Unified discontinuous Galerkin scheme for a large class of elliptic equations

Nils L. Fischer[1, *] and Harald P. Pfeiffer[1]

[1] *Max Planck Institute for Gravitational Physics (Albert Einstein Institute), Am Mühlenberg 1, Potsdam 14476, Germany*

(Dated: January 13, 2022)

We present a discontinuous Galerkin internal-penalty scheme that is applicable to a large class of linear and nonlinear elliptic partial differential equations. The unified scheme can accommodate all second-order elliptic equations that can be formulated in first-order flux form, encompassing problems in linear elasticity, general relativity, and hydrodynamics, including problems formulated on a curved manifold. It allows for a wide range of linear and nonlinear boundary conditions, and accommodates curved and nonconforming meshes. Our generalized internal-penalty numerical flux and our Schur-complement strategy of eliminating auxiliary degrees of freedom make the scheme compact without requiring equation-specific modifications. We demonstrate the accuracy of the scheme for a suite of numerical test problems. The scheme is implemented in the open-source `SpECTRE` numerical relativity code.

I. INTRODUCTION

Many problems in physics involve the numerical solution of second-order elliptic partial differential equations (PDEs). Such elliptic problems often represent static field configurations under the effect of external forces and arise, for example, in electrodynamics, in linear or nonlinear elasticity, and in general relativity. Elliptic problems also often accompany time evolutions, where they constrain the evolved fields at every instant in time or provide admissible initial data for the evolution.

Discontinuous Galerkin (DG) methods are gaining popularity in the computational physics and engineering community and are currently most prevalently used for time evolutions of hyperbolic boundary-value problems [1–4]. Many properties that make DG methods advantageous for time evolutions also apply to elliptic problems, which lead to the development of DG schemes for elliptic PDEs [5, 6]. In particular, DG schemes provide a flexible mechanism for refining the computational grid, retaining exponential convergence even in the presence of discontinuities when adaptive mesh refinement (AMR) techniques are employed [7, 8]. Furthermore, some difficulties with DG schemes in time evolutions, such as shock capturing, are not present in elliptic problems and their static nature makes it often (but not always) straightforward to place grid boundaries at discontinuities, thus relieving the AMR scheme from the responsibility of resolving them. See, e.g., Ref. [2] and the seminal paper [6] for extensive discussions of DG schemes for the Poisson equation, and Refs. [9–11] for discussions of linear and nonlinear elasticity.

In the context of relativistic astrophysics and numerical relativity, DG methods have been developed for hyperbolic equations on curved manifolds thus far [12–14]. In Ref. [8] we explored the feasibility of the DG method for elliptic problems in numerical relativity confined to flat Poisson-type equations with nonlinear sources. In this article we present a DG scheme suitable to solve a significantly larger class of elliptic problems that arise in numerical relativity. Most notably, the scheme encompasses the extended conformal thin sandwich (XCTS) formulation of the general-relativistic Einstein constraint equations on a curved manifold, and associated boundary conditions [15–17]. Solutions to the XCTS equations provide admissible initial data for general-relativistic time evolutions, for scenarios such as two orbiting black holes or neutron stars [18–21]. To our knowledge, this article presents the first discontinuous Galerkin solution of the full Einstein constraint equations. Aimed at applications in numerical relativity, the scheme is implemented in the publicly available `SpECTRE` code [13, 22, 23].

Furthermore, the elliptic DG scheme presented in this article is not limited to applications in numerical relativity. It applies to all second-order elliptic problems that can be formulated in first-order flux form. Besides the classic Poisson and elasticity equations it covers a large class of elliptic problems in general relativity and hydrodynamics, including coupled systems of equations and those formulated on a curved manifold. With our unified DG scheme, new elliptic systems can be implemented by supplying their first-order fluxes and sources, hence no knowledge of the DG technology or of finite-element formulations is required. This lowers the barrier for extending the capabilities of a simulation code. We pay particular attention to support a wide range of linear and nonlinear boundary conditions so our DG scheme is suited to solve many real-world scenarios (as well as some out-of-this-world scenarios such as initial data for evolutions of black holes and neutron stars). We are aware only of Ref. [24] studying a nonlinear boundary condition for an elliptic DG problem.

To formulate the unified DG scheme we present a generalized internal-penalty numerical flux, which avoids problem-specific parameters that are needed, e.g., in Refs. [25–27]. We eliminate auxiliary degrees of freedom that arise from the first-order form with a Schur-complement strategy, which has proven more suitable to the unified DG scheme than primal formulations that are commonly employed in the literature [2, 6]. The resulting DG scheme is compact, in the sense that it involves only

* nils.fischer@aei.mpg.de

nearest-neighbor couplings and no auxiliary degrees of freedom, and symmetric, unless the symmetry is broken by the elliptic equations.

This article is structured as follows. Section II details the generic first-order flux formulation that serves as the starting point for our DG discretization. Section III develops the unified DG scheme. In Section IV we apply the DG scheme to a set of increasingly challenging test problems. The test problems include scenarios derived from general relativity that feature sets of coupled, strongly nonlinear equations on a curved manifold with nonlinear boundary conditions, solved on curved meshes. We conclude in Section V.

II. FIRST-ORDER FLUX FORMULATION

We consider second-order elliptic PDEs of one or more "primal" variables $u_A(\boldsymbol{x})$, where the index A labels the variables. The variables can be scalars (like in the Poisson equation) or tensorial quantities (like in an elasticity problem). We reduce the PDEs to first order by introducing "auxiliary" variables $v_A(\boldsymbol{x})$, which typically are gradients of the primal variables. We then restrict our attention to problems that can be formulated in first-order flux form

$$-\partial_i \mathcal{F}_\alpha{}^i[u_A, v_A; \boldsymbol{x}] + \mathcal{S}_\alpha[u_A, v_A; \boldsymbol{x}] = f_\alpha(\boldsymbol{x}), \tag{1}$$

where the index α enumerates both u_A and v_A. Here the fluxes $\mathcal{F}_\alpha{}^i$ and the sources $\mathcal{S}_\alpha$ are functionals of the variables u_A and v_A, but not their derivatives, as well as the coordinates $\boldsymbol{x}$. The fixed sources $f_\alpha(\boldsymbol{x})$ are independent of the variables. Lowercase Latin indices i, j, k, l enumerate spatial dimensions, and we employ the Einstein sum convention to sum over repeated indices.

The flux form (1) is general enough to encompass a wide range of elliptic problems. For example, a flat-space Poisson equation in Cartesian coordinates

$$-\partial_i \partial_i u(\boldsymbol{x}) = f(\boldsymbol{x}) \tag{2}$$

has the single primal variable $u(\boldsymbol{x})$. Choosing the auxiliary variable $v_i = \partial_i u$ we can formulate the Poisson equation with the fluxes and sources

$$\mathcal{F}_{v\,j}{}^i = u\,\delta^i_j, \quad \mathcal{S}_{v\,j} = v_j, \quad f_{v\,j} = 0, \tag{3a}$$
$$\mathcal{F}_u{}^i = v_i, \qquad \mathcal{S}_u = 0, \qquad f_u = f(\boldsymbol{x}), \tag{3b}$$

where δ^i_j denotes the Kronecker delta. Note that Eq. (3a) is the definition of the auxiliary variable, and Eq. (3b) is the Poisson equation (2).

The equation of linear elasticity in Cartesian coordinates,

$$-\partial_i Y^{ijkl} \partial_{(k}\,\xi_{l)} = f^j(\boldsymbol{x}), \tag{4}$$

has the primal variable $\xi^i(\boldsymbol{x})$, describing the vectorial deformation of an elastic material. The constitutive relation $Y^{ijkl}(\boldsymbol{x})$ captures the elastic properties of the material in the linear regime. Choosing the symmetric strain $S_{ij} = \partial_{(i}\,\xi_{j)} = (\partial_i\,\xi_j + \partial_j\,\xi_i)/2$ as auxiliary variable we can formulate the elasticity equation with the fluxes and sources

$$\mathcal{F}_S{}^i{}_{jk} = \delta^i_{(j}\,\xi_{k)}, \qquad \mathcal{S}_{S\,jk} = S_{jk}, \quad f_{S\,jk} = 0, \tag{5a}$$
$$\mathcal{F}_\xi{}^{ij} = Y^{ijkl} S_{kl}, \qquad \mathcal{S}_\xi{}^j = 0, \qquad f_\xi{}^j = f^j(\boldsymbol{x}). \tag{5b}$$

Again, Eq. (5a) is the definition of the auxiliary variable and Eq. (5b) is the elasticity equation (4). The fluxes and sources for the elasticity system (5) have higher rank than those for the Poisson system (3).

The first-order flux form (1) also accommodates equations formulated on a curved manifold which is equipped with a metric $g_{ij}(\boldsymbol{x})$. Such equations typically involve covariant derivatives ∇_i compatible with g_{ij}. To formulate the equations in flux form (1) we expand covariant derivatives in partial derivatives and Christoffel symbols $\Gamma^i_{jk} = \frac{1}{2} g^{il} \left(\partial_j g_{kl} + \partial_k g_{jk} - \partial_l g_{jk}\right)$. Christoffel symbols also appear when formulating equations in curvilinear coordinates. In our scheme, the terms with partial derivatives are assigned to the fluxes $\mathcal{F}^i$ and the terms with Christoffel symbols are assigned to the sources $\mathcal{S}$. For example, a curved-space Poisson equation

$$-g^{ij}\nabla_i \nabla_j u(\boldsymbol{x}) = f(\boldsymbol{x}) \tag{6}$$

with auxiliary variable $v_i = \nabla_i u$ can be formulated with the fluxes and sources

$$\mathcal{F}_{v\,j}{}^i = u\,\delta^i_j, \quad \mathcal{S}_{v\,j} = v_j, \qquad f_{v\,j} = 0, \tag{7a}$$
$$\mathcal{F}_u{}^i = g^{ij} v_j, \quad \mathcal{S}_u = -\Gamma^i_{ij} g^{jk} v_k, \quad f_u = f(\boldsymbol{x}). \tag{7b}$$

Our strategy of expanding covariant derivatives differs from the formulations employed for relativistic hyperbolic conservation laws in Ref. [12], where fluxes are always vector fields and therefore the covariant divergence can always be written in terms of partial derivatives and the metric determinant.[1] In contrast, fluxes in the elliptic equations (1) can be higher-rank tensor fields, as exemplified in Eq. (5).

The fixed sources $f_\alpha(\boldsymbol{x})$ could, in principle, be absorbed in the sources $\mathcal{S}_\alpha$. However, it is useful to keep these variable-independent contributions separate for two reasons. First, they remain constant throughout an elliptic solve, so they need not be recomputed when the dynamic variables change. Second, the constant contributions represent a nonlinearity in the variables u_A and v_A when included in the sources $\mathcal{S}_\alpha$. Assigning the constant contributions to the fixed sources f_α eliminates this particular nonlinearity, hence allowing us to avoid an explicit linearization procedure if the remaining sources $\mathcal{S}_\alpha$ are linear.

The Appendix lists fluxes and sources for selected elliptic problems. Our focus on systems in generic first-order

[1] See Eq. (2.3) in Ref. [12].

flux form allows us to solve a variety of elliptic systems by only implementing their fluxes and sources. We now proceed to discretize this generic formulation.

III. DG DISCRETIZATION OF THE FLUX FORMULATION

In this section we develop the unified DG scheme for elliptic equations in flux form, Eq. (1). Novel features of our scheme are the formulation of DG residuals and boundary conditions in terms of generic fluxes and sources of arbitrary tensor rank (Sections III B and III E), and the generalized internal-penalty numerical flux (Section III D). The Schur-complement strategy of eliminating auxiliary degrees of freedom has been employed before, e.g., in Ref. [28], but we generalize it to a larger class of equations, including equations with nonlinear fluxes or sources (Section III C). We follow Ref. [12] whenever possible and refer to Ref. [2] for details that have become standard in the DG literature.[2]

A. Domain decomposition

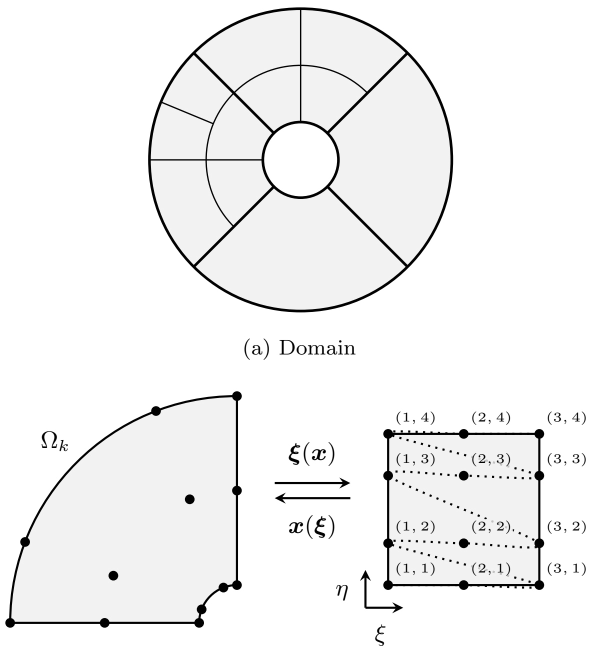
We adopt the same domain decomposition based on deformed cubes detailed in Refs. [8, 12, 13] and summarize it here.

A d-dimensional computational domain $\Omega \subset \mathbb{R}^d$ is composed of *elements* $\Omega_k \subset \Omega$ such that $\Omega = \bigcup_k \Omega_k$. Elements do not overlap, but they share boundaries, as illustrated in Fig. 1a. Each element carries an invertible map $\boldsymbol{\xi}(\boldsymbol{x})$ from the coordinates $\boldsymbol{x} \in \Omega_k$, in which the elliptic equations (1) are formulated, to *logical* coordinates $\boldsymbol{\xi} \in [-1,1]^d$ representing a d-dimensional reference cube. Inversely, $\boldsymbol{x}(\boldsymbol{\xi})$ maps the reference cube to the element Ω_k. We define its Jacobian as

$$\mathrm{J}^i_j \coloneqq \frac{\partial x^i}{\partial \xi^j} \tag{8}$$

with determinant J and inverse $(\mathrm{J}^{-1})^j_i = \partial\xi^j/\partial x^i$.

Within each element Ω_k we choose a set of $N_{k,i}$ grid points in every dimension i. We place them at logical coordinates ξ_{p_i}, where the index $p_i \in \{1,\dots,N_{k,i}\}$ identifies the grid point along dimension i. The points are laid out in a regular grid along the logical coordinate axes, so an element has a total of $N_k = \prod_{i=1}^d N_{k,i}$ d-dimensional grid points $\boldsymbol{\xi}_p = (\xi_{p_1},\dots,\xi_{p_d})$. The index $p \in \{1,\dots,N_k\}$ identifies the grid point regardless of dimension. The full domain has $N_\mathrm{points} = \sum_k N_k$ grid points. The grid points within each element are not uniformly spaced in



(a) Domain

(b) Element

FIG. 1. *Top:* Geometry of a two-dimensional computational domain composed of four wedge-shaped blocks. Each block is split in one or more nonoverlapping elements Ω_k. *Bottom:* The coordinate transformation $\boldsymbol{\xi}(\boldsymbol{x})$ maps an element to a reference cube $[-1,1]^2$ with logical coordinate axes $\boldsymbol{\xi} = (\xi, \eta)$. In this example we chose $N_{k,\xi} = 3$ and $N_{k,\eta} = 4$ Legendre-Gauss-Lobatto collocation points along ξ and η, respectively. Each grid point is labeled with its index (p_ξ, p_η). The dotted line connects points in the order they are enumerated in by the index p.

logical coordinates. Instead, we choose Legendre-Gauss-Lobatto (LGL) collocation points, i.e., the points ξ_{p_i} fall at the roots of the $(N_{k,i}-1)$th Legendre polynomial plus a point on each side of the element, at -1 and 1.[3] It is equally possible to choose Legendre-Gauss (LG) collocation points, i.e., the roots of the $N_{k,i}$th Legendre polynomial.[4] Figure 1b illustrates the geometry of an element.

