# A Space-Time DPG Method for Fluid Dynamics

**by**

**Truman E. Ellis, M.S.**

**DISSERTATION PROPOSAL**

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2014

# Table of Contents

# Chapter 1

# Introduction

## 1.1 Motivation

Computational science has revolutionized the engineering design process – enabling design analysis and optimization to be done virtually before expensive physical prototypes need to be built. However, some fields of engineering analysis lend themselves to a computational approach much easier than others. Fluid dynamics has long been one of the most challenging engineering disciplines to simulate via numerical techniques. Aside from the inherent modeling challenges presented by fluid turbulence, many fluid flows can be characterized as singularly perturbed problems – problems in which the viscosity length scale is many orders of maginitude smaller than the large scale features of the flow. This has necessitated the need for meshes with large gradations in resolution to enable resolution of boundary layers while being computationally efficient in the free stream. Traditionally, these meshes would be custom designed by a domain expert who could predict which parts of the domain would need more resolution than others. On top of this, many numerical techniques would fail to converge unless the presented initial mesh was in the "asymptotic regime", i.e. the physics could by somewhat sufficiently represented. These requirements made mesh generation a laborious and far from automated procedure.

The failure of many numerical methods in the "pre-asymptotic regime" can be characterized mathematically as a loss of stability on coarse meshes. The stability characteristics of a broad class of finite element methods can be analyzed according to the Ladyženskaja-Babuška-Brezzi

condition. Leszek Demkowicz and Jay Gopalakrishnan first proposed the discontinuous Petrov-Galerkin method in 2009[1] in order to address stability issues for a very broad class of problems. The DPG method automatically satisfies stability criteria by construction which enables DPG simulations to remain stable and convergent even in the pre-asymptotic regime. By nature, the DPG method also comes with a built-in error representation function, effectively eliminating the need for other a posteriori error estimators. Practically, this means that a simulation could start with just the coarsest mesh necessary to represent the geometry of the solution and adaptively refine toward a resolved solution in a very automatic way. Carried to its logical conclusion, this capability could significantly cut down on the time intensive manual mesh generation (and tweaking) that dominates a good amount of simulation and analysis time. Where a current numerical method might falter on a poorly designed mesh, necessitating an engineer to manually enter the problem and fix the offending mesh nodes, a DPG simulation would converge on the poor mesh, mark the offending cells, refine, and continue toward a solution.

Another benefit to the enhanced stability properties of DPG is the ability to consider high order and *hp*-adaptive methods. Many popular numerical methods for CFD (such as the discontinuous Galerkin method) are stable for low polynomial orders, but require additional stabilizing terms for higher orders. Additionally, one of the longstanding issues with *hp*-adaptive techniques was that they suffered stability problems when the polynomial order rose to high. Polynomial order presents no issue at all to DPG methods – allowing us to recover the high order convergence rates of high uniform $p$ methods or even the exponential convergence rates of $hp$ methods.

The biggest limitation to past explorations of the DPG method is that they were all limited to steady state problems. Obviously, this seriously limits the variety of interesting problems we could consider. 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 CFL condition was not binding since we were interested in implicit time integration schemes, but the CFL condition 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 order 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.

### 1.1.1 Investigating a new methodology

Much of science is driven by curiosity, and this especially holds for computational science. There is inherent value in exploring new methodologies because they may hold the keys to solving new problems or old problems in a better way. A new method may also help us to better understand existing methods. The variational multiscale approach to finite element analysis helped to illucidate on some of the success of the much older streamwise upwind Petrov-Galerkin method while generalizing and improving it. The DPG method itself can be viewed as a generalization of least-squares finite elements or even of mixed methods.

Curiousity similarly motivates the desire to explore a space-time DPG formulation for

computational fluid dynamics. Based on our past experience with steady DPG, we anticipate space-time DPG to be a very interesting technique that could extend the automaticity of DPG in very novel ways.

## 1.2 Literature review

### 1.2.1 Computational fluid dynamics

#### 1.2.1.1 Finite difference and finite volume methods

#### 1.2.1.2 Stabilized finite element methods

**SUPG**

**VMS**

**DG**

**HDG**

### 1.2.2 Space-time finite elements

Oden (first to propose), Bob Haber, Tayfun Tezduyar, Neumüller [2] [4]

### 1.2.3 DPG

General ideas 1-2 pages

## 1.3 Goal

# Chapter 2

# Space-time DPG

We summarize some completed work on space-time DPG. At the time of writing, Camellia does not officially support space-time computations, but we can fake it for 1D spatial problems by pretending the $y$-direction is time. Complications arise when the PDE under consideration is parabolic (i.e. contains second derivatives in space, but only first derivatives in time). Mathematically, this leaves traces undefined on element edges without a spatial normal component. Practically, this means that I had to hack the Camellia code in order to support these "spatial traces". We say that the code was "hacked" to indicate that we modified the code in an "ugly" manner in order to obtain the following results, but plan is to do this according to better software practices in the proposed work, since the current implementation is not very maintainable.

All of the following problems (unless otherwise noted) are run with the graph norm which is simply defined from the adjoint of the system supplemented with $L^2$ terms to upgrade it to a full norm.

## 2.1 Heat equation

The simplest space-time problem we can consider where the the spatial and temporal dimensions are treated differently is the heat equation. We start with a general $n$-dimensional spatial derivation and later simplify to spatially 1D with a few numerical experiments.

