SIAM J. SCI. COMPUT.
Vol. 42, No. 5, pp. B1227–B1249

AN ADAPTIVE DISCONTINUOUS PETROV–GALERKIN METHOD FOR THE GRAD–SHAFRANOV EQUATION*

ZHICHAO PENG†, QI TANG‡, AND XIAN-ZHU TANG‡

Abstract. In this work, we propose and develop an arbitrary-order adaptive discontinuous Petrov–Galerkin (DPG) method for the nonlinear Grad–Shafranov equation. An ultraweak formulation of the DPG scheme for the equation is given based on a minimal residual method. The DPG scheme has the advantage of providing more accurate gradients compared to conventional finite element methods, which is desired for numerical solutions to the Grad–Shafranov equation. The numerical scheme is augmented with an adaptive mesh refinement approach, and a criterion based on the residual norm in the minimal residual method is developed to achieve dynamic refinement. Nonlinear solvers for the resulting system are explored and a Picard iteration with Anderson acceleration is found to be efficient to solve the system. Finally, the proposed algorithm is implemented in parallel on MFEM using a domain-decomposition approach, and our implementation is general, supporting arbitrary order of accuracy and general meshes. Numerical results are presented to demonstrate the efficiency and accuracy of the proposed algorithm.

Key words. discontinuous Petrov–Galerkin method, adaptive mesh refinement, high-order, Grad–Shafranov

AMS subject classifications. 65N30, 65N50, 65N55, 65F10

DOI. 10.1137/19M1309894

1. Introduction.

The magnetohydrodynamic (MHD) equilibrium is critical in many applications of plasma systems, such as magnetic confinement fusion. Steady-state fusion reactor operation requires that an MHD equilibrium is reached and sustained. In computational plasma physics, an efficient and accurate solver for the MHD equilibrium serves as the basis for linear stablity analysis and nonlinear simulations of plasma transport and off-normal events such as tokamak disruptions. For example, the confining magnetic fields computed from an MHD equilibrium solver can be used as a starting point for the relativistic Fokker–Planck equation in the runaway electron study [23], or a starting point to evolve a nonlinear MHD simulation [27]. For more details on the importance to have accurate and efficient equilibrium solvers, see the discussions in [32], for instance.

The axisymmetric MHD equilibrium is governed by the Grad–Shafranov equation, which is an elliptic equation with a nonlinear source term [22, 50]. There are two types of equilibrium problems, the so-called fixed boundary and free boundary equilibria. In the fixed boundary equilibrium problem, it is assumed that the separatrix of the MHD equilibrium is known and a Dirichlet boundary condition is imposed at the separatrix for the Grad–Shafranov equation. (In practice, a closed flux surface just inside the separatrix, the so-called q_{95} surface, is used for the computational boundary.) The free

*Submitted to the journal's Computational Methods in Science and Engineering section January 10, 2020; accepted for publication (in revised form) July 9, 2020; published electronically October 8, 2020.

https://doi.org/10.1137/19M1309894

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.

†Department of Mathematics, Michigan State University, East Lansing, MI 48824 USA (pengzhic@msu.edu).

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

boundary problem considers the case of an unbounded domain when the separatrix is not known and is in fact to be determined self-consistently from both the plasma current and electrical currents in the external field coils. Thus, a free boundary solver needs extra capabilities, such as a boundary integral term that addresses the effect of a far-field boundary and numerical routines to locate the separatrix in a given trial solution. As an example, see [24] for a finite-element-based free boundary solver. In this work, we focus on the fixed boundary problem and develop a fast and high-order finite-element-based solver on general meshes. The fixed boundary solver is a necessary building block for a free boundary solver.

In recent years, the fixed boundary solvers have drawn significant attention due to its importance in magnetic confinement fusion. Many numerical schemes have been explored and developed. These include hybridizable discontinuous Galerkin methods [48, 49], spectral elements [26, 38], boundary integral approaches [39, 32], Hermite finite element [28, 33], and so on. Some of them, such as [49], also considered adaptive mesh refinement (AMR) approaches to minimize numerical errors and improve the efficiency of the algorithm. The derivative of the solutions determines physically important quantities such as magnetic fields for computing the charged particle trajectory. Hence, other than efficiency and accuracy of the algorithms, many of the previous works also focused on minimizing errors in the derivatives of the solutions, including using bicubic elements or introducing auxiliary variables for its derivatives. Constructing accurate derivatives and effective AMR will be the two primary objectives of the current work.

Among many available high-order schemes, we are interested in using discontinuous Petrov Galerkin (DPG) methods for the Grad–Shafranov equation. The DPG method proposed in [13, 14] enjoys the following properties that are particularly attractive for our purpose. First, with an ultraweak formulation and suitable choice of the discrete test space, the DPG method can provide high-order accurate approximation to both solutions and the first derivatives [21]. Second, the DPG method provides a natural built-in error estimator for adaptivity based on a numerical residual [17, 8], while a standard AMR method relies on the calculation of local numerical fluxes or physical features like high-order derivatives.

Next we briefly review the DPG methodology. More details on the formulations can be found in section 3.1. The DPG method is a Petrov–Galerkin finite element method, which uses on-the-fly computed optimal test functions [13, 14]. A discontinuous or broken test space is used so that these optimal test functions can be locally computed on each element. Besides the perspective of optimal functions, the DPG method can be equivalently seen as a minimal residual method [18, 16, 30, 11, 44]. The DPG method preserves a discrete version of the inf-sup condition and provides the best error approximation for an energy norm defined in the trial space [18]. Another attractive feature of the method is that it provides a natural built-in error estimator [17, 8]. AMR and *hp*-adaptivity algorithms [17, 11, 44, 41] are designed based on this estimator. DPG methods for nonlinear problems have also been considered. For instance, in [12, 11, 44], the authors first linearize nonlinear problems and then apply the DPG method to the linearized problems. In [35], a Picard iteration is applied to solve DPG method for a nonlinear optical problem. In [34], from the perspective of minimal residual method, a hybridized DPG method is designed and directly applied to nonlinear fluid problems, and a Newton method is used for the nonlinear solver. In [6], a partial differential equation constrained optimization method is considered as a nonlinear solver for the DPG method. Recently, a posteriori error analysis is also generalized to the nonlinear problem in [7].

In this work, we would like to take full advantage of the DPG methodology as described above and develop an efficient and accurate solver for the Grad–Shafranov equation, which produces not only a high-order accurate solution but also high-order accurate derivatives. The major contributions of this work include (1) we extended the DPG scheme to the nonlinear Grad–Shafranov equation using the formulation based on a minimal residual method; (2) nonlinear solvers to the DPG formulation are considered, including both Newton's methods and a Picard iteration method, of which the Picard iteration method accelarated with Anderson acceleration and algebraic multigrid preconditioners is found to be more efficient; (3) an AMR approach is developed based on the residual norm coming along with the DPG scheme; (4) the algorithm is implemented in parallel on MFEM (`mfem.org`), and the solver supports arbitrary order accuracy and general meshes for complex geometry.

The remainder of the paper is structured as follows. The MHD equilibrium and Grad–Shafranov equation are briefly described in section 2. The formulation of the DPG scheme is given in section 3. Our discussion starts with a review of the abstract DPG formulation based on the minimal residual method. The ultraweak formulation is then described for the Grad–Shafranov equation. The details on discretizations such as its matrix-vector form follow. In section 4, we focus on efficient nonlinear solvers for the resulting matrix-vector form of the DPG scheme. In section 5, we discuss adaptive mesh refinement, followed by details of the implementation in section 6. Finally, numerical results are presented in section 7.

2. Governing equation. In this section, we briefly introduce the MHD equilibrium and Grad–Shafranov equation, and then define the fixed boundary problem that is solved by the DPG scheme in this work.

An MHD equilibrium for magnetically confined plasma means the force balancing

$$\mathbf{j} \times \mathbf{B} = \nabla p,$$

where $\mathbf{B}$ is the magnetic field, $\mathbf{j}$ is the plasma current density that satisfies Ampere's law

$$\mu_0 \mathbf{j} = \nabla \times \mathbf{B},$$

with μ_0 the magnetic permeability, and p is the plasma pressure. If, further, assuming the problem is axisymmetric (e.g., the equilibria in tokamaks) and considering the equilibrium in (r, z)-coordinates, the Grad–Shafranov equation for the magnetic flux function ψ can be written as

$$r\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial \psi}{\partial r}\right) + \frac{\partial^2 \psi}{\partial z^2} = -\mu_0 r^2 \frac{dp}{d\psi} - I\frac{dI}{d\psi},$$

where I is a function of ψ and it is associated with the toroidal part of the magnetic field. The magnetic field satisfies

$$\mathbf{B} = \frac{1}{r}\nabla\psi \times \hat{e}_\theta + \frac{I}{r}\hat{e}_\theta.$$

Note that the quantities of interest in practice are the derivatives of ψ (corresponding to the magnetic field) and its second derivatives (corresponding to the current). For more details on the Grad–Shafranov equation and plasma equilibrium, the readers are referred to plasma textbooks such as [29].

Define a gradient operator $\widetilde{\nabla} = (\partial_r, \partial_z)^T$. In this work, we consider the Grad–Shafranov equation with the fixed boundary condition and the problem is rewritten

as

$$\begin{aligned} \widetilde{\nabla}\cdot\left(\frac{1}{r}\widetilde{\nabla}\psi\right) &= -\frac{F(r,z,\psi)}{r}, && x\in\Omega, \\ \psi &= \psi_D, && x\in\partial\Omega, \end{aligned} \tag{2.1}$$

where $\Omega\subset\mathbb{R}^2$ is the physical domain with a Lipschitz boundary $\partial\Omega$. In tokamaks, the physical domain Ω corresponds to the cross section of the device. The source term $F(r,z,\psi)$ depends on p and I and, in practice, both of them are functions of ψ, which are from either experimental measurements or theoretical design. Therefore, the problem is nonlinear through the source term and constructing efficient nonlinear solvers is one of the main focuses of the current work.

3. Formulation of the DPG method. We begin with a quick review of the formulation of the DPG method as a minimal residual method for a general abstract nonlinear problem. Based on this abstract formulation, we will show some details of an ultraweak DPG formulation for the Grad–Shafranov equation.

3.1. Abstract DPG method as a minimal residual method. Consider an abstract weak formulation for a general nonlinear problem. Let U denote the trial space and V the test space, both of which are Hilbert spaces. The weak formulation for a nonlinear problem is to find $u\in U$ such that,

$$b_N(u,v) = l(v) \qquad \forall\, v\in V, \tag{3.1}$$

where $l(\cdot): V\to\mathbb{R}$ is a linear form and $b_N(u,v): U\times V\to\mathbb{R}$ is nonlinear for $u\in U$ while linear for $v\in V$. A residual operator can be defined as

$$r(u,v) = b_N(u,v) - l(v). \tag{3.2}$$

Let V' denote the dual space of V. Since $b_N(u,v)$ is linear for v, based on the Riesz representation theory, there exists a linear operator $\mathcal{B}: U\to V'$ such that $\forall u\in U, <\mathcal{B}u, v>_{V'\times V} = b_N(u,v)\forall v\in V$. It is also convenient to define the inner products $(\cdot,\cdot)_U$ and $(\cdot,\cdot)_V$ as the inner products in U and V, respectively.

