

LA-UR-20-29880

Accepted Manuscript

A Parallel Cut-Cell Algorithm for the Free-Boundary Grad--Shafranov Problem

Liu, Shuang
Tang, Qi
Tang, Xianzhu

Provided by the author(s) and the Los Alamos National Laboratory (2022-04-21).

To be published in: SIAM Journal on Scientific Computing

DOI to publisher's version: 10.1137/20M1385470

Permalink to record:

<http://permalink.lanl.gov/object/view?what=info:lanl-repo/lareport/LA-UR-20-29880>



Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by Triad National Security, LLC for the National Nuclear Security Administration of U.S. Department of Energy under contract 89233218CNA000001. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

A PARALLEL CUT-CELL ALGORITHM FOR THE FREE-BOUNDARY GRAD-SHAFRANOV PROBLEM*

SHUANG LIU[†], QI TANG[‡], AND XIAN-ZHU TANG[§]

Abstract. A parallel cut-cell algorithm is described to solve the free-boundary problem of the Grad-Shafranov equation. The algorithm reformulates the free-boundary problem in an irregular bounded domain and its important aspects include a searching algorithm for the magnetic axis and separatrix, a surface integral along the irregular boundary to determine the boundary values, an approach to optimize the coil current based on a targeting plasma shape, Picard iterations with Aitken's acceleration for the resulting nonlinear problem, and a Cartesian grid embedded boundary method to handle the complex geometry. The algorithm is implemented in parallel using a standard domain-decomposition approach and a good parallel scaling is observed. Numerical results verify the accuracy and efficiency of the free-boundary Grad-Shafranov solver.

Key words. Cut-cell, finite difference, free-boundary problem, Grad-Shafranov equation

AMS subject classifications. 65N06, 65N55, 76W05

1 Introduction Tokamak fusion relies on magnetic confinement of a plasma at a temperature of around 10 keV and a particle density of $10^{20}/\text{m}^3$. The force balance is achieved by running a current inside the plasma that produces a Lorentz force to counter the plasma pressure gradient. In an axisymmetric configuration like a Tokamak, such an equilibrium is described by an elliptic equation for the poloidal magnetic flux, commonly known as the Grad-Shafranov equation. In addition to the plasma current, electrical currents are also carried in toroidal and poloidal magnetic field coils outside the plasma chamber to produce the confining magnetic field. The coil current can be individually adjusted to accommodate a variety of plasma pressure and current profiles in terms of plasma positioning and shaping.

In a fixed-boundary Grad-Shafranov solver, both the location of the computational boundary and the boundary condition for the poloidal magnetic flux are known. Many numerical approaches, including spectral elements [36, 24], hybridizable discontinuous Galerkin methods [45, 46], boundary integral approaches [42, 33], Hermite finite element [40, 25], and discontinuous Petrov Galerkin methods [43], have been extended to solve the fixed-boundary problem. A common use of the fixed-boundary Grad-Shafranov solver is to set the computational boundary to be the targeted last closed flux surface, so the plasma shaping is enforced by the computational boundary on which the poloidal flux is a constant. The free-boundary Grad-Shafranov solver aims to account for the coil currents as well as the plasma current on plasma positioning and shaping. It is an essential tool for tokamak machine design and specific experimental discharge planning. Most tokamak facilities around the world have in-house development and/or maintenance of free-boundary Grad-Shafranov solvers. There have also been more recent interest that have a focus on the numerical aspects of the free-boundary problem [23, 16, 22].

Computationally the free-boundary problem for the Grad-Shafranov equation is more challenging as the plasma shape is not known *a priori*. The general solution strategy involves an iterative approach that combines a fixed-boundary solver with a Green's function representation for the contribution to the poloidal magnetic flux on the computational boundary from the coil current external to the computational boundary. The solution process is iterative by nature and Picard iteration is usually deployed to drive the convergence of the poloidal magnetic flux on the computational boundary. The choice of this fiducial computational boundary is constrained by two considerations: (1) the computational boundary shall enclose all chamber volume where current-carrying plasma can access; and (2) all external currents should be outside the computational boundary. Figure 1 shows the plasma region (colored by red) and the external currents (coils and solenoids). The most natural place for such a fiducial computational boundary in a free-boundary Grad-Shafranov solver

*Submitted to the editors January 31, 2022.

Funding: This work was supported by the U.S. Department of Energy through the Fusion Theory Program of the Office of Fusion Energy Sciences, and the Tokamak Disruption Simulation (TDS) SciDAC partnerships between the Office of Fusion Energy Science and the Office of Advanced Scientific Computing.

[†]Los Alamos National Laboratory, Los Alamos, NM 87545, USA (shuangliu@lanl.gov). Current address: Department of Mathematics, University of California, San Diego, La Jolla, CA 92093, USA (shl083@ucsd.edu).

[‡]Los Alamos National Laboratory, Los Alamos, NM 87545, USA (qtang@lanl.gov).

[§]Los Alamos National Laboratory, Los Alamos, NM 87545, USA (xtang@lanl.gov).

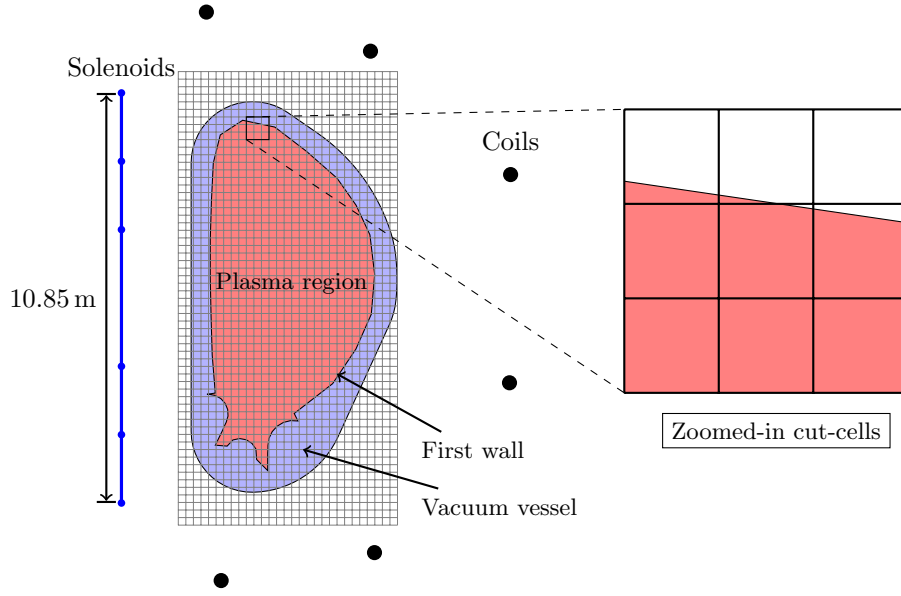


Fig. 1: ITER tokamak geometry. Left: ITER plasma region, external currents, and an exemplar cut-cell mesh. Right: zoomed-in view. Note there are two types of external currents in ITER, i.e., six coils (point source) and solenoids of five segments (line source). The geometry uses the actual dimensions from ITER.

43 is the vacuum vessel (the blue region in Figure 1) of which the poloidal field currents are all outside, or the
 44 first wall (as indicated in Figure 1). Both first wall and vacuum vessel in modern tokamaks are strongly
 45 shaped and give rise to an irregular computational domain, for which some variation of finite element/vol-
 46 ume on an unstructured grid would be a ready choice for the underlying fixed-boundary Grad-Shafranov
 47 solver. It is interesting to note that the essential ingredients of free-boundary Grad-Shafranov solvers, for
 48 example, both the Green's function formulation and iterative solution procedure, were first developed with
 49 finite difference discretization over a structured grid in a regular domain [32, 30, 27, 28]. This is partly
 50 due to the considerable freedom in the placement of the fiducial computational boundary despite the two
 51 general constraints noted earlier, and partly due to the fact that many tokamaks have enough space between
 52 current-carrying plasmas and poloidal field coils for a rectangular computational boundary in-between them.
 53 For some of the next generation tokamaks, it becomes more desirable to have the computational boundary
 54 aligned with the better flux conserving, for example the vacuum vessel in ITER [2], or with the first wall, such
 55 as the case with the SPARC tokamak [13] where the coils are very close to the plasma region. Also, note
 56 that it is desirable to represent the field on a structured grid, which can significantly accelerate the particle
 57 pusher in a particle-in-cell code. We find that for these applications, cut-cells (as indicated in Figure 1) can
 58 be used in combination with finite difference in a structured grid to accommodate an irregular computational
 59 domain for a free-boundary Grad-Shafranov solver.

60 Numerical solutions to elliptic problems with complex geometry, of which the Grad-Shafranov equilib-
 61 rium is one example, have been considered by many approaches, including finite difference, finite volume,
 62 and finite element using various meshing techniques. Among those numerical approaches, the cut-cell ap-
 63 proach has a number of advantages, which include simplifying the grid generation process for complicated
 64 geometries, enabling fast computation of the solution in parallel, and shifting the complexity of dealing with
 65 complex geometries to the discretization approach. For more details, one can consult the review in [6].
 66 The cut-cell approach, a.k.a. the Cartesian grid embedded boundary method, generates the mesh using a
 67 background regular mesh and taking special care of cut-cells where the geometry intersects the grid. A
 68 variety of work have addressed the successful use of the cut-cell approach for solving elliptic equations with
 69 finite volume [29, 48, 14, 15] and finite difference discretization [31, 19, 18, 35]. When discretized using the
 70 flux-divergence form, finite volume methods have the advantage of keeping discretely conservative [34], which
 71 is important in solving many problems such as heat and mass transfer. In this work, we employ the cut-cell

72 approach with a conservative discretization to address the Grad-Shafranov equation in irregular domains.

73 The paper is organized as follows. [Section 2](#) gives a brief description of the Grad-Shafranov equation and
 74 the associated fixed-boundary and free-boundary problems. [Section 3](#) discusses the cut-cell algorithm, which
 75 is a Cartesian grid embedded boundary method, as well as a level-set approach to improve the efficiency of
 76 the cut-cell algorithm. This is followed by a detailed description of a free-boundary Grad-Shafranov solver,
 77 focusing on solving the free-boundary problem reformulated on a bounded domain in [Section 4](#). In [Section 5](#),
 78 the parallel implementation is described and some details of the implementation of the free-boundary Grad-
 79 Shafranov solver are presented. Finally, numerical tests are presented in [Section 6](#), demonstrating the
 80 accuracy and efficiency of the free-boundary Grad-Shafranov solver.

81 **2 Grad-Shafranov equation** In this section, the Grad-Shafranov equation, fixed-boundary problem,
 82 and free-boundary problem are briefly introduced.

83 The Grad-Shafranov equation is derived from the MHD equilibrium equations given by

$$84 \quad (2.1a) \quad \nabla p = \mathbf{J} \times \mathbf{B},$$

$$85 \quad (2.1b) \quad \mu_0 \mathbf{J} = \nabla \times \mathbf{B},$$

$$86 \quad (2.1c) \quad \nabla \cdot \mathbf{B} = 0,$$

88 where \mathbf{J} is the electric current, \mathbf{B} is the magnetic field, μ_0 is the magnetic permeability, and p is the plasma
 89 pressure. [Equation \(2.1a\)](#) is the force balance equation. [Equation \(2.1b\)](#) is the Ampère's law. [Equation \(2.1c\)](#)
 90 is the divergence free constraint for the magnetic field. In the case of equilibrium, the above system [\(2.1\)](#)
 91 holds in the entire region \mathbb{R}^3 .

92 The Grad-Shafranov equation is usually written in a cylindrical coordinate system, denoted by (R, ϕ, Z) ,
 93 for an axisymmetric plasma such as those in a tokamak. It is well known that the magnetic field in the
 94 tokamak can be represented by

$$95 \quad \mathbf{B} = \nabla \phi \times \nabla \psi + g(\psi) \nabla \phi,$$

96 where ψ is commonly referred to as the poloidal flux function and g is a scalar function. The Grad-Shafranov
 97 equation can be derived from [Equation \(2.1\)](#) as

$$98 \quad (2.2) \quad \Delta^* \psi = \mu_0 R J_\phi(R, \psi),$$

99 where the toroidal elliptic operator is defined as

$$100 \quad \Delta^* \psi \equiv \frac{\partial^2 \psi}{\partial R^2} - \frac{1}{R} \frac{\partial \psi}{\partial R} + \frac{\partial^2 \psi}{\partial Z^2},$$

101 and the source term satisfies

$$102 \quad \mu_0 R J_\phi(R, \psi) = - \left[\mu_0 R^2 \frac{dp(\psi)}{d\psi} + g(\psi) \frac{dg(\psi)}{d\psi} \right],$$

103 in which the plasma pressure, p , is a scalar function of ψ . Note that the toroidal elliptic operator can be
 104 rewritten in a divergence form

$$105 \quad \Delta^* \psi = R \tilde{\nabla} \cdot \left(\frac{1}{R} \tilde{\nabla} \psi \right),$$

106 where $\tilde{\nabla} \equiv [\partial_R, \partial_Z]^T$. This conservative form will be used when discretizing the problem with the cut-cell
 107 approach.

108 Note that in a tokamak plasma, J_ϕ is carried by the magnetically confined plasma that resides inside the
 109 magnetic separatrix in the formulation of both fixed- and free-boundary Grad-Shafranov equilibria, which
 110 we shall explain next.

111 **2.1 Fixed-boundary problem** The fixed-boundary problem refers to the situation when the boundary
 112 condition for the equilibrium problem is known. Mathematically, the problem can be simply cast as a
 113 boundary value problem given by

$$114 \quad (2.3) \quad \begin{aligned} \Delta^* \psi &= \mu_0 R J_\phi(R, \psi), & (R, Z) \in \Omega. \\ \psi &= \psi_D, & (R, Z) \in \partial\Omega. \end{aligned}$$

115 where Ω is the physical domain with a Lipschitz boundary $\partial\Omega$. In tokamaks, the physical domain Ω cor-
 116 responds to the cross section of the device bounded by the first wall or more commonly, the last closed
 117 flux surface beyond which J_ϕ vanishes. In this work, the fixed-boundary problem is also implemented as an
 118 important tool to verify the numerical scheme.

