A Space-Time Discontinuous Galerkin Spectral Element Method for Nonlinear Hyperbolic Problems

Chaoxu Pei, Mark Sussman, M. Yousuff Hussaini
Department of Mathematics, Florida State University, Tallahassee, FL, 32306, USA.

Abstract

A space-time discontinuous Galerkin (DG) spectral element method is presented for the solution to nonlinear hyperbolic problems. The space-time method is combined with two different approaches for treating problems with discontinuous solutions: (i) space-time dependent artificial viscosity is added in order to capture discontinuities, and (ii) the sharp discontinuity is tracked with space-time spectral accuracy, as it moves through the grid. To capture the discontinuity whose location is initially unknown, an artificial viscosity term is strategically introduced; the strategy due to Persson and Peraire [1] is extended to space-time spectral methods. The amount of artificial viscosity prescribed varies with time within a given space-time slab. It is found that spectral accuracy is recovered everywhere except in the "troublesome element(s)" where the unresolved steep/sharp gradient exists. When the location of a discontinuity is initially known, a space-time spectrally accurate tracking method has been developed so that the spectral accuracy of the position of the discontinuity, and the space-time spectral accuracy of the

---

Email addresses: `cpei@math.fsu.edu` (Chaoxu Pei), `sussman@math.fsu.edu` (Mark Sussman), `myh@math.fsu.edu` (M. Yousuff Hussaini)

solution on either side of the discontinuity is preserved. A Picard iteration method is developed in order to handle nonlinear terms. Within each Picard iterate, a linear equation is solved, which is derived from the space-time DG spectral element discretization. Spectral accuracy in both space and time is first demonstrated for the Burgers' equation. Space-time spectral accuracy enables better accuracy at extrema away from the shock, and enables better accuracy at capturing the shock strength at a shock with higher order polynomials in both space and time compared to lower order polynomials. Simultaneously, Gibbs oscillations are suppressed by the combination of higher order polynomials and the present shock capturing scheme, as well as by the combination of higher order polynomials and the present shock tracking scheme. Such features are demonstrated for both a scalar hyperbolic linear problem with a discontinuous initial condition and the inviscid Burgers' equation with a discontinuous initial condition as well as a smooth initial condition that evolves into a sharp discontinuity. The sensitivity of the number of Picard iterations to the temporal order is discussed. The spectral accuracy of the shock speed and location is demonstrated for the solution of the inviscid Burgers' equation obtained by the shock tracking method.

Keywords: Space-time, Discontinuous Galerkin, Spectral accuracy, Shock capturing, Shock tracking, Picard iteration

1. Introduction

In this paper, we present a numerical method for solving scaler conservation laws, which is spectrally accurate in both space and time in the region in which the solution is smooth. For treating problems with sharp discontinuities, we present two approaches: (i) if a sharp discontinuity develops as the solution evolves, then we strategically introduce artificial viscosity, which is similar to the ones proposed in Persson and Peraire [1], Barter and Darmofal [2], Reisner et al [3], Casoni et al [4], and (ii) if the location of a discontinuity is initially known, then we introduce a space-time spectrally accurate tracking method so that the spectral accuracy of the position of the discontinuity, and the space-time spectral accuracy of the solution on either side of the discontinuity is preserved.

We present here a brief history of numerical methods leading to the present method. In 1989, Shu and Osher [5] developed essentially non-oscillatory (ENO) numerical methods for hyperbolic conservation laws. These methods can be implemented up to any prescribed spatial and temporal order of accuracy. Unfortunately, regardless of the order, these methods are dissipative. Weighted ENO (WENO) schemes, first developed in 1996 by Jiang and Shu [6] and Balsara and Shu [7], are less dissipative than ENO methods. In 1999, Yee et al [8] proposed a numerical method that is a hybrid of the low-dissipative fourth order method (4th order in space and time) of Gustafsson and Olsson [9] with a shock capturing method developed by Harten [10]. Improvements upon the hybrid approach of Yee et al [8] have since followed. Ren et al [11] hybridized the compact scheme of Pirozzoli [12] with the WENO scheme of Jiang and Shu [6]. Sjogreen and Yee [13] also hybridized a low-dissipative fourth order method of Gustafsson and Olsson [9] with the multiresolution method developed by Harten [10]. The difference between Sjogreen and Yee's method [13] and the hybrid method of Yee et al [8], is that the smoothness sensor in this work did not depend on empirical tuning parameters. Instead, sensors were developed by looking at the coefficients of the wavelet transform Gerritsen and Olsson [14], Mallat and Zhong [15], Daubechies [16], and Mallat [17]. Reisner et al [3] developed a sensor that switched on or off an artificial viscosity term depending on the solutions of a linear scalar reaction-diffusion equation, which is referred to as C-method. With artificial viscosity deactivated, their method reduced to either a second order finite element method or a 5th order WENO method. Abbassi et al [18] proposed an entropy-based artificial viscosity for spectral collocation method. To smoothen artificial viscosity, an element-level filtering approach is introduced. With such a treatment, the resulting artificial viscosity is locally large near discontinuities.

Besides methods that switch between spatially dissipative and relatively less dissipative discretization schemes, methods have been developed for limiting the order of the discretization in time. Duraisamy et al [19] developed time limiters for the second order 2-stage diagonally implicit Runge Kutta (DIRK2) and the Trapezoid time discretization methods. They implemented the 5th order monotonicity preserving method of Suresh and Huynh [20] for the spatial discretization.

Concurrent with the development of shock capturing methods, there have been the development of methods for tracking discontinuities in hyperbolic conservation laws. We refer to the article by Touil et al [21] and the references therein; they resorted to refining the mesh in discontinuous regions which introduces an inordinate number of additional degrees of freedom.

In this work, we develop a space-time discontinuous Galerkin (DG) spectral element method. DG methods in space have been developed for solving a wide range of physical problems, especially for dealing with flows with discontinuities computed on unstructured meshes. Detailed surveys can be found in, for example, Cockburn et al [22], Canuto et al [23], Hesthaven et al [24], Kopriva [25], and Karniadakis and Sherwin [26]. Recently, space-time DG finite element methods have been explored by Sudirham et al [27], Klaij et al [28], van der Vegt and Sudirham [29], Sollie et al [30], and Rhebergen et al [31], for problems that require moving and deforming meshes. Space-time DG schemes result in implicit time integration and naturally allow for time adaptivity. van der Vegt and van der Ven [32] present a space-time DG finite element method for Euler equations of gas dynamics in time-dependent flow domains, which involves adding artificial dissipation to guarantees monotone solutions. Lörcher et al [33] propose an explicit DG scheme for inviscid compressible flow in one space dimension, in which a Taylor expansion in space and time to define a space-time polynomial is used to obtain an arbitrary order of accuracy in space and time. Their approach require a local stability criterion to be satisfied in every grid zone.

Motivated by the properties of space-time DG finite element methods, we present a space-time DG spectral element method for solving scaler conservation laws, which is integrated with shock capturing and shock tracking procedures. If the position of a discontinuity is initially unknown, then we employ a sensor (developed by Persson and Peraire [1]) to locate, what we call,, the troublesome element wherein the unresolved steep gradient/shock causes Gibbs phenomenon. The shock is contained within one spatial element. Since the proposed method is in a space-time framework, the amount of artificial viscosity is computed at each time "node" independently within one space-time slab, which allows one to take a large time step with optimal diffusion to capture strong shock without Gibbs phenomenon. By increasing the polynomial order in both space and time, one can resolve the shock strength, i.e., the extrema on either side of the shock. If the initial position of a discontinuity is known, or if the solution is smooth everywhere, then the present method is spectrally accurate in both space and time. We note that the present method does not align the computational elements with a discontinuity, which distinguishes it from a body fitted method [27, 29, 34]. Instead, two global transformations: a backward transformation and a forward transformation are applied, which combines an Eulerian description, i.e., a fixed frame of reference, with a Lagrangian description, i.e., a moving frame of reference (see [35]). In the proposed tracking method, the discontinuity may cut an element in such a way that one piece can be very small. Small cut cells do not affect the spectral accuracy of the proposed tracking method. In order to deal with nonlinear terms, we employ a Picard iteration procedure (see e.g., [36, 34]). At each iteration, a linear equation is solved by a space-time DG spectral element method.

The organization of this paper is as follows. In section 2, we define the nonlinear initial boundary-value problem to be solved by a space-time discontinuous Galerkin spectral element method, which is spectrally accurate in both space and time. The space-time ("slab") discretization framework is introduced in section 3. In section 4, the DG spectral element method is presented in the space-time framework. Two treatments for shocks are then presented in section 5. The outline of the proposed method is given in section 6. Numerical experiments are presented in one spatial dimension in section 7 in order to evaluate the accuracy and efficiency of the present shock treatment methods. The summary of results and conclusions are provided in the final section 8.

2. Problem formulation

On a Cartesian domain, $\Omega \in \mathbb{R}^d$, we consider a scalar conservation law of the form,

$$u_t + \nabla \cdot \boldsymbol{F}(u) = 0 \quad (\boldsymbol{x}, t) \in \Omega \times [0, T], \tag{1}$$

$$u(\boldsymbol{x}, t) = g_b(\boldsymbol{x}, t), \quad (\boldsymbol{x}, t) \in \partial\Omega \times [0, T], \tag{2}$$

$$u(\boldsymbol{x}, 0) = u_0, \quad \boldsymbol{x} \in \Omega, \tag{3}$$

where $\boldsymbol{F}(u)$ is the vector of fluxes in the spatial directions, u_0 is the initial data and $g_b(\boldsymbol{x}, t)$ denotes the boundary data.

It is well known that such problems may develop discontinuous solutions even if the initial values are smooth. In order to suppress the Gibbs-type oscillations caused by applying higher order numerical methods to discontinuous solutions, one can either introduce a dissipative term (e.g.,[1]) or track the discontinuities explicitly (e.g., [21]) when the initial location of a discontinuity is known. In the former case, we rewrite (1) as,

$$u_t + \nabla \cdot \boldsymbol{F}(u) = \nabla \cdot (\boldsymbol{\epsilon}(u) \nabla u) \quad (\boldsymbol{x}, t) \in \Omega \times [0, T], \tag{4}$$

where $\boldsymbol{\epsilon}(u)$ is a matrix with non-negative components depending on the solution u.

3. Space-time discretization

3.1. Space-time tessellation

In this section, we follow the notation of, e.g. [27, 29, 34], to introduce the definitions of the space-time domain, space-time slabs and space-time elements.

In a space-time discretization, we introduce a *space-time domain* $\mathcal{E}$ by considering the domain $\mathcal{E} = \Omega \times [0, T]$ in $\mathbb{R}^{d+1}$, where d denotes the spatial dimension. The coordinates of a point $\bar{\boldsymbol{x}} \in \mathcal{E}$ are defined as $\bar{\boldsymbol{x}} = (\boldsymbol{x}, x_{d+1})$ with the spatial variables $\boldsymbol{x} = (x_1, x_2, ..., x_d)$ and the time variable $t = x_{d+1}$. To tessellate the space-time domain $\mathcal{E}$, we first divide $\mathcal{E}$ into $E^{(t)}$ *space-time slabs* by the time levels $0 = t_0 < t_1 < ... < t_{E^{(t)}} = T$. The n-th space-time slab is then denoted as $\mathcal{E}^n = \mathcal{E} \cap I_n$, where $I_n = [t_{n-1}, t_n]$ is the n-th time interval with length $\Delta t_n = t_n - t_{n-1}$. Now, we describe the construction of *space-time elements* in the space-time slab $\mathcal{E}^n$. Divide the spatial domain $\Omega_{t_{n-1}}$ ($\Omega_{t_{n-1}} := \{(\boldsymbol{x}, t) \in \Omega \times t_{n-1}\}$) into $E^{(\boldsymbol{x})}$ non-overlapping spatial elements K^{n-1} with a uniformed size h and similarly for the domain Ω_{t_n} with $E^{(\boldsymbol{x})}$ spatial elements K^n. A space-time element $\mathcal{K}^n$ is then obtained by connecting K^{n-1} and K^n via linear interpolation. The outward unit space-time normal on $\partial\mathcal{K}$ is denoted by $\bar{\boldsymbol{n}}_{\mathcal{K}} = (\boldsymbol{n}, n_{d+1})$, where $\boldsymbol{n} = (n_1, n_2, ..., n_d)$ is the spatial part of the space-time normal while n_{d+1} is the time component.

The tessellation of the space-time slab $\mathcal{E}^n$ is defined as $\mathcal{T}_h^n$, which consists of all space-time elements, $\mathcal{K}^n$, defined in $\mathcal{E}^n$. Then the tessellation of the space-time domain $\mathcal{E}$ is denoted as $\mathcal{T}_h = \cup_n \mathcal{T}_h^n$. A sketch of a space-time domain, $\mathcal{E} \in \mathbb{R}^2$, with a tessellation $\mathcal{T}_h^n$ is demonstrated in Fig. 1.