As pointed out in [34, 6, 30, 7] for nonlinear problems, the DPG method is equivalent to a minimal residual method. In this work, we focus on this interpretation of the DPG method in both the discussion and implementation. Here, we briefly review the minimal residual interpretation, and more discussions can be found in [34, 18, 11, 43, 16, 30]. Suppose $U_h\subset U$ and $V_h\subset V$ to be the finite dimensional discrete trial and test spaces. The DPG method can be defined as looking for $u_h\in U_h$ such that

$$\begin{aligned} u_h &= \arg\min_{w_h\in U_h} ||\mathcal{B}\omega_h - l||^2_{V'} \\ &= \arg\min_{w_h\in U_h}\left(\sup_{v_h\in V_h,\, v_h\neq 0}\frac{|b_N(w_h,v_h)-l(v_h)|^2}{||v_h||^2_V}\right) \\ &= \arg\min_{w_h\in U_h}\left(\sup_{v_h\in V_h,\, v_h\neq 0}\frac{|r(w_h,v_h)|^2}{||v_h||^2_V}\right). \end{aligned} \tag{3.3}$$

Let $\{e_{v_i}\}_{i=1}^N$ be a basis of the discrete test space V_h and $\mathbf{e_v} := (e_{v_1},\ldots,e_{v_N})^T$, then for $\forall v_h\in V$, it can be expanded as $v_h = \mathbf{v}^T\mathbf{e_v} = \sum_{i=1}^N v_i\, e_{v_i}$. Since both $l(v)$ and

$b_N(u,v)$ are linear with respect to the argument v, we have

$$r(w_h, v_h) = r\left(w_h, \sum_{i=1}^{N} v_i\, e_{v_i}\right) = \sum_{i=1}^{N} v_i r\left(w_h, e_{v_i}\right) = \mathbf{v}^T \mathbf{r}(w_h),$$

where $\mathbf{r}(w_h) := \left(r(w_h, e_{v_1}), \ldots, r(w_h, e_{v_N})\right)^T$. Then, we note that

$$\frac{|r(w_h, v_h)|^2}{||v_h||_V^2} = \frac{|\mathbf{v}^T \mathbf{r}(w_h)|^2}{\mathbf{v}^T G \mathbf{v}}, \tag{3.4}$$

where the matrix $G \in \mathbb{R}^{N\times N}$ is defined as

$$G_{ij} := \left(e_{v_i}, e_{v_j}\right)_V. \tag{3.5}$$

G is often called a Gram matrix, and here it can be interpreted as a mass matrix corresponding to the inner product $(\cdot,\cdot)_V$.

The Gateaux derivative of (3.4) with respect to $\mathbf{v}$ is

$$\frac{2}{(\mathbf{v}^T G \mathbf{v})^2}\Big[(\mathbf{v}^T \mathbf{r}(w_h))(\mathbf{v}^T G \mathbf{v})\mathbf{r}(w_h) - (\mathbf{v}^T \mathbf{r}(w_h))^2 G\mathbf{v}\Big]. \tag{3.6}$$

It is easy to see when $\mathbf{v}$ satisfies

$$\mathbf{v} = \frac{\mathbf{v}^T G \mathbf{v}}{\mathbf{v}^T \mathbf{r}(w_h)} G^{-1} \mathbf{r}(w_h),$$

the Gateaux derivative is 0, and the supremum of $\frac{|r(w_h,v_h)|^2}{||v_h||_V^2}$ is obtained as

$$\sup_{v_h \in V_h,\, v_h \neq 0} \frac{|r(w_h, v_h)|^2}{||v_h||_V^2} = \mathbf{r}(w_h)^T G^{-1} \mathbf{r}(w_h). \tag{3.7}$$

Therefore, the DPG method (3.3) is equivalent to finding

$$u_h = \underset{w_h \in U_h}{\arg\min} \Big(\mathbf{r}(w_h)^T G^{-1} \mathbf{r}(w_h)\Big). \tag{3.8}$$

Taking the Gateaux derivative of (3.8) gives

$$J^T(u_h) G^{-1} \mathbf{r}(\mathbf{u_h}) = 0, \tag{3.9}$$

where $J(u_h)$ is the Jacobian matrix of the residual $\mathbf{r}(u_h)$ with repect to u_h. Finally, the DPG method based on the minimal residual method is given by (3.9), which will be the focus of the current work. More details for the minimal residual interpretation of the DPG methodology can be found in [34, 30].

The original DPG scheme was defined by computing optimal test functions on the fly; see [14] for instance. The DPG scheme in [14] is equivalent to the form (3.9) derived from the minimal residual method. We refer the readers to [34] for details of its equivalence proof. In summary, the methodology of the DPG method uses special "optimal" test functions, which preserves a discrete inf-sup condition with a discrete inf-sup constant of the same order as the original continuous inf-sup constant and guarantees the "optimal convergence" results; see [14, 18] for instance.

Finally, consider the special case of linear problems. Let $\{e_{u_i}\}_{i=1}^{M}$ be the basis of the discrete trial space U_h and $\mathbf{e_u} := (e_{u_1}, \ldots, e_{u_M})^T$. For $\forall u_h \in U_h$, it can be expanded as $u_h = \mathbf{u}^T \mathbf{e_u}$. For a linear problem when $b_N(u, v)$ becomes a bilinear form, one can verify that (3.9) becomes

$$B^T G^{-1} B\mathbf{u} = B^T G^{-1} \ell,$$

where the matrix $B \in \mathbb{R}^{N\times M}$ defined by $B_{ij} = b(e_{u_j}, e_{v_i})$ and ℓ is a constant vector determined by operator $l(\cdot)$. Hence, for linear problems, the DPG method always results in a symmetric positive definite linear system, while nonlinear problems do not have such a property, which will be discussed in section 4.

3.2. An ultraweak DPG formulation for the Grad–Shafranov equation. We present the details of the DPG formulation for problem (2.1). The motivation to use the DPG scheme is to obtain a more accurate approximation to the gradient of ψ, and we therefore consider the ultraweak formulation to discretize (2.1). Define an auxiliary variable of the vector

$$\mathbf{q} := -\frac{\widetilde{\nabla}\psi}{r},$$

and rewrite (2.1) into its first order form,

$$r\mathbf{q} + \widetilde{\nabla}\psi = 0, \qquad x \in \Omega, \tag{3.10a}$$

$$\widetilde{\nabla} \cdot \mathbf{q} = \frac{1}{r} F(r, z, \psi), \qquad x \in \Omega, \tag{3.10b}$$

$$\psi = \psi_D, \qquad x \in \partial\Omega. \tag{3.10c}$$

Let $\Omega_h = \{K\}$ be a partition of the physical domain Ω, where $K \in \Omega_h$ are disjoint elements. Let $\partial\Omega_h = \{\partial K, K \in T_h\}$ denote its skeleton (edge) and $\Gamma_h = \partial\Omega_h \cap \partial\Omega$ be the set of the edges on the physical boundary. Define $(\cdot,\cdot)_K$ and $\langle\cdot,\cdot\rangle_{\partial K}$ as the standard L^2 inner product of $L^2(K)$ and $L^2(\partial K)$, respectively. Let $(\cdot,\cdot)_{\Omega_h} = \sum_{K\in\Omega_h}(\cdot,\cdot)_K$, $\langle\cdot,\cdot\rangle_{\partial\Omega_h} = \sum_{\partial K\in\partial\Omega_h}\langle\cdot,\cdot\rangle_{\partial K}$ and $\langle\cdot,\cdot\rangle_{\Gamma_h} = \sum_{\partial K\in\Gamma_h}\langle\cdot,\cdot\rangle_{\partial K}$.

We generalize the DPG method for the linear Poisson problem in [21] to the nonlinear Grad–Shafranov equation. Define the trace spaces

$$H^{-\frac{1}{2}}(\partial\Omega_h) := \{\widehat{q_n} : \exists\, \mathbf{q} \in H(\mathrm{div};\Omega) \text{ such that } \widehat{q_n} = \mathbf{q}\cdot n|\partial_K \ \ \forall K \in \Omega_h\},$$

$$H^{\frac{1}{2}}(\partial\Omega_h) := \{\widehat{\psi} : \exists\, \psi \in H^1(\Omega) \text{ such that } \widehat{\psi} = \psi|\partial_K \ \ \forall K \in \Omega_h\},$$

and the broken Sobolev spaces

$$H(\mathrm{div},\Omega_h) := \{\boldsymbol{\phi} \in L^2(\Omega_h) : \ \boldsymbol{\phi}|K \in H(\mathrm{div},K) \ \ \forall K \in \Omega_h\},$$

$$H^1(\Omega_h) := \{\tau \in L^2(\Omega_h) : \ \tau|K \in H^1(K) \ \ \forall K \in \Omega_h\}.$$

The trial space U and the test space V are chosen as

$$U = \left(L^2(\Omega)\right)^2 \times L^2(\Omega) \times H^{-\frac{1}{2}}(\partial\Omega_h) \times H^{\frac{1}{2}}(\partial\Omega_h),$$
$$V = H(\mathrm{div},\Omega_h) \times H^1(\Omega_h).$$

The ultraweak form associated with (3.10) finds $\mathbf{u} = (\mathbf{q}, \psi, \widehat{q_n}, \widehat{\psi}) \in U$ such that for $\forall\, \mathbf{v} = (\boldsymbol{\phi}, \tau) \in V$

$$(r\mathbf{q}, \boldsymbol{\phi})_{\Omega_h} - \Big(\psi, \widetilde{\nabla} \cdot \boldsymbol{\phi}\Big)_{\Omega_h} + \langle \widehat{\psi}, \mathbf{n} \cdot \boldsymbol{\phi} \rangle_{\partial\Omega_h} = 0, \tag{3.11a}$$

$$-(\mathbf{q}, \widetilde{\nabla}\tau)_{\Omega_h} + \langle \widehat{q_n}, \tau \rangle_{\Omega_h} = \left(\frac{F(r,z,\psi)}{r}, \tau\right)_{\Omega_h}, \tag{3.11b}$$

$$\langle \widehat{\psi}, \tau \rangle_{\Gamma_h} = \langle \psi_D, \tau \rangle_{\Gamma_h}. \tag{3.11c}$$

The source term $F(r,z,\psi)$ can be rewritten as the summation of a nonlinear part and a linear part $F(r,z,\psi) = F_N(r,z,\psi) + F_L(r,z)$. Based on the weak form (3.11), we define

$$l(\mathbf{v}) = \left(\frac{F_L(r,z)}{r}, \tau\right)_{\Omega_h}, \tag{3.12a}$$

$$\begin{aligned} b_N(\mathbf{u}, \mathbf{v}) &= (r\mathbf{q}, \boldsymbol{\phi})_{\Omega_h} - \Big(\psi, \widetilde{\nabla} \cdot \boldsymbol{\phi}\Big)_{\Omega_h} + \langle \widehat{\psi}, \mathbf{n} \cdot \boldsymbol{\phi} \rangle_{\partial\Omega_h} \\ &\quad - (\mathbf{q}, \widetilde{\nabla}\tau)_{\Omega_h} + \langle \widehat{q_n}, \tau \rangle_{\Omega_h} - \left(\frac{F_N(r,z,\psi)}{r}, \tau\right)_{\Omega_h}. \end{aligned} \tag{3.12b}$$

In addition, we define the test norm $||\cdot||_V$ as

$$||\mathbf{v}||_V^2 := ||(\boldsymbol{\phi}, \tau)||_V^2 = ||\boldsymbol{\phi}||^2 + ||\widetilde{\nabla} \cdot \boldsymbol{\phi}||^2 + ||\tau||^2 + ||\widetilde{\nabla}\tau||^2, \tag{3.13}$$

and let $(\cdot,\cdot)_V$ be its corresponding inner product.

