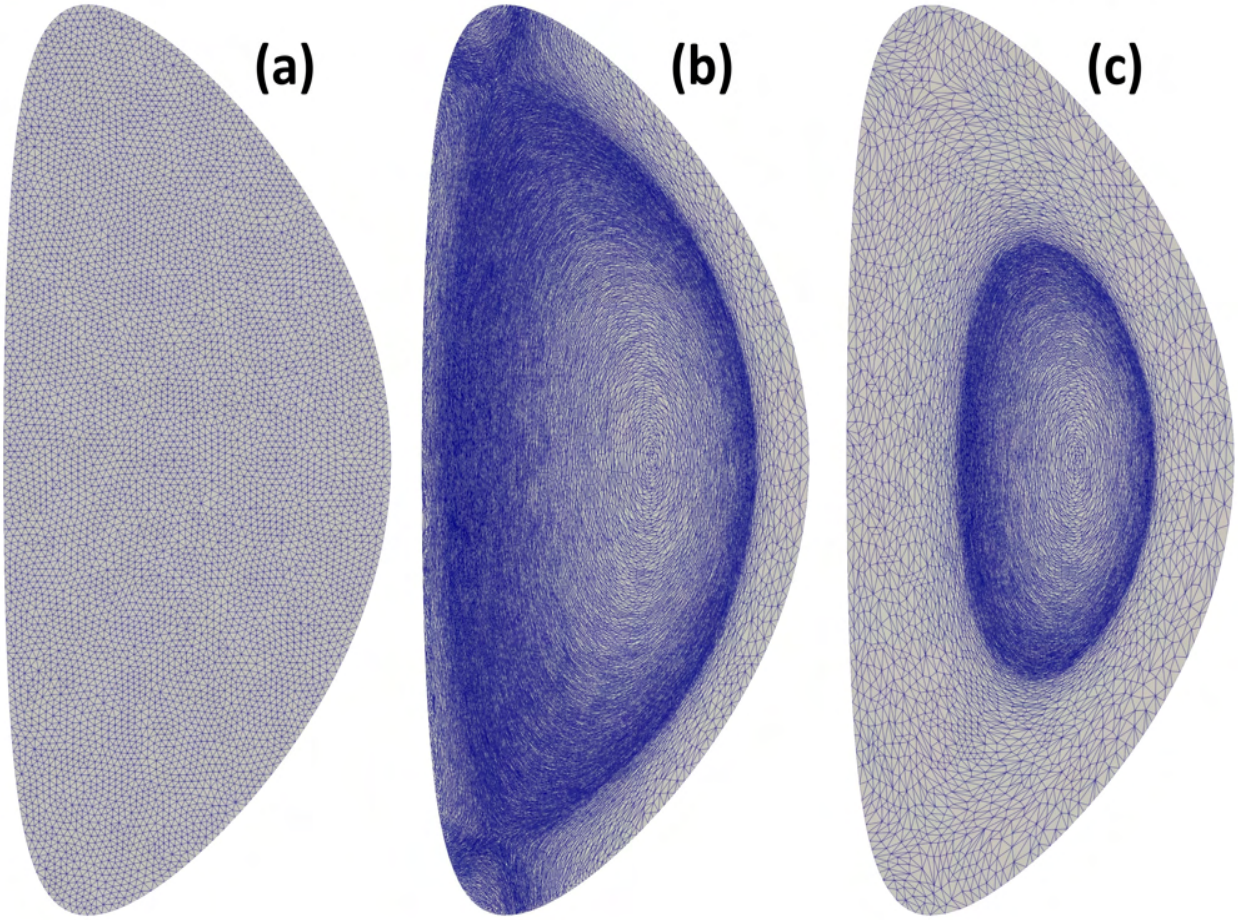

Figure 4.13: (a) Original mesh, (b) modified mesh with $a_1 = \tilde{\psi} = 1$, (c) modified mesh with $a_1 = \tilde{\psi} = 0.5$.

Certain areas in the plasma are critical to the simulation results and a refined mesh is needed for such critical areas. One example is the introduction of impurities (pellets) to plasma for disruption mitigation requires fine mesh sizes in local regions. To support the modification of the mesh based on a size field evaluated from a-priori knowledge of the problem, a general mesh modification framework is developed. This framework requires the definition of a size field (two edge lengths and one unit vector along one of the edges) on

every node of the mesh. This size field could be evaluated using an analytic expression based on a-priori knowledge of problems such as pellet locations, coil locations, etc., and provided to the mesh modification framework for mesh modifications. Figure 4.14 shows the modification of a mesh based on the size field defined from an arbitrary analytical expression.

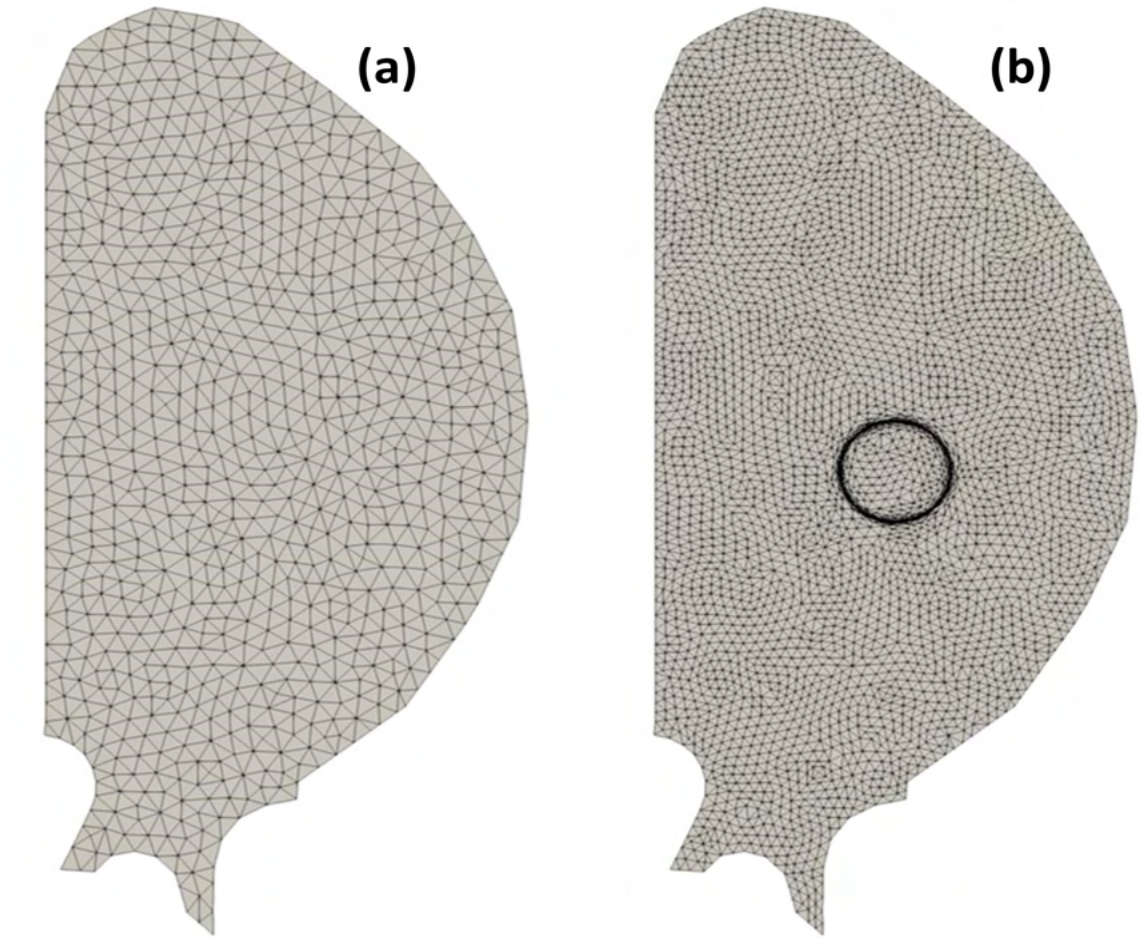

Figure 4.14: (a) Original coarse mesh, (b) modified mesh based on size field defined from an arbitrary analytical expression.

4.8 Selected Face Modification and Mesh Size Transitions

In certain cases, the mesh modification is desired only on certain model faces based on the a-priori knowledge of the physics (most commonly plasma core region). In such cases, it is desirable to modify the mesh only in such key areas with minimal altering the mesh in other areas. This is achieved using the PUMI mesh modification functionality by altering the

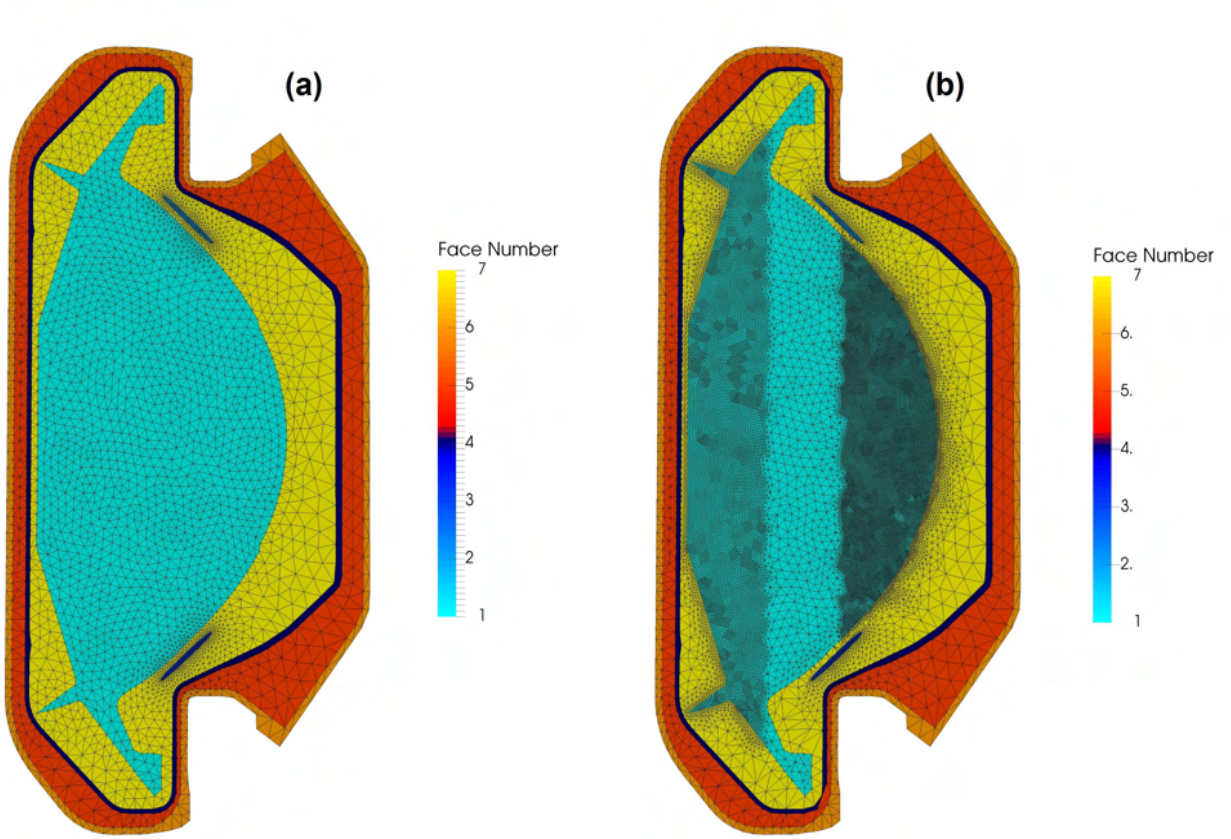

Figure 4.15: (a) The initial starting mesh with no modifications, (b) the mesh modified in the plasma core face and neighboring face.

mesh size in the face of interest. To ensure smooth mesh transitions, the mesh on the selected face is modified as requested and the mesh on neighboring faces starting from the common boundaries to the modified face is modified as needed to ensure a smooth mesh transition. Figure 4.15 demonstrates an example where the mesh size in the core plasma region was made finer in a specific spatially based manner over that face. The mesh modification procedures modified the mesh in that face to satisfy the requested mesh size as well as in neighboring faces as needed to ensure a smooth mesh size transition.

CHAPTER 5
ERROR ESTIMATION AND MESH ADAPTATION IN M3D-C^1

5.1 Introduction

Modeling analytically insoluble complex physical systems with governing differential equations requires numerical methods such as the finite element methods (FEM) to solve equations. Since the numerical methods approximate the solutions there are discretization errors in the solution. The focus of this chapter is the application of adaptive mesh control to reduce discretization error in the solution of a system solved by FEM. Since the exact solution is unknown in problems that are numerically solved, the exact error cannot be determined, however a-posteriori estimates of the error can be used to determine how to improve mesh most effectively to reduce the error. The a-posteriori error estimators employ the solution on the current mesh to determine element-level contributions to the discretization error.

The commonly used error estimators are divided into two basic types, residual-based error estimators and recovery-based error estimators. Common residual-based error estimators include explicit [71], [72], and implicit [73], [74] error estimators. In explicit error estimation, the residual from the approximated solution on the current mesh and problem data are used directly to estimate the error in the solution [75]. On the other hand, in implicit error estimation, the solution to auxiliary boundary value problems that employs the residuals are used to compute the estimated error [75]. An alternative method that is used in this work is a recovery-based error estimator [76], [77] often referred to as the Super-convergent Patch Recovery (SPR) method. In SPR, a typical C^{-1} field variable is "recovered" as a C^0 field that in some cases has been shown to be more accurate than the C^{-1} finite element field. The field values at the nodes are recovered by doing a least-square polynomial fit to integration point values of the select field using the integration points of the patch of elements that bound the node. The error for individual elements is then calculated by an element-level integration of the difference between the recovered and finite element fields.

In the M3D-C^1 adaptive simulation loop, the SPR method is used to estimate more accurate nodal values of plasma current density $\boldsymbol{J}$. A new mesh size field is defined using the elemental error estimates at selected time steps in the simulations. The mesh is updated based on the new mesh size field and the next simulation time step in the loop uses the

updated mesh.

This chapter provides an overview of the SPR error estimation and its application to mesh adaptation in M3D-C^1 simulations. Section 5.2 overviews the SPR error estimator and evaluation of the desired mesh size field from the estimated error. This section also discusses the solution field used for the estimation of the error. Section 5.3 summarizes the workflow for 2D mesh adaptation in M3D-C^1 and presents related results. The 2.5D mesh adaptation workflow and results are presented in section 5.4 and section 5.5 respectively.

5.2 SPR Error Estimation and Size Field Computation

Fundamentally, the discretization error $\boldsymbol{e}$ is the deviation of the finite element solution $\boldsymbol{\varepsilon_h}$ from the exact solution $\boldsymbol{\varepsilon}$ and can be expressed as an L_2 norm of this deviation as shown in equation 5.1.

$$||\boldsymbol{e}||_{L_2} = \Big(\sum_{e=0}^{n_e-1} \int_{\Omega} (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon_h})^T (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon_h}) d\Omega \Big)^{\frac{1}{2}} \tag{5.1}$$

where n_e is the number of elements in the domain Ω. Since the exact solution $\boldsymbol{\varepsilon}$ of the problem is not available, standard practice of mesh adaptation is to employ a-posteriori error estimators that numerically obtain an improved approximate solution $\boldsymbol{\varepsilon^*}$ for the selected field. The idea of estimated error $\boldsymbol{e^*}$ from the approximated solution $\boldsymbol{\varepsilon^*}$ can be represented as:

$$||\boldsymbol{e^*}||_{L_2} = \Big(\sum_{e=0}^{n_e-1} \int_{\Omega} (\boldsymbol{\varepsilon^*} - \boldsymbol{\varepsilon_h})^T (\boldsymbol{\varepsilon^*} - \boldsymbol{\varepsilon_h}) d\Omega \Big)^{\frac{1}{2}} \tag{5.2}$$

We employ a super-convergent patch recovery process proposed and developed by Zienkiewics and Zhu [76], [77] (SPR or Z-Z error estimator) for the recovery of solution at nodes that are used to fit a C^0 field defining $\boldsymbol{\varepsilon^*}$. In finite elements, there are certain points within each element where the accuracy of the field derivative is higher. Since the numerically obtained solution at these points converge faster compared to any other point, these points are referred as super convergent points [76]. The key goal of SPR is to recover a more accurate solution of the field at a node by "interpolation", using the field values at super-convergent points or near super convergent points of the elements in the nodal patch, where a nodal patch for a mesh vertex n is defined by the elements that contain n as one

of their vertices as shown in figure 5.1. This task of recovering the solution at node n is performed by a least-square fitting of a polynomial to the integration points of the nodal patch. The following paragraph overviews the procedure of field recovery at nodes.

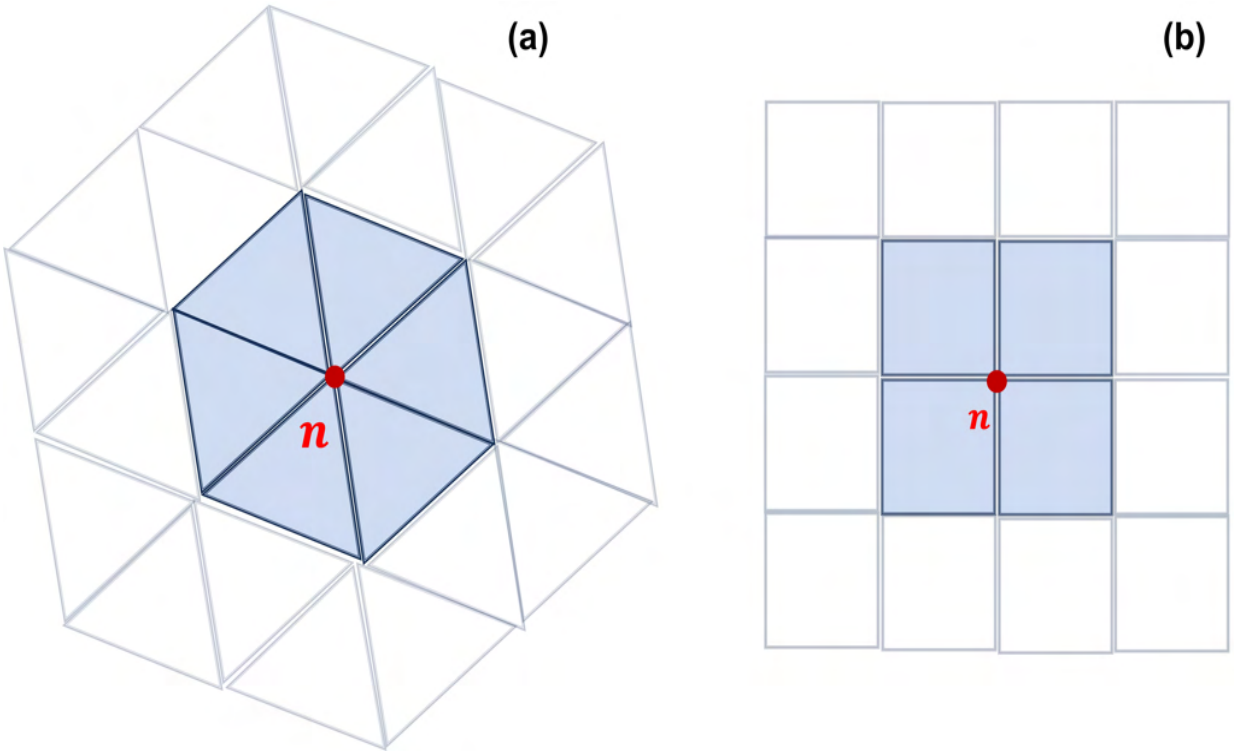

Figure 5.1: An example of patch defined at node n constituting (a) 2D triangular elements, (b) 2D quadrilateral elements.