119 **2.2 Free-boundary problem** The free-boundary problem refers to the situation when the plasma do-
 120 main, denoted as $\mathcal{P}(\psi)$, is unknown. In tokamak, the plasma domain refers to the region inside the device
 121 where the confinement actually happens, and thus $\mathcal{P}(\psi)$ is filled with hot plasmas. In a diverted tokamak
 122 plasma, $\mathcal{P}(\psi)$ is the magnetic separatrix that demarcates the region of nested closed flux surfaces and the
 123 scrape-off layer that has magnetic field lines intercept the divertor and first wall. For a limited plasma,
 124 $\partial\mathcal{P}(\psi)$ is the last closed flux surface beyond which the flux surfaces are intercepted by one or multiple lim-
 125 iters that protrude from the chamber wall. It is important to note that $\mathcal{P}(\psi)$ is only meaningful when such
 126 an MHD equilibrium exists. Such a constraint poses an extra challenge to numerical algorithms, since the
 127 success of finding a numerical solution to the free-boundary problem implicitly indicates the existence of
 128 such a domain, while a perturbed numerical solution may indicate an unrealistic plasma domain, resulting
 129 into the divergence of algorithms. Therefore, it is necessary to design the numerical algorithm to be robust
 130 with respect to such a perturbation.

131 Assuming an unbounded domain $\mathcal{H} := [0, \infty) \times (-\infty, \infty)$, the free boundary problem is given by

$$132 \quad (2.4) \quad \Delta^* \psi = \begin{cases} \mu_0 R J_\phi(R, \psi), & (R, Z) \in \mathcal{P}(\psi), \\ \mu_0 R I_i, & (R, Z) = (R_i^c, Z_i^c), i = 1, 2, \dots, n_c. \\ 0, & \text{otherwise.} \end{cases}$$

133 with the asymptotic constraint

$$134 \quad (2.5) \quad \psi(0, Z) = 0 \quad \text{and} \quad \lim_{\|(R, Z)\| \rightarrow \infty} \psi(R, Z) = 0.$$

135 Here I_i is the external current density in the i -th poloidal field coil located at $(R_i^c, Z_i^c) \in \mathcal{H}$ (see [Figure 1](#) for
 136 the locations of coils; solenoids can be handled similarly), and $J_\phi(R, \psi)$ is the prescribed toroidal component
 137 of the plasma current density, generally a non-linear function of ψ , in the plasma domain $\mathcal{P}(\psi)$. Note that
 138 the magnetic potential outside of the plasma region simply satisfies Poisson's equation. We further introduce
 139 the limiter domain \mathcal{L} , satisfying $\mathcal{P}(\psi) \subset \mathcal{L} \subset \mathcal{H}$. For this purpose, the limiter corresponds to the first wall in
 140 the tokamak device surrounding the confinement region. The limiter domain is the entire region accessible
 141 by the plasma, while the plasma domain $\mathcal{P}(\psi)$ is the region bounded by the last closed poloidal flux surface
 142 inside the limiter bounded domain, see the discussion in [\[16\]](#) for instance.

143 The free-boundary problem is still underdetermined because (a) the source term J_ϕ (more specifically,
 144 p and g) is typically provided as a function of the normalized $\bar{\psi} \in [0, 1]$ from experiment (its normalization
 145 shall be explained later); (b) the coil currents are not known and in fact they need to be optimized based
 146 on the targeted shape of $\mathcal{P}(\psi)$, along with solving the free-boundary problem. Both of the uncertainties
 147 introduce more challenges to solve the free-boundary problem. However, the uncertainties are closely related
 148 to the concept of plasma shape control in tokamaks, which is a critical topic in tokamak discharge design as
 149 well as in tokamak machine design.

150 To address (a), one can define a normalized flux function,

$$151 \quad \bar{\psi} \equiv \frac{\psi - \psi_o}{\psi_X - \psi_o},$$

152 where ψ_X is the value of ψ at the plasma boundary, $\partial\mathcal{P}(\psi)$, and ψ_o is the value at the magnetic axis. In a
 153 diverted tokamak plasma, a magnetic *separatrix* demarcates the topologically distinct closed and open flux

154 regions. Plasmas also reside in the open flux region outside the separatrix, which is known as the scrape-off
 155 layer, but it has relatively small pressure gradient and plasma current. For the purpose of defining a Grad-
 156 Shafranov equilibrium, the plasma boundary is usually given by the separatrix or a closed flux surface in
 157 the immediate vicinity of the separatrix. In this paper, we will set ψ_X to be the ψ value of the saddle point
 158 that labels the magnetic separatrix. In a limited plasma, ψ_X labels the last closed flux surface beyond which
 159 magnetic field lines interact a limiter protruded from camber wall. Since ψ is the poloidal flux, it must be a
 160 monotonic function between ψ_o and ψ_X in a tokamak. The normalized flux $\bar{\psi}$ is therefore between 0 and 1
 161 for all ψ values inside $\mathcal{P}(\psi)$. Numerical techniques to efficiently locate those points will be discussed in the
 162 following section. Note it is expected that the overall source term can have jumps on the right hand side
 163 in (2.4). The existence of solutions to (2.4) under certain normalization and jump source terms has been
 164 considered in some PDE theory studies, see [1] for instance.

165 In addressing (b), one usually imposes some constraints based on a targeted plasma shape $\mathcal{P}_{\text{target}}$. The
 166 targeted plasma shape is predetermined (up to some small variation) when the fusion device is designed.
 167 The standard approach is to impose a few control points along the desired shape such that the computed
 168 poloidal flux on those points is equal to the value on the separatrix, i.e.,

$$169 \quad \psi(R_k, Z_k) = \psi_X, \quad k = 1, \dots, n_p,$$

170 where (R_k, Z_k) is selected along the plasma boundary $\partial\mathcal{P}_{\text{target}}$ and ψ_X is determined numerically during
 171 the normalization. It essentially means we minimize the distance between the separatrix from the numerical
 172 solution and the targeted plasma shape. In the following section, we will discuss how to impose a proper
 173 optimization problem to determine the coil current based on the above constraints.

174 **3 Cut-cell algorithm** A cut-cell algorithm for the free-boundary Grad-Shafranov problem is described
 175 in this section. The basic algorithm closely follows the cut-cell algorithm proposed for Poisson's equation
 176 in [29]. We further introduce a level set function to improve the efficiency of the cut-cell algorithm and suit
 177 the specific needs of the Grad-Shafranov equation.

178 Consider a uniform grid defined by

$$179 \quad \mathbf{x}_{i,j} = (R_i, Z_j) = (R_{\min} + i\Delta R, Z_{\min} + j\Delta Z).$$

180 Let $\psi_{i,j}$ be a grid-point approximation to $\psi(\mathbf{x}_{i,j})$ and rewrite Equation (2.2) as

$$181 \quad \tilde{\nabla} \cdot \left(\frac{1}{R} \tilde{\nabla} \psi \right) = \mu_0 J_\phi(R, \psi)$$

182 Integrating it on the cell corresponding to the grid point (i, j) and applying the divergence theorem give

$$183 \quad (3.1) \quad \frac{1}{\Delta R \Delta Z} \int_{\partial C} \left(\frac{1}{R} \tilde{\nabla} \psi \right) \cdot \mathbf{n} ds = \frac{1}{\Delta R \Delta Z} \int_C \mu_0 J_\phi(R, \psi) dR dZ, \quad C \in \Omega$$

184 where \mathbf{n} denotes the unit outward normal direction vector of ∂C . Assuming C is a full cell, Equation (3.1)
 185 can be discretized as

$$186 \quad (3.2) \quad \frac{1}{\Delta R \Delta Z} \left[F_{i,j+\frac{1}{2}} - F_{i,j-\frac{1}{2}} + F_{i+\frac{1}{2},j} - F_{i-\frac{1}{2},j} \right] = \mu_0 J_\phi(R_{i,j}, \psi_{i,j}),$$

where $F_{r,s}$ is the flux along the cell edge, given by

$$F_{i,j+\frac{1}{2}} = \frac{\Delta R}{R_{i,j+\frac{1}{2}}} \frac{\psi_{i,j+1} - \psi_{i,j}}{\Delta Z}, \quad F_{i+\frac{1}{2},j} = \frac{\Delta Z}{R_{i+\frac{1}{2},j}} \frac{\psi_{i+1,j} - \psi_{i,j}}{\Delta R}.$$

187 Note that (3.2) on the full cell is a standard second-order discrete operator.

188 If C is a cut-cell as shown in Figure 2, introducing a volume fraction $\Lambda_{i,j} \in [0, 1]$, the resulting dis-
 189 cretization of (3.1) can be written as

$$190 \quad (3.3) \quad \frac{1}{\Delta R \Delta Z \Lambda_{i,j}} \left[F_{i,j+\frac{1}{2}} - F_{i,j-\frac{1}{2}} + F_{i+\frac{1}{2},j} - F_{i-\frac{1}{2},j} - F_{i,j}^f \right] = \mu_0 J_\phi(R_{i,j}, \psi_{i,j}),$$

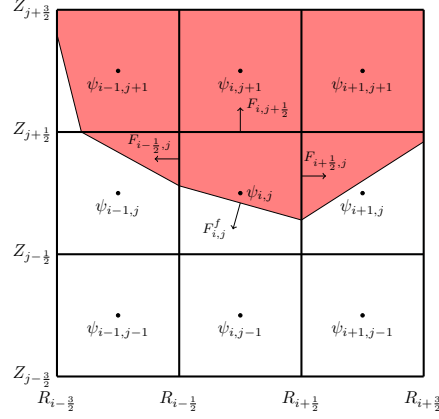


Fig. 2: Grid points and cut-cells. The red region is the computational domain.

191 where $F_{i,j}^f$ is the flux along the boundary in the cut-cell. Note that $F_{r,s}$ can be 0 if the corresponding edge
 192 is not involved in the control volume.

193 We further introduce a level set function for efficiently distinguishing interior cell, exterior cell, and
 194 cut-cell as well as determining whether a cell edge is a partial edge or a full edge. We first construct a level
 195 set function $\rho(R, Z)$ given by

$$196 \quad \rho(R, Z) = \begin{cases} +d, & (R, Z) \in \Omega^c, \\ 0, & (R, Z) \in \partial\Omega, \\ -d, & (R, Z) \in \Omega, \end{cases}$$

197 where d is the shortest distance from the point (R, Z) to the boundary $\partial\Omega$. For convenience, let $\rho_{i+\frac{1}{2},j+\frac{1}{2}}$
 198 denote $\rho(\mathbf{x}_{i+\frac{1}{2},j+\frac{1}{2}})$, which is evaluated on the corners of each cell.

For the boundary described by an analytical expression, defining a level set value on a given point is straightforward and therefore it is ignored. However, in a practical problem of a tokamak geometry, the geometry is described by a set of points on the boundary, and thus it is less obvious to define the value of level set function on a point. Here, we present how to define the level set function for a general irregular domain Ω . Given a set of data $\{(r_1, z_1), (r_2, z_2), (r_3, z_3), \dots, (r_n, z_n)\}$ which describes the boundary of the irregular domain, we connect all the points as a polygon. Let

$$\bar{r} = \sum_{i=1}^n \frac{r_i}{n}, \quad \bar{z} = \sum_{i=1}^n \frac{z_i}{n}, \quad \bar{P} = (\bar{r}, \bar{z}), \quad P_i = (r_i, z_i), \quad i = 1, 2, \dots, n.$$

199 To compute d for a given point $P = (r, z)$, we locate the boundary point P_i which has the shortest distance
 200 to P . Then we determine d as the shorter distance between P and the two boundary segments connected
 201 with P_i .

202 The next step is to decide whether P is in or out of domain if $d \neq 0$. We choose (r_1, z_1) as a starting
 203 point and compute the rotated angle θ_i from $\overrightarrow{PP_1}$ to $\overrightarrow{PP_i}$ where $\theta_i \in (-\pi, \pi]$. For any given point $P = (r, z)$,
 204 by checking the value of the angle θ rotating from $\overrightarrow{PP_1}$ to \overrightarrow{PP} , we can get the interval $[\theta_i, \theta_{i+1}]$ where θ
 205 belongs. We denote $P_{intersect}$ as the intersection point of line passing through P and \bar{P} and the line passing
 206 through P_i and P_{i+1} . By comparing values of the distance between \bar{P} and $P_{intersect}$ and the distance between
 207 \bar{P} and P , we can determine whether the point P is in or out of domain. Some care is needed for the corner
 208 point where the curve becomes not convex.

209 The precomputed level set function can be used to distinguish the cell category. For example, in [Figure 2](#),
 210 since ρ at each corner of the cell $(i, j-1)$ is positive, we determine the cell $(i, j-1)$ is an exterior cell. Similarly,
 211 the cell $(i, j+1)$ is an interior cell in the domain Ω , as ρ values at all the corners of the cell are negative. The
 212 cell (i, j) is a cut-cell which contains part of the boundary as signs of ρ value at each corner are not the same.
 213 The precomputed level set function can also be used to evaluate some geometry quantities associated with

214 the cut-cells [38, 37]. For a partial edge, we can use the information of the level set function ρ at the two
 215 endpoints to approximate its aperture. Suppose $\mathbf{x}_f = (R_{i+\frac{1}{2}}, Z_f) \in \partial\Omega$, assuming two points $\mathbf{x}_{i+\frac{1}{2},j+\frac{1}{2}} \in \Omega$
 216 and $\mathbf{x}_{i+\frac{1}{2},j-\frac{1}{2}} \in \Omega^c$ border \mathbf{x}_f , we use the formula given in [12]

$$217 \quad Z_{i+\frac{1}{2},j+\frac{1}{2}} - Z_f = \frac{\rho_{i+\frac{1}{2},j+\frac{1}{2}} \Delta Z}{\rho_{i+\frac{1}{2},j+\frac{1}{2}} - \rho_{i+\frac{1}{2},j-\frac{1}{2}}},$$

$$218 \quad Z_f - Z_{i+\frac{1}{2},j-\frac{1}{2}} = \frac{-\rho_{i+\frac{1}{2},j-\frac{1}{2}} \Delta Z}{\rho_{i+\frac{1}{2},j+\frac{1}{2}} - \rho_{i+\frac{1}{2},j-\frac{1}{2}}},$$
 219

220 to evaluate the distances between \mathbf{x}_f and two points $\mathbf{x}_{i+\frac{1}{2},j+\frac{1}{2}}$ and $\mathbf{x}_{i+\frac{1}{2},j-\frac{1}{2}}$. Therefore, as an example, the
 221 fraction of the right edge of the partial cell (i, j) in Figure 2, denoted as $a_{i+\frac{1}{2},j}$, is given by

$$222 \quad a_{i+\frac{1}{2},j} = \frac{\rho_{i+\frac{1}{2},j+\frac{1}{2}}}{\rho_{i+\frac{1}{2},j+\frac{1}{2}} - \rho_{i-\frac{1}{2},j+\frac{1}{2}}}.$$

223 After calculating the apertures of each cell edge, the area of the front, denoted as A^f , in the cut-cells
 224 can be evaluated based on the relationship

$$225 \quad A_{i,j}^f \mathbf{n}_{i,j}^{in} = \Delta R (a_{i+\frac{1}{2},j} - a_{i-\frac{1}{2},j}) \hat{i} + \Delta Z (a_{i,j+\frac{1}{2}} - a_{i,j-\frac{1}{2}}) \hat{j},$$

226 where \mathbf{n}^{in} is the inward-facing normal at the boundary in cut-cell. Note that for interior cells, since all
 227 aperture values are 1, A^f is also applicable to the interior cells as $A^f = 0$.