Finally, we need to determine the discrete trial space U_h and the discrete test space V_h. Let $P^k(K)$ and $P^k(\partial K)$ be the space of polynomials with degree at most k on the element K and its edge ∂K. The discrete trial space U_h and the discrete test space V_h are, respectively, chosen as

$$\begin{aligned} U_h^k &= \Big\{ u_h = (\mathbf{q}_h, \psi_h, \widehat{q}_{n,h}, \widehat{\psi}_h) : \mathbf{q}_h|K \in \big(P^k(K)\big)^2,\ \psi_h|K \in P^k(K), \\ &\qquad \widehat{q}_{n,h}|\partial K \in P^k(\partial K) \cap H^{-\frac{1}{2}}(\partial\Omega_h),\ \widehat{\psi}_h|\partial K \in P^{k+1}(\partial K) \cap H^{\frac{1}{2}}(\partial\Omega_h) \Big\}, \\ V_h^{k,s} &= \Big\{ v_h = (\boldsymbol{\phi}_h, \tau_h) : \boldsymbol{\phi}_h|K \in \big(P^{k+s}(K)\big)^2,\ \tau_h|K \in P^{k+s}(K) \Big\}, \quad s \geq 2. \end{aligned}$$

According to [8], the key to prove the convergence of the DPG scheme is to define a Fortin-type operator $\Pi : V \to V_h = V_h^{k,s}$ such that $b_N(w_h, (\mathbf{I} - \Pi)v) = 0$ for $\forall w_h \in U_h, v \in V$. As proved in [21], $s \geq 2$ is a sufficient condition for the existence of such a Fortin-type operator Π, which further guarantees the convergence of the scheme. We therefore choose $s \geq 2$ in this work. A parameter study for the Fortin operator is carried out in [36]. Finally, based on the given U_h, V_h, (3.12) and (3.13), we are ready to determine G, $J(u_h)$, and $\mathbf{r}(u_h)$ in (3.9) and define our DPG scheme.

Remark 3.1. The choice of the test norm $||\cdot||_V$ is not unique. An attractive alternative is the adjoint graph norm [52, 18, 40]:

$$||\mathbf{v}||_{V,\text{adjoint}}^2 := ||(\boldsymbol{\phi}, \tau)||_{V,\text{adjoint}}^2 = ||r\boldsymbol{\phi} - \widetilde{\nabla}\tau||^2 + ||\widetilde{\nabla} \cdot \boldsymbol{\phi}||^2 + ||\boldsymbol{\phi}||^2 + ||\tau||^2, \tag{3.14}$$

which delivers a quasi-optimal error estimate in L^2 sense. In [15, 21], our current choice $||\cdot||_V$ is also proved to provide a quasi-optimal error estimate for linear elliptical problems.

3.3. Matrix-vector form of the DPG scheme. In this section, the details of the DPG scheme are presented in a matrix-vector form. The following notations will be used throughout this section. Let

$$U_h \supset \{e_{u_j}\}_{j=1}^{M} := \{\mathbf{e}^{\mathbf{q}}_{j_{\mathbf{q}}}\}_{j_{\mathbf{q}}=1}^{M_{\mathbf{q}}} \times \{e^{\psi}_{j_\psi}\}_{j_\psi=1}^{M_\psi} \times \{e^{\widehat{q_n}}_{j_{\widehat{q_n}}}\}_{j_{\widehat{q_n}}=1}^{M_{\widehat{q_n}}} \times \{e^{\widehat{\psi}}_{j_{\widehat{\psi}}}\}_{j_{\widehat{\psi}}=1}^{M_{\widehat{\psi}}}$$

and

$$V_h \supset \{e_{v_i}\}_{i=1}^{N} := \{\mathbf{e}^{\boldsymbol{\phi}}_{i_{\boldsymbol{\phi}}}\}_{i_{\boldsymbol{\phi}}=1}^{N_{\boldsymbol{\phi}}} \times \{e^{\tau}_{i_\tau}\}_{i_\tau=1}^{N_\tau}$$

be the bases of the discrete trial space U_h and the test space V_h, respectively. Note that $\mathbf{e}^{\mathbf{q}}_j$ and $\mathbf{e}^{\boldsymbol{\phi}}_i$ are vector valued, while other basis functions are scalar valued. Define $\mathbf{e}^{\xi}_{\mathbf{u}} := (e^{\xi}_1, \ldots, e^{\xi}_{M_\xi})^T$ and $\mathbf{e}^{\eta}_{\mathbf{v}} := (e^{\eta}_1, \ldots, e^{\eta}_{N_\eta})^T$, where $\xi = \mathbf{q}$, ψ, $\widehat{q_n}$, $\widehat{\psi}$ and $\eta = \boldsymbol{\phi}$, η. Then, q_h, ψ_h, $\widehat{q}_{n,h}$ and $\widehat{\psi}_h$ can be rewritten as

$$\mathbf{q}_h = \mathbf{Q}^T \mathbf{e}^{\mathbf{q}}_{\mathbf{u}}, \qquad \psi_h = \boldsymbol{\Psi}^T \mathbf{e}^{\psi}_{\mathbf{u}}, \qquad \widehat{q}_{n,h} = \widehat{\mathbf{Q}_{\mathbf{n}}}^T \mathbf{e}^{\widehat{q_n}}_{\mathbf{u}}, \qquad \widehat{\psi}_h = \widehat{\boldsymbol{\Psi}}^T \mathbf{e}^{\widehat{\psi}}_{\mathbf{u}}.$$

Define $\mathbf{U} := (\mathbf{Q}^T, \boldsymbol{\Psi}^T, \widehat{\mathbf{Q}_{\mathbf{n}}}^T, \widehat{\boldsymbol{\Psi}}^T)^T$ and, on each element, $\mathbf{Q}$ contains two components corresponding to the vector $\mathbf{q}$. Now, we are ready to rewrite the ultraweak formulation (3.11) into its matrix-vector form and obtain the residual as

$$\begin{aligned}
\mathbf{r}(u_h) &= b_N(u_h, v_h) - l(v_h) \\
&= B_L \mathbf{U} - B_N(\mathbf{U}) - \mathbf{F_L} \\
&= \begin{pmatrix} M_r & \widetilde{\nabla}_h & 0 & T_{\widehat{\psi}} \\ \mathrm{div}_h & 0 & T_{\widehat{q_n}} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{Q} \\ \boldsymbol{\Psi} \\ \widehat{\mathbf{Q}_{\mathbf{n}}} \\ \widehat{\boldsymbol{\Psi}} \end{pmatrix} - \begin{pmatrix} 0 \\ \mathbf{N}(\boldsymbol{\Psi}) \end{pmatrix} - \begin{pmatrix} 0 \\ \mathbf{L} \end{pmatrix},
\end{aligned} \tag{3.15}$$

where the block matrices M_r, $\widetilde{\nabla}_h$, $T_{\widehat{\psi}}$, div_h, and $T_{\widehat{q_n}}$ are defined as

$$\begin{aligned}
&(M_r)_{ij} := \left(\mathbf{e}^{\mathbf{q}}_j, \mathbf{e}^{\boldsymbol{\phi}}_i\right)_{\Omega_h}, \qquad (\widetilde{\nabla}_h)_{ij} := -\left(e^{\psi}_j, \widetilde{\nabla} \cdot \mathbf{e}^{\boldsymbol{\phi}}_i\right)_{\Omega_h}, \\
&(T_{\widehat{\psi}})_{ij} := \langle e^{\widehat{\psi}}_j, \mathbf{n} \cdot \mathbf{e}^{\boldsymbol{\phi}}_i \rangle_{\partial\Omega_h}, \quad (\mathrm{div}_h)_{ij} := -\left(\mathbf{e}^{\mathbf{q}}_j, \widetilde{\nabla} e^{\tau}_i\right)_{\Omega_h}, \quad (T_{\widehat{q_n}})_{ij} := \langle e^{\widehat{q_n}}_j, e^{\tau}_i \rangle_{\partial\Omega_h},
\end{aligned}$$

and the vectors $\mathbf{N}(\boldsymbol{\Psi})$ and $\mathbf{L}$ are defined as

$$(\mathbf{N}(\boldsymbol{\Psi}))_i := (F_N(r, z, \psi_h)/r, e^{\tau}_i)_{\Omega_h}, \qquad \mathbf{L}_i := (F_L(r, z)/r, e^{\tau}_i)_{\Omega_h}.$$

The Jacobian matrix of $\mathbf{r}(u_h)$ can therefore be derived as

$$J(u_h) = \begin{pmatrix} M_r & \widetilde{\nabla}_h & 0 & T_{\widehat{\psi}} \\ \mathrm{div}_h & -D_N(\psi_h) & T_{\widehat{q_n}} & 0 \end{pmatrix}, \tag{3.16}$$

where the block matrix $D_N(\psi_h)$ is defined as $(D_N(\psi_h))_{ij} := (\frac{\partial (F_N(r,z,\psi_h)/r)}{\partial \psi} e^{\psi}_j, e^{\tau}_i)_{\Omega_h}$ and $\frac{\partial (F_N(r,z,\psi_h)/r)}{\partial \psi}$ is the derivative of $F_N(r, z, \psi_h)/r$ with respect to its third argument. Recall the definitions of the Gram matrix G in (3.5) and the test norm $||\cdot||_V$ in (3.13), and we get

$$G = \begin{pmatrix} G_V & 0 \\ 0 & G_S \end{pmatrix}, \tag{3.17}$$

where the block matrices are defined as

$$(G_V)_{ij} = \left(\mathbf{e}_j^{\boldsymbol{\phi}}, \mathbf{e}_i^{\boldsymbol{\phi}}\right)_{\Omega_h} + \left(\widetilde{\nabla}\cdot\mathbf{e}_j^{\boldsymbol{\phi}}, \widetilde{\nabla}\cdot\mathbf{e}_i^{\boldsymbol{\phi}}\right)_{\Omega_h},$$
$$(G_S)_{ij} = \left(e_j^{\tau}, e_i^{\tau}\right)_{\Omega_h} + \left(\widetilde{\nabla}e_j^{\tau}, \widetilde{\nabla}e_i^{\tau}\right)_{\Omega_h}.$$

Substituting (3.15), (3.16), and (3.17) into (3.9), we further rewrite the matrix-vector form of the DPG scheme as

$$J^T(\mathbf{U})G^{-1}\Big[B_L\mathbf{U} - B_N(\mathbf{U}) - \mathbf{F_L}\Big] = 0. \tag{3.18}$$

The above system is the final formulation of the DPG scheme for the Grad–Shafranov equation. The system is nonlinear due to the source term and thus it requires special care to solve it efficiently. In section 4 we will focus on exploring its nonlinear solvers.