Fields are represented numerically by their values at the grid points. To facilitate this we construct the one-dimensional Lagrange polynomials

$$\ell_{p_i}(\xi) \coloneqq \prod_{\substack{q_i=1 \\ q_i\neq p_i}}^{N_{k,i}} \frac{\xi - \xi_{q_i}}{\xi_{p_i} - \xi_{q_i}} \quad \text{with} \quad \xi \in [-1,1] \tag{9}$$

and employ their product to define the d-dimensional basis functions

$$\psi_p(\boldsymbol{\xi}) \coloneqq \prod_{i=1}^{d} \ell_{p_i}(\xi^i) \quad \text{with} \quad \boldsymbol{\xi} \in [-1,1]^d. \tag{10}$$

[2] Reference [12] underpins the *hyperbolic* DG formulations in the SpECTRE code. Formulating elliptic and hyperbolic DG schemes in a similar way allows us to share some of the DG implementation details.

[3] See, e.g., Algorithm 25 in Ref. [29].
[4] See, e.g., Algorithm 23 in Ref. [29].

The choice of Lagrange polynomials makes Eq. (10) a nodal basis with the useful property $\psi_p(\boldsymbol{\xi}_q) = \delta_{pq}$. We use the nodal basis (10) to approximate any field $u(\boldsymbol{x})$ within an element Ω_k by its discretization

$$u^{(k)}(\boldsymbol{x}) \coloneqq \sum_{p=1}^{N_k} u_p \psi_p(\boldsymbol{\xi}(\boldsymbol{x})) \quad \text{with} \quad \boldsymbol{x} \in \Omega_k, \tag{11}$$

where the coefficients $u_p = u(\boldsymbol{x}(\boldsymbol{\xi}_p))$ are the field values at the grid points. We denote the set of discrete field values within an element Ω_k as

$$\underline{u}^{(k)} = (u_1, \ldots, u_{N_k}), \tag{12}$$

and the collection of discrete field values over *all* elements as $\underline{u}$. The discretization (11) approximates fields with polynomials of degree $(N_{k,i} - 1)$ in dimension i. Although rarely needed, field values at other points within an element can be obtained by Lagrange interpolation (11). The field values at element boundaries are double valued because the Lagrange interpolation from neighboring elements to their shared boundary is double valued. Therefore, field approximations will in general be discontinuous at element boundaries.

The test problems in Section IV illustrate a few examples of domain decompositions. We refer the reader to, e.g., Ref. [2] for further details on the choice of collocation points, basis functions and their relation to spectral properties of DG schemes.

B. DG residuals

The DG residuals represent the set of equations to be solved for the discrete primal field values $\underline{u}_A$. The derivation in this section follows the standard procedure, e.g., laid out in Ref. [2], applied to the generic elliptic flux formulation (1), and taking details such as a curved manifold into account.

In the spirit of a Galerkin scheme we project our target PDEs (1) onto the same set of basis functions $\psi_p(\boldsymbol{\xi})$ that is used to approximate fields within an element Ω_k,

$$-(\psi_p, \partial_i \mathcal{F}^i)_{\Omega_k} + (\psi_p, \mathcal{S})_{\Omega_k} = (\psi_p, f)_{\Omega_k}. \tag{13}$$

Here we dropped the index α that enumerates the equations, and we define the inner product on Ω_k,

$$(\phi, \pi)_{\Omega_k} \coloneqq \int_{\Omega_k} \phi(\boldsymbol{x}) \pi(\boldsymbol{x}) \sqrt{g}\, \mathrm{d}^d x \tag{14a}$$

$$= \int_{[-1,1]^d} \phi(\boldsymbol{x}(\boldsymbol{\xi})) \pi(\boldsymbol{x}(\boldsymbol{\xi})) \sqrt{g}\, \mathrm{J}\, \mathrm{d}^d \xi\,. \tag{14b}$$

These integrals are defined with respect to proper volume $\mathrm{d}V = \sqrt{g}\, \mathrm{d}^d x = \sqrt{g}\, \mathrm{J}\, \mathrm{d}^d \xi$, where g denotes the metric determinant in the coordinates $\boldsymbol{x}$ in which Eq. (1) is formulated. It refers to the metric that covariant derivatives in the equations are compatible with. Since the basis polynomials, Eq. (10), are functions of logical coordinates, we abbreviate $\psi_p(\boldsymbol{\xi}(\boldsymbol{x}))$ with $\psi_p(\boldsymbol{x})$ here.

The terms without derivatives in Eq. (13) are straightforward to discretize. We approximate the field f, or similarly $\mathcal{S}$, using the expansion in basis functions (11) to find

$$(\psi_p, f)_{\Omega_k} \approx (\psi_p, \psi_q)_{\Omega_k} f_q = M_{pq} f_q, \tag{15}$$

using the symmetric *mass matrix* on the element Ω_k,

$$M_{pq} \coloneqq (\psi_p, \psi_q)_{\Omega_k} \tag{16a}$$

$$= \int_{[-1,1]^d} \psi_p(\boldsymbol{\xi}) \psi_q(\boldsymbol{\xi}) \sqrt{g}\, \mathrm{J}\, \mathrm{d}^d \xi\,. \tag{16b}$$

We will discuss strategies to evaluate the mass matrix on the elements of the computational domain in Section III F.

The divergence term in Eq. (13) encodes the principal part of the elliptic PDEs and requires more care in its discretization. The derivatives in this term will help us couple grid points across element boundaries. To this end we integrate by parts to obtain a boundary term

$$(\psi_p, \partial_i \mathcal{F}^i)_{\Omega_k} = -(\partial_i \psi_p, \mathcal{F}^i)_{\Omega_k} + (\psi_p, n_i \mathcal{F}^i)_{\partial\Omega_k}, \tag{17}$$

where n_i is the outward-pointing unit normal one form on the element boundary $\partial\Omega_k$. The *unnormalized* face normal is computed from the Jacobian as $\tilde{n}_i = \mathrm{sgn}(\xi^j)(\mathrm{J}^{-1})^j_i$, where ξ^j is the logical coordinate that is constant on the particular face and no sum over j is implied. The face normal is normalized as $n_i = \tilde{n}_i / \sqrt{\tilde{n}_k \tilde{n}_l g^{kl}}$ using the inverse metric $g^{ij}(\boldsymbol{x})$. The surface integral in Eq. (17) is defined just like Eq. (14),

$$(\phi, \pi)_{\partial\Omega_k} \coloneqq \int_{\partial\Omega_k} \phi(\boldsymbol{x}) \pi(\boldsymbol{x}) \sqrt{g^\Sigma}\, \mathrm{d}^{d-1} x \tag{18a}$$

$$= \int_{[-1,1]^{d-1}} \phi(\boldsymbol{x}(\boldsymbol{\xi})) \pi(\boldsymbol{x}(\boldsymbol{\xi})) \sqrt{g^\Sigma}\, \mathrm{J}^\Sigma\, \mathrm{d}^{d-1} \xi\,, \tag{18b}$$

using the element boundary's $(d-1)$-dimensional proper volume $\mathrm{d}\Sigma = \sqrt{g^\Sigma}\, \mathrm{d}^{d-1} x = \sqrt{g^\Sigma}\, \mathrm{J}^\Sigma\, \mathrm{d}^{d-1} \xi$, where g^Σ is the surface metric determinant induced by the metric g_{ij} and J^Σ is the surface Jacobian.

The crucial step that couples grid points across element boundaries follows from the field $n_i \mathcal{F}^i$ being double valued on any section of the boundary that an element shares with a neighbor, with one value arising from either side. We must make a choice how to combine the two values from either side of a shared element boundary. This choice is often referred to as a *numerical flux*. For now we will denote the function that combines values from both sides of a boundary as $(n_i \mathcal{F}^i)^*$ and refer to Section III D for details on our particular choice of numerical flux. Substituting the numerical flux in Eq. (17) yields the *weak* form of the equations,

$$(\psi_p, \partial_i \mathcal{F}^i)_{\Omega_k} = -(\partial_i \psi_p, \mathcal{F}^i)_{\Omega_k} + (\psi_p, (n_i \mathcal{F}^i)^*)_{\partial\Omega_k}. \tag{19}$$

The numerical flux in Eq. (19) introduces a coupling between neighboring elements that allows us to obtain numerical solutions spanning the full computational domain. Another integration by parts of Eq. (19) yields the *strong* form of the equations,

$$(\psi_p, \partial_i \mathcal{F}^i)_{\Omega_k} = (\psi_p, \partial_i \mathcal{F}^i)_{\Omega_k} + (\psi_p, (n_i \mathcal{F}^i)^* - n_i \mathcal{F}^i)_{\partial\Omega_k}. \tag{20}$$

We will make use of both the strong and the weak form to obtain symmetric DG operators (see Section III I). Approximating $\mathcal{F}^i$ using its expansion in basis functions (11) we find

$$(\psi_p, \partial_i \mathcal{F}^i)_{\Omega_k} \approx (\psi_p, \partial_i \psi_q)_{\Omega_k} \mathcal{F}^i_q = M\!\!D_{i,pq} \mathcal{F}^i_q, \tag{21}$$

where the *stiffness matrix* on the element Ω_k is

$$M\!\!D_{i,pq} \coloneqq (\psi_p, \partial_i \psi_q)_{\Omega_k} \tag{22a}$$

$$= \int_{[-1,1]^d} \psi_p(\boldsymbol{\xi}) \frac{\partial \psi_q}{\partial \xi^j}(\boldsymbol{\xi}) \, (\mathrm{J}^{-1})^j_i \sqrt{g} \, \mathrm{J} \, \mathrm{d}^d\xi \,. \tag{22b}$$

The divergence term in its weak form can be expressed in terms of the stiffness-matrix transpose $M\!\!D^T_{i,pq} = M\!\!D_{i,qp}$,

$$-(\partial_i \psi_p, \mathcal{F}^i)_{\Omega_k} \approx -(\partial_i \psi_p, \psi_q)_{\Omega_k} \mathcal{F}^i_q = -M\!\!D^T_{i,pq} \mathcal{F}^i_q. \tag{23}$$

Evaluation of the stiffness matrix and its transpose is discussed in Section III F.

We now turn towards discretizing the last remaining piece of the DG residuals, the boundary integrals in Eqs. (19) and (20). It involves a "lifting" operation: the integral only depends on field values on the element boundary but it may contribute to every component p of the DG residual, hence it is "lifted" to the volume. However, on an LGL grid all components p that correspond to grid points away from the boundary evaluate to zero because they contain at least one Lagrange polynomial that vanishes at the boundary collocation point. This is not the case on an LG grid, where evaluating the Lagrange polynomials on the boundary produces an interpolation into the volume. Expanding the boundary fluxes in basis functions (11) we find

$$(\psi_p, n_i \mathcal{F}^i)_{\partial\Omega_k} \approx (\psi_p, \psi_q)_{\partial\Omega_k} (n_i \mathcal{F}^i)_q = M\!\!L_{pq} (n_i \mathcal{F}^i)_q, \tag{24}$$

where we have defined the *lifting operator* on the element Ω_k,

$$M\!\!L_{pq} \coloneqq (\psi_p, \psi_q)_{\partial\Omega_k} \tag{25a}$$

$$= \int_{[-1,1]^{d-1}} \psi_p(\boldsymbol{\xi}) \psi_q(\boldsymbol{\xi}) \sqrt{g^\Sigma} \, \mathrm{J}^\Sigma \, \mathrm{d}^{d-1}\xi \,. \tag{25b}$$

Section III F provides details on evaluating the lifting operator.

Assembling the pieces of the discretization and restoring the index α that enumerates the equations, the DG residuals on the element Ω_k in strong form are

$$-M\!\!D_i \cdot {\mathcal{F}_\alpha}^i - M\!\!L \cdot ((n_i {\mathcal{F}_\alpha}^i)^* - n_i {\mathcal{F}_\alpha}^i) + M \cdot \mathcal{S}_\alpha = M \cdot f_\alpha, \tag{26a}$$

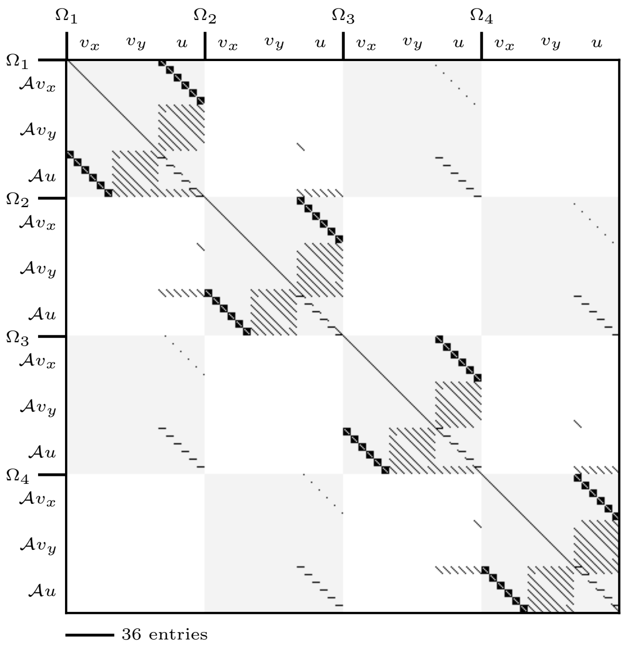


FIG. 2. Matrix representation of the noncompact DG operator in strong form (26a) for a two-dimensional Poisson equation (3). The computational domain is partitioned into 2×2 elements with 6×6 LGL grid points each. The image shows the nonzero entries of the operator matrix, i.e., its sparsity pattern, in the order laid out in Fig. 1b.

where $\cdot$ denotes a matrix multiplication with the field values over the computational grid of an element. The DG residuals in weak form are

$$M\!\!D^T_i \cdot {\mathcal{F}_\alpha}^i - M\!\!L \cdot (n_i {\mathcal{F}_\alpha}^i)^* + M \cdot \mathcal{S}_\alpha = M \cdot f_\alpha. \tag{26b}$$

We can choose either the strong or the weak form for each variable α.

Since the fluxes and sources are computed from the primal and auxiliary variables, the DG residuals (26) are algebraic equations for the discrete values $\underline{u}_A$ and $\underline{v}_A$ on all elements and grid points in the computational domain. The left-hand side of Eq. (26) is an operator $\mathcal{A}(\underline{u}_A, \underline{v}_A)$ and the right-hand side of Eq. (26) is a fixed value at every grid point, so Eq. (26) has the structure

$$\mathcal{A}(\underline{u}_A, \underline{v}_A) = \underline{b}. \tag{27}$$

If the fluxes and sources are linear, the DG operator $\mathcal{A}(\underline{u}_A, \underline{v}_A)$ can be represented as a square matrix, and Eq. (27) is a matrix equation. The size of the DG operator $\mathcal{A}(\underline{u}_A, \underline{v}_A)$ is the product of N_points with the number of both primal and auxiliary variables.