### 2.1.1 Derivation

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

$$\frac{\partial u}{\partial t} - \mu \Delta u = f\,, \quad \boldsymbol{x} \in \Omega\,,\ t \in (t_0, T) \tag{2.1}$$

where $u$ is unknown heat, $\epsilon$ 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:

$$\begin{aligned} \boldsymbol{\sigma} - \epsilon \nabla u &= 0 \\ \frac{\partial u}{\partial t} - \nabla \cdot \boldsymbol{\sigma} &= f\,, \end{aligned} \tag{2.2}$$

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

$$\begin{aligned} \frac{1}{\epsilon}\boldsymbol{\sigma} - \nabla u &= 0 \\ \nabla_{xt} \cdot \begin{pmatrix} -\boldsymbol{\sigma} \\ u \end{pmatrix} &= f\,. \end{aligned} \tag{2.3}$$

We now proceed with the standard DPG practice and multiply by test functions $\boldsymbol{\tau}$ and $v$ and integrate by parts over each space-time element $K$:

$$\begin{aligned} \left(\frac{1}{\epsilon}\boldsymbol{\sigma}, \boldsymbol{\tau}\right) + (u, \nabla \cdot \boldsymbol{\tau}) - \langle \hat{u}, \boldsymbol{\tau} \cdot \boldsymbol{n}_x \rangle &= 0 \\ -\left(\begin{pmatrix} -\boldsymbol{\sigma} \\ u \end{pmatrix}, \nabla_{xt} v\right) + \langle \hat{t}, v \rangle &= f\,, \end{aligned} \tag{2.4}$$

where

$$\hat{u} := \mathrm{tr}(u)$$

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

are new unknowns that live on the mesh skeleton introduced by the integration by parts. 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 2.1.

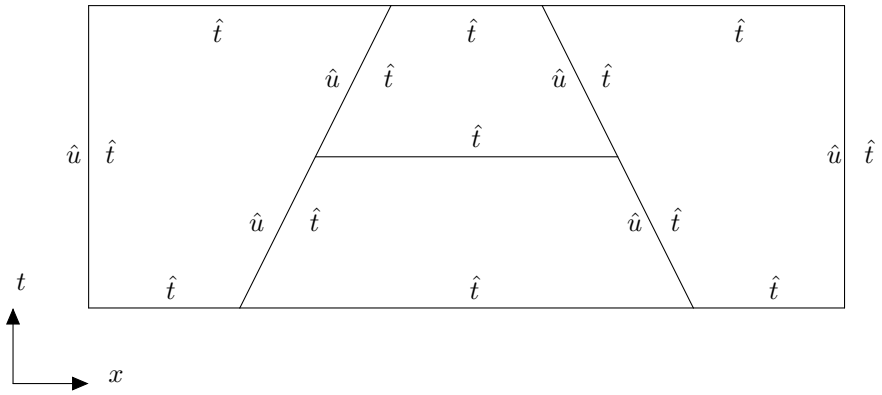



Figure 2.1: Support of flux and spatial trace variables

### 2.1.2 Problems considered

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 2.2.