At a given node n, let ε_n^* denote a component of the continuous field $\boldsymbol{\varepsilon}^*$ defined inside the nodal patch Ω (of node n). Within the patch, in order to eliminate the inter-elemental discontinuities, the structure of the scalar values field ε_n^* is assumed to be a p order polynomial in two variables as shown in equation 5.3. The order of polynomial p depends on the number of integration points in the elements of the patch.

$$\varepsilon_n^* = \boldsymbol{P}(R, Z)\boldsymbol{\alpha} \tag{5.3}$$

where (R, Z) notation is consistent with the coordinate system used in this work. As illustrated in figure 2.3, for a torus, the labels (R, Z, ϕ) correspond to the major radius, vertical axis and the toroidal angle, respectively. For 2D triangular elements with three integration points, we use quadratic polynomial fit to recover nodal values. For a quadratic fit, the vector of polynomial terms $\boldsymbol{P}$ and coefficients terms $\boldsymbol{\alpha}$ is written as:

$$\boldsymbol{P} = [1, R, Z, R^2, RZ, Z^2] \tag{5.4}$$

$$\boldsymbol{\alpha} = [\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5]^T \tag{5.5}$$

The task of recovering the nodal values to be used to define a C^0 continuous field translates to the evaluation of the unknown coefficients $\boldsymbol{\alpha}$. For a nodal patch, this is done by formulating a least-square fit as shown in equation 5.6. For each component of the solution field, the unknown coefficients ($\boldsymbol{\alpha}$) are obtained such that the sum of the squared deviation of the finite element solution ε^h and recovered solution ε_n^* is minimal. Mathematically, the least-square problem can be represented as shown below,

$$\textbf{Minimize} \quad \mathcal{J}(\boldsymbol{\alpha}) = \sum_{i=1}^{nk} (\varepsilon^h(R_i, Z_i) - \varepsilon_n^*(R_i, Z_i))^2 \tag{5.6}$$

where (R_i, Z_i) are the coordinates of the integration points in the patch, n is the number of elements in the patch Ω and k is number of integration points in each element.

To minimize $\mathcal{J}(\boldsymbol{\alpha})$ in equation 5.6, the partial derivatives of $\mathcal{J}(\boldsymbol{\alpha})$ with respect to the coefficients in the vector $\boldsymbol{\alpha}$ must be zero.

$$\frac{\partial \mathcal{J}(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}} = \frac{\partial \mathcal{J}(\boldsymbol{\alpha})}{\partial \alpha_0}, \frac{\partial \mathcal{J}(\boldsymbol{\alpha})}{\partial \alpha_1}, \cdots, \frac{\partial \mathcal{J}(\boldsymbol{\alpha})}{\partial \alpha_{K-1}} = 0 \tag{5.7}$$

where K is the number of coefficients in the vector α. Substituting expression 5.6 in equation 5.7, we obtain the following:

$$\frac{\partial \mathcal{J}(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}} = \sum_{i=1}^{nk} 2\Big(\varepsilon^h(R_i, Z_i) - \boldsymbol{P}(R_i, Z_i)\boldsymbol{\alpha}\Big)\Big(- \boldsymbol{P}(R_i, Z_i)\Big) = 0 \tag{5.8}$$

Expanding and rearranging the above equation, we obtain the relationship shown in equation 5.9.

$$\sum_{i=1}^{nk} \boldsymbol{P}^T(R_i, Z_i)\boldsymbol{P}(R_i, Z_i)\boldsymbol{\alpha} = \sum_{i=1}^{nk} \boldsymbol{P}^T(R_i, Z_i)\varepsilon^h(R_i, Z_i) \tag{5.9}$$

The vector $\boldsymbol{\alpha}$ is the polynomial coefficients that are determined by solving

$$\boldsymbol{A}\boldsymbol{\alpha} = \boldsymbol{b} \tag{5.10}$$

Here, $\boldsymbol{A} = \sum_{i=1}^{nk} \boldsymbol{P}^T(R_i, Z_i)\boldsymbol{P}(R_i, Z_i)$ and the vector $\boldsymbol{b} = \sum_{i=1}^{nk} \boldsymbol{P}^T(R_i, Z_i)\varepsilon^h(R_i, Z_i)$. Rewriting the above in a matrix form, we have

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ R_1 & R_2 & \dots & R_m \\ Z_1 & Z_2 & \dots & Z_m \\ R_1^2 & R_2^2 & \dots & R_m^2 \\ R_1Z_1 & R_2Z_2 & \dots & R_mZ_m \\ Z_1^2 & Z_2^2 & \dots & Z_m^2 \end{bmatrix} \begin{bmatrix} 1 & R_1 & Z_1 & R_1^2 & R_1Z_1 & Z_1^2 \\ 1 & R_2 & Z_2 & R_2^2 & R_2Z_2 & Z_2^2 \\ \vdots & \vdots & \vdots & & & \\ 1 & R_m & Z_m & R_m^2 & R_mZ_m & Z_m^2 \end{bmatrix} \tag{5.11}$$

$$\boldsymbol{b} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ R_1 & R_2 & \dots & R_m \\ Z_1 & Z_2 & \dots & Z_m \\ R_1^2 & R_2^2 & \dots & R_m^2 \\ R_1Z_1 & R_2Z_2 & \dots & R_mZ_m \\ Z_1^2 & Z_2^2 & \dots & Z_m^2 \end{bmatrix} \begin{bmatrix} \varepsilon^h(R_1, Z_1) \\ \varepsilon^h(R_2, Z_2) \\ \vdots \\ \varepsilon^h(R_m, Z_m) \end{bmatrix} \tag{5.12}$$

where $m = nk$. To avoid a singular matrix A, total number of integration points m in the patch must be greater than that of coefficients of the polynomial.

Once the solution ε_n^* at nodes is recovered using the above procedure, a continuous smooth recovered field $\boldsymbol{\varepsilon}^*$ that is used for the evaluation of discretization error is constructed. For each element, a continuous smooth field $\boldsymbol{\varepsilon}^*$ from ε_n^* is defined as:

$$\boldsymbol{\varepsilon}^*(\xi, \eta) = \sum_{n=1}^{n_e} N_n^e(\xi, \eta)\varepsilon_n^* \tag{5.13}$$

where n_e is the number of nodes in the element, and N_a^e are the element shape functions. From the recovered field $\boldsymbol{\varepsilon}^*$, an L_2 norm is used to compute the discretization error over the element as shown in equation 5.14.

$$||\boldsymbol{e^*}||_{L_2} = \Big(\int_{\Omega_e} (\boldsymbol{\varepsilon}^* - \boldsymbol{\varepsilon_h})^T (\boldsymbol{\varepsilon}^* - \boldsymbol{\varepsilon_h}) d\Omega_e \Big)^{\frac{1}{2}} \tag{5.14}$$

The scaler elemental size field from the estimated error $\boldsymbol{e^*}$ is evaluated using the expression:

$$h_e^{new} = h_e^{current} ||\boldsymbol{e}^*||_e^{-\frac{2}{2p+d}} \left(\frac{\hat{\eta^2} ||\boldsymbol{\varepsilon}^*||^2}{\sum_{e=0}^{n_e-1} ||\boldsymbol{e}^*||_e^{\frac{2}{2p+d}}} \right)^{\frac{1}{2p}} \tag{5.15}$$

where h_e^{new} is the desired element size, $h_e^{current}$ is the current element size defined by longest edge, $||\boldsymbol{e}^*||_e$ is the discretization error over an element, d is the element dimension, p is the polynomial order of the recovered field, $\hat{\eta}$ is a threshold parameter for adaptivity, and n_e is the number of elements in the mesh.

5.2.1 Solution Field Used for Error Estimation

Plasma equilibrium is of prime importance in the operation of tokamaks. The magnetic forces and pressure of the plasma must be balanced to achieve equilibrium in the plasma. The current density (J), along with the magnetic field B, defines the plasma pressure. Hence, the accurate calculation of plasma current density during the simulations is critical in studying the equilibrium in plasma [78]. The plasma current density is closely related to other plasma-defining quantities, including plasma shape and toroidal magnetic field [79]; therefore, an accurate representation of the current density plays a critical role in studying the confinement of plasma and its stability. In the permeability normalized implementation in M3D-C^1, the plasma current density (J) is calculated from the curl of the magnetic field (B) as

$$\boldsymbol{J} = \nabla \times \boldsymbol{B} \tag{5.16}$$

Expanding equation 5.16 gives:

$$\boldsymbol{J} = \nabla \times \boldsymbol{B} = -\frac{\partial B_\phi}{\partial Z}\hat{R} + (\frac{\partial B_R}{\partial Z} - \frac{\partial B_Z}{\partial R})\hat{\phi} + \frac{1}{R}\frac{\partial (RB_\phi)}{\partial R}\hat{Z} \tag{5.17}$$

where B_R, B_Z, and B_ϕ are the components of the magnetic field in the R, Z, and toroidal direction ϕ respectively as shown in figure 2.5. The plasma current density can be described in terms of poloidal current density (J_p) and toroidal current density (J_ϕ) as

$$\boldsymbol{J} = J_p + J_\phi = [-\frac{\partial B_\phi}{\partial Z}\hat{R} + \frac{1}{R}\frac{\partial (RB_\phi)}{\partial R}\hat{Z}] + [(\frac{\partial B_R}{\partial Z} - \frac{\partial B_Z}{\partial R})\hat{\phi}] \tag{5.18}$$

The toroidal current density is defined in terms of components of the magnetic field as

$$J_\phi = (\frac{\partial B_R}{\partial Z} - \frac{\partial B_Z}{\partial R}) \tag{5.19}$$

The components of the magnetic field (B_R and B_Z) along R and Z directions in terms of magnetic flux field (ψ) are

$$B_R = -\frac{1}{R}\frac{\mathrm{d}\psi}{\mathrm{d}Z} \tag{5.20}$$

$$B_Z = \frac{1}{R}\frac{\partial \psi}{\partial R} \tag{5.21}$$

Substituting the values of B_R and B_Z in equation 5.19 gives

$$J_\phi = \frac{\partial}{\partial Z}(-\frac{1}{R}\frac{\partial \psi}{\partial Z}) - \frac{\partial}{\partial R}(\frac{1}{R}\frac{\partial \psi}{\partial R}) \tag{5.22}$$

$$J_\phi = -\frac{1}{R}[\,\frac{\partial^2 \psi}{\partial Z^2} - \frac{1}{R}\frac{\partial \psi}{\partial R} + \frac{\partial^2 \psi}{\partial R^2}\,] \tag{5.23}$$

The toroidal current density is defined in term of an elliptic operator (∇^*) operating on ψ as:

$$J_\phi = -\frac{1}{R}.\nabla^*\psi \tag{5.24}$$

The primary fields in M3D-C^1 are C^1 continuous, which are continuous in value and first derivative. Hence, the fields that are defined by the second-order derivatives are the first ones to be C^-1. Since the J_ϕ is directly related to the second-order derivatives of the ψ field, a field depending on the gradient of the $\psi_{,R}$ and $\psi_{,Z}$ is used in the error estimation.

5.3 2D Mesh Adaptation

The 2D M3D-C^1 simulations are carried out on 2D poloidal plane meshes. In a 2D adaptive workflow, the target solution field ($\nabla(\psi_{,R}, \psi_{,Z})$) for error estimation from the current mesh is collected at predefined time-step (defined by an interval n). The SPR error estimation method is used to recover a more accurate solution field at the nodes from the given solution field and is also used to compute the discretization error at the element level. The estimated error from SPR is used to compute a new mesh size field by using the equation

5.15. Afterwards, the new mesh size field is used to adapt 2D poloidal plane mesh using the PUMI mesh adaptation procedures [61], [62]. This adapted mesh is used in the next time-steps as the input mesh until we adapt mesh again at the next predefined time-step. Note that a better algorithm to define the interval at which mesh adaptation operation is carried out is in the future plans (see chapter 6).

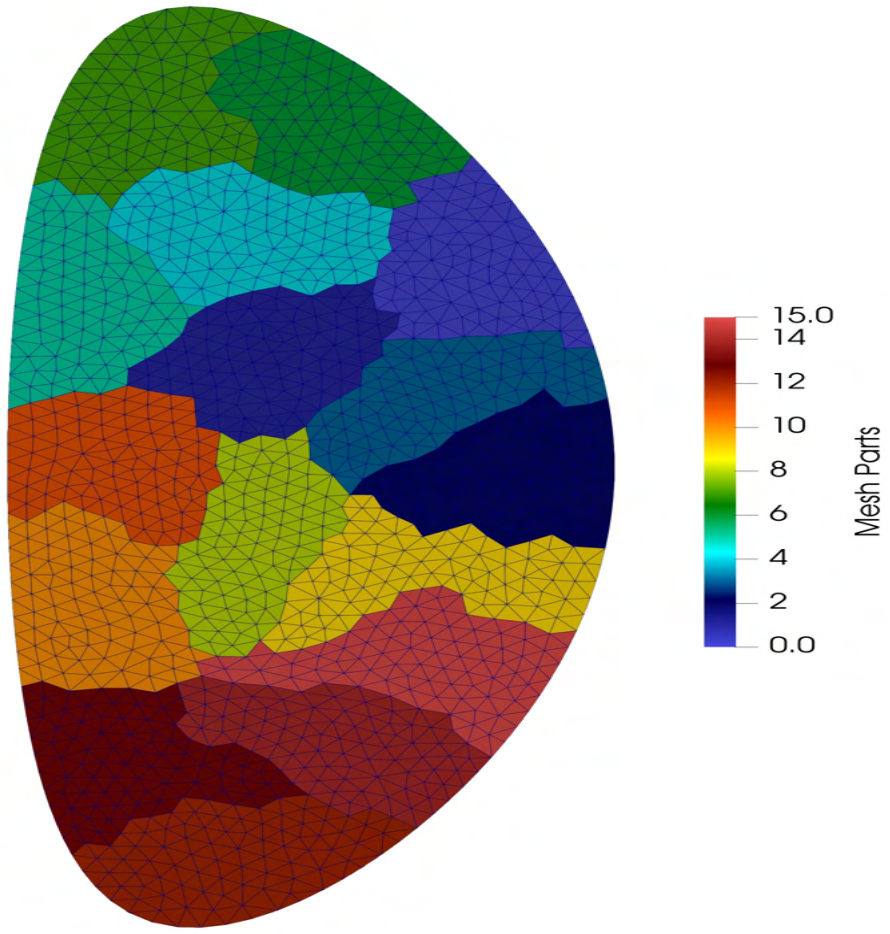

Figure 5.2: 16-part initial tokamak mesh.

The a-posteriori mesh adaptation procedures described in the sections above are applied to a coarse 16-part tokamak mesh as shown in figure 5.2 in a 2D M3D-C^1 simulation. The initial mesh is adapted at an interval of 10 time-steps and simulation was run for a total of 701 time-steps. The results for the field of interest (toroidal current density (J_ϕ)) are provided. The adapted meshes at every 100^{th} time-step are shown. Also, the solutions at very next time-step from mesh adaptation are provided.

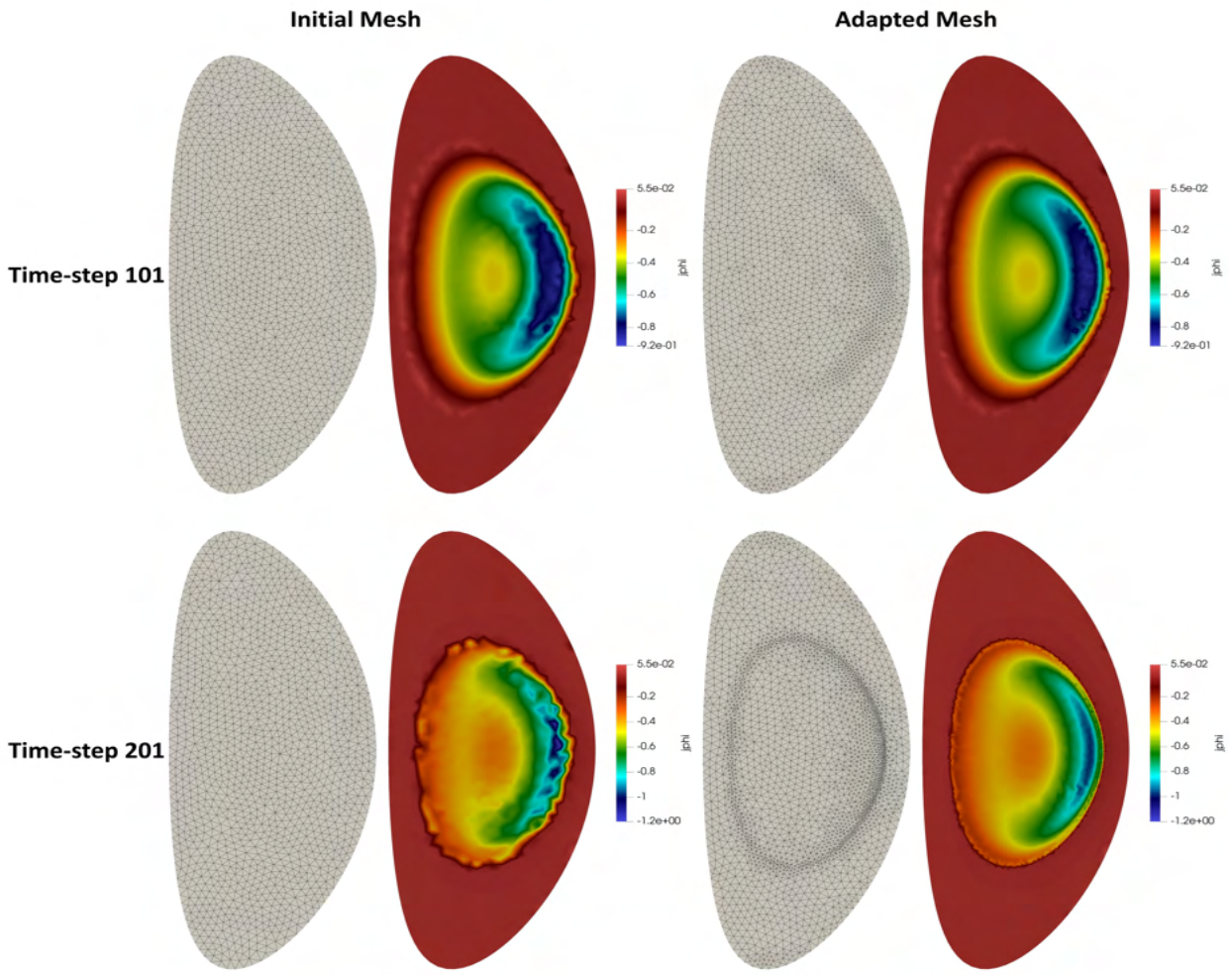

Figure 5.3: The initial and adapted meshes and solutions at time-step 101, 201. Note that the mesh was adapted using the solution field from time-step 100, 200.