Figure 2 presents a visualization of the DG operator $\mathcal{A}(\underline{u}_A, \underline{v}_A)$ for a Poisson equation on a regular grid. The axes annotate entries of the operator that correspond to the "input" variables v_i and u, and to the corresponding "output" DG residuals. The mass matrix applied to v_i appears as a diagonal line (see Section III F) and the stiffness matrices applied to both v_i and u appear as block-diagonal and shaded regions for derivatives in x and y, respectively. The remaining entries represent the coupling between neighboring elements through the numerical flux (see Section III D). Note that the elements Ω_1 and Ω_4 as well as Ω_2 and Ω_3 decouple, because they share no boundaries as they are placed diagonally across the 2×2 grid of elements. Solving the Poisson equation amounts to inverting the matrix pictured in Fig. 2. However, it is significantly cheaper to invert the equivalent compact operator pictured in Fig. 3, which we derive in the following section.

C. Eliminating auxiliary degrees of freedom

So far we have treated the primal and the auxiliary equations of the first-order formulation on the same footing, which means the discretized DG operator applies to the primal variables as well as to the auxiliary variables. However, the auxiliary equations inflate the size of the operator significantly, increasing both its memory usage and the computational cost for solving it. In this section we eliminate the auxiliary degrees of freedom from the DG operator, demoting them to quantities that are only computed temporarily.

Many publications on DG formulations adopt a "primal formulation" to eliminate auxiliary degrees of freedom from the DG operator.[5] However, in practice we have found a simpler approach taking a Schur complement of the discretized equations in flux form, e.g., applied in Ref. [28], more suited to the generic implementation of DG schemes. The resulting DG operator remains equivalent to the original operator; i.e., it has the same solutions up to numerical precision. This strategy is facilitated by the auxiliary equations defining the auxiliary variables v_A, such as Eqs. (3a) and (5a). We assume here that the auxiliary fluxes depend only on the primal variables, ${\mathcal{F}_{v_A}}^i = {\mathcal{F}_{v_A}}^i[u_A; \boldsymbol{x}]$, and the auxiliary sources have the form

$$\mathcal{S}_{v_A} = v_A + \tilde{\mathcal{S}}_{v_A}[u_A; \boldsymbol{x}], \tag{28}$$

where $\tilde{\mathcal{S}}_{v_A}$ depends only on the primal variables. We further assume $f_{v_A} = 0$ for convenience. All elliptic systems that we consider in this article fulfill these assumptions. We insert Eq. (28) into the strong DG residuals (26a) and solve for v_A by inverting the mass matrix to find

$$v_A = D_i \cdot {\mathcal{F}_{v_A}}^i + L \cdot ((n_i {\mathcal{F}_{v_A}}^i)^* - n_i {\mathcal{F}_{v_A}}^i) - \tilde{\mathcal{S}}_{v_A}, \tag{29}$$

where we define $D_i := M^{-1} M\!\!D_i$ and $L := M^{-1} M\!\!L$. Note that the right-hand side of Eq. (29) depends only on the primal variables u_A. Evaluating the DG residuals now

[5] See, e.g., Section 7.2.2 in Ref. [2] or Section 3 in Ref. [6] for derivations of primal formulations for the Poisson equation.

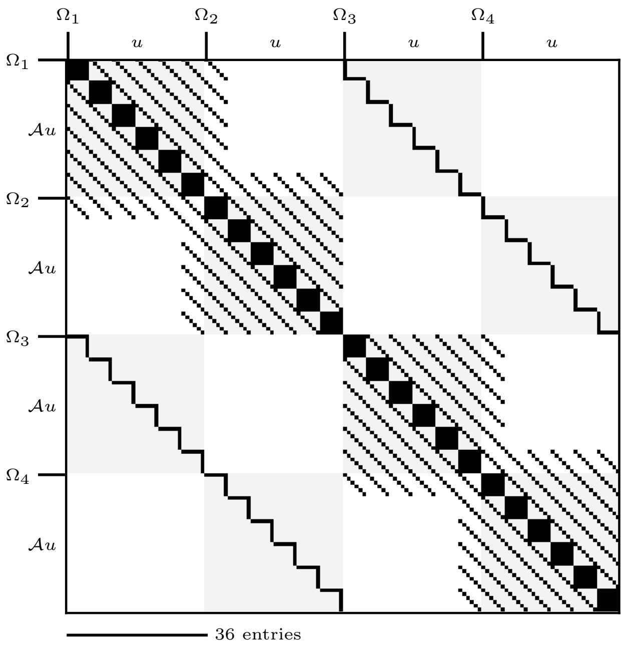


FIG. 3. Matrix representation of the compact DG operator in strong form (30a) for the two-dimensional Poisson problem detailed in Fig. 2. No auxiliary degrees of freedom inflate the size of the operator. This matrix is the Schur complement to the matrix pictured in Fig. 2.

amounts to first computing the auxiliary variables v_A by Eq. (29), and then using them to evaluate the primal equations. This approach preserves the freedom to use either the strong form (26a) for the primal equations,

$$-M\!\!D_i \cdot {\mathcal{F}_{u_A}}^i - M\!\!L \cdot ((n_i {\mathcal{F}_{u_A}}^i)^* - n_i {\mathcal{F}_{u_A}}^i) + M \cdot \mathcal{S}_{u_A} = M \cdot f_{u_A}, \tag{30a}$$

or the weak form (26b),

$$M\!\!D_i^T \cdot {\mathcal{F}_{u_A}}^i - M\!\!L \cdot (n_i {\mathcal{F}_{u_A}}^i)^* + M \cdot \mathcal{S}_{u_A} = M \cdot f_{u_A}. \tag{30b}$$

The strong scheme is slightly easier to implement because the primal and auxiliary equations involve the same set of operators. The strong-weak scheme, i.e., selecting the strong form for the auxiliary equations (29) and the weak form for the primal equations (30b), has the advantage that the DG operator can be symmetric as discussed in Section III I.

For linear equations the strategy employed in Eq. (29) of eliminating the auxiliary variables is equivalent to taking a Schur complement of the DG operator with respect to the (invertible) mass matrix, but the strategy works for nonlinear equations as well. The result is an operator $\mathcal{A}(\underline{u}_A)$ of only the discrete primal variables on all elements and grid points in the computational domain, so the DG residuals (30) have the form

$$\mathcal{A}(\underline{u}_A) = \underline{b}. \tag{31}$$

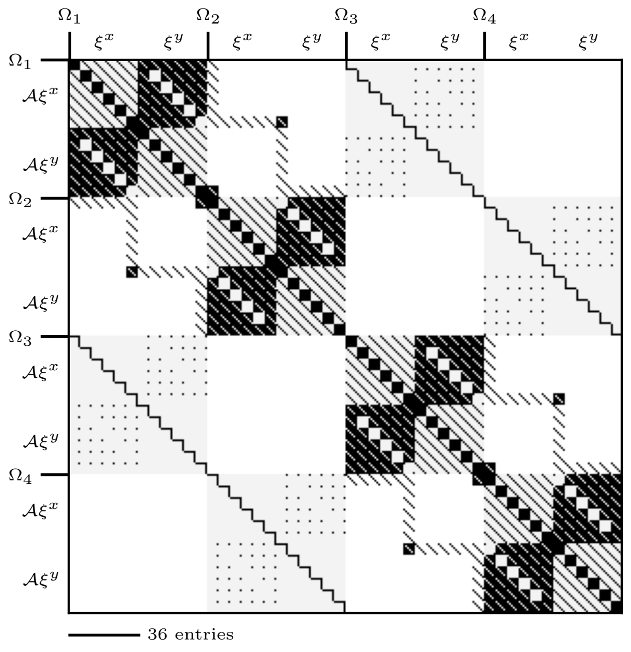


FIG. 4. Matrix representation of the compact DG operator in strong form (30a) for a two-dimensional elasticity problem (5) with an isotropic-homogeneous constitutive relation Y^{ijkl}. The computational domain is again partitioned into 2×2 elements with 6×6 LGL grid points each.

The size of the DG operator $\mathcal{A}(\underline{u}_A)$ is the product of N_points with the number of primal variables. No auxiliary degrees of freedom from the first-order formulation inflate the size of the operator. We refer to such DG operators $\mathcal{A}(\underline{u}_A)$ of only the primal degrees of freedom as *compact* if they also only involve couplings between nearest-neighbor elements [30]. The coupling between elements is related to the choice of numerical flux $(n_i\mathcal{F}_\alpha{}^i)^*$ and the subject of Section III D. If the fluxes and sources are linear, $\mathcal{A}(\underline{u}_A)$ can be represented as a square matrix.

Figures 3 and 4 present visualizations of $\mathcal{A}$ for a Poisson equation and an elasticity equation, respectively. The block-diagonal structure in Fig. 3 represents the DG-discretized two-dimensional Laplacian on the four elements of the computational domain. The entries that break the block-diagonal structure represent the coupling between nearest-neighbor elements through the numerical flux (Section III D).

D. A generalized internal-penalty numerical flux

Up to this point we have not made a choice for the numerical flux $(n_i\mathcal{F}^i)^*$ that combines double-valued field values on element boundaries. The numerical flux is a function of the field values on both sides of the boundary. From the perspective of one of the two adjacent elements we refer to the field values on itself as *interior* and to the field values the neighboring element as *exterior*. Contrary to much of the DG literature we formulate the numerical flux entirely in terms of the primal and auxiliary *boundary flux* quantities $n_i\mathcal{F}_{u_A}{}^i$ and $n_i\mathcal{F}_{v_A}{}^i$ on either side of the boundary instead of the primal and auxiliary variables u_A and v_A. This choice keeps our scheme applicable to the wide range of elliptic problems defined in Section II. The numerical flux presented here is a generalization of the symmetric internal penalty (SIP) scheme that is widely used in the literature [2, 6, 8, 31]. Our *generalized internal-penalty numerical flux* is

$$(n_i\mathcal{F}_{v_A}{}^i)^* = \frac{1}{2}\Big[n_i^\mathrm{int}\mathcal{F}_{v_A}{}^i(u_A^\mathrm{int}) - n_i^\mathrm{ext}\mathcal{F}_{v_A}{}^i(u_A^\mathrm{ext})\Big], \quad (32a)$$

$$\begin{aligned}(n_i\mathcal{F}_{u_A}{}^i)^* = \frac{1}{2}\Big[&n_i^\mathrm{int}\mathcal{F}_{u_A}{}^i\big(\partial_j\mathcal{F}_{v_A}{}^j(u_A^\mathrm{int}) - \tilde{\mathcal{S}}_{v_A}(u_A^\mathrm{int})\big)\\ &- n_i^\mathrm{ext}\mathcal{F}_{u_A}{}^i\big(\partial_j\mathcal{F}_{v_A}{}^j(u_A^\mathrm{ext}) - \tilde{\mathcal{S}}_{v_A}(u_A^\mathrm{ext})\big)\Big]\\ -\sigma\Big[&n_i^\mathrm{int}\mathcal{F}_{u_A}{}^i\big(n_j^\mathrm{int}\mathcal{F}_{v_A}{}^j(u_A^\mathrm{int})\big)\\ &- n_i^\mathrm{ext}\mathcal{F}_{u_A}{}^i\big(n_j^\mathrm{ext}\mathcal{F}_{v_A}{}^j(u_A^\mathrm{ext})\big)\Big]. \quad (32b)\end{aligned}$$

The numerical flux for the auxiliary equations, Eq. (32a), averages the boundary fluxes of the two adjacent elements. The numerical flux for the primal equations, Eq. (32b), is an average augmented with a penalty contribution with parameter σ.

Note that the numerical flux (32) involves only the primal fields and their derivatives, and thus is independent of the auxiliary fields altogether, as is typical for internal-penalty schemes. This has the practical advantage that the contribution from either side of the boundary to both the primal and the auxiliary numerical flux in Eqs. (29) and (30) can be computed early in the algorithm and communicated together, coupling only nearest-neighbor elements and thus making the DG operator compact. If the primal numerical flux (32b) depended on the auxiliary fields, evaluating the DG operator (30) would require a separate communication once the boundary corrections have been added to the auxiliary equation (29), effectively coupling nearest-neighbor elements as well as next-to-nearest-neighbor elements.[6]

DG literature usually assumes that the face normals on either side of the boundary are exactly opposite, $n_i^\mathrm{ext} = -n_i^\mathrm{int}$. This assumption breaks when the background geometry responsible for the normalization of face normals depends on the dynamic variables, since those are discontinuous across the boundary. All of the elliptic problems that we are expecting to solve in the near future are formulated on a fixed background geometry, but it is useful to distinguish between the interior and exterior face normals nonetheless because the quantity $n_i\mathcal{F}^i$ is cheaper

[6] Couplings to next-to-nearest-neighbor elements is a well-known disadvantage of LDG-type ("local discontinuous Galerkin") numerical fluxes and has led to the development of compact schemes such as Ref. [30].

to communicate than $\mathcal{F}^i$. Therefore, we always project an element's boundary fluxes onto the face normal before communicating the quantity.

For a simple flat-space Poisson system (3) our generalized internal-penalty numerical flux (32) reduces to the canonical SIP,[7]

$$(n_i\mathcal{F}_{v_j}{}^i)^* = n_j^\mathrm{int} u^* = \frac{1}{2} n_j^\mathrm{int}\big(u^\mathrm{int} + u^\mathrm{ext}\big) \tag{33a}$$

$$\begin{aligned}(n_i\mathcal{F}_{u}{}^i)^* = n_i^\mathrm{int} v^{i*} = \frac{1}{2} n_i^\mathrm{int}\big(\partial^i u^\mathrm{int} + \partial^i u^\mathrm{ext}\big) \\ - \sigma\big(u^\mathrm{int} - u^\mathrm{ext}\big).\end{aligned} \tag{33b}$$

As is well studied for the canonical SIP numerical flux, the penalty factor σ is responsible for removing zero eigenmodes and impacts the conditioning of the linear operator to be solved [2, 32]. It scales inversely with the element size h and quadratically with the polynomial degree p, both orthogonal to the element face.[8] Both h and p can be different on either side of the element boundary, so we choose

$$\sigma = C\,\frac{\big(\max(p^\mathrm{int}, p^\mathrm{ext}) + 1\big)^2}{\min(h^\mathrm{int}, h^\mathrm{ext})}, \tag{34}$$

where we follow Ref. [33] in choosing the scaling with the polynomial degree p on hexahedral meshes, and we follow Ref. [8] in our definition of the element size $h = 2/\sqrt{\tilde{n}_i\tilde{n}_j g^{ij}}$.[9] Note that h generally varies over the element face on curved meshes or on a curved manifold, and that the min operation in Eq. (34) is taken pointwise, so σ also varies over the element face. The remaining *penalty parameter* $C \geq 1$ remains freely specifiable. Note also that we do not need to include a problem-specific scale in the penalty factor, as is done in Refs. [25–27], because the generic numerical flux (32b) already includes such scales in the fluxes $\mathcal{F}^i$.