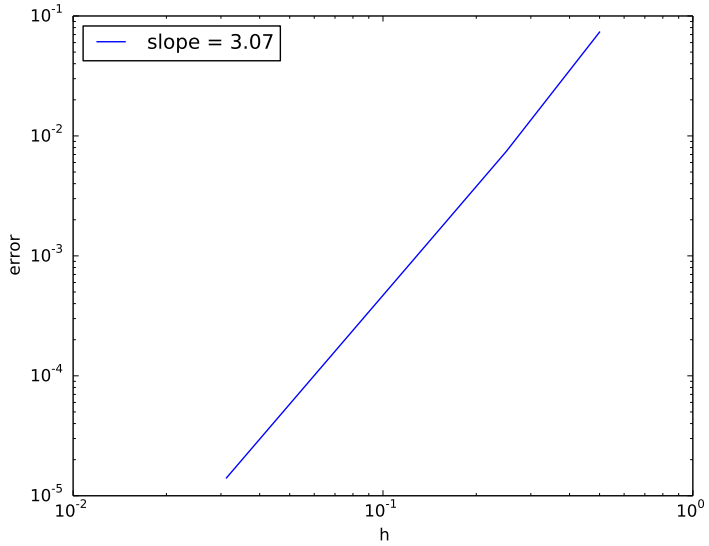


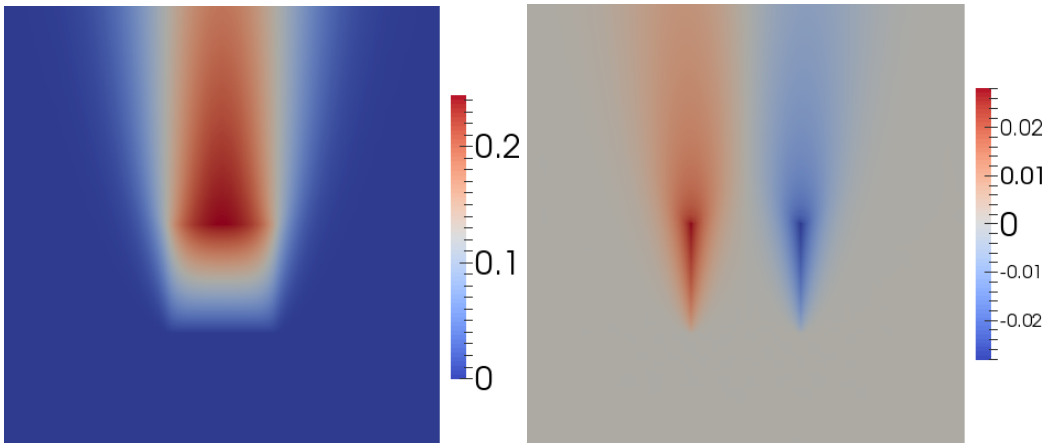

Figure 2.2: $L^2$ convergence of $u$ for the space-time heat equation

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 2.3. Notice that $\hat{u}$ in Figure 2.3c only lives on vertical edges as was discussed earlier. Also notice that the full mesh shown in Figure 2.3d automatically adapts spatially and temporally to where features are rapidly changing.

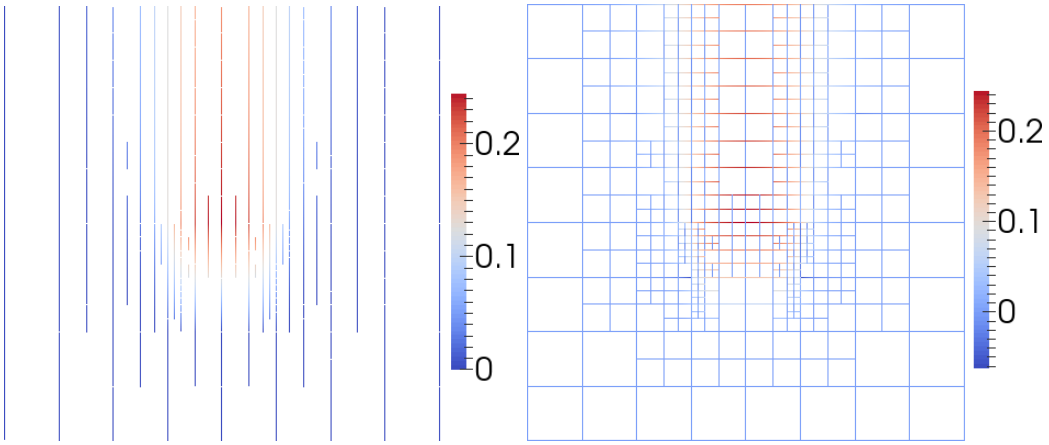
## 2.2 Convection-Diffusion

Transient convection-diffusion is identical to the heat equation with the addition of a convective term:

$$\frac{\partial u}{\partial t} + \nabla \cdot (\boldsymbol{\beta} u) - \epsilon \Delta u = f \, .$$




(a) $u$ (b) $\sigma$




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

Figure 2.3: Pulsed space-time heat problem after 4 refinements

The $d$-dimensional transient convection-diffusion equations could be viewed as a $d+1$ steady convection-diffusion problem with zero diffusion in the time direction.

### 2.2.1 Derivation

As a first order system in space-time, this is

$$\begin{aligned} \frac{1}{\epsilon}\boldsymbol{\sigma} - \nabla u &= 0 \\ \nabla_{xt} \cdot \begin{pmatrix} \boldsymbol{\beta} u - \boldsymbol{\sigma} \\ u \end{pmatrix} &= f \,. \end{aligned} \tag{2.5}$$

Multiplying (2.5) by test functions, and integrating by parts over each element, we obtain the following bilinear form:

$$\begin{aligned} \left(\frac{1}{\epsilon}\boldsymbol{\sigma}, \boldsymbol{\tau}\right) + (u, \nabla \cdot \boldsymbol{\tau}) - \langle \hat{u}, \tau_n \rangle &= 0 \\ -\left(\begin{pmatrix} \boldsymbol{\beta} u - \boldsymbol{\sigma} \\ u \end{pmatrix}, \nabla_{xt} v\right) + \langle \hat{t}, v \rangle &= f \,, \end{aligned} \tag{2.6}$$

where now $\hat{t} = \mathrm{tr}\,(\boldsymbol{\beta} u - \boldsymbol{\sigma}) \cdot \boldsymbol{n}_x + \mathrm{tr}(u) \cdot n_t$, and $\hat{u}$ is as before. Our test functions, $\boldsymbol{\tau}$ and $v$ live in the same spaces as for the heat equation.

### 2.2.2 Problems considered

Since space-time convection-diffusion is identical the heat equation with the addition of a convective term, we only pursue one numerical experiment to demonstrate that everything works as expected. This problem is inspired by the previous heat problem with the spatial domain extended to prevent the convected heat from impinging on the right wall. It might be interesting to impose a zero boundary condition on $\hat{u}$ and watch a boundary layer build up on the right wall, but instead we enforce a zero flux condition and content ourselves with the inner layer that forms around the source pulse. This is an arbitrary requirement necessitated by the "hackish" nature of this code. We haven't taken the time to allow enforcement of Dirichlet boundary conditions on spatial fluxes,

and since the code in this state was intended to be short-lived, it doesn't make sense to invest too heavily in adding features. Thus we enforce zero flux conditions on both walls as before. For this problem, the domain extends from $[0, 1.5] \times [0, 1]$ with the pulse occurring at $[0.25, 0.5] \times [0.25, 0.5]$.