228 Next, we discuss the discretization of Equation (3.3) on cut-cells. $F_{i,j+\frac{1}{2}}$, $F_{i,j-\frac{1}{2}}$, $F_{i+\frac{1}{2},j}$, and $F_{i-\frac{1}{2},j}$ are
 229 evaluated at the midpoint of cell edge covered by Ω . To evaluate flux on a partial edge, in order to keep a
 230 second-order accurate approximation of the fluxes through cell edges, a linear interpolation between values
 231 at the midpoints of full edges is used. For example, in Figure 2, the flux $F_{i+\frac{1}{2},j}$ is evaluated at the midpoint
 232 of the partial edge centered at $\mathbf{x}_{i+\frac{1}{2},j}$ by linearly interpolating the flux at neighboring full edge centered at
 233 $\mathbf{x}_{i+\frac{1}{2},j+1}$, which is stated as the following

$$234 \quad F_{i+\frac{1}{2},j} = \frac{1}{R_{i+\frac{1}{2},j}} \Delta Z \left[\frac{1 + a_{i+\frac{1}{2},j}}{2} \frac{(\psi_{i+1,j} - \psi_{i,j})}{\Delta R} + \frac{1 - a_{i+\frac{1}{2},j}}{2} \frac{(\psi_{i+1,j+1} - \psi_{i,j+1})}{\Delta R} \right].$$

235 The flux F^f is evaluated at the midpoint of the boundary covered by the cut-cell. A three-point gradient
 236 stencil is proposed in [29] to evaluate the normal component of the gradient of ψ and deployed in the current
 237 work. The readers are referred to [29] for the details of the discretization and other techniques such as small
 238 cell ignorance, which are also deployed here.

239 Finally, we discuss how to evaluate the source term of the Grad-Shafranov equation along with the
 240 cut-cells. Note that $\mu_0 J_\phi(R_{i,j}, \psi_{i,j})$ should be evaluated at the centroid of the covered part in each cut-
 241 cell. There are three types of cut-cells covered by the physical domain, triangle, trapezoid, and pentagon,
 242 based on the piecewise-linear representation of the boundary in each cut-cell. With the geometry information
 243 gathered, we can now turn the problem into computing the centroid of a convex and closed polygon. Suppose
 244 a convex and closed polygon is defined by the ordered vertices $\{(r_1, z_1), (r_2, z_2), \dots, (r_{n+1}, z_{n+1})\}$ with
 245 $(r_{n+1}, z_{n+1}) = (r_1, z_1)$, the centroid of a convex polygon is computed as the following

$$246 \quad C_{polygon} = \frac{1}{3} \left(\frac{\sum_{i=1}^n (r_i + r_{i+1})(r_i z_{i+1} - r_{i+1} z_i)}{\sum_{i=1}^n (r_i z_{i+1} - r_{i+1} z_i)}, \frac{\sum_{i=1}^n (z_i + z_{i+1})(r_i z_{i+1} - r_{i+1} z_i)}{\sum_{i=1}^n (r_i z_{i+1} - r_{i+1} z_i)} \right),$$

247 which will be used during the evaluation of the source term.

248 **4 Free-boundary Grad-Shafranov solver** A common approach to solve the free-boundary prob-
 249 lem (2.4) is to reformulate the problem in a bounded computational domain Ω , see [26, 16, 28] for instance.
 250 The requirement for Ω is that it should enclose the plasma domain $\mathcal{P}(\psi)$, but all the poloidal field coils,
 251 including those for vertical stability control and divertor flux shaping, should be outside Ω . This is to ac-
 252 commodate a Green's function approach in calculating the coil current contribution to the Grad-Shafranov

253 equilibrium through the poloidal flux ψ on $\partial\Omega$. As the plasma domain $\mathcal{P}(\psi)$ is unknown *a priori*, the bounded
 254 domain Ω , in practice, should contain at least the limiter bounded domain \mathcal{L} . The precise choice for $\partial\Omega$
 255 constrains the numerical discretization for the Grad-Shafranov solver and motivates the cut-cell approach in
 256 section 3.

257 In this section we will focus on the overall scheme for the free-boundary Grad-Shafranov solver. The
 258 main algorithm is based on a Picard iteration

$$\begin{aligned} 259 \quad \Delta^* \psi &= \mathcal{S}(\psi^{\text{old}}), & (R, Z) \in \Omega, \\ 260 \quad \psi &= \mathcal{D}(\psi^{\text{old}}), & (R, Z) \in \partial\Omega, \end{aligned}$$

262 where the operators \mathcal{S} and \mathcal{D} stand for the steps to determine the source term and boundary values from
 263 the old solution ψ^{old} , respectively. The details of \mathcal{S} and \mathcal{D} will be given in the following discussion. Note
 264 that unlike a fixed-boundary problem [43], Newton's method is challenging to employ here, which will be
 265 further discussed in Subsection 4.2.

266 The discussions will start from determining the source term from an old solution, i.e., the operator
 267 \mathcal{S} , which includes a search algorithm to locate ψ_o and ψ_X and a barycentric interpolation to interpolate
 268 the given source profile. After the source term is determined, the approaches to determine the boundary
 269 condition, i.e., the operator \mathcal{D} , are discussed. Then a minimization problem is introduced to optimize the coil
 270 current density based on the old solution. Finally, the full algorithm based on Picard iteration with Aitken's
 271 acceleration is summarized. Note that all the approaches in this section are applicable to a free-boundary
 272 solver on a rectangular domain and some of them were actually initially proposed for the rectangular domain.

273 **4.1 Evaluate the source term** Evaluating the source term in the free-boundary problem is a multi-step
 274 process. Since ψ_o and ψ_X are to be found from the solution, Grad-Shafranov equilibria are usually specified
 275 by prescribing the source term as functions of $\bar{\psi}$, which is always well-defined for any trial solution that has
 276 varying ψ_o and ψ_X . The most straightforward case has known $g(\bar{\psi})$ and $p(\bar{\psi})$ profile, often supplemented by
 277 integral constraints of specified plasma beta and plasma toroidal current. A practically useful alternative is
 278 to specify the safety factor profile, $q(\bar{\psi})$, for $\bar{\psi} \in [0, 0.95]$.

279 *Search for magnetic axis and saddle points.* Since the trial solution of Grad-Shafranov equation gives ψ
 280 on grid points, it is necessary to locate ψ_o and ψ_X first. As discussed in Section 2, ψ_o and ψ_X correspond to
 281 the critical points where $\|\nabla\psi\| = 0$. Therefore, a simple minimization problem can be defined to find them,

$$282 \quad (4.1) \quad \min_{R,Z} \|\nabla\psi\|^2.$$

283 To simplify the procedure, we use a two-step search algorithm to locate those points. We first evaluate
 284 numerical gradient on all the grid points in order to identify several candidate points with small gradient
 285 values. We choose 10 candidate points in our solver (here 10 points are chosen to locate all the saddle points
 286 and magnetic axis and they often converge to one of those points. During the searching process, if the point
 287 locates out of the computational domain, we delete the corresponding candidate). Newton's method for
 288 optimization is then used to solve Equation (4.1) using the candidate points as the initial guess. During
 289 the optimization, the Newton's method needs a continuous representation of ψ for any (R, Z) , for which
 290 a barycentric interpolation formula is used. Since the initial guesses are very good, the Newton's method
 291 typically only needs a few iterations to converge. After all the critical points are found, a further check is
 292 performed to remove the duplicate critical points. Finally, ψ_o and ψ_X are determined based on the Hessian
 293 of ψ . If the critical point has $\partial_{rr}\psi \partial_{zz}\psi - (\partial_{rz}\psi)^2 > 0$, it is a local minimum or maximum, which corresponds
 294 to the magnetic axis (ψ_o), and if it has $\partial_{rr}\psi \partial_{zz}\psi - (\partial_{rz}\psi)^2 < 0$, it is a saddle point corresponding to the
 295 separatrix (ψ_X). We comment that the above algorithm does not guarantee to find a global minimum for a
 296 general solution. However, for a physical MHD equilibrium, the solution has a well-defined local minimum
 297 problem as we shall see in the numerical section, and thus the above algorithm works well. In the case
 298 when multiple local minimum/maximum points are found, we determine the solution is invalid. When that
 299 happens in the free-boundary solver, it means the solver has diverged and the iteration will stop. In the case
 300 when multiple saddle points are found, if the poloidal flux is convex inside the plasma region, we choose the
 301 smallest (or largest if ψ is concave) value among all the saddle points. Note the algorithm described above
 302 is a critical piece in the free-boundary solver and the subroutine has to be called routinely in the solver. As

303 long as all the saddle points and magnetic axis are successfully located, this ensures an accurate evaluation of
 304 the source term. Therefore, the number of candidate points (10 in our case) has no impact on the accuracy.

305 Given the plasma pressure profile and the toroidal field function profile as functions of the poloidal
 306 flux on a set of known points, finding a proper interpolation method to describe the source term in the
 307 Grad-Shafranov equation is very important. We note however, at least on equispaced case, polynomial
 308 interpolation is not proper, no matter what form we use, due to Runge Phenomenon of high order. It is
 309 reasonable to shift to rational interpolation [7, 8, 17, 47]. In our work, the barycentric rational interpolation
 310 (a 4th-order rational interpolation) is employed to evaluate the plasma pressure and the toroidal field function
 311 of the poloidal flux. This should allow highly accurate representation of the source term on grid points from
 312 numerical data profiles of $p(\bar{\psi})$ and $g(\bar{\psi})$.

313 **4.2 Determine ψ_b on $\partial\Omega$** The approach to determine the boundary condition value from the external
 314 coil source is well known [26, 16, 28]. We give a brief discussion on two choices we have experimented. Note
 315 that there is a known Green's function for the toroidal elliptic operator Δ^*

$$316 \quad G(\mathbf{R}; \mathbf{R}') = \frac{1}{2\pi} \frac{\sqrt{RR'}}{k} [(2 - k^2)K(k) - 2E(k)],$$

317 where $K(k)$ and $E(k)$ are complete elliptic integrals of the first and second kind [44] and k is given by

$$318 \quad k^2 = \frac{4RR'}{(R + R')^2 + (Z - Z')^2}.$$

319 A straightforward approach based on the Green's third identity defines the boundary value as

$$320 \quad (4.2) \quad \psi_b(R', Z') = - \int_{\Omega} \mu_0 G(R, Z; R', Z') \tilde{J}_{\phi}(R, Z) drdz - \sum_{i=1}^{n_c} \mu_0 G(R_i^c, Z_i^c; R', Z') I_i,$$

321 where the source is given by $\tilde{J}_{\phi}(R, Z) = J_{\phi}(R, \psi^{\text{old}})$. A more efficient approach that only computes a line
 322 integral was proposed in [49]. It involves solving the elliptic problem with the same source term but a
 323 homogenous boundary condition,

$$324 \quad (4.3) \quad \begin{aligned} \Delta^* U &= \mu_0 R \tilde{J}_{\phi}(R, Z), & (R, Z) \in \Omega, \\ U &= 0, & (R, Z) \in \partial\Omega. \end{aligned}$$

325 Integrating the following identity

$$326 \quad \nabla \cdot [U \frac{1}{R^2} \nabla G(\mathbf{x}; \mathbf{x}')] - \nabla \cdot [G(\mathbf{x}; \mathbf{x}') \frac{1}{R^2} \nabla U] = \frac{1}{R^2} U \Delta^* G(\mathbf{x}; \mathbf{x}') - \frac{1}{R^2} G(\mathbf{x}; \mathbf{x}') \Delta^* U$$

327 over the computational domain Ω leads to the formulations for both boundary and interior points,

$$328 \quad (4.4) \quad \psi_b(R', Z') = - \int_{\partial\Omega} \frac{dl}{R} G(R, Z; R', Z') \frac{\partial U}{\partial n} - \sum_{i=1}^{n_c} \mu_0 G(R_i^c, Z_i^c; R', Z') I_i,$$

$$329 \quad (4.5) \quad \psi_{interior}(R', Z') = - \int_{\partial\Omega} \frac{dl}{R} G(R, Z; R', Z') \frac{\partial U}{\partial n} - \sum_{i=1}^{n_c} \mu_0 G(R_i^c, Z_i^c; R', Z') I_i + U(R', Z').$$

330
 331 The second approach comes at the cost of inverting an elliptic operator, which only accounts for an extra
 332 cost of $O(N \log N)$ thanks to the algebraic multigrid (AMG) preconditioner we used. Thus, we typically
 333 choose the second approach in our solver for its efficiency. The second approach can also provide an initial
 334 guess on the entire domain through (4.5) in the initialization stage, which is useful to provide a better initial
 335 guess and speed up the convergence. One issue is that the Green's function is singular when $\mathbf{R}' = \mathbf{R}$, which
 336 is more problematic in the line integration of the second approach. In the implementation, when \mathbf{R}' is too
 337 close to \mathbf{R} , we perturb \mathbf{R}' with a small distance of ϵ , which typically happens in one or two cells. It appears
 338 to be sufficient to produce a smooth boundary condition.