Remark 3.1. *In the shock capturing method, the spatial domain is fixed*

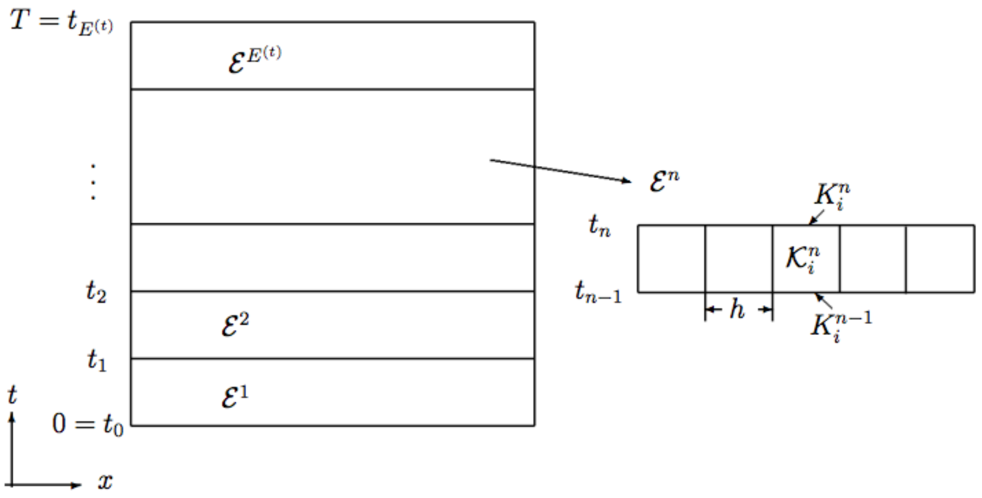

Figure 1: An Example of a space-time domain, $\mathcal{E} = \Omega \times [0, T] \in \mathbb{R}^2$, with the coordinates of a point $\bar{\boldsymbol{x}}$ denoted by $\bar{\boldsymbol{x}} = (x_1, x_2)$, where $x_1 = x$ is the spatial variable and x_2 denotes the time variable t. The space-time domain $\mathcal{E}$ is divided into $E^{(t)}$ *space-time slabs* by the time levels $0 = t_0 < t_1 < ... < t_{E^{(t)}} = T$. On the right, the rectangular mesh is an example of a tessellation $\mathcal{T}_h^n$ of the space-time slab $\mathcal{E}^n$, where $\mathcal{E}^n = \mathcal{E} \cap I_n$, where $I_n = [t_{n-1}, t_n]$. $\mathcal{K}_i^n$ denotes a space-time element in $\mathcal{E}^n$.

in time, which is different from the one proposed in [27, 29, 34]. We then obtain a rectangular space-time mesh in each space-time slab for one spatial dimension or a cubic space-time mesh for two spatial dimensions.

Remark 3.2. *In the shock tracking method, we introduce a moving frame of reference, in which the spatial domain is fixed in time. In the moving frame of reference, the mesh is a rectangular space-time mesh in each space-time slab for one spatial dimension or a cubic space-time mesh for two spatial dimensions. However, A cut cell is introduced in such mesh. (See [35])*

3.2. Function spaces and notation

First, we define a mapping, $\bar{\boldsymbol{x}} = \boldsymbol{G}_{|\mathcal{K}_j^n}(\boldsymbol{\xi}) \in \mathbb{R}^{d+1}$, which connects a space-time element $\mathcal{K}_j^n$ and the master element $\mathcal{R} = [-1, 1]^{d+1}$, where $\bar{\boldsymbol{x}} = (\boldsymbol{x}, t) \in \mathcal{K}_j^n$ and $\boldsymbol{\xi} = (\xi_1, \xi_2, ..., \xi_{d+1})$. Take two spatial dimension, i.e., d=2, as an example, we have,

$$\boldsymbol{G}_{|\mathcal{K}_j^n} : \mathcal{R} \to \mathcal{K}_j^n : \boldsymbol{\xi} \to \bar{\boldsymbol{x}} = \sum_{i=1}^{8} \bar{\boldsymbol{x}}^i(\mathcal{K}_j^n)\chi_i(\boldsymbol{\xi}) \tag{5}$$

where $\{\bar{\boldsymbol{x}}^i(\mathcal{K}_j^n)\}_{i=1,...,8}$ is the vertices of the space-time element $\mathcal{K}_j^n$, and $\chi_i(\boldsymbol{\xi})$ is the standard trilinear finite element shape functions for hexahedra. Examples of the transformation $\bar{\boldsymbol{x}} = \boldsymbol{G}(\boldsymbol{\xi}) \in \mathbb{R}^2$ can be found in [23, 25].

Next, Let's introduce two function spaces, Q_h and $\boldsymbol{V}_h$, associated with the tessellation $\mathcal{T}_h$,

$$V_h = \{\nu \in (L^2(\mathcal{E})) : \nu|_{\mathcal{K}} \circ \boldsymbol{G}_{\mathcal{K}} \in \mathcal{P}^{\boldsymbol{p}}(\mathcal{R}), \forall \mathcal{K} \in \mathcal{T}_h\}, \tag{6}$$

$$\boldsymbol{\Sigma}_h = \{\boldsymbol{\tau} \in L^2(\mathcal{E})^d : \boldsymbol{\tau}|_{\mathcal{K}} \circ \boldsymbol{G}_{\mathcal{K}} \in (\mathcal{P}^{\boldsymbol{p}}(\mathcal{R}))^d, \forall \mathcal{K} \in \mathcal{T}_h\}, \tag{7}$$

where $\mathcal{P}^{\boldsymbol{p}}(\mathcal{R})$ is the set of all polynomials of degree at most $\boldsymbol{p} = (p^{(\bar{\boldsymbol{x}})}) = (p^{(\boldsymbol{x})}, p^{(t)}) \in \mathbb{R}^{d+1}$ on $\mathcal{R}$, with $p^{(\boldsymbol{x})}$ in the spatial direction and $p^{(t)}$ in the time direction.

Spectral element methods with non-periodic boundaries use orthogonal polynomial approximations of the solution [23, 25]. In the space-time domain $\mathcal{E} \in \mathbb{R}^{d+1}$, we introduce a tensor product basis, $\psi(\boldsymbol{\xi}) \in \mathcal{P}^{\boldsymbol{p}}(\mathcal{R})$, such that

$$\psi(\boldsymbol{\xi}) = \prod_{i=1}^{d+1} \ell(\boldsymbol{\xi}_i) \tag{8}$$

where

$$\ell(\boldsymbol{\xi}_i) = \ell_j(\xi_i) = \prod_{\substack{n=0 \\ n\neq j}}^{p^{(x_i)}} \frac{\xi_i - \xi_{i,n}}{\xi_{i,j} - \xi_{i,n}}, \quad 1 \leqslant i \leqslant d+1 \tag{9}$$

where $\{\xi_{i,j}\}_{j=0,\dots,p^{(x_i)}}$ is the set of interpolation points in ξ_i direction of the master element $\mathcal{R}$. We choose Legendre Gauss nodes in the spatial directions and Legendre Gauss-Lobatto nodes in the time direction. In addition, we define the trace of $\nu \in V_h$ on $\mathcal{K}$ as

$$\nu^- = \lim_{\epsilon \to 0} \nu(\bar{\boldsymbol{x}} - \epsilon \bar{\boldsymbol{n}}_{\mathcal{K}}) \tag{10}$$

where $\bar{\boldsymbol{x}} = (\boldsymbol{x}, t) \in \partial\mathcal{K}$ and $\bar{\boldsymbol{n}}_{\mathcal{K}}$ is outward unit normal of $\mathcal{K}$. The trace of $\boldsymbol{\tau} \in \boldsymbol{\Sigma}_h$ is defined similarly.

Due to the discontinuous approximation spaces, the approximations are double valued across element boundaries. Thus, we introduce the average $\{\{\cdot\}\}$ and jump $[\![\cdot]\!]$ operators. Considering two adjacent elements $\mathcal{K}_j^n$ and $\mathcal{K}_{j+1}^n$ such that $\mathcal{S}_{j,j+1}^n = \mathcal{K}_j^n \cap \mathcal{K}_{j+1}^n$, let $\bar{\boldsymbol{n}}_{\mathcal{K}_j^n}$ and $\bar{\boldsymbol{n}}_{\mathcal{K}_{j+1}^n}$ denote the corresponding outward unit normal of $\mathcal{K}_j^n$ and $\mathcal{K}_{j+1}^n$ on $\mathcal{S}_{j,j+1}^n$. Then the average $\{\{\cdot\}\}$ and jump $[\![\cdot]\!]$ operators are defined as,

$$\{\{\nu\}\} = (\nu^-_{\mathcal{K}_j^n} + \nu^-_{\mathcal{K}_{j+1}^n})/2, \quad \{\{\boldsymbol{\tau}\}\} = (\boldsymbol{\tau}^-_{\mathcal{K}_j^n} + \boldsymbol{\tau}^-_{\mathcal{K}_{j+1}^n})/2, \tag{11}$$

$$[\![\nu]\!] = \nu^-_{\mathcal{K}_j^n} \bar{\boldsymbol{n}}_{\mathcal{K}_j^n} + \nu^-_{\mathcal{K}_{j+1}^n} \bar{\boldsymbol{n}}_{\mathcal{K}_{j+1}^n}, \quad [\![\boldsymbol{\tau}]\!] = \boldsymbol{\tau}^-_{\mathcal{K}_j^n} \cdot \bar{\boldsymbol{n}}_{\mathcal{K}_j^n} + \boldsymbol{\tau}^-_{\mathcal{K}_{j+1}^n} \cdot \bar{\boldsymbol{n}}_{\mathcal{K}_{j+1}^n}. \tag{12}$$

Note that the jump $[\![\nu]\!]$ is vector, whereas the jump $[\![\boldsymbol{\tau}]\!]$ is a scaler.

4. Space-time DG spectral element method

4.1. Space-time formulation

In the space-time domain, we introduce a vector function $\boldsymbol{B} \in \mathbb{R}^{1\times(d+1)}$ and a matrix $\boldsymbol{A} \in \mathbb{R}^{(d+1)\times(d+1)}$, as,

$$\boldsymbol{B} = \begin{pmatrix} \boldsymbol{F}(u)/u, & 1 \end{pmatrix}, \quad \boldsymbol{A} = \begin{pmatrix} \boldsymbol{\epsilon}(u) & \boldsymbol{0} \\ \boldsymbol{0}^\top & 0 \end{pmatrix}, \tag{13}$$

where $\mathbf{0} \in \mathbb{R}^{d\times 1}$ and d is the spatial dimensions. We then transform (4) into a space-time formulation as follows,

$$\boldsymbol{\nabla} \cdot (\boldsymbol{B}u - \boldsymbol{A}\boldsymbol{\nabla}u) = 0, \quad \boldsymbol{x} \in \mathcal{E}, \tag{14}$$

where $\boldsymbol{\nabla} = (\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, ..., \frac{\partial}{\partial x_{d+1}})$ denotes the gradient operator in the space-time domain, whereas $\nabla = (\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, ..., \frac{\partial}{\partial x_d})$ is the gradient operator with respect to the spatial variables.

To solve Eq. (14), we first use a Picard iteration scheme [36, 34] for which at each Picard iteration a linear advection-diffusion equation has to be solved in the space-time framework. That is, at k-th Picard iteration, we solve,

$$\boldsymbol{\nabla} \cdot (\boldsymbol{B}^{(k)}u^{(k+1)} - \boldsymbol{A}^{(k)}\boldsymbol{\nabla}u^{(k+1)}) = 0. \tag{15}$$

To derive the space-time DG weak formulation for Eq. (15), we follow the same approach presented in [37]. By introducing an auxiliary variable $\boldsymbol{\sigma}$, we rewrite Eq. (15) into a first order system

$$\boldsymbol{\sigma}^{(k+1)} = \boldsymbol{\nabla}u^{(k+1)}, \tag{16}$$

$$\boldsymbol{\nabla} \cdot (\boldsymbol{B}^{(k)}u^{(k+1)} - \boldsymbol{A}^{(k)}\boldsymbol{\sigma}^{(k+1)}) = 0. \tag{17}$$

4.2. Discretization of DG spectral element method

In this section, we present the derivation of the space-time DG spectral element method for Eqs. (16) and (17). Without loss of generality, we drop the iteration symbols, i.e., $^{(k)}$ and $^{(k+1)}$ when deriving the space-time DG weak formulations.