The plasma profile changes over the time in the M3D-C^1 simulations and the goal of the mesh adaptation is to track the change in equilibrium profile and refine the mesh accordingly for high quality simulation results. Figure 5.3 represents the mesh adapted at time-step 100 in the region where the lowest J_ϕ values define a distinct region (marked as blue). The adapted mesh in this region resulted in a smoother result compared to the non-adapted mesh at time-step 101 as shown in 5.3. The J_ϕ changes over time and mesh is being refined in the region that marks the outer boundary of the profile at time-step 201 as shown in figure 5.3. There is some noise in the solution, but it is visibly less in all the regions for the adapted mesh over that of the solution on non-adapted mesh at time-step 201.

The results on the adapted meshes demonstrate minimal noise at time-steps 301 and

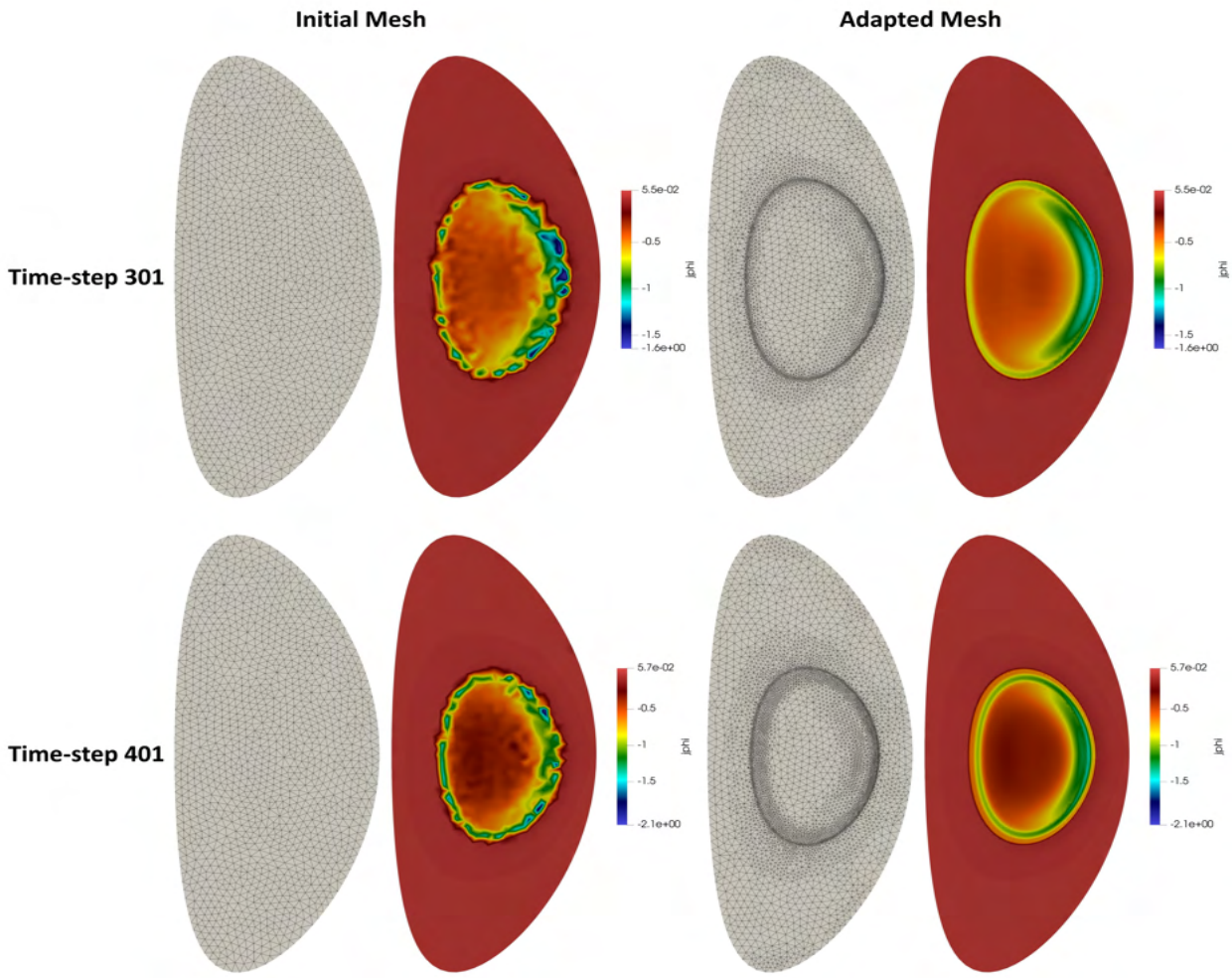

Figure 5.4: The initial and adapted meshes and solutions at time-step 301, 401. Note that the mesh was adapted using the solution field from time-step 300, 400.

401 as shown in figures 5.4. At time-step 501, an inner circle showing the J_ϕ profile is starting to be established, and the mesh is refined accordingly in that region as shown in figure 5.5 resulting in well-defined results at time-step 501. At time-steps 601 and 701, the profile rings that are aligned with flux curves are visible on the adapted mesh as shown in figures 5.5 and 5.6. At this stage, the resulting solution on the adapted mesh shows a well defined current density profiles with no distortion. On the other hand, the solution on the non-adapted initial mesh is clearly not defining the distinct density profiles.

Figure 5.7a shows a comparison of simulation times on the original and adapted meshes. The time difference and the element count in initial 200 time-steps is insignificant. However, the plasma profile starts to change rapidly somewhere between time-steps 200 and 300 as

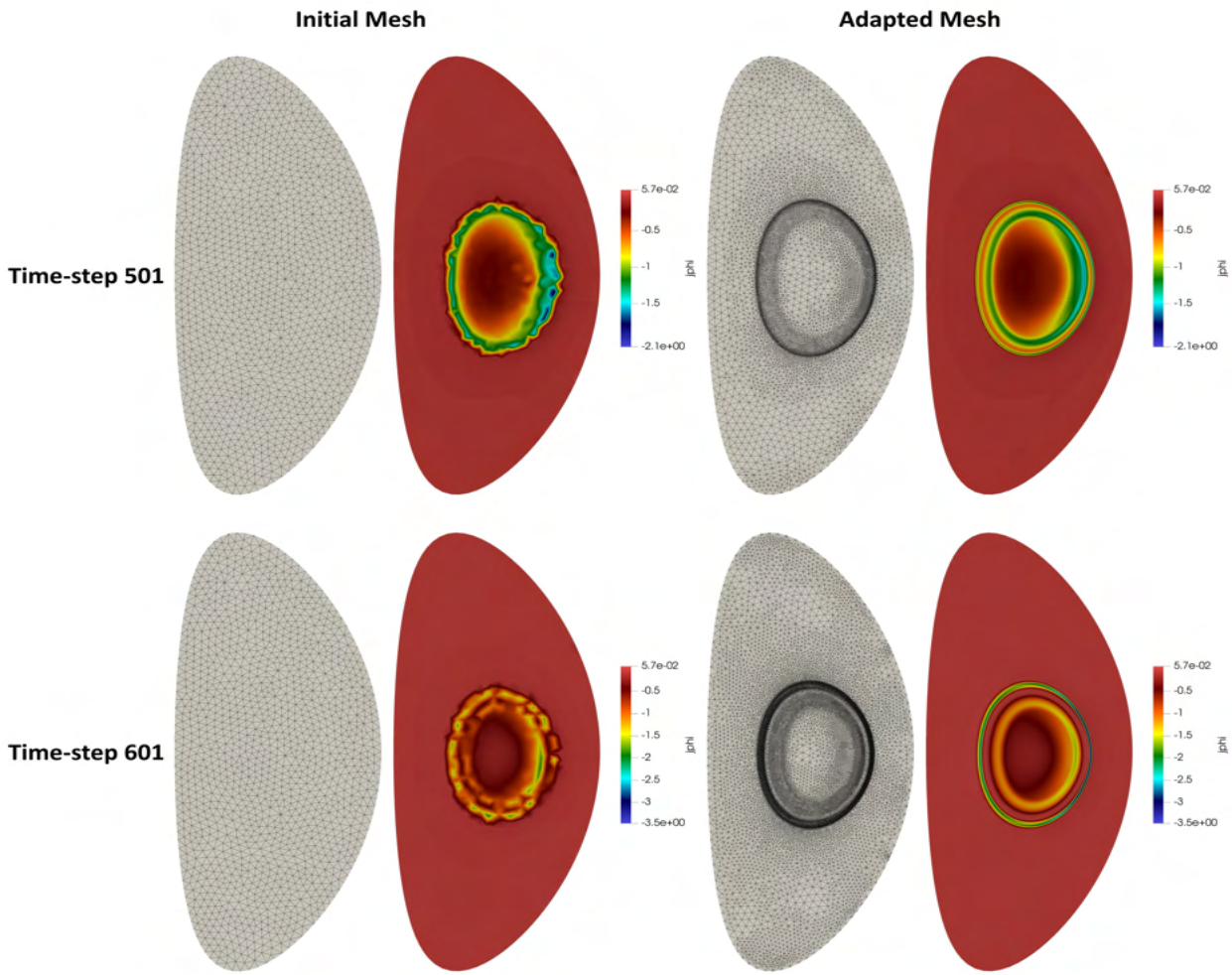

Figure 5.5: The initial and adapted meshes and solutions at time-step 501, 601. Note that the mesh was adapted using the solution field from time-step 500, 600.

can be confirmed from the earlier presented results too, and to account for those changes the mesh is extremely refined in certain critical areas of the mesh. This increase in the refinement of the mesh results in much higher simulation times as can be seen in figure 5.7a. Figure 5.7b demonstrates the count of elements on the adapted mesh at different steps as compared to the number of elements on the original mesh. The mesh is most refined at time-step 570 with 105 thousand elements which is roughly 40 times higher than the original mesh. In the present simulation, the plasma changes gradually for the first 500 time-steps and after that the change in the plasma profile is drastic. This trend is visible in figure 5.7b, where it can be seen that the number of elements are oscillating to account for those drastic changes in the plasma between time-step 500 to time-step 700.

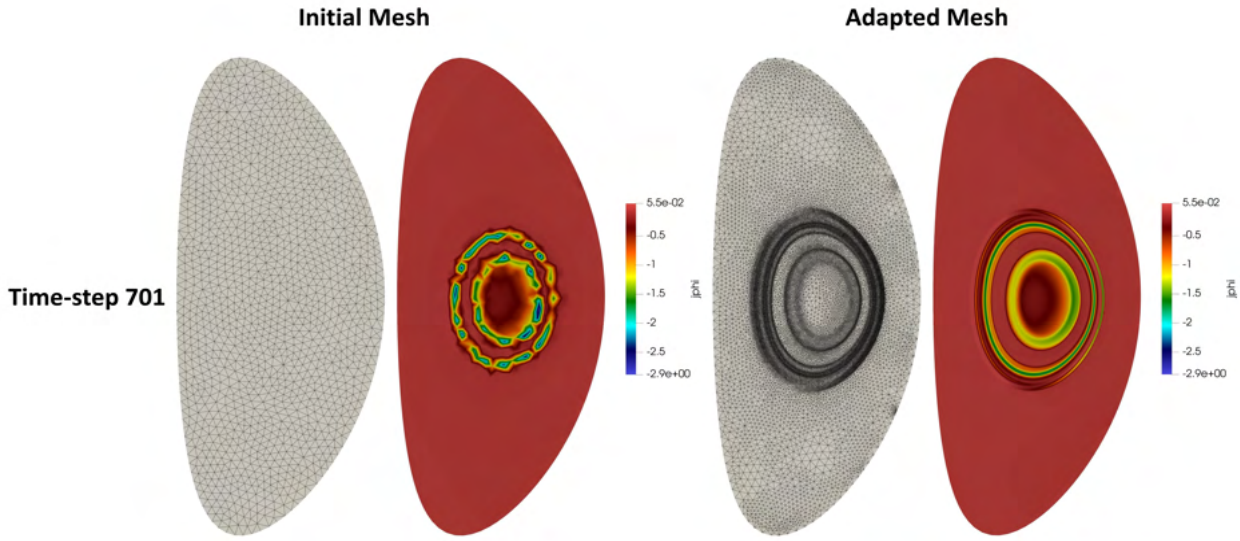

Figure 5.6: The initial and adapted meshes and solutions at time-step 701. Note that the mesh was adapted using the solution field from time-step 700.

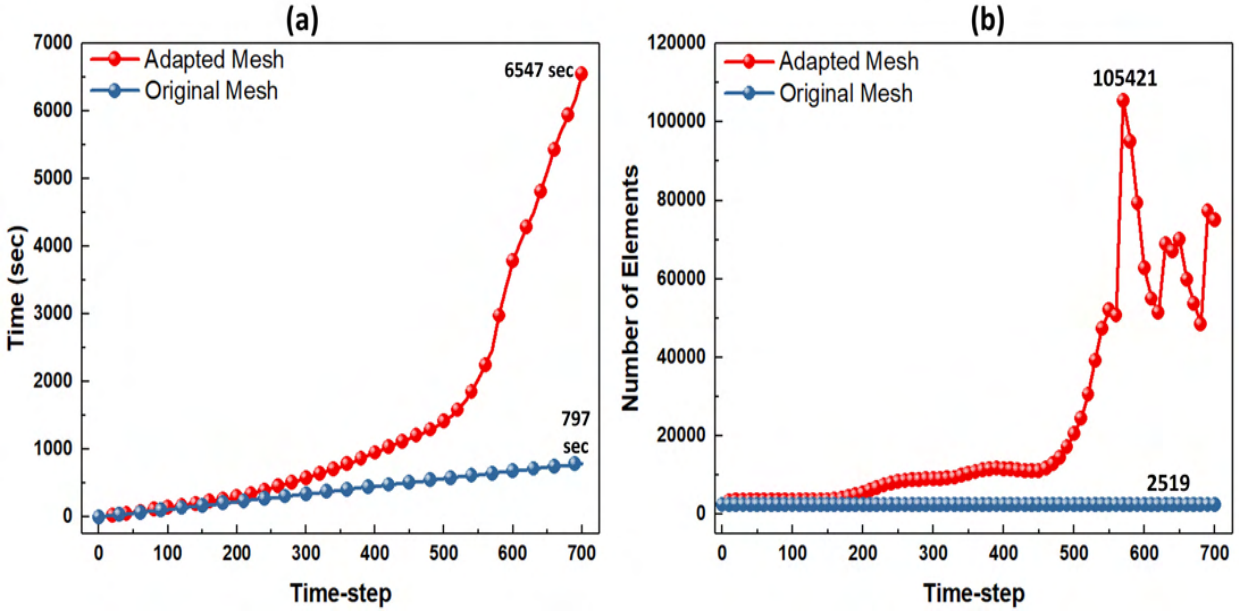

Figure 5.7: (a) Comparison of simulation time on original and adapted meshes, (b) comparison of number of elements on original and adapted meshes. The number of elements on original mesh stays constant throughout the simulation.

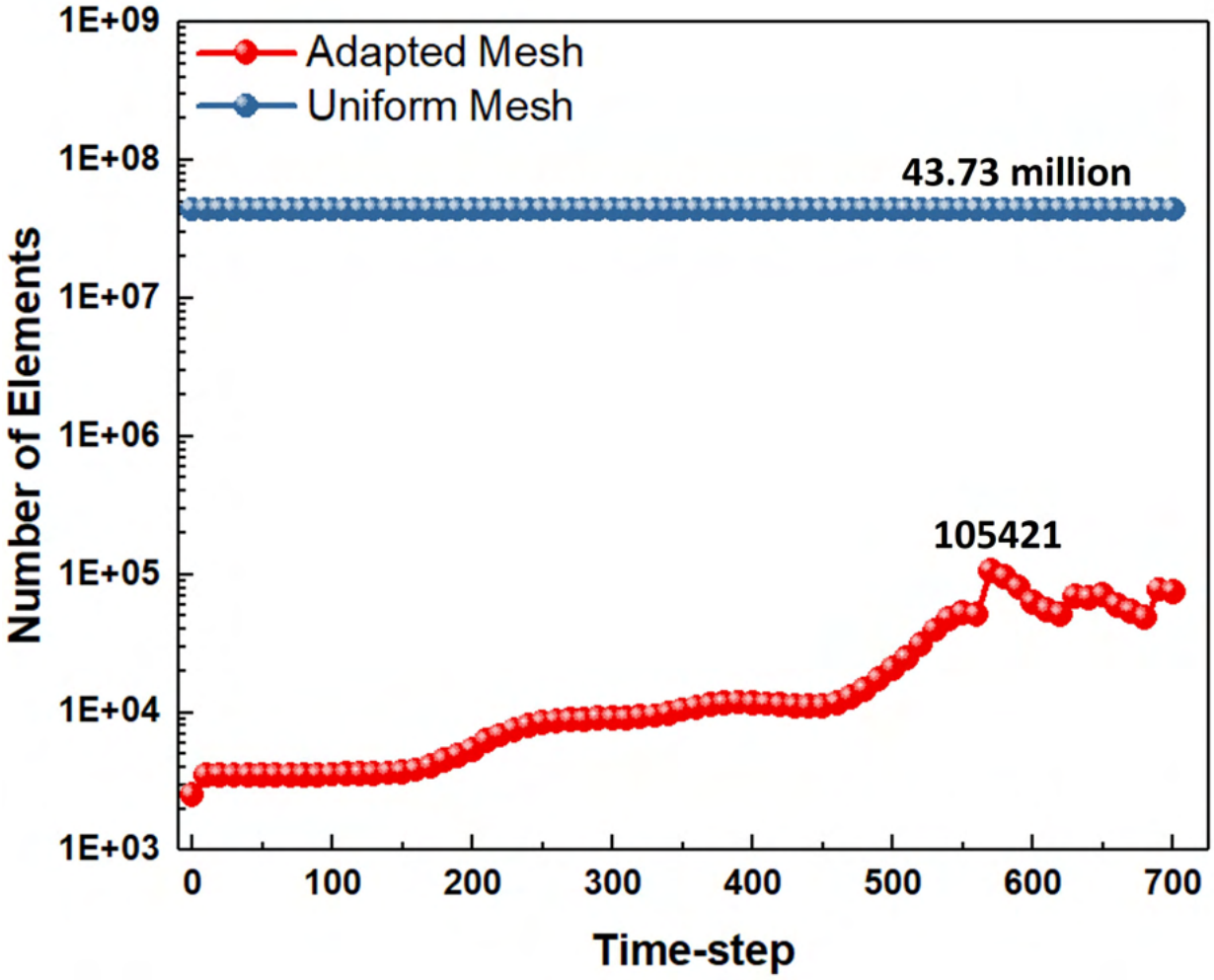

Figure 5.8: Comparison of number of elements on adapted mesh and uniform mesh with a size equal to smallest edge length in adapted mesh.