## 2.3 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. The compressible Navier-Stokes equations are

$$\frac{\partial}{\partial t}\begin{bmatrix} \rho \\ \rho \boldsymbol{u} \\ \rho e_0 \end{bmatrix} + \nabla \cdot \begin{bmatrix} \rho \boldsymbol{u} \\ \rho \boldsymbol{u} \otimes \boldsymbol{u} + p\boldsymbol{I} - \mathbb{D} \\ \rho \boldsymbol{u} e_0 + \boldsymbol{u} p + \boldsymbol{q} - \boldsymbol{u} \cdot \mathbb{D} \end{bmatrix} = \begin{bmatrix} f_c \\ \boldsymbol{f}_m \\ f_e \end{bmatrix}, \tag{2.7}$$

where $\rho$ 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 $\lambda = -\frac{2}{3}\mu$,

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

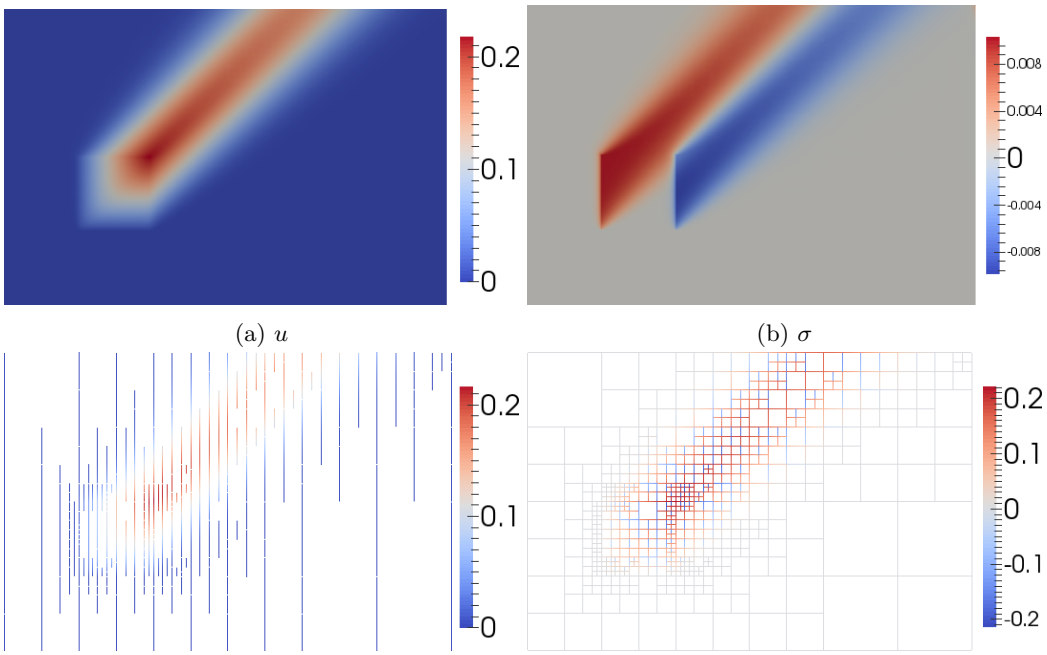
where $\boldsymbol{S}^*$ is the trace-less viscous strain rate tensor. The heat flux is given by Fourier's law:

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

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 $\gamma$ 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 $\gamma$, $C_p$, $C_v$, and $R$ are constant properties of the




(a) $u$ (b) $\sigma$

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

Figure 2.4: Space-time convection-diffusion problem after 4 refinements

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:

$$\frac{1}{\mu}\mathbb{D} - \left(\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^T\right) + \frac{2}{3}\nabla\cdot\boldsymbol{u}\boldsymbol{I} = 0 \tag{2.8a}$$

$$\frac{Pr}{C_p\mu}\boldsymbol{q} + \nabla T = 0 \tag{2.8b}$$

$$\nabla_{xt}\cdot\begin{pmatrix}\rho\boldsymbol{u}\\ \rho\end{pmatrix} = f_c \tag{2.8c}$$

$$\nabla_{xt}\cdot\begin{pmatrix}\rho\boldsymbol{u}\otimes\boldsymbol{u} + \rho RT\boldsymbol{I} - \mathbb{D}\\ \rho\boldsymbol{u}\end{pmatrix} = \boldsymbol{f}_m \tag{2.8d}$$

$$\nabla_{xt}\cdot\begin{pmatrix}\rho\boldsymbol{u}\left(C_vT + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right) + \boldsymbol{u}\rho RT + \boldsymbol{q} - \boldsymbol{u}\cdot\mathbb{D}\\ \rho\left(C_vT + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right)\end{pmatrix} = f_e\,, \tag{2.8e}$$