4.2.1. Space-time DG weak formulations

The auxiliary equation. To derive the space-time DG weak formulation for the auxiliary equation, we multiply (16) with a test function $\boldsymbol{\tau} \in \boldsymbol{\Sigma}_h$, substitute u with the approximation $u_h \in V_h$, and integrate over a space-time element $\mathcal{K}_j^n \in \mathcal{T}_h^n$. Next, we perform integration by parts once to obtain,

$$\int_{\mathcal{K}_j^n} \boldsymbol{\sigma}_h \cdot \boldsymbol{\tau} d\mathcal{K}_j^n = -\int_{\mathcal{K}_j^n} u_h \boldsymbol{\nabla} \cdot \boldsymbol{\tau} d\mathcal{K}_j^n + \int_{\partial\mathcal{K}_j^n} \hat{u}_h^{\epsilon} \bar{\boldsymbol{n}} \cdot \boldsymbol{\tau}^- d\partial\mathcal{K}_j^n. \tag{18}$$

where $\bar{\boldsymbol{n}}$ denotes the outward unit normal vector of the space-time element $\mathcal{K}_j^n$ and $\hat{u}_h^{\epsilon}$ denotes numerical flux.

The governing equation. We derive the space-time DG weak formulation for (17) by multiplying it with a test function $\nu \in V_h$, substituting u and $\boldsymbol{\sigma}$ with the approximations $u_h \in V_h$ and $\boldsymbol{\sigma}_h \in \boldsymbol{\Sigma}_h$, and integrating over a space-time element $\mathcal{K}_j^n \in \mathcal{T}_h^n$. By performing integration by parts once, we have,

$$-\int_{\mathcal{K}_j^n} (\boldsymbol{B} u_h - \boldsymbol{A}\boldsymbol{\sigma}_h) \cdot \boldsymbol{\nabla}\nu d\mathcal{K}_j^n + \int_{\partial\mathcal{K}_j^n} (\boldsymbol{B}\hat{u}_h - \boldsymbol{A}\hat{\boldsymbol{\sigma}}_h^{\epsilon}) \cdot \bar{\boldsymbol{n}}\nu^- d\partial\mathcal{K}_j^n = 0, \tag{19}$$

where $\hat{u}_h$ and $\hat{\boldsymbol{\sigma}}_h^{\epsilon}$ are numerical fluxes.

4.2.2. Definition of numerical fluxes

We separate the numerical fluxes into two categories: the convective fluxes, i.e., $\hat{u}_h$; the numerical fluxes related to the artificial viscosity term, i.e., $\hat{u}_h^{\epsilon}$ and $\hat{\boldsymbol{\sigma}}_h^{\epsilon}$.

For the convective fluxes, we choose the upwind flux,

$$\boldsymbol{B}\hat{u}_h = \{\{\boldsymbol{B}\}\}\{\{u_h\}\} + \frac{1}{2}|\{\{\boldsymbol{B}\}\} \cdot \bar{\boldsymbol{n}}| [\![u_h]\!]. \tag{20}$$

For the numerical fluxes $\hat{u}_h^{\epsilon}$ and $\hat{\boldsymbol{\sigma}}_h^{\epsilon}$, there are several different formulations presented Arnold et al [38]. Here, we choose the local discontinuous

Galerkin (LDG) fluxes [37] for its compact stencils,

$$\hat{u}_h^{\epsilon} = \{\{u_h\}\} - \boldsymbol{\beta} \cdot [\![u_h]\!], \tag{21}$$

$$\hat{\boldsymbol{\sigma}}_h^{\epsilon} = \{\{\boldsymbol{\sigma}_h\}\} + \boldsymbol{\beta} \cdot [\![\boldsymbol{\sigma}_h]\!] - \alpha_j [\![u_h]\!], \tag{22}$$

where $\boldsymbol{\beta} \cdot \boldsymbol{n} = sign(\boldsymbol{v} \cdot \boldsymbol{n})/2$ ([39]) with $\boldsymbol{v} = (1, 1, 1)^\top$ and $\alpha_j \geq 0$ is referred as the stabilization parameter.

4.2.3. The space-time slab linear system

From (18) and (19), we notice that the outward normal vectors, $\bar{\boldsymbol{n}}$, at the faces of K^n and $K^{(n-1)}$ in a space-time slab $\mathcal{E}^n$ are $(\mathbf{0}, 1)$ and $(\mathbf{0}, -1)$, respectively. This makes the space-time slab $\mathcal{E}^n$ only depend on its previous space-time slab $\mathcal{E}^{n-1}$, which leads to weakly coupled space-time slabs. We then perform summation over all elements of the tessellation $\mathcal{T}_h^n$ in one space-time slab $\mathcal{E}^n$ for (18) and (19),

$$\sum_j \int_{\mathcal{K}_j^n} \boldsymbol{\sigma}_h \cdot \boldsymbol{\tau} d\mathcal{K}_j^n = \sum_j \left(- \int_{\mathcal{K}_j^n} u_h \boldsymbol{\nabla} \cdot \boldsymbol{\tau} d\mathcal{K}_j^n + \int_{\partial \mathcal{K}_j^n} \hat{u}_h^{\epsilon} \bar{\boldsymbol{n}} \cdot \boldsymbol{\tau}^- d\partial\mathcal{K}_j^n \right), \tag{23}$$

$$\sum_j \left(- \int_{\mathcal{K}_j^n} (\boldsymbol{B} u_h - \boldsymbol{A}\boldsymbol{\sigma}_h) \cdot \boldsymbol{\nabla} \nu d\mathcal{K}_j^n + \int_{\partial \mathcal{K}_j^n} (\boldsymbol{B} \hat{u}_h - \boldsymbol{A} \hat{\boldsymbol{\sigma}}_h^{\epsilon}) \cdot \bar{\boldsymbol{n}} \nu^- d\partial\mathcal{K}_j^n \right) = 0. \tag{24}$$

Remark 4.1. *The weak coupling between space-time slabs reduces the number of unknowns of the linear system obtained at each Picard iteration from* $\prod_{j=1}^{d+1}(p^{(x_j)} + 1)E^{(x)}E^{(t)}$ *to* $\prod_{j=1}^{d+1}(p^{(x_j)} + 1)E^{(x)}$*, and enables the local resolution to be flexible; i.e., polynomial orders and grid resolutions can vary from one space-time slab to another.*

According to the tensor product basis (8) with $d = 2$, the approximations u_h and $\boldsymbol{\sigma}_h$ are written in nodal Lagrange form,

$$u \sim u_h(x_1, x_2, x_3) = \sum_{k=0}^{p^{(x_3)}} \sum_{j=0}^{p^{(x_2)}} \sum_{i=0}^{p^{(x_1)}} u_{i,j} \ell_i(\xi_1) \ell_j(\xi_2) \ell_k(\xi_3), \tag{25}$$

$$\boldsymbol{\sigma} \sim \boldsymbol{\sigma}_h(x_1, x_2, x_3) = \sum_{k=0}^{p^{(x_3)}} \sum_{j=0}^{p^{(x_2)}} \sum_{i=0}^{p^{(x_1)}} \boldsymbol{\sigma}_{i,j} \ell_i(\xi_1) \ell_j(\xi_2) \ell_k(\xi_3). \tag{26}$$

To proceed, we approximate the integrals in (23) and (24) using Gaussian quadrature. Here, we choose the quadrature nodes to be the same as the interpolation nodes, i.e., Legendre-Gauss nodes in the spatial directions and Legendre-Gauss-Lobatto nodes in the time direction. The reason for such choice is that an identity mass matrix is obtained in each space-time element.

In addition, the numerical flux $\hat{u}_h^\epsilon$ only depends on the variable u_h. So the auxiliary variable $\boldsymbol{\sigma}_h$ can be eliminated from the formulation (24) by solving for $\boldsymbol{\sigma}_h$ in (23) and then substituting the resulting $\boldsymbol{\sigma}_h$ into (24). At the k-th Picard iteration, we obtain a "space-time slab", which is local to the slab in contrast to global to the whole time domain, linear system for the space-time slab $\mathcal{E}^n$ as,

$$\boldsymbol{H}(u_h^{(k)}) u_h^{(k+1)} = \boldsymbol{f}(u_h^{n-1}, g_{b,h}^k), \quad \boldsymbol{x} \in \mathcal{E}^n \tag{27}$$

where $\boldsymbol{H}(u_h^{(k)})$ denotes the matrix of the linear system due to (13) and (15), and the vector $\boldsymbol{f}(u_h^{n-1}, g_{b,h}^k)$ takes into account the solution at $\Omega_{(t_{n-1})}$ computed from the previous space-time slab $\mathcal{E}^{(n-1)}$ as well as the boundary conditions imposed on $\mathcal{E}^n \setminus (\Omega_{(t_{n-1})} \cup \Omega_{(t_n)})$. Note that there is no boundary condition imposed on $\Omega_{(t_n)}$. (See Fig. 2)

Now, we examine the structure of the matrix $\boldsymbol{H}(u_h^{(k)})$ in the first space-time slab $\mathcal{E}^1$. The matrix $\boldsymbol{H}(u_h^{(k)})$ depends on the polynomial order in space,

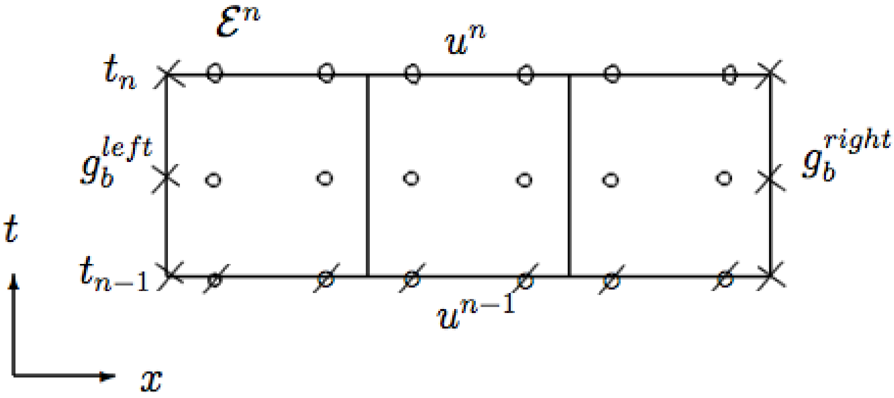

Figure 2: An Example of nodes distribution of the space-time slab $\mathcal{E}^n \in \mathbb{R}^2$ $(d = 2)$, with the coordinates of a point $\bar{\boldsymbol{x}}$ denoted by $\bar{\boldsymbol{x}} = (x_1, x_2)$, where $x_1 = x$ is the spatial variable and the time variable t is denoted as x_2. In the spatial direction x, the total number of Legendre Gauss points is 2, while 3 Legendre Gauss-Lobatto points are used in the time direction t. g_b^{left} and g_b^{right} denote the boundary data. The symbol ø denotes the known values, u^{n-1}, while the symbol × denotes the boundary values, $g_{b,h}^k$. All these values form the vector $\boldsymbol{f}(u_h^{n-1}, g_{b,h}^k)$ in (27).

$p^{(\boldsymbol{x})}$, the polynomial order in time, $p^{(t)}$, and the number of element, $E^{(\boldsymbol{x})}$. For example, we solve the Burgers' equation in one spatial dimension, such that $f(u) = u^2/2$, and choose $p^{(x)} = 2$, $p^{(t)} = 2$ and $E^{(x)} = 2$. The order of the matrix $\boldsymbol{H}(u_h^{(k)})$ is (18×18), i.e. $(p^{(x)} + 1) \times (p^{(t)} + 1) \times E^{(x)} = 18$, and the structure of the matrix, $\boldsymbol{H}(u_h^{(k)})$, is shown as follows

$$\boldsymbol{H}(u_h^{(k)})_{(18\times18)} = \begin{pmatrix}
\ast & \ast & \ast & \ast & 0 & 0 & \ast & 0 & 0 & \ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & 0 \\
\ast & \ast & \ast & 0 & \ast & 0 & 0 & \ast & 0 & \ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & 0 \\
\ast & \ast & \ast & 0 & 0 & \ast & 0 & 0 & \ast & \ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & 0 \\
\ast & 0 & 0 & \ast & \ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast & 0 & 0 & 0 \\
0 & \ast & 0 & \ast & \ast & \ast & 0 & \ast & 0 & 0 & 0 & 0 & \ast & \ast & \ast & 0 & 0 & 0 \\
0 & 0 & \ast & \ast & \ast & \ast & 0 & 0 & \ast & 0 & 0 & 0 & \ast & \ast & \ast & 0 & 0 & 0 \\
\ast & 0 & 0 & \ast & 0 & 0 & \ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast \\
0 & \ast & 0 & 0 & \ast & 0 & \ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast \\
0 & 0 & \ast & 0 & 0 & \ast & \ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast \\
\ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast & \ast & 0 & 0 & \ast & 0 & 0 \\
\ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast & 0 & \ast & 0 & 0 & \ast & 0 \\
\ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast & 0 & 0 & \ast & 0 & 0 & \ast \\
0 & 0 & 0 & \ast & \ast & \ast & 0 & 0 & 0 & \ast & 0 & 0 & \ast & \ast & \ast & \ast & 0 & 0 \\
0 & 0 & 0 & \ast & \ast & \ast & 0 & 0 & 0 & 0 & \ast & 0 & \ast & \ast & \ast & 0 & \ast & 0 \\
0 & 0 & 0 & \ast & \ast & \ast & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast & \ast & 0 & 0 & \ast \\
0 & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast & \ast & 0 & 0 & \ast & 0 & 0 & \ast & \ast & \ast \\
0 & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast & 0 & \ast & 0 & 0 & \ast & 0 & \ast & \ast & \ast \\
0 & 0 & 0 & 0 & 0 & 0 & \ast & \ast & \ast & 0 & 0 & \ast & 0 & 0 & \ast & \ast & \ast & \ast
\end{pmatrix}$$

Note that $\boldsymbol{H}(u_h^{(k)})$ is a sparse non symmetric matrix.

