Space-Time DPG: Designing a Method for Massively Parallel CFD

Truman E. Ellis[a,], Leszek F. Demkowicz[a], Jesse L. Chan[b], Robert D. Moser[a]

[a] *Institute for Computational Engineering and Sciences, University of Texas at Austin 201 East 24th St, Stop C0200, Austin, TX 78703*

[b] *Computational and Applied Mathematics, Rice University 6100 Main MS-134, Houston, TX 77005*

Abstract

We develop a space-time discontinuous Petrov-Galerkin method (DPG) for transient parabolic partial differential equations with an ultimate goal of deployment on high performance computing systems. The space-time framework allows automatic adaptivity in both space and time. The heat equation serves as a model problem, but the ideas developed here are then immediately applicable to compressible Navier-Stokes. Preliminary spatially 1D results for the Sod shock tube and Noh implosion problems are promising.

Keywords:
Discontinuous Petrov-Galerkin, Finite Elements, Space-Time, Navier-Stokes

1. Introduction

Initial mesh design for computational fluid dynamics can be a time-consuming and expensive process. The stability properties and nonlinear convergence of most numerical methods rely on a minimum level of mesh resolution. This means that unless the initial computational mesh is fine enough, convergence can not be guaranteed. Any meshes below this minimum resolution level are termed to be in the "pre-asymptotic regime." This condition implies that meshes need to in some way anticipate the solution before it is known. On top of the minimum requirement that the surface meshes must adequately represent the geometry of the problem under consideration,

Email address: truman@ices.utexas.edu (Truman E. Ellis)

resolution requirements on the volume mesh make the CFD practitioner's job significantly more time consuming. This is not to mention auxiliary requirements from turbulence models needed to accurately resolve boundary layers. Thus, mesh design and simulation become an iterative trial and error process. A common process is for an engineer to study the problem at hand and attempt to predict regions that require extra resolution. Then they will spend hours coaxing the mesher to produce an adequate mesh before importing the mesh into the solver. Too often, the solver will fail to converge on the given mesh due to some unforeseen mesh inadequacy on part of the solution domain. The engineer then needs to descend back into the mesher to fix the problem elements. This process is repeated until the solver is satisfied and convergence can be reached. This iterative trial and error is undesirable while working on personal computers or modestly sized compute clusters, but becomes increasing costly as the problem is scaled up to the tens or hundreds of thousands of processors common in high performance computing environments.

In contrast to most other numerical methods, the discontinuous Petrov-Galerkin finite element method retains exceptional stability on extremely coarse meshes. DPG is also inherently very adaptive. It is possible to compute the residual error without knowledge of the exact solution, which can be used to robustly drive adaptivity. This results in a very automated technology, as the user can initialize a computation on the coarsest mesh which adequately represents the geometry then step back and let the program solve and adapt iteratively until it resolves the solution features.

2. Overview of DPG

For a full treatment of the various ideas in DPG, please see [1]. The basic ideas are fairly straight-forward; DPG minimizes the residual in a user defined energy norm. Consider a variational problem: find $u \in U$ such that

$$b(u, v) = l(v) \quad \forall v \in V$$

with operator $B : U \to V' \quad (V'$ is the dual space to $V)$ defined by $b(u, v) = \langle Bu, v \rangle_{V' \times V}$. This gives the operator equation:

$$Bu = l \in V' .$$

We wish to minimize the residual $Bu - l$ in V':

$$u_h = \underset{w_h \in U_h}{\arg\min} \frac{1}{2} \|Bu - l\|_{V'}^2 \ .$$

This is a very natural mathematical framework based soundly in functional analysis, but it is not yet a practical method as the V' norm is not especially tractable to work with. The insight is that since we are working with Hilbert spaces, we can use the Riesz representation theorem to find a complementary object in V rather than V'. Let $R_V : V \ni v \to (v, \cdot) \in V'$ be the Riesz map. Then the inverse Riesz map (which is an isometry) lets us represent our residual in V:

$$u_h = \underset{w_h \in U_h}{\arg\min} \frac{1}{2} \left\|R_V^{-1}(Bu - l)\right\|_V^2 \ .$$

Taking the Gâteaux derivative to be zero in all directions $\delta u \in U_h$ gives,

$$\left(R_V^{-1}(Bu_h - l), R_V^{-1} B\delta u\right)_V = 0, \quad \forall \delta u \in U,$$

which by definition of the Riesz map is equivalent to

$$\left\langle Bu_h - l, R_V^{-1} B\delta u_h \right\rangle = 0 \quad \forall \delta u_h \in U_h \,,$$

with optimal test functions $v_{\delta u_h} \coloneqq R_V^{-1} B\delta u_h$ for each trial function δu_h. This gives a simple bilinear form

$$b(u_h, v_{\delta u_h}) = l(v_{\delta u_h}),$$

with $v_{\delta u_h} \in V$ that solves the auxiliary problem

$$(v_{\delta u_h}, \delta v)_V = \langle R_V v_{\delta u_h}, \delta v \rangle = \langle B\delta u_h, \delta v \rangle = b(\delta u_h, \delta v) \quad \forall \delta v \in V.$$

We might call this an *optimal Petrov-Galerkin* method. We arrive at the same method by realizing the supremum in the inf-sup condition, motivating the *optimal* nomenclature. These optimal Petrov-Galerkin methods produce Hermitian, positive-definite stiffness matrices since

$$b(u_h, v_{\delta \boldsymbol{u}_h}) = (v_{u_h}, v_{\delta \boldsymbol{u}_h})_V = \overline{(v_{\delta \boldsymbol{u}_h}, v_{u_h})} = \overline{b(\delta u_h, v_{u_h})} \,.$$

We can calculate the energy norm (defined by $\|u\|_E := \|Bu\|_{V'}$) of the Galerkin error without knowing the exact solution by using the residual:

$$\|u_h - u\|_E = \|B(u_h - u)\|_{V'} = \|Bu_h - l\|_{V'} = \left\|R_V^{-1}(Bu_h - l)\right\|_V \ ,$$

where we designate $R_V^{-1}(Bu_h - l)$ the *error representation function*. This has proven to be a very reliable *a-posteriori* error estimator for driving adaptivity.

Babuška's theorem[2] says that discrete stability and approximability imply convergence. That is, if M is the continuity constant for $b(u, v)$ which satisfies the discrete inf-sup condition with constant γ_h,

$$\sup_{v_h \in V_h} \frac{|b(u, v)|}{\|v_h\|_V} \geq \gamma_h \|u_h\|_U \ ,$$

then the Galerkin error satisfies the bound

$$\|u_h - u\|_U \leq \frac{M}{\gamma_h} \inf_{w_h \in U_h} \|w_h - u\|_U \ .$$

Optimal test functions realize the supremum in the discrete discrete inf-sup condition such that $\gamma_h \geq \gamma$, the infinite-dimensional inf-sup constant. If we then use the energy norm for $\|\cdot\|_U$, then $M = \gamma = 1$ and Babuška's estimate implies that the optimal Petrov-Galerkin method is the most stable Petrov-Galerkin method possible.

There are still many features of the method that are left to be decided, for example the U and V spaces. If V is taken to be a continuous space, then the auxiliary problem becomes global in scope, something that we would like to avoid. In order to ensure the auxiliary problem can be solved element-by-element, we take V to be a space of functions that are discontinuous between elements. (Technically, V should also be infinite dimensional, but we have found it to be sufficient to use an "enriched" space of higher polynomial dimension than the trial space[3].) The downside to using discontinuous test functions is that it introduces new interface unknowns. When the equations are integrated by parts over each element, the jump in test functions introduces new unknowns on the mesh skeleton that would have disappeared with continuous test functions. Moro *et al.* [4] handle the flux unknowns with a numerical flux in the hybridized DPG (HDPG) method, but the standard DPG method treats these as new unknowns to be solved for. We still haven't specified our trial space U, but the rule is that for every integration by parts, a new skeleton unknown is introduced. Most DPG considerations have used the ultra-weak variational formulation and break a second order PDE into a system of first order PDEs. This introduces a trace unknown (through the constitutive law) and a flux unknown (through the conservation law) with field variables that live in L^2. But Demkowicz and Gopalakrishnan also formulated a *primal DPG* method[5] for second order equations that does not

introduce a trace unknown. The overall number of interface unknowns in the primal DPG method is the same, however, since the solution is required to be H^1 conforming and the trace unknowns are essentially hidden here.