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

### 2.3.1 Derivation of Space-Time DPG Formulation

We start with (2.8) 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 $K$:

$$\left(\frac{1}{\mu}\mathbb{D},\mathbb{S}\right) + (2\boldsymbol{u},\nabla\cdot\mathbb{S}) - \left(\frac{2}{3}\boldsymbol{u},\nabla\operatorname{tr}\mathbb{S}\right) - \left\langle\frac{4}{3}\hat{\boldsymbol{u}},\mathbb{S}\boldsymbol{n}_x\right\rangle = 0 \tag{2.9a}$$

$$\left(\frac{Pr}{C_p\mu}\boldsymbol{q},\boldsymbol{\tau}\right) - (T,\nabla\cdot\boldsymbol{\tau}) + \left\langle\hat{T},\tau_n\right\rangle = 0 \tag{2.9b}$$

$$-\left(\begin{pmatrix}\rho\boldsymbol{u}\\ \rho\end{pmatrix},\nabla_{xt}v_c\right) + \left\langle\hat{t}_c,v_c\right\rangle = (f_c,v_c) \tag{2.9c}$$

$$-\left(\begin{pmatrix}\rho\boldsymbol{u}\otimes\boldsymbol{u} + \rho RT\boldsymbol{I} - \mathbb{D}\\ \rho\boldsymbol{u}\end{pmatrix},\nabla_{xt}\boldsymbol{v}_m\right) + \left\langle\hat{\boldsymbol{t}}_m,\boldsymbol{v}_m\right\rangle = (\boldsymbol{f}_m,\boldsymbol{v}_m) \tag{2.9d}$$

$$-\left(\begin{pmatrix}\rho\boldsymbol{u}\left(C_vT + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right) + \boldsymbol{u}\rho RT + \boldsymbol{q} - \boldsymbol{u}\cdot\mathbb{D}\\ \rho\left(C_vT + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right)\end{pmatrix},\nabla_{xt}v_e\right) + \left\langle\hat{t}_e,v_e\right\rangle = (f_e,v_e)\,, \tag{2.9e}$$

where

$$\hat{\boldsymbol{u}} = \operatorname{tr}(\boldsymbol{u})$$

$$\hat{T} = \operatorname{tr}(T)$$

$$\hat{t}_c = \operatorname{tr}(\rho \boldsymbol{u}) \cdot \boldsymbol{n}_x + \operatorname{tr}(\rho)\, n_t$$

$$\hat{\boldsymbol{t}}_m = \operatorname{tr}(\rho \boldsymbol{u} \otimes \boldsymbol{u} + \rho RT \boldsymbol{I} - \mathbb{D}) \cdot \boldsymbol{n}_x + \operatorname{tr}(\rho \boldsymbol{u})\, n_t$$

$$\hat{t}_e = \operatorname{tr}\left(\rho \boldsymbol{u}\left(C_v T + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right) + \boldsymbol{u}\rho RT + \boldsymbol{q} - \boldsymbol{u}\cdot\mathbb{D}\right)\cdot \boldsymbol{n}_x + \operatorname{tr}\left(\rho\left(C_v T + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}\right)\right) n_t\,.$$

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 weird requirements: each $d \times d$ components must be at least in $L^2(K)$, $\nabla \cdot \mathbb{S} \in \mathbf{L}^2(K)$, and $\nabla \operatorname{tr} \mathbb{S} \in \mathbf{L}^2(K)$. In practice, we will probably just seek each component in $H^1(K)$.

#### 2.3.1.1 Linearization

We follow a standard residual-Jacobian linearization procedure coupled with a Gauss-Newton solve. 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 (2.9) while $\Delta U$ contains the full linear terms and the updates to the nonlinear terms. Also, we drop the $\Delta$ and $\tilde{\cdot}$ notation for linear terms. Define residual $R(U)$ as the left hand side of (2.9) 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:

$$R'(\tilde{U})\Delta U = -R(\tilde{U})\,. \tag{2.10}$$

This is an instance of a Gauss-Newton nonlinear solve. We only need to define our Jacobian and residual for each component of (2.9). 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\,\mathrm{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) \\
&+ \left\langle\hat{t}_e,v_e\right\rangle\,.
\end{aligned} \tag{2.11}$$

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

$$\begin{aligned}
&\left(\frac{1}{\mu}\tilde{\mathbb{D}},\mathbb{S}\right) + (2\tilde{\boldsymbol{u}},\nabla\cdot\mathbb{S}) - \left(\frac{2}{3}\tilde{\boldsymbol{u}},\nabla\,\mathrm{tr}\,\mathbb{S}\right) \\
&- \left(\tilde{T},\nabla\cdot\boldsymbol{\tau}\right) \\
&- \left(\begin{pmatrix}\tilde{\rho}\tilde{\boldsymbol{u}}\\ \tilde{\rho}\end{pmatrix},\nabla_{xt}v_c\right) - (f_c,v_c) \\
&- \left(\begin{pmatrix}\tilde{\rho}\tilde{\boldsymbol{u}}\otimes\tilde{\boldsymbol{u}}+\tilde{\rho}R\tilde{T}\boldsymbol{I}-\tilde{\mathbb{D}}\\ \tilde{\rho}\tilde{\boldsymbol{u}}\end{pmatrix},\nabla_{xt}\boldsymbol{v}_m\right) - (\boldsymbol{f}_m,\boldsymbol{v}_m) \\
&- \left(\begin{pmatrix}\tilde{\rho}\tilde{\boldsymbol{u}}\left(C_v\tilde{T}+\frac{1}{2}\tilde{\boldsymbol{u}}\cdot\tilde{\boldsymbol{u}}\right)+\tilde{\boldsymbol{u}}\tilde{\rho}R\tilde{T}-\tilde{\boldsymbol{u}}\cdot\tilde{\mathbb{D}}\\ \tilde{\rho}\left(C_v\tilde{T}+\frac{1}{2}\tilde{\boldsymbol{u}}\cdot\tilde{\boldsymbol{u}}\right)\end{pmatrix},\nabla_{xt}v_e\right) - (f_e,v_e)\,.
\end{aligned} \tag{2.12}$$