E. Imposing boundary conditions

The flux formulation allows imposing a wide range of boundary conditions relatively easily "through the fluxes" without the need to treat external boundaries any differently than internal boundaries between neighboring elements. Imposing boundary conditions amounts to specifying the exterior quantities in the numerical flux, Eq. (32). This strategy is often referred to as imposing boundary conditions through "ghost" elements. As suggested in, e.g., Ref. [2], we impose boundary conditions on the *average* of the boundary fluxes to obtain faster convergence. Therefore, on external boundaries, we choose for the exterior quantities in the numerical flux (32)

$$(n_i\mathcal{F}_\alpha{}^i)^\mathrm{ext} = (n_i\mathcal{F}_\alpha{}^i)^\mathrm{int} - 2(n_i\mathcal{F}_\alpha{}^i)^\mathrm{b}, \tag{35}$$

where we set the quantities $(n_i\mathcal{F}_\alpha{}^i)^\mathrm{b}$ according to the boundary conditions at hand. Here we define $n_i^\mathrm{b} = n_i^\mathrm{int}$, i.e., we choose to compute external boundary fluxes with the face normal pointing *out* of the computational domain. The symmetry between the primal and the auxiliary equations in the first-order flux formulation (1) that we employ throughout this article makes this approach of imposing boundary conditions particularly straightforward: a choice of *auxiliary* boundary fluxes $(n_i\mathcal{F}_{v_A}{}^i)^\mathrm{b}$ imposes Dirichlet-type boundary conditions and a choice of *primal* boundary fluxes $(n_i\mathcal{F}_{u_A}{}^i)^\mathrm{b}$ imposes Neumann-type boundary conditions. The choice between Dirichlet-type and Neumann-type boundary conditions can be made separately for every primal variable u_A and every external boundary face, simply by setting either $(n_i\mathcal{F}_{u_A}{}^i)^\mathrm{b}$ or $(n_i\mathcal{F}_{v_A}{}^i)^\mathrm{b}$ and setting the remaining boundary fluxes to their interior values $(n_i\mathcal{F}^i)^\mathrm{b} = (n_i\mathcal{F}^i)^\mathrm{int}$. Note that we neither need to distinguish between primal and auxiliary variables in Eq. (35), nor take the choice of Dirichlet-type or Neumann-type boundary conditions into account, but we require only that $(n_i\mathcal{F}^i)^\mathrm{b}$ be chosen appropriately for every variable. Then, the Neumann-type boundary conditions enter the DG residuals directly through the numerical flux in Eq. (30), and the Dirichlet-type boundary conditions enter the DG residuals through the numerical flux in Eq. (29), which is substituted in Eq. (30).

In practice, this setup means we can initialize $(n_i\mathcal{F}_\alpha{}^i)^\mathrm{b} = (n_i\mathcal{F}_\alpha{}^i)^\mathrm{int}$ for all variables on a particular external boundary face when preparing to apply the numerical flux, decide which boundary fluxes to modify based on the boundary conditions we wish to impose on the particular face, and then evaluate Eq. (35) to compute the exterior quantities in the numerical flux (32). To impose Neumann-type boundary conditions we set the primal boundary fluxes $(n_i\mathcal{F}_{u_A}{}^i)^\mathrm{b}$ directly, but to impose Dirichlet-type boundary conditions we typically choose the primal field values u_A^b and compute the auxiliary boundary fluxes as $(n_i\mathcal{F}_{v_A}{}^i)^\mathrm{b} = n_i^\mathrm{int}\mathcal{F}_{v_A}{}^i(u_A^\mathrm{b})$.

The auxiliary (Dirichlet-type) external boundary fluxes may depend on the interior primal fields u_A^int, and the primal (Neumann-type) external boundary fluxes may depend on both the interior primal fields u_A^int as well as the interior auxiliary fields v_A^int. This means we can impose a wide range of boundary conditions that may depend linearly or nonlinearly on the dynamic fields. For example, a Robin boundary condition for the Poisson equation (2) or (6),

$$a\,u + b\,n^i\partial_i u = g(\boldsymbol{x}) \quad \text{on } \partial\Omega, \tag{36}$$

where a and b are constants and $g(\boldsymbol{x})$ is a function defined on the boundary, can be implemented as Neumann-type

[7] See Eq. (3.21) in Ref. [6] or Section 7.2 in Ref. [2].

[8] See Ref. [32] for sharp results for the optimal penalty factor on triangular and tetrahedral meshes, and Table 3.1 in Ref. [33] for a generalization to hexahedral meshes.

[9] Note that Ref. [8] omits the factor of 2 in the definition of h, which we include so the definition reduces to the canonical element size on rectilinear meshes.

for $b \neq 0$,

$$(n_i \mathcal{F}_u{}^i)^\mathrm{b} = \frac{1}{b}\big(g(\boldsymbol{x}) - a\, u^\mathrm{int}\big), \tag{37}$$

and as Dirichlet-type for $b = 0$,

$$(n_i \mathcal{F}_v{}^i)^\mathrm{b} = n_i^\mathrm{int} \mathcal{F}_v{}^i(u^\mathrm{b}) \quad \text{with} \quad u^\mathrm{b} = \frac{1}{a} g(\boldsymbol{x}). \tag{38}$$

An important consideration is that boundary conditions are generally nonlinear. Even for linear PDEs, such as the Poisson equation, a simple inhomogeneous Dirichlet boundary condition $u^\mathrm{b} \neq 0$ introduces a nonlinearity in the DG operator because $\mathcal{A}(0) \neq 0$. Therefore, we always linearize boundary conditions. For nonlinear equations the boundary conditions linearize along with the DG operator and require no special attention (see Section III J). However, for linear equations the inhomogeneity in the boundary conditions is the only nonlinearity present in the DG operator, so we skip the full linearization procedure. Instead, we contribute the inhomogeneous boundary conditions $\mathcal{A}(0)$ to the fixed sources, leaving only the linearized boundary conditions in the DG operator,

$$\frac{\delta \mathcal{A}}{\delta u} \underline{u} = \underline{b} - \mathcal{A}(0), \tag{39}$$

where $\frac{\delta \mathcal{A}}{\delta u}$ is just $\mathcal{A}$ with linearized boundary conditions. Note that this strategy is equivalent to the full linearization procedure described in Section III J at $\underline{u} = 0$. In practice, evaluating $\mathcal{A}(0)$ simplifies significantly for linear equations because only the lifted external boundary corrections contribute to it.

F. Evaluating the mass, stiffness, and lifting matrices

The mass matrix (16), the stiffness matrix (22), and the lifting operator (25) are integrals that must be evaluated on every element of the computational domain. We evaluate these integrals on the same grid on which we expand the dynamic fields, which amounts to a Gauss-Lobatto quadrature of an order set by the number of collocation points in the element. This strategy is commonly known as *mass lumping*.[10] Employing mass lumping and our choice of nodal basis (10), the mass matrix (16) evaluates to

$$M_{pq} \approx \delta_{pq} \sqrt{g}\big|_p \, \mathrm{J}\big|_p \prod_{i=1}^{d} w_{p_i}. \tag{40}$$

[10] This is the approach taken in Ref. [12]. See Eq. (3.7) in Ref. [12] for details on the mass-lumped mass matrix on d-dimensional hexahedral elements. Note that Ref. [12] absorbs the metric determinant in the dynamic variables.

Here the coefficients w_{p_i} denote the Legendre-Gauss-Lobatto quadrature weights associated with the collocation points ξ_{p_i}, and the geometric quantities $\sqrt{g}$ and J are evaluated directly on the collocation points.[11] Recall from Section III A that the index p enumerates grid points in the regular d-dimensional grid and that p_i denotes the grid point's index along the ith dimension. The diagonal mass-lumping approximation (40) has the advantage that it is computationally efficient to apply, invert and store since it amounts to a pointwise multiplication over the computational grid. Note that Eq. (40) is exact on a rectilinear LG grid with a flat background geometry, and can be made exact on rectilinear LGL grids by including a correction term without increasing the computational cost for applying or inverting it [34]. The quadrature weights w_{p_i} can be cached and reused by all elements with the same number of collocation points in a dimension.

The strong stiffness matrix (22) evaluates to

$$MD_{i,pq} \approx M_{pr} D_{i,rq}, \tag{41a}$$

with

$$D_{i,rq} = \sum_{j=1}^{d} (\mathrm{J}^{-1})^j_i \Big|_r \, \ell'_{q_j}(\xi_{r_j}) \prod_{\substack{k=1\\k\neq j}}^{d} \delta_{q_k r_k}. \tag{41b}$$

Here $\ell'_{q_j}(\xi_{r_j})$ are the one-dimensional "logical" differentiation matrices obtained by differentiating the Lagrange polynomials (9) and evaluating them at the collocation points.[12] The stiffness matrix is essentially a "massive" differentiation operator that decomposes into one-dimensional differentiation matrices due to the product structure of the basis functions (10) on our hexahedral meshes. The one-dimensional logical differentiation matrices can be cached and reused by all elements with the same number of collocation points in a dimension, keeping the memory cost associated with the stiffness operator to a minimum. The weak stiffness matrix can be computed analogously from the transpose of the logical differentiation matrices.

The lifting operator (25) evaluates to

$$ML_{pq} \approx M_{pr} L_{rq}, \tag{42a}$$

with

$$L_{rq} = \delta_{rq} \sum_{i=1}^{d} (\delta_{q_i 1} + \delta_{q_i N_{k,i}}) \frac{1}{w_{q_i}} \sqrt{g^{jk} (\mathrm{J}^{-1})^i_j (\mathrm{J}^{-1})^i_k}\Big|_q . \tag{42b}$$

[11] See, e.g., Algorithm 25 in Ref. [29] for details on computing Legendre-Gauss-Lobatto quadrature weights, and Algorithm 23 for Legendre-Gauss quadrature weights.

[12] See, e.g., Ref. [2] and Algorithm 37 in Ref. [29] for details on computing the one-dimensional logical differentiation matrix $\ell'_{q_j}(\xi_{r_j})$ from properties of Legendre polynomials.

It is diagonal and has a contribution from every face of the element. Note that each face only contributes to the LGL grid points on the respective face. On LG grids additional interpolation matrices from the face into the volume appear in this expression. Also note that the root in Eq. (42b) is simply the magnitude of the unnormalized face normal $\tilde{n}_j$ [12].

Recall that the objective of these matrices is to evaluate the compact DG operator (30) along with Eq. (29) on every element of the computational domain. In practice, neither matrix must be assembled explicitly and stored on the elements: the mass matrix (40) reduces to a multiplication over the computational grid, the stiffness matrix (41) involves logical one-dimensional differentiation matrices that are shared between elements, and the lifting operation (42) reduces to a multiplication over the grid points on the element face. Since both the stiffness and the lifting operation decompose into a mass matrix and a "massless" operation, the same set of operations can be used to evaluate both Eqs. (29) and (30), and the mass matrix factors out of the DG operator entirely. Nevertheless, we will see in Section III I that it is advantageous to keep the mass matrix in the operator (30).

G. A note on dealiasing

The integral expressions discussed in Section III F involve geometric quantities that are typically known analytically, namely the Jacobian and the background metric. Limiting the resolution of the integrals to the quadrature order of the elements can make the scheme susceptible to geometric aliasing because these quantities are resolved with limited precision, potentially reducing the accuracy of the scheme on curved meshes or on a curved manifold. To combat geometric aliasing we can, in principle, precompute the matrices on every element at high accuracy, but at a significant memory cost. Precomputing the matrices is possible in elliptic problems because the geometric quantities remain constant. This is different to time-evolution systems that often involve time-dependent Jacobians ("moving meshes"). Alternatively, a number of dealiasing techniques are available to combat geometric aliasing, and also to combat aliasing arising from evaluating other background quantities on the collocation points, i.e., quantities in the PDEs that are independent of the dynamic variables and known analytically [35]. For example, Ref. [8] interpolates data from the primary LGL grid to an auxiliary LG grid, on which the Jacobian is evaluated, to take advantage of the higher-order quadrature. However, these dealiasing techniques can significantly increase the computational cost for applying the DG operator. We have chosen to employ the simple mass-lumping scheme detailed in Section III F to minimize the complexity, computational cost, and memory consumption of the DG operator. Detailed studies of cost-efficient dealiasing techniques are a possible subject of future work.

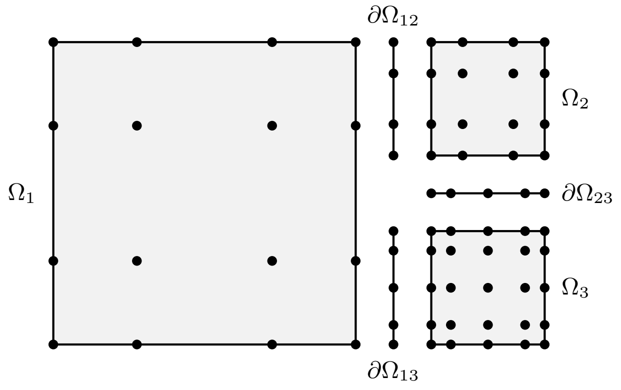


FIG. 5. A representation of nonconforming element boundaries with mortars in two dimensions. The left element Ω_1 faces two elements Ω_2 and Ω_3 towards the right. Since both Ω_1 and Ω_2 have four vertical grid points their shared mortar $\partial\Omega_{12}$ also has four grid points, but covers only a logical half of Ω_1 (h nonconforming). The element Ω_3 has five vertical grid points, so the mortar $\partial\Omega_{13}$ has $\max(4,5) = 5$ grid points and also covers only a logical half of Ω_1 (hp nonconforming). Elements Ω_2 and Ω_3 are h conforming but differ in their number of horizontal grid points, so their shared mortar $\partial\Omega_{23}$ has $\max(4,5) = 5$ grid points (p nonconforming). Note that the empty space between the elements in this visualization is not part of the computational domain.

H. Mesh refinement

The domain decomposition into elements, each with their own set of basis functions, allows for two avenues to control the resolution: we can split the domain into more and smaller elements (h refinement) or increase the number of basis functions within an element (p refinement). We can perform h and p refinement in each dimension independently.

Both h refinement and p refinement can lead to nonconforming boundaries between elements, meaning that grid points on the two sides of the boundary do not coincide. Since we need to work with data from both sides of an element boundary when considering numerical fluxes (see Section III D) we place *mortars* between elements. A mortar is a $(d-1)$-dimensional mesh that has sufficient resolution to exactly represent discretized fields from both adjacent element faces. Specifically, a mortar $\partial\Omega_{k\bar{k}}$ between the elements Ω_k and $\Omega_{\bar{k}}$ that share a boundary orthogonal to dimension j has $\max(N_{k,i}, N_{\bar{k},i})$ grid points in dimension $i \neq j$. We limit the h refinement of our computational domains such that an element shares its boundary with at most two neighbors per dimension in every direction ("two-to-one balance"). This means a mortar covers either the full element face or a logical half of it in every dimension. Figure 5 illustrates an hp-refined scenario with nonconforming element boundaries.

To project field values from an element face to a mortar we employ the $(d-1)$-dimensional *prolongation operator*

$$P_{\tilde{p}p} = \prod_{i=1}^{d-1} \ell_{p_i}(\tilde{\xi}_{\tilde{p}_i}), \tag{43}$$

where p enumerates grid points on the coarser (element face) mesh, $\tilde{p}$ enumerates grid points on the finer (mortar) mesh, and $\tilde{\xi}_{\tilde{p}_i}$ are the coarse-mesh logical coordinates of the fine-mesh collocation points. For mortars that cover the full element face in dimension i the coarse-mesh logical coordinates are just the fine-mesh collocation points, $\tilde{\xi}_{\tilde{p}_i} = \xi_{\tilde{p}_i}$. For mortars that cover the lower or upper logical half of the element face in dimension i they are $\tilde{\xi}_{\tilde{p}_i} = (\xi_{\tilde{p}_i} - 1)/2$ or $\tilde{\xi}_{\tilde{p}_i} = (\xi_{\tilde{p}_i} + 1)/2$, respectively. Note that the prolongation operator (43) is just a Lagrange interpolation from the coarser (element face) mesh to the finer (mortar) mesh. The interpolation retains the accuracy of the polynomial approximation because the mortar has sufficient resolution. The prolongation operator is also an L_2 projection (or *Galerkin* projection) because it minimizes the L_2 norm $\int_{\partial\Omega_{k\bar{k}}} (u^{(k)} - u^{(\tilde{k})})^2 \sqrt{g}\,\mathrm{d}^{d-1}x$, where $u^{(\tilde{k})}$ denotes the prolongated field values on the finer (mortar) mesh.