4. Nonlinear solvers. In this work, we have considered two types of nonlinear solvers, Newton's method and Picard iteration equipped with Anderson acceleration, for the matrix-vector form (3.18). Some advantages and disadvantages are found during the numerical experiments, which will be addressed in this section. All the nonlinear solvers discussed in this work are implemented through PETSc SNES solvers [4].

4.1. Newton's method. The most popular method for nonlinear problems is Newton's method, which is the first type of solvers we have investigated during this work. The challenge to implement a Newton's method for the system (3.18) is to assemble its Jacobian matrix. It is easy to see that the Jacobian of the system (3.18) is given by

$$\mathcal{J} = J^T(\mathbf{U})G^{-1}J(\mathbf{U}) + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & \mathcal{H}\mathbf{r}_2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{4.1}$$

Here $\mathbf{r}_2$ stands for the second component in the residual $\mathbf{r}(u_h)$ and $\mathcal{H}$ is the Hessian defined by $-\frac{\partial D_N}{\partial \boldsymbol{\Psi}}$, where D_N is the matrix associated with the nonlinear source term. To implement the Jacobian matrix $\mathcal{J}$, one has to implement a Hessian-vector product associated with $\mathcal{H}$, which is not trivial especially in finite element packages such as MFEM. Instead, we avoid assembling the Hessian matrix by using the Jacobian-free Newton-Krylov (JFNK) method that estimates $\mathcal{J}$ through a finite difference approximation of

$$\mathcal{J} \approx \frac{\mathbf{G}(\mathbf{U}+\epsilon\mathbf{V}) - \mathbf{G}(\mathbf{U})}{\epsilon},$$

where the nonlinear function in this case is

$$\mathbf{G}(\mathbf{U}) = J^T(\mathbf{U})G^{-1}[B_L\mathbf{U} - B_N(\mathbf{U}) - \mathbf{F_L}].$$

Without a proper preconditioner, the JFNK solver is typically very inefficient. Through looking at its exact form of (4.1), it seems natural to provide a preconditioner of $J^T(\mathbf{U})G^{-1}J(\mathbf{U})$ for JFNK which only ignore one diagonal term related to $\mathcal{H}$. A block Jacobi preconditioner is further provided to accelerate the inversion of $J^T(\mathbf{U})G^{-1}J(\mathbf{U})$ (the block Jacobi preconditioner is identical to the preconditioner for $\mathbf{J}(\mathbf{U})G^{-1}B_L$ that will be discussed in the next section and we therefore skip the discussion here).

A GMRES solver for the approximated $\mathcal{J}$ is further used in the outer iteration to guarantee the convergence of the full nonlinear solver.

Such an approach works fine for problems that have very weak nonlinearity, while it fails to converge for hard problems when the nonlinear source term is strong. The ignorance of the block matrix $\mathcal{H}\mathbf{r}_2$ appears to be critical in those problems. For instance, for the nonlinear problems presented in section 7, the JFNK solver with the above preconditioning strategy fails, while the same solver works well for the linear problems presented there. Therefore, we further explore another approach that is based on Picard iteration, which is discussed in the next section.

4.2. Anderson acceleration method. The second type of nonlinear solvers we have investigated is the Picard iteration with the Anderson acceleration (Anderson mixing) method [1, 51]. We present the algorithm of Anderson acceleration that is used in this work.

Algorithm 4.1 Anderson acceleration solving nonlinear fixed point problem $\mathbf{U} - A(\mathbf{U}) = 0$.

Given initial guess $\mathbf{U}^0$. Set the initial residual $\mathbf{R}_0$ as $\mathbf{R}_0 = \mathbf{U}_0 - A(\mathbf{U}_0)$.
Set $\mathbf{U}_1 = (1-\lambda_0)U_0 + \lambda_0 A(\mathbf{U}_0)$, $\mathbf{R}_1 = \mathbf{U}_1 - A(\mathbf{U}_1)$ and $k = 1$.
while $\Big(||r_k||_2 \geq \max(rtol||r_0||_2, atol)$ and $||x_k - x_{k-1}|| \geq stol||x_{k-1}||\Big)$ **do**
  Set $m_k = \min\{k, m\}$.
  Find $(\alpha_0^k, \ldots, \alpha_{m_k}^k)$ to be the solution of the constrained minimization problem:

$$(\alpha_0^k, \ldots, \alpha_{m_k}^k) = \arg\min \left\| \sum_{i=0}^{m_k} \alpha_i^k \mathbf{R}_{k-i} \right\|_2^2, \text{ such that } \sum_{i=0}^{m_k} \alpha_i^k = 1.$$

  Set $\mathbf{U}_{k+1} = (1-\lambda_k)\sum_{i=0}^{m_k} \alpha_i^k \mathbf{U}_{k-i} + \lambda_k \sum_{i=0}^{m_k} \alpha_i^k A(\mathbf{U}_{k-i})$.
  Calculate the residual: $\mathbf{R}_{k+1} = \mathbf{U}_{k+1} - A(\mathbf{U}_{k+1})$ and $k := k+1$.
end while

Here λ_k is determined by a cubic backtracking line search algorithm [19].

Note that the system (3.18) is not in the form of a fixed point problem and, hence, some transformations are needed before applying the Anderson acceleration. We have investigated two types of rewritten systems. A straightforward rewritten form of (3.18) compatible with the Anderson acceleration leads to the following system

$$\mathbf{U} - \Big\{ \mathbf{U} - J^T(\mathbf{U}) G^{-1} \Big[B_L \mathbf{U} - B_N(\mathbf{U}) - \mathbf{F_L} \Big] \Big\} = 0.$$

However, we found that the nonlinear solver for this system converges poorly even with the Anderson acceleration. Instead, the following nonlinear fixed point problem is used in our implementation:

$$\mathbf{U} - \left(J^T(\mathbf{U}) G^{-1} B_L \right)^{-1} \Big[J^T(\mathbf{U}) G^{-1} (B_N(\mathbf{U}) - \mathbf{F_L}) \Big] = 0. \tag{4.2}$$

It is straightforward to see that (4.2) is equivalent to (3.18). Note that a similar strategy was applied in [48, 49]. According to [5], this reformulation strategy can be viewed as a "preconditioning" procedure for the nonlinear solver.

At each iteration, we need to invert the block matrix of $J^T(\mathbf{U})G^{-1}B_L$. Using (3.15), (3.16), and (3.17), with some work, we found that

$$
\begin{aligned}
&J^T(\mathbf{U})G^{-1}B_L\\
&=\begin{pmatrix} M_r & \widetilde{\nabla}_h & 0 & T_{\widehat{\psi}} \\ \mathrm{div}_h & -D_N(\psi_h) & T_{\widehat{q_n}} & 0 \end{pmatrix}^T \begin{pmatrix} G_V & 0 \\ 0 & G_S \end{pmatrix}^{-1} \begin{pmatrix} M_r & \widetilde{\nabla}_h & 0 & T_{\widehat{\psi}} \\ \mathrm{div}_h & 0 & T_{\widehat{q_n}} & 0 \end{pmatrix}\\
&=\begin{pmatrix} M_r^T G_V^{-1} M_r + \mathrm{div}_h^T G_S^{-1}\mathrm{div}_h & M_r^T G_V^{-1}\widetilde{\nabla}_h & \mathrm{div}_h^T G_S^{-1} T_{\widehat{q_n}} & M_r^T G_V^{-1} T_{\widehat{\psi}} \\ \widetilde{\nabla}_h^T G_V^{-1} M_r - D_N(\psi_h)^T G_S^{-1}\mathrm{div}_h & \widetilde{\nabla}_h^T G_V^{-1}\widetilde{\nabla}_h & -D_N(\psi_h)^T G_S^{-1} T_{\widehat{q_n}} & \widetilde{\nabla}^T G_V^{-1} T_{\widehat{\psi}} \\ T_{\widehat{q_n}}^T G_S^{-1}\mathrm{div}_h & 0 & T_{\widehat{q_n}}^T G_S^{-1} T_{\widehat{q_n}} & 0 \\ T_{\widehat{\psi}}^T G_V^{-1} M_r & T_{\widehat{\psi}}^T G_V^{-1}\widetilde{\nabla}_h & 0 & T_{\widehat{\psi}}^T G_V^{-1} T_{\widehat{\psi}} \end{pmatrix}.
\end{aligned}
$$

Note that the block matrix is not symmetric due to the appearance of the term $-D_N(\psi_h)$, and the flexible GMRES solver [46] is applied to invert this block matrix. Since the final form of $J^T(\mathbf{U})G^{-1}B_L$ is rather complicated, it is necessary to provide a few words on some details of matrix assembling when implementing the scheme. In our implementation, we assemble most of the small blocks in the beginning of the simulation except for two blocks associated with $-D_N(\psi_h)$. During each iteration of the nonlinear solver, we only need to update two small block matrices resulting from the nonlinear source term. The inversion of the Gram matrices, G_V and G_S, is implemented exactly by inverting the corresponding small matrix on each element during assembling. This is sufficient since our test space is discontinuous. The rest of matrix assembling uses standard approaches.

The iterative linear solver to invert $J^T(\mathbf{U})G^{-1}B_L$ is still inefficient without preconditioning. Since this inversion needs to be performed at each iteration of the nonlinear solver, it is required that such an iterative linear solver must be efficient. To improve the efficiency of this linear solver, we provide a block Jacobian preconditioner associated with $J^T(\mathbf{U})G^{-1}B_L$:

$$
P = \begin{pmatrix} P_{11} & & & \\ & P_{22} & & \\ & & P_{33} & \\ & & & P_{44} \end{pmatrix},
$$

where the blocks are given by

$$
\begin{aligned}
P_{11} &:= M_r^T G_V^{-1} M_r + \mathrm{div}_h^T G_S^{-1}\mathrm{div}_h, \qquad & P_{22} &:= \widetilde{\nabla}_h^T G_V^{-1}\widetilde{\nabla}_h,\\
P_{33} &:= T_{\widehat{q_n}}^T G_S^{-1} T_{\widehat{q_n}}, & P_{44} &:= T_{\widehat{\psi}}^T G_V^{-1} T_{\widehat{\psi}}.
\end{aligned}
$$

In the preconditioning stage, we inverted each block of P_{ii} with algebraic multigrid preconditioners provided by HYPRE. We found it is sufficient to use the standard algebraic multigrid (AMG) method [45] with one V-cycle to invert the blocks of P_{11}, P_{22}, and P_{44}. The auxiliary space Maxwell solver (AMS) method [31] with one V-cycle is applied to invert P_{33}. Note that the block P_{33} is corresponding to $\widehat{q}_{n,h}$, which is associated with the trace of an $H(\mathrm{div})$ space. As an AMG method designed for $H(\mathrm{div})$ and $H(\mathrm{curl})$ spaces based on the idea of an auxiliary space method [25], the AMS method improves the efficiency over the standard AMG method for those spaces. Hence, compared with the standard AMG method, we found that the AMS method is a better choice to invert the block P_{33}.