The final unresolved choice is what norm to apply to the V space. This is one of the most important factors in designing a robust DPG method as this norm needs to be inverted to solve for the optimal test functions. If the norm produces unresolved boundary layers in the auxiliary problem, then many of the attractive features of DPG may fall apart. For a full discussion of robust test norms for convection-diffusion problems, see [6, 7]. But elimination of boundary layers in the auxiliary solve is not the only requirement at play. This choice also controls what norm the residual is minimized in. Often we want this norm to be equivalent to the L^2 norm. Fortunately, we have found that it is possible to design test norms such that the implied energy norm is provably robust and equivalent to L^2 for convection-diffusion which serves as the most relevant model problem for our research. Norms for Navier-Stokes are derived by analogy to the convection-diffusion norm.

2.1. DPG for Nonlinear Problems

In the context of linear problems, DPG has been successfully applied to a wide range of physical applications. Theoretical groundwork for DPG applied to the Poisson equation was laid in [8]. The time-harmonic Helmholtz equation was the focus of [9, 10] and [11]. Linear elasticity and plate problems were addressed in [12], [13], and [14], and Maxwell was addressed in [15, 16]. Linear fluid dynamics problems include convection-diffusion[17, 6, 7, 18, 19] and stationary Stokes flow[20, 19].

In the following section, we extend the idea of DPG to nonlinear problems. In an abstract sense, we consider the discrete nonlinear variational problem

$$\langle B(u_h) - \ell, v\rangle_{V^*\times V} \coloneqq \langle R(u_h), v\rangle_{V^*\times V} = 0, \quad \forall v \in V.$$

The linearization of this problem leads to the Newton iteration, defined by the step

$$\langle B'(u_h)\Delta u, v\rangle_{V^*\times V} = \langle R(u_h), v\rangle_{V^*\times V}\,,$$

where $B'(u_h)$ is the Gateux derivative of B given u_h. It can be shown that DPG applied to this linearized problem leads to a Gauss-Newton iteration, where

$$\min_{\Delta u} \|B'(u_h)\Delta u - r(u_h)\|^2_{V'}$$

is solved at every iteration [21].

The robust stability properties of DPG appear to carry over into the nonlinear regime as well; Moro, Peraire and Nguyen observed [4, 22] that on a single element, the HDPG method (referring to DPG applied to a hybridized Discontinuous Galerkin (HDG) method) converges, whereas HDG on its own does not. Moreover, HDPG required an order of magnitude less artificial diffusion to converge to a nonlinear solution of the compressible Navier-Stokes. Chan *et al.* [21] showed that for both the viscous Burgers' equation and the compressible Navier Stokes equations at high Mach/Reynolds number, DPG converges on an extremely coarse mesh, allowing the use of an adaptive scheme to capture shock and boundary layer phenomena starting from an initial mesh of $O(1)$ elements. Roberts reports similar results boasting similar stability for high Reynolds numbers for the incompressible Navier-Stokes equations [23].

2.2. DPG for High Performance Computing

Many of the features inherent in the DPG method appear promising in the context of high performance computing. Our goal is to design a method that eliminates human intervention as much as possible. The superior stability of the method promises to prevent a simulation from crashing which could eliminate expensive restarts on large systems. Preliminary studies on convection-diffusion suggest exceptional robustness of the method in terms of diminishing viscosity, promising successful application to a large class of flow problems. The adaptivity lent by the *error representation function* provides a reliable and automated way to start from a coarse mesh and only refine toward solution features in need. This uses compute resources much more efficiently than uniform refinements, allowing larger simulations with fewer resources. These features combine to produce a high degree of automaticity. Ultimately, it is desirable that an engineer could produce a rough mesh that just captures the geometry of the problem and start a DPG simulation that automatically picks up solution features without the user needing to jump back in and fix things.

DPG is very compute intensive compared to the associated communication and memory costs. Most of the work is spent in embarrassingly parallel local solves for the optimal test functions and local stiffness matrix assembly. Additionally, the stability properties of DPG make high order stability a triviality, and in general, high order methods tend to have a more attractive

compute/communication profile than low order methods. In our codes, we use QR factorization for optimal test function solves, but this factorization is recyclable as we essentially have many right hand sides. The division of degrees of freedom into internal vs skeleton unknowns produces a global system which can be statically condensed into a solve purely in terms of the skeleton degrees of freedom. In addition to significantly cutting down on the size of the global solve, this produces a embarrassingly parallel post-processing solve for the internal degrees of freedom. This property was one of the motivations behind the development of the hybridized discontinuous Galerkin[24] method. No matter what system of equations is being considered, DPG always produces a Hermitian (symmetric if real) positive definite stiffness matrix for the global solve. This property has not really been leveraged in our simulations so far, since we have focused on direct rather than iterative solvers, but we anticipate it might be an attractive feature in the future. As compute resources scale up, many more HPC simulations are increasingly becoming coupled in multiphysics simulations. Since the only requirement for a well-defined discrete DPG method is a well-defined continuous problem, it is certainly possible that each different part of the multiphysics simulation could be discretized with DPG – no need to develop many different methods for each part of the simulation. Already DPG has been successfully applied to a wide variety of problems in computational mechanics, as noted above.

3. Space-Time DPG

Previous explorations of the DPG method focused exclusively on steady state problems. The easiest extension of steady DPG to transient problems would be to do an implicit time stepping technique in time and use DPG for only the spatial solve at each time step. We did indeed explore this approach, but it didn't seem to be a natural fit with the adaptive features of DPG. Clearly the CourantFriedrichsLewy (CFL) condition was not binding since we were interested in implicit time integration schemes, but it can be a guiding principle for temporal accuracy in this case. So if we are interested in temporally accurate solutions, we are limited by the fact that our smallest mesh elements (which may be orders of magnitude smaller than the largest elements) are constrained to proceed at a much smaller time step than the mesh as a whole. We can either restrict the whole mesh to the smallest time step, or we can attempt some sort of local time stepping. A space-time DPG formulation presents an attractive choice as we will be able to preserve

our natural adaptivity from the steady problems while extending it in time. Thus we achieve an adaptive solution technique for transient problems in a unified framework. The obvious downside to such an approach is that for 2D spatial problems, we now have to compute on a three dimensional mesh while a spatially 3D problem becomes four dimensional.

3.1. Heat Equation

The simplest parabolic space-time problem we can consider is the heat equation. We start with a general d-dimensional spatial derivation and later simplify to spatially 1D with a few numerical experiments.

Let $\Omega(t) \subset \mathbb{R}^d$ be the spatial domain with boundary $\partial\Omega$. The heat equation is

$$\frac{\partial u}{\partial t} - \epsilon \Delta u = f\,, \quad \boldsymbol{x} \in \Omega\,,\ t \in (t_0, T) \tag{1}$$

where u is unknown heat, ϵ is the diffusion scale, f is the source term, t_0 is the start time, and T is the final time. Let $Q \subset \mathbb{R}^{d+1}$ denote the full space-time domain which is then tessellated into space-time elements K.

The second order formulation of the heat equation is really just a composition of Fourier's law and conservation of energy:

$$\begin{aligned} \boldsymbol{\sigma} - \epsilon \nabla u &= 0 \\ \frac{\partial u}{\partial t} - \nabla \cdot \boldsymbol{\sigma} &= f\,, \end{aligned} \tag{2}$$

where $\boldsymbol{\sigma}$ is the heat flux. The key insight that we will use over and over in the following problems is that we can rewrite our conservation equation in terms of a space-time divergence operator: $\nabla_{xt} \cdot () := \nabla \cdot () + \frac{\partial ()}{\partial t}$. Our new system is then

$$\begin{aligned} \frac{1}{\epsilon}\boldsymbol{\sigma} - \nabla u &= 0 \\ \nabla_{xt} \cdot \begin{pmatrix} -\boldsymbol{\sigma} \\ u \end{pmatrix} &= f\,. \end{aligned} \tag{3}$$

We now proceed with the standard DPG practice and multiply by test func-

tions $\boldsymbol{\tau}$ and v and integrate by parts over each space-time element K:

$$\begin{aligned}\left(\frac{1}{\epsilon}\boldsymbol{\sigma},\boldsymbol{\tau}\right)+(u,\nabla\cdot\boldsymbol{\tau})-\langle\hat{u},\boldsymbol{\tau}\cdot\boldsymbol{n}_x\rangle=0\\ -\left(\begin{pmatrix}-\boldsymbol{\sigma}\\ u\end{pmatrix},\nabla_{xt}v\right)+\langle\hat{t},v\rangle=f\,,\end{aligned}\tag{4}$$

where $\boldsymbol{\tau}$ is in a broken tensor product space of $\boldsymbol{H}(\mathrm{div})$ spatially multiplied by L^2 functions temporally, while v is in a broken H^1 space in full space-time. Also

$$\begin{aligned}\hat{u}&:=\mathrm{tr}(u)\\ \hat{t}&:=\mathrm{tr}(-\boldsymbol{\sigma})\cdot\boldsymbol{n}_x+\mathrm{tr}(u)\cdot n_t\end{aligned}$$

are new unknowns that live on the mesh skeleton introduced by the integration by parts (parenthesis indicate volume integrals while angle brackets indicate surface integrals). Note that the constitutive law was only integrated by parts over spatial dimensions, which means that *spatial trace* $\hat{u}$ only exists on mesh boundaries with a nonzero spatial normal component. On the other hand, *flux* $\hat{t}$ exists on all mesh boundaries, but changes nature between pure spatial and temporal edges while taking on a mixed nature on slanted boundaries. We illustrate the support of these skeleton variables in Figure 1.