To project field values from a mortar back to an element face we employ an adjoint R of the prolongation operator such that $RP = \mathbb{1}$. We also refer to this operation as a *restriction* because it truncates higher modes from the mortar down to the resolution of the element face. Specifically, we employ the mass-conservative adjoint

$$\begin{aligned}\int_{\partial\Omega_{k\bar{k}}} & R(u^{(\tilde{k})})\, u^{(k)} \sqrt{g}\,\mathrm{d}^{d-1}x \\ &= \int_{\partial\Omega_{k\bar{k}}} u^{(\tilde{k})} P(u^{(k)}) \sqrt{g}\,\mathrm{d}^{d-1}x \quad \forall u^{(k)}, u^{(\tilde{k})}.\end{aligned} \tag{44}$$

In matrix notation the *restriction operator* reduces to

$$R = M^{-1} P^T \tilde{M}, \tag{45}$$

where M^{-1} is the inverse mass matrix on the coarser (element face) mesh, $\tilde{M}$ is the mass matrix on the finer (mortar) mesh, and P^T is the transpose of the prolongation operator (43).

Note that the d-dimensional restriction and prolongation operators can serve not only to project field values to and from mortars, but also to project field values to and from elements that cover the computational domain at different h- and p-refinement levels. We make no use of projections across refinement levels in this article but will do so in upcoming work for the purpose of adaptive mesh-refinement strategies and for multigrid solvers.[13]

[13] See also Sections 3.2 and 3.3 in Ref. [28] for details on the restriction and prolongation operators in the context of multigrid solvers.

I. A note on symmetry

For practical applications it is often advantageous to work with a symmetric operator. For example, some iterative linear solvers such as conjugate gradients take advantage of the symmetry to invert the operator more efficiently. One can also often show stronger convergence bounds for iterative linear solvers applicable to nonsymmetric matrices, such as GMRES, if the matrix is symmetric [36].

The compact strong-weak scheme presented in Eq. (30b) with the generalized SIP numerical flux (32) is symmetric unless the elliptic equations break the symmetry, e.g., with an asymmetric coupling between equations. Note that a curved manifold will typically break the symmetry because it involves first derivatives in Christoffel-symbol contributions to the primal sources [see, e.g., Eq. (7)]. It is straightforward to see how the strong-weak scheme can make the DG operator symmetric if the elliptic equations allow it: the strong-weak operator involves a symmetric stiffness term of the schematic form $(MD)^T D = D^T MD$, whereas the strong scheme has a nonsymmetric expression of the form MDD instead. Note that the "massless" variant of the strong-weak scheme, schematically $M^{-1}D^T MD$, is not generally symmetric, and neither is the "massless" strong scheme DD.

J. Linearizing the operator

To solve nonlinear elliptic equations $\mathcal{A}(\underline{u}) = \underline{b}$ we typically employ a correction scheme, repeatedly solving the linearized equations for a correction quantity $\Delta\underline{u}$. For example, a simple Newton-Raphson correction scheme solves the linearized problem $\frac{\delta\mathcal{A}}{\delta u}(\underline{u})\,\Delta\underline{u} = \underline{b} - \mathcal{A}(\underline{u})$ at fixed $\underline{u}$ and then iteratively corrects $\underline{u} \to \underline{u} + \Delta\underline{u}$. Since the fluxes $\mathcal{F}_\alpha{}^i$ are already linear for all elliptic systems we consider, the linearization $\frac{\delta\mathcal{A}}{\delta u}(\underline{u})$ involves only linearizing the sources $\mathcal{S}_\alpha$ and the boundary conditions.

K. Variations of the scheme

We have made a number of choices to formulate the DG discretization in the preceding sections. This section summarizes some of the choices and presents possible variations to explore in future work.

Massive vs massless scheme: We can eliminate the mass matrix in Eq. (30) to obtain a "massless" DG operator. However, we have found evidence that iterative linear solvers converge faster when solving the "massive" DG operator. We attribute this behaviour to the symmetry considerations discussed in Section III I.

Mass-lumping: We diagonally approximate the mass matrix to reduce the computational cost to apply, invert and store it, and to simplify the scheme (see Section III F). Dealiasing techniques can potentially increase the accuracy of the scheme on curved meshes as discussed in Section III G.

LGL vs LG mesh: We chose to discretize the DG operator on LGL meshes to take advantage of the collocation points on element boundaries, which simplify computations of boundary corrections. Switching to LG meshes can have the advantage that quadratures are one degree more precise, making the mass-lumping exact on rectilinear grids (see Section III F).

Numerical flux: The generalized internal-penalty numerical flux presented in Section III D has proven a viable choice for a wide range of problems so far. However, the ability to switch out the numerical flux is a notable strength of DG methods, and augmenting the numerical flux in the elliptic DG scheme may improve its convergence properties or accuracy. In particular, the choice of penalty, Eq. (34), on curved meshes remains a subject of further study.

Strong vs weak formulation: We have chosen the strong formulation (30a) over the strong-weak formulation (30b) because it is slightly simpler and we have, so far, found no evidence that the strong-weak formulation converges faster than the strong formulation, despite the symmetry considerations discussed in Section III I. However, the strong-weak formulation can be of interest if a symmetric DG operator is necessary, e.g., to take advantage of specialized iterative solvers.

Flux vs primal formulation: We have eliminated auxiliary degrees of freedom in the DG operator with a Schur-complement strategy. An alternative strategy is to derive a "primal formulation" of the DG operator (see Section III C). We have found the flux formulation easier to implement due to its similarity to hyperbolic DG schemes. Furthermore, Ref. [28] suggests that the flux formulation can be advantageous in conjunction with a multigrid solver.

IV. TEST PROBLEMS

The following numerical tests confirm the DG scheme presented in this article can solve a variety of elliptic problems. The test problems involve linear and nonlinear systems of PDEs with nonlinear boundary conditions on curved manifolds, discretized on hp-refined domains with curved meshes and nonconforming element boundaries.

For test problems that have an analytic solution we quantify the accuracy of the numerical solutions by computing an L_2 error over all primal variables,

$$\|u - u_{\mathrm{analytic}}\| := \left(\frac{\sum_{A,k}\int_{\Omega_k}(u_A - u_{A,\mathrm{analytic}})^2\,\mathrm{d}V}{\sum_k\int_{\Omega_k}\mathrm{d}V}\right)^{1/2}, \tag{46}$$

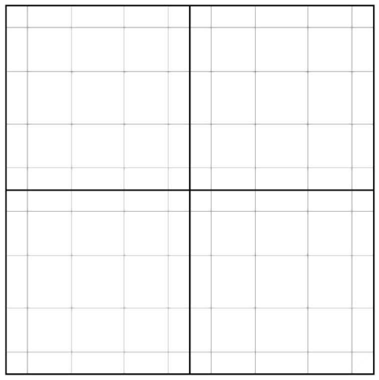


FIG. 6. The two-dimensional rectilinear domain used in the Poisson problem (Section IV A). Black lines illustrate element boundaries and gray lines represent the LGL grid within each element. This domain is isotropically h refined once, i.e., split once in both dimensions, resulting in four elements. Each element has six grid points per dimension, so fields are represented as polynomials of degree five. This is the domain that Figs. 2 and 3 are based on, and that is circled in Fig. 7.

where the integrals are evaluated with Gauss-Lobatto quadrature on the elements of the computational domain.

To assess the DG operator is functional for our test problems we study the convergence of the discretization error (46) under uniform hp refinement of the computational domain (see Section III H). We compute the h-convergence order under pure uniform h refinement

$$\tau_h := \frac{\Delta_h \ln(\|u - u_{\mathrm{analytic}}\|)}{\Delta_h \ln(h)}, \tag{47}$$

where Δ_h denotes the difference between successive h-refinement levels and h is the size of an element. Since we always split elements in half along all logical axes we use $\Delta_h \ln(h) = \ln(2)$. We also compute the exponential convergence scale under pure uniform p refinement

$$\tau_p := \Delta_p \log_{10}(\|u - u_{\mathrm{analytic}}\|), \tag{48}$$

where Δ_p denotes the difference between successive p-refinement levels.

A. A Poisson solution

With this first test problem we establish a simple baseline that the following tests build upon. It is reduced to the absolute essentials to illustrate the basic concepts of the scheme. We solve a flat-space Poisson equation (2) in two dimensions for the analytic solution

$$u_{\mathrm{analytic}}(\boldsymbol{x}) = \sin(\pi x)\sin(\pi y) \tag{49}$$

on a rectilinear domain $\Omega = [0,1]^2$. The domain is illustrated in Fig. 6. To obtain the solution (49) numerically we choose the fixed source $f(\boldsymbol{x}) = 2\pi^2 \sin(\pi x)\sin(\pi y)$,

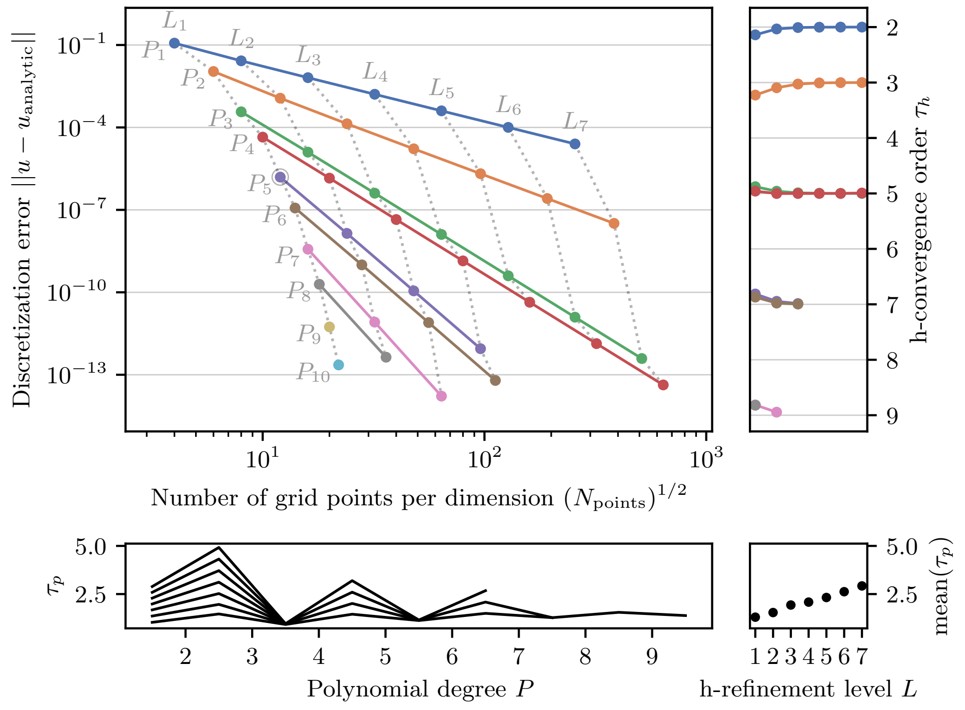


FIG. 7. Convergence of the two-dimensional Poisson problem detailed in Section IV A with uniform hp refinement. Solid lines connect numerical solutions where the domain is split into an increasing number of elements (isotropic h refinement), and dotted lines connect numerical solutions with increasing polynomial order (isotropic p refinement). The DG scheme recovers optimal $\mathcal{O}(h^{P+1})$ convergence with odd-order superconvergence under h refinement (right panel) and exponential convergence under p refinement (bottom panels). For reference, the circled configuration is pictured in Fig. 6.

select homogeneous Dirichlet boundary conditions $u^{\mathrm{b}} = 0$, and solve the strong compact DG-discretized problem (30a) with $C = 1$. This essentially means we invert the matrix depicted in Fig. 3 and apply it to the discretization of the fixed source $f(\boldsymbol{x})$. Instead of inverting the matrix directly we employ the iterative elliptic solver of the `SpECTRE` code [22] presented in Ref. [23]. However, note that the technology we use to solve the DG-discretized problem is not relevant for the purpose of this article, since the matrix equation has a unique solution. Assuming the matrix equation is solved to sufficient precision, Eq. (46) quantifies the discretization error of the DG scheme.

We solve the problem on a series of uniformly and isotropically refined domains and present the convergence of the discretization error in Fig. 7. Under h refinement the scheme recovers optimal $\mathcal{O}(h^{P+1})$ convergence, where P denotes the polynomial degree of the elements. It also recovers the odd-order superconvergence feature expected for the antisymmetric problem (49).[14] Under p refinement the scheme recovers exponential convergence. The exponential convergence scale τ_p is modulated by the superconvergence feature and its mean increases linearly with the h-refinement level.

[14] See also Fig. 7.9 in Ref. [2].

Beam width	r_0	177 µm
Outer radius	R	600 µm
Poisson ratio	ν	0.17
Young's modulus	E	72 GPa

TABLE I. Parameters used in the thermal-noise problem (Section IV B). The beam width and the material properties correspond to Table 1 in Ref. [37]. These material properties characterize a fused-silica mirror, which is a material used in the LIGO gravitational-wave detectors.

B. Thermal noise in a cylindrical mirror

In this second test problem we solve the equations of linear elasticity (4) on a curved mesh with nonconforming element boundaries. The test problem represents a cylindrical mirror that is deformed by pressure from a laser beam incident on one of the sides. This problem arises in studies of Brownian thermal noise in interferometric

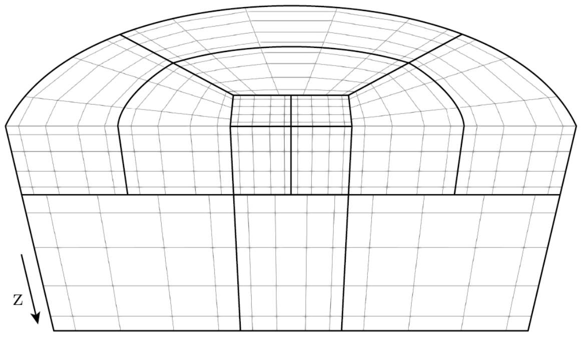


FIG. 8. A cut through the cylindrical domain used in the elasticity problem (Section IV B). The domain consists of four wedges enveloping a cuboid, and two vertical layers. The layers are partitioned vertically at $z = r_0$ and the cuboid lies radially within $r = r_0$. In the top layer, the wedges are h refined radially once and the cuboid is h refined in the x and y directions once, resulting in 12 elements in the top layer and 5 elements in the bottom layer. Elements in this example have six grid points per dimension, and the wedge-shaped elements have two additional grid points in their angular direction.

gravitational-wave detectors [37, 38].[15] Here we consider an analytic solution to this problem that applies in the limit of an isotropic and homogeneous mirror material with constitutive relation

$$Y^{ijkl} = \lambda\,\delta^{ij}\delta^{kl} + \mu\left(\delta^{ik}\delta^{jl} + \delta^{il}\delta^{jk}\right), \tag{50}$$

characterized by the Lamé parameter λ and the shear modulus μ, or equivalently by the Poisson ratio $\nu = \frac{\lambda}{2(\lambda+\mu)}$, Young's modulus $E = \frac{\mu(3\lambda+2\mu)}{\lambda+\mu}$, or the bulk modulus $K = \lambda + \frac{2}{3}\mu$. We assume the material fills the infinite half-space $z \geq 0$, choose a vanishing force density $f^j(x) = 0$, and a Gaussian profile of the laser beam incident at $z = 0$,

$$n_i T^{ij} = n^j \frac{1}{\pi r_0^2} e^{-r^2/r_0^2}. \tag{51}$$

Here $T^{ij} = -Y^{ijkl} S_{kl}$ is the *stress*, n_i is the unit normal pointing away from the mirror, i.e., in negative z direction, $r = \sqrt{x^2+y^2}$ is the radial coordinate distance from the axis of symmetry, and r_0 is the beam width. Under these assumptions the displacement field $\xi^i(\boldsymbol{x})$ has the analytic solution [39–41]

$$\xi^r = \frac{1}{2\mu}\int_0^\infty \mathrm{d}k\, J_1(kr) e^{-kz}\left(1 - \frac{\lambda+2\mu}{\lambda+\mu} + kz\right)\tilde{p}(k). \tag{52a}$$

$$\xi^z = \frac{1}{2\mu}\int_0^\infty \mathrm{d}k\, J_0(kr) e^{-kz}\left(1 + \frac{\mu}{\lambda+\mu} + kz\right)\tilde{p}(k) \tag{52b}$$

[15] See also Section 11.9.2 in Ref. [39] for an introduction to the thermal noise problem.

and $\xi^\phi = 0$ in cylindrical coordinates $\{r, \phi, z\}$. Here J_0 and J_1 are Bessel functions of the first kind, and $\tilde{p}(k) = \frac{1}{2\pi} e^{-(kr_0/2)^2}$ is the Hankel transform of the laser-beam profile. We evaluate these integrals numerically at every collocation point in the computational domain to determine the analytic solution.