339 Note that when Newton’s method is applied to the free-boundary problem, (4.2) is more suitable to use
 340 in its nonlinear residual. This, however, results into a globally coupled system due to the global operator
 341 in (4.2). The Jacobian matrix assembly and its inversion are thus expensive. Moreover, AMG typically
 342 cannot solve such a system involving non-local boundary conditions. All those facts contribute to the
 343 challenge of employing Newton’s method here. To resolve the issue due to ψ_b , Ref. [21] proposed to extend
 344 the computational domain to include all the external currents. This results into a much less banded system,
 345 which is easier to assemble, and the success of Newton’s method but at the cost of solving on a domain much
 346 larger than that in the current work.

347 **4.3 Determine coil currents** The last important piece in the algorithm is to determine the coil current
 348 after a solution ψ is given. It is important to adjust the coil current based on the solution so that the
 349 resulting plasma has the intended position and shape. We therefore optimize the current density through
 350 a minimization problem based on some constraints imposed through the targeted plasma shape. Given n_c
 351 coils in the device and the desired plasma region $\mathcal{P}_{\text{target}}$, we first select n_p points along its boundary $\partial\mathcal{P}_{\text{target}}$
 352 such that the points are evenly spaced and $n_p > n_c$. Equation (4.5) provides a good approximation to ψ at
 353 those points for any current density. The search algorithm on the other hand provides the current separatrix
 354 value ψ_X . Minimizing the ψ values at those points with respect to ψ_X gives a minimization problem to find
 355 the required coil currents,

$$356 \quad (4.6) \quad \mathbf{I}_c = \arg \min_{\mathbf{I}_c} \left\{ \gamma \sum_{k=1}^{n_c} I_k^2 + \sum_{j=1}^{n_p} \left[- \int_{\partial\Omega} \frac{dl}{R} G(R, Z; R'_j, Z'_j) \frac{\partial U}{\partial n} - \sum_{i=1}^{n_c} \mu_0 G(R_i^c, Z_i^c; R'_j, Z'_j) I_i + U(R'_j, Z'_j) - \psi_X \right]^2 \right\},$$

357 where \mathbf{I}_c is a vector of size n_c , consisting of the current values I_i , and the first term is a penalty term to
 358 prevent the system from becoming ill-posed.

359 We found that the resulting current density is sensitive to the choice of γ . Part of the reason is that
 360 the problem setup is based on practical units and the density values can be dramatically different from each
 361 other in different coils. To address that, we found the Generalized Cross validation (GCV) approach in [20]
 362 provides a good estimate of γ . To utilize GCV, the minimization problem (4.6) is written in the form of

$$363 \quad \mathbf{I}_c = \arg \min_{\mathbf{I}_c} \left(\gamma \|\mathbf{I}_c\|^2 + \frac{1}{n_p} \|X \mathbf{I}_c - \mathbf{y}\|^2 \right),$$

365 where (4.6) indicates X is a matrix of size $n_p \times n_c$, and \mathbf{y} is a vector of size n_p . The GVC approach suggests
 366 the best γ should be computed as the minimizer of $V(\gamma)$, given by

$$367 \quad V(\gamma) = \frac{1}{n_p} \|(I - A(\gamma))\mathbf{y}\|^2 / \left[\frac{1}{n_p} \text{Trace}(I - A(\gamma)) \right]^2,$$

368 where $A(\gamma) \equiv X(X^T X + n_p \gamma I)^{-1} X^T$ and I is an identity matrix of size $n_p \times n_p$. In the initial experiment,
 369 we implement a simple search algorithm to locate the optimized value of γ in each iteration for determining
 370 the current. We found that the optimized γ falls within the range of $[10^{-16}, 10^{-13}]$ for the practical ITER
 371 equilibrium problem. Therefore, to avoid recalculating γ in each iteration, we fix $\gamma = 10^{-15}$ in all of our
 372 numerical examples, which is sufficient to provide less sensitive current values in the final solver. The current
 373 values of the equilibrium are found in the range of $[10^3, 10^7]$, and thus the regularization term is not negligible
 374 with the given γ value.

375 **4.4 Full algorithm** With all the important pieces laid out in the previous sections, a full algorithm for
 376 the free-boundary Grad-Shafranov equation will be described. In particular, two versions of the algorithm
 377 will be discussed and an additional acceleration technique will be adopted to improve the robustness and
 378 efficiency of the solver.

379 *Three-loop algorithm.* The first version of the algorithm would consist of three loops as the main
 380 steps (Figure 3a): the most inner loop inverts the operator Δ^* with a given boundary condition ψ_b ; the

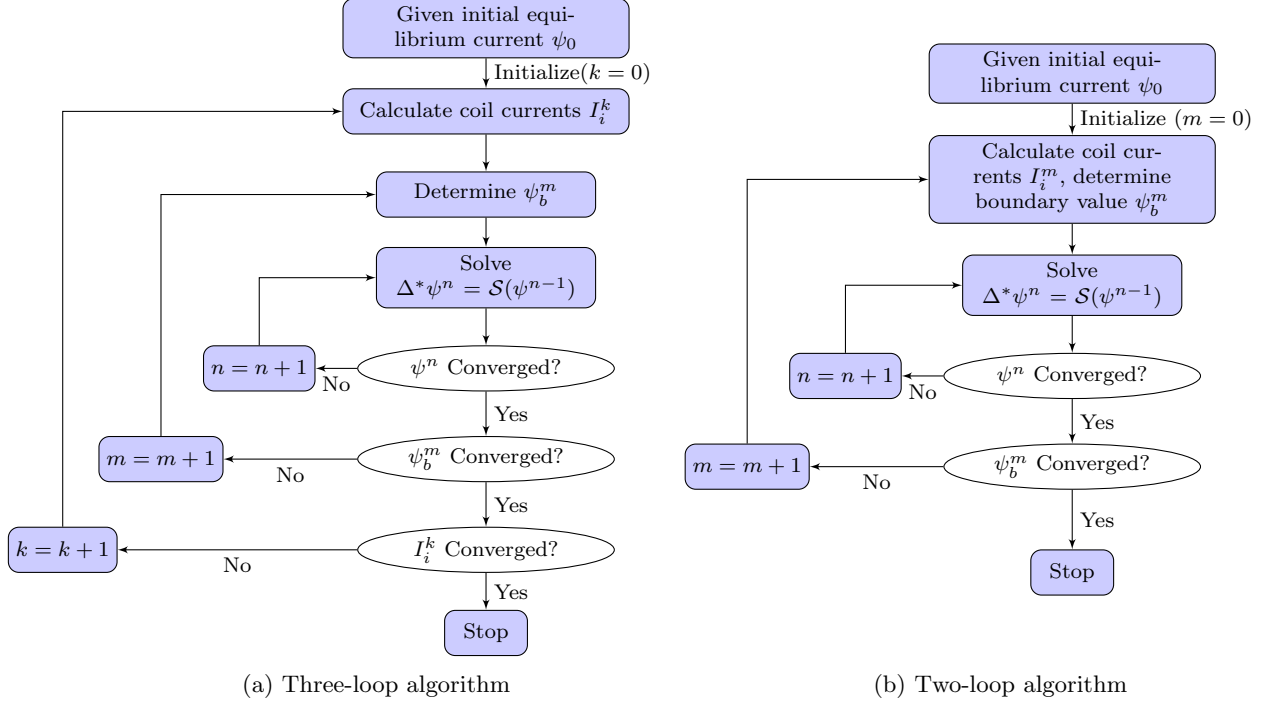


Fig. 3: Flowcharts for the free-boundary Grad-Shafranov solvers.

381 outer loop will adjust ψ_b until some convergence threshold is reached for some fixed current density $\{I_i\}$ (see
 382 [Figure 1](#) for the current locations); then the most outer loop would adjust the coil currents such that the
 383 magnetic separatrix will align with the control points. Although this idea appears to be plausible and in
 384 fact it is the first algorithm we implemented, we found that this version of the algorithm is sensitive to a
 385 small perturbation of the current, which could easily result into the failure of convergence of the two inner
 386 loops. One of the fundamental issues in this approach is that there is an underlying assumption that for any
 387 given current density, there exists a corresponding equilibrium, despite not satisfying the plasma domain
 388 constraint. This, however, may not be true in general, and even if there exists such an equilibrium, its g and
 389 p profiles could be very different from the profiles being used, and therefore the evaluated source term could
 390 be very inaccurate, which could contribute to the divergence of the solver. Therefore, we do not pursue
 391 further using this version of the algorithm in the current work.

392 *Two-loop algorithm* [30]. The second version of the algorithm consists of two loops as the main steps ([Fig-](#)
 393 [ure 3b](#)): the inner loop still inverts the operator Δ^* with a given boundary condition ψ_b while the outer
 394 loop simultaneously updates ψ_b and $\{I_i\}$ until some threshold with respect to ψ_b is reached. This version
 395 turns out to be much more robust than the previous version, which is the main algorithm we used in the
 396 free-boundary solver. Its robustness can be further improved by the acceleration techniques discussed below.

Aitken's acceleration As previously discussed, the general procedure of the algorithm is a Picard iteration,
 and its drawback is its slow convergence. A common approach to accelerate its convergence is through under-
 relaxation, i.e.,

$$\psi^{new} = (1 - \alpha)\psi^{old} + \alpha\tilde{\psi}^{new}.$$

397 where α is the under-relaxation coefficient, typically between 0 and 1. The optimal value of α , however, is
 398 problem dependent and for nonlinear problems it can vary during the iterations. The optimal relaxation
 399 value can be approximated through the well known Aitken's acceleration [41]. The version we adopted is

400 given by

$$\begin{aligned}
 D\psi^{n+1} &= \psi^n - \tilde{\psi}^{n+1}, \\
 \lambda^{n+1} &= \lambda^n + (\lambda^n - 1) \frac{(D\psi^n - D\psi^{n+1})^T D\psi^{n+1}}{\|D\psi^n - D\psi^{n+1}\|^2}, \\
 \alpha &= 1 - \max(\min(\lambda^{n+1}, \lambda_{\max}), \lambda_{\min}), \\
 \psi^{n+1} &= (1 - \alpha)\psi^n + \alpha\tilde{\psi}^{n+1},
 \end{aligned}
 \tag{4.7}$$

402 where $\tilde{\psi}^{n+1}$ is the solution directly solved from the Picard iteration, and $\lambda_{\min} = 0$ and $\lambda_{\max} = 0.95$ are
 403 typically chosen. Ref. [9] suggests a preset value of $\lambda = 0.3$ in the first iteration, which is adopted in the
 404 current work. Note that the technique fits into the current framework very well due to its simplicity and
 405 computational expedience, and we use it to accelerate both the inner and outer loops. It is found that the
 406 Aitken's acceleration greatly improves the convergence of the full algorithm, which will be demonstrated in
 407 the numerical section. We note however that there are other techniques such as Anderson's acceleration and
 408 nonlinear GMRES available for the Picard iteration. We plan to explore more acceleration techniques in
 409 future work.

410 Finally, we summarize the full algorithm in [Algorithm 4.1](#), which consists of two main loops, Aitken's
 411 accelerations and all the important steps described in the previous sections. Note that Step 2 corresponds
 412 to all the steps related to the cut-cell algorithm, which will be addressed carefully in the next section. In
 413 the nonlinear solver, we set the relative difference of the boundary values $\epsilon_{out} = 1e-4$ as the convergence
 414 criterion for the outer loop and the relative difference of the solution values $\epsilon_{in} = 1e-3$ as the convergence
 415 criterion for the inner loop. For Krylov linear solvers in PETSc, we set the relative tolerance of convergence
 416 as $1e-5$, and the absolute tolerance of convergence as $1e-8$.

417 **5 Parallel implementation** Some aspects of the implementation of the free boundary Grad-Shafranov
 418 solver with the cut-cell algorithm will be discussed in this section. The free boundary Grad-Shafranov solver
 419 described in this work is implemented in parallel under the PETSc framework [3] using a standard domain
 420 decomposition approach. All the vectors, arrays and matrices use the parallel distributed data structure
 421 provided by PETSc and its communication between sub-domains is based on the message-passing interface
 422 (MPI). All the linear and nonlinear solvers described in this work are implemented through PETSc and the
 423 elliptic operator, Δ_h^* , is preconditioned with algebraic multigrid preconditioners to improve its efficiency and
 424 scalability.

425 The data structure is based on a Cartesian structured mesh (DMDA) provided by PETSc. The solution
 426 is stored as grid points while its control volume is straightforwardly implied. This is a common approach
 427 to adapt a finite volume algorithm in a finite difference code, see [10] for instance. To develop a cut-cell
 428 algorithm based on DMDA, all the grid points are distinguished into three categories of active points, inactive
 429 points and cut-cell points, based on its underlying control volume. If the control volume has the full cell size,
 430 then it is an active point, otherwise it is either a cut-cell point (if the volume fraction is between 0 and 1)
 431 or an inactive point (if the volume fraction is 0). Note that the approach distinguishing active and inactive
 432 grid points is commonly used in overlapping grids, where some of the grid points are not involved in the
 433 discretization, see [4, 5] for instance. Since the scheme is based on a five-point stencil, the discrete operator
 434 on the active points is well-defined, and meanwhile a redundant equation is imposed on all the inactive
 435 points when discretizing the elliptic operator on those points (simply solving $a\psi_{inactive} = 0$ where a is a large
 436 constant). The main difference between the free-boundary solvers on a standard Cartesian mesh and on cut
 437 cells thus lies in the additional treatment related to the cut-cell points. Involving a redundant equation into
 438 the discretization leads to an operator of the same size as the standard operator on a Cartesian grid. All the
 439 approaches described above guarantee many PETSc functions for DMDA can be directly used in the cut-cell
 440 algorithm, which eases some data structure allocations and implementations of linear or nonlinear operators.
 441 It also provides a rather straightforward path for the future improvement through geometry multigrid.

442 To further ease the treatment related to cut-cells, several 2D arrays are precomputed to store the
 443 geometry information and the information related to level set function. For example, volume fractions and
 444 area fractions of each cell, apertures of each edge and midpoint values of each cut edge and interpolation
 445 points are precomputed before performing the Picard iterations, which greatly improves the solver efficiency.