It is worth noting that one technique used to control the cost of the extra dimensional solve is to divide the space-time domain into a number of space-time slabs. Since information does not travel backwards in time, we can march forward solving and adapting on each times slab before moving to the next one. The flux on the final time boundary of one slab acts as the initial condition on the next.

3.1.1. Numerical Experiments

All numerical results in this paper were generated with the Camellia DPG framework[25]. If we consider a domain $\Omega=[0,1]^2$ with an initial condition of $u=\cos(2\pi x)$ with zero flux conditions at the boundaries, the exact solution is

$$u=\cos(2\pi x)e^{-4\pi^2\epsilon t}\,.$$

We ran this with $\epsilon=10^{-2}$ on a sequence of uniform meshes and $p=2$ for the field representation of u. We were able to achieve the expected third order convergence as demonstrated in Figure 2.

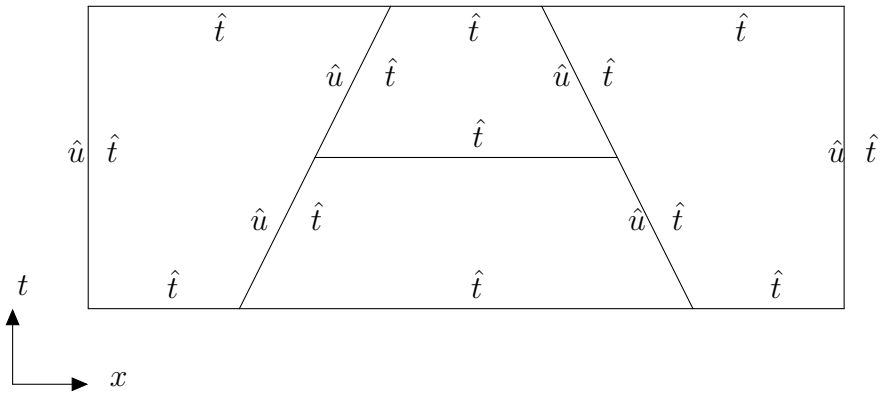


Figure 1: Support of flux and spatial trace variables

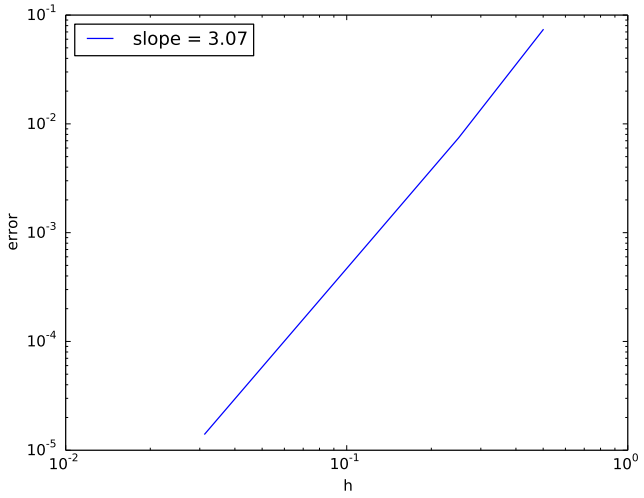
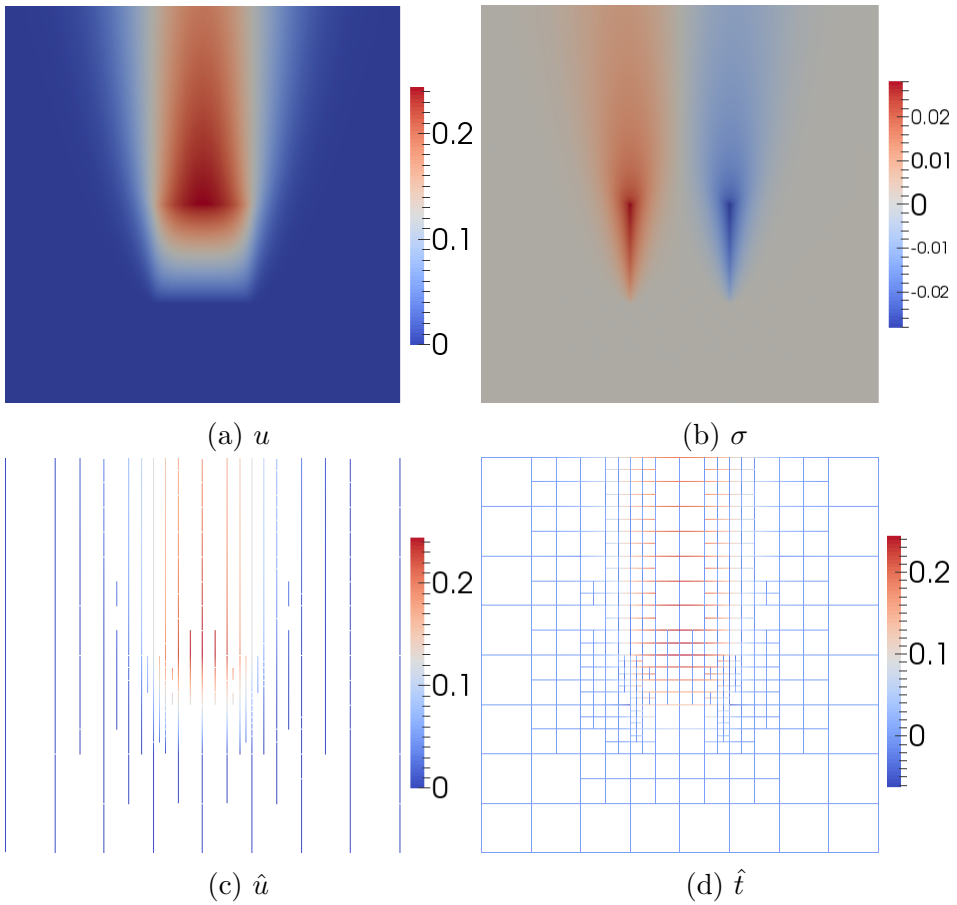


Figure 2: L^2 convergence of u for the space-time heat equation

In order to demonstrate local space-time adaptivity we consider one more problem for the heat equation. On the same domain, and with the same boundary conditions as the previous example, we let the initial heat distribution be zero. Then between $t = 0.25$ and $t = 0.5$ we turn on a pulse source term of one on $0.375 \leq x \leq 0.625$. Starting from an initial mesh of $4x4$,

we adaptively refine four times and obtain the results in Figure 3. Notice that $\hat{u}$ in Figure 3c only lives on vertical edges as was discussed earlier. Also notice that the full mesh shown in Figure 3d automatically adapts spatially and temporally to where features are rapidly changing.



(a) u (b) σ

(c) $\hat{u}$ (d) $\hat{t}$

Figure 3: Pulsed space-time heat problem after 4 refinements

4. Transient Compressible Navier-Stokes

We make a large jump from convection-diffusion to the compressible Navier-Stokes equations. The following discussion holds in any dimension, but the provided results are only for spatially 1D flows (at the time of this

writing, our code only supported a maximum of two dimensional meshes). The compressible Navier-Stokes equations are

$$\frac{\partial}{\partial t}\begin{bmatrix} \rho \\ \rho \boldsymbol{u} \\ \rho e_0 \end{bmatrix} + \nabla \cdot \begin{bmatrix} \rho \boldsymbol{u} \\ \rho \boldsymbol{u} \otimes \boldsymbol{u} + p\boldsymbol{I} - \mathbb{D} \\ \rho \boldsymbol{u} e_0 + \boldsymbol{u} p + \boldsymbol{q} - \boldsymbol{u} \cdot \mathbb{D} \end{bmatrix} = \begin{bmatrix} f_c \\ \boldsymbol{f}_m \\ f_e \end{bmatrix}, \tag{5}$$

where ρ is the density, $\boldsymbol{u}$ is the velocity, p is the pressure, $\boldsymbol{I}$ is the identity matrix, $\mathbb{D}$ is the deviatoric stress tensor or viscous stress, e_0 is the total energy, $\boldsymbol{q}$ is the heat flux, and f_c, $\boldsymbol{f}_m$, and f_e are the source terms for the continuity, momentum, and energy equations, respectively. Assuming Stokes hypothesis that $\lambda = -\frac{2}{3}\mu$,