5. Procedures for treating shocks

5.1. Shock capturing

Since the idea of adding artificial viscosity is to eliminate Gibbs oscillations, it is crucial to determine where the solution is discontinuous and how much artificial viscosity is needed. The smoothness indicator introduced by Persson and Peraire [1], which is similar to error indicators for adaptation in spectral element methods [40], assumes that the polynomial expansion has a similar behavior to the Fourier expansion. Here, we apply the smoothness indicator:

$$S_K = \frac{\int_K (u_h - \hat{u}_h)^2 dK}{\int_K (u_h)^2 dK}, \tag{28}$$

where u_h is the approximation of order $p(\boldsymbol{x})$ and $\hat{u}_h$ is the truncated representation of u_h at order $p(\boldsymbol{x})-1$. Since the coefficients in a Fourier expansion decay like $1/p(\boldsymbol{x})^2$ and the sensor (28) involves square quantities, we have that the approximation u_h is at most $\mathcal{C}^0$ for element K if $S_K > 1/p(\boldsymbol{x})^4$.

Once a shock is detected in a spatial element, we determine the amount of artificial viscosity by the following elementwise-constant formulation, which is due to Persson and Peraire [1]:

$$\epsilon_K = \begin{cases} 0, \quad s_K < s_0 - \kappa, \\ \frac{\epsilon_0}{2}(1 + \sin\frac{\pi(s_k - s_0)}{2\kappa}), \quad s_0 - \kappa \le s_K \le s_0 + \kappa, \\ \epsilon_0, \quad s_K > s_0 + \kappa, \end{cases} \tag{29}$$

where $s_K = \log_{10} S_K$, ϵ_0 is a maximum value, s_0 and κ are empirical tuning parameters. The parameter s_0 should be scaled as $s_0 \sim -4\log_{10} p(\boldsymbol{x})$ due to the coefficients of the polynomial expansion decay like $1/p(\boldsymbol{x})^4$.

Remark 5.1. *The integral in* (28) *is over a spatial element instead of a space-time element in a space-time slab, which allows us to apply different amount of artificial viscosity at different time level in one space-time slab.*

Remark 5.2. *Compared to the elementwise-constant formulation, there are some work on a smooth artificial viscosity formulation, i.e., [2], [3].*

5.2. Shock tracking

Discontinuous initial data produces Gibbs-type oscillations, which persist at least for a short time even in the artificial viscosity method. To avoid this spurious behavior, one can either smooth the discontinuous initial profiles, e.g., a hyperbolic-tangent smoothing mentioned in [3], or track the discontinuity explicitly [21].

Incorporating the shock tracking procedure in the present space-time discontinuous Galerkin spectral element method follows the work of Pei et al. [35]. The idea is to combine the Eulerian description, i.e., a fixed frame of reference, with a Lagrangian description, i.e., a moving frame of reference by two global transformations, a backward transformation and a forward transformation. For one spatial dimension problems, two global transformations are defined as follows,

- the backward transformation,

$$\tilde{x}(x,t) = x - \int_{t_n}^{t} W(s)ds, \tag{30}$$

- the forward transformation,

$$x(\tilde{x},t) = \tilde{x} + \int_{t_n}^{t} W(s)ds,. \tag{31}$$

x is the spatial variable corresponding to the fixed frame of reference and $\tilde{x}$ is the coordinate of the moving frame of reference. W is the normal interface velocity, i.e., the shock speed. In the moving frame of reference, the transformed governing equation (4) is then,

$$\tilde{u}_t + (f(\tilde{u}) - W\tilde{u})_{\tilde{x}} = (\epsilon(\tilde{u})\tilde{u}_{\tilde{x}})_{\tilde{x}}. \tag{32}$$

The corresponding space-time formulation is written as

$$\tilde{\boldsymbol{\nabla}} \cdot (\tilde{\boldsymbol{B}}^{*} \tilde{u} - \tilde{\boldsymbol{A}} \tilde{\boldsymbol{\nabla}} \tilde{u}) = 0, \tag{33}$$

where $\tilde{\boldsymbol{\nabla}}$ is the gradient operator defined in the moving frame of reference, and $\tilde{\boldsymbol{A}}$, $\tilde{\boldsymbol{B}}^{*}$ are defined as follows,

$$\tilde{\boldsymbol{B}}^{*} = \begin{pmatrix} f(\tilde{u})/\tilde{u} - W, & 1 \end{pmatrix}, \quad \tilde{\boldsymbol{A}} = \begin{pmatrix} \epsilon(\tilde{u}) & 0 \\ 0 & 0 \end{pmatrix}. \tag{34}$$

Remark 5.3. *The computed shock speed, W, needs to be updated along with the solution in each space-time slab due to its dependence on the solution on either side of the front and its influence on the transformations* (30) *and* (31)*.*

Remark 5.4. *In higher spatial dimensions, the derivation of the transformations and transformed governing equation can be found in [35].*

Remark 5.5. *The interface is updated by a level set procedure which captures the location of the discontinuity, e.g., [41, 42, 21].*

6. Outline of the space-time DG spectral element method

We present an outline of our space-time DG spectral element method for solving nonlinear hyperbolic problems in Algorithm 1. Combined with either the shock capturing method or the shock tracking method, we illustrate our method based on the space-time formulation (33) in one spatial dimension. At the k-th Picard iteration, (33) is written as

$$\tilde{\boldsymbol{\nabla}} \cdot (\tilde{\boldsymbol{B}}^{*,(k)} \tilde{u}^{(k+1)} - \tilde{\boldsymbol{A}}^{(k)} \tilde{\boldsymbol{\nabla}} \tilde{u}^{(k+1)}) = 0, \tag{35}$$

Note that $\tilde{\boldsymbol{B}}^{*,(k)}$ and $\tilde{\boldsymbol{A}}^{(k)}$ are nonlinear terms, such that,

$$\tilde{\boldsymbol{B}}^{*,(k)} = \tilde{\boldsymbol{B}}^{*,(k)}(\tilde{u}^{(k)}, W^{(k)}), \quad \tilde{\boldsymbol{A}}^{(k)} = \tilde{\boldsymbol{A}}(\tilde{u}^{(k)}). \tag{36}$$

Algorithm 1 the space-time discontinuous Galerkin spectral element method for solving nonlinear hyperbolic problems in a space-time slab $\mathcal{E}^{n+1}$.

```
Given the solution $u^n$ and the level set function $\phi^n$ at time $t_n$ in the
    space-time slab $\mathcal{E}^{n+1}$ with slab thickness $\Delta t$.
Compute $W^n$ at the discontinuity by the Rankine-Hugoniot jump condition.
Define a space-time slab $\tilde{\mathcal{E}}^n$ in the moving frame of reference by the
    backward transformation (30) and the forward transformation (31).
Compute the vector $\tilde{\boldsymbol{B}}^{*,n}$ based on $\tilde{u}^n$, and define $V^n = f(\tilde{u}^n)/\tilde{u}^n$;
Compute the indicator $S_K$ (28) and the matrix $\tilde{\boldsymbol{A}}^n$.
Set $V^{j,(k)} = V^n$ and $W^{j,(k)} = W^n$, where $k = 0$ and $j = 1, ..., p^{(t)}$. We
    initialize $\tilde{\boldsymbol{A}}$ as $\tilde{\boldsymbol{A}}^{0,(k)} = \tilde{\boldsymbol{A}}^n$ and $\tilde{\boldsymbol{A}}^{j,(k)} = 0$. Note that $V^{p^{(t)},(k)} = V^{n+1,(k)}$,
    $W^{p^{(t)},(k)} = W^{n+1,(k)}$ and $\tilde{\boldsymbol{A}}^{p^{(t)},(k)} = \tilde{\boldsymbol{A}}^{n+1,(k)}$.
for $k_s = 1, 2$ do
    (a) while ($\|V^{n+1,(k+1)} - V^{n+1,(k)}\| > tol$) do
        i Use the DG spectral element method to discretize (35). The
          discretization of (35) is discussed in section 4.2.
        ii Solve the resulting space-time slab linear system (27), which is
          discussed in section 4.2.3
        iii Compute $V^{j,(k+1)}$ ($j = 1, ..., p^{(t)}$).
        iv Update $W^{j,(k+1)}$ at $j = 1, ..., p^{(t)}$, and $\tilde{\boldsymbol{B}}^{*,(k+1)}$.
    (b) end while
    (c) Compute $V^j$ and $W^j$ at $j = 1, ..., p^{(t)}$, and the then update $\tilde{\boldsymbol{B}}^*$.
    (d) Compute the indicator $S_K$ (28) and $\tilde{\boldsymbol{A}}^{j,(k+1)}$ at $j = 1, ..., p^{(t)}$.
end for
Compute $\phi^{p^{(t)},(k)}$ by the forward transformation, and $u^{p^{(t)},(k+1)}$ by the
    backward transformation.
return $u^{n+1}$ and $\phi^{n+1}$.
```

7. Numerical Experiments

In this section, we test the space-time DG spectral element method on a variety of one dimensional test problems to demonstrate the capabilities of our approach in preserving the accuracy and capturing or tracking sharp shock profiles. Note that $\boldsymbol{H}(u_h^{(k)})$ in Eq. (27) is a non-symmetric matrix due to the discretization of the convection term. In our tests, we apply a direct solver from the LAPACK library in order to invert the resulting linear system. The tolerance of the Picard iteration is set to be 10^{-13}.

7.1. Convergence analysis

In this test, the numerical results are compared with the exact solutions in order to examine the convergence rate with respect to both space and time. In the last space-time slab $\mathcal{E}^{E^{(t)}}$, the errors are measured in the l_∞ norm at time $t_{E^{(t)}}$, i.e. $t_{E^{(t)}} = T$, where T is the final computational time.

$$\|Err_u\|_\infty = \max_{i=0,\ldots,p^{(x)}} \|u_i^{E^{(t)}} - (u_h)_i^{E^{(t)}}\| \tag{37}$$

$u_i^{E^{(t)}}$ denotes the exact solution evaluated at node x_i, and $(u_h)_i^{E^{(t)}}$ is the approximation evaluated at the same node.

We consider the Burgers' equation

$$u_t + (\frac{u^2}{2})_x = (\epsilon u_x)_x, \quad x \in [-1.0, 1.0], \tag{38}$$

with an exact traveling wave solution

$$u(x,t) = -\tanh(\frac{x + 0.5 - t}{2\epsilon}) + 1.0. \tag{39}$$

Dirichlet boundary conditions are imposed by using the exact solution. We refer the reader to [43] and the references therein for deriving an exact solution of the Burgers' equation. We choose $\epsilon = 0.5$.

First, we demonstrate the spectral accuracy of the approximation by plotting the maximum errors in the solution as a function of the spatial polynomial order, $p^{(x)}$. The simulation is computed over the time $t = 0$ to $t = 1.0$, and the comparison is made among different numbers of spatial elements, i.e., $E^{(x)} = 5$ and $E^{(x)} = 10$. The ratio of the time step to the spatial element size is, $\Delta t/h = 0.125$ for $E^{(x)} = 5$ and $\Delta t/h = 0.25$ for $E^{(x)} = 10$. On the left part of Fig. 3, the polynomial order in the temporal direction, $p^{(t)}$, is chosen to be the same as $p^{(x)}$, while $p^{(t)}$ is chosen to be 12 so that the temporal errors are negligible on the right part. Both choices of $p^{(t)}$ exhibit spectral accuracy in space.

Next, we demonstrate the spectral accuracy of the approximation by plotting the maximum errors in the solution as a function of the temporal polynomial order, $p^{(t)}$. The simulation is computed over the time $t = 0$ to $t = 1.0$, and the spatial domain is divided into 5 elements, i.e., $E^{(x)} = 5$. The comparison is made among $E^{(t)} = 10$ and $E^{(t)} = 20$. The ratio of the time step to the spatial element size is, $\Delta t/h = 0.25$ for $E^{(t)} = 10$ and $\Delta t/h = 0.125$ for $E^{(t)} = 20$. On the left part of Fig. 4, the polynomial order in the spatial direction, $p^{(x)}$, is chosen to be the same as $p^{(t)}$, while $p^{(x)}$ is chosen to be 12 so that the spatial errors are negligible on the right part. Both choices of $p^{(x)}$ exhibit spectral accuracy in time.