Algorithm 4.1 Free-boundary Grad-Shafranov solver

```

1: Choose an irregular domain  $\Omega$  to reformulate the free boundary problem (2.4) and (2.5)
2: Construct a level set function  $\tilde{\phi}$  and prepare a cut-cell mesh
3: Given an initial equilibrium  $\psi^0$ , solve Equation (4.3) for  $U(R, Z)$ 
4: Calculate coil currents  $I_i^0$  according to Equation (4.6)
5: Determine boundary value  $\psi_b^0$  according to Equation (4.4)
6: //Outer loop
7: for  $m = 0$  to  $m_{\max}$  do
8:   //Inner loop
9:   for  $n = 0$  to  $n_{\max}$  do
10:    Search  $\psi_o$  and  $\psi_x$  in  $\psi^n$ 
11:    Solve  $\Delta_h^* \tilde{\psi}^{n+1} = \mathcal{S}(\psi^n)$  with the boundary condition  $\psi_b^m$ 
12:    if  $n=0$  then
13:       $\lambda_{in}^0 = 0.3$ 
14:       $\alpha_{in} = 1 - \lambda_{in}^0$ 
15:    else
16:       $\lambda_{in}^{n+1} = \lambda_{in}^n + (\lambda_{in}^n - 1) \frac{(D\psi^n - D\psi^{n+1})^T D\psi^{n+1}}{\|D\psi^n - D\psi^{n+1}\|^2}$  with  $D\psi^{n+1} = \psi^n - \tilde{\psi}^{n+1}$ 
17:       $\alpha_{in} = 1 - \max(\min(\lambda_{in}^{n+1}, \lambda_{\max}), \lambda_{\min})$ 
18:    end if
19:     $\psi^{n+1} = (1 - \alpha_{in})\psi^n + \alpha_{in}\tilde{\psi}^{n+1}$ 
20:    Check convergence by  $\Delta\psi = \frac{\|\psi^{n+1} - \psi^n\|_{\infty, \Omega}}{\|\psi^0\|_{\infty, \Omega}}$ 
21:    if  $(\Delta\psi < \epsilon_{in})$  then
22:      break
23:    end if
24:  end for
25:  Given  $\psi^{n+1}$ , solve Equation (4.3) for  $U(R, Z)$ 
26:  Update coil currents  $I_i^m$  according to Equation (4.6)
27:  Determine boundary value  $\tilde{\psi}_b^m$  according to Equation (4.4)
28:  if  $m=0$  then
29:     $\lambda_{out}^0 = 0.3$ 
30:     $\alpha_{out} = 1 - \lambda_{out}^0$ 
31:  else
32:     $\lambda_{out}^{m+1} = \lambda_{out}^m + (\lambda_{out}^m - 1) \frac{(D\psi_b^m - D\psi_b^{m+1})^T D\psi_b^{m+1}}{\|D\psi_b^m - D\psi_b^{m+1}\|^2}$  with  $D\psi_b^{m+1} = \psi_b^m - \tilde{\psi}_b^{m+1}$ 
33:     $\alpha_{out} = 1 - \max(\min(\lambda_{out}^{m+1}, \lambda_{\max}), \lambda_{\min})$ 
34:  end if
35:   $\psi_b^{m+1} = (1 - \alpha_{out})\psi_b^m + \alpha_{out}\tilde{\psi}_b^{m+1}$ 
36:  Check convergence by  $\Delta\psi_b = \frac{\|\psi_b^{m+1} - \psi_b^m\|_{\infty, \partial\Omega}}{\|\psi_b^0\|_{\infty, \partial\Omega}}$ 
37:  if  $\Delta\psi_b < \epsilon_{out}$  then
38:    break
39:  end if
40: end for

```

446 There are some details associated with the main algorithm worthwhile to mention. The search algorithm
447 in parallel needs first identify candidate points through performing search on the sub-domain and then gather
448 all the candidate points into the root processor to finalize the locations of ψ_x and ψ_o . Some MPI collective
449 communications are necessary to successfully locate those points. Another important aspect is to perform
450 an efficient line integral along the boundary. The efficiency of computing the line integral can be improved
451 by precomputing certain weights associated with the Green's function and the quadratures. For instance,
452 the terms changing in Equation (4.4) in each iteration are $\frac{\partial U}{\partial n}$ and I_i , and we therefore precompute the
453 weights associated with $\frac{dl}{R}G(R, Z; R', Z')$ and $\mu_0 G(R_i^c, Z_i^c; R', Z')$ and store them in two distributed dense
454 matrices. Parallel matrix-vector operations are called to evaluate the boundary values ψ_b efficiently during
455 the iterations.

456 Finally, to verify the performance of the full free-boundary Grad-Shafranov solver, a strong scaling

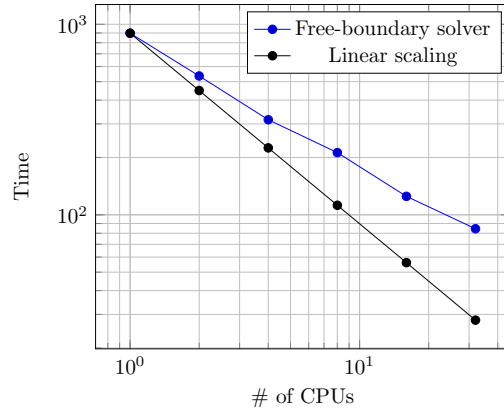


Fig. 4: Strong scaling result of the free-boundary cut-cell Grad-Shafranov solver. The computational times of one and up to 32 CPUs are presented. A good parallel scaling up to 32 processors is observed. A cut-cell mesh based on a Cartesian mesh of 512×1024 is used in the strong scaling study. Details of the problem setup can be found in the last free-boundary example in the numerical section.

457 study is performed. In the study, a cut-cell mesh based on a Cartesian mesh of size 512×1024 is used.
 458 We verify the performance of the solver using different numbers of processors up to 32. Figure 4 shows the
 459 strong scaling result and a good scaling is observed. The corresponding parallel efficiency for 2, 4, 8, 16,
 460 and 32 processors are 83.9%, 71.1%, 52.9%, 44.9%, and 33.2%. Note that since the algorithm is based on
 461 Picard iteration fundamentally, we do not expect such an algorithm would be scalable when the number
 462 of processors increase to some large number. Nevertheless, the performance of the algorithm is decent and
 463 32 processors are likely sufficient for a 2D steady-state problem. In the scaling study, the full algorithm
 464 described in Algorithm 4.1 is used and all the elliptic operators use an algebraic multigrid preconditioner
 465 provided by PETSc. Details of the problem setup can be found in the free-boundary cut-cell example in the
 466 numerical section later.

467 **6 Numerical results** The performances of the cut-cell algorithm, the free-boundary Grad-Shafranov
 468 solver and Aitken’s acceleration are demonstrated through several numerical tests. We start with the accu-
 469 racy test of the cut-cell solver through two examples: Grad-Shafranov equation with a linear source term
 470 and Grad-Shafranov equation with a nonlinear source term. Numerical examples of the free-boundary prob-
 471 lem in a rectangular domain and in the limiter-bounded domain \mathcal{L} are presented separately to show the
 472 performance of the free-boundary Grad-Shafranov solver and cut-cell algorithm. Finally, we compare the
 473 performance of Picard iteration with different under-relaxation coefficients and Aitken’s acceleration.

474 **6.1 Convergence study for the fixed-boundary cut-cell solver** In this section, two numerical
 475 examples are presented to demonstrate the accuracy of the cut-cell algorithm. Here we use two fixed-
 476 boundary problems with cut-cells to verify its accuracy. These cases are chosen for the known analytical
 477 solutions. Note that compared with a free-boundary solver, the fixed-boundary solver do not have the surface
 478 integral and the barycentric interpolation, both of which can be verified easily through standalone tests.

479 In the first example, we consider the linear Soloviev profiles from [11]. This test is used to compare the
 480 accuracy of the solution ψ solved by our fixed boundary Grad-Shafranov solver and the analytical solution.
 481 The second example considers a manufactured solution with a nonlinear source. We use it to demonstrate
 482 the accuracy of our scheme in nonlinear problems.

6.1.1 Linear case Consider a linear Grad-Shafranov equation of

$$\Delta^* \psi = -\frac{1}{R} \frac{\partial \psi}{\partial R} + \frac{\partial^2 \psi}{\partial R^2} + \frac{\partial^2 \psi}{\partial Z^2} = R^2,$$

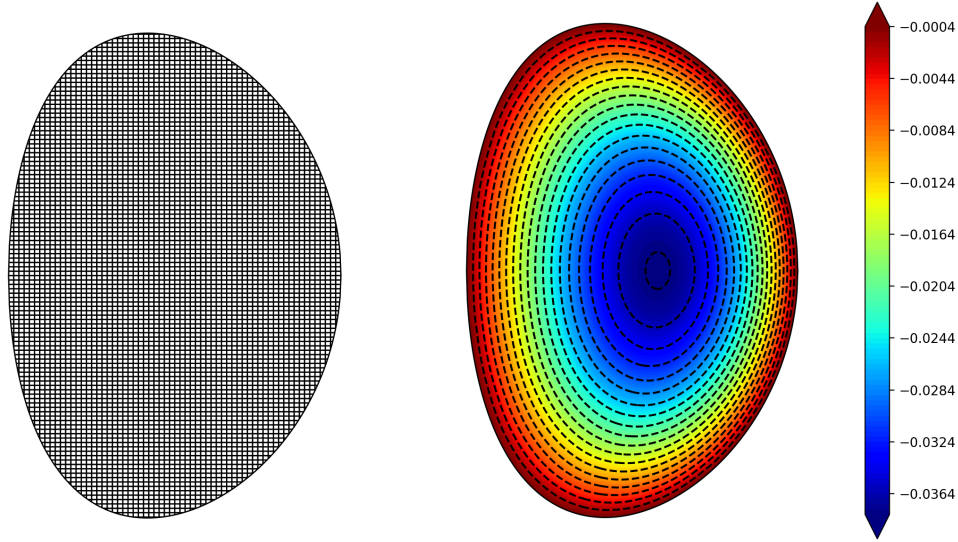


Fig. 5: Linear accuracy test. Left: cut-cell mesh on a base mesh of 121×161 (only active points are shown) and geometry. Right: corresponding numerical solution ψ .

which has an exact solution in the form of

$$\psi(R, Z) = \frac{R^4}{8} + D_1 + D_2 R^2 + D_3 (R^4 - 4R^2 Z^2),$$

483 where the parameters D_1 , D_2 , and D_3 are determined so that the contour $\phi = 0$ represents a reasonable
 484 plasma cross section.

Table 1: L_1 -errors, L_2 -errors, L_∞ -errors and corresponding convergence rates of ψ in the linear accuracy test using the cut-cell mesh. The grid points in the meshes represent all the grid points in a Cartesian grid including both active and inactive points.

$N_x \times N_y$	L_1 -Error	Order	L_2 -Error	Order	L_∞ -Error	Order
31×41	2.310e-01		1.276e-02		1.519e-03	
61×81	5.925e-02	1.96	3.361e-03	1.92	5.513e-04	1.46
121×161	1.467e-02	2.01	8.327e-04	2.01	9.177e-05	2.59
241×321	3.774e-03	1.96	2.155e-04	1.95	2.229e-05	2.04
481×641	1.086e-03	1.80	6.198e-05	1.80	5.422e-06	2.04

485 Ref. [42] introduced three characteristic quantities describing the shape of the cross section in a magnetic
 486 confinement device: the inverse aspect ratio ε , the elongation κ , and the triangularity δ . We adopt the same
 487 manufactured solutions in the accuracy test. The parameters are determined by the following linear system

$$488 \begin{bmatrix} 1 & (1 + \varepsilon)^2 & & (1 + \varepsilon)^4 \\ 1 & (1 - \varepsilon)^2 & & (1 - \varepsilon)^4 \\ 1 & (1 - \delta\varepsilon)^2 & (1 - \delta\varepsilon)^4 - 4(1 - \delta\varepsilon)^2 \kappa^2 \varepsilon^2 & \end{bmatrix} \begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix} = -\frac{1}{8} \begin{bmatrix} (1 + \varepsilon)^4 \\ (1 - \varepsilon)^4 \\ (1 - \delta\varepsilon)^4 \end{bmatrix},$$

489

490 of which the equations correspond to the boundary conditions of $\psi(1 + \varepsilon, 0) = 0$, $\psi(1 - \varepsilon, 0) = 0$, and
 491 $\psi(1 - \delta\varepsilon, \kappa\varepsilon) = 0$, respectively. The test is taken with the ITER-like configuration of $\varepsilon = 0.32$, $\kappa = 1.7$,
 492 $\delta = 0.33$. In particular, the computational boundaries are described by Chebyshev nodes along the R
 493 direction and Z coordinates are then determined by solving $\psi = 0$. The cut-cell meshes are then accordingly
 494 generated.

495 Numerical errors and corresponding convergence rates are reported in [Table 1](#). The numerical solution
 496 and an example cut-cell mesh are presented in [Figure 5](#). We observed a second-order accuracy for all the
 497 error norms.

6.1.2 Nonlinear case We consider the same ITER geometry as for the linear test in the previous section but with a nonlinear source term. A manufactured solution

$$\psi(R, Z) = \sin(K_R(R + R_0)) \cos(K_Z Z),$$

can be constructed for the nonlinear Grad-Shafranov equation of

$$\Delta^* \psi = -F(R, Z, \psi),$$

498 where the source term $F(R, Z, \psi)$ is given by

$$499 \quad F(R, Z, \psi) = (K_R^2 + K_Z^2)\psi + \frac{K_R}{R} \cos(K_R(R + R_0)) \cos(K_Z Z) + R \left[\sin^2(K_R(R + R_0)) \cos^2(K_Z Z) \right. \\
 500 \quad \left. - \psi^2 + \exp(-\sin(K_R(R + R_0)) \cos(K_Z Z)) - \exp(-\psi) \right], \\
 501$$

502 and the coefficients are $K_R = 1.15\pi$, $K_Z = 1.15$, and $R_0 = -0.5$.

Table 2: L_1 -errors, L_2 -errors, L_∞ -errors and corresponding convergence rates of ψ in the nonlinear accuracy test using cut-cell algorithm. The grid points in the meshes represent all the grid points in a Cartesian grid including both active and inactive points.