$$\mathbb{D} = 2\mu \boldsymbol{S}^* = 2\mu \left[\frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T \right) - \frac{1}{3} \nabla \cdot \boldsymbol{u} \boldsymbol{I} \right],$$

where $\boldsymbol{S}^*$ is the trace-less viscous strain rate tensor. The heat flux is given by Fourier's law:

$$\boldsymbol{q} = -C_p \frac{\mu}{Pr} \nabla T,$$

where C_p is the specific heat at constant pressure and Pr is the laminar Prandtl number: $Pr := \frac{C_p \mu}{\lambda}$. We need to close these equations with an equation of state. An ideal gas assumption gives

$$\gamma := \frac{C_p}{C_v}, \quad p = \rho R T, \quad e = C_v T, \quad C_p - C_v = R,$$

where γ is the ratio of specific heats, C_v is the specific heat at constant volume, R is the gas constant, e is the internal energy, T is the temperature, and γ, C_p, C_v, and R are constant properties of the fluid. The total energy is defined by

$$e_0 = e + \frac{1}{2} \boldsymbol{u} \cdot \boldsymbol{u}.$$

We can write our first order system of equations in space-time as follows:

$$\frac{1}{\mu}\mathbb{D} - \left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T\right) + \frac{2}{3}\nabla \cdot \boldsymbol{u}\boldsymbol{I} = 0 \tag{6a}$$

$$\frac{Pr}{C_p \mu}\boldsymbol{q} + \nabla T = 0 \tag{6b}$$

$$\nabla_{xt} \cdot \begin{pmatrix} \rho \boldsymbol{u} \\ \rho \end{pmatrix} = f_c \tag{6c}$$

$$\nabla_{xt} \cdot \begin{pmatrix} \rho \boldsymbol{u} \otimes \boldsymbol{u} + \rho R T \boldsymbol{I} - \mathbb{D} \\ \rho \boldsymbol{u} \end{pmatrix} = \boldsymbol{f}_m \tag{6d}$$

$$\nabla_{xt} \cdot \begin{pmatrix} \rho \boldsymbol{u}\left(C_v T + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right) + \boldsymbol{u}\rho R T + \boldsymbol{q} - \boldsymbol{u}\cdot\mathbb{D} \\ \rho\left(C_v T + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right) \end{pmatrix} = f_e\,, \tag{6e}$$

where our solution variables are ρ, $\boldsymbol{u}$, T, $\mathbb{D}$, and $\boldsymbol{q}$.

4.1. Derivation of Space-Time DPG Formulation

We start with (6) and multiply by test functions $\mathbb{S}$ (symmetric tensor), $\boldsymbol{\tau}$, v_c, $\boldsymbol{v}_m$, v_e, then integrate by parts over each space-time element K:

$$\left(\frac{1}{\mu}\mathbb{D}, \mathbb{S}\right) + (2\boldsymbol{u}, \nabla\cdot\mathbb{S}) - \left(\frac{2}{3}\boldsymbol{u}, \nabla \operatorname{tr}\mathbb{S}\right) - \left\langle \frac{4}{3}\hat{\boldsymbol{u}}, \mathbb{S}\boldsymbol{n}_x \right\rangle = 0 \tag{7a}$$

$$\left(\frac{Pr}{C_p\mu}\boldsymbol{q}, \boldsymbol{\tau}\right) - (T, \nabla\cdot\boldsymbol{\tau}) + \left\langle \hat{T}, \tau_n \right\rangle = 0 \tag{7b}$$

$$-\left(\begin{pmatrix} \rho\boldsymbol{u} \\ \rho \end{pmatrix}, \nabla_{xt} v_c\right) + \left\langle \hat{t}_c, v_c \right\rangle = (f_c, v_c) \tag{7c}$$

$$-\left(\begin{pmatrix} \rho\boldsymbol{u}\otimes\boldsymbol{u} + \rho R T \boldsymbol{I} - \mathbb{D} \\ \rho\boldsymbol{u} \end{pmatrix}, \nabla_{xt}\boldsymbol{v}_m\right) + \left\langle \hat{\boldsymbol{t}}_m, \boldsymbol{v}_m \right\rangle = (\boldsymbol{f}_m, \boldsymbol{v}_m) \tag{7d}$$

$$-\left(\begin{pmatrix} \rho\boldsymbol{u}\left(C_v T + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right) + \boldsymbol{u}\rho R T + \boldsymbol{q} - \boldsymbol{u}\cdot\mathbb{D} \\ \rho\left(C_v T + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right) \end{pmatrix}, \nabla_{xt} v_e\right) + \left\langle \hat{t}_e, v_e \right\rangle = (f_e, v_e)\,, \tag{7e}$$

where

$$\begin{aligned}
\hat{\boldsymbol{u}} &= \operatorname{tr}(\boldsymbol{u}) \\
\hat{T} &= \operatorname{tr}(T) \\
\hat{t}_c &= \operatorname{tr}(\rho \boldsymbol{u}) \cdot \boldsymbol{n}_x + \operatorname{tr}(\rho)\, n_t \\
\hat{\boldsymbol{t}}_m &= \operatorname{tr}(\rho \boldsymbol{u} \otimes \boldsymbol{u} + \rho R T \boldsymbol{I} - \mathbb{D}) \cdot \boldsymbol{n}_x + \operatorname{tr}(\rho \boldsymbol{u})\, n_t \\
\hat{t}_e &= \operatorname{tr}\left(\rho \boldsymbol{u}\left(C_v T + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right) + \boldsymbol{u}\rho R T + \boldsymbol{q} - \boldsymbol{u}\cdot\mathbb{D}\right)\cdot \boldsymbol{n}_x + \operatorname{tr}\left(\rho\left(C_v T + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right)\right) n_t\,.
\end{aligned}$$

Note that integrating $\mathbb{S}$ against the symmetric gradient only picks up the symmetric part. This is a much more complicated system of equations than we had for the space-time heat equation, but the situation has many similarities. Test function $\boldsymbol{\tau} \in \boldsymbol{H}(\operatorname{div}, K)$ where the divergence is taken only over spatial dimensions, $v_c, v_e \in H^1(K)$, and $\boldsymbol{v}_m \in \boldsymbol{H}^1(K)$. These are all familiar spaces from our work with the heat equation. Unfortunately, $\mathbb{S}$ has some nonstandard requirements: each of the $d \times d$ components must be at least in $L^2(K)$, $\nabla \cdot \mathbb{S} \in \mathbf{L}^2(K)$, and $\nabla \operatorname{tr} \mathbb{S} \in \mathbf{L}^2(K)$.

4.1.1. Linearization

We follow a standard residual-Jacobian linearization procedure discussed above. Let $U = \left\{\rho, \boldsymbol{u}, T, \mathbb{D}, \boldsymbol{q}, \hat{\boldsymbol{u}}, \hat{e}, \hat{t}_c, \hat{\boldsymbol{t}}_m, \hat{t}_e\right\}$ be a group solution variable which we can decompose into two parts: $U := \tilde{U} + \Delta U$, where $\tilde{U} = \left\{\tilde{\rho}, \tilde{\boldsymbol{u}}, \tilde{T}, \tilde{\mathbb{D}}, \mathbf{0}, \mathbf{0}, 0, 0, \mathbf{0}, 0\right\}$ is the previous iteration approximation, and $\Delta U = \left\{\Delta\rho, \Delta\boldsymbol{u}, \Delta T, \Delta\mathbb{D}, \boldsymbol{q}, \hat{\boldsymbol{u}}, \hat{e}, \hat{t}_c, \hat{\boldsymbol{t}}_m, \hat{t}_e\right\}$ is the update. Note that $\tilde{U}$ only contains terms which participate in nonlinearities in (7) while ΔU contains the full linear terms and the updates to the nonlinear terms. Also, we drop the Δ and $\tilde{\cdot}$ notation for linear terms. Define residual $R(U)$ as the left hand side of (7) minus the right hand side. Approximating $R(U) = 0$ by $R(\tilde{U}) + R'(\tilde{U})\Delta U = 0$, where $R'(\tilde{U})$ is the Jacobian of R evaluated at $\tilde{U}$, we get a linear system:

$$R'(\tilde{U})\Delta U = -R(\tilde{U})\,. \tag{8}$$

This is an instance of a Gauss-Newton nonlinear solve. We only need to define our Jacobian and residual for each component of (7). The Jacobian