The smallest edge length L_s in the adapted mesh during the simulation is 0.000677 meters. The parts of mesh refined to this size at time-step 570 when number of elements were 105 thousand. A uniform mesh was generated with a mesh size equal to L_s, which resulted in 43.73 million elements. To get a similar accuracy in results with a uniform mesh as was shown with the adapted mesh, the simulation requires roughly 400 times more elements and computing time than the adapted mesh. A comparison of the number of elements is demonstrated in figure 5.8.

5.4 2.5D Mesh Adaptation

For 3D simulations, wedge elements are constructed on between 32 to 128 poloidal planes using C^1 cubic Hermite polynomials in the toroidal direction, which when combined with the C^1 triangular elements, yields the desired C^1 volume elements. Since there are multiple planes, the target mesh size field based on the error estimate is different on all the planes. However, since the only available 3D element is a wedge, all poloidal planes must be refined to be the same using a single poloidal plane mesh size field. To achieve this, the method used is to collect the size fields from all the planes to the master plane and set the size field to the smallest of the requested size field from all the planes for every node yielding a 2.5D mesh adaptation procedure. The workflow for the 2.5D mesh adaptation for N planes is presented below:

- Extract the target solution field for error estimation from N planes.
- Transfer the fields requested for solution transfer from non-master to master plane.
- Compute element error for each of the N planes.

$$||\boldsymbol{e^*}||_{(e,p)} = \Big(\int_{\Omega_{(e,p)}} (\boldsymbol{\varepsilon^*} - \boldsymbol{\varepsilon_h})^T (\boldsymbol{\varepsilon^*} - \boldsymbol{\varepsilon_h}) d\Omega_{(e,p)} \Big)^{\frac{1}{2}} \tag{5.25}$$

  where e ranges from 0 to $n_e - 1$, and p ranges from 0 to $N - 1$.

- Compute the desired size field (new edge length $h^{new}_{(e,p)}$) for each element of N planes using equation 5.15.
- Set the size field on every node of the master plane ($N = 0$) to the minimum of the N size fields.

$$h^{new}_{(e,0)} = min\big(h^{new}_{(e,p)}\big) \tag{5.26}$$

- Adapt the 2D mesh on the master plane.
- As the mesh is being adapted, the local solution transfer is carried out for each of the poloidal planes.
- Reconstruct the adapted 3D mesh by extrusion.

5.5 2.5D Results

This section demonstrates the application of 2.5D mesh adaptation procedures to two M3D-C^1 simulation cases. In the first case, resonant magnetic perturbations (RMPs) are used to control the edge-localized modes (ELMs) instabilities in the plasma. The second test case uses pellet injections to mitigate the plasma disruptions.

5.5.1 RMP

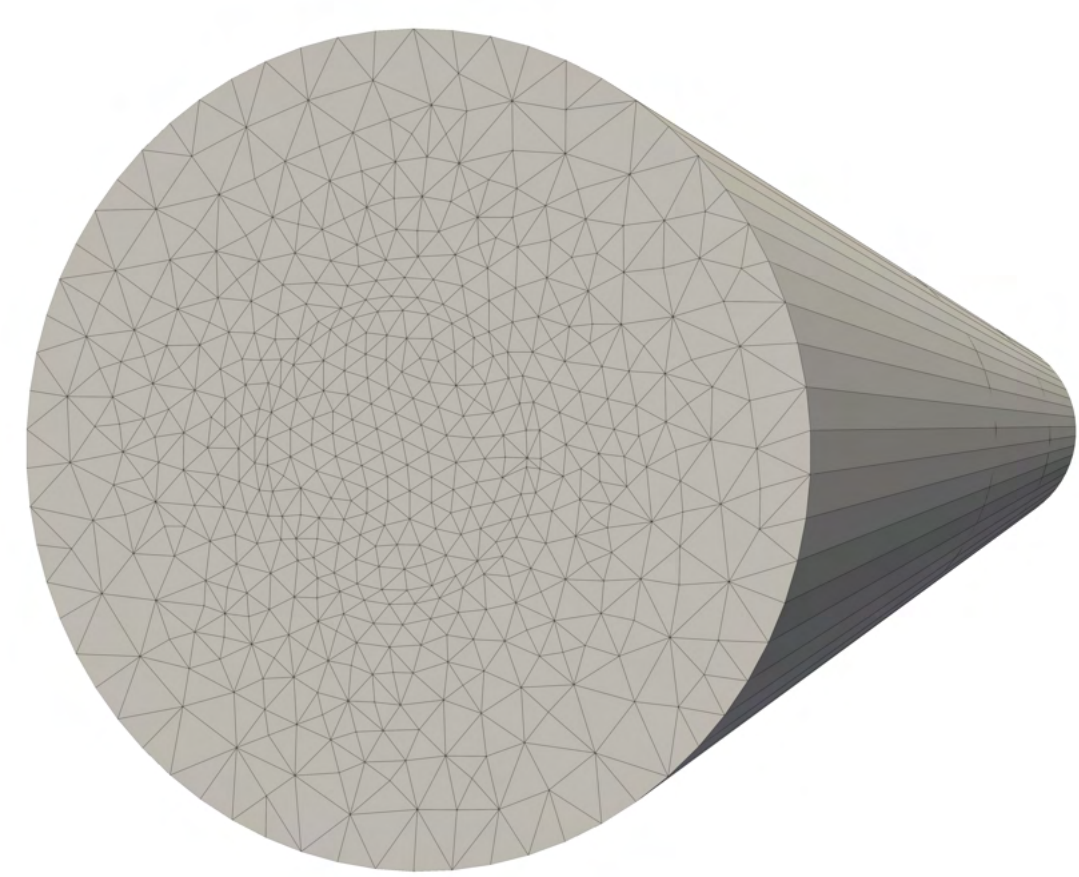

Figure 5.9: Initial mesh with four poloidal cross sections.

The a-posteriori 2.5D mesh adaptation procedure are applied to a 128-part mesh with 4 poloidal plane as shown in figure 5.9. The 3D mesh is created by the extrusion of a 32-part mesh on a circular cross section to 4 poloidal planes. The mesh is adapted at an interval of 50 time-steps and simulation was run for a total of 251 time-steps. The results for the

toroidal current density (J_ϕ) and its derivative in toroidal direction ($J_{\phi,\phi}$) after every mesh adaptation step are presented here.

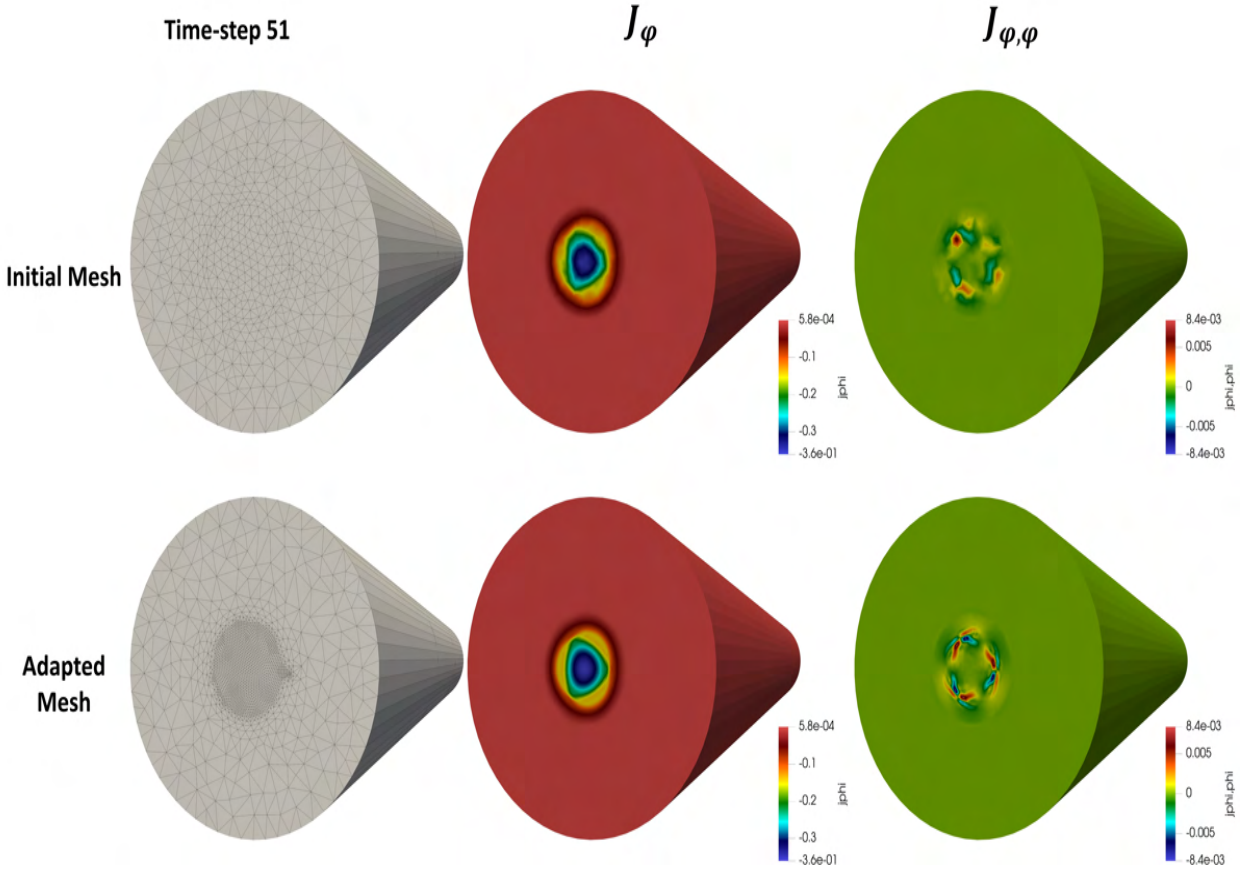

Figure 5.10: The initial and adapted meshes and solutions at time-step 51 for the RMP case. Note that the mesh was adapted using the solution field from time-step 50.

Figures 5.10, and 5.11 show that the toroidal current density profile is smoothly defined on the adapted mesh as compared to the profile on the original mesh. As can be seen from the results, the plasma profile doesn't change significantly over time. However, the rate of change in toroidal current density along the torus has significantly complex profile, that needs fine mesh resolution for the accurate representation. It can be seen in figures 5.12, 5.13, and 5.14 that mesh resolution is changing dynamically with time-steps to accurately simulate $J_{\phi,\phi}$ profiles. The simulations on the adapted mesh showed improved results as compared to the simulations on the original mesh for all time-steps.

Figure 5.15a shows a comparison of simulation times on the original and adapted meshes. The simulation time on the adapted mesh started to increase significantly after first

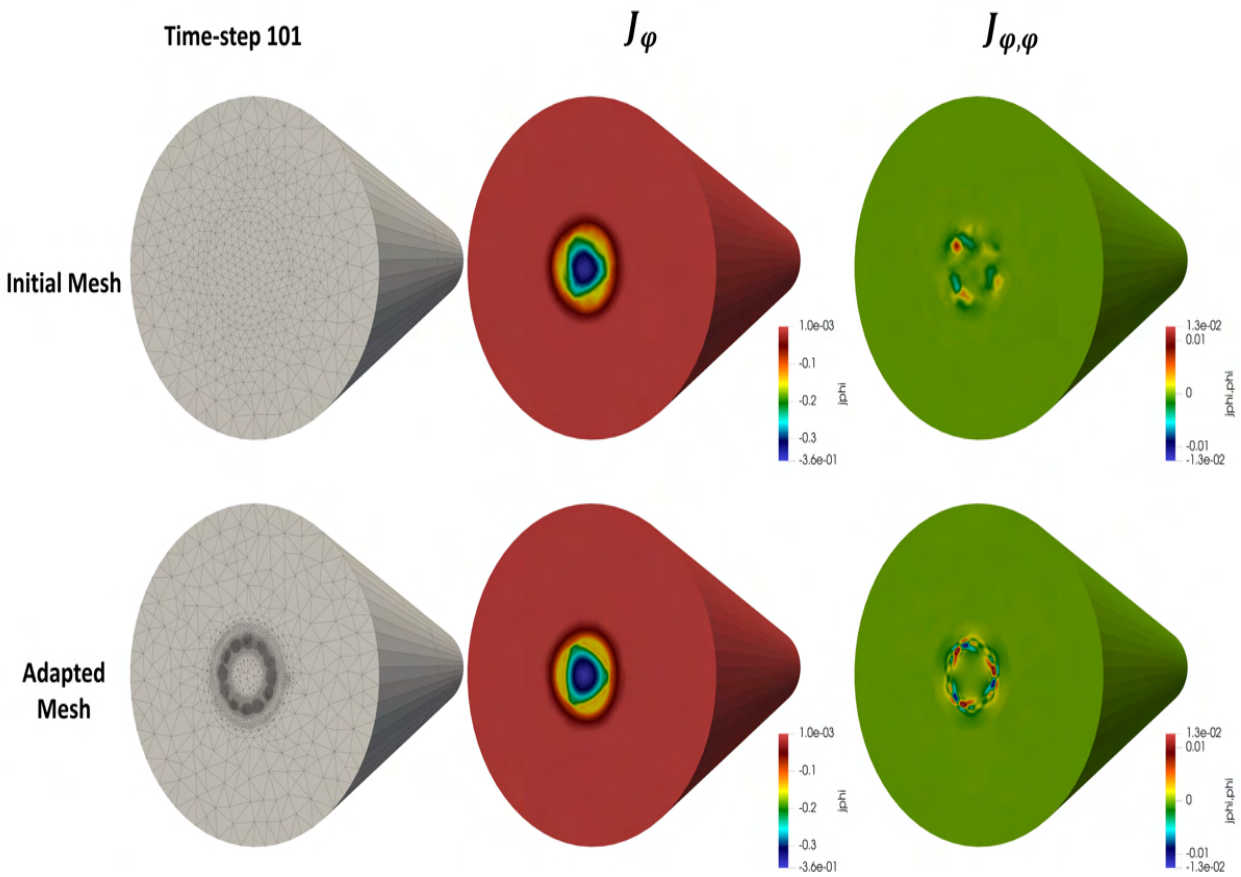

Figure 5.11: The initial and adapted meshes and solutions at time-step 101 for the RMP case. Note that the mesh was adapted using the solution field from time-step 100.

adaptation at time-step 50. However, the time curve slope became even steeper after second mesh adaptation at time-step 100 due to big jump in the number of elements as can be seen in figure 5.15b. Since there are no big changes in plasma profile, once the desired resolution to effectively simulate plasma is achieved, the changes in mesh resolution in the next mesh adaptation iterations are small which are visible from the plot line in figure 5.15b.

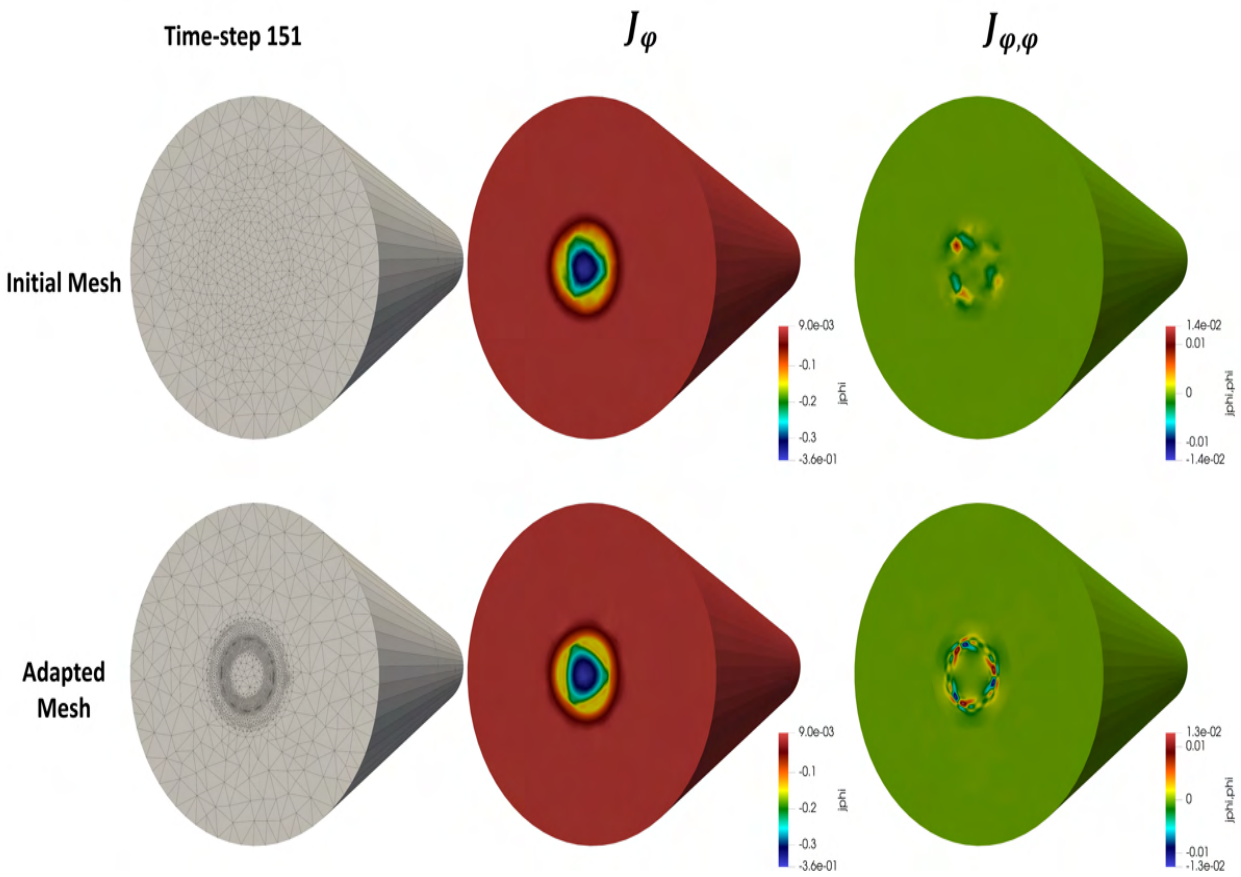

Figure 5.12: The initial and adapted meshes and solutions at time-step 151 for the RMP case. Note that the mesh was adapted using the solution field from time-step 150.