$N_x \times N_y$	L_1 -Error	Order	L_2 -Error	Order	L_∞ -Error	Order
31×41	1.486e+00		8.493e-02		1.359e-02	
61×81	4.476e-01	1.73	2.584e-02	1.72	4.480e-03	1.60
121×161	1.340e-01	1.74	7.688e-03	1.75	8.288e-04	2.43
241×321	3.371e-02	1.99	1.912e-03	2.01	2.154e-04	1.94
481×641	7.825e-03	2.11	4.429e-04	2.11	4.148e-05	2.38

503 A nonlinear solver (Picard iteration accelerated with Aitken’s) is used in this example. Numerical errors
 504 and corresponding convergence rates are reported in [Table 2](#). The solution and an example cut-cell mesh
 505 are presented in [Figure 6](#). Again, as expected, we observe a second-order accuracy of space discretization.

506 **6.2 Examples of the free-boundary solver** In this section, numerical solutions from the free-boundary
 507 Grad-Shafranov solver are presented. One particular interest will be the performance of the solver in keep-
 508 ing the targeted shape of $\mathcal{P}(\psi)$, i.e., the shape control. Numerical tests are scattered into the following two
 509 examples, a rectangular domain and a limiter-bounded domain.

510 The problem is based on prescribed numerical $p(\bar{\psi})$ and $g(\bar{\psi})$ profiles for a proposed ITER discharge
 511 at 15 MA toroidal plasma current [39], which carries the ITER reference number “ABT4ZL”. In addition
 512 to the numerical $p(\bar{\psi})$ and $g(\bar{\psi})$ profiles, there is also the numerical solution for $\psi(R, Z)$ from a different
 513 Grad-Shafranov solver [39], which we will use for the initial data. The coil currents corresponding to the
 514 equilibrium were not given and they will be found by our free-boundary Grad-Shafranov solver in this test.
 515 The ITER poloidal field coil locations, however, are fixed and given in [Table 3](#). We will solve or more
 516 accurately, resolve this free-boundary Grad-Shafranov equilibrium problem in two ways. One is based on a
 517 Cartesian mesh over a rectangular domain and the other one is based on a cut-cell mesh that has the first
 518 wall as the computational boundary.

519 **6.2.1 Rectangular domain** The first example uses a rectangular computational domain that contains
 520 the limiter regions but excludes the current regions. This is a conventional approach implemented in many
 521 practical Grad-Shafranov solvers. We present this case to verify our implementation of the full solver as well
 522 as to compare the results with the solutions from the cut-cell solver, which will be discussed in the next
 523 section.

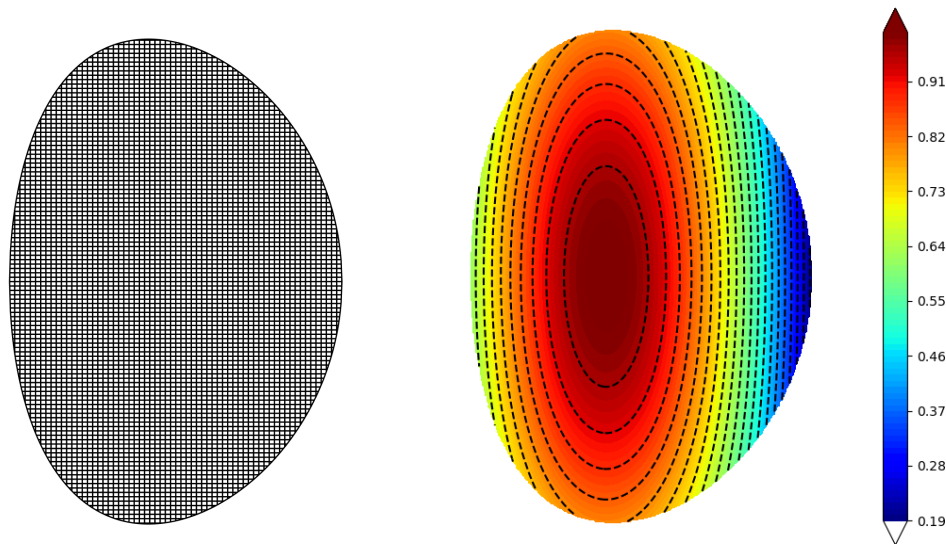


Fig. 6: Nonlinear accuracy test. Left: cut-cell mesh on a base mesh of 121×161 (only active points are shown) and geometry. Right: corresponding numerical solution ψ .

Table 3: Coil Information for the free-boundary problem. The data is based on the ITER configurations. The total current in the coil is set to current timing the value of turns. The type “coil” refers to a point source. The type “solenoid” refers to a coil with a positive length for which the range of length is also given. The units is meter.

Coils	Type	R	Z	Turns
PF1	Coil	3.9431	7.5741	248.6
PF2	Coil	8.2851	6.5398	115.2
PF3	Coil	11.9919	3.2752	185.9
PF4	Coil	11.9630	-2.2336	169.9
PF5	Coil	8.3908	-6.7269	216.8
PF6	Coil	4.3340	-7.4665	459.4

Coils	Type	R	(Z_{min}, Z_{max})	Turns
CS1	Solenoid	1.696	(-5.415, -3.6067)	553
CS2	Solenoid	1.696	(-3.6067, -1.7983)	553
CS3	Solenoid	1.696	(-1.7983, 1.8183)	1106
CS4	Solenoid	1.696	(1.8183, 3.6267)	553
CS5	Solenoid	1.696	(3.6267, 5.435)	553

524 The computational domain is $\Omega = \{(R, Z) \in \mathcal{H} \mid 3.55 \leq R \leq 8.88, -3.84 \leq Z \leq 4.92\}$ and the grid used
525 in this case is a Cartesian grid of size 196×375 . The proposed free-boundary Grad-Shafranov solver is used to
526 solve the problem on the rectangular domain Ω . The converged magnetic flux from the free-boundary Grad-
527 Shafranov solver and magnetic flux from the equilibrium data file are presented in Figure 8. Black dots in
528 Figure 8a represent the shape control points on targeted magnetic separatrix of the fixed plasma shape from
529 the initial data. Twenty one points are selected along $\psi = \psi_X$ in the initial data as the control points. We can
530 observe from Figure 8a that the converged magnetic flux from free-boundary Grad-Shafranov solver keeps
531 the predetermined plasma shape very well. The solution is comparable to the initial data and its difference
532 is presented in Figure 8c. Note that the largest difference is around the corner points. This is expected
533 since the computational domain is very close to two current coils at $(8.3908, -6.7269)$ and $(8.2851, 6.5398)$.
534 The magnetic axes ψ_o are presented in Figure 8a and Figure 8b as red cross points, which are clearly local
535 minima in the solutions. There are two candidate of ψ_X points shown in Figure 8a and Figure 8b, using
536 blue cross points. It is clear that those are saddle points of the solutions. Based on the criteria we discussed
537 in the algorithm section, the point in the bottom portion of the domain is chosen as ψ_X . It is seen that the
538 selected ψ_X point in the solution is very close to the control points. Both ψ_o and ψ_X points are found to
539 change slightly throughout the iterations.

540 Another interesting question to investigate is whether the converged magnetic flux from the free-

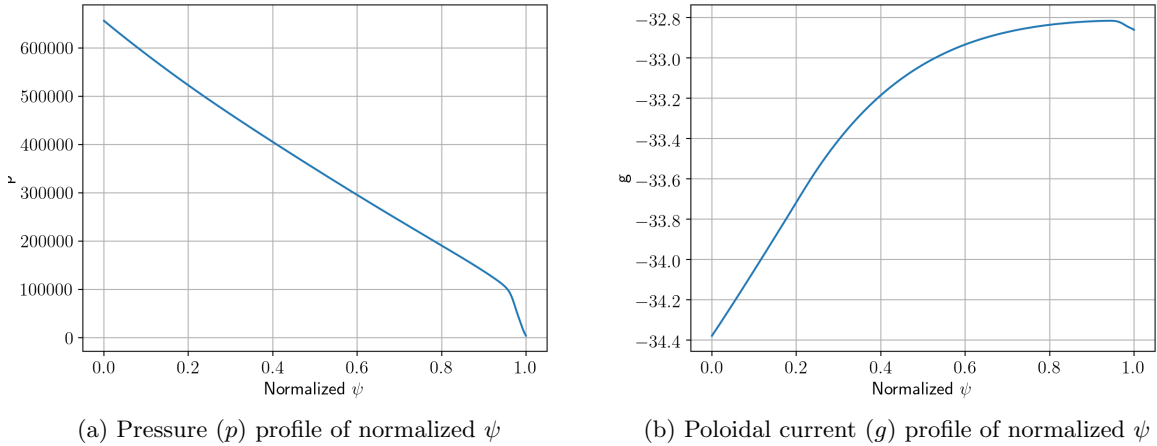


Fig. 7: Numerical $p(\bar{\psi})$ and $g(\bar{\psi})$ profiles from the equilibrium data generated in Ref. [39].

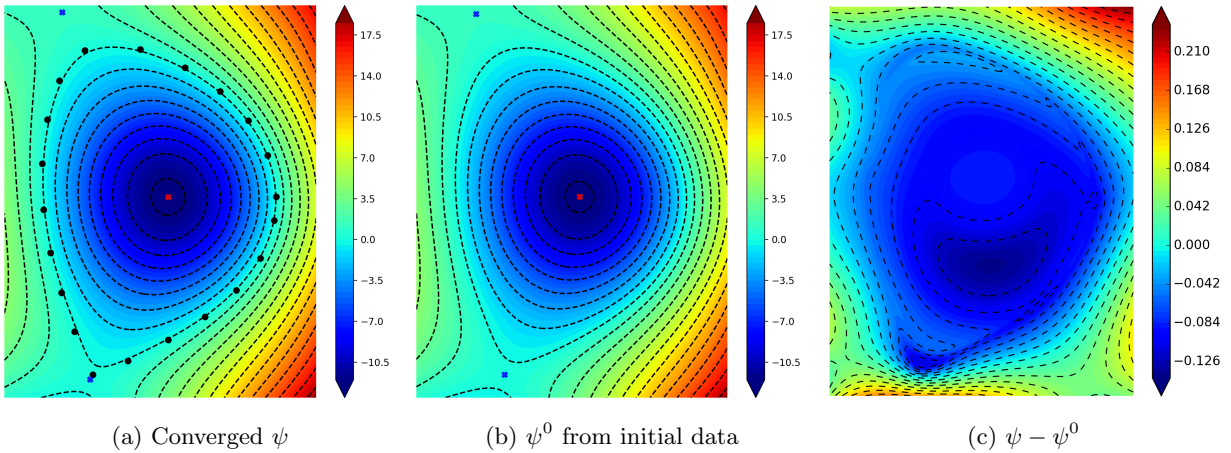


Fig. 8: Rectangular geometry: (a) converged magnetic flux from free-boundary Grad-Shafranov solver, (b) magnetic flux from initial equilibrium data file, (c) the difference between magnetic flux from free-boundary Grad-Shafranov solver and initial equilibrium data.

541 boundary Grad-Shafranov solver can keep the predetermined plasma shape if we modify the source term of the
 542 Grad-Shafranov equation. This is a test to verify the effectiveness of the shape control techniques we
 543 implemented in the solver. As an example, we drop the pressure to 80% of the original pressure profile and
 544 solve the same free-boundary problem using the solver. Its results are presented in Figure 9. The converged
 545 magnetic flux from the free-boundary Grad-Shafranov solver and magnetic flux from the equilibrium data
 546 file are also compared in Figure 9. It indicates that the converged magnetic flux still keeps the predetermined
 547 plasma shape. It also suggests that the free-boundary Grad-Shafranov solver performs well in keeping the
 548 targeted shape of $\mathcal{P}(\psi)$ in the rectangular domain.

549 **6.2.2 Limiter-bounded domain \mathcal{L}** In the second example, by combining cut-cell algorithm, we directly
 550 choose the irregular limiter-bounded domain \mathcal{L} as Ω to solve the free boundary problem. The domain \mathcal{L}
 551 consists of a bunch of straight lines due to the design of ITER, which will be used to construct the cut-cell

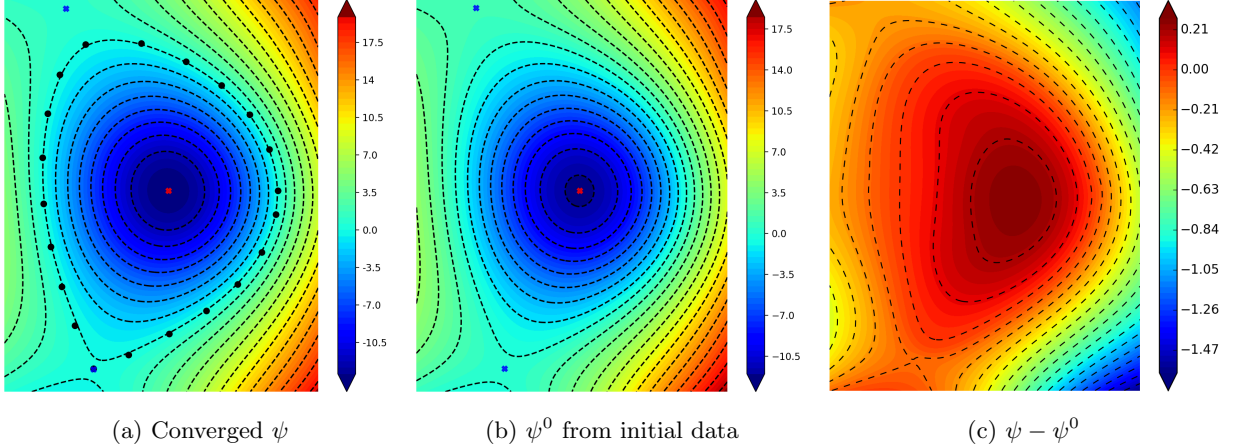


Fig. 9: Rectangular geometry: (a) converged magnetic flux from free-boundary Grad-Shafranov solver, (b) magnetic flux from initial equilibrium data file, (c) the difference between magnetic flux from free-boundary Grad-Shafranov solver and the initial data. The pressure is dropped to 80% of the given profile.

552 mesh. This example is used to verify the full algorithm we proposed in this work.