5. Adaptivity. An AMR approach is developed in our implementation to improve the efficiency of our DPG solver. The basic idea of AMR is to refine a mesh dynamically based on some estimate of local errors in the current solution. For the steady state problem such as the case we consider in this work, AMR can improve the convergence of nonlinear solvers for some hard problems compared with a uniformly refined mesh, since a significant amount of work is focused on the region where errors are large in an adaptive mesh thanks to the error estimator. One advantage of the DPG method in the context of AMR is that it comes with a natural error estimator, while other numerical schemes typically rely on certain estimates, such as the Zienkiewicz–Zhu error estimator which estimates the errors using local numerical fluxes or error estimators through Richardson extrapolation. For instance, in [8] a natural a posteriori error estimator based on the residual $||B_u - l||_{V'}$ is derived for the DPG method solving linear problems. In [7] the posteriori error analysis is generalized to nonlinear problems and an AMR algorithm is presented for the nonlinear problem.

Here we first present the details of the error estimator for the DPG method and then describe the details of an AMR algorithm based on it.

5.1. Error estimator. Recall (3.8), and the DPG method seeks u_h such that

$$u_h = \arg\min_{w_h \in U_h} ||\mathcal{B}\omega_h - l||^2_{V'}. \tag{5.1}$$

A posteriori error estimate for DPG method solving linear problems is given in [8] as

$$C_1||\mathcal{B}u_h - l||_{V'} \le ||\mathbf{u} - u_h||_U \le C_2||\mathcal{B}u_h - l||_{V'} + C_3\mathrm{osc}(l), \tag{5.2}$$

where C_1, C_2, and C_3 are constants independent of the mesh. The term $\mathrm{osc}(l)$ does not depend on the numerical solution u_h, and it is a high order term with respect to the mesh size h, determined by the approximation properties of the finite element spaces. With regularity assumptions, [7] generalizes the results to nonlinear problems. As shown in section 3.1,

$$||\mathcal{B}u_h - l||^2_{V'} = \sup_{v_h \in V_h,\ v_h \neq 0} \frac{|r(u_h, v_h)|^2}{||u_h||^2_V} = \mathbf{r}(\mathbf{u_h})^T G^{-1} \mathbf{r}(\mathbf{u_h}). \tag{5.3}$$

Using the Riesz representation, we can find $\varepsilon_{u_h} \in V$ such that

$$(\varepsilon_{u_h}, v)_V = b_N(u_h, v) - l(v) = r(u_h, v) = (\mathcal{B}u_h - l)(v)\ \forall v \in V.$$

Furthermore, $||\varepsilon_{u_h}||_V = ||\mathcal{B}u_h - l||_{V'} = \sqrt{\mathbf{r}(\mathbf{u_h})^T G^{-1} \mathbf{r}(\mathbf{u_h})}$. Then, we define the local error estimator E_K on each element K as

$$E_K := ||\varepsilon_{u_h}||_{V(K)}. \tag{5.4}$$

The corresponding total error estimator for the whole domain is defined as

$$E_{\mathrm{total}} := \sqrt{\sum_{K \in \Omega_h} ||\varepsilon_{u_h}||^2_{V(K)}}. \tag{5.5}$$

5.2. Adaptive strategy. We only consider conforming h-adaptive refinement with a fixed polynomial order in this work, although the nonconforming AMR is

recently developed in MFEM [10]. We use the following mesh-refinement strategy in the implementation. After each nonlinear solver for the DPG scheme, an updated solution $\mathbf{u}_h$ for the Grad–Shafranov equation is obtained and its residual norm on each element can be estimated using (5.4). In our implementation, for a given element K, it will be refined if all the following three conditions hold:

$$\begin{cases} (1)\ \varepsilon_K > \mathrm{atol}_{\mathrm{amr}}, \\ (2)\ \varepsilon_K > \theta_{\max} \max\limits_{K' \in \Omega_h} \varepsilon_{K'}, \\ (3)\ \varepsilon_K > \theta_{\mathrm{total}}\, E_{\mathrm{total}}/\sqrt{N_{mesh}}, \end{cases}$$

where $\theta_{\max} \in [0, 1)$, $\theta_{\mathrm{total}} \in [0, 1)$ and $\mathrm{atol}_{\mathrm{amr}} \geq 0$ are some given thresholds. In summary, we refine the element K based on the criterion that its local error is less than a predetermined tolerance $\mathrm{atol}_{\mathrm{amr}}$, and its estimated error is relatively larger compared to the errors in other elements. Finally, the stoppage criterion for the AMR iteration is when the total number of elements is larger than some upper bound, or no element satisfies the three conditions above.

6. Implementation. The described DPG algorithm for the Grad–Shafranov equation is implemented under the framework of c++ library MFEM [2]. It would be appropriate to say a few words about our implementation under MFEM.

The algorithm is implemented in parallel using a standard domain decomposition method. All the vectors and small block matrices use the parallel distributed data structure provided by the package and its communication between sub-domains is based on the message-passing interface (MPI). All the linear and nonlinear solvers described in this work are implemented through PETSc [4] and some of the small matrices are preconditioned with HYPRE [20] algebraic multigrid preconditioners to improve efficiency. All the matrix assemblies are performed under MFEM to minimize the communication cost between MFEM and PETSc.

Our implementation of the DPG algorithm is general, supporting arbitrary order of accuracy and general meshes including triangular, quadrilateral and high-order curvilinear meshes, taking full advantage of the capabilities from the package. However, our current implementation only supports conforming AMR, but the implementation still offers refining and dynamic load-balancing, which will be performed after the mesh is updated in each AMR iteration. As a result, in section 7, we mainly focus on the numerical results on triangular meshes and its conforming AMR to verify the scheme. Another reason to focus on triangular meshes is due to the limitation of mesh generations. Currently, for all the problems with complex geometry, we generate triangular meshes using the Gmsh software. To the best of our knowledge, curvilinear meshes with complex geometry obtained from a mesh generator are not fully supported by MFEM. Nevertheless, the focus of the current work is on the DPG scheme, AMR, and efficient nonlinear solvers, while mesh generators are well beyond the scope of this work. Therefore, it is sufficient to use triangular meshes generated by Gmsh for the numerical tests.

Reproducibility. The implementation of the DPG scheme for the Grad–Shafranov equation, as well as the numerical examples presented in this work, is freely available as an MFEM fork at https://github.com/ZhichaoPengMath/mfem/tree/dpg_petsc/miniapps/dpg_grad_shafranov.

7. Numerical tests. We demonstrate the performance of our DPG scheme through a set of numerical examples. Several examples are presented to study the

accuracy of the DPG scheme, including two linear examples and one nonlinear example. The linear examples are further used to compare the performance of the DPG scheme with other finite element methods. We then consider a nonlinear case involving slightly more complicated geometry. Finally, several nonlinear examples are presented to study the performance of AMR.

In practice, we use the trace of the Raviart–Thomas space [42, 37] for the space corresponding to the third component of U_h^k. For the discrete test space $V_h^{k,s}$, $s = 2$ is chosen throughout the numerical study. Unless otherwise noted, the results presented here use the Picard iteration with Anderson acceleration as the nonlinear solver.

7.1. Accuracy tests. Two numerical examples are presented to demonstrate the accuracy of the DPG scheme. The first example considers the linear Solov'ev profiles from [39, 9]. This test is further used to compare the accuracy for both the solution ψ and its derivative $\widetilde{\nabla}\psi/r$ solved by our DPG method, conventional continuous Galerkin (CG) method, and the hybridized discontinuous Galerkin (HDG) method. The second example considers a manufactured solution with a nonlinear source. We use it to demonstrate the accuracy of our scheme in nonlinear problems.

7.1.1. Linear tests. The linear Grad–Shafranov equation of

$$r\widetilde{\nabla}\cdot\left(\frac{1}{r}\widetilde{\nabla}\psi\right) = r^2$$

has an exact solution [39, 9] in the form of

$$\psi(r,z) = \frac{r^4}{8} + d_1 + d_2 r^2 + d_3(r^4 - 4r^2z^2),$$

where the parameters d_1, d_2, and d_3 determine the reasonable plasma cross section.

In [39] it has been described how to use the inverse aspect ratio ε, the elongation κ, and the triangularity δ to determine the parameters d_1, d_2, and d_3. We adopt the same manufactured solutions in the accuracy test. The parameters are determined by the following linear system:

$$\begin{pmatrix} 1 & (1+\varepsilon)^2 & (1+\varepsilon)^4 \\ 1 & (1-\varepsilon)^2 & (1-\varepsilon)^4 \\ 1 & (1-\delta\varepsilon)^2 & (1-\delta\varepsilon)^4 - 4(1-\delta\varepsilon)^2\kappa^2\varepsilon^2 \end{pmatrix} \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} = -\frac{1}{8}\begin{pmatrix} (1+\varepsilon)^4 \\ (1-\varepsilon)^4 \\ (1-\delta\varepsilon)^4 \end{pmatrix}.$$

Two tests are considered in this case, including the ITER-like [3] configuration of $\varepsilon = 0.32$, $\kappa = 1.7$, $\delta = 0.33$ and NSTX-like configuration [47] of $\varepsilon = 0.78$, $k = 2$, $\delta = 0.35$. The computational domains and initial meshes are presented in Figure 1. The meshes are generated using Gmsh by describing the computational domains using the exact solutions. In particular, the computational boundaries are described by Chebyshev nodes along the r direction and z coordinates are then determined by solving $\psi = 0$. The numerical solutions are shown in Figure 2 and the presented results include ψ and its two derivatives of ψ_r/r and ψ_z/r. The results show a good agreement with their exact solutions.