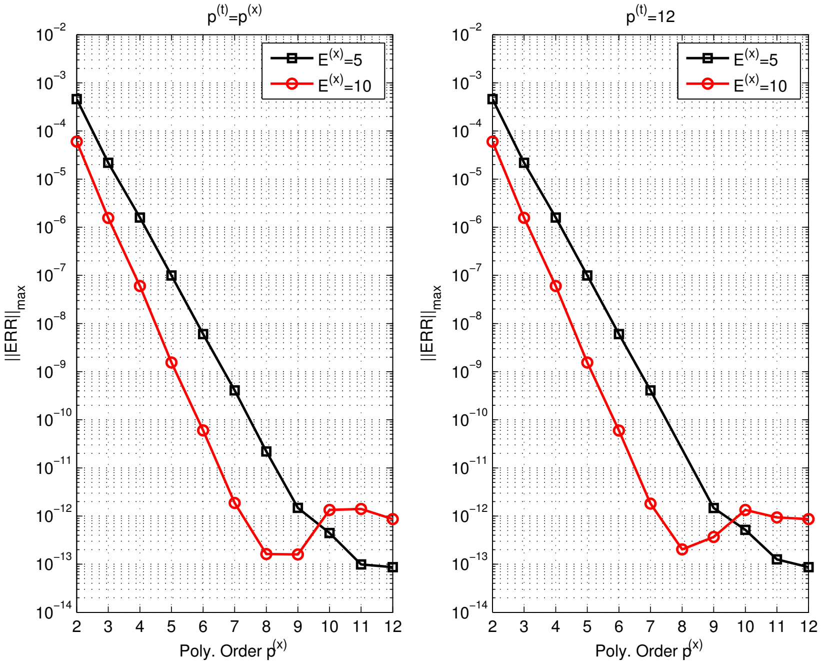

Figure 3: Errors in the solution as a function of polynomial order $p^{(x)}$. the polynomial order in the temporal direction, $p^{(t)}$, is chosen to be the same as $p^{(x)}$ on the left part, and to be 12 on the right part. The comparison is made between $E^{(x)} = 5$ and $E^{(x)} = 10$. The ratio of the time step to the spatial element size is, $\Delta t/h = 0.125$ for $E^{(x)} = 5$ and $\Delta t/h = 0.25$ for $E^{(x)} = 10$. The simulation is computed over the time $t = 0$ to $t = 1.0$.

7.2. The spatio-temporal artificial viscosity

In order to feature the gains of our artificial diffusion approach, we test our approach on a linear advection problem with periodic boundary conditions as follows,

$$\begin{cases} u_t + u_x = 0, \quad -1 \leq x \leq 1, t > 0 \\ u(x,0) = u_0(x). \end{cases}$$

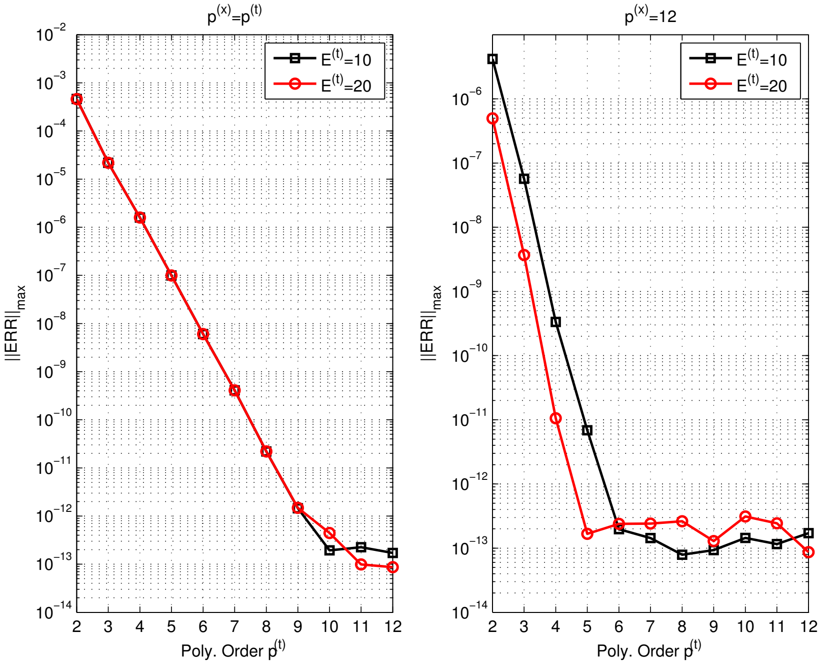

Figure 4: Errors in the solution as a function of polynomial order $p^{(t)}$. the polynomial order in the spatial direction, $p^{(x)}$, is chosen to be the same as $p^{(t)}$ on the left part, and to be 12 on the right part. The comparison is made among $E^{(t)} = 10$ and $E^{(t)} = 20$. The ratio of the time step to the spatial element size is, $\Delta t/h = 0.25$ for $E^{(t)} = 10$ and $\Delta t/h = 0.125$ for $E^{(t)} = 20$. The simulation is computed over the time $t = 0$ to $t = 1.0$, and $E^{(x)} = 5$.

The initial condition is chosen to be

$$u_0(x) = \begin{cases} 1.0, & \text{if } x \in [-1.0, -0.5), \\ \sin(\pi(x + 0.5)), & \text{if } x \in [-0.5, 0.5), \\ 1.0, & \text{if } x \in [0.5, 1.0], \end{cases}$$

First, we compute the solutions without adding any artificial viscosity. Fig. 5 compares the numerical solutions with the exact solutions at time $t = 2.0$. Comparisons between different pairs of polynomial orders are also shown, i.e., $(p^{(x)}, p^{(t)}) = (10, 5)$ (no the left) and $(p^{(x)}, p^{(t)}) = (10, 7)$ (on the right). The results obtained by a polynomial with a higher temporal order, $p^{(t)} = 7$, are the same as the one computed by the one with a lower temporal order, $p^{(t)} = 5$.

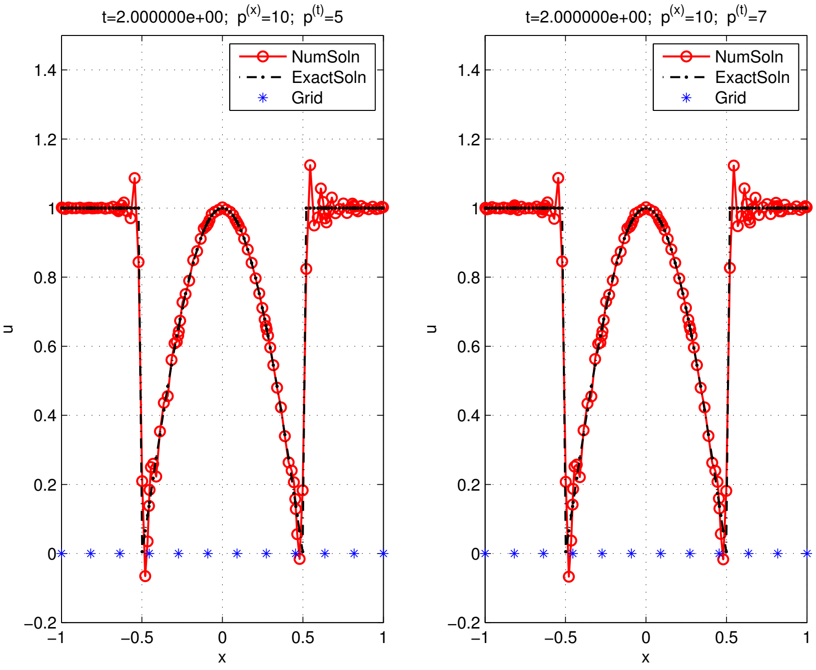

Figure 5: A comparison between the numerical results, without any artificial viscosity, and exact solutions using different pairs of polynomial orders, $(p^{(x)}, p^{(t)}) = (10, 5)$ (on the left) and $(p^{(x)}, p^{(t)}) = (10, 7)$ (on the right). The simulation is computed over the time $t = 0$ to $t = 2.0$, and $E^{(x)} = 11$. The ratio of the time step to the element size, $\Delta t/h = 0.055$.

Now, we demonstrate the performance of adding artificial viscosity. Initially, in a given space-time slab, the only known information in order to compute the amount of artificial viscosity is at the beginning of the slab. In order to obtain the amount of artificial viscosity in the whole space-time slab, a *prediction-correction* scheme is implemented. The simulation is computed over the time $t = 0$ to $t = 2.0$, and $E^{(x)} = 10$. The ratio of the time step to the element size, $\Delta t/h = 0.25$. For the empirical tuning parameters in (29), we choose $\kappa = 2.5$ and $\epsilon_0 = 0.25h/p^{(x)}$.

In Figs. 6, 7 and 8, comparisons between the numerical solution and the exact solution are plotted on the left, while the amount of artificial viscosity is plotted on the right. Clearly, the oscillations are reduced by adding artificial viscosity. Since our strategy for computing the amount of artificial viscosity is time varying, approximations with a higher order in time, e.g., $p^{(t)} = 5$ or $p^{(t)} = 7$, obtain better results compared to the one with a low order in time, e.g. $p^{(t)} = 3$.

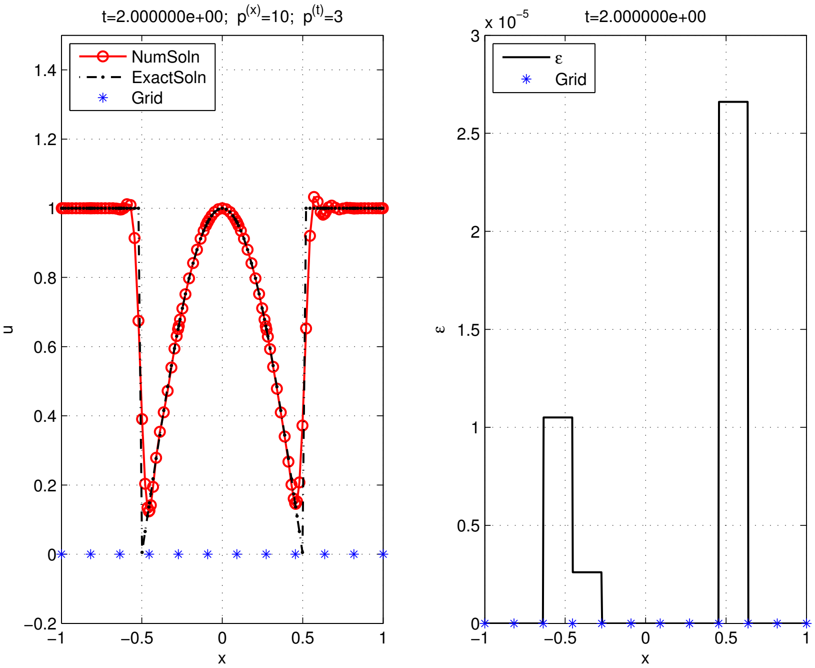

Figure 6: A comparison between the numerical results and exact solutions using $(p^{(x)}, p^{(t)}) = (10, 3)$. On the left, comparisons between the numerical solution and the exact solution are plotted, while the amount of artificial viscosity is plotted on the right.

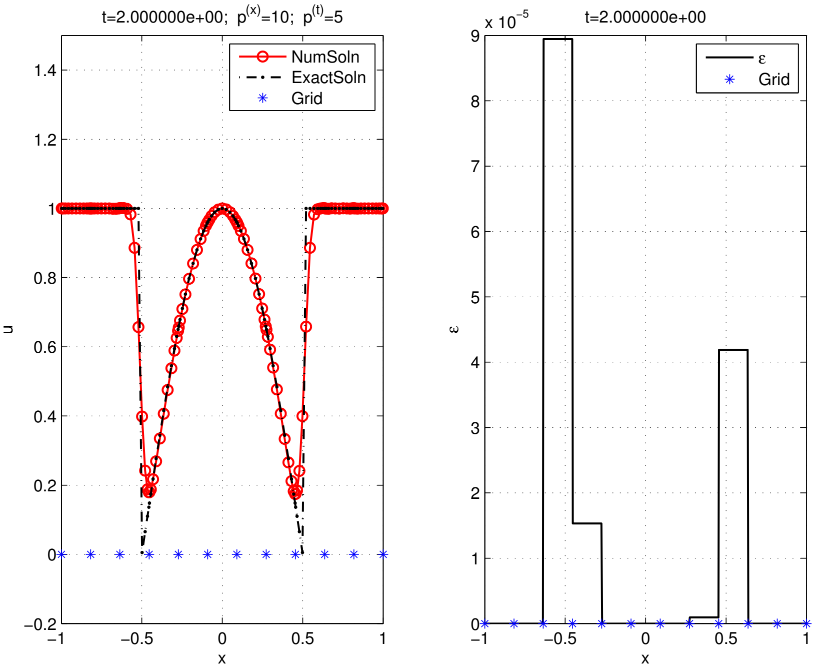

Figure 7: A comparison between the numerical results and exact solutions using $(p^{(x)}, p^{(t)}) = (10, 5)$. On the left, comparisons between the numerical solution and the exact solution are plotted, while the amount of artificial viscosity is plotted on the right.