To obtain numerical solutions to the thermal noise problem we DG discretize the equations of linear elasticity (5) on a cylindrical domain with height and radius R, employing the strong compact DG operator (30a). Since the stress $T^{ij} = -\mathcal{F}_\xi{}^{ij}$ is the negative primal flux in the elasticity equations (5) we impose Eq. (51) as a Neumann-type boundary condition on the base of the cylinder at $z = 0$. We impose the analytic solution (52) as Dirichlet-type boundary conditions on the remaining external boundaries of the domain, i.e., on the base at $z = R$ and on the mantle at $r = R$. These boundary conditions mean that we solve for a finite cylindrical section of the infinite half-space analytic solution (52). We choose a penalty parameter of $C = 100$ for this problem to eliminate variations in the discretization error arising from curved-mesh contributions to the penalty (34) at high resolutions. Table I summarizes the remaining parameters we use in the numerical solutions.

Figure 8 illustrates the cylindrical domain. It is refined more strongly toward the origin $\boldsymbol{x} = 0$ where the Gaussian laser beam applies pressure. The refinement is both anisotropic and inhomogeneous, leading to nonconforming element boundaries with different polynomial degrees on either side of the boundary, multiple neighbors adjacent to an element face, or both. Specifically, elements facing the top-layer cuboid or the interface between top and bottom layer are matched two-to-one, and wedge-shaped elements have two additional angular grid points. Therefore, the elements facing the cuboid are both p nonconforming and h nonconforming in the top layer, and p nonconforming in the bottom layer. The elements facing the layer interface are h nonconforming.

Figure 9 presents the convergence of the discretization error under uniform hp refinement. Specifically, we split every element in two along all three dimensions to construct additional h-refinement levels, and increment every polynomial degree by one to construct additional p-refinement levels, retaining the nonconforming element boundaries. Note that the wedge-shaped elements retain a higher polynomial degree of $P+2$ along their angular direction throughout the refinement procedure, where P is the polynomial degree of all other elements and dimensions. The DG scheme recovers optimal $\mathcal{O}(h^{P+1})$ convergence under h refinement and exponential convergence under p refinement. Note that the exponential convergence scale τ_p depends on the domain geometry, the structure of the solution, the placement of grid points and the refinement strategy. We have chosen to refine the domain as uniformly as possible here to reliably measure convergence properties of the DG scheme. Optimizing the distribution of elements and grid points with adaptive mesh refinement (AMR) strategies to increase the rate of

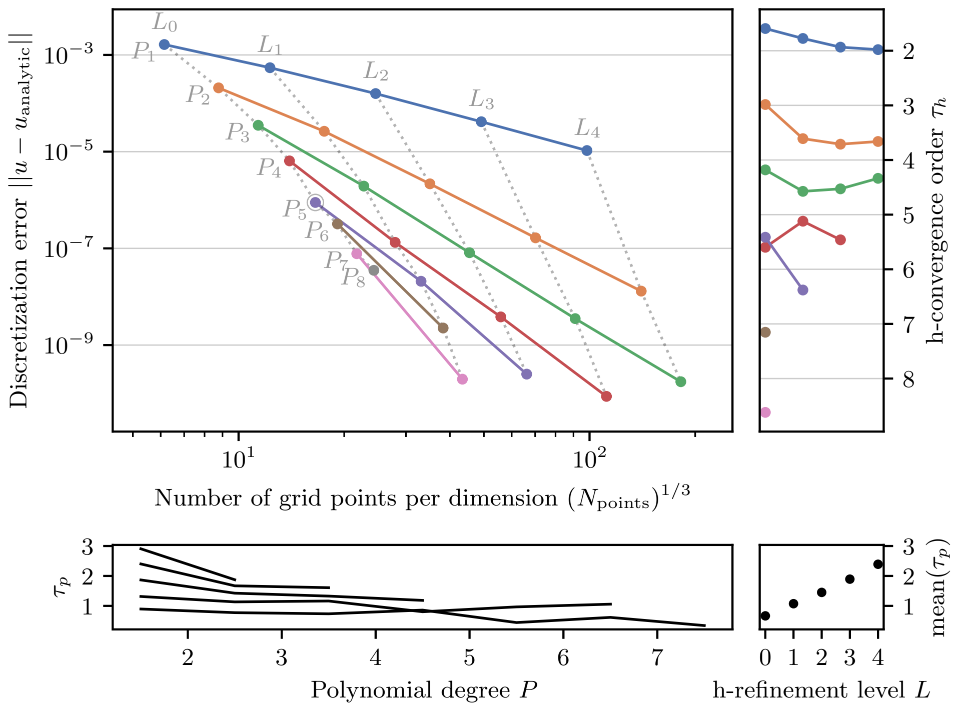


FIG. 9. Convergence of the three-dimensional elasticity problem detailed in Section IV B under uniform hp refinement. Plotted is the L_2 error (46) over the three components of the displacement field $\xi^i(\boldsymbol{x})$. The refinement is based on the domain pictured in Fig. 8 (circled configuration) with curved meshes and nonconforming element boundaries. The DG scheme recovers optimal $\mathcal{O}(h^{P+1})$ convergence under h refinement (right panel) and exponential convergence under p refinement (bottom panels).

convergence is the subject of ongoing work.

C. A black hole in general relativity

Now we apply the DG scheme to solve the Einstein constraint equations of general relativity in the XCTS formulations, which is a set of coupled, nonlinear, elliptic PDEs on a curved manifold (see Appendix 2). Solutions to the XCTS equations describe admissible configurations of general-relativistic spacetime and provide initial data for general-relativistic time evolutions.

In this test problem we solve the XCTS equations (A.6) for a Schwarzschild black hole in Kerr-Schild coordinates,

$$\psi = 1, \tag{53a}$$

$$\alpha = \left(1 + \frac{2M}{r}\right)^{-1/2}, \tag{53b}$$

$$\beta^i = \frac{2M}{r}\alpha^2 l^i, \tag{53c}$$

with the background quantities

$$\bar{\gamma}_{ij} = \delta_{ij} + \frac{2M}{r} l_i l_j \tag{53d}$$

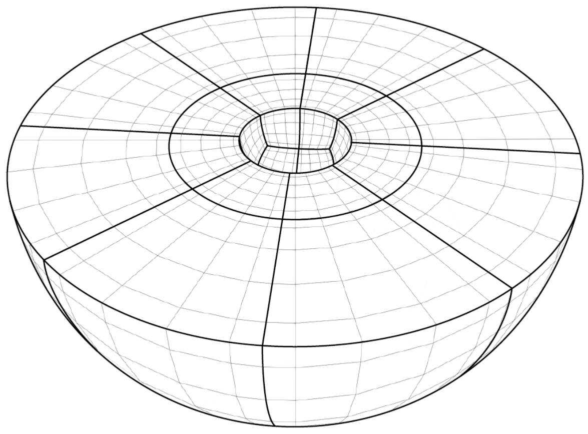


FIG. 10. A cut through the uniformly refined spherical-shell domain used in the black hole problem (Section IV C). The domain consists of six wedges with a logarithmic radial coordinate map enveloping an excised sphere. In this example each wedge is isotropically h refined once, i.e., split once in all three dimensions, resulting in a total of 48 elements. Note the elements are split in half along their logical axes, so the element size scales logarithmically in radial direction just like the distribution of grid points within the elements. Each element has six grid point per dimension, so fields are represented as polynomials of degree five.

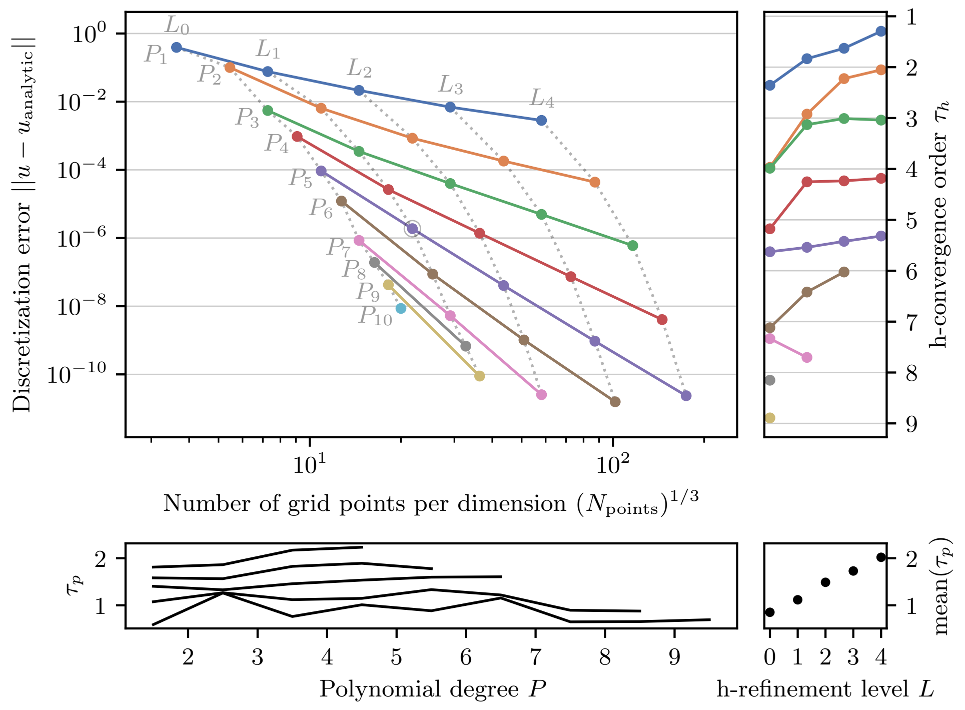


FIG. 11. Convergence of the three-dimensional black-hole solution with uniform *hp* refinement. Plotted is the L_2 error (46) over all variables of the XCTS equations $\{\psi, \alpha\psi, \beta^i\}$. The circled configuration is pictured in Fig. 10. The DG scheme recovers $\mathcal{O}(h^P)$ convergence under h refinement (right panel) and exponential convergence under p refinement (bottom panels).

and

$$K = \frac{2M\alpha^3}{r^2}\left(1 + \frac{3M}{r}\right), \tag{53e}$$

where M is the mass parameter, $r = \sqrt{x^2 + y^2 + z^2}$ is the Euclidean coordinate distance, and $l^i = l_i = x^i/r$.[16] The time-derivative quantities $\bar{u}_{ij}$ and $\partial_t K$ in the XCTS equations (A.6) vanish, as do the matter sources ρ, S, and S^i. Note that we have chosen a conformal decomposition with $\psi = 1$ here, but other choices of ψ and $\bar{\gamma}_{ij}$ that keep the spatial metric $\gamma_{ij} = \psi^4\bar{\gamma}_{ij}$ invariant are equally admissible.

We solve the XCTS equations numerically for the conformal factor ψ, the product $\alpha\psi$, and the shift β^i. The conformal metric $\bar{\gamma}_{ij}$ and the trace of the extrinsic curvature K are background quantities that are chosen in advance and remain fixed throughout the solve. They are a source of aliasing when evaluated on the computational grid (see Section III G). Importantly for this test problem the conformal metric $\bar{\gamma}_{ij}$ is not flat, resulting in a problem formulated on a curved manifold. For example, unit normal one forms in the DG scheme are normalized with respect to the conformal metric $\bar{\gamma}_{ij}$ and the metric determinant appears in the mass matrix and in the L_2 error (46).

[16] See Table 2.1 in Ref. [17].

To solve the black hole problem numerically we employ the strong compact DG scheme (30a) with $C = 1$ to discretize the XCTS equations (A.6) on a three-dimensional spherical shell, as illustrated in Fig. 10. The domain envelops an excised sphere that represents the black hole, so it has an outer and an inner external boundary that require boundary conditions. To obtain the Schwarzschild solution in Kerr-Schild coordinates we impose Eqs. (53a) to (53c) as Dirichlet-type boundary conditions at the outer boundary of the spherical shell at $r = 10M$. We place the inner radius of the spherical shell at $r = 2M$ and impose nonspinning apparent-horizon boundary conditions at the inner boundary,

$$\begin{aligned} n^k\partial_k\psi =& \frac{\psi^3}{8\alpha}n_i n_j\left((\bar{L}\beta)^{ij} - \bar{u}^{ij}\right) \\ &- \frac{\psi}{4}\bar{m}^{ij}\bar{\nabla}_i n_j - \frac{1}{6}K\psi^3, \end{aligned} \tag{54a}$$

$$\beta^i = -\frac{\alpha}{\psi^2}n^i, \tag{54b}$$

where $\bar{m}^{ij} = \bar{\gamma}^{ij} - n^i n^j$. These boundary conditions are not specific to the Schwarzschild solution but ensure the excision surface is an *apparent horizon* [16].[17] Since the

[17] See e.g., Section 12.3.2 in Ref. [17] for an introduction to the apparent-horizon boundary conditions.

Schwarzschild solution in Kerr-Schild coordinates has an apparent horizon at $r = 2M$ we recover the solution (53) when we place the inner radius of the spherical shell at that radius. The apparent-horizon boundary conditions (54) do not constrain the lapse α, so we impose Eq. (53b) at the inner boundary. The apparent-horizon boundary conditions are of Neumann-type for the variable ψ, of Dirichlet-type for $\alpha\psi$ and β^i, and nonlinear.

Since the XCTS equations (A.4) and the apparent-horizon boundary conditions (54) are nonlinear the initial guess for the iterative nonlinear solver becomes relevant. We choose an initial guess close to the analytic solution to ensure fast convergence of the iterative solver to the numerical solution. Note that the initial guess and other details of the iterative solve do not affect the discretization error of the numerical solution once the solve has converged to sufficient precision.