This example is further used to study the accuracy of the DPG schemes as well as several commonly used finite element schemes. We presented the numerical solutions solved by two other schemes, the HDG and CG methods. To approximate $\mathbf{q} = \widetilde{\nabla}\psi/r$, note that the ultraweak DPG and HDG methods directly solve $\mathbf{q}_h$ in the weak formulations, while the CG method only solve ψ_h and its $\mathbf{q}_h$ is obtained by solving

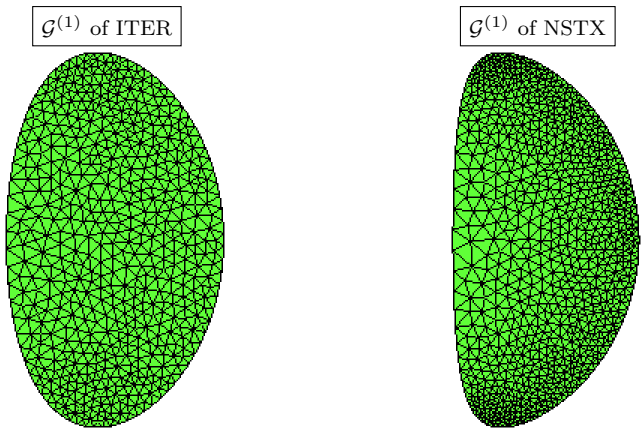

FIG. 1. *Computational domains and initial meshes of $\mathcal{G}^{(1)}$ for two linear accuracy tests. The meshes are generated through Gmsh by describing the computational boundaries.*

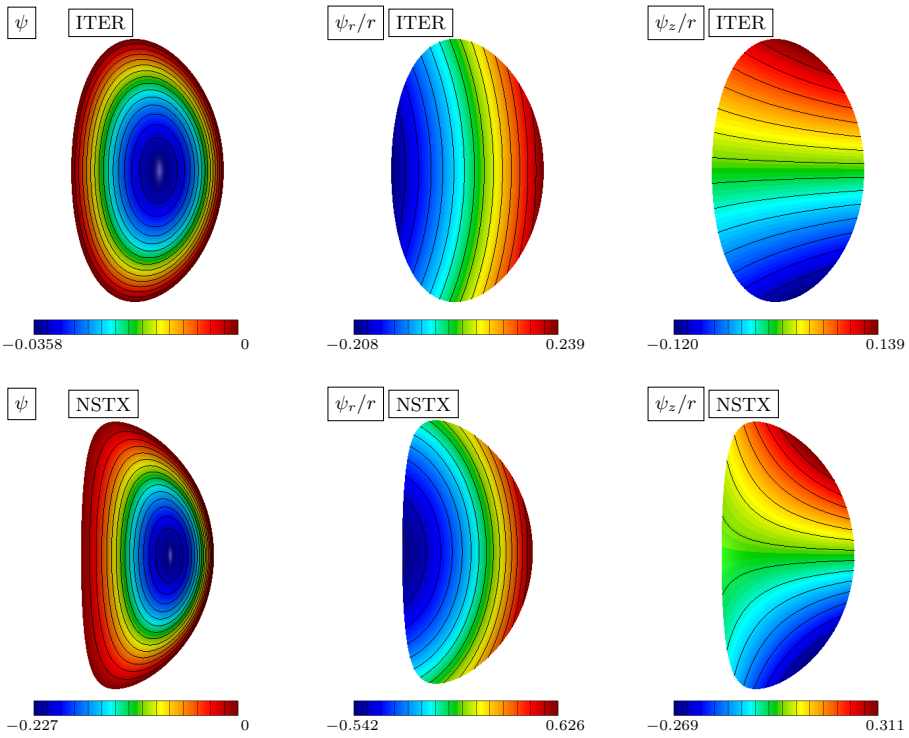

FIG. 2. *Numerical solutions and their derivatives for two linear accuracy tests in section* 7.1.1. *The DPG scheme is used with the meshes of $\mathcal{G}^{(1)}$ and quadratic polynomials.*

$(\mathbf{q}_h, \phi) = (\widetilde{\nabla}\psi_h/r, \phi)$ for arbitrary ϕ in the discrete test space. It is well known that such an approach to compute $\mathbf{q}_h$ will result in accuracy reduction by at least one order. On the other hand, the results of the HDG method do not perform the postprocessing reconstruction, which typically could lift one order of accuracy for ψ. To demonstrate the results, as an example, quadratic polynomials are used for the trial space of all three schemes. The uniform refinement is performed during the convergence study

TABLE 1

L^∞-errors and orders of ψ and $\mathbf{q} = -\widetilde{\nabla}\psi/r$ for the linear problems in section 7.1.1. The DPG, HDG, and CG schemes are used with P^2 polynomials. It demonstrates the accuracy of the DPG scheme and also shows the performance of the different schemes when computing its derivative $\mathbf{q}$.

L^∞-errors and orders of ψ							
Test	Grid	DPG		HDG		CG	
		ψ	order	ψ	order	ψ	order
ITER	$\mathcal{G}^{(1)}$	2.877e-07	-	4.611e-07	-	3.242e-06	-
	$\mathcal{G}^{(2)}$	3.609e-08	2.99	6.035e-08	2.93	4.236e-07	2.94
	$\mathcal{G}^{(4)}$	4.532e-09	2.99	7.745e-09	2.96	5.335e-08	2.99
	$\mathcal{G}^{(8)}$	5.682e-10	3.00	9.809e-10	2.98	6.688e-09	3.00
NSTX	$\mathcal{G}^{(1)}$	4.476e-06	-	6.238e-06	-	4.666e-05	-
	$\mathcal{G}^{(2)}$	6.062e-07	2.88	9.802e-07	2.67	6.500e-06	2.84
	$\mathcal{G}^{(4)}$	7.790e-08	2.96	1.288e-07	2.93	8.502e-07	2.93
	$\mathcal{G}^{(8)}$	9.874e-09	2.98	1.670e-08	2.95	1.102e-07	2.95

L^∞-errors and orders of $\mathbf{q}$							
Test	Grid	DPG		HDG		CG	
		$\mathbf{q}$	order	$\mathbf{q}$	order	$\mathbf{q}$	order
ITER	$\mathcal{G}^{(1)}$	6.742e-07	-	6.338e-06	-	4.248e-04	-
	$\mathcal{G}^{(2)}$	8.696e-08	2.95	8.162e-07	2.96	1.210e-04	1.81
	$\mathcal{G}^{(4)}$	1.084e-08	3.00	1.036e-08	2.98	3.129e-05	1.95
	$\mathcal{G}^{(8)}$	1.369e-09	2.99	1.322e-09	2.97	7.798e-06	2.00
NSTX	$\mathcal{G}^{(1)}$	4.858e-05	-	1.088e-04	-	8.700e-03	-
	$\mathcal{G}^{(2)}$	6.384e-06	2.93	1.548e-05	2.81	2.194e-03	1.99
	$\mathcal{G}^{(4)}$	8.188e-07	2.96	2.039e-06	2.02	5.468e-04	2.00
	$\mathcal{G}^{(8)}$	1.037e-07	2.98	2.634e-07	2.05	1.342e-04	2.03

and the initial meshes are given in Figure 1. L^∞-errors of ψ and $\mathbf{q}$ are presented for different schemes in Table 1. Here the L^∞-error is defined as

$$L^\infty_{\text{error}}(u_h) = \max_{K\in\Omega_h} ||u_h(x) - u_{\text{exact}}(x)||_{\infty,K},$$

where $||\cdot||_{\infty,K}$ is the standard L^∞ norm on K. The L^∞-error for the vector $\mathbf{q}$ is taken as the maximal error among all its components. All three schemes achieve a third-order accuracy for ψ as expected. The accuracy for $\mathbf{q}$ of the DPG and HDG methods are third order, while the CG method only has second-order accuracy. Compared with the HDG method, though with the same accuracy order, the error of the DPG method is smaller for both ψ and $\mathbf{q}$. We note that the HDG method is computationally more efficient than the DPG scheme for this case, as it results in a linear system of smaller size through a Schur complement. The Schur complement for the matrix of $B^TG^{-1}B$ in the linear DPG scheme is not as obvious as the HDG scheme. It becomes even more challenging for the nonlinear problems we will consider later. Therefore, we do not investigate this direction in this work. It is also important to mention that the postprocessing reconstruction for the HDG scheme cannot improve the accuracy of $\mathbf{q} = \widetilde{\nabla}\psi/r$, though it lifts the accuracy order of ψ.

The convergence results indicate the DPG scheme produce more accurate derivates than the other two schemes.

7.1.2. Nonlinear test. Next consider a manufactured solution given in [48]. The solution satisfies

$$\psi(r,z) = \sin\left(k_r(r+r_0)\right)\cos(k_z z),$$

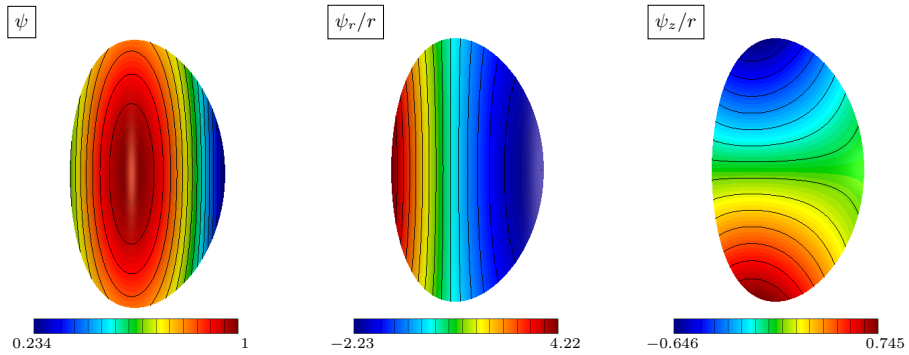

FIG. 3. *Numerical solutions and their derivatives for the nonlinear accuracy problem in section* 7.1.2. *The DPG scheme is used with the meshes of* $\mathcal{G}^{(1)}$ *and quadratic polynomials.*

TABLE 2

L^∞ errors and orders of ψ *and* $\mathbf{q} = -\widetilde{\nabla}\psi/r$ *for the nonlinear problem in section* 7.1.2. P^1, P^2, *and* P^3 *polynomial spaces are used for the DPG scheme.*

L^∞-errors and orders of ψ						
Grid	P^1		P^2		P^3	
$\mathcal{G}^{(1)}$	1.074e-03	-	2.892e-05	-	2.294e-07	-
$\mathcal{G}^{(2)}$	2.648e-04	2.02	2.604e-06	3.47	1.314e-08	4.13
$\mathcal{G}^{(4)}$	6.572e-05	2.01	2.448e-07	3.41	5.446e-09	1.27
$\mathcal{G}^{(8)}$	1.639e-05	2.00	2.451e-08	3.32	1.720e-08	-1.66

L^∞-errors and orders of $\mathbf{q}$						
Grid	P^1		P^2		P^3	
$\mathcal{G}^{(1)}$	2.226e-03	-	4.338e-05	-	7.934e-07	-
$\mathcal{G}^{(2)}$	5.685e-04	1.97	4.697e-06	3.21	5.174e-08	3.94
$\mathcal{G}^{(4)}$	1.433e-04	1.99	5.729e-07	3.04	3.321e-09	3.96
$\mathcal{G}^{(8)}$	3.598e-05	1.99	7.978e-08	2.84	2.385e-10	3.80

with the source term given by

$$F(r,z,\psi) = (k_r^2+k_z^2)\psi + \frac{k_r}{r}\cos(k_r(r+r_0))\cos(k_z z) + r\Big[\sin^2\left(k_r(r+r_0)\right)\cos^2(k_z z) - \psi^2 + \exp\left(-\sin\left(k_r(r+r_0)\right)\cos(k_z z)\right) - \exp(-\psi)\Big],$$

where the coefficients are $k_r = 1.15\pi$, $k_z = 1.15$, and $r_0 = -0.5$. We consider the same ITER geometry as the first linear test in the previous section. The computational domain and the initial mesh are identical to the ITER case given in Figure 1. For this case, a nonhomogenous Dirichlet boundary condition given by the exact solution is used throughout the computations. The solutions are presented in Figure 3, in which quadratic polynomials are used for the trial space.