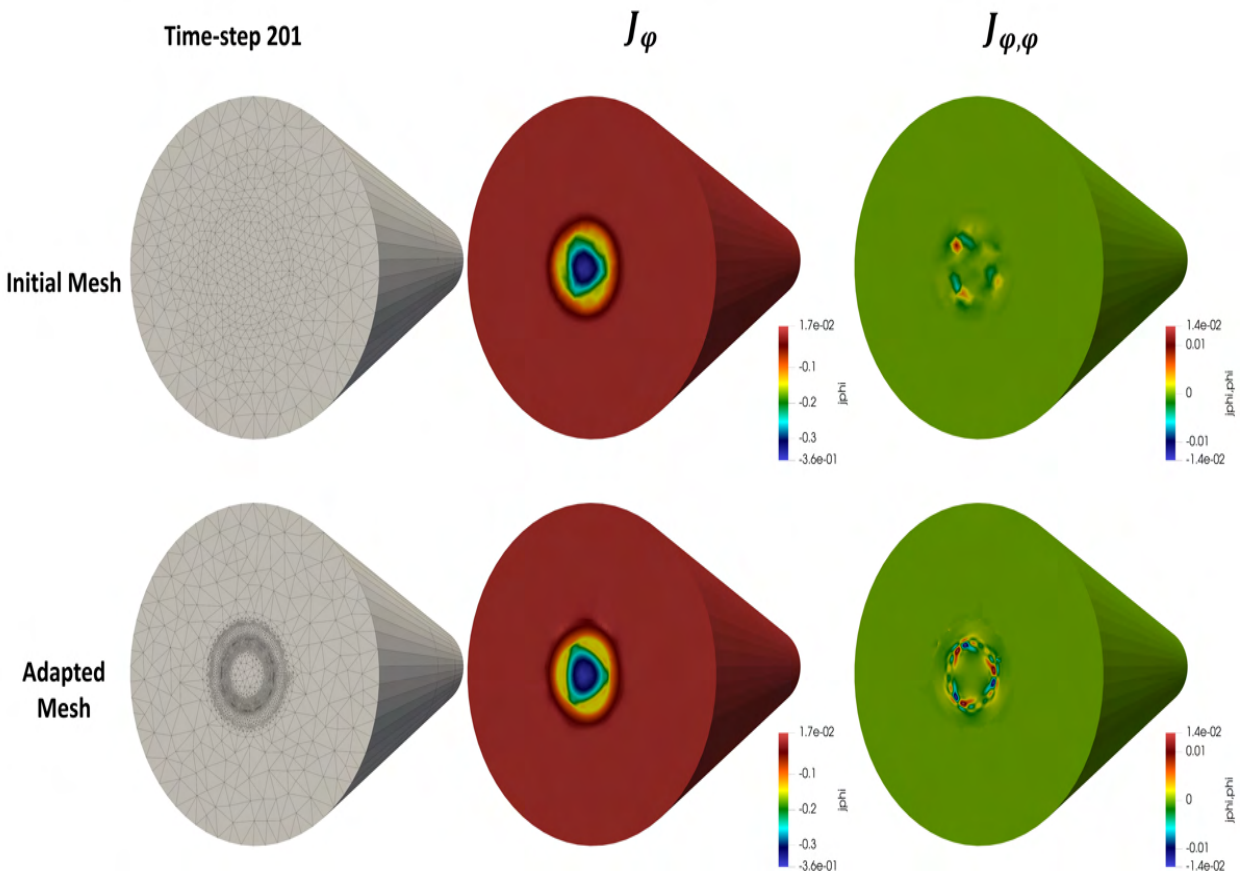

Figure 5.13: The initial and adapted meshes and solutions at time-step 201 for the RMP case. Note that the mesh was adapted using the solution field from time-step 200.

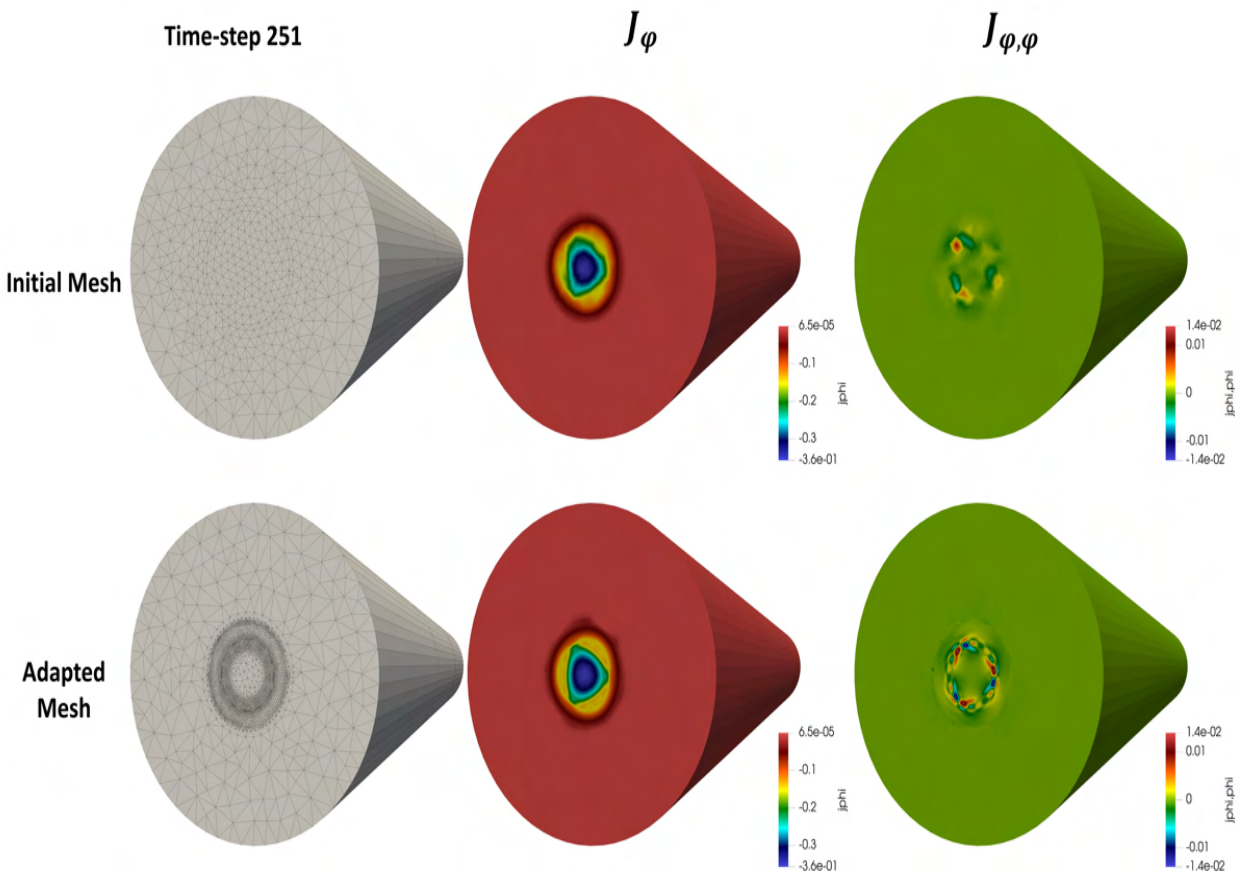

Figure 5.14: The initial and adapted meshes and solutions at time-step 251 for the RMP case. Note that the mesh was adapted using the solution field from time-step 250.

The smallest edge length L_s in the adapted mesh during the RMP simulation is 0.007627 meters. A uniform mesh was generated with a mesh size equal to L_s, which resulted in 6.27 million elements. To get a similar accuracy in results with a uniform mesh as was shown with the adapted mesh, the simulation requires roughly 62 times more elements than the adapted mesh. A comparison of the number of elements is demonstrated in figure 5.16.

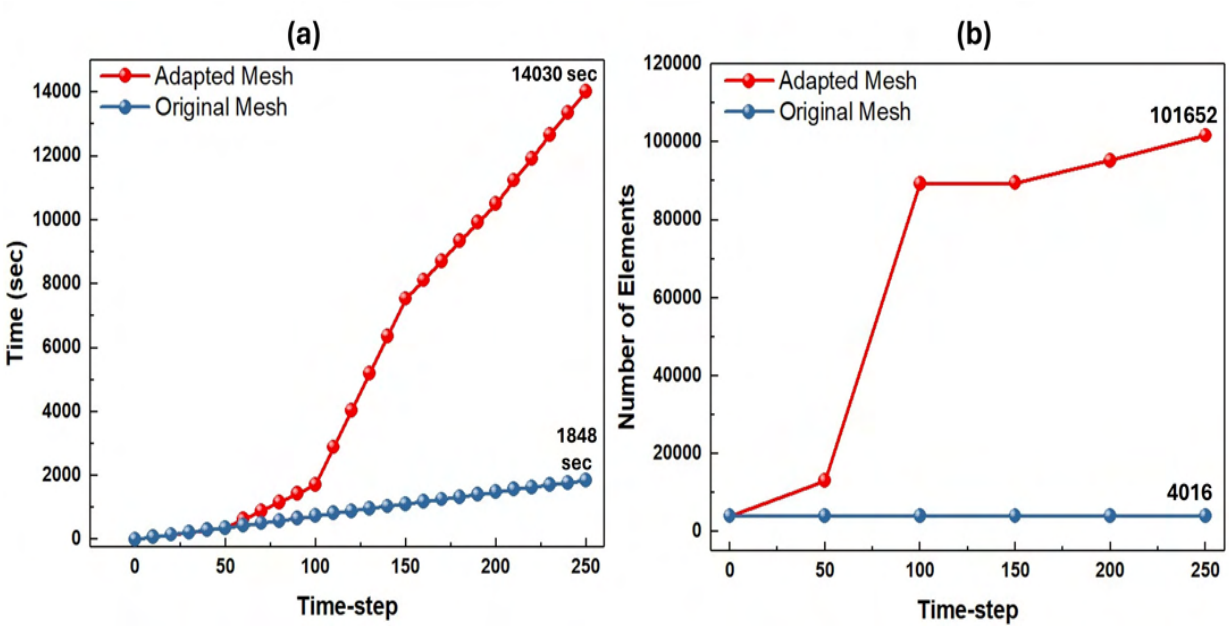

Figure 5.15: (a) Comparison of simulation time on original and adapted meshes for RMP case, (b) comparison of number of elements on original and adapted meshes. The number of elements on original mesh stays constant throughout the simulation.

5.5.2 Pellet Case

The RMP case presented above is stable with no major changes in plasma profile. The plasma profile in Pellet case presented in this section changes significantly over time. An initial 32-part 2D poloidal plane is extruded to 4 poloidal plane to get a 128-parts 3D mesh as shown in figure 5.17. The mesh is adapted at an interval of 50 time-steps and simulation was run for a total of 251 time-steps. The results for the toroidal current density (J_ϕ) and its derivative in toroidal direction ($J_{\phi,\phi}$) after every mesh adaptation step are presented.

Figure 5.18 shows that J_ϕ profile on the initial mesh at time-step 50 is well defined, however the J_ϕ profile at time-step 100 started to show a noise on the original mesh as can be seen in figure 5.19. This distortion in profile became more evident in later time-steps which is

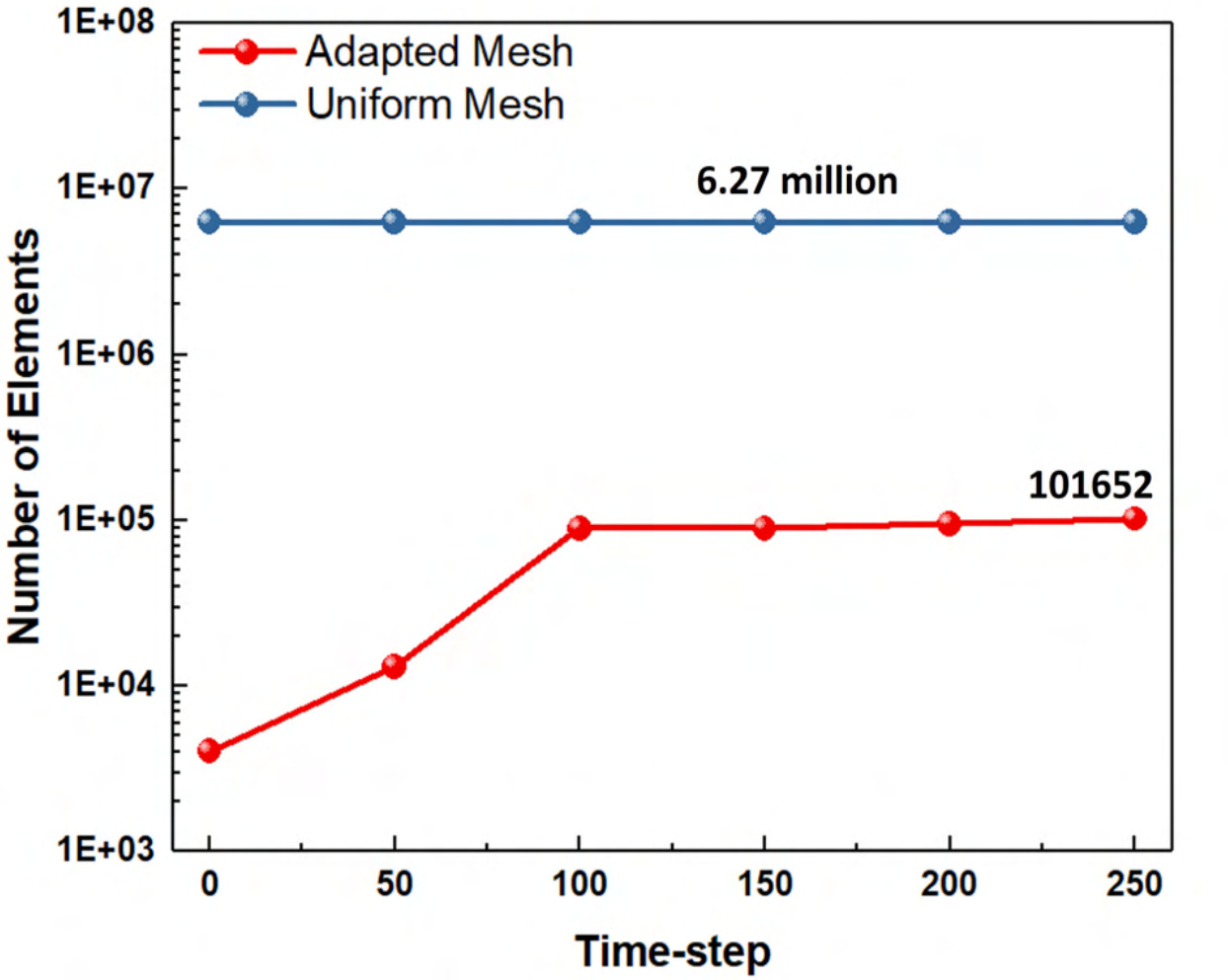

Figure 5.16: Comparison of number of elements on adapted mesh and uniform mesh with a size equal to smallest edge length in adapted mesh for the RMP case.

visible in figures 5.20, 5.21, and 5.22. However the results on the adapted mesh show smooth J_ϕ profiles at all time-steps demonstrating the effectiveness of mesh adaptation procedures in these simulations. On the other hand, the $(J_{\phi,\phi})$ profiles on the original mesh at all time-steps are poorly defined while on the adaptive mesh they are significantly improved. These improvements are more prominent at time-step 101, and 151 as shown in figures 5.19 and 5.20 where the distortions in the $(J_{\phi,\phi})$ profiles at the original mesh were large.

Figure 5.23a shows a comparison of simulation times on the original and adapted meshes. The simulation time curve on the adapted mesh shows nearly linear trend after mesh is adapted first time at time-step 50. The total simulation time on adapted mesh is nearly 9 times higher than that of the initial mesh. Since the number of elements on the

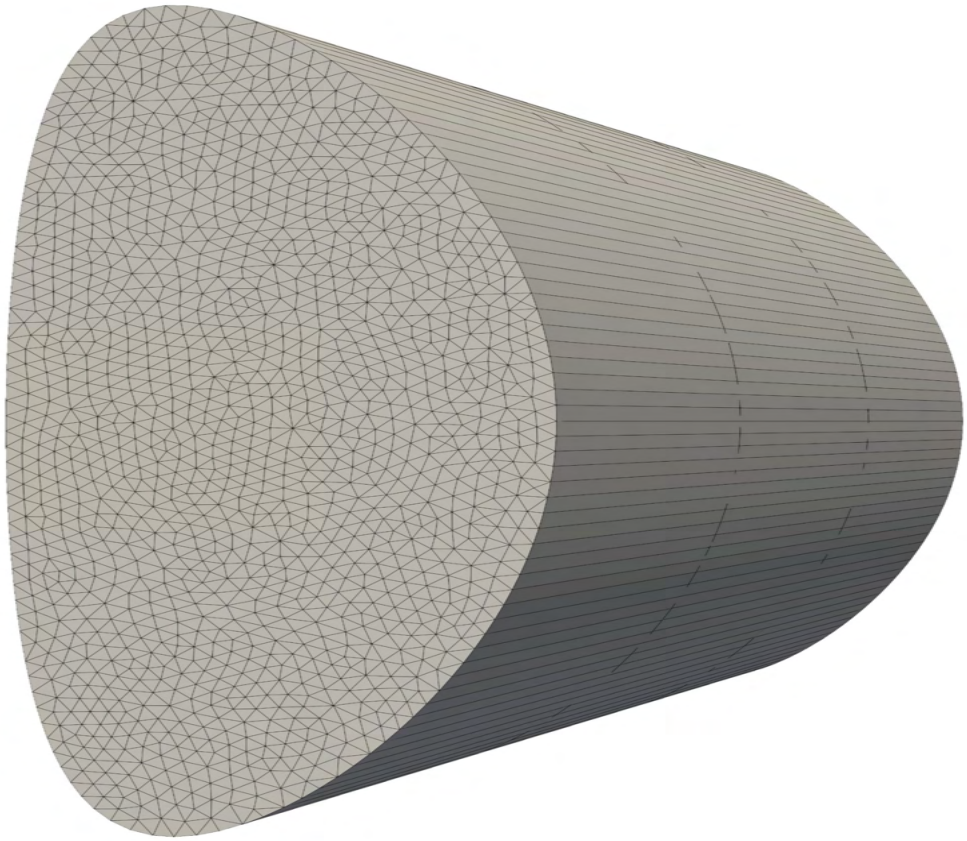

Figure 5.17: Initial mesh with four poloidal cross sections.

adapted mesh are roughly 8-10 higher than that of the original mesh for major part of the simulation, the total simulation time has direct relation to the number of elements in the mesh. Since the plasma profile changes significantly over time in this simulation, the number of elements fluctuates as shown in figure 5.23b to compensate for these changes.