We present the convergence of the discretization error under uniform hp refinement in Fig. 11. The DG scheme for the nonlinear black hole problem recovers $\mathcal{O}(h^P)$ convergence under h refinement, which is an order lower than that obtained for the two preceding linear test problems. We find higher-order convergence for pure Dirichlet boundary conditions for this problem, suggesting the apparent-horizon boundary conditions (54) are responsible for the reduction of the convergence order. For a Poisson problem with nonlinear boundary conditions the authors of Ref. [24] also find a loss of convergence under h refinement. Under p refinement the scheme recovers exponential convergence and the mean exponential convergence scale τ_p increases linearly with the h-refinement level.

D. A black hole binary

Finally, we solve the Einstein constraint equations in the XCTS formulation as in Section IV C, but now we choose background quantities and boundary conditions that represent two black holes in orbit. This binary black hole problem is of significant relevance in numerical relativity to procure initial data for simulations of merging black holes [15, 17–19].

Following the formalism for *superposed Kerr-Schild* initial data, e.g., laid out in Ref. [18, 19], we set the conformal metric and the trace of the extrinsic curvature to the superpositions

$$\bar{\gamma}_{ij} = \delta_{ij} + \sum_{n=1}^{2} e^{-r_n^2/w_n^2}\,(\gamma_{ij}^{(n)} - \delta_{ij}) \tag{55a}$$

and

$$K = \sum_{n=1}^{2} e^{-r_n^2/w_n^2} K^{(n)}, \tag{55b}$$

where $\gamma_{ij}^{(n)}$ and $K^{(n)}$ are the conformal metric and extrinsic-curvature trace of two isolated Schwarzschild

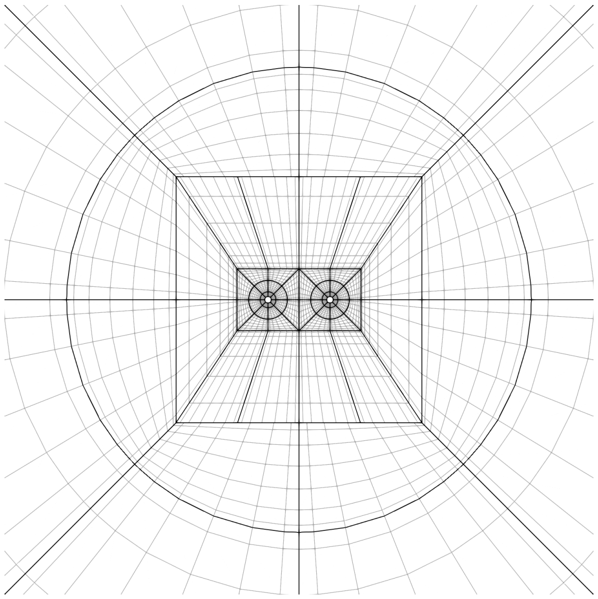


FIG. 12. A cut through the three-dimensional black-hole binary domain used in Section IV D. It involves two excised spheres centered at $\boldsymbol{C}_n$ along the x axis and extends to a spherical outer surface at radius R. The domain is h refined such that spherical wedges have equal angular size, so the cube-to-sphere boundary is nonconforming. All elements in this picture have eight angular grid points, and $\{7, 8, 8, 9, 11, 11\}$ radial grid points in the layers ordered from outermost to innermost.

black holes in Kerr-Schild coordinates as given in Eqs. (53). They have mass parameters M_n and are centered at coordinates $\boldsymbol{C}_n$, with r_n being the Euclidean coordinate distance from either center. The superpositions are modulated by two Gaussians with widths w_n. The time-derivative quantities $\bar{u}_{ij}$ and $\partial_t K$ in the XCTS equations (A.4) vanish, as do the matter sources ρ, S and S^i.

To handle orbital motion we split the shift in a *background* and an *excess* contribution [42],

$$\beta^i = \beta^i_{\text{background}} + \beta^i_{\text{excess}}, \tag{56}$$

and choose the background shift

$$\beta^i_{\text{background}} = (\boldsymbol{\Omega}_0 \times \boldsymbol{x})^i, \tag{57}$$

where $\boldsymbol{\Omega}_0$ is the orbital angular velocity. We insert Eq. (56) in the XCTS equations (A.4) and henceforth solve them for β^i_{excess}, instead of β^i.

We solve the XCTS equations on the domain depicted in Fig. 12. It has two excised spheres with radius $2M_n$ that are centered at $\boldsymbol{C}_n$, and correspond to the two black holes, and an outer spherical boundary at finite radius R. We impose boundary conditions on these three boundaries as follows. At the outer spherical boundary we impose asymptotic flatness,

$$\psi = 1, \quad \alpha\psi = 1, \quad \beta^i_{\text{excess}} = 0. \tag{58}$$

Since the outer boundary is at a finite radius, the solution will only be approximately asymptotically flat. On the

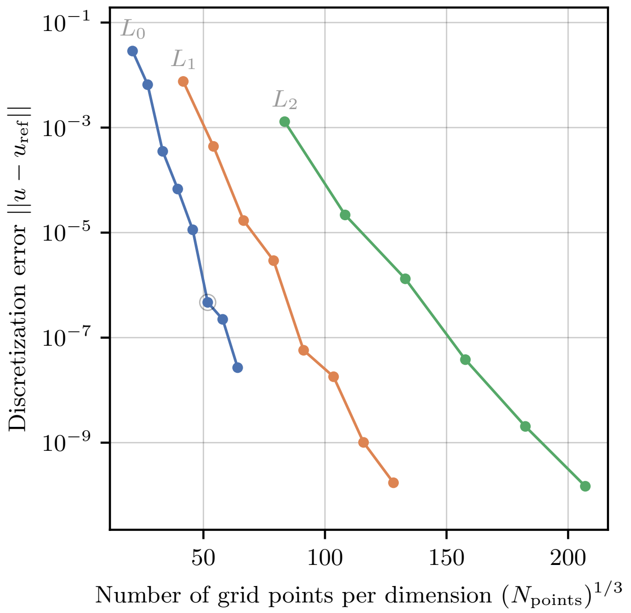


FIG. 13. Exponential convergence of the three-dimensional black-hole binary problem under uniform p refinement (solid lines) for three uniform h refinement levels. The circled configuration is pictured in Fig. 12. Plotted here is the L_2 error (59) over all variables $\{\psi, \alpha\psi, \beta^i_\mathrm{excess}\}$ and all three interpolation points $\boldsymbol{x}_m$.

two excision boundaries we impose nonspinning apparent-horizon boundary conditions, Eq. (54). For the lapse we choose to impose the isolated solution (53b) as Dirichlet conditions at both excision surfaces. Note that this choice differs slightly from Ref. [19], where the *superposed* isolated solutions are imposed on the lapse at both excision surfaces.

Since the binary black hole problem has no analytic solution we assess the precision of numerical solutions by comparing them to a high-resolution reference configuration. Specifically, we interpolate all five fields $u_A = \{\psi, \alpha\psi, \beta^i_\mathrm{excess}\}$ to a set of sample points $\boldsymbol{x}_m$. Then, we compute the discretization error as an L_2 norm of the difference to the high-resolution reference run over all fields and sample points,

$$\|u - u_\mathrm{ref}\| := \left(\sum_{A,m} \left(u_A(\boldsymbol{x}_m) - u_{A,\mathrm{ref}}(\boldsymbol{x}_m)\right)^2\right)^{1/2}. \quad (59)$$

Figure 13 presents the convergence of the discretization error under uniform hp refinement for our strong compact DG scheme (30a) with $C = 1$. Specifically, we obtain h-refinement levels from the domain depicted in Fig. 12 by splitting all elements in two along their three logical axes. We obtain p-refinement levels by incrementing the number of grid points by one in all elements and dimensions. The DG scheme recovers exponential convergence under p refinement, and suggests the same $\mathcal{O}(h^P)$ convergence

	$\boldsymbol{x}_1 = (8.846, 0, 0)$	$\boldsymbol{x}_2 = (0, 0, 0)$	$\boldsymbol{x}_3 = (100, 0, 0)$
ψ	1.0919141	1.0602545	1.0033643
$\alpha\psi$	0.7072066	0.9381658	0.9966282
β^x_excess	0.3870172	0	0.0008802
β^y_excess	−0.1273493	0	−0.0003467
β^z_excess	0	0	0

TABLE II. Sample points $\boldsymbol{x}_m$ used in (59) and the value of the reference solution at the sample points.

under h refinement that we have found for the single black hole problem in Section IV C. We have chosen $M_n = 0.4229$, $\boldsymbol{C}_n = (\pm 8, 0, 0)$, $\Omega_0 = 0.0144$, $w_n = 4.8$, $R = 300$, and sample points along the x axis at $x_1 = 8.846$ (near horizon), $x_2 = 0$ (origin) and $x_3 = 100$ (far field) here. For the high-resolution reference configuration in Eq. (59) we use a run that is h refined twice, and has one grid point more per element and dimension than the highest-resolution configuration included in Fig. 13. The reference values at the interpolation points are listed in Table II. We have verified that these values are consistent with the same problem solved with the `SpEC` [43, 44] code up to an absolute error of at most 10^{-7}, which is the precision we report in Table II.

In forthcoming work we intend to employ the DG scheme that we have presented here to develop a scalable initial-data solver for binaries involving black holes and neutron stars in the `SpECTRE` numerical relativity code [23].

V. CONCLUSION AND FUTURE WORK

We have presented a unified discontinuous Galerkin (DG) internal-penalty scheme that is applicable to a wide range of elliptic equations. Our scheme applies to linear and nonlinear second-order elliptic PDEs of one or more variables, where the variables can be scalars, vectors, or tensors of higher rank. It does not require problem-specific modifications of the DG discretization or of the numerical fluxes that couple neighboring elements. The scheme supports a wide range of linear and nonlinear boundary conditions, and applies to equations formulated on curved manifolds. We demonstrate its versatility by solving a simple Poisson problem, a linear elasticity problem on a curved mesh with nonconforming element boundaries, and two nonlinear problems in general relativity involving black holes. The unified DG scheme is capable of solving these problems with no structural changes. It recovers optimal $\mathcal{O}(h^{P+1})$ convergence for the linear test problems and $\mathcal{O}(h^P)$ convergence for the nonlinear test problems, where P is the polynomial degree of the elements. The scheme is implemented in the open-source `SpECTRE` code [22] and the results presented in this article are reproducible with the supplemental input-file configurations.[18]

The DG scheme developed here can potentially be improved in multiple ways in future work. Dealiasing techniques have the potential to increase the accuracy of the scheme on curved meshes and for equations with background quantities. The choice of penalty on curved meshes remains a subject of ongoing study. Furthermore, detailed studies of the symmetry of the DG operator and related adjustments to the scheme, such as switching to the strong-weak formulation, can potentially make the DG operator faster to solve.

Since the convergence properties of the DG scheme are sensitive to the specifics of the computational domain, we have chosen to refine the domains as uniformly as possible while retaining some important features, such as curved meshes and nonconforming element boundaries. For practical applications it is typically more important to obtain steep rather than uniform convergence, in order to conserve computational resources and thus achieve faster or more precise solves. Therefore, a focus of future work will be to develop adaptive mesh-refinement strategies for the elliptic DG scheme that place grid points in regions and dimensions of the domain that dominate the discretization error.

Once the DG discretization of the elliptic equations is at hand, numerical techniques for solving the resulting matrix equation become important. Sophisticated linear and nonlinear iterative algorithms are necessary to solve high-resolution elliptic problems in parallel on large computing clusters. Many of the choices we have made in the development of the DG scheme are motivated by such large-scale applications. For this purpose we are developing a scalable multigrid-Schwarz preconditioned Newton-Krylov iterative solver with task-based parallelism that will be presented in [23].

ACKNOWLEDGMENTS

The authors thank Trevor Vincent, Hannes Rüter, Nils Deppe, Will Throwe, Lawrence Kidder, and Saul Teukolsky for helpful discussions. N. F. also thanks the Cornell Center for Astrophysics and Planetary Science and TAPIR at Caltech for the hospitality and financial support during research stays. Computations were performed with the `SpECTRE` code [22] on the Minerva cluster at the Max Planck Institute for Gravitational Physics. The figures in this article were produced with `dgpy` [45], `matplotlib` [46, 47], `TikZ` [48], and `ParaView` [49].

[18] https://arxiv.org/src/2108.05826/anc

Appendix: Physical systems

1. Puncture equation

A popular approach to produce initial data for general-relativistic time evolutions involving black holes is to reduce the Einstein constraint equations to a single non-linear elliptic PDE, the *puncture equation* [19]

$$-\partial_i\partial_i u = \beta\left(\alpha\,(1+u)+1\right)^{-7}, \tag{A.1}$$

written here in Cartesian coordinates. The puncture equation (A.1) is solved for the field $u(\boldsymbol{x})$ from which an admissible spacetime metric can be constructed [17]. The quantities

$$\frac{1}{\alpha} = \sum_I \frac{M_I}{r_I}, \tag{A.2a}$$

$$\beta = \frac{1}{8}\alpha^7 \bar{A}_{ij}\bar{A}^{ij}, \tag{A.2b}$$

$$\bar{A}^{ij} = \frac{3}{2}\sum_I \frac{1}{r_I^2}\Big(2P_I^{(i}n_I^{j)} - (\delta^{ij} - n_I^i n_I^j)P_I^k n_I^k + \frac{4}{r_I} n_I^{(i}\epsilon^{j)kl} S_k^I n_l^I\Big) \tag{A.2c}$$

are background fields that define a configuration of multiple black holes. The black holes are parametrized by their puncture masses M_I, positions $\boldsymbol{C}_I$, linear momenta $\boldsymbol{P}_I$ and spins $\boldsymbol{S}_I$. In Eq. (A.2), $r_I = \|\boldsymbol{x} - \boldsymbol{C}_I\|$ is the Euclidean coordinate distance to the Ith black hole and $\boldsymbol{n}_I = (\boldsymbol{x} - \boldsymbol{C}_I)/r_I$ is the radial unit normal to the Ith black hole.

To formulate the puncture equation (A.1) in first-order flux form (1) we can choose the auxiliary variable $v_i = \partial_i u$ and the fluxes and sources

$$\mathcal{F}_{v\,j}^{\;i} = u\,\delta^i_j, \quad \mathcal{S}_{v\,j} = v_j, \tag{A.3a}$$

$$\mathcal{F}_u^{\;i} = v_i, \qquad \mathcal{S}_u = -\beta\left(\alpha\,(1+u)+1\right)^{-7}, \tag{A.3b}$$

along with both $f_\alpha = 0$. Other Poisson-type equations with nonlinear sources can be formulated analogously.

[19] See, e.g., Section 12.2 in Ref. [17] for an introduction to puncture initial data.