In Fig. 9, we show the time varying artificial viscosity in the last space-time slab. The amount of artificial viscosity is varying amongst the different time "nodes" in a given space-time slab as the discontinuities travel.

7.3. Inviscid Burgers' equation

The present method is tested on two nonlinear problems. In the first problem, a shock forms at $t = 0.25$, and we apply the artificial viscosity method to capture the shock; the second problem has a sharp discontinuity at the initial time, and we apply the shock tracking procedure.

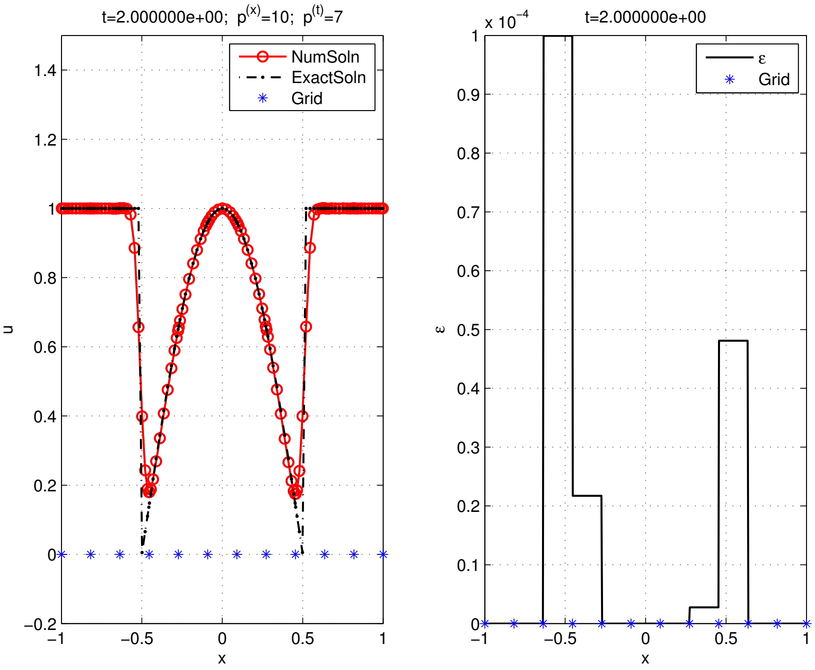

Figure 8: A comparison between the numerical results and exact solutions using $(p^{(x)}, p^{(t)}) = (10, 7)$. On the left, comparisons between the numerical solution and the exact solution are plotted, while the amount of artificial viscosity is plotted on the right.

7.4. Shock capturing

In this section, we study the sensitivity of the error and the number of Picard iterations on varying the temporal and spatial orders, the number of elements, the ratio of the time step to the element size, and whether or not artificial viscosity is present. For the sensitivity study, the inviscid Burgers' equation is solved,

$$\begin{cases} u_t + \partial f(u)/\partial x = 0, \quad 0 \le x \le 1, t > 0 \\ u(x,0) = \frac{1}{2} + \sin(2\pi x). \end{cases}$$

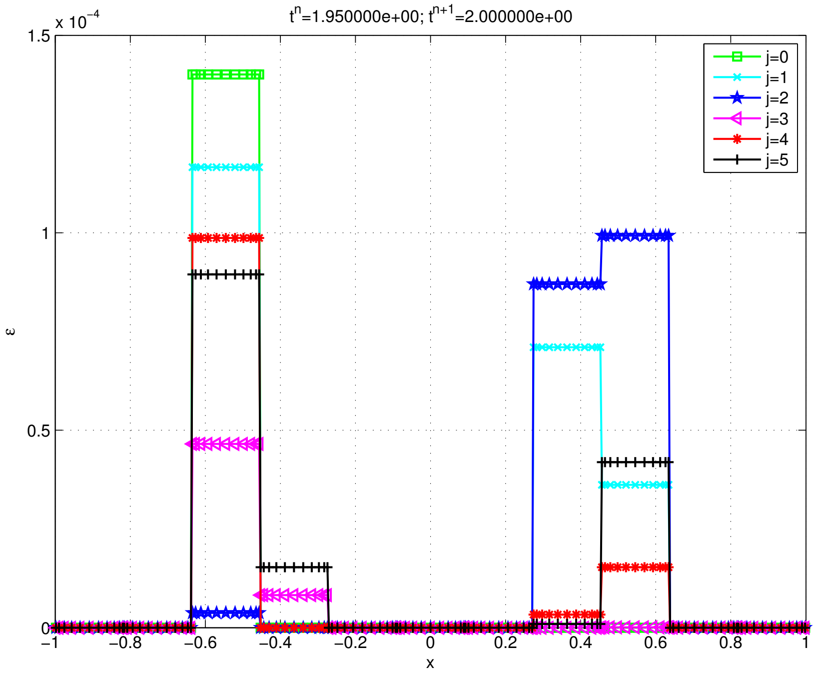

Figure 9: A plot of the time varying artificial viscosity for an approximation with $(p^{(x)}, p^{(t)}) = (10, 5)$ in one space-time slab. The amount of artificial viscosity is plotted for each time level (node), j, in the last time slab.

Periodic boundary conditions are enforced at $x = 0$ and $x = 1$. The solution of this problem forms a shock at t=0.25 that moves to the right. For the empirical tuning parameters in (29), we choose $\kappa = 2.5$ and $\epsilon_0 = h/p^{(x)}$.

First we report in Table 1 the sensitivity of the number of Picard iterations as a function of the temporal order and the ratio of the time step to the spatial element size. The simulation is computed over the time $t = 0.0$ to $t = 0.25$, at which point the shock forms. The number of Picard iterations, which is reported in Table 1, represents the maximum number during the simulation process. As illustrated in the table, approximations with higher temporal orders reduce the number of Picard iterations for this test. Note that the Picard iteration algorithm does not converge within the maximum iteration number (one thousand) in the prediction step for $p^{(t)} = 1$ and $p^{(t)} = 2$ (noted by $(*)$ in Table 1). However, the Picard iteration, in the correction step, converges in 163 ($p^{(t)} = 1$) or 164 ($p^{(t)} = 2$) iterations. We reason that this is because the prediction step, in spite of not converging, yielded an artificial viscosity that was sufficient to damp out noise for carrying out the correction step.

In Figs. 10, 11, 12, 13 and 14, the numerical solutions (on the left) and the amount of artificial viscosity (on the right) are plotted. The simulation is computed over the time $t = 0.0$ to $t = 0.5$. The ratio of the time step to the spatial element size is set as $\Delta t/h = 0.08$. The shock is well-resolved within one element as shown in Fig. 10. In addition, the shock is well-resolved for approximations with different higher orders approximations, i.e., $(p^{(x)}, p^{(t)}) = (8, 5)$ and $(p^{(x)}, p^{(t)}) = (13, 7)$. In comparing Fig. 11 to Fig. 13, an approximation with higher order, $(p^{(x)}, p^{(t)}) = (13, 7)$, resolves the solution extrema more accurately than the one with $(p^{(x)}, p^{(t)}) = (8, 5)$ (the same number of spatial elements are used).

7.5. Shock tracking

The inviscid Burgers' equation with periodic boundary conditions is solved with the following initial conditions,

$$u(x,0) = \begin{cases} \sin(2\pi(x+0.25)) + 0.25, & -0.25 \le x \le 0, \\ -\cos(2\pi x) + 0.25, & -0.25 < x \le 0.25. \end{cases}$$

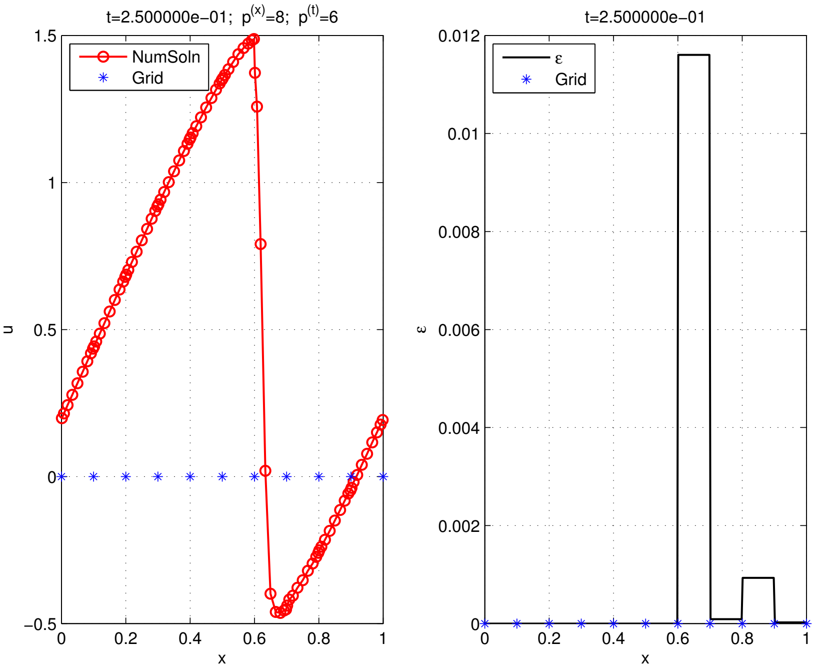

Figure 10: A plot of the approximate solution at $t = 0.25$ (on the left) and the amount of artificial viscosity (on the right). The spatial polynomial order is $p^{(x)} = 8$ and the temporal polynomial order is $p^{(t)} = 6$. The number of spatial elements is $E^{(x)} = 10$. The ratio of the time step to the spatial element size is set as $\Delta t/h = 0.08$.

Since the location of the discontinuity is known at the initial time, we track the discontinuity explicitly. In this section, we report the error of our tracking algorithm with respect to the spatial and temporal polynomial orders. We demonstrate the spectral accuracy of the shock velocity and shock location by plotting the maximum errors as a function of polynomial orders, $p^{(x)}$ and $p^{(t)}$. The errors are computed by comparing our numerical results with the ones computed by a first order Godunov method with ten thousand grid points. Since the temporal resolution affects the number of Picard iterations, see section 7.4, we choose the time step for the convergence tests to be 0.01

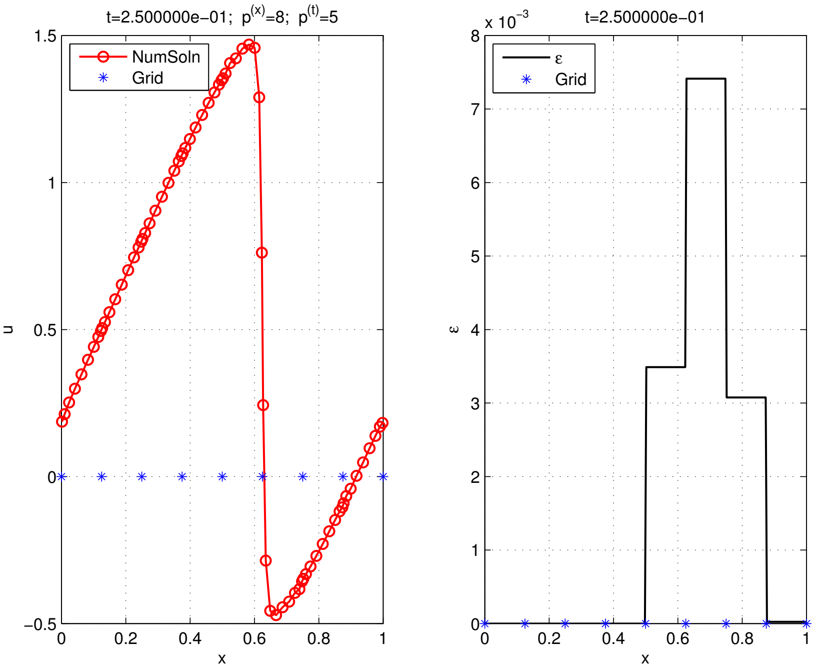

Figure 11: A plot of the approximate solution at $t = 0.25$ (on the left) and the amount of artificial viscosity (on the right). The spatial polynomial order is $p^{(x)} = 8$ and the temporal polynomial order is $p^{(t)} = 5$. The number of spatial elements is $E^{(x)} = 8$. The ratio of the time step to the spatial element size is set as $\Delta t/h = 0.08$.

so that the ratio of the time step to the spatial element size is 0.04 for $E^{(x)} = 2$ and 0.06 for $E^{(x)} = 3$. This setting for Δt allows the Picard iteration algorithm to converge for a lower temporal order, e.g., $p^{(t)} = 1$.