of our compressible Navier-Stokes system, $R'(\tilde{U})\Delta U$ is

$$
\begin{aligned}
&\left(\frac{1}{\mu}\Delta\mathbb{D},\mathbb{S}\right)+(2\Delta\boldsymbol{u},\nabla\cdot\mathbb{S})-\left(\frac{2}{3}\Delta\boldsymbol{u},\nabla\operatorname{tr}\mathbb{S}\right)-\left\langle\frac{4}{3}\hat{\boldsymbol{u}},\mathbb{S}\boldsymbol{n}_x\right\rangle\\
&+\left(\frac{Pr}{C_p\mu}\boldsymbol{q},\boldsymbol{\tau}\right)-(\Delta T,\nabla\cdot\boldsymbol{\tau})+\left\langle\hat{T},\tau_n\right\rangle\\
&-\left(\begin{pmatrix}\Delta\rho\tilde{\boldsymbol{u}}+\tilde{\rho}\Delta\boldsymbol{u}\\ \Delta\rho\end{pmatrix},\nabla_{xt}v_c\right)+\langle\hat{t}_c,v_c\rangle\\
&-\left(\begin{pmatrix}\Delta\rho\tilde{\boldsymbol{u}}\otimes\tilde{\boldsymbol{u}}+\tilde{\rho}\Delta\boldsymbol{u}\otimes\tilde{\boldsymbol{u}}+\tilde{\rho}\tilde{\boldsymbol{u}}\otimes\Delta\boldsymbol{u}+\left(\Delta\rho R\tilde{T}+\tilde{\rho}R\Delta T\right)\boldsymbol{I}-\Delta\mathbb{D}\\ \Delta\rho\tilde{\boldsymbol{u}}+\tilde{\rho}\Delta\boldsymbol{u}\end{pmatrix},\nabla_{xt}\boldsymbol{v}_m\right)+\langle\hat{\boldsymbol{t}}_m,\boldsymbol{v}_m\rangle\\
&-\left(\begin{pmatrix}\left[C_v\Delta\rho\tilde{T}\tilde{\boldsymbol{u}}+C_v\tilde{\rho}\Delta T\tilde{\boldsymbol{u}}+C_v\tilde{\rho}\tilde{T}\Delta\boldsymbol{u}+\frac{1}{2}\left(\Delta\rho\tilde{\boldsymbol{u}}\cdot\tilde{\boldsymbol{u}}\tilde{\boldsymbol{u}}+\tilde{\rho}\Delta\boldsymbol{u}\cdot\tilde{\boldsymbol{u}}\tilde{\boldsymbol{u}}+\tilde{\rho}\tilde{\boldsymbol{u}}\cdot\Delta\boldsymbol{u}\tilde{\boldsymbol{u}}+\tilde{\rho}\tilde{\boldsymbol{u}}\cdot\tilde{\boldsymbol{u}}\Delta\boldsymbol{u}\right)\right.\\ \left.+R\left(\Delta\rho\tilde{T}\tilde{\boldsymbol{u}}+\tilde{\rho}\Delta T\tilde{\boldsymbol{u}}+\tilde{\rho}\tilde{T}\Delta\boldsymbol{u}\right)+\boldsymbol{q}-\Delta\boldsymbol{u}\cdot\tilde{\mathbb{D}}-\tilde{\boldsymbol{u}}\cdot\Delta\mathbb{D}\right]\\ C_v\Delta\rho\tilde{T}+C_v\tilde{\rho}\Delta T+\frac{1}{2}\left(\Delta\rho\tilde{\boldsymbol{u}}\cdot\tilde{\boldsymbol{u}}+\tilde{\rho}\Delta\boldsymbol{u}\cdot\tilde{\boldsymbol{u}}+\tilde{\rho}\tilde{\boldsymbol{u}}\cdot\Delta\boldsymbol{u}\right)\end{pmatrix},\nabla_{xt}v_e\right)\\
&+\langle\hat{t}_e,v_e\rangle\ .
\end{aligned}
\tag{9}
$$

The residual, $R(\tilde{U})$, is then

$$
\begin{aligned}
&\left(\frac{1}{\mu}\tilde{\mathbb{D}},\mathbb{S}\right)+(2\tilde{\boldsymbol{u}},\nabla\cdot\mathbb{S})-\left(\frac{2}{3}\tilde{\boldsymbol{u}},\nabla\operatorname{tr}\mathbb{S}\right)\\
&-\left(\tilde{T},\nabla\cdot\boldsymbol{\tau}\right)\\
&-\left(\begin{pmatrix}\tilde{\rho}\tilde{\boldsymbol{u}}\\ \tilde{\rho}\end{pmatrix},\nabla_{xt}v_c\right)-(f_c,v_c)\\
&-\left(\begin{pmatrix}\tilde{\rho}\tilde{\boldsymbol{u}}\otimes\tilde{\boldsymbol{u}}+\tilde{\rho}R\tilde{T}\boldsymbol{I}-\tilde{\mathbb{D}}\\ \tilde{\rho}\tilde{\boldsymbol{u}}\end{pmatrix},\nabla_{xt}\boldsymbol{v}_m\right)-(\boldsymbol{f}_m,\boldsymbol{v}_m)\\
&-\left(\begin{pmatrix}\tilde{\rho}\tilde{\boldsymbol{u}}\left(C_v\tilde{T}+\frac{1}{2}\tilde{\boldsymbol{u}}\cdot\tilde{\boldsymbol{u}}\right)+\tilde{\boldsymbol{u}}\tilde{\rho}R\tilde{T}-\tilde{\boldsymbol{u}}\cdot\tilde{\mathbb{D}}\\ \tilde{\rho}\left(C_v\tilde{T}+\frac{1}{2}\tilde{\boldsymbol{u}}\cdot\tilde{\boldsymbol{u}}\right)\end{pmatrix},\nabla_{xt}v_e\right)-(f_e,v_e)\ .
\end{aligned}
\tag{10}
$$

4.1.2. Test Norm

Usually our first choice for a DPG test norm is the graph norm which comes from upgrading the seminorm derived from the adjoint to a full norm.

We start by grouping terms in (9) by trial variable to get

$$
\begin{aligned}
&\left(\Delta\mathbb{D}, \frac{1}{\mu}\mathbb{S} + \nabla\boldsymbol{v}_m + \nabla v_e \otimes \tilde{\boldsymbol{u}}\right) \\
&+ \left(\boldsymbol{q}, \frac{Pr}{C_p\mu}\boldsymbol{\tau} - \nabla v_e\right) \\
&+ \left(\Delta\rho, -\tilde{\boldsymbol{u}}\cdot\nabla v_c - \frac{\partial v_c}{\partial t} - \tilde{\boldsymbol{u}}\otimes\tilde{\boldsymbol{u}} : \nabla\boldsymbol{v}_m - R\tilde{T}\nabla\cdot\boldsymbol{v}_m - \tilde{\boldsymbol{u}}\cdot\frac{\partial\boldsymbol{v}_m}{\partial t}\right. \\
&\quad \left. -C_v\tilde{T}\tilde{\boldsymbol{u}}\cdot\nabla v_e - \frac{1}{2}\tilde{\boldsymbol{u}}\cdot\tilde{\boldsymbol{u}}\tilde{\boldsymbol{u}}\cdot\nabla v_e - R\tilde{T}\tilde{\boldsymbol{u}}\nabla v_e - C_v\tilde{T}\frac{\partial v_e}{\partial t} - \frac{1}{2}\tilde{\boldsymbol{u}}\cdot\tilde{\boldsymbol{u}}\frac{\partial v_e}{\partial t}\right) \\
&+ \left(\Delta\boldsymbol{u}, 2\nabla\cdot\mathbb{S} - \frac{2}{3}\nabla\,\mathrm{tr}\,\mathbb{S} - \tilde{\rho}\nabla v_c - \tilde{\rho}\tilde{\boldsymbol{u}}\cdot\nabla\boldsymbol{v}_m - \tilde{\rho}\nabla\boldsymbol{v}_m\cdot\tilde{\boldsymbol{u}}\right. \\
&\quad -\tilde{\rho}\frac{\partial\boldsymbol{v}_m}{\partial t} - C_v\tilde{\rho}\tilde{T}\nabla v_e - R\tilde{\rho}\tilde{T}_\nabla v_e + \tilde{\mathbb{D}}\cdot\nabla v_e \\
&\quad \left. -\frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}}\cdot\tilde{\boldsymbol{u}}\nabla v_e - \frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}}\cdot\nabla v_e\tilde{\boldsymbol{u}} - \frac{1}{2}\tilde{\rho}\nabla v_e\cdot\tilde{\boldsymbol{u}}\tilde{\boldsymbol{u}} - \frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}}\frac{\partial v_e}{\partial t} - \frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}}\frac{\partial v_e}{\partial t}\right) \\
&+ \left(\Delta T, -\nabla\cdot\boldsymbol{\tau} - R\tilde{\rho}\nabla\cdot\boldsymbol{v}_m - C_v\tilde{\rho}\tilde{\boldsymbol{u}}\nabla v_e - R\tilde{\rho}\tilde{\boldsymbol{u}}\nabla v_e - C_v\tilde{\rho}\frac{\partial v_e}{\partial t}\right) \\
&+ \left(\hat{\boldsymbol{u}}, -\frac{4}{3}\mathbb{S}\boldsymbol{n}_x\right) + \left(\hat{T}, \tau_n\right) + \left(\hat{t}_c, v_c\right) + \left(\hat{\boldsymbol{t}}_m, \boldsymbol{v}_m\right) + \left(\hat{t}_e, v_e\right)\ .
\end{aligned}
\tag{11}
$$