553 We create a Cartesian mesh of size 216×493 in the domain $\{(R, Z) \in \mathcal{H} \mid 3.0 \leq R \leq 8.88, -5.53 \leq$
 554 $Z \leq 6.0\}$, which contains the irregular limiter-bounded domain \mathcal{L} . The cut-cell mesh is created accordingly,
 555 and the total number of active cell is 44668, which includes 1266 cut-cells. The proposed free-boundary
 556 Grad-Shafranov solver combining the cut-cell algorithm is used to solve the problem on the limiter-bounded
 557 domain \mathcal{L} . The converged magnetic flux from the solver and magnetic flux from the equilibrium data file are
 558 presented in Figure 10. Black dots in Figure 10a represent the shape control points on targeted magnetic
 559 separatrix of the fixed plasma shape from the initial data. The same 21 points in subsection 6.2.1 are
 560 selected along $\psi = \psi_X$ in the initial data as the control points. We can observe from Figure 10a that the
 561 converged magnetic flux from free-boundary Grad-Shafranov solver keeps the predetermined plasma shape
 562 very well. The solution is comparable to the initial data and its difference is presented in Figure 10c. Note
 563 that the largest difference is around the bottom corner, which is due to the non-smooth boundary in the
 564 computational domain. The magnetic axes ψ_o are presented in Figure 10a and Figure 10b as red cross points,
 565 which are clearly local minima in the solutions. Different from the example in the rectangular domain, only
 566 one ψ_X point is shown in Figure 10a and Figure 10b, using a green cross point. This is caused by the choice
 567 of the limiter-bounded domain \mathcal{L} , as the initial plasma domain is bounded by the last closed poloidal flux
 568 line inside the limiter-bounded domain. It is clear that the ψ_X point is the saddle point of the solutions.
 569 Both ψ_o and ψ_X are found to change slightly throughout the iterations.

570 In Figure 10d, the converged magnetic fluxes of the free-boundary problem on the regular domain and
 571 the limiter-bounded domain are compared to each other. It is found that the difference is small, which
 572 indicates both solvers produce the same equilibrium. When solving on the rectangular domain, the outer
 573 iteration is 12, and the average inner iteration is 32, while on the limiter-bounded domain the outer iteration
 574 is 11, and the average inner iteration is 9.

575 In this example, we again investigate whether the converged magnetic flux from the solver can keep the
 576 predetermined plasma shape in the limiter-bounded domain \mathcal{L} when we modify the source term of the Grad-
 577 Shafranov equation. Similarly, we drop the pressure to 80% of the original pressure profile and solve the
 578 same free-boundary problem using the solver combining with the cut-cell algorithm. Its results are presented
 579 in Figure 11. The converged magnetic flux from the free-boundary Grad-Shafranov solver and the magnetic
 580 flux from the equilibrium data file are also presented in Figure 11. It indicates that the converged magnetic
 581 flux still keeps the predetermined plasma shape. It also suggests that the free-boundary Grad-Shafranov
 582 solver performs well in keeping the targeted shape of $\mathcal{P}(\psi)$ in the limiter-bounded domain \mathcal{L} .

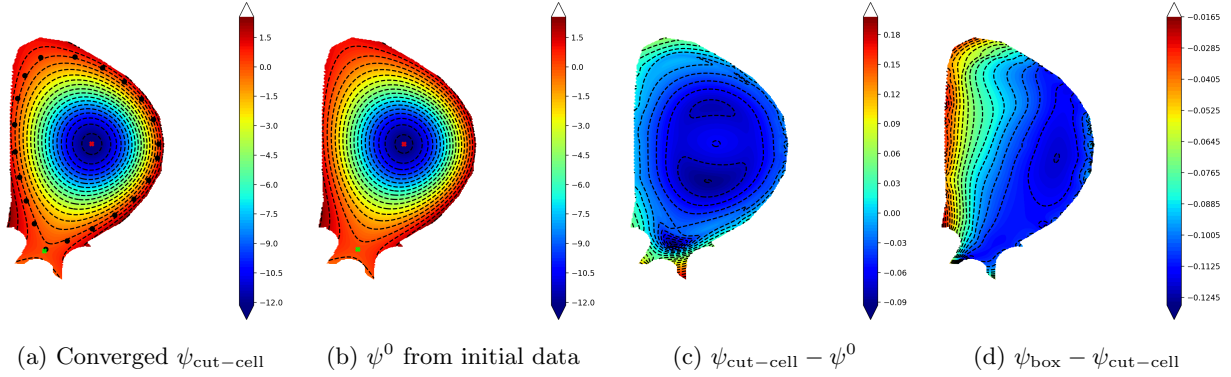


Fig. 10: Limiter geometry: (a) converged magnetic flux from the free-boundary Grad-Shafranov solver with the cut-cell algorithm, (b) magnetic flux from the initial equilibrium data file, (c) difference between magnetic flux from the free-boundary Grad-Shafranov solver with the cut-cell algorithm and the initial equilibrium data file, (d) difference between the solutions of the rectangular domain and the limiter-bounded domain.

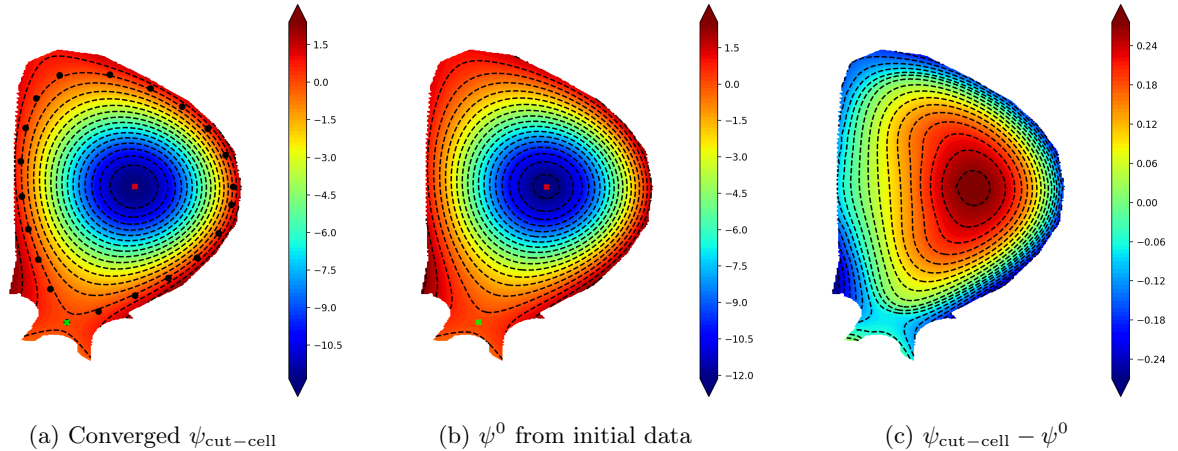


Fig. 11: Limiter geometry: (a) converged magnetic flux from the free-boundary Grad-Shafranov solver with the cut-cell algorithm, (b) magnetic flux from the initial equilibrium data file, (c) difference between magnetic flux from the free-boundary Grad-Shafranov solver with the cut-cell algorithm and the initial equilibrium data file. The pressure is dropped to 80% of the given profile.

583 **6.3 Performance of Aitken's acceleration** In the final examples, we focus on the demonstration of
 584 the efficiency of Aitken's acceleration. Results from Aitken's acceleration are compared to results from Picard
 585 iteration with fixed under-relaxation coefficients α in the free-boundary Grad-Shafranov solver. As presented
 586 in [Algorithm 4.1](#), both inner and outer loops use under-relaxations, either through Aitken's acceleration or
 587 a fixed under-relaxed parameter. We consider both the rectangular geometry and the geometry with the
 588 limiter-bounded domain in the previous test. Aitken's acceleration is found to converge much faster than
 589 Picard iterations with fixed under-relaxation coefficients in the both inner and outer loops.

590 **6.3.1 Rectangular domain** In this example, we test the efficiency of Aitken's acceleration when com-
 591 pared with Picard iteration with fixed under-relaxation coefficients α on the rectangular domain with mesh
 592 size of 196×375 . Here we set the range $[\lambda_{\min}, \lambda_{\max}] = [0, 0.95]$.

593 [Figure 12](#) shows the convergence histories of Picard iteration with different under-relaxation coefficients

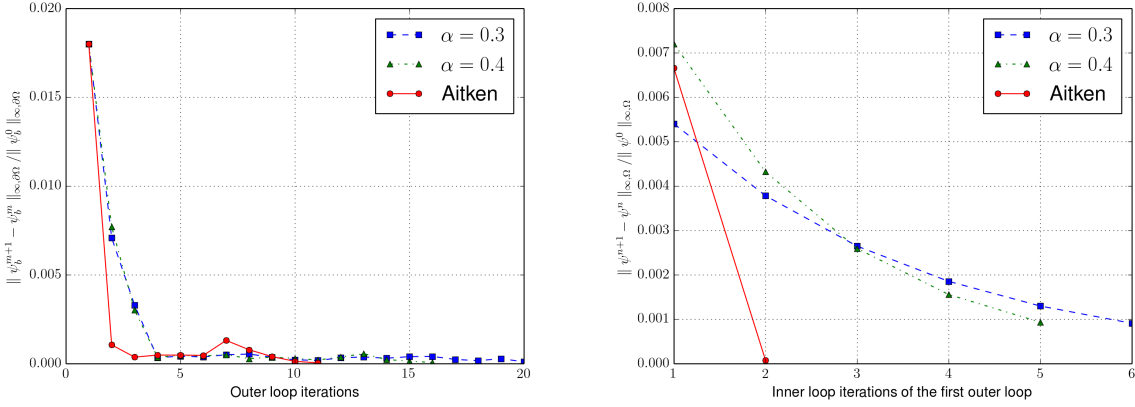


Fig. 12: Comparison of convergence history of the free-boundary solver algorithm with Aitken’s acceleration and Picard iterations. Left: convergence history of the outer loop. Right: convergence history of the first inner loop. The convergence histories of two fixed under-relaxation parameter, $\alpha = 0.3$ and $\alpha = 0.4$, are also presented. The problem is solved on a rectangular computational domain with mesh size of 196×375 . Note that Aitken’s acceleration significantly improve the convergence of Picard iterations with a predetermined under-relaxation.

594 α and Aitken’s acceleration. The results of two parameters, $\alpha = 0.3$ and $\alpha = 0.4$, are presented for
 595 comparison. The left figure shows that the free-boundary Grad-Shafranov solver algorithm with under-
 596 relaxed Picard iterations need to take 16 or 20 outer loops to satisfy the desired convergent threshold for
 597 the boundary value ψ_b , while Aitken’s acceleration only needs 11 iterations. On the other hand, the right
 598 figure shows the convergence histories in the first inner loop. To have a fair comparison, the first inner loop
 599 is chosen since all the first inner loops use the same initial guess. It is found that the Aitken’s acceleration
 600 only needs 2 iterations to satisfy the the desired convergence tolerance for the magnetic flux ψ , while the
 601 under-relaxed Picard iterations need to take 2 to 3 times more iterations. Therefore, the performance of
 602 Aitken’s acceleration in the inner loops is also better than that of Picard iterations.

603 **6.3.2 Limiter-bounded domain \mathcal{L}** The final example focuses on the efficiency of Aitken’s acceleration
 604 on the limiter-bounded domain \mathcal{L} with a cut-cell mesh of size 216×493 (base Cartesian mesh). Here we set
 605 the range $[\lambda_{min}, \lambda_{max}] = [0, 0.7]$.

606 Figure 13 shows the convergence histories of Picard iterations with different under-relaxation coefficients
 607 α and Aitken’s acceleration. The results of two parameters, $\alpha = 0.7$ and $\alpha = 0.5$ are presented for comparison.
 608 We set the the maximum iteration to be 40 for the outer loop. The under-relaxed Picard iterations fail to
 609 reach the desired tolerance before reaching the maximum iteration, while the Aitken’s acceleration only
 610 needs 6 iterations to reach the same tolerance. The left figure presents the maximum relative difference of
 611 ψ_b in the first 20 iterations. The right figure shows the first inner loop, for which the Aitken’s acceleration
 612 only needs 6 iterations, while the under-relaxed Picard iterations take 12 or 21 iterations.

613 In conclusion, we find that the Aitken’s acceleration could significantly accelerate Picard iterations in
 614 both inner and outer loops. Other acceleration techniques may also show similar performance, which will be
 615 studied in the future work.

616 **7 Conclusions** This work discusses the development of a parallel free-boundary Grad-Shafranov solver.
 617 As traditionally done, the free-boundary problem is reformulated in a bounded computational domain that
 618 encloses all current-carrying plasmas. The coil current contribution to the plasma equilibrium enters through
 619 the value of the magnetic flux that is evaluated by Green’s function methods. The main focus here is to solve
 620 the free boundary problem of the nonlinear Grad-Shafranov equation in an irregular computational domain
 621 by combining the cut-cell algorithm with a regular mesh. A key application of free-boundary Grad-Shafranov

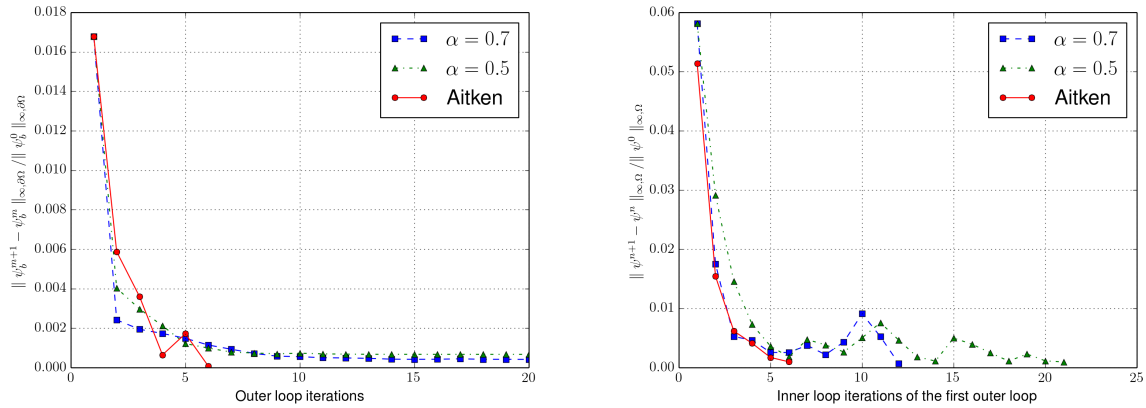


Fig. 13: Comparison of convergence history of the free-boundary solver algorithm with Aitken's acceleration and Picard iterations. Left: convergence history of the outer loop. Right: convergence history of the first inner loop. The convergence histories of two fixed under-relaxation parameter, $\alpha = 0.7$ and $\alpha = 0.5$, are also presented. The problem is solved on the limiter-bounded domain \mathcal{L} with a cut-cell mesh of size 216×493 (base Cartesian mesh). Note that Aitken's acceleration significantly improve the convergence of Picard iterations with a predetermined under-relaxation.