#### 2.3.1.2 Test Norm

The most obvious first choice for test norm in the local solve is the graph norm, which comes from the problem adjoint. We start by grouping terms in (2.11) 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\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}} - \tilde{\rho}\frac{\partial\boldsymbol{v}_m}{\partial t} - C_v\tilde{\rho}\tilde{T}\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}} - R\tilde{\rho}\tilde{T}_\nabla v_e + \tilde{\mathbb{D}}\cdot\nabla v_e - \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{2.13}
$$

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}} - \tilde{\rho}\frac{\partial \boldsymbol{v}_m}{\partial t} - C_v \tilde{\rho}\tilde{T}\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}} - R\tilde{\rho}\tilde{T}\nabla v_e + \tilde{\mathbb{D}} \cdot \nabla v_e - \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\| \\
&+ \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\| + \alpha_f \left\| \boldsymbol{\tau} \right\| ,
\end{aligned}
\tag{2.14}
$$

where $\alpha_c$, $\alpha_m$, $\alpha_e$, $\alpha_s$, and $\alpha_f$ are scaling constants, usually one.

Unfortunately, the graph norm is known to not be robust for steady convection-diffusion or Navier-Stokes, and we saw that non-robustness manifest 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 has not been done (this is part of the proposed work), so we are on shakier footing developing a robust norm for transient Navier-Stokes. 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. Indeed, this selection proved to be more robust for the test problems considered in the next section. This

modified graph norm is then:

$$
\begin{aligned}
&\left\|\nabla \boldsymbol{v}_m + \nabla v_e \otimes \tilde{\boldsymbol{u}}\right\|^2 \\
&+ \left\|-\nabla v_e\right\|^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 \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}} - R\tilde{\rho}\tilde{T}\nabla v_e + \tilde{\mathbb{D}} \cdot \nabla v_e - \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\| \\
&+ \frac{1}{\mu}\left\|\mathbb{S}\right\| + 2\left\|\nabla \cdot \mathbb{S} - \frac{1}{3}\nabla \operatorname{tr}\mathbb{S}\right\| + \frac{Pr}{c_p\mu}\boldsymbol{\tau}\left\|\boldsymbol{\tau}\right\| + \left\|\nabla \cdot \boldsymbol{\tau}\right\| \\
&+ \left\|v_c\right\|^2 + \left\|\boldsymbol{v}_m\right\|^2 + \left\|v_e\right\|^2 \ .
\end{aligned}
\tag{2.15}
$$

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.

### 2.3.2 Numerical Results

We consider two 1D test problems as verification. The Sod shock tube and Noh implosion both have analytical solutions derived 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 Sod and $\mu = 10^{-3}$ for Noh. Essentially we are just simulating low viscosity Navier-Stokes as a

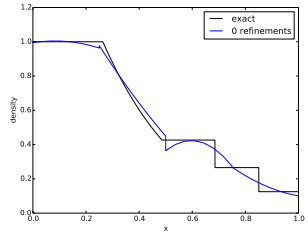
stand-in for the unsolvable pure Euler. We mentioned previously that the test norm we are using is not entirely robust, and in fact these viscosity values were on the lower end of what we could simulate with this preliminary norm. Following the same polynomial representation as we did in the section on local conservation (Section **??**), the field variables were represented with quadratics.

#### 2.3.2.1 Sod Shock Tube

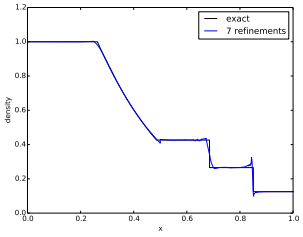
The Sod shock tube problem was developed by Gary Sod in 1978[6], and has proven to be a popular problem for verification of compressible Navier-Stokes and Euler solvers. It serves to verify that a numerical method can effectively handle a rarefaction wave, material discontinuity, and shock wave all in 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 $(\rho_R, p_R, u_R) = (0.125, 0.1, 0)$. 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.

#### 2.3.2.2 Noh Implosion

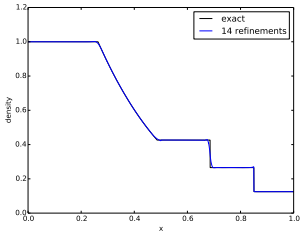
Noh[3] The longer time nature of this problem recommended the use of multiple time slabs rather than a single solve like the previous problem.