Then the graph norm would be defined by

$$
\begin{aligned}
&\left\| \frac{1}{\mu}\mathbb{S} + \nabla \boldsymbol{v}_m + \nabla v_e \otimes \tilde{\boldsymbol{u}} \right\|^2 \\
&+ \left\| \frac{Pr}{C_p \mu}\boldsymbol{\tau} - \nabla v_e \right\|^2 \\
&+ \Bigg\| -\tilde{\boldsymbol{u}} \cdot \nabla v_c - \frac{\partial v_c}{\partial t} - \tilde{\boldsymbol{u}} \otimes \tilde{\boldsymbol{u}} : \nabla \boldsymbol{v}_m - R\tilde{T}\nabla \cdot \boldsymbol{v}_m - \tilde{\boldsymbol{u}} \cdot \frac{\partial \boldsymbol{v}_m}{\partial t} \\
&\quad - C_v \tilde{T}\tilde{\boldsymbol{u}} \cdot \nabla v_e - \frac{1}{2}\tilde{\boldsymbol{u}} \cdot \tilde{\boldsymbol{u}}\tilde{\boldsymbol{u}} \cdot \nabla v_e - R\tilde{T}\tilde{\boldsymbol{u}}\nabla v_e - C_v \tilde{T}\frac{\partial v_e}{\partial t} - \frac{1}{2}\tilde{\boldsymbol{u}} \cdot \tilde{\boldsymbol{u}}\frac{\partial v_e}{\partial t} \Bigg\|^2 \\
&+ \Bigg\| 2\nabla \cdot \mathbb{S} - \frac{2}{3}\nabla \operatorname{tr}\mathbb{S} - \tilde{\rho}\nabla v_c - \tilde{\rho}\tilde{\boldsymbol{u}} \cdot \nabla \boldsymbol{v}_m - \tilde{\rho}\nabla \boldsymbol{v}_m \cdot \tilde{\boldsymbol{u}} \\
&\quad - \tilde{\rho}\frac{\partial \boldsymbol{v}_m}{\partial t} - C_v \tilde{\rho}\tilde{T}\nabla v_e - R\tilde{\rho}\tilde{T}\nabla v_e + \tilde{\mathbb{D}} \cdot \nabla v_e \\
&\quad - \frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}} \cdot \tilde{\boldsymbol{u}}\nabla v_e - \frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}} \cdot \nabla v_e \tilde{\boldsymbol{u}} - \frac{1}{2}\tilde{\rho}\nabla v_e \cdot \tilde{\boldsymbol{u}}\tilde{\boldsymbol{u}} - \frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}}\frac{\partial v_e}{\partial t} - \frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}}\frac{\partial v_e}{\partial t} \Bigg\|^2 \\
&+ \left\| -\nabla \cdot \boldsymbol{\tau} - R\tilde{\rho}\nabla \cdot \boldsymbol{v}_m - C_v \tilde{\rho}\tilde{\boldsymbol{u}}\nabla v_e - R\tilde{\rho}\tilde{\boldsymbol{u}}\nabla v_e - C_v \tilde{\rho}\frac{\partial v_e}{\partial t} \right\|^2 \\
&+ \alpha_c \left\| v_c \right\|^2 + \alpha_m \left\| \boldsymbol{v}_m \right\|^2 + \alpha_e \left\| v_e \right\|^2 + \alpha_s \left\| \mathbb{S} \right\|^2 + \alpha_f \left\| \boldsymbol{\tau} \right\|^2 ,
\end{aligned}
\tag{12}
$$

where α_c, α_m, α_e, α_s, and α_f are scaling constants, usually one.

Unfortunately, the graph norm develops unresolvable internal boundary layers in the optimal test functions leading to a non-robust local solve for steady convection-diffusion or Navier-Stokes, and we observed that non-robustness manifests when we tried to use this norm for transient simulations as well. For steady state DPG, we developed a robust test norm for convection-diffusion and drew analogies to create a robust norm for Navier-Stokes.

A similar analysis for transient convection-diffusion is forthcoming, so it is less clear how to proceed on developing a robust norm for transient Navier-Stokes (development of robust norms for compressible Navier-Stokes will be the topic of an upcoming paper). Nevertheless, we can make some guesses about how to modify the test norm in order to obtain some preliminary results. The graph norm has proven to be sufficient for simulations of pure convection. So an obvious first guess might be to take the graph norm on the convective quantities and decouple the viscous terms. This modified graph

norm is then:

$$
\begin{aligned}
&\|\nabla \boldsymbol{v}_m + \nabla v_e \otimes \tilde{\boldsymbol{u}}\|^2 \\
&+ \|-\nabla v_e\|^2 \\
&+ \left\| -\tilde{\boldsymbol{u}} \cdot \nabla v_c - \frac{\partial v_c}{\partial t} - \tilde{\boldsymbol{u}} \otimes \tilde{\boldsymbol{u}} : \nabla \boldsymbol{v}_m - R\tilde{T}\nabla \cdot \boldsymbol{v}_m - \tilde{\boldsymbol{u}} \cdot \frac{\partial \boldsymbol{v}_m}{\partial t} \right. \\
&\quad \left. -C_v\tilde{T}\tilde{\boldsymbol{u}} \cdot \nabla v_e - \frac{1}{2}\tilde{\boldsymbol{u}} \cdot \tilde{\boldsymbol{u}}\tilde{\boldsymbol{u}} \cdot \nabla v_e - R\tilde{T}\tilde{\boldsymbol{u}}\nabla v_e - C_v\tilde{T}\frac{\partial v_e}{\partial t} - \frac{1}{2}\tilde{\boldsymbol{u}} \cdot \tilde{\boldsymbol{u}}\frac{\partial v_e}{\partial t} \right\|^2 \\
&+ \left\| -\tilde{\rho}\nabla v_c - \tilde{\rho}\tilde{\boldsymbol{u}} \cdot \nabla \boldsymbol{v}_m - \tilde{\rho}\nabla \boldsymbol{v}_m \cdot \tilde{\boldsymbol{u}} - \tilde{\rho}\frac{\partial \boldsymbol{v}_m}{\partial t} - C_v\tilde{\rho}\tilde{T}\nabla v_e - R\tilde{\rho}\tilde{T}\nabla v_e + \tilde{\mathbb{D}} \cdot \nabla v_e \right. \\
&\quad \left. -\frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}} \cdot \tilde{\boldsymbol{u}}\nabla v_e - \frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}} \cdot \nabla v_e\tilde{\boldsymbol{u}} - \frac{1}{2}\tilde{\rho}\nabla v_e \cdot \tilde{\boldsymbol{u}}\tilde{\boldsymbol{u}} - \frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}}\frac{\partial v_e}{\partial t} - \frac{1}{2}\tilde{\rho}\tilde{\boldsymbol{u}}\frac{\partial v_e}{\partial t} \right\|^2 \\
&+ \left\| -R\tilde{\rho}\nabla \cdot \boldsymbol{v}_m - C_v\tilde{\rho}\tilde{\boldsymbol{u}}\nabla v_e - R\tilde{\rho}\tilde{\boldsymbol{u}}\nabla v_e - C_v\tilde{\rho}\frac{\partial v_e}{\partial t} \right\|^2 \\
&+ \frac{1}{\mu}\|\mathbb{S}\|^2 + \left\| 2\nabla \cdot \mathbb{S} - \frac{2}{3}\nabla \operatorname{tr}\mathbb{S} \right\|^2 + \frac{Pr}{c_p\mu}\|\boldsymbol{\tau}\|^2 + \|\nabla \cdot \boldsymbol{\tau}\|^2 \\
&+ \|v_c\|^2 + \|\boldsymbol{v}_m\|^2 + \|v_e\|^2 \ .
\end{aligned}
\tag{13}
$$

From a number of numerical tests, it appears that this norm is not completely robust, but it does seem to perform somewhat better than the standard graph norm.

