RICE UNIVERSITY

Entropy Stable Discontinuous Galerkin-Fourier Methods

By

Yimin Lin

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Master of Arts

APPROVED, THESIS COMMITTEE

Jesse Chan
Assistant Professor of Computational and Applied Mathematics

Beatrice Riviere
Noah Harding Chair and Professor of Computational and Applied Mathematics

Matthias Heinkenschloss
Noah Harding Chair and Professor of Computational and Applied Mathematics

HOUSTON, TEXAS
September 2020
Abstract

Entropy Stable Discontinuous Galerkin-Fourier Methods

by

Yimin Lin

Entropy stable discontinuous Galerkin methods for nonlinear conservation laws replicate an entropy inequality at semi-discrete level. The construction of such methods depends on summation-by-parts (SBP) operators and flux differencing using entropy conservative finite volume fluxes. In this work, we propose a discontinuous Galerkin-Fourier method for systems of nonlinear conservation laws, which is suitable for simulating flows with spanwise homogeneous geometries. The resulting method is semi-discretely entropy conservative or entropy stable. Computational efficiency is achieved by GPU acceleration using a two-kernel splitting. Numerical experiments in 3D confirm the stability and accuracy of the proposed method.
Contents

1 Introduction 1

2 Literature Review 5
  2.1 High order methods .............................................. 6
    2.1.1 Pseudo-spectral methods .................................. 7
    2.1.2 Spectral/hp element methods .............................. 7
    2.1.3 Discontinuous Galerkin methods ........................... 8
  2.2 Entropy Stable Discontinuous Galerkin methods ................. 9

3 Entropy Stable Discontinuous Galerkin-Fourier Method 13
  3.1 Entropy stability for systems of nonlinear conservation laws 13
  3.2 Discretely Entropy stable DG-Fourier method .................. 15
    3.2.1 Mathematical assumptions and notaions ................. 15
    3.2.2 Choice of reference element, approximation space and quadrature rules 16
    3.2.3 Reference element operators ................................ 19
    3.2.4 Flux differencing and entropy projection ............... 27
    3.2.5 An entropy stable DG-Fourier method on the reference element . 29
    3.2.6 An entropy stable DG-Fourier method on mapped elements . 32
  3.3 GPU acceleration ................................................... 36
  3.4 Numerical Experiments: 3D Compressible Euler Equations .... 39
    3.4.1 Vortex Propagation and Sine wave ....................... 41
    3.4.2 Doubly periodic shear layers ............................ 44

4 Conclusion 50

Bibliography 51
Acknowledgments

I would like to thank my advisor Professor Jesse Chan for helping me with every aspect of this thesis project. As a teacher, he shares a lot of knowledge and insights about computational mathematics with me. As an advisor, he is responsible and genuinely cares about my work. His enthusiasm for the subject always reminds me of why I want to pursue the degree. I cannot wait to learn and work with him more in the next few years.

I would like to thank my mentors during my undergraduate at Berkeley. Michael Franco introduced me to the field of computational mathematics. The first subject we worked together is pseudo-spectral methods, which play an essential role in this thesis coincidently. Graduated from Rice University, he introduced me to CAAM, and I enjoy being in this department. Professor Per-Olof Persson showed me what research in computational mathematics is and referred me to Professor Jesse Chan. Professor Rainer Sachs taught me how to become a good researcher. His passion for research is part of the reason I was determined to continue studying.

I would like to thank my committee members Professor Heinkenschloss and Professor Rivière for their suggestions on my master thesis. I would like to thank Kaihang for fruitful discussions we had together. I would like to thank my father, my mother, Xin Yang, and my other family members for continuous support throughout the years. They always support and encourage me to pursue what I love. Last but not least, I would like to thank Hedy for such a fantastic first year in Houston together.
Chapter 1

Introduction

Computational fluid dynamics (CFD) involves the simulation of complex fluid flows with transonic and turbulent features, and applies to a wide range of engineering problems, including aerospace engineering, biological science, and weather simulation. CFD simulations increasingly demand higher accuracy and improved efficiency. High order numerical methods provide a solution due to their increased accuracy per degree of freedom compared to low order methods (for sufficiently regular solutions). They can achieve engineering accuracy (e.g., below 5% error) more efficiently [1].

However, because high order schemes do not benefit from the stabilizing effects of numerical dissipation present in low order schemes, they need extra stabilization procedures. Typical stabilization procedures require heuristic tuning of parameters [2, 3, 4]. As a result, high order discretizations with stabilization procedures are not robust. For linear wave propagation problems, semi-discretely energy stable numerical methods can be constructed
[5], and high order methods can be applied in a stable manner while keeping the discretizations robust. However, it may not be possible to derive an energy inequality for nonlinear conservation laws. Instead, the concept of a mathematical entropy generalizes the notion of the “energy” to nonlinear systems. Entropy conditions also provide a way to characterize physically realizable solutions when shocks occur [6, 7].

Proofs of entropy stability have traditionally assumed exact integration [8]. However, exact integration for nonlinear conservation laws is generally unachievable. Inexact integration results in the loss of the chain rule, which causes the loss of discrete entropy stability in numerical discretizations. As a result, high order methods tend to blow up for solutions with shocks or underresolved features.

In [9], Tadmor provides a way to construct discretely entropy stable finite volume methods via entropy conservative fluxes, and entropy conservative formulations were extended to high order finite volume discretizations in [10, 11]. Entropy conservative fluxes were later combined with SBP (Summation-By-Parts) operators, which replicate integration by parts at a discrete level through a volume flux differencing technique [12, 13]. This combination enabled the construction of discretely entropy conservative high order discontinuous Galerkin (DG) discretizations [14].