We further perform a convergence study through a uniform refinement. This case is also used to demonstrate the capability of the arbitrary-order DPG scheme in our implementation. To test the accuracy, U_h^k with $k = 1, 2, 3$ are considered for the discrete trial space. L^∞ numerical errors and the corresponding orders are presented in Table 2. For the discrete trial space U_h^k with $k = 1, 2$, the optimal convergence of $(k+1)$th order is observed for both the solutions and their derivatives. For U_h^3,

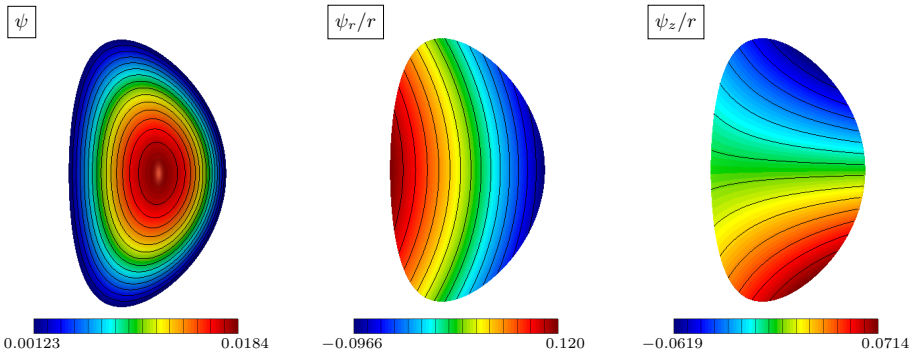

FIG. 4. *Numerical solutions for the case of the D-shaped geometry in section* 7.2.

though order reduction of ψ occurs, optimal convergence for its derivatives is still observed.

7.2. Nonlinear tests. Next we consider a D-shaped geometry taken from [48]. The boundary of computational domain ω is determined by

$$r(s) = 1 + \varepsilon\cos(s + \arcsin(\delta\sin(s)), \quad z(s) = \varepsilon\kappa\sin(s),$$

where $s \in [0, 2\pi]$, $\varepsilon = 0.32$, $\delta = 0.33$, and $\kappa = 1.7$. The source term is

$$F(r, z, \psi) = r^2\Big[1 - \frac{1}{2}(1 - \psi^2)^2\Big]. \tag{7.1}$$

A homogenous Dirichlet boundary condition is used in the simulation. Numerical results with trial space U_h^2 are shown in Figure 4. This case demonstrates the ability to handle different geometry and nonlinear sources in our DPG implementation.

7.3. Adaptivity. In the final part of the numerical example, we focus on the performance of the DPG scheme when the AMR strategy is applied. Two tests are presented with one involving a rectangular geometry and one involving D-shaped geometry. We set $\theta_{\max} = 0.025$ and $\theta_{\text{global}} = 0.025$. The choice of atol_{amr} and the discrete trial space U_h^k will be specified in each test.

7.3.1. Rectangular geometry. First consider a rectangle computational domain of $\Omega = [0.1, 1.6] \times [-0.75, 0.75]$ with a homogenous boundary condition of $\psi = 0.25$. The source term is nonlinear given by

$$F(r, \psi) = 2r^2\psi\Big[c_2(1 - \exp(-\psi^2/\sigma^2) + \frac{1}{\sigma^2}(c_1 + c_2\psi^2)\exp(-\psi^2/\sigma^2)\Big],$$

where $\sigma^2 = 0.005$, $c_1 = 0.8$, and $c_2 = 0.2$. The reason for first considering the rectangular geometry is to eliminate the impact of the boundary condition and the extra complexity associated with a complex geometry when performing AMR. The focus of this case is therefore on the improvement of numerical errors of AMR over a uniform mesh refinement.

For this test, we set atol_{amr} =1e-8 and use quadratic polynomial space U_h^2 as the discrete trial space. The initial mesh is presented in Figure 5. Note that the initial mesh is chosen carefully so that it retains a symmetry along $y = 0$, which is

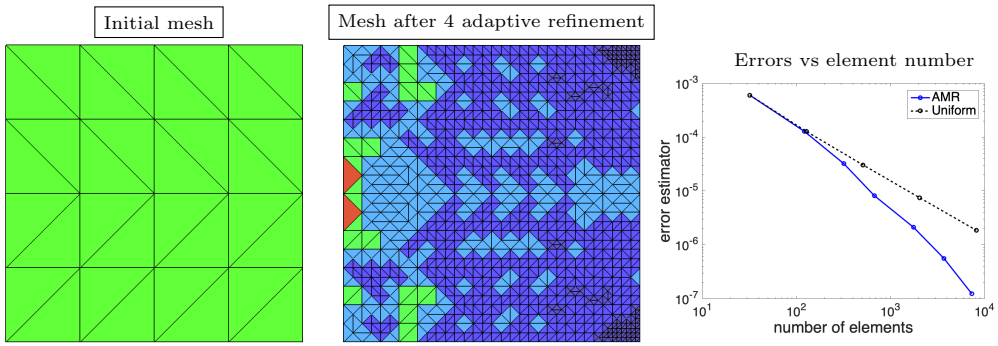

FIG. 5. *Left: initial mesh for the test in section* 7.3.1. *Middle: mesh after four adaptive refinement iterations. Right: errors verse total number of elements for both uniform and adaptive refinements.*

consistent with the exact solution. In our solver, we perform one refinement check after each nonlinear solve, and the updated adaptive mesh will be used again in the next nonlinear solve. An iteration procedure (which is referred to as AMR iterations in this work) is performed until a targeting error goal is achieved or the total number of elements is larger than an upper bound.

The mesh after four AMR iterations is presented in Figure 5. It is accompanied by the numerical solutions on the initial and adaptive meshes presented in Figure 6. The color in the adaptive mesh of Figure 5 indicates different refinement levels of AMR. We first find that the adaptive mesh remains symmetric throughout the AMR iterations. Note that $\mathbf{q} = \widetilde{\nabla}\psi/r$ has sharp features near the top-right corner and the bottom-right corner, which are not well-resolved on the initial mesh. After four AMR iterations, these features are better captured and most refinements are performed near them. Overall, the computational efforts are focused on the region where interesting physics happens during the AMR iteration. All those facts indicate our AMR strategy is effective and efficient.

As a further verification of the AMR strategy, we also compare the numerical errors under uniform and adaptive refinements. Here the numerical error is measured by the value of our error estimator defined in (5.5). The convergence histories for both AMR and uniform refinement are also presented in Figure 5. The numerical errors are presented under an increasing total number of elements. Note that the errors on AMR are much smaller than errors on uniform meshes when the total number of elements (degree of freedom) are the same. Another important observations is that as the meshes are refined, the numerical errors on AMR decay quicker than the errors on uniform meshes. Therefore, the convergence histories quantitatively confirm the efficiency of the AMR approach in this work.

7.3.2. D-shaped geometry. To further confirm the effectiveness of AMR on problems involving complex geometry, we consider the same nonlinear test in D-shaped geometry as described in section 7.2. For this test, we set atol_{amr} =1e-6 and use linear polynomial space U_h^1 as the discrete trial space for the DPG scheme.

The initial mesh and the convergence history of AMR and uniform refinement are presented in Figure 7. We observe that, compared with the uniform refinement, the numerical error on AMR is much smaller and decays faster. This example implies the efficiency and effectiveness of our AMR strategy on problems with complex geometry.

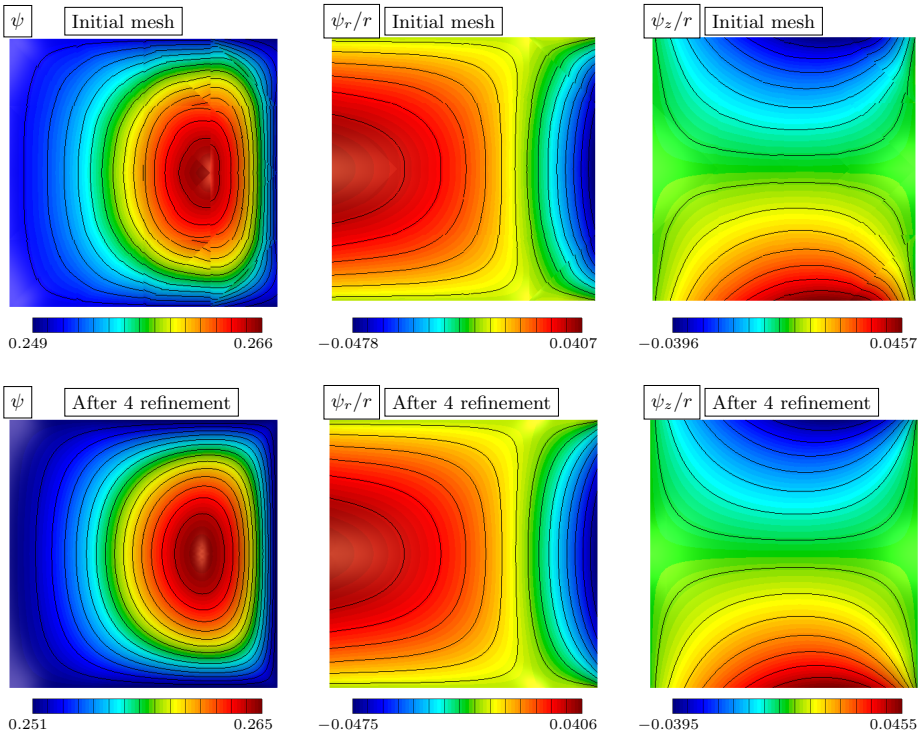

FIG. 6. *Numerical solutions and their derivatives for the rectangular geometry case in section* 7.3.1. *The first row shows the results on the initial mesh. The second row shows the results after four AMR iterations.*

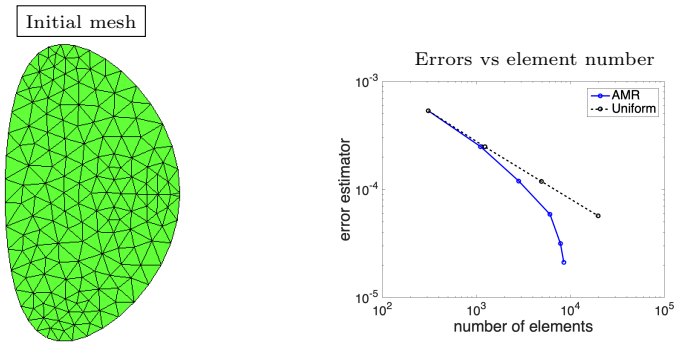

FIG. 7. *Left: initial mesh for the test in section* 7.3.2. *Right: errors versus total number of elements for both uniform and adaptive refinements.*

8. Conclusions. This work focuses on designing and developing an arbitrary-order adaptive DPG method for the nonlinear Grad–Shafranov equation. The main focus of the work is to investigate advantages of the DPG scheme when solving the derivatives as the magnetic field in the Grad–Shafranov equation. The ultraweak formulation of the DPG scheme is developed based on a minimal residual method and the efficient nonlinear solvers are proposed and studied. The algorithm is augmented with an AMR strategy to improve the efficiency of the algorithm. The proposed algorithm is implemented in parallel under the framework of MFEM.