622 solvers is to find the required coil currents for precision plasma positioning and shaping. This is cast into
 623 a proper optimization problem that determines the coil current with targeted plasma shape and for which
 624 a cut-cell algorithm is described. To accelerate Picard iterations commonly used in previous works, we also
 625 propose an improvement based on Aitken's acceleration. This is applied to both the inner and outer loops
 626 of the algorithm. The proposed algorithm is implemented in parallel under the PETSc framework. Strong
 627 scaling study of the free-boundary Grad-Shafranov solver with cut-cell algorithm demonstrates its parallel
 628 performance.

629 A series of numerical tests is presented to demonstrate the accuracy of the cut-cell algorithm and the effi-
 630 ciency of the free-boundary Grad-Shafranov solver. In particular, a refinement study based on manufactured
 631 solutions confirms a second-order accuracy of the cut-cell implementation and tokamak examples, including
 632 that of ITER, are presented to verify the effectiveness and efficiency fo the full algorithm. Moreover, numer-
 633 ical results demonstrate that the converged magnetic flux keeps the predetermined plasma domain shape
 634 even with different pressure profile in the source term of the Grad-Shafranov equation both in the rectan-
 635 gular domain and the irregular limiter-bounded domain. It is also found that the Aitken's acceleration we
 636 employed in the algorithm is more efficient than Picard iteration with fixed under-relaxation. In conclusion,
 637 numerical results shows a good performance of the free-boundary cut-cell Grad-Shafranov solver.

638 **Acknowledgments** The authors wish to thank Dr. Yueqiang Liu of General Atomics for sharing the Grad-
 639 Shafranov equilibrium of ITER reference number "ABT4ZL". This research used resources of the National
 640 Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User
 641 Facility operated.

642

REFERENCES

- 643 [1] A. AMBROSETTI, M. CALAHORRANO, AND F. DOBARRO, *Remarks on the grad-shafranov equation*, Applied Mathematics
 644 Letters, 3 (1990), pp. 9–11.
 645 [2] M. ARIOLA, A. PIRONTI, AND A. PORTONE, *Vertical stabilization and plasma shape control in the iter-feat tokamak*, in
 646 Proceedings of the 2000 IEEE International Conference on Control Applications, Los Alamitos, CA, USA, sep 2000,
 647 IEEE Computer Society, pp. 401,402,403,404,405.
 648 [3] S. BALAY, S. ABHYANKAR, M. ADAMS, J. BROWN, P. BRUNE, K. BUSCHELMAN, L. DALCIN, A. DENER, V. EIJKHOUT,
 649 W. GROPP, ET AL., *Petsc users manual*, (2019).
 650 [4] J. W. BANKS, W. D. HENSHAW, D. W. SCHWENDEMAN, AND Q. TANG, *A stable partitioned fsi algorithm for rigid bodies*

- 651 *and incompressible flow. part ii: General formulation*, Journal of Computational Physics, 343 (2017), pp. 469–500.
- 652 [5] J. W. BANKS, W. D. HENSHAW, D. W. SCHWENDEMAN, AND Q. TANG, *A stable partitioned fsi algorithm for rigid bodies*
- 653 *and incompressible flow in three dimensions*, Journal of Computational Physics, 373 (2018), pp. 455–492.
- 654 [6] M. BERGER, *Cut cells: Meshes and solvers*, in Handbook of Numerical Analysis, vol. 18, Elsevier, 2017, pp. 1–22.
- 655 [7] J.-P. BERRUT, *Rational functions for guaranteed and experimentally well-conditioned global interpolation*, Computers &
- 656 *Mathematics with Applications*, 15 (1988), pp. 1–16.
- 657 [8] J.-P. BERRUT, R. BALTENSPERGER, AND H. D. MITTELMANN, *Recent developments in barycentric rational interpolation*,
- 658 *Journal of Computational and Applied Mathematics*, 259 (2005), pp. 95–107.
- 659 [9] I. BORAZJANI, L. GE, AND F. SOTIROPOULOS, *Curvilinear immersed boundary method for simulating fluid structure*
- 660 *interaction with complex 3d rigid bodies*, Journal of Computational physics, 227 (2008), pp. 7587–7620.
- 661 [10] D. L. BROWN, W. D. HENSHAW, AND D. J. QUINLAN, *Overture: An object-oriented framework for solving partial differential*
- 662 *equations*, in International Conference on Computing in Object-Oriented Parallel Environments, Springer, 1997,
- 663 pp. 177–184.
- 664 [11] A. J. CERFON AND J. P. FREIDBERG, *“One size fits all” analytic solutions to the Grad–Shafranov equation*, Physics of
- 665 *Plasmas*, 17 (2010), p. 032502.
- 666 [12] S. CHEN, B. MERRIMAN, S. OSHER, AND P. SMERKA, *A simple level set method for solving stefan problems*, Journal of
- 667 *Computational Physics*, 135 (1997), pp. 8–29.
- 668 [13] A. J. CREEELY, M. J. GREENWALD, S. B. BALLINGER, D. BRUNNER, J. CANIK, J. DOODY, T. FÜLÖP, D. T. GARNIER,
- 669 R. GRANETZ, T. K. GRAY, AND ET AL., *Overview of the sparx tokamak*, Journal of Plasma Physics, 86 (2020),
- 670 p. 865860502.
- 671 [14] D. DEVENDRAN, D. GRAVES, AND H. JOHANSEN, *A higher-order finite-volume discretization method for poisson’s equation*
- 672 *in cut cell geometries*, arXiv preprint arXiv:1411.4283, (2014).
- 673 [15] D. DEVENDRAN, D. GRAVES, H. JOHANSEN, AND T. LIGOCKI, *A fourth-order cartesian grid embedded boundary method*
- 674 *for poisson’s equation*, Communications in Applied Mathematics and Computational Science, 12 (2017), pp. 51–79.
- 675 [16] B. FAUGERAS AND H. HEUMANN, *Fem-bem coupling methods for tokamak plasma axisymmetric free-boundary equilibrium*
- 676 *computations in unbounded domains*, Journal of Computational Physics, 343 (2017), pp. 201–216.
- 677 [17] M. S. FLOATER AND K. HORMANN, *Barycentric rational interpolation with no poles and high rates of approximation*,
- 678 *Numerische Mathematik*, 107 (2007), pp. 315–331.
- 679 [18] F. GIBOU AND R. FEDKIW, *A fourth order accurate discretization for the laplace and heat equations on arbitrary domains,*
- 680 *with applications to the stefan problem*, Journal of Computational Physics, 202 (2005), pp. 577–601.
- 681 [19] F. GIBOU, R. P. FEDKIW, L.-T. CHENG, AND M. KANG, *A second-order-accurate symmetric discretization of the poisson*
- 682 *equation on irregular domains*, Journal of Computational Physics, 176 (2002), pp. 205–227.
- 683 [20] G. H. GOLUB, M. HEATH, AND G. WAHBA, *Generalized cross-validation as a method for choosing a good ridge parameter*,
- 684 *Technometrics*, 21 (1979), pp. 215–223.
- 685 [21] H. HEUMANN, J. BLUM, C. BOULBE, B. FAUGERAS, G. SELIG, P. HERTOUT, E. NARDON, J.-M. ANÉ, S. BRÉMOND, AND
- 686 V. GRANDGIRARD, *Quasi-static free-boundary equilibrium of toroidal plasma with cedres++: Computational methods*
- 687 *and applications*, Journal of Plasma Physics, (2015), p. 35.
- 688 [22] H. HEUMANN AND F. RAPETTI, *A finite element method with overlapping meshes for free-boundary axisymmetric plasma*
- 689 *equilibria in realistic geometries*, Journal of Computational Physics, 334 (2017), pp. 522–540.
- 690 [23] M. HONDA, *Simulation technique of free-boundary equilibrium evolution in plasma ramp-up phase*, Computer Physics
- 691 *Communications*, 181 (2010), pp. 1490–1500.
- 692 [24] E. HOWELL AND C. R. SOVINEC, *Solving the grad-shafranov equation with spectral elements*, Computer Physics Commu-
- 693 *nications*, 185 (2014), pp. 1415–1421.
- 694 [25] G. HUYSMANS, J. GOEDBLOED, W. KERNER, ET AL., *Isoparametric bicubic hermite elements for solution of the grad-*
- 695 *shafranov equation*, International Journal of Modern Physics C, 2 (1991), pp. 371–376.
- 696 [26] S. JARDIN, *Computational methods in plasma physics*, CRC Press, 2010.
- 697 [27] S. C. JARDIN, N. POMPHREY, AND J. DELUCIA, *Dynamic modeling of transport and positional control of tokamaks*, Journal
- 698 *of computational Physics*, 66 (1986), pp. 481–507.
- 699 [28] Y. M. JEON, *Development of a free-boundary tokamak equilibrium solver for advanced study of tokamak equilibria*, Journal
- 700 *of the Korean Physical Society*, 67 (2015), pp. 843–853.
- 701 [29] H. JOHANSEN AND P. COLELLA, *A cartesian grid embedded boundary method for poisson’s equation on irregular domains*,
- 702 *Journal of Computational Physics*, 147 (1998), pp. 60–85.
- 703 [30] J. L. JOHNSON, H. DALHED, J. GREENE, R. GRIMM, Y. HSIEH, S. JARDIN, J. MANICKAM, M. OKABAYASHI, R. STORER,
- 704 A. TODD, ET AL., *Numerical determination of axisymmetric toroidal magnetohydrodynamic equilibria*, Journal of
- 705 *Computational Physics*, 32 (1979), pp. 212–234.
- 706 [31] Z. JOMAA AND C. MACASKILL, *The embedded finite difference method for the poisson equation in a domain with an*
- 707 *irregular boundary and dirichlet boundary conditions*, Journal of Computational Physics, 202 (2005), pp. 488–506.
- 708 [32] K. LACKNER, *Computation of ideal mhd equilibria*, Computer Physics Communications, 12 (1976), pp. 33–44.
- 709 [33] J. LEE AND A. CERFON, *Ecom: A fast and accurate solver for toroidal axisymmetric mhd equilibria*, Computer Physics
- 710 *Communications*, 190 (2015), pp. 72–88.
- 711 [34] R. J. LEVEQUE, *Numerical methods for conservation laws*, vol. 3, Springer, 1992.
- 712 [35] R. J. LEVEQUE AND Z. LI, *The immersed interface method for elliptic equations with discontinuous coefficients and*
- 713 *singular sources*, SIAM Journal on Numerical Analysis, 31 (1994), pp. 1019–1044.
- 714 [36] H. LI AND P. ZHU, *Solving the grad-shafranov equation using spectral elements for tokamak equilibrium with toroidal*
- 715 *rotation*, Computer Physics Communications, (2020), p. 107264.
- 716 [37] S. LIU, Y. DU, AND X. LIU, *Numerical studies of a class of reaction-diffusion equations with stefan conditions*, Interna-
- 717 *tional Journal of Computer Mathematics*, 97 (2020), pp. 959–979.
- 718 [38] S. LIU AND X. LIU, *Numerical methods for a two-species competition-diffusion model with free boundaries*, Mathematics,

- 6 (2018), p. 72.
- [39] Y. LIU, R. AKERS, I. CHAPMAN, Y. GRIBOV, G. HAO, G. HUIJSMANS, A. KIRK, A. LOARTE, S. PINCHES, M. REINKE, ET AL., *Modelling toroidal rotation damping in iter due to external 3d fields*, Nuclear Fusion, 55 (2015), p. 063027.
- [40] H. LÜTJENS, A. BONDESON, AND O. SAUTER, *The chease code for toroidal mhd equilibria*, Computer physics communications, 97 (1996), pp. 219–260.
- [41] D. MOK, W. WALL, AND E. RAMM, *Accelerated iterative substructuring schemes for instationary fluid-structure interaction*, Computational fluid and solid mechanics, 2 (2001), pp. 1325–1328.
- [42] A. PATAKI, A. J. CERFON, J. P. FREIDBERG, L. GREENGARD, AND M. O’NEIL, *A fast, high-order solver for the grad-shafranov equation*, Journal of Computational Physics, 243 (2013), pp. 28–45.
- [43] Z. PENG, Q. TANG, AND X.-Z. TANG, *An adaptive discontinuous petrov-galerkin method for the grad-shafranov equation*, SIAM Journal on Scientific Computing, 42 (2020), pp. B1227–B1249.
- [44] A. D. POLYANIN AND A. V. MANZHIROV, *Handbook of mathematics for engineers and scientists*, CRC Press, 2006.
- [45] T. SÁNCHEZ-VIZUET AND M. E. SOLANO, *A hybridizable discontinuous galerkin solver for the grad-shafranov equation*, Computer Physics Communications, 235 (2019), pp. 120–132.
- [46] T. SÁNCHEZ-VIZUET, M. E. SOLANO, AND A. J. CERFON, *Adaptive hybridizable discontinuous galerkin discretization of the grad-shafranov equation by extension from polygonal subdomains*, Computer Physics Communications, (2020), p. 107239.
- [47] C. SCHNEIDER AND W. WERNER, *Some new aspects of rational interpolation*, Mathematics of computation, 47 (1986), pp. 285–299.
- [48] P. SCHWARTZ, M. BARAD, P. COLELLA, AND T. LIGOCKI, *A cartesian grid embedded boundary method for the heat equation and poisson’s equation in three dimensions*, Journal of Computational Physics, 211 (2006), pp. 531–550.
- [49] K. VON HAGENOW AND K. LACKNER, eds., *Proceedings of the 7th Conf. on the Numerical Simulation of Plasmas*, 1975.