4.2. Numerical Experiments

We consider two 1D test problems as verification, each of which has an analytical solution based on an inviscid flow assumption (Euler's equations). However, in the absence of viscosity, Euler's equations can have multiple solutions, and most numerical methods introduce a certain amount of artificial viscosity in order to select a unique solution. Most schemes also require the artificial viscosity to scale in some sense with mesh size so that they can effectively handle shocks. We run our simulations without any artificial viscosity, but in order to get a well-posed problem, we do introduce a small amount of physical viscosity: $\mu = 10^{-5}$ for the Sod shocktube and $\mu = 10^{-3}$ for the Noh implosion. Essentially we are just simulating low viscosity Navier-Stokes as a stand-in for the unsolvable pure Euler. We solve with second order poly-

nomial representation of field and flux variables, third order for traces, and fifth order for test functions.

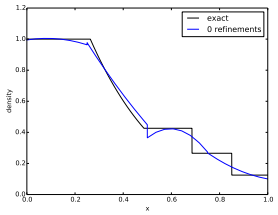
4.2.1. Sod Shock Tube

The Sod shock tube problem was developed by Gary Sod in 1978[26], and has proven to be a popular problem for verification of compressible Navier-Stokes and Euler solvers. It serves to verify that a numerical method can effectively handle a rarefaction wave, material discontinuity, and shock wave all in one domain. The domain of interest is a shock tube of length 1 with a material interface in the middle. The material on the left has initial conditions of $(\rho_L, p_L, u_L) = (1, 1, 0)$ while the material on the right has $(\rho_R, p_R, u_R) = (0.125, 0.1, 0)$; both materials have $\gamma = 1.4$. The $t = 0$ the interface between the materials is broken, and shock wave propagates into the right material, while a rarefaction wave moves left. The analytical solution is self-similar, but it is common to take $t = 0.2$ as a final time. At this time the shock wave and rarefaction waves have not hit the boundaries, so it is sufficient to set boundary conditions corresponding to the initial conditions. In our case, we set $\hat{t}_c = \hat{t}_m = \hat{t}_e = 0$ on the left and right boundaries, while the fluxes are set equal to the discontinuous initial conditions on the $t = 0$ boundary. No boundary condition is required on the $t = 0.2$ boundary since the equations are hyperbolic in time. We solve this with one continuous time slab starting with only 4 space-time elements. The field variables are represented with quadratics.

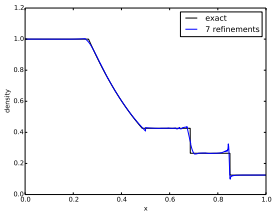
The results are plotted in Figure 4 for three different refinement levels: the initial coarse mesh, 7 adaptive refinements, and 14 refinements. The coarsest mesh is obviously not sufficient to resolve the features of the flow, however it is somewhat representative of the exact solution. We see significant overshoots and undershoots as adaptivity begins to refine near the shock, but these die away as we resolve to the viscous length scale. There are some unnecessary refinements in a sliver to the right of the shock, but we believe that this may be due to the unoptimized test norm that we are using. The ability of the test norm to affect refinement patterns was observed in [19].

4.2.2. Noh Implosion

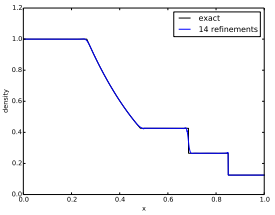
The Noh implosion problem[27] is another standard test for Euler solvers. The initial conditions are of an ideal gas with $\gamma = 5/3$, zero pressure, uniform initial density of 1, and uniform velocity toward the center of the domain. An infinitely strong shock propagates outward at a speed of 1/3. For 1D



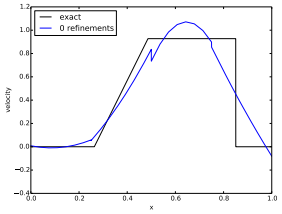
(a) Density, initial mesh



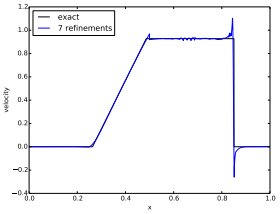
(b) After 7 refinements



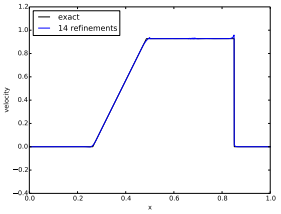
(c) After 14 refinements



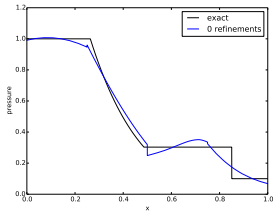
(d) Velocity, initial mesh



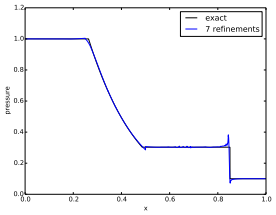
(e) After 7 refinements



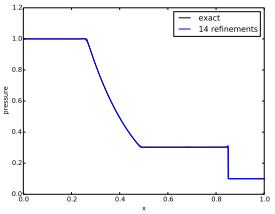
(f) After 14 refinements



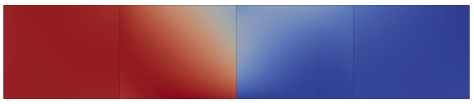
(g) Pressure, initial mesh



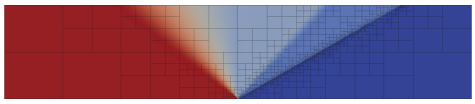
(h) After 7 refinements



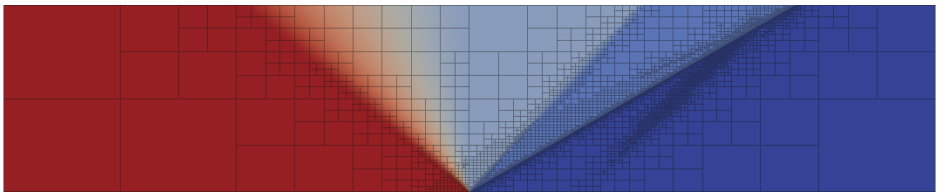
(i) After 14 refinements



(j) Density with initial mesh



(k) Density after 7 refinements



(l) Density after 14 refinements

Figure 4: Sod problem with final time $t = 0.2$

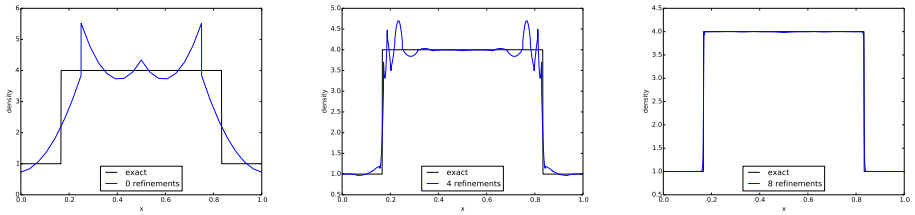
flow, the post shock density jumps to 4. We run this problem to a final time of $t = 1.0$. The longer time nature of this problem recommended the use of multiple time slabs rather than a single solve like the previous problem. We run with four time slabs of thickness 0.25 each with 4 initial space-time elements. We run the first slab to 8 adaptive refinements and set the initial conditions on the next slab to the refined solution on the previous slab.

Each of the slabs are put together into one long time solution in Figure 5. Again we plot the solution on the initial mesh, a halfway resolved mesh, and a final mesh after 8 refinement steps. We get some spurious behavior around the shock on the middle mesh, but this goes away by the final mesh. We observe the same behavior with overshoots and undershoots that we saw with the Sod problem. Again, unnecessary refinements show up, but in this case, we believe they are more of a response to the spurious shock behavior on mid-resolution meshes.

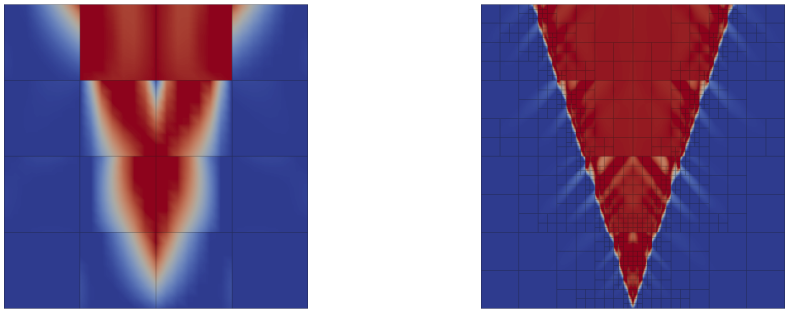
5. Conclusions and Future Work

We've presented the initial development of a space-time DPG method for parabolic PDEs with automatic adaptivity in space and time. The features of the method appear to be promising for deployment on high performance computing systems. The key contrast to steady state DPG is fluxes appear from integration of parts of a conservation law over all of space-time while traces only appear on the spatial skeleton of the mesh. Aside from this implementation detail, the other features DPG carry through to space-time completely intact. Preliminary results for spatially 1D flow are encouraging. As with steady state DPG, choice of test norm for convection-diffusion type problems can have a significant impact on the quality of the solution.