In Fig. 15, we choose the number of spatial elements to be $E^{(x)} = 3$ (on the left) and $E^{(x)} = 2$ (on the right). For the case of $E^{(x)} = 2$, the discontinuity is initialized at the inter-element boundary, which means that there are no cut cells at the beginning of the simulation. After the first time step, a small cell is generated on the left side of the discontinuity so that $\Delta t / \min_K\{h_K\}$ could be greater than 1. For such scenarios, we refine the

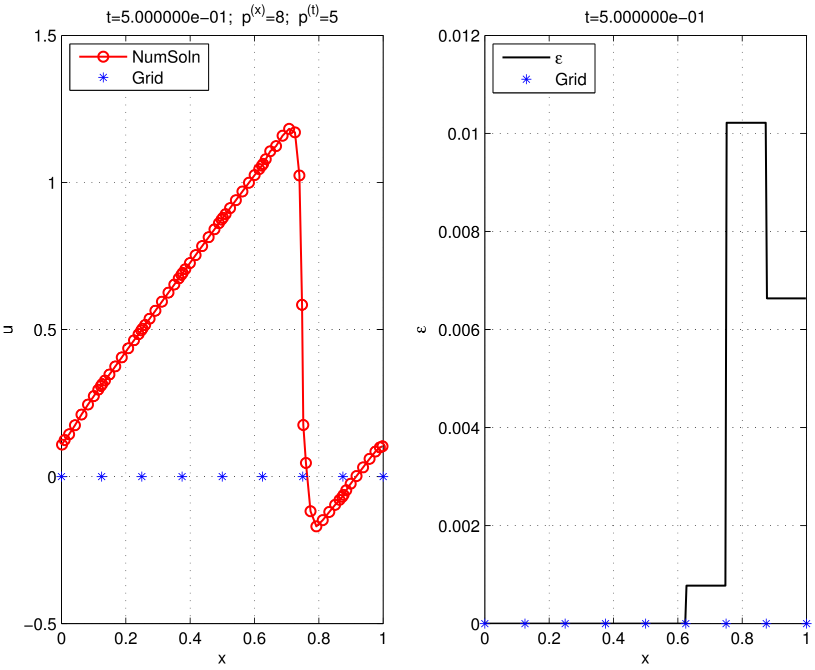

Figure 12: A plot of the approximate solution at $t = 0.5$ (on the left) and the amount of artificial viscosity (on the right). The spatial polynomial order is $p^{(x)} = 8$ and the temporal polynomial order is $p^{(t)} = 5$. The number of spatial elements is $E^{(x)} = 8$. The ratio of the time step to the spatial element size is set as $\Delta t/h = 0.08$.

time step in the given space-time slab. We show that the small cell does not affect the spectral accuracy in our tracking method. Both the interface velocity and the interface location exhibit spectral accuracy in both space and time.

The approximation with $(p^{(x)}, p^{(t)}) = (6, 6)$, in Fig. 16, behaves well with the appearance of the small cell. In Fig. 17, the approximation with $(p^{(x)}, p^{(t)}) = (8, 5)$ behaves well after crossing an inter-element boundary.

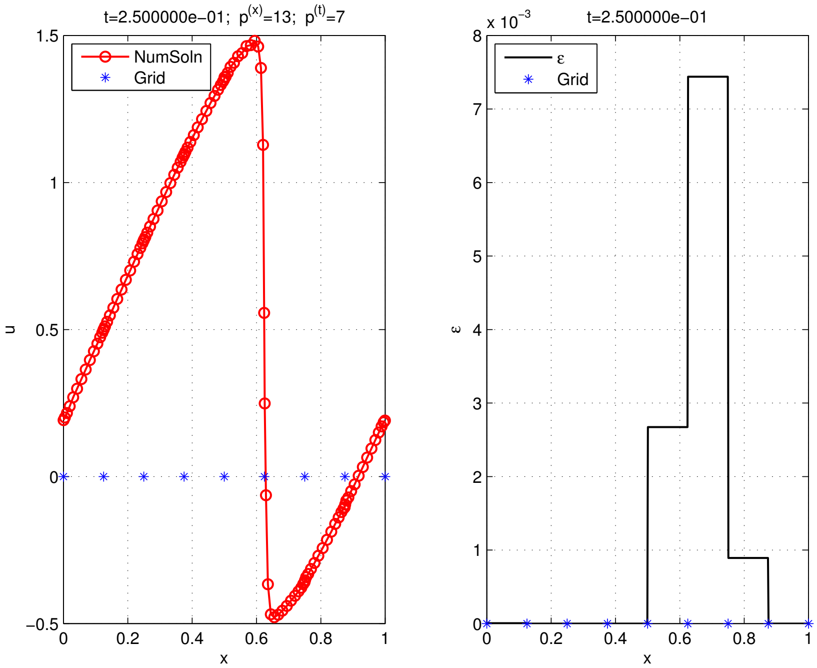

Figure 13: A plot of the approximate solution at $t = 0.25$ (on the left) and the amount of artificial viscosity (on the right). The spatial polynomial order is $p^{(x)} = 13$ and the temporal polynomial order is $p^{(t)} = 7$. The number of spatial elements is $E^{(x)} = 8$. The ratio of the time step to the spatial element size is set as $\Delta t/h = 0.08$.

8. Summary and Conclusions

A space-time DG spectral element method for solving nonlinear hyperbolic problems has been presented in one spatial dimension. In order to treat shocked solutions of hyperbolic problems, two methods are introduced in the space-time framework: (i) the shock capturing approach and (ii) the shock tracking approach.

Numerical experiments on the Burgers' equation demonstrate the spectral convergence in both space and time. The spatio-temporal artificial dif-

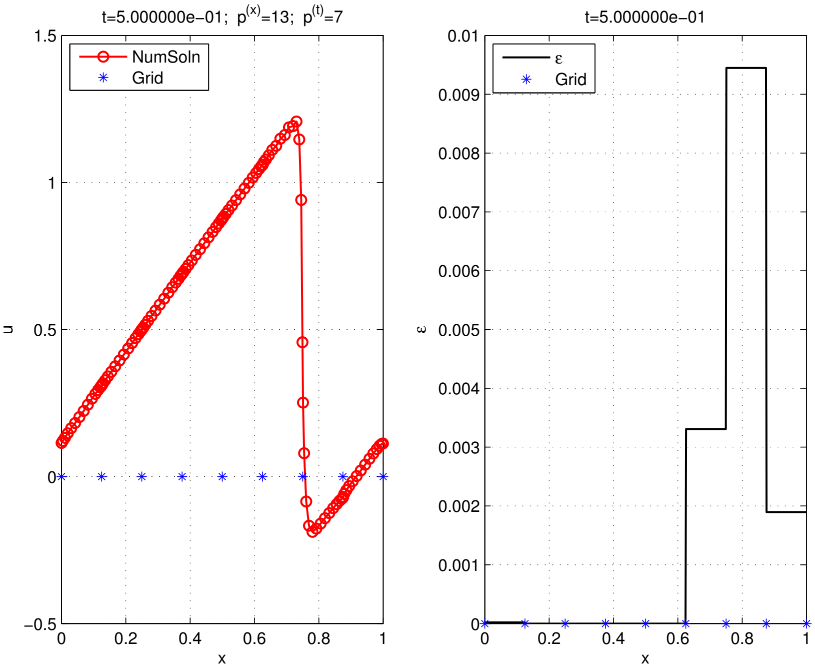

Figure 14: A plot of the approximate solution at $t = 0.5$ (on the left) and the amount of artificial viscosity (on the right). The spatial polynomial order is $p^{(x)} = 13$ and the temporal polynomial order is $p^{(t)} = 7$. The number of spatial elements is $E^{(x)} = 8$. The ratio of the time step to the spatial element size is set as $\Delta t/h = 0.08$.

fusion approach is tested on both a scalar hyperbolic linear problem with a discontinuous initial condition and the inviscid Burgers' equation with a solution that evolves into a sharp discontinuity. In the present shock capturing method, the amount of artificial viscosity is computed at each time level (node) in a given space-time slab based on the smoothness indicator (28), which results in optimal diffusion to suppress Gibbs phenomenon. Space-time spectral accuracy enables better accuracy at extrema away from the shock, and enables better accuracy at capturing the shock strength at a shock with higher order polynomials in both space and time compared to lower order

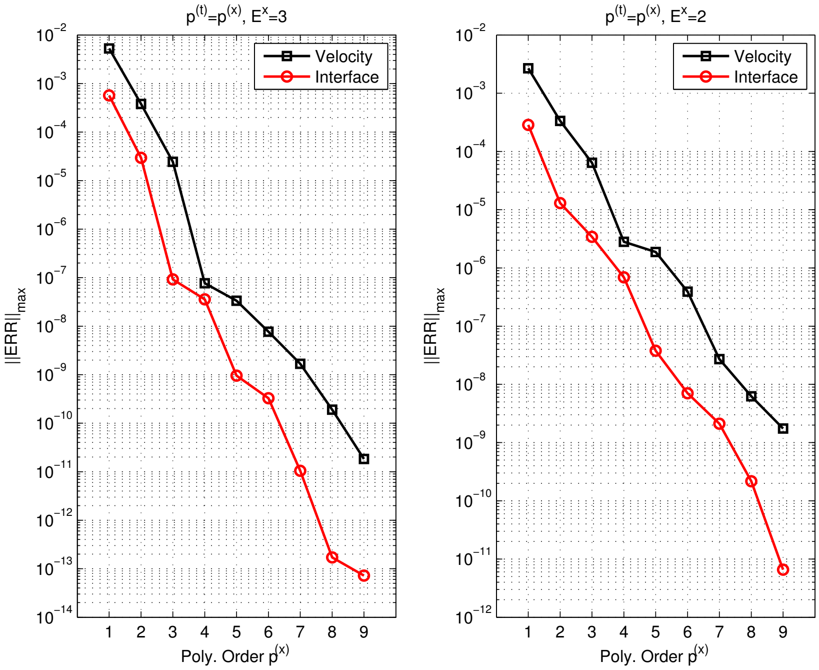

Figure 15: Errors in the shock velocity and shock location as a function of polynomial order $p^{(x)}$. the polynomial order in the temporal direction, $p^{(t)}$, is chosen to be the same as $p^{(x)}$. The number of elements are chosen to be $E^{(x)} = 3$ (on the left) and $E^{(x)} = 2$ (on the right). The simulation is computed over the time $t = 0$ to $t = 0.2$. The ratio of the time step to the spatial element size is $\Delta/h = 0.06$ for $E^{(x)} = 3$ and $\Delta/h = 0.04$ for $E^{(x)} = 2$.

polynomials. Simultaneously, Gibbs oscillations are suppressed by the combination of higher order polynomials and the present shock capturing scheme. Such features are demonstrated for both a scalar hyperbolic linear problem with higher order polynomials, e.g., $(p^{(x)}, p^{(t)}) = (10, 7)$ (Fig. 8) and the inviscid Burgers' equation with higher order polynomials, e.g., $(p^{(x)}, p^{(t)}) = (13, 7)$ (Fig. 14), respectively. The sensitivity of the number of Picard iterations as a function of the temporal order, with different ratios of the time step to

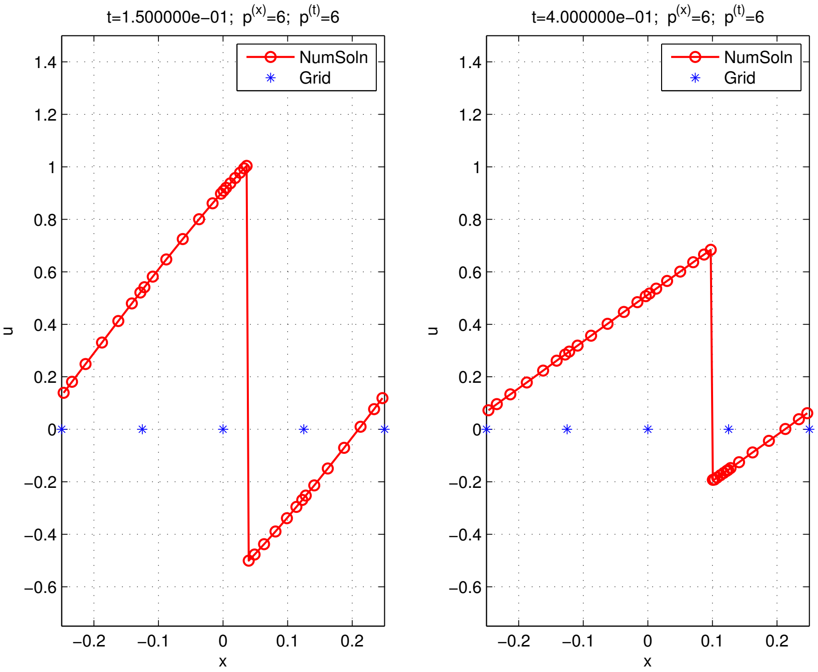

Figure 16: A plot of the numerical solution at different times. The ratio of the time step to the spatial element size is $\Delta t/h = 0.08$, and $E^{(x)} = 4$. The polynomial orders are set to be $(p^{(x)}, p^{(t)}) = (6, 6)$.

the spatial element size, is discussed. It has been found that increasing the temporal order reduces the number of Picard iterations in the nonlinear case (Table 1) and eliminates oscillations in the linear case (Figs. 6 and 7).