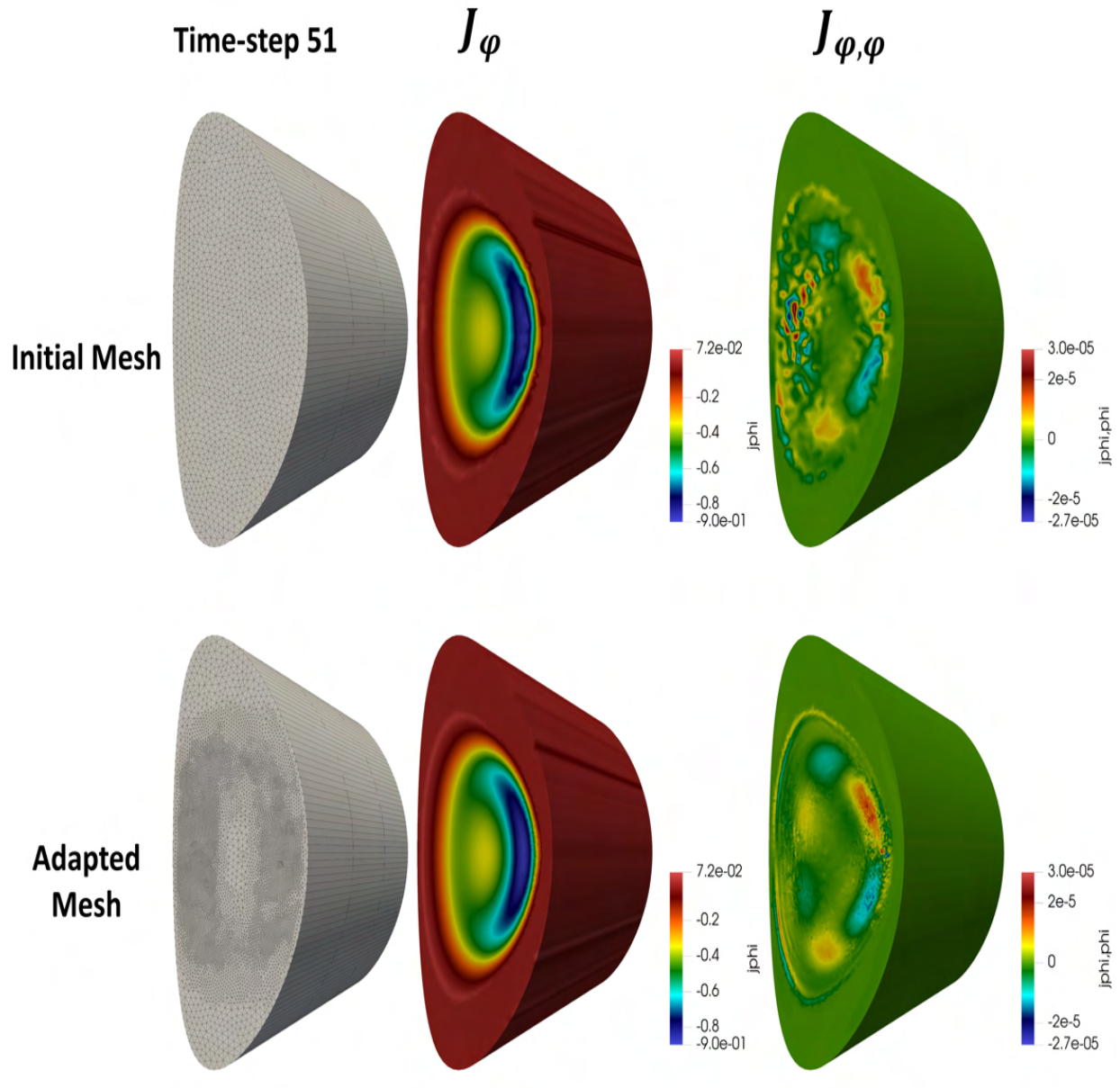

Figure 5.18: The initial and adapted meshes and solutions at time-step 51 for the pellet case. Note that the mesh was adapted using the solution field from time-step 50.

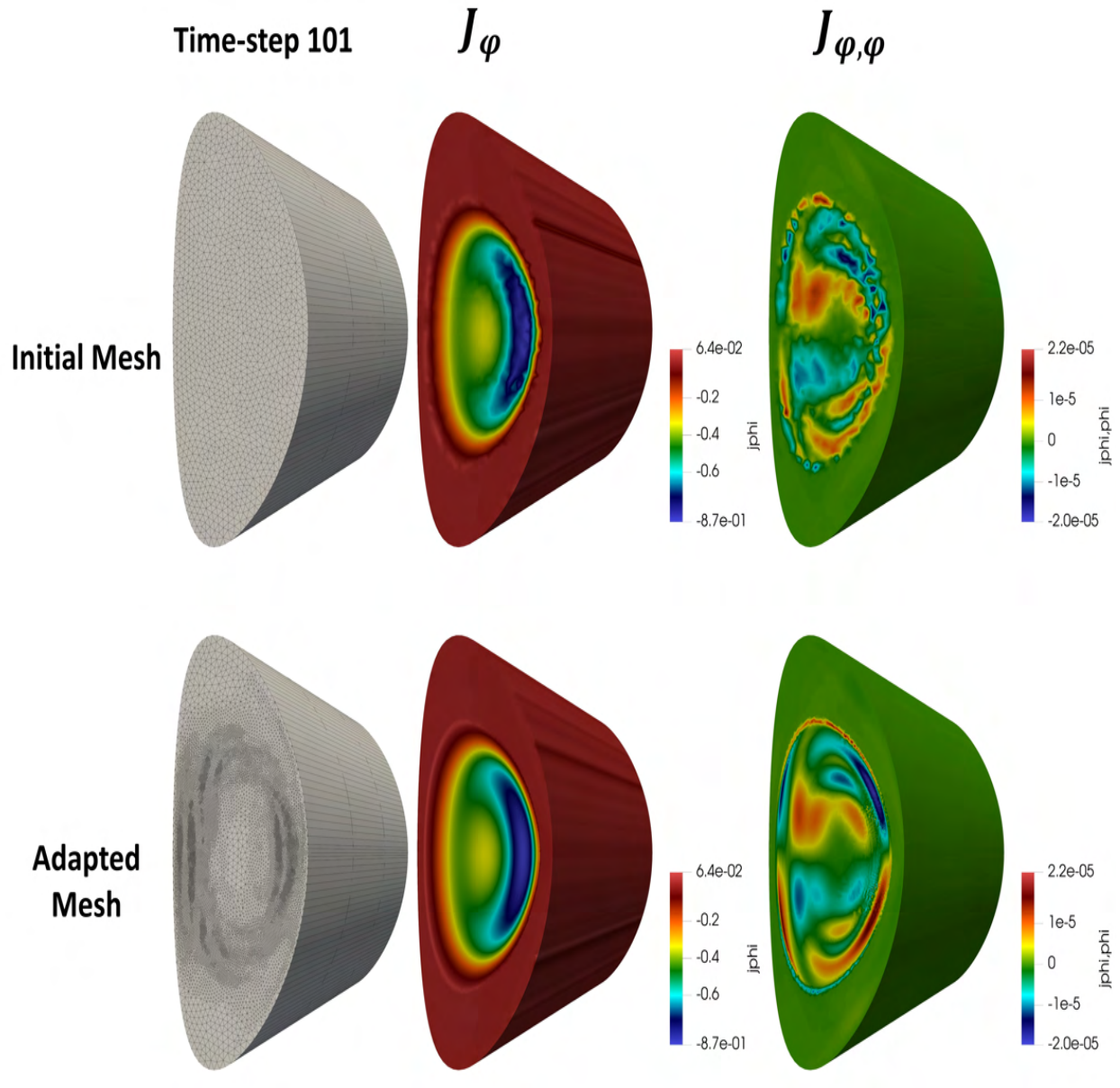

Figure 5.19: The initial and adapted meshes and solutions at time-step 101 for the pellet case. Note that the mesh was adapted using the solution field from time-step 100.

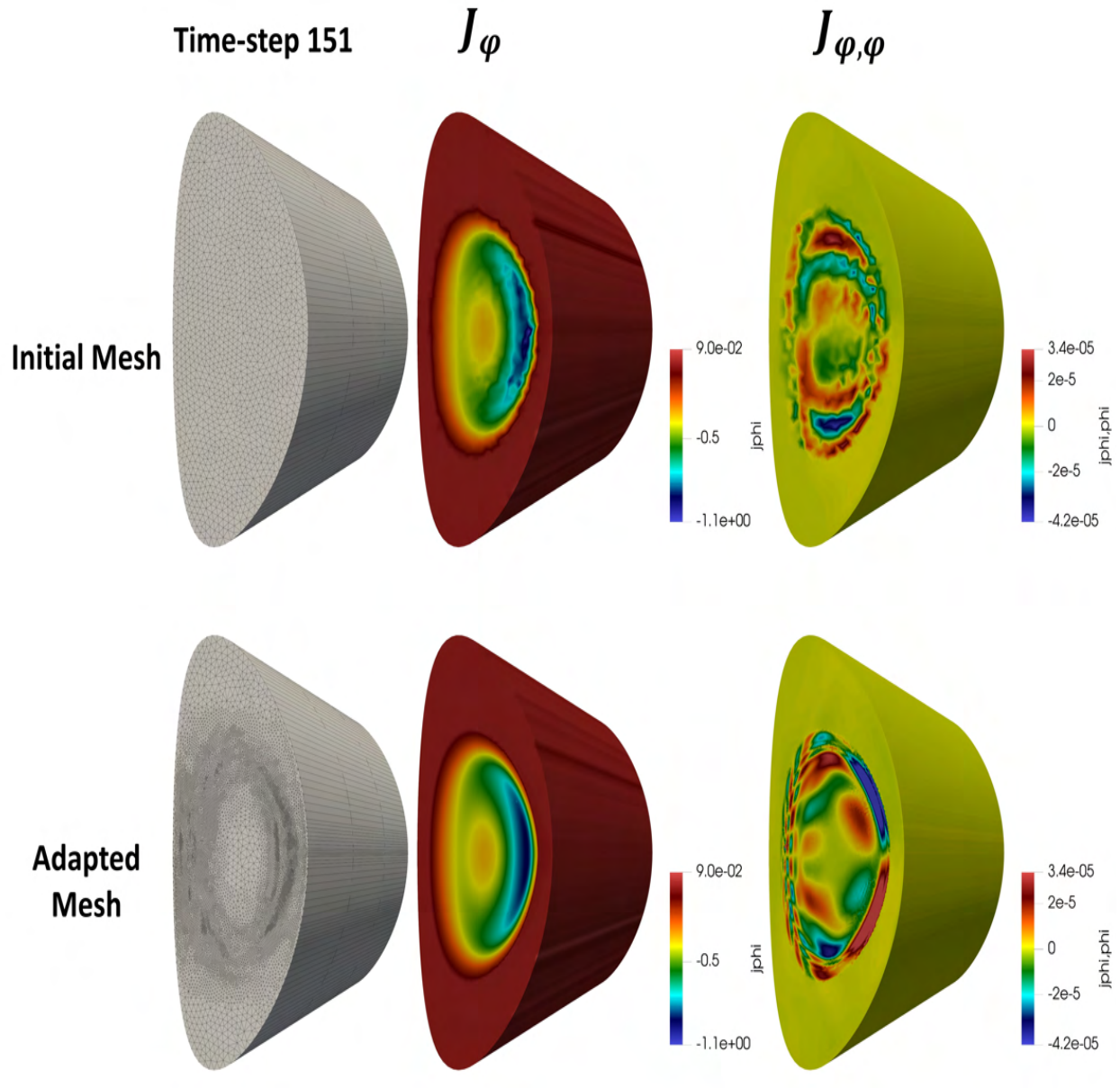

Figure 5.20: The initial and adapted meshes and solutions at time-step 151 for the pellet case. Note that the mesh was adapted using the solution field from time-step 150.

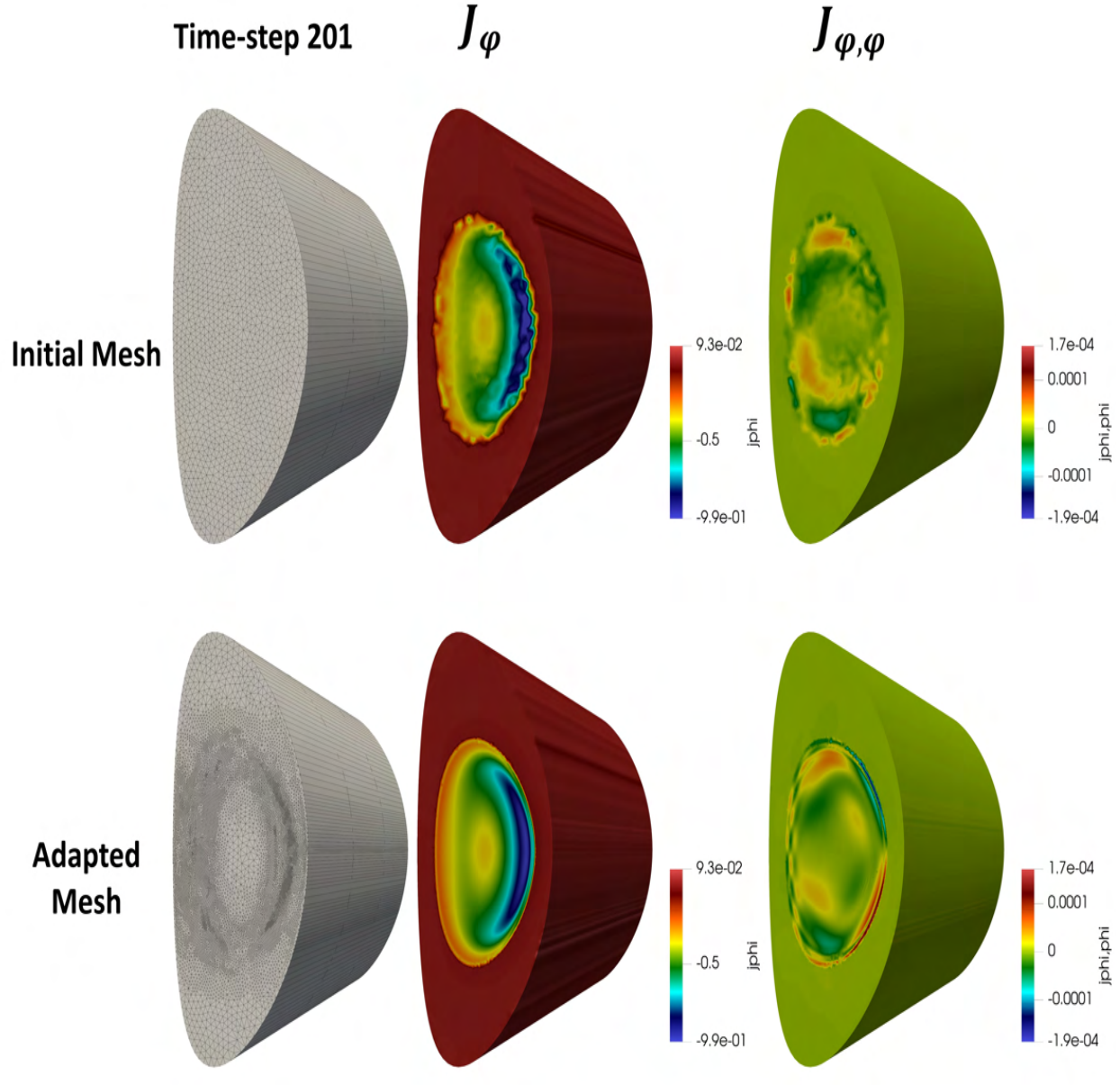

Figure 5.21: The initial and adapted meshes and solutions at time-step 201 for the pellet case. Note that the mesh was adapted using the solution field from time-step 200.

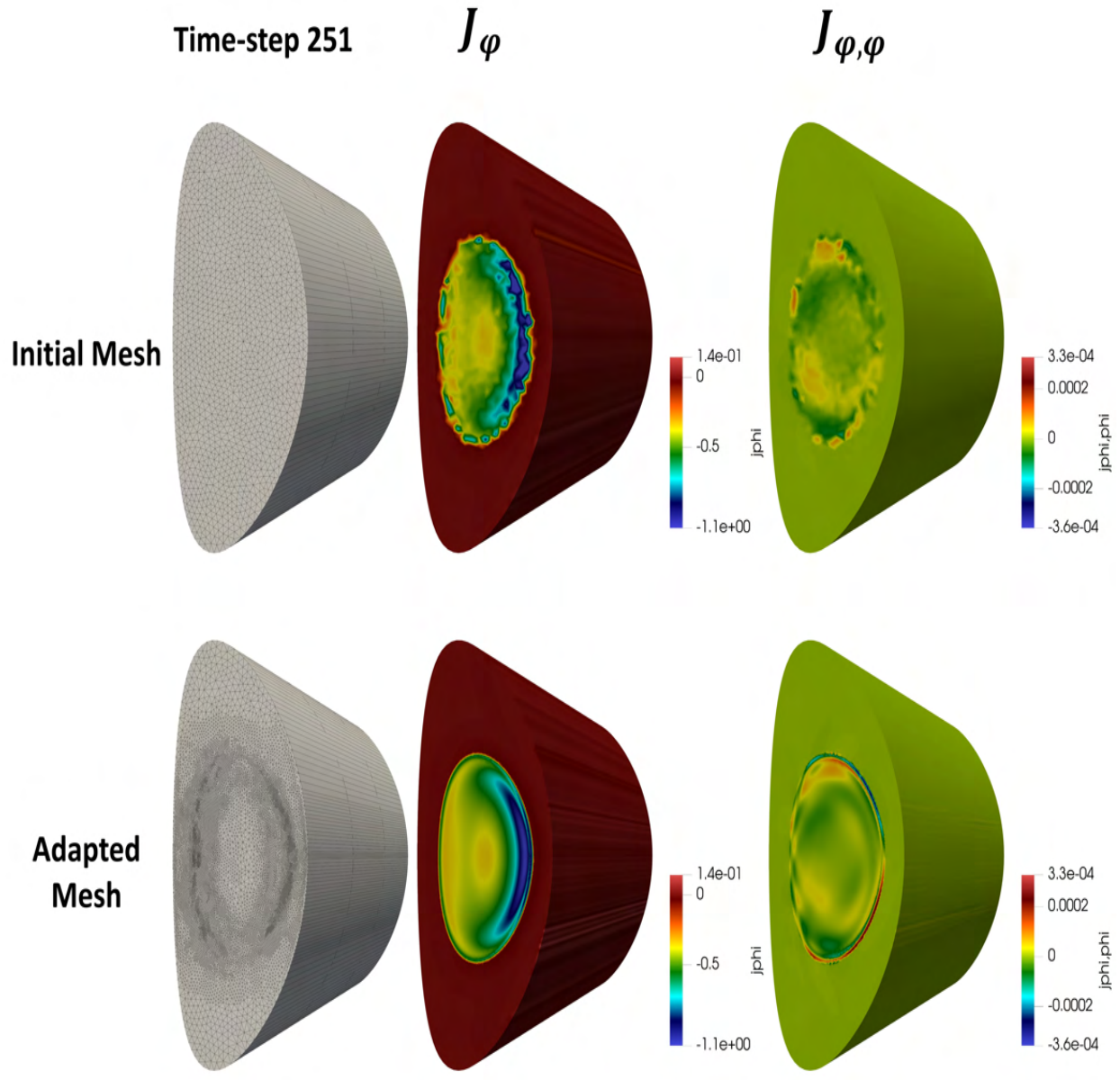

Figure 5.22: The initial and adapted meshes and solutions at time-step 251 for the pellet case. Note that the mesh was adapted using the solution field from time-step 250.

The smallest edge length L_s in the adapted mesh during the pellet case simulation is 0.005713 meters. A uniform mesh with a mesh size equal to L_s was created, which resulted in 1.2 million elements. For a similar accuracy in results with a uniform mesh as was shown with the adapted mesh, the pellet case requires roughly 12 times more elements and computing time than the adapted mesh. A comparison of the number of elements is demonstrated in figure 5.24.