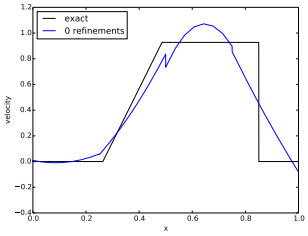

(a) Density on initial mesh



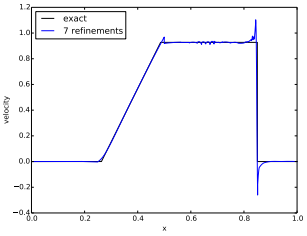

(b) Density after 7 refinements



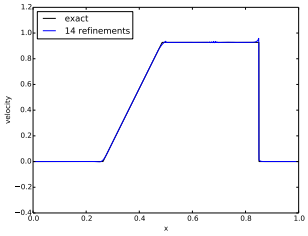

(c) Density after 14 refinements



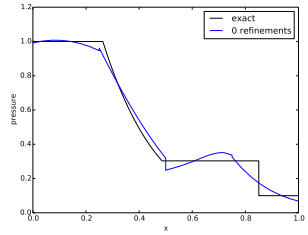

(d) Velocity on initial mesh



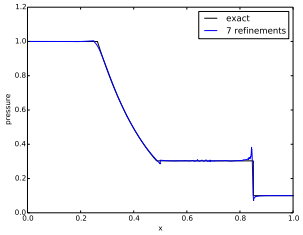

(e) Velocity after 7 refinements



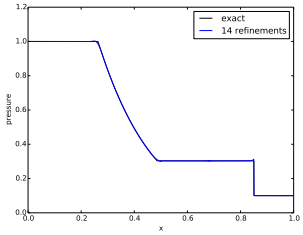

(f) Velocity after 14 refinements



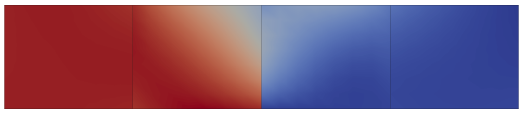

(g) Pressure on initial mesh



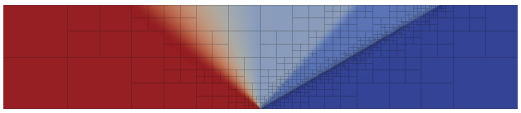

(h) Pressure after 7 refinements



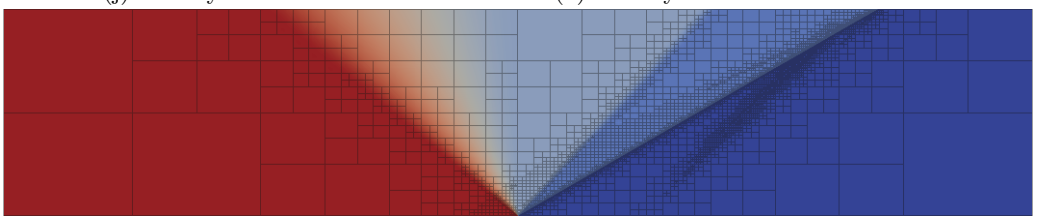

(i) Pressure after 14 refinements



(j) Density with initial mesh

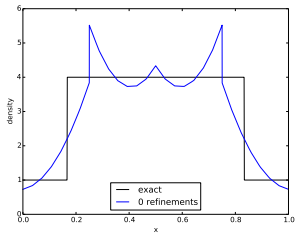


(k) Density with mesh after 7 refinements

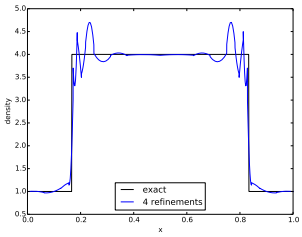


(l) Density with mesh after 14 refinements

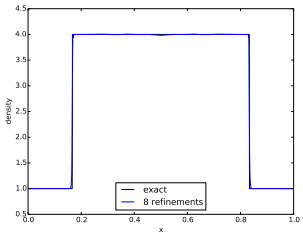
Figure 2.5: Sod problem with final time $t = 0.2$