A series of numerical results is presented to verify the accuracy and efficiency of the algorithm. In particular, the algorithm is found to produce more accurate numerical derivatives than some of the commonly used finite element schemes such as CG and HDG schemes. It is also demonstrated that the scheme produces optimal convergence for both the solution and derivatives for kth order polynomial spaces. The nonlinear solver based on Anderson iteration is found to be efficient in solving problems involving strong nonlinearity. Finally, numerical examples also quantitatively demonstrate the improvement of efficiency through the AMR strategy compared with the uniform refinements. In conclusion, the numerical results confirm that the adaptive DPG scheme is a good candidate for nonlinear problems when accurate derivatives are desired, such as the Grad–Shafranov equation considered in this work and many other problems in computational plasma physics.

REFERENCES

[1] D. G. Anderson, *Iterative procedures for nonlinear integral equations*, J. ACM, 12 (1965), pp. 547–560.

[2] R. Anderson et al., *MFEM: Modular Finite Element Methods Library*, Comput. Math. Appl., to appear.

[3] R. Aymar, P. Barabaschi, and Y. Shimomura, *The ITER design*, Plasma Phys. Control. Fusion, 44 (2002), pp. 519–566.

[4] S. Balay et al., *PETSc Users Manual*, Technical Report, ANL-95/11 Rev. 313, Argonne National Laboratory, Argonne, IL, 2019.

[5] P. R. Brune, M. G. Knepley, B. F. Smith, and X. Tu, *Composing scalable nonlinear algebraic solvers*, SIAM Rev., 57 (2015), pp. 535–565.

[6] T. Bui-Thanh and O. Ghattas, *A PDE-constrained optimization approach to the discontinuous Petrov–Galerkin method with a trust region inexact Newton-CG solver*, Comput. Methods Appl. Mech. Engrg., 278 (2014), pp. 20–40.

[7] C. Carstensen, P. Bringmann, F. Hellwig, and P. Wriggers, *Nonlinear discontinuous Petrov–Galerkin methods*, Numer. Math., 139 (2018), pp. 529–561.

[8] C. Carstensen, L. Demkowicz, and J. Gopalakrishnan, *A posteriori error control for DPG methods*, SIAM J. Numer. Anal., 52 (2014), pp. 1335–1353.

[9] A. J. Cerfon and J. P. Freidberg, *"One size fits all" analytic solutions to the Grad–Shafranov equation*, Phys. Plasmas, 17 (2010), 032502.

[10] J. Červenỳ, V. Dobrev, and T. Kolev, *Nonconforming mesh refinement for high-order finite elements*, SIAM J. Sci. Comput., 41 (2019), pp. C367–C392.

[11] J. Chan, L. Demkowicz, and R. Moser, *A DPG method for steady viscous compressible flow*, Comput. & Fluids, 98 (2014), pp. 69–90.

[12] J. Chan, L. Demkowicz, R. Moser, and N. Roberts, *A Class of Discontinuous Petrov–Galerkin Methods. Part* V: *Solution of* 1*D Burgers and Navier–Stokes Equations*, ICES Report 10-25, University of Texas at Austin, Austin, TX, 2010.

[13] L. Demkowicz and J. Gopalakrishnan, *A class of discontinuous Petrov–Galerkin methods. Part* I: *The transport equation*, Comput. Methods Appl. Mech. Engrg., 199 (2010), pp. 1558–1572.

[14] L. Demkowicz and J. Gopalakrishnan, *A class of discontinuous Petrov–Galerkin methods.* II. *Optimal test functions*, Numer. Methods Partial Differential Equations, 27 (2011), pp. 70–105.

[15] L. Demkowicz and J. Gopalakrishnan, *Analysis of the DPG method for the Poisson equation*, SIAM J. Numer. Anal., 49 (2011), pp. 1788–1809.

[16] L. Demkowicz and J. Gopalakrishnan, *Discontinuous Petrov–Galerkin (DPG) method*, in Encyclopedia of Computational Mechanics, 2nd ed., (2017).

[17] L. Demkowicz, J. Gopalakrishnan, and A. H. Niemi, *A class of discontinuous Petrov–Galerkin methods. Part* III: *Adaptivity*, Appl. Numer. Math., 62 (2012), pp. 396–427.

[18] L. F. Demkowicz and J. Gopalakrishnan, *An overview of the discontinuous Petrov Galerkin method*, in Recent Developments in Discontinuous Galerkin Finite Element Methods for Partial Differential Equations, Springer, Cham, Switzerland, 2014, pp. 149–180.

[19] J. E. Dennis Jr. and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Classics Appl. Math. 16, SIAM, Philadelphia, 1996.

[20] R. D. Falgout and U. M. Yang, *HYPRE: A library of high performance preconditioners*, in International Conference on Computational Science, Springer, Berlin, 2002, pp. 632–641.
[21] J. Gopalakrishnan and W. Qiu, *An analysis of the practical DPG method*, Math. Comp., 83 (2014), pp. 537–552.
[22] H. Grad and H. Rubin, *Hydromagnetic equilibria and force-free fields*, J. Nuclear Energy, 7 (1958), pp. 284–285.
[23] Z. Guo, X.-Z. Tang, and C. J. McDevitt, *Models of primary runaway electron distribution in the runaway vortex regime*, Phys. Plasmas, 24 (2017), 112508.
[24] H. Heumann and F. Rapetti, *A finite element method with overlapping meshes for free-boundary axisymmetric plasma equilibria in realistic geometries*, J. Comput. Phys., 334 (2017), pp. 522–540.
[25] R. Hiptmair and J. Xu, *Nodal auxiliary space preconditioning in H (curl) and H (div) spaces*, SIAM J. Numer. Anal., 45 (2007), pp. 2483–2509.
[26] E. Howell and C. R. Sovinec, *Solving the Grad–Shafranov equation with spectral elements*, Comput. Phys. Commun., 185 (2014), pp. 1415–1421.
[27] G. Huysmans and O. Czarny, *MHD stability in X-point geometry: Simulation of ELMs*, Nuclear Fusion, 47 (2007), pp. 659–666.
[28] G. Huysmans, J. Goedbloed, and W. Kerner, *Isoparametric bicubic Hermite elements for solution of the Grad-Shafranov equation*, Int. J. Modern Phys. C, 2 (1991), pp. 371–376.
[29] S. Jardin, *Computational Methods in Plasma Physics*, CRC Press, Boca Raton, FL, 2010.
[30] B. Keith, S. Petrides, F. Fuentes, and L. Demkowicz, *Discrete least-squares finite element methods*, Comput. Methods Appl. Mech. Engrg., 327 (2017), pp. 226–255.
[31] T. V. Kolev and P. S. Vassilevski, *Parallel auxiliary space AMG for H (curl) problems*, J. Comput. Math., 27 (2009), pp. 604–623.
[32] J. Lee and A. Cerfon, *ECOM: A fast and accurate solver for toroidal axisymmetric MHD equilibria*, Comput. Phys. Commun., 190 (2015), pp. 72–88.
[33] H. Lütjens, A. Bondeson, and O. Sauter, *The chease code for toroidal MHD equilibria*, Comput. Phys. Commun., 97 (1996), pp. 219–260.
[34] D. Moro, N. Nguyen, and J. Peraire, *A hybridized discontinuous Petrov–Galerkin scheme for scalar conservation laws*, Internat. J. Numer. Methods Engrg., 91 (2012), pp. 950–970.
[35] S. Nagaraj, J. Grosek, S. Petrides, L. F. Demkowicz, and J. Mora, *A 3D DPG Maxwell approach to nonlinear Raman gain in fiber laser amplifiers*, J. Comput. Phys. X, 2 (2019), 100002.
[36] S. Nagaraj, S. Petrides, and L. F. Demkowicz, *Construction of DPG Fortin operators for second order problems*, Comput. Math. Appl., 74 (2017), pp. 1964–1980.
[37] J.-C. Nédélec, *Mixed finite elements in* $\mathbb{R}^3$, Numer. Math., 35 (1980), pp. 315–341.
[38] A. Palha, B. Koren, and F. Felici, *A mimetic spectral element solver for the Grad–Shafranov equation*, J. Comput. Phys., 316 (2016), pp. 63–93.
[39] A. Pataki, A. J. Cerfon, J. P. Freidberg, L. Greengard, and M. O'Neil, *A fast, high-order solver for the Grad–Shafranov equation*, J. Comput. Phys., 243 (2013), pp. 28–45.
[40] S. Petrides, *Adaptive Multilevel Solvers for the Discontinuous Petrov–Galerkin Method with an Emphasis on High-frequency Wave Propagation Problems*, Ph.D. thesis, University of Texas at Austin, Austin, TX, 2019.
[41] S. Petrides and L. F. Demkowicz, *An adaptive DPG method for high frequency time-harmonic wave propagation problems*, Comput. Math. Appl., 74 (2017), pp. 1999–2017.
[42] P.-A. Raviart and J.-M. Thomas, *A mixed finite element method for 2nd order elliptic problems*, in Mathematical Aspects of Finite Element Methods, Springer, Berlin, 1977, pp. 292–315.
[43] N. V. Roberts, T. Bui-Thanh, and L. Demkowicz, *The DPG method for the Stokes problem*, Comput. Math. Appl., 67 (2014), pp. 966–995.
[44] N. V. Roberts, L. Demkowicz, and R. Moser, *A discontinuous Petrov–Galerkin methodology for adaptive solutions to the incompressible Navier–Stokes equations*, J. Comput. Phys., 301 (2015), pp. 456–483.
[45] J. W. Ruge and K. Stüben, *Algebraic multigrid*, in Multigrid Methods, Frontiers Appl. Math. 3, SIAM, Philkadelphia, 1987, pp. 73–130.
[46] Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput., 14 (1993), pp. 461–469.
[47] S. Sabbagh, et al., *Equilibrium properties of spherical torus plasmas in NSTX*, Nuclear Fusion, 41 (2001), pp. 1601–1612.
[48] T. Sánchez-Vizuet and M. E. Solano, *A Hybridizable Discontinuous Galerkin solver for the Grad–Shafranov equation*, Comput. Phys. Commun., 235 (2019), pp. 120–132.

[49] 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*, Comput. Phys. Commun., 255 (2020), 107239.

[50] V. Shafranov, *On magnetohydrodynamical equilibrium configurations*, Sov. Phys. JETP, 6 (1958), pp. 545–554.

[51] H. F. Walker and P. Ni, *Anderson acceleration for fixed-point iterations*, SIAM J. Numer. Anal., 49 (2011), pp. 1715–1735.

[52] J. Zitelli, I. Muga, L. Demkowicz, J. Gopalakrishnan, D. Pardo, and V. M. Calo, *A class of discontinuous Petrov–Galerkin methods. Part* IV: *The optimal test norm and time-harmonic wave propagation in* 1*D*, J. Comput. Phys., 230 (2011), pp. 2406–2432.