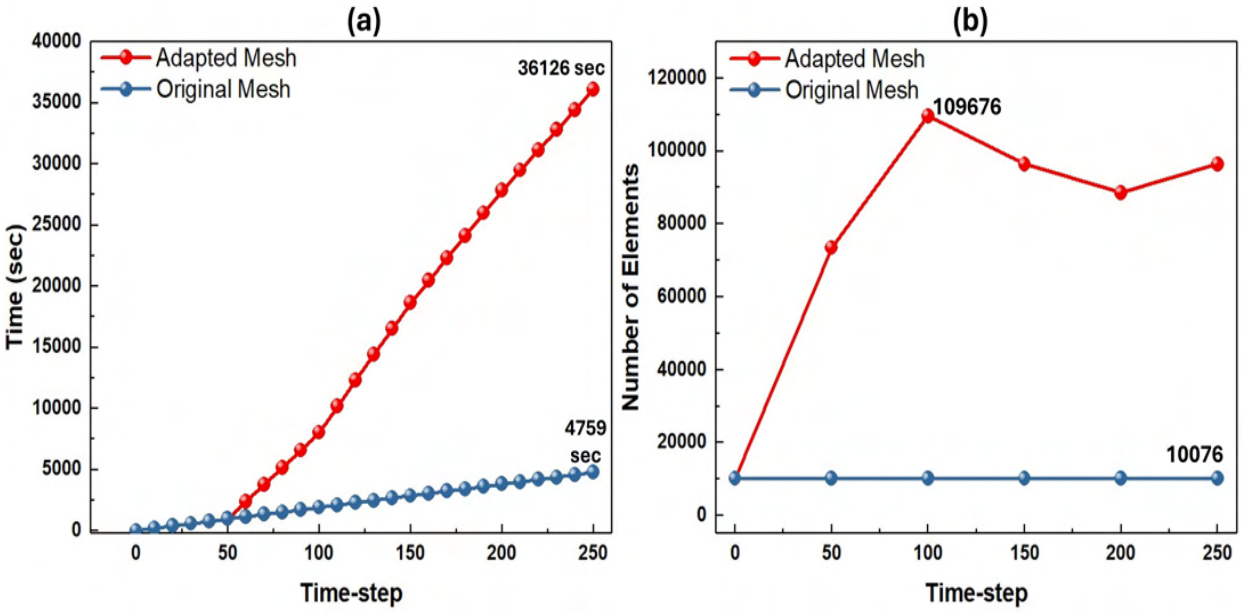

Figure 5.23: (a) Comparison of simulation time on original and adapted meshes for pellet case, (b) comparison of number of elements on original and adapted meshes. The number of elements on original mesh stays constant throughout the simulation.

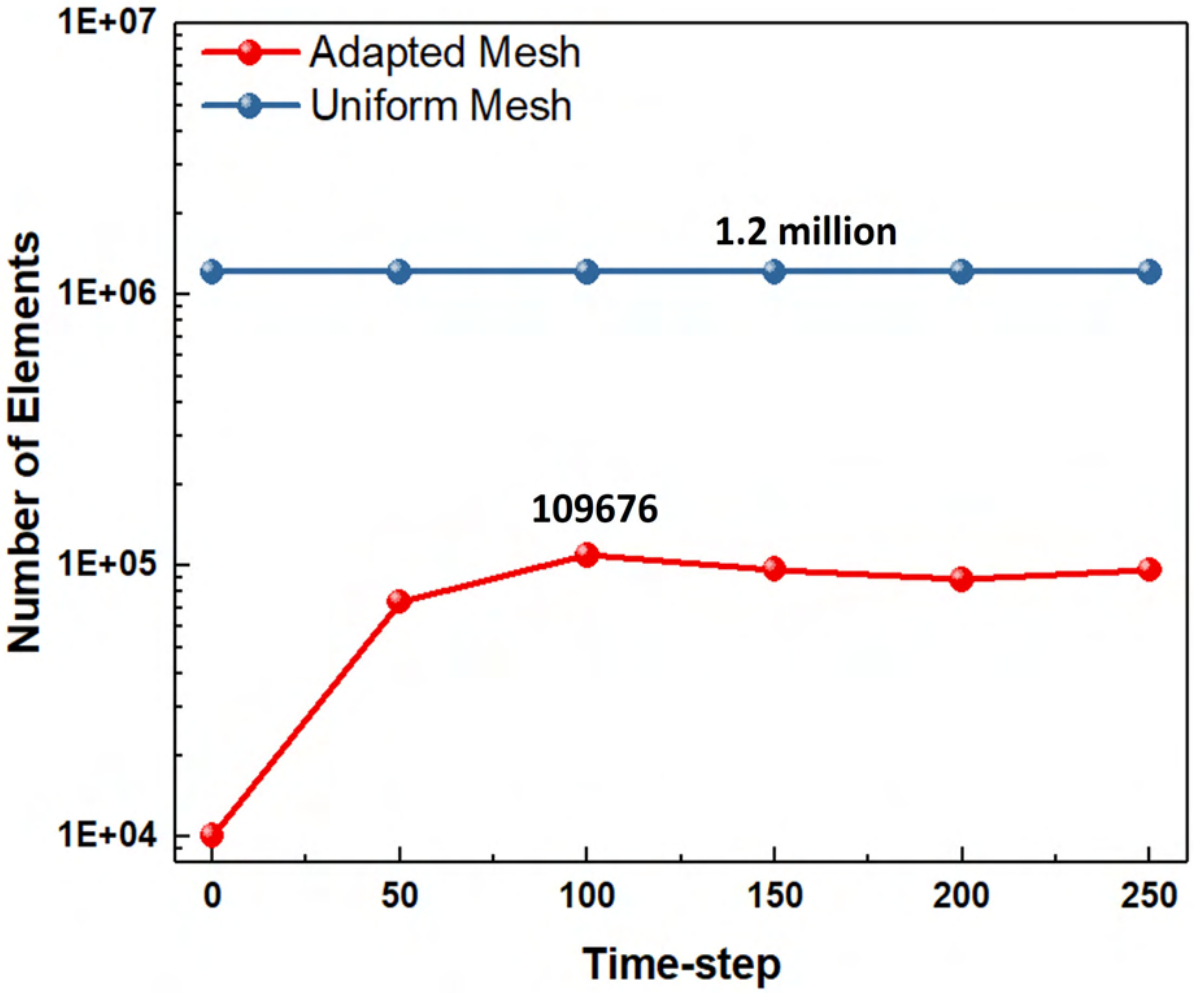

Figure 5.24: Comparison of number of elements on adapted mesh and uniform mesh with a size equal to smallest edge length in adapted mesh for the pellet case.

CHAPTER 6
CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

This thesis presented tools to meet the mesh generation and mesh adaptation needs of selected fusion plasma simulation codes. The first tool is the TOMMS mesh generator to support complex geometries and generate well-controlled meshes for production scale XGC simulations. The second tool supports the generation of M3D-C^1 meshes for more general configurations. The third tool is a framework to support M3D-C^1 adaptive simulations.

In chapter 2, a brief introduction to the fusion edge plasma simulations, the governing equations in XGC and the coordinate systems used in this work is presented. This was followed with a review of the specific meshing requirements of XGC that includes field-aligned mesh vertices on the flux curves and maintaining one-element deep meshing between flux curves. A discussion of the magnetic field aligned meshing in XGC is also provided in this chapter.

In chapter 3, an overview of the TOMMS workflow and recent developments to support field-aligned meshing of more-complex tokamak geometries are presented. In this work, the methods to find the set of critical points (X-points, and O-points) are extended to effectively search any combinations of X-points and O-points for any configurations of divertors. This enabled the generation of meshes for new configurations of divertors in tokamaks such as SPARC. The development of new methods to model the tokamak wall boundary improved the quality of meshes on the tokamak boundaries. In production meshes for XGC simulations, where mesh vertices on the flux curves are tightly spaced, the one-element deep mesh requirement between the flux curves results in unsatisfactory elements near the X-point and in the areas where open flux curves approaches tokamak wall. The mesh generation procedures are updated to support a transition of one-element deep meshes to well-defined unstructured meshes in such areas between the flux curves. This is done by decomposing the model faces that interact with the tokamak wall, in such a way that the specific meshing procedures can be applied in each sub-region. The meshes as a result of this development showed significant improvements with no unsatisfactory elements. This is confirmed from the study of the ratios of the area of the largest element on a model face between flux curves to the area of an element with minimum area of all neighboring elements. The peak ratio on

an NSTX production scale mesh decreased from 381 to 2 when transition layers procedures were applied. The XGC simulations results presented in this work also showed the improvements in the quality of simulations as a result of the new meshing procedures. The XGC operations requires the structuring of the mesh data in a way that is flux curve driven. The ordering of the mesh vertices starts with the magnetic axis, and then it spirals outward with the increasing values of ψ of flux curves. The methods to order the mesh data in a way that is most suitable to the XGC operations and present mesh data in a flux curve based data structure were developed in this work.

Chapter 4 summarises the recent developments in model and mesh generation in M3D-C^1. The original mesh generation procedures were limited to a standard three-region configurations. In this work, an automated and generalized modeling and meshing infrastructure is developed to support the meshing of the tokamak geometries with any combination of physical loops and model faces. As part of this work, a new curve offset algorithm is developed that can model tokamak finite thickness walls of any complexity. The improved mesh controls on the model entities resulted in well defined tokamak meshes for M3D-C^1 simulations. The development of this generalized framework with superior mesh control extended the application of M3D-C^1 to a broader range of tokamak geometries with any number of geometric loops and physics regions. A set of a-priori mesh modifications options is developed in this work. This includes the mesh modification based on a size-field from the magnetic flux field (ψ). This modification operation refines the mesh in the areas where plasma changes are maximum, thus improving the quality of simulations in such areas. In some cases, mesh modification is required only on certain physics regions (mostly plasma core region) based on the a-priori knowledge of problem. The mesh modification options in M3D-C^1 allow the adaptation of mesh in the desired regions with smooth transition of mesh to the neighboring region. This allows more control on the resolution of meshes by modifying only the regions that are desirable.

In chapter 5, an a-posteriori error-based mesh adaptation procedure based on SPR error estimation is given. A procedure to evaluate a desired mesh size field from the a-posteriori error estimation is developed for 2D and 3D simulations. In 2D simulations, a mesh size field from the estimated error dictates the mesh adaptation on a poloidal plane. In 3D simulations, the target size field is different on all the poloidal planes. To handle this, a 2.5D mesh adaptation framework is developed, where master poloidal plane is adapted

by using the smallest size field on a node from all the poloidal planes. The 3D mesh is reconstructed by extrusion of adapted master poloidal plane plane. The 2D and 3D M3D-C^1 simulations demonstrating the improvements in the simulations as a result of error-based mesh adaptation are presented. The results showed significant improvements in the quality of plasma profiles as a result of mesh adaptation.

6.2 Future Work

6.2.1 TOMMS

1. As demonstrated in chapter 3, a separatrix curve is generated by pushing out four vectors out of the X-points, such that these vectors are at a right angle to each other and have a length equal to desired mesh vertices spacing on the separatrix curve. As can be seen in the figure 6.1a, the end points of these pushed vectors marked as point 1,2,3, and 4 are not exactly on the separatrix curve and have different poloidal flux field values marked as ψ_1, ψ_2,ψ_3, and ψ_4 as compared to ψ_{sep} on separatrix. These points are projected to the separatrix curves by using an iterative method that tries to find a point on the desired flux curve (ψ_{sep}). While this method works for most of the cases, it has the following two limitations:

   (a) If the two legs of the separatrix curve are at a roughly equal distance from a pushed point as shown in figure 6.1, the point might get projected to the wrong leg of the separatrix. This might result in two points on the same leg and zero point on one of the other legs failing the construction of the valid separatrix curve.

   (b) This method is limited to four legged X-points and will not work for an arbitrary number of separatrix legs. In some special cases such as snow-flake divertors, the number of legs coming out of an X-point is six. In order to handle more complicated tokamak configurations, a method is required that is not limited to a predefined number of legs but can handle any arbitrary number of separatrix legs.

   In order to overcome the above mentioned limitations, we propose a method that will directly find the points on the separatrix curves instead of points being projected to separatrix curve and also not restricted to any predefined number of legs. In this method, we traverse a circle with a radius equal to the desired node spacing on the

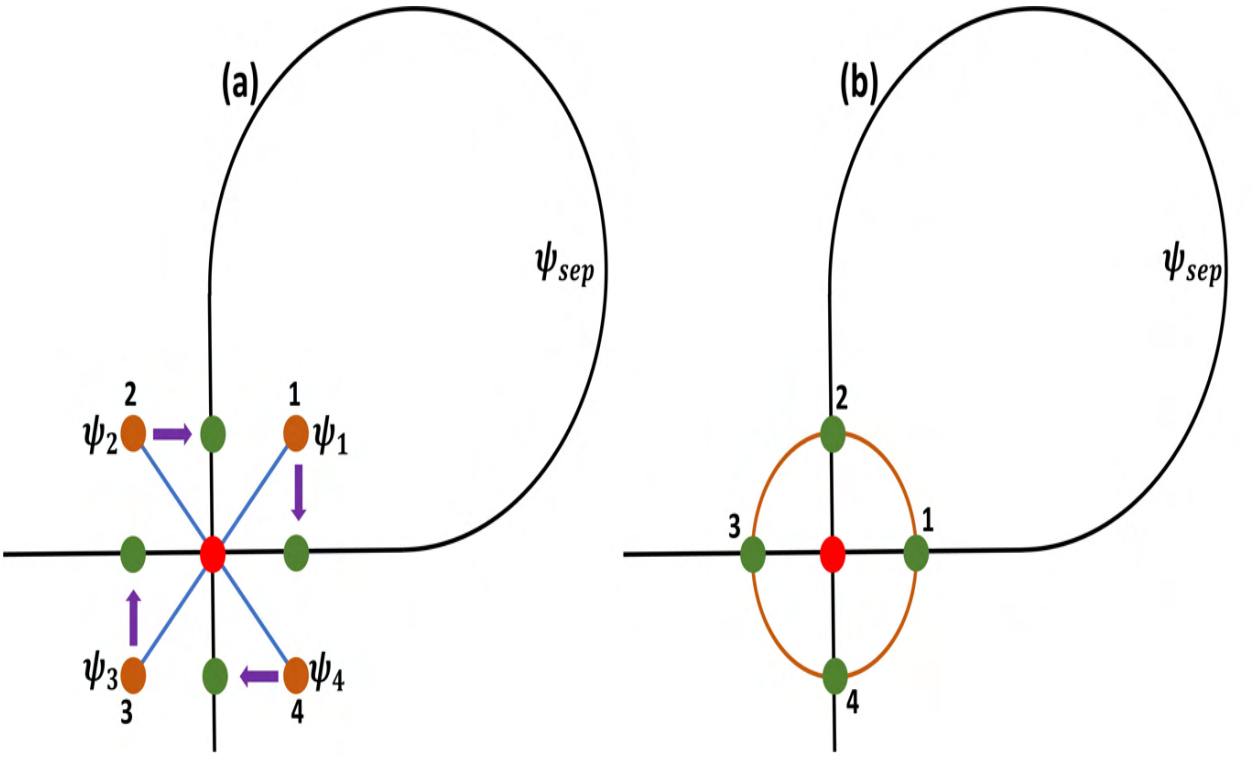

Figure 6.1: The emanation of points from the X-point in (a) current method, (b) proposed method.

separatrix, and evaluate all the points with ψ_{sep} on this circle. A fine step size along the circle ensures that all the points with ψ_{sep} are detected correctly. This idea is demonstrated in figure 6.1b.

2. The XGC meshes generated by TOMMS [8] are approximately field-following in nature. To understand this, we define the safety factor q as shown in equation below.

$$q = \frac{m}{n} \tag{6.1}$$

where m is the number of toroidal circuits and n is the number of poloidal circuits. The safety factor says that for every poloidal circuit, the q number of toroidal circuits are required to get a field line back to the point where it started. The field line error means that the magnetic field line doesn't hit on the starting point of the field line on the primary poloidal plane after q number of toroidal circuits (for every poloidal circuit).

The effects of field line error (misalignment of field line as described above) on the

simulation results can be reduced/eliminated by using a set of strategies including:

(a) Slightly adjust the ψ value of each flux curve such that the safety factor q is a rational number (the number of toroidal circuits m and poloidal circuit n are integers). If q is adjusted such that it is a rational number, there will be no field-alignment error.

(b) Use flux surfaces as given in the input but distribute the alignment error over the whole flux surface such that the magnetic field line is slightly off at each vertex. This will not let all the error concentrated at one vertex.

3. The current version of TOMMS can generate models and meshes of most frequently used tokamak configurations. Lately, plasma scientists have been exploring new configurations of the divertors, few of which are extremely complex. The current methods of critical points search and flux curves tracing can handle such cases effectively. The next step is to generate the model entities (model vertices, edges, and faces) from given physics entities. The model generation capabilities in its current state can only construct models for predefined divertors configurations. The next step is to develop a generalized automated framework for the generation of geometric models with an arbitrary number and configuration of X-points.

4. Extending TOMMS to generate field-aligned poloidal plane meshes for stellarators. As opposed to a single poloidal plane mesh for tokamaks, in stellarator version, there will be multiple poloidal plane meshes for a single case.

6.2.2 M3D-C^1

The frequency of mesh adaptation can play an important role in the accuracy of solution fields on the adapted meshes. A mesh adapted after very few time steps increase the computational costs which also depends on the size of the solution data to be transferred; however, mesh adaptation after a large number of time steps can lead to fairly inaccurate results as can be seen at time-step = 300 and time-step = 500 in figure 6.2. On the other hand, at time-step = 400 in figure 6.2, the results show minimal difference between the current density profiles on the meshes adapted after 10 and 50 time steps. This shows that a mesh needs to be adapted more frequently at time-steps where plasma profile shifts rapidly.

However in the same simulation, there are certain patches of the time where mesh adaptation is not required frequently.