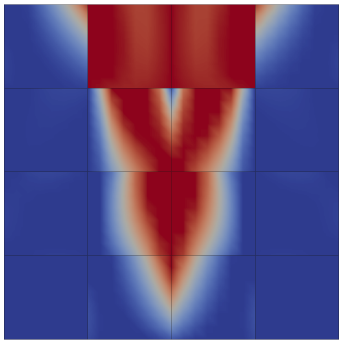

(a) Density on initial mesh



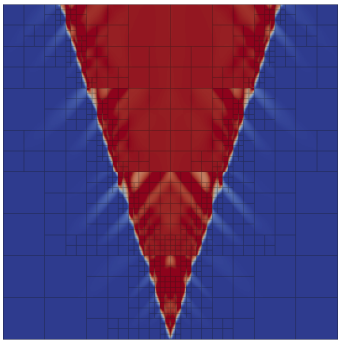

(b) Density after 4 refinements



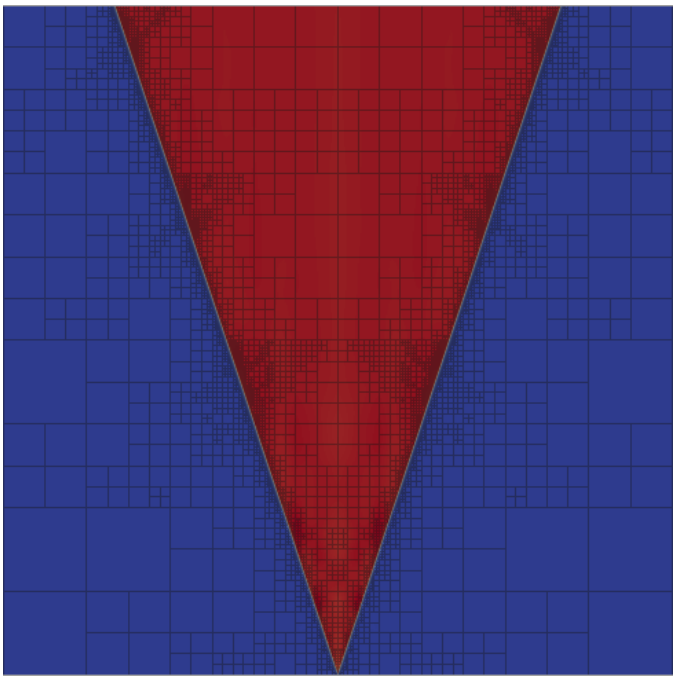

(c) Density after 8 refinements



(d) Density with initial mesh



(e) Density with mesh after 4 refinements



(f) Density with mesh after 8 refinements

Figure 2.6: Noh problem with final time $t = 1.0$

# Chapter 3

# Proposed work

**Area A: Applicable mathematics.** DPG is a method built on rigorous mathematical theory. While rooted in the standard theory of finite elements, DPG is novel enough that many of the old analysis tools do not directly apply. As a consequence, much of the early DPG literature has been laden with mathematical proofs of stability, convergence, and robustness. While developing the theory for a locally conservative DPG formulation we performed a stability and robustness analysis. Another robustness analysis will be necessary as we explore space-time convection-diffusion since the equation parabolic in space-time. Past DPG robustness analysis focused on purely elliptic problems.

One significant remaining obstacle to obtaining robust solutions for compressible Navier-Stokes is guaranteeing positivity of the density and energy in the presence of unresolved shocks. Positivity is an auxiliary stability condition that is not guaranteed by traditional stabilization techniques (including DPG), so it we are going to need to augment an additional technology to the method in order to guarantee physically realizable densities and energies. Unfortunately, positivity preservation is itself an incredibly difficult problem. We have several ideas we want to try in the context of DPG, but solving the issue may not lie within the scope of this dissertation.

**Area B: Numerical analysis and scientific computation.** I will support development of the DPG software framework *Camellia*[5] including running verification on several test problems. This will include the development and verification of spatially 2D space-time simulations. I will also develop a means to perform time stepping via space-time slabs which should reduce the overall run time and memory requirements for longer simulations. I've already contributed several auxiliary features to *Camellia* including mesh readers and solution export, but I also plan on looking into adding HDF5 support along with parallel solution output.

A space-time implementation adds additional dimensionality to the problem under consideration increasing the problem size and memory requirements. This makes parallel simulation an increasingly important aspect of this work. The 16 core *Nozomi* cluster is sufficient for early experimentation, but as we ramp up to larger problems, I plan to run on allocations at TACC and on Mira at Argonne National Laboratory.

The major bottleneck to parallel simulations at the moment is that *Camellia* still relies on a serial direct solver for the global solve. Time permitting, I would like to investigate the implementation of an effective iterative solver for DPG.

**Area C: Mathematical modeling and applications.** As we are exploring a method that is still very much in development, I will be running a number of standard test problems. I've already explored the conservation properties of DPG through a couple of simulations of Stokes flow around a cylinder and over a backward facing step.

The steady state DPG formulation failed to converge on the Carter plate problem for Reynolds numbers of $10^7$. One hypothesis is that this was due to transient effects on the unresolved mesh which pushed the time step requirements very low. The hope is that by using a space-time formulation with temporal adaptivity, we would be able to resolve these temporal oscillations and they would die away to the correct steady state solution. Another hypothesis is that the non-convergence was caused by Gibbs phenomenon near the unresolved leading edge driving the density negative. Our line search algorithm scales the Newton updates to prevent negative densities, but sometimes the lines search factor can be driven so small as to effectively stall convergence. Therefore my first priority will be to investigate which factor is stalling convergence and attempt to solve it either via adaptive space-time refinements or some sort of positivity preserving technology yet to be decided upon. If this proves to be an effective way of handling transient shock problems, I would also like to simulate the Sedov explosion problem and the Noh implosion problem. Time permitting, I would also like to run the triple point shock interaction problem and a Rayleigh-Taylor instability problem. These will be ideal for demonstrating the local adaptivity of DPG.

That said, I believe that the most promising application for space-time DPG will be the incompressible Navier-Stokes equations. Here we don't run into any of the same issues with Gibbs phenomenon and we can use vanilla space-time DPG. The Taylor-Green vortex problem is an obvious choice since it has a exact solution that we can compare to. There are several vortex shedding problems that would be of interest including flow over a triangle, square, and cylinder.

# Bibliography


[1] L. Demkowicz and J. Gopalakrishnan. A class of discontinuous Petrov-Galerkin methods. Part I: The transport equation. *Comput. Methods Appl. Mech. Engrg.*, 2009.

[2] C. M. Klaij, J. J. W. van der Vegt, and H. van der Ven. Space-time discontinuous galerkin method for the compressible navier-stokes equations. *J. Comput. Phys.*, 217(2):589–611, September 2006.

[3] W.F Noh. Errors for calculations of strong shocks using an artificial viscosity and an artificial heat flux. *Journal of Computational Physics*, 72(1):78 – 120, 1987.

[4] Sander Rhebergen, Bernardo Cockburn, and Jaap J. W. Van Der Vegt. A space-time discontinuous Galerkin method for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 233:339–358, January 2013.

[5] N. V. Roberts, D. Ridzal, P. B. Bochev, and L. F. Demkowicz. A toolbox for a class of discontinuous Petrov-Galerkin methods using Trilinos. Technical Report SAND2011-6678, Sandia National Laboratories.

[6] Gary A Sod. A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws. *Journal of Computational Physics*, 27(1):1 – 31, 1978.