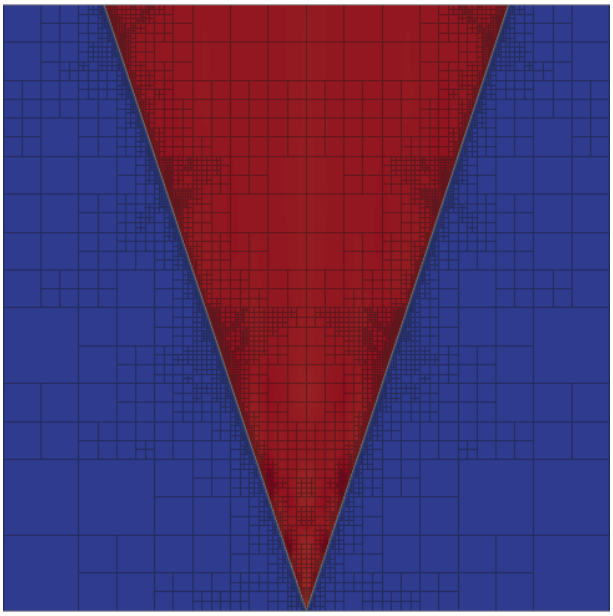
The choice of test norm for transient convection-diffusion is currently under active research and will appear in an upcoming paper. Included in this paper will be a proof of robustness of the method in space-time. Our foremost priority currently is to extend these results to 2D and 3D flows. With a 2D capable code, we can then pursue transient incompressible problems. Finally, there is a lot to study as we scale the code up to larger compute systems. Already several bottlenecks have emerged in our current process and we are working on addressing these. One such bottleneck is our current use of a direct solver for the global solve. An iterative solver should scale much better, and we are actively exploring preconditioners for DPG.



(a) Density, initial mesh (b) After 4 refinements (c) After 8 refinements



(d) Density with initial mesh (e) Density after 4 refinements



(f) Density after 8 refinements

Figure 5: Noh problem with final time $t = 1.0$

References

[1] L. Demkowicz, J. Gopalakrishnan, Recent Developments in Discontinuous Galerkin Finite Element Methods for Partial Differential Equations (eds. X. Feng, O. Karakashian, Y. Xing), Vol. 157, IMA Volumes in Mathematics and its Applications, 2014, Ch. An Overview of the DPG Method, pp. 149–180.

[2] I. Babuška, Error-bounds for finite element method, Numer. Math 16.

[3] J. Gopalakrishnan, W. Qiu, An analysis of the practical DPG method, Tech. rep., IMA, submitted (2011).

[4] D. Moro, N. Nguyen, J. Peraire, A hybridized discontinuous Petrov-Galerkin scheme for scalar conservation laws, Int. J. Num. Meth. Eng.

[5] L. Demkowicz, J. Gopalakrishnan, A primal DPG method without a first order reformulation, Comp. Math. Appl. 66 (2013) 1058–1064.

[6] L. Demkowicz, N. Heuer, Robust DPG method for convection-dominated diffusion problems, SIAM J. Numer. Anal. 51 (5) (2013) 1514–2537.

[7] J. Chan, N. Heuer, T. Bui-Thanh, L. Demkowicz, A robust DPG method for convection-dominated diffusion problems II: Adjoint boundary conditions and mesh-dependent test norms, Comp. Math. Appl. 67 (4) (2014) 771 – 795, high-order Finite Element Approximation for Partial Differential Equations. doi:http://dx.doi.org/10.1016/j.camwa.2013.06.010.
URL http://www.sciencedirect.com/science/article/pii/S0898122113003751

[8] L. Demkowicz, J. Gopalakrishnan, Analysis of the DPG method for the Poisson equation, SIAM J. Numer. Anal. 49 (5) (2011) 1788–1809. doi:10.1137/100809799.

[9] L. Demkowicz, J. Gopalakrishnan, I. Muga, J. Zitelli, Wavenumber explicit analysis of a DPG method for the multidimensional Helmholtz equation, Comput. Methods in Appl. Mech. Eng. 213216 (0) (2012) 126 – 138. doi:http://dx.doi.org/10.1016/j.cma.2011.11.024.

[10] J. Gopalakrishnan, I. Muga, N. Olivares, Dispersive and dissipative errors in the DPG method with scaled norms for helmholtz equation, SIAM J. Sci. Comput. 36 (1) (2014) A20–A39. doi:10.1137/130918186.

[11] J. Zitelli, I. Muga, L. Demkowicz, J. Gopalakrishnan, D. Pardo, V. Calo, A class of discontinuous Petrov-Galerkin methods. Part IV: Wave propagation problems, J. Comp. Phys. 230 (2011) 2406–2432.

[12] J. Bramwell, L. Demkowicz, J. Gopalakrishnan, W. Qiu, A locking-free *hp* DPG method for linear elasticity with symmetric stresses, Numer. Math. 122 (4) (2012) 671–707. doi:10.1007/s00211-012-0476-6.

[13] A. Niemi, J. Bramwell, L. Demkowicz, Discontinuous Petrov-Galerkin method with optimal test functions for thin-body problems in solid mechanics, Comput. Methods in Appl. Mech. Eng. 200 (9-12) (2011) 1291–1300.

[14] J. Bramwell, L. Demkowicz, W. Qiu, Solution of dual–mixed elasticity equations using Arnold–Falk–Winther element and discontinuous Petrov Galerkin method. A comparison, Tech. Rep. 23, ICES (2010).

[15] L. Demkowicz, J. Li, Numerical simulations of cloaking problems using a DPG method, Comput. Mech. 51 (5) (2013) 661–672. doi:10.1007/s00466-012-0744-4.

[16] C. Wieners, B. Wohlmuth, Robust operator estimates, Tech. rep., Oberwolfach Reports (2013).

[17] L. Demkowicz, J. Gopalakrishnan, A. Niemi, A class of discontinuous Petrov-Galerkin methods. Part III: Adaptivity, Appl. Numer. Math. 62 (4) (2012) 396–427. doi:10.1016/j.apnum.2011.09.002.

[18] J. Chan, J. Gopalakrishnan, L. Demkowicz, Global properties of DPG test spaces for convection-diffusion problems, Tech. rep., ICES (2013).

[19] T. Ellis, L. Demkowicz, J. Chan, Locally conservative discontinuous Petrov-Galerkin finite elements for fluid problems, Comp. Math. Appl.

[20] N. Roberts, T. Bui-Thanh, L. Demkowicz, The DPG method for the Stokes problem, Comp. Math. Appl. 67 (4) (2014) 966 – 995, high-order Finite Element Approximation for Partial Differential Equations. doi:http://dx.doi.org/10.1016/j.camwa.2013.12.015.

[21] J. Chan, L. Demkowicz, R. Moser, A DPG method for steady viscous compressible flow, Comput. Fluids 98 (0) (2014) 69

– 90, 12th USNCCM mini-symposium of High-Order Methods for Computational Fluid Dynamics - A special issue dedicated to the 80th birthday of Professor Antony Jameson. doi:http://dx.doi.org/10.1016/j.compfluid.2014.02.024.
URL http://www.sciencedirect.com/science/article/pii/S0045793014000875

[22] D. Moro, N. Nguyen, J. Peraire, A Hybridized Discontinuous Petrov-Galerkin scheme for compressible flows, Master's thesis, Massachusetts Institute of Technology (2011).

[23] N. Roberts, A discontinuous Petrov-Galerkin methodology for incompressible flow problems, Ph.D. thesis, University of Texas at Austin (2013).

[24] B. Cockburn, J. Gopalakrishnan, R. Lazarov, Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems, SIAM J. Numer. Anal. 47 (2) (2009) 1319–1365.

[25] N. Roberts, Camellia: A software framework for discontinuous Petrov-Galerkin methods, Comp. Math. Appl.Submitted.

[26] G. Sod, A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws, J. Comp. Phys. 27 (1) (1978) 1 – 31. doi:http://dx.doi.org/10.1016/0021-9991(78)90023-2.

[27] W. Noh, Errors for calculations of strong shocks using an artificial viscosity and an artificial heat flux, J. Comp. Phys. 72 (1) (1987) 78 – 120. doi:http://dx.doi.org/10.1016/0021-9991(87)90074-X.