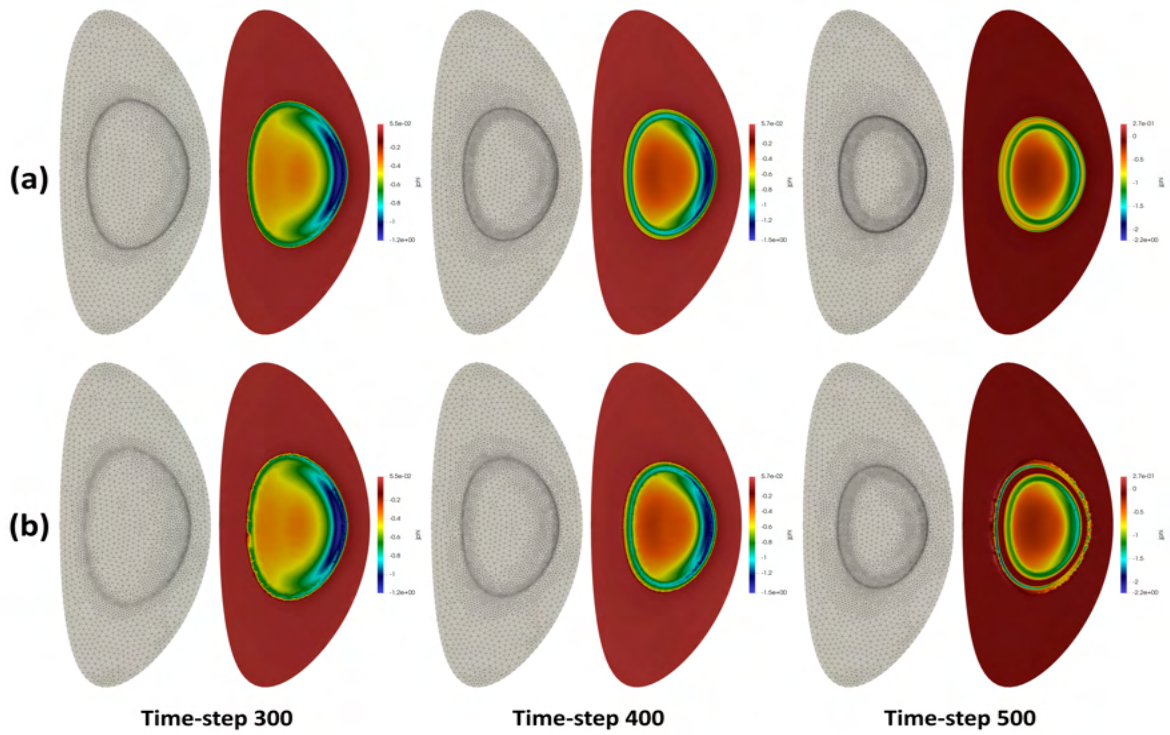

Figure 6.2: The adapted meshes and corresponding current density solution at time-step 300, 400, and 500. Mesh adapted every (a) ten time-steps, and (b) fifty time-steps.

To account for this, future work should develop a robust mesh adaptation method that can dynamically adjust the frequency of the mesh adaptation during simulation.

REFERENCES

[1] Lawrence Livermore National Laboratory, "National ignition facility achieves fusion ignition," Accessed: Apr. 12 2023. [Online]. Available: https://www.llnl.gov/news/national-ignition-facility-achieves-fusion-ignition.

[2] V. D. Shafranov, B. D. Bondarenko, G. A. Goncharov, O. A. Lavrent'ev, and A. D. Sakharov, "On the history of the research into controlled thermonuclear fusion," *Phys. Usp.*, vol. 44, pp. 835–843, Aug. 2001.

[3] ITER, "What is iter?" Accessed: Apr. 12 2023. [Online]. Available: https://www.iter.org/proj/inafewlines.

[4] Fusion Industry Association, "Fusion industry investment passes $6bn," Accessed: Aug. 20 2023. [Online]. Available: https://www.fusionindustryassociation.org/fusion-industry-investment-passes-6bn.

[5] 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 no. 115021.

[6] S. C. Jardin *et al.*, "The M3D-C1 approach to simulating 3d 2-fluid magnetohydrodynamics in magnetic fusion experiments," *J. Phys. Conf. Ser.*, vol. 125, Jul. 2008, Art no. 012044.

[7] Y. Nishimura and Z. Lin, "A finite element mesh in a tokamak edge geometry," *Contrib. Plasma Phys.*, vol. 46, pp. 551–556, Aug. 2006.

[8] U. Riaz, E. Seegyoung Seol, R. Hager, and M. S. Shephard, "Modeling and meshing for tokamak edge plasma simulations," *Comput. Phys. Commun.*, vol. 295, Feb. 2024, Art no. 108982.

[9] F. Zhang *et al.*, "Mesh generation for confined fusion plasma simulation," *Eng. Comput.*, vol. 32, pp. 285–293, Sep. 2016.

[10] R. W. Fundamenski, "Tokamak edge plasma modeling using an improved onion-skin method," Ph.D. Thesis, Inst. Aerosp. Stud. ,University of Toronto, Canada, 1999.

[11] S. Krasheninnikov, A. Smolyakov, and A. Kukushkin, "Edge plasma issues in magnetic fusion devices," in *On the Edge of Magnetic Fusion Devices*, Cham, Switzerland: Springer Int., 2020, ch . 1, pp. 1-12.

[12] L. J. Spitzer, "The stellarator concept," *Phys. Fluids.*, vol. 1, pp. 253–264, Jul. 1958.

[13] D. K. Morozov and Y. I. Pozdnyakov, "Edge plasma cooling during noble gas injection into tokamaks," *Plasma Phys. Contr. Fusion*, vol. 49, May 2007, Art no. 929.

[14] S. Wiesen *et al.*, "The new solps-iter code package," *J. Nucl. Mater.*, vol. 463, pp. 480–484, Aug. 2015.

[15] M. S. d'Abusco *et al.*, "Core-edge 2D fluid modeling of full tokamak discharge with varying magnetic equilibrium: from west start-up to ramp-down," *Nucl. Fusion*, vol. 62, May 2022, Art no. 086002.

[16] A. Y. Pigarov *et al.*, "DIII-d edge plasma simulations with uedge code including non-diffusive anomalous cross-field transport," *J. Nucl. Mater.*, vol. 313-316, pp. 1076–1080, Mar. 2003.

[17] C. Norscini *et al.*, "Turbulent transport close to marginal instability: role of the source driving the system out of equilibrium," *J. Phys. Conf. Ser.*, vol. 561, Nov. 2014, Art no. 012013.

[18] P. Tamain *et al.*, "TOKAM-3d: A 3d fluid code for transport and turbulence in the edge plasma of tokamaks," *J. Comput. Phys.*, vol. 229, pp. 361–378, Jan. 2010.

[19] B. D. Dudson and J. Leddy, "Hermes: global plasma edge fluid turbulence simulations," *Plasma Phys. Contr. Fusion*, vol. 59, Apr. 2017, Art no. 054010.

[20] A. Stegmeir, D. Coster, A. Ross, O. Maj, K. Lackner, and E. Poli, "Grillix: a 3d turbulence code based on the flux-coordinate independent approach," *Plasma Phys. Contr. Fusion*, vol. 60, Jan. 2018, Art no. 035005.

[21] S. Ku *et al.*, "A fast low-to-high confinement mode bifurcation dynamics in the boundary-plasma gyrokinetic code XGC1," *Phys. Plasmas*, vol. 25, Apr. 2018, Art no. 056107.

[22] E. D'Azevedo *et al.*, "The fusion code XGC: Enabling kinetic study of multi-scale edge turbulent transport in iter," in *Exascale Scientific Applications, Scalability and Performance Portability*, New York, NY, USA: Chapman Hall, 2017, ch. 24, pp. 529-552.

[23] M. F. Adams, S.-H. Ku, P. Worley, E. D'Azevedo, J. C. Cummings, and C.-S. Chang, "Scaling to 150k cores: Recent algorithm and performance engineering developments enabling XGC1 to run at scale," *J. Phys. Conf. Ser.*, vol. 180, Jul. 2009, Art no. 012036.

[24] R. Hager, S. Ku, A. Y. Sharma, C. S. Chang, R. M. Churchill, and A. Scheinberg, "Electromagnetic total-f algorithm for gyrokinetic particle-in-cell simulations of boundary plasma in xgc," *Phys. Plasmas*, vol. 29, Nov. 2022, Art no. 112308.

[25] C. Zhang, G. Diamond, C. W. Smith, and M. S. Shephard, "Development of an unstructured mesh gyrokinetic particle-in-cell code for exascale fusion plasma simulations on gpus," *Comput. Phys. Commun.*, vol. 291, Oct. 2023, Art no. 108824.

[26] T. Görler *et al.*, "The global version of the gyrokinetic turbulence code gene," *J. Comput. Phys.*, vol. 230, pp. 7053–7071, Aug. 2011.

[27] S. Jolliet *et al.*, "A global collisionless pic code in magnetic coordinates," *Comput. Phys. Commun.*, vol. 177, pp. 409–425, Sep. 2007.

[28] R. Hazeltine and J. Meiss, "Confined plasma equilibrium," in *Plasma Confinement*, Mineola, NY, USA: Dover, 2003, ch. 3, pp. 49-107.

[29] T. Moritaka *et al.*, "Development of a gyrokinetic particle-in-cell code for whole-volume modeling of stellarators," *Plasma*, vol. 2, pp. 179–200, May 2019.

[30] F. Hariri and M. Ottaviani, "A flux-coordinate independent field-aligned approach to plasma turbulence simulations," *Comput. Phys. Commun.*, vol. 184, pp. 2419–2429, Nov. 2013.

[31] D. Michels, A. Stegmeir, P. Ulbl, D. Jarema, and F. Jenko, "Gene-x: A full-f gyrokinetic turbulence code based on the flux-coordinate independent approach," *Comput. Phys. Commun.*, vol. 264, Jul. 2021, Art no. 107986.

[32] R. Hager and C. S. Chang, "Gyrokinetic neoclassical study of the bootstrap current in the tokamak edge pedestal with fully non-linear Coulomb collisions," *Phys. Plasmas*, vol. 23, Apr. 2016, Art no. 042503.

[33] R. Hager, J. Dominski, and C. S. Chang, "Cross-verification of neoclassical transport solutions from XGCa against NEO," *Phys. Plasmas*, vol. 26, no. 10, Oct. 2019, Art no. 104502.

[34] J. Dominski, C. S. Chang, R. Hager, P. Helander, S. Ku, and E. S. Yoon, "Study of up–down poloidal density asymmetry of high-z impurities with the new impurity version of xgca," *J. Plasma Phys.*, vol. 85, Oct. 2019, Art no. 905850510.

[35] N. M. Ferraro, S. C. Jardin, L. L. Lao, M. S. Shephard, and F. Zhang, "Multi-region approach to free-boundary three-dimensional tokamak equilibria and resistive wall instabilities," *Phys. Plasmas*, vol. 23, May 2016, Art no. 056114.

[36] G. Giorgiani *et al.*, "A hybrid discontinuous galerkin method for tokamak edge plasma simulations in global realistic geometry," *J. Comput. Phys.*, vol. 374, pp. 515–532, Dec. 2018.

[37] S. Shiraiwa, J. C. Wright, P. T. Bonoli, T. Kolev, M. Stowell, and J. Hillairet, "RF wave simulation for cold edge plasmas using the MFEM library," in *22 Topical Conf. RF Power Plasmas*, Oct. 2017, Art no. 03048.

[38] D. N. Dhyanjyoti *et al.*, "A 3D unstructured mesh based particle tracking code for impurity transport simulation in fusion tokamaks," *Comput. Phys. Commun.*, vol. 292, Nov. 2023, Art no. 108861.

[39] S. Ku *et al.*, "Gyrokinetic particle simulation of neoclassical transport in the pedestal/scrape-off region of a tokamak plasma," *J. Phys. Conf. Ser.*, vol. 46, Sep. 2006, Art no. 87.

[40] Scientific Computation Research Center, "Tokamak modeling and meshing software (TOMMS) user's guide," Accessed: Jul. 12 2023. [Online]. Available: https://github.com/SCOREC/Fusion_Public/wiki/Tokamak-Modeling-and-Meshing-Software-(TOMMS)-User's-Guide.

[41] Princeton Plasma Physics Lab, "PSPLINE help," Accessed: Jun. 10 2020. [Online]. Available: https://w3.pppl.gov/ pshare/help/pspline.htm.

[42] J. Luxon, "A design retrospective of the diii-d tokamak," *Nucl. Fusion*, vol. 42, May 2002, Art no. 614.

[43] X. Jiao and M. Heath, "Feature detection for surface meshes," in *Proc. 8th Int. Conf. Numer. Grid Gener. Comput. Field Simul.*, Jan. 2002, pp. 705–714.

[44] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Comput. J.*, vol. 7, pp. 308–313, Jan. 1965.

[45] H. Press, S. TeuKolsky, W. Vetterling, and B. Flannery, "Minimization or maximization of functions," in *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge, UK: Cambridge Univ. Press, 1992, ch. 10, pp. 394-444.

[46] H. Press, S. TeuKolsky, W. Vetterling, and B. Flannery, "Root finding and nonlinear sets of equations," in *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge, U.K.: Cambridge Univ. Press, 1992, ch. 9, pp. 347-383.

[47] J. Stewart, "Partial derivatives," in *Multivariable Calculus*, Belmont, CA, USA: Cengage Learn., 2010, ch. 14, pp. 901-996.

[48] S. Parker, R. Procassini, C. Birdsall, and B. Cohen, "A suitable boundary condition for bounded plasma simulation without sheath resolution," *J. Comput. Phys.*, vol. 104, pp. 41 – 49, Jan. 1993.

[49] D. Stotler, J. Lang, C. Chang, R. Churchill, and S. Ku, "Neutral recycling effects on ITG turbulence," *Nucl. Fusion*, vol. 57, Jul. 2017, Art no. 086028.

[50] M. W. Beall and M. S. Shephard, "A general topology-based mesh data structure," *Int. J. Numer. Meth. Eng.*, vol. 40, pp. 1573–1596, Dec. 1997.

[51] M. Zhou, O. Sahni, M. Shephard, C. Carothers, and K. Jansen, "Adjacency-based data reordering algorithm for acceleration of finite element computations," *Sci. Program.*, vol. 18, pp. 107–123, Jan. 2010.

[52] F. Yang *et al.*, "A parallel interface tracking approach for evolving geometry problems," *Eng. Comput.*, vol. 38, pp. 4289–4305, Aug. 2022.

[53] S. Jardin, "A triangular finite element with first-derivative continuity applied to fusion MHD applications," *J. Comput. Phys.*, vol. 200, pp. 133–152, Oct. 2004.

[54] B. C. Lyons *et al.*, "Axisymmetric benchmarks of impurity dynamics in extended-magnetohydrodynamic simulations," *Plasma Phys. Contr. Fusion*, vol. 61, Apr. 2019, Art no. 064001.

[55] N. M. Ferraro, S. C. Jardin, and P. B. Snyder, "Ideal and resistive edge stability calculations with M3D-C1," *Phys. Plasmas*, vol. 17, no. 10, Oct. 2010, Art no. 102508.

[56] D. Pfefferlé, N. Ferraro, S. C. Jardin, I. Krebs, and A. Bhattacharjee, "Modelling of NSTX hot vertical displacement events using M3D-C1," *Phys. Plasmas*, vol. 25, no. 5, Apr. 2018, Art no. 056106.

[57] N. Ferraro, B. Lyons, C. Kim, Y. Liu, and S. Jardin, "3D two-temperature magnetohydrodynamic modeling of fast thermal quenches due to injected impurities in tokamaks," *Nucl. Fusion*, vol. 59, Nov. 2018, Art no. 016001.

[58] S. C. Jardin, N. Ferraro, and I. Krebs, "Self-organized stationary states of tokamaks," *Phys. Rev. Lett.*, vol. 115, Nov. 2015, Art no. 215001.

[59] Scientific Discoverery through Advanced Computing, "Fluid modeling of fusion plasmas with M3D-C1," Accessed: Mar. 29 2024. [Online]. Available: http://www.mcs.anl.gov/uploads/cels/papers/scidac11/final/ferraro nathaniel.pdf.

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

[61] D. A. Ibanez, E. S. Seol, C. W. Smith, and M. S. Shephard, "PUMI: Parallel unstructured mesh infrastructure," *ACM Trans. Math. Softw.*, vol. 42, pp. 1–28, May 2016.

[62] Scientific Computation Research Center, "Parallel unstructured mesh infrastructure," Accessed: Mar. 28 2024. [Online]. Available: https://www.scorec.rpi.edu/pumi.

[63] Simmetrix, "Mesh generation," Accessed: Feb. 20 2024. [Online]. Available: http://www.simmetrix.com/index.php/technologies/mesh-generation.

[64] F. Zhang, "Parallel adaptive infrastructure for magnetically confined fusion plasma simulations," Ph.D. Thesis, Dept. Mech. Nucl. Aerosp., Rensselaer Polytechnic Institute, Troy, NY, USA, 2015.

[65] G. Strang and G. Fix, *An Analysis of the Finite Element Method*, New York, NY, USA: Prentice-Hall Inc., 1973.

[66] PETSc, "PETSc overview," Accessed: Mar. 20 2023. [Online]. Available: https:// petsc.org/release/overview.

[67] Princeton Plasma Physics Lab, "The M3D-C1 users guide," Accessed: Mar. 28 2024. [Online]. Available: https://m3dc1.pppl.gov/NEWDOC-latest.pdf.

[68] A. J. Creely *et al.*, "Overview of the sparc tokamak," *J. Plasma Phys.*, vol. 86, Sep. 2020, Art no. 865860502.

[69] C. Lee, T. Phan, and D. Kim, "2D curve offset algorithm for pockets with islands using a vertex offset," *Int. J. Precis. Eng. Man.*, vol. 10, pp. 127–135, Jul. 2009.

[70] H. Kim, S. Lee, and M. Yang, "A new offset algorithm for closed 2d lines with islands," *Int. J. Adv. Man. Technol.*, vol. 29, pp. 1169–1177, Sep. 2005.

[71] I. Babuška and W. C. Rheinboldt, "A-posteriori error estimates for the finite element method," *Int. J. Numer. Meth. Eng.*, vol. 12, pp. 1597–1615, Jan. 1978.

[72] D. W. Kelly, J. P. De S. R. Gago, O. C. Zienkiewicz, and I. Babuska, "A posteriori error analysis and adaptive processes in the finite element method: Part 1 - error analysis," *Int. J. Numer. Meth. Eng.*, vol. 19, pp. 1593–1619, Nov. 1983.

[73] R. Bank and A. Weiser, "Some a posteriori error estimators for elliptic partial differential equations," *Math. Comput.*, vol. 44, pp. 283–301, Jun. 1999.

[74] C. Carstensen and S. A. Funken, "Fully reliable localized error control in the FEM," *SIAM J. Sci. Comput.*, vol. 21, pp. 1465–1484, Jan. 1999.

[75] M. Ainsworth and J. T. Oden, "Recovery based error estimators," in *A Posterior Error Estimation in Finite Element Analysis*, New York, NY, USA: Wiley Intersci, 2000, ch.4, pp. 65-84.

[76] O. C. Zienkiewicz and J. Z. Zhu, "The superconvergent patch recovery and a posteriori error estimates. part 1: The recovery technique," *Int. J. Numer. Meth. Eng.*, vol. 33, pp. 1331–1364, May 1992.

[77] O. C. Zienkiewicz and J. Z. Zhu, "The superconvergent patch recovery and a posteriori error estimates. part 2: Error estimates and adaptivity," *Int. J. Numer. Meth. Eng.*, vol. 33, pp. 1365–1382, May 1992.

[78] S. Gibson, "Measurement of the current profile in fusion tokamaks using the motional stark effect diagnostic," Ph.D. Thesis, Dept. Phys., Durham University, Durham, England, 2021.

[79] Z. Costanza, "Modeling and control of the current density profile in tokamaks and its relation to electron transport," Ph.D. Thesis, Fac. Basic Sci., École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, 2009.