The spectral accuracy of the shock speed and location is demonstrated for the shock tracking method. Also, the space-time spectral accuracy of the solution on either side of the discontinuity is preserved. A small cut cell may be generated in the shock tracking process, which does not affect the spectral accuracy. In addition, higher order approximations, e.g., $(p^x, p^{(t)}) = (6, 6)$ and $(p^x, p^{(t)}) = (8, 5)$, behave well for the case in which a small cell

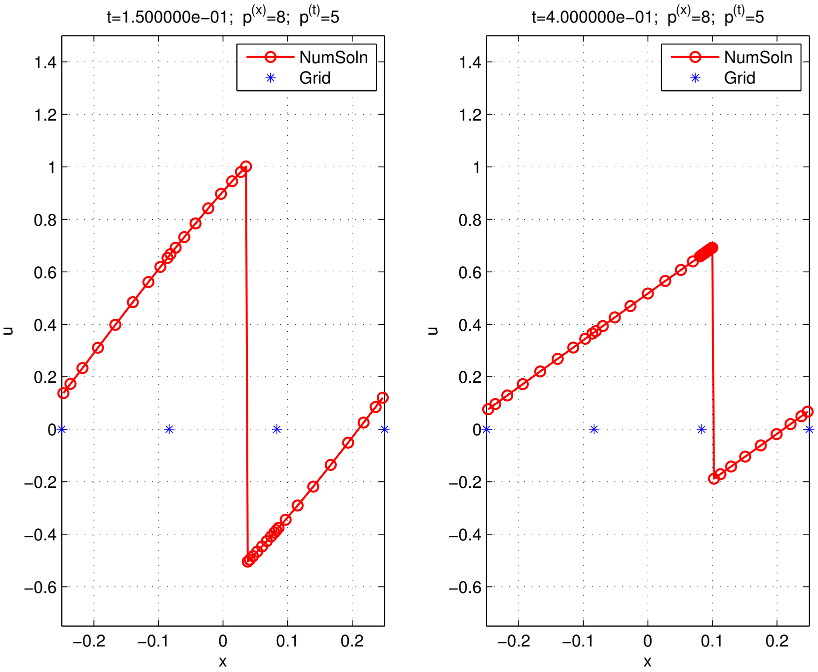

Figure 17: A plot of the numerical solution at different times. The ratio of the time step to the spatial element size is $\Delta t/h = 0.06$, and $E^{(x)} = 3$. The polynomial orders are set to be $(p^{(x)}, p^{(t)}) = (8, 5)$.

is generated in the tracking process. However, a drawback of the tracking method is the time step restriction caused by a tiny cut cell. A tiny cut cell can result in the collision of characteristics for the nonlinear problem; this problem is manifested in the Picard iteration not converging unless the time step is sufficiently refined. The ratio of the time step to the spatial element size, $\Delta t/\min_K\{h_K\}$ must be restricted depending on $\max\{\partial f(u)/\partial u\}$ if a tiny cut cell is generated in one space-time slab. An alternate to reduce Δt is to adjust the polynomial order used in a given element. The capability for subelements in a cut-cell to have a low order approximation that is distinct from its neighbors is the topic of future research.

The present method can be extended to higher spatial dimensions, however, an alternative faster direct solver methodology, such as Martinsson [44], needs to be explored for the resulting large sparse linear system (27) obtained in each Picard iteration. This too will be the subject of future study.

Acknowledgments

This work was partially supported by the National Science Foundation under contract DMS 1418983.

References

References

[1] P. Persson and J. Peraire. Sub-Cell Shock Capturing for Discontinuous Galerkin Methods. *AIAA Aerospace Sciences Meeting and Exhibit*, 44th, 2006.

[2] Garrett E. Barter and David L. Darmofal. Shock capturing with PDE-based artificial viscosity for DGFEM. I. Formulation. *J. Comput. Phys.*, 229(5):1810–1827, 2010.

[3] J. Reisner, J. Serencsa, and S. Shkoller. A space-time smooth artificial viscosity method for nonlinear conservation laws. *J. Comput. Phys.*, 235:912–933, 2013.

[4] E. Casoni, J. Peraire, and A. Huerta. One-dimensional shock-capturing for high-order discontinuous Galerkin methods. *Internat. J. Numer. Methods Fluids*, 71(6):737–755, 2013.

[5] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes, ii. *Journal of Computational Physics*, 83(1):32 – 78, 1989.

[6] Guang-Shan Jiang and Chi-Wang Shu. Efficient implementation of weighted eno schemes. *Journal of Computational Physics*, 126(1):202 – 228, 1996.

[7] Dinshaw S. Balsara and Chi-Wang Shu. Monotonicity preserving weighted essentially non-oscillatory schemes with increasingly high or-

der of accuracy. *Journal of Computational Physics*, 160(2):405 – 452, 2000.

[8] H.C Yee, N.D Sandham, and M.J Djomehri. Low-dissipative high-order shock-capturing methods using characteristic-based filters. *Journal of Computational Physics*, 150(1):199 – 238, 1999.

[9] Bertil Gustafsson and Pelle Olsson. Fourth-order difference methods for hyperbolic ibvps. *Journal of Computational Physics*, 117(2):300–317, 1995.

[10] Amiram Harten. The artificial compression method for computation of shocks and contact discontinuities. iii. self-adjusting hybrid schemes. *Mathematics of Computation*, 32(142):363–389, 1978.

[11] Yu-Xin Ren, Miao'er Liu, and Hanxin Zhang. A characteristic-wise hybrid compact-weno scheme for solving hyperbolic conservation laws. *Journal of Computational Physics*, 192(2):365 – 386, 2003.

[12] Sergio Pirozzoli. Conservative hybrid compact-weno schemes for shock-turbulence interaction. *Journal of Computational Physics*, 178(1):81 – 117, 2002.

[13] Björn Sjögreen and HC Yee. Multiresolution wavelet based adaptive numerical dissipation control for high order methods. *Journal of Scientific Computing*, 20(2):211–255, 2004.

[14] Margot Gerritsen and Pelle Olsson. Designing an efficient solution strategy for fluid flows: 1. a stable high order finite difference scheme and

sharp shock resolution for the euler equations. *Journal of Computational Physics*, 129(2):245–262, 1996.

[15] Stephane Mallat and Sifen Zhong. Characterization of signals from multiscale edges. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 14(7):710–732, 1992.

[16] Ingrid Daubechies. *Ten lectures on wavelets, CBMS-NSF regional conference series in applied mathematics.*, volume 61. SIAM, 1992.

[17] Stéphane Mallat. *A wavelet tour of signal processing.* Academic press, 1999.

[18] Hesam Abbassi, Farzad Mashayek, and Gustaaf B. Jacobs. Shock capturing with entropy-based artificial viscosity for staggered grid discontinuous spectral element method. *Computers and Fluids*, 98:152 – 163, 2014. 12th USNCCM mini-symposium of High-Order Methods for Computational Fluid Dynamics - A special issue dedicated to the 80th birthday of Professor Antony Jameson.

[19] Karthikeyan Duraisamy, James D Baeder, and Jian-Guo Liu. Concepts and application of time-limiters to high resolution schemes. *Journal of scientific computing*, 19(1-3):139–162, 2003.

[20] A Suresh and HT Huynh. Accurate monotonicity-preserving schemes with runge–kutta time stepping. *Journal of Computational Physics*, 136(1):83–99, 1997.

[21] H. Touil, M. Y. Hussaini, and M. Sussman. Tracking discontinuities in

hyperbolic conservation laws with spectral accuracy. *J. Comput. Phys.*, 225(2):1810–1826, 2007.

[22] B. Bernardo Cockburn, George Karniadakis, and Chi-Wang Shu, editors. *Discontinuous Galerkin methods : theory, computation, and applications.* Lecture notes in computational science and engineering. Springer, Berlin, New York, 2000.

[23] Claudio Canuto, M. Yousuff Hussaini, Alfio Quarteroni, and Thomas A. Zang. *Spectral methods: Fundamentals in single domains.* Berlin, Springer, 2006.

[24] Jan S. Hesthaven and Tim Warburton. *Nodal discontinuous Galerkin methods: Algorithms, Analysis, and Applications*, volume 54 of *Texts in Applied Mathematics*. Springer, New York, 2008.

[25] David A. Kopriva. *Implementing spectral methods for partial differential equations: Algorithms for Scientists and Engineers.* Scientific Computation. Springer, Berlin, 2009.

[26] George Em Karniadakis and Spencer J. Sherwin. *Spectral/hp element methods for CFD.* Numerical Mathematics and Scientific Computation. Oxford University Press, Oxford, second edition, 2013.

[27] J. J. Sudirham, J. J. W. van der Vegt, and R. M. J. van Damme. Space-time discontinuous Galerkin method for advection-diffusion problems on time-dependent domains. *Appl. Numer. Math.*, 56(12):1491–1518, 2006.

[28] 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, 2006.

[29] J. J. W. van der Vegt and J. J. Sudirham. A space-time discontinuous Galerkin method for the time-dependent Oseen equations. *Appl. Numer. Math.*, 58(12):1892–1917, 2008.

[30] W. E. H. Sollie, O. Bokhove, and J. J. W. van der Vegt. Space-time discontinuous Galerkin finite element method for two-fluid flows. *J. Comput. Phys.*, 230(3):789–817, 2011.

[31] 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, 2013.

[32] J. J. W. van der Vegt and H. van der Ven. Space-time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows. I. General formulation. *J. Comput. Phys.*, 182(2):546–585, 2002.

[33] F. Lörcher, G. Gassner, and C.-D. Munz. A discontinuous Galerkin scheme based on a space-time expansion. I. Inviscid compressible flow in one space dimension. *J. Sci. Comput.*, 32(2):175–199, 2007.

[34] Sander Rhebergen and Bernardo Cockburn. A space-time hybridizable discontinuous Galerkin method for incompressible flows on deforming domains. *J. Comput. Phys.*, 231(11):4185–4204, 2012.

[35] Chaoxu Pei, Mark Sussman, and Yousuff M. Hussaini. A space-time discontinuous galerkin spectral element method for the stefan problem. Submitted to J. Sci. Comput., 2015.

[36] Michele Benzi and Maxim A. Olshanskii. An augmented Lagrangian-based approach to the Oseen problem. *SIAM J. Sci. Comput.*, 28(6):2095–2113, 2006.

[37] Bernardo Cockburn and Chi-Wang Shu. The local discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM J. Numer. Anal.*, 35(6):2440–2463, 1998.

[38] Douglas N. Arnold, Franco Brezzi, Bernardo Cockburn, and L. Donatella Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM J. Numer. Anal.*, 39(5):1749–1779, 2001.

[39] Bernardo Cockburn, Guido Kanschat, Ilaria Perugia, and Dominik Schötzau. Superconvergence of the local discontinuous Galerkin method for elliptic problems on Cartesian grids. *SIAM J. Numer. Anal.*, 39(1):264–285 (electronic), 2001.

[40] Catherine Mavriplis. Adaptive mesh strategies for the spectral element method. *Comput. Methods Appl. Mech. Engrg.*, 116(1-4):77–86, 1994. ICOSAHOM'92 (Montpellier, 1992).

[41] Mark Sussman, Peter Smereka, and Stanley Osher. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*, 114(1):146 – 159, 1994.

[42] Mark Sussman and M. Y. Hussaini. A discontinuous spectral element method for the level set equation. *J. Sci. Comput.*, 19(1-3):479–500, 2003. Special issue in honor of the sixtieth birthday of Stanley Osher.

[43] David Furbish, M. Yussuf Hussaini, François-Xavier Le Dimet, Pierre Ngnepieba, and Yonghui Wu. On discretization error and its control in variational data assimilation. *Tellus A*, 60(5):979–991, 2008.

[44] P. G. Martinsson. A direct solver for variable coefficient elliptic PDEs discretized via a composite spectral collocation method. *J. Comput. Phys.*, 242:460–479, 2013.

$E^{(x)}$	$p^{(x)}$	$p^{(t)}$	$\Delta t/h$	Number of iterations
8	8	1	0.08	65
8	8	2	0.08	47
8	8	3	0.08	33
8	8	4	0.08	25
8	8	5	0.08	24
8	8	6	0.08	22
8	8	7	0.08	19
8	8	1	0.2	$> N_{max}$ $(*)$
8	8	2	0.2	$> N_{max}$ $(*)$
8	8	3	0.2	124
8	8	4	0.2	51
8	8	5	0.2	46
8	8	6	0.2	40
8	8	7	0.2	33

Table 1: Dependence of the number of Picard iterations on the time step and the temporal order. The criterion for convergence of the Picard iteration is $tol = 10^{-13}$. The maximum iteration number, N_{max}, is set to be 1000.