2. The XCTS equations of general relativity

The XCTS equations

$$\bar{\nabla}^2\psi = \frac{1}{8}\psi\bar{R} + \frac{1}{12}\psi^5 K^2 - \frac{1}{8}\psi^{-7}\bar{A}_{ij}\bar{A}^{ij} - 2\pi\psi^5\rho \tag{A.4a}$$

$$\bar{\nabla}^2(\alpha\psi) = \alpha\psi\left(\frac{7}{8}\psi^{-8}\bar{A}_{ij}\bar{A}^{ij} + \frac{5}{12}\psi^4K^2 + \frac{1}{8}\bar{R} + 2\pi\psi^4(\rho+2S)\right) - \psi^5\partial_t K + \psi^5\beta^i\bar{\nabla}_i K \tag{A.4b}$$

$$\bar{\nabla}_i(\bar{L}\beta)^{ij} = (\bar{L}\beta)^{ij}\bar{\nabla}_i\ln(\bar{\alpha}) + \bar{\alpha}\bar{\nabla}_i\left(\bar{\alpha}^{-1}\bar{u}^{ij}\right) + \frac{4}{3}\bar{\alpha}\psi^6\bar{\nabla}^j K + 16\pi\bar{\alpha}\psi^{10}S^j \tag{A.4c}$$

with $\bar{A}^{ij} = \frac{1}{2\bar{\alpha}}\left((\bar{L}\beta)^{ij} - \bar{u}^{ij}\right)$ and $\bar{\alpha} = \alpha\psi^{-6}$ are a set of nonlinear elliptic equations that the spacetime metric of general relativity must satisfy at all times [15].[20] They are solved for the conformal factor ψ, the product of lapse and conformal factor $\alpha\psi$, and the shift vector β^j. The remaining quantities in the equations, i.e., the conformal metric $\bar{\gamma}_{ij}$, the trace of the extrinsic curvature K, their respective time derivatives $\bar{u}_{ij}$ and $\partial_t K$, the energy density ρ, the stress-energy trace S and the momentum density S^i, are freely specifiable fields that define the scenario at hand. Of particular importance is the conformal metric $\bar{\gamma}_{ij}$, which defines the background geometry, the covariant derivative $\bar{\nabla}$, the Ricci scalar $\bar{R}$ and the longitudinal operator

$$\left(\bar{L}\beta\right)^{ij} = \bar{\nabla}^i\beta^j + \bar{\nabla}^j\beta^i - \frac{2}{3}\bar{\gamma}^{ij}\bar{\nabla}_k\beta^k. \tag{A.5}$$

Note that the XCTS equations are essentially two Poisson equations and one elasticity equation with coupled, nonlinear sources on a curved manifold. In this analogy, the longitudinal operator plays the role of the elastic constitutive relation that connects the symmetric "shift strain" $\bar{\nabla}_{(i}\beta_{j)}$ with the "stress" $(\bar{L}\beta)^{ij}$ of which we take the divergence in the momentum constraint (A.4c). This particular constitutive relation is equivalent to an isotropic and homogeneous material (50) with bulk modulus $K = 0$ (not to be confused with the extrinsic curvature trace K in this context) and shear modulus $\mu = 1$.

To formulate the XCTS equations in first-order flux form (1) we choose for auxiliary variables the gradient of the conformal factor, $v_i = \partial_i\psi$, the gradient of the lapse times the conformal factor, $w_i = \partial_i(\alpha\psi)$, and the symmetric shift strain $B_{ij} = \bar{\nabla}_{(i}\beta_{j)}$. Then, the XCTS equations (A.4) can be formulated with the fluxes and sources

$$\mathcal{F}_v{}^i{}_j = \delta^i_j\psi, \qquad \mathcal{S}_{v\,j} = v_j, \tag{A.6a}$$

$$\mathcal{F}_\psi{}^i = \bar{\gamma}^{ij}v_j, \qquad \mathcal{S}_\psi = -\bar{\Gamma}^i_{ij}\mathcal{F}_\psi{}^j + \frac{1}{8}\psi\bar{R} + \frac{1}{12}\psi^5K^2 - \frac{1}{8}\psi^{-7}\bar{A}_{ij}\bar{A}^{ij} - 2\pi\psi^5\rho \tag{A.6b}$$

for Eq. (A.4a),

$$\mathcal{F}_w{}^i{}_j = \delta^i_j\alpha\psi, \qquad \mathcal{S}_{w\,j} = w_j, \tag{A.6c}$$

$$\mathcal{F}_{(\alpha\psi)}{}^i = \bar{\gamma}^{ij}w_j, \quad \mathcal{S}_{(\alpha\psi)} = -\bar{\Gamma}^i_{ij}\mathcal{F}_{(\alpha\psi)}{}^j + \alpha\psi\left(\frac{7}{8}\psi^{-8}\bar{A}_{ij}\bar{A}^{ij} + \frac{5}{12}\psi^4K^2 + \frac{1}{8}\bar{R} + 2\pi\psi^4(\rho+2S)\right) - \psi^5\partial_t K + \psi^5\beta^i\bar{\nabla}_i K \tag{A.6d}$$

for Eq. (A.4b), and

$$\mathcal{F}_B{}^i{}_{jk} = \delta^i_{(j}\bar{\gamma}_{k)l}\beta^l, \qquad \mathcal{S}_{B\,jk} = B_{jk} + \bar{\Gamma}_{ijk}\beta^i, \tag{A.6e}$$

$$\mathcal{F}_\beta{}^{ij} = 2\left(\bar{\gamma}^{ik}\bar{\gamma}^{jl} - \frac{1}{3}\bar{\gamma}^{ij}\bar{\gamma}^{kl}\right)B_{kl}, \quad \mathcal{S}_\beta{}^i = -\bar{\Gamma}^j_{jk}\mathcal{F}_\beta{}^{ik} - \bar{\Gamma}^i_{jk}\mathcal{F}_\beta{}^{jk} + \left(\mathcal{F}_\beta{}^{ij} - \bar{u}^{ij}\right)\bar{\gamma}_{jk}\left(\frac{\mathcal{F}_{(\alpha\psi)}{}^k}{\alpha\psi} - 7\frac{\mathcal{F}_\psi{}^k}{\psi}\right) + \bar{\nabla}_j\bar{u}^{ij} + \frac{4}{3}\alpha\bar{\nabla}^i K + 16\pi\alpha\psi^4S^i \tag{A.6f}$$

for Eq. (A.4c). All fixed sources $f_\alpha(\boldsymbol{x})$ vanish. Note that in Eq. (A.6f), $\mathcal{F}_\beta{}^{ij} = \left(\bar{L}\beta\right)^{ij}$, expressed in the auxiliary variables.

[20] See, e.g., Ref. [17] for an introduction to the XCTS equations, in particular Box 3.3.

[1] W. H. Reed and T. R. Hill, *Triangular mesh methods for the neutron transport equation*, Tech. Rep. LA-UR-73-479 (Los Alamos Scientific Lab., N. Mex., USA, 1973).

[2] J. S. Hesthaven and T. Warburton, *Nodal Discontinuous Galerkin Methods* (Springer, New York, 2008).

[3] B. Cockburn, G. E. Karniadakis, and C.-W. Shu, The Development of Discontinuous Galerkin Methods, in *Discontinuous Galerkin Methods* (Springer, Berlin, Heidelberg, 2000) pp. 3–50.

[4] B. Cockburn, Devising discontinuous Galerkin methods for non-linear hyperbolic conservation laws, J. Comput. Appl. Math. **128**, 187 (2001).

[5] M. F. Wheeler, An elliptic collocation-finite element method with interior penalties, SIAM J. Numer. Anal. **15**, 152 (1978).

[6] D. Arnold, F. Brezzi, B. Cockburn, and D. L. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, SIAM J. Numer. Anal. **39**, 1749 (2002).

[7] D. Schötzau, C. Schwab, T. Wihler, and M. Wirz, Exponential convergence of hp-DGFEM for elliptic problems in polyhedral domains, in *Spectral and High Order Methods for Partial Differential Equations - ICOSAHOM 2012* (Springer, Cham, 2014) pp. 57–73.

[8] T. Vincent, H. P. Pfeiffer, and N. L. Fischer, hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity, Phys. Rev. D **100**, 084052 (2019), arXiv:1907.01572 [physics.comp-ph].

[9] P. Hansbo and M. G. Larson, Discontinuous Galerkin methods for incompressible and nearly incompressible elasticity by Nitsche's method, Comput. Methods Appl. Mech. Eng. **191**, 1895 (2002).

[10] B. Cockburn, D. Schötzau, and J. Wang, Discontinuous Galerkin methods for incompressible elastic materials, Comput. Methods. Appl. Mech. Eng. **195**, 3184 (2006).

[11] C. Ortner and E. Süli, Discontinuous Galerkin finite element approximation of nonlinear second-order elliptic and hyperbolic systems, SIAM J. Numer. Anal. **45**, 1370 (2007).

[12] S. A. Teukolsky, Formulation of discontinuous Galerkin methods for relativistic astrophysics, J. Comput. Phys. **312**, 333 (2016), arXiv:1510.01190 [gr-qc].

[13] L. E. Kidder *et al.*, `SpECTRE`: A task-based discontinuous Galerkin code for relativistic astrophysics, J. Comput. Phys. **335**, 84 (2017), arXiv:1609.00098 [astro-ph.HE].

[14] F. Fambri, M. Dumbser, S. Köppel, L. Rezzolla, and O. Zanotti, ADER discontinuous Galerkin schemes for general-relativistic ideal magnetohydrodynamics, Mon. Not. R. Astron. Soc. **477**, 4543 (2018), arXiv:1801.02839 [physics.comp-ph].

[15] H. P. Pfeiffer, The initial value problem in numerical relativity, J. Hyperbol. Diff. Equat. **2**, 497 (2005), arXiv:gr-qc/0412002.

[16] G. B. Cook and H. P. Pfeiffer, Excision boundary conditions for black hole initial data, Phys. Rev. D **70**, 104016 (2004), arXiv:gr-qc/0407078.

[17] T. W. Baumgarte and S. L. Shapiro, *Numerical Relativity: Solving Einsteins Equations on the Computer* (Cambridge University Press, Cambridge, England, 2010).

[18] G. Lovelace, R. Owen, H. P. Pfeiffer, and T. Chu, Binary-black-hole initial data with nearly-extremal spins, Phys. Rev. D **78**, 084017 (2008), arXiv:0805.4192 [gr-qc].

[19] V. Varma, M. A. Scheel, and H. P. Pfeiffer, Comparison of binary black hole initial data sets, Phys. Rev. D **98**, 104011 (2018), arXiv:1808.08228 [gr-qc].

[20] N. Tacik *et al.*, Binary neutron stars with arbitrary spins in numerical relativity, Phys. Rev. D **92**, 124012 (2015), arXiv:1508.06986 [gr-qc].

[21] N. Tacik, F. Foucart, H. P. Pfeiffer, C. Muhlberger, L. E. Kidder, M. A. Scheel, and B. Szilágyi, Initial data for black hole–neutron star binaries, with rotating stars, Class. Quant. Grav. **33**, 225012 (2016), arXiv:1607.07962 [gr-qc].

[22] N. Deppe, W. Throwe, L. E. Kidder, N. L. Fischer, F. Hébert, J. Moxon, C. Armaza, G. S. Bonilla, P. Kumar, G. Lovelace, E. O'Shea, H. P. Pfeiffer, M. A. Scheel, S. A. Teukolsky, *et al.*, `SpECTRE v2021.12.15`, 10.5281/zenodo.5784324 (2021).

[23] N. L. Fischer *et al.*, A scalable elliptic solver with task-based parallelism for the `SpECTRE` numerical relativity code (2021), arXiv:2111.06767 [gr-qc].

[24] M. Feistauer, F. Roskovec, and A.-M. Sändig, Discontinuous Galerkin method for an elliptic problem with nonlinear Newton boundary conditions in a polygon, IMA J. Numer. Anal. **39**, 423 (2019).

[25] R. Hartmann and P. Houston, An optimal order interior penalty discontinuous Galerkin discretization of the compressible Navier–Stokes equations, J. Comput. Phys. **227**, 9670 (2008).

[26] Y. Epshteyn and B. Rivière, Estimation of penalty parameters for symmetric interior penalty Galerkin methods, J. Comput. Appl. Math. **206**, 843 (2007).

[27] J. D. De Basabe, M. K. Sen, and M. F. Wheeler, The interior penalty discontinuous Galerkin method for elastic wave propagation: Grid dispersion, Geophys. J. Int. **175**, 83 (2008).

[28] D. Fortunato, C. H. Rycroft, and R. Saye, Efficient operator-coarsening multigrid schemes for local discontinuous Galerkin methods, SIAM J. Sci. Comput. **41**, A3913 (2019).

[29] D. A. Kopriva, *Implementing Spectral Methods for Partial Differential Equations* (Springer, New York, 2009).

[30] J. Peraire and P.-O. Persson, The compact discontinuous Galerkin (CDG) method for elliptic problems, SIAM J. Sci. Comput. **30**, 1806 (2008), arXiv:math/0702353 [math.NA].

[31] J. Douglas and T. Dupont, Interior penalty procedures for elliptic and parabolic Galerkin methods, in *Computing Methods in Applied Sciences* (Springer, Berlin, Heidelberg, 1976) pp. 207–216.

[32] K. Shahbazi, An explicit expression for the penalty parameter of the interior penalty method, J. Comput. Phys. **205**, 401 (2005).

[33] K. Hillewaert, *Development of the Discontinuous Galerkin Method for High-resolution, Large Scale CFD and Acoustics in Industrial Geometries*, Ph.D. thesis, Université catholique de Louvain (2013).

[34] S. A. Teukolsky, Short note on the mass matrix for Gauss-Lobatto grid points, J. Comput. Phys. **283**, 408 (2015), arXiv:1412.2276 [math.NA].

[35] G. Mengaldo, D. De Grazia, D. Moxey, P. E. Vincent, and S. J. Sherwin, Dealiasing techniques for high-order spectral element methods on regular and irregular grids, J. Comput. Phys. **299**, 56 (2015).

[36] Y. Saad and M. H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput. **7**, 856 (1986).

[37] G. Lovelace, N. Demos, and H. Khan, Numerically modeling Brownian thermal noise in amorphous and crystalline thin coatings, Class. Quant. Grav. **35**, 025017 (2018), arXiv:1707.07774 [gr-qc].

[38] Y. Levin, Internal thermal noise in the LIGO test masses: A direct approach, Phys. Rev. D **57**, 659 (1998), arXiv:gr-qc/9707013.

[39] K. S. Thorne and R. D. Blandford, *Modern Classical Physics: Optics, Fluids, Plasmas, Elasticity, Relativity, and Statistical Physics* (Princeton University Press, Princeton, NJ, 2017).

[40] Y. T. Liu and K. S. Thorne, Thermoelastic noise and homogeneous thermal noise in finite sized gravitational wave test masses, Phys. Rev. D **62**, 122002 (2000), arXiv:gr-qc/0002055.

[41] G. Lovelace, The dependence of test-mass coating and substrate thermal noise on beam shape in the advanced laser interferometer gravitational-wave observatory (advanced LIGO), Class. Quant. Grav. **24**, 4491 (2007), arXiv:gr-qc/0610041.

[42] H. P. Pfeiffer, *Initial Data for Black Hole Evolutions*, Ph.D. thesis, Cornell University (2003), gr-qc/0510016.

[43] H. P. Pfeiffer, L. E. Kidder, M. A. Scheel, and S. A. Teukolsky, A multidomain spectral method for solving elliptic equations, Comput. Phys. Commun. **152**, 253 (2003).

[44] L. E. Kidder, H. P. Pfeiffer, M. A. Scheel, *et al.*, Spectral Einstein Code (`SpEC`), `black-holes.org/code/SpEC`.

[45] N. L. Fischer, `dgpy v0.1`, 10.5281/zenodo.5086181 (2021).

[46] J. D. Hunter, Matplotlib: A 2d graphics environment, Comput. Sci. Eng, **9**, 90 (2007).

[47] T. A. Caswell *et al.*, `matplotlib v3.3.3`, 10.5281/zenodo.4268928 (2020).

[48] T. Tantau, `pgf` - a portable graphic format for TeX, `github:pgf-tikz/pgf` (2021).

[49] J. Ahrens, B. Geveci, and C. Law, `ParaView`: An end-user tool for large-data visualization, in *Visualization Handbook* (Butterworth-Heinemann, Burlington, 2005).