Entropy stable discontinuous Galerkin methods were initially proposed on tensor product elements using a spectral collocation method based on Gauss-Lobatto quadrature points [13, 14]. The formulation was later extended to simplicial elements using generalized Gauss-Lobatto type quadrature rules [15, 16]. Entropy stable DG methods were then extended to
CHAPTER 1. INTRODUCTION

general quadrature rules through over-integration on boundaries and an entropy projection step [17, 18, 19, 20].

Although high order entropy stable DG formulations inherit the flexibility and generality of finite element methods, the computational cost associated with entropy stable formulation is high, especially in 3D and on unstructured tetrahedral meshes. The main bottleneck is the on-the-fly evaluation of a Hadamard product, which requires $O((N_q+N_{f,q})^2)$ flux evaluations, where $N_q+N_{f,q}$ is the total number of volume and face quadrature points.

If the domain is periodic and homogeneous in one direction, a natural approach is to approximate the solution on the spanwise direction using a Fourier basis. This work proposes an entropy stable DG-Fourier method suitable for simulating compressible flows in such geometries [21]. In the spanwise direction, we used a Fourier pseudo-spectral discretization, which preserves discrete entropy stability through properties of the spectral differentiation matrix under the periodic trapezoid rule. The “spectral accuracy” of the Fourier basis [22] implies that only a small number of Fourier modes are necessary for high order accurate simulations, which reduces the computational cost of a full 3D simulation substantially.

The outline of this thesis is as follows: Chapter 2 will review the literature relevant to the proposed method. Chapter 3 will provide details on the formulation of Entropy Stable (ES) DG-Fourier methods, broken into four sections. The first section will briefly review the entropy stability theory for nonlinear conservation laws, and the theory forms theoretical foundations for discretely entropy stable schemes. The second section will discuss the ingredients necessary for the proposed method and describe the DG formulation. We
briefly discuss the implementation and GPU acceleration of the proposed method in Section 3. Finally, we present some numerical experiments illustrating convergence and robustness of the proposed method in 3D. In Chapter 4, we summarize our contributions.
Chapter 2

Literature Review

The entropy stable discontinuous Galerkin-Fourier method proposed in this work is a combination of a 2D high order discontinuous Galerkin method and a Fourier pseudo-spectral method. This method achieves discrete entropy stability through summation-by-parts properties of discrete operators and a technique referred to as flux-differencing. This chapter will discuss the literature relevant to the proposed scheme and entropy stable discontinuous Galerkin methods. Section 2.1 will review high order discontinuous Galerkin method and pseudo-spectral methods. A summary of the development of entropy stable quadrature-based DG methods and their key ingredients are highlighted in Section 2.2.
2.1 High order methods

In the CFD community, high order methods generally refer to numerical methods with the order of accuracy three or higher. Such methods range from spectral methods to high order discontinuous Galerkin methods. High order methods are more accurate than low order methods per degree of freedom for sufficiently regular solutions so that they can achieve engineering accuracy (e.g., 5% error) more efficiently [1]. Compared to low order schemes, high order methods possess low numerical dissipation and dispersion, which contributes to better accuracy in high-fidelity long-time simulations of vorticular, turbulent, and under-resolved flows [23].

However, because high order schemes do not benefit from the stabilizing effects of numerical dissipation present in low order schemes, various stabilization procedures seek to mitigate the stability issues intrinsic to high order methods. For example, artificial viscosity [2], slope limiting [3], and filtering [4] are typical procedures used to stabilize high order methods. However, most require heuristic tuning of stabilization parameters, which may in turn change based on physical and discretization parameters. As a result, high order discretizations with stabilization procedures may not be robust and can still fail. The recent development of entropy stable discontinuous Galerkin methods is a step forward towards provably stable high order simulations for nonlinear conservation laws.
2.1.1 Pseudo-spectral methods

Pseudo-spectral methods, introduced by Orszag, Kreiss, Gottlieb, and their coauthors [24], can be viewed as the discrete version of Fourier methods for solving partial differential equations. In contrast to traditional finite difference methods, where derivatives are evaluated locally (e.g., the stencil only depends on neighboring nodes) stencils for pseudo-spectral methods are global. One of the most appealing features of the pseudo-spectral method is so-called “spectral accuracy”, where for sufficiently regular solutions, the approximation error converges exponentially fast as the number of degrees of freedom increases [22]. Note that in general, a pseudo-spectral method can also refer to spectral collocation schemes, as well as methods such as Galerkin spectral methods and Tau spectral methods, which are described in detail in [25]. In this work, we use pseudo-spectral methods to refer to Fourier pseudo-spectral methods.

2.1.2 Spectral/hp element methods

Compared with Fourier pseudo-spectral methods, spectral/hp element methods use a polynomial basis instead of a Fourier basis for approximation. They are built on a standard finite element framework with polynomial degree $p$ and element size $h$. We can view spectral/hp element methods as applying a “spectral-like” approach over each element and enforcing $C^0$ continuity across interfaces [26]. One of the limitations of Fourier pseudo-spectral methods is that the geometry of the computational domain needs to be periodic. In contrast,
spectral/hp element methods retain the geometric flexibility of finite element discretizations while enabling explicit time stepping through mass lumping. Compared with the standard $h$-version of finite element methods, spectral/hp element methods achieve spectral accuracy for smooth solutions by increasing the polynomial degree [27]. A spectral element-Fourier method was proposed in [21], where a spectral element discretization was extended to spanwise periodic geometries using the Fourier basis. However, the spectral element method\(^1\) is only limited to tensor product geometries (quadrilateral or hexahedral elements), and automatic generation of unstructured hexahedral mesh is challenging on complex geometries [28].

### 2.1.3 Discontinuous Galerkin methods

The first discontinuous Galerkin method was introduced by Reed and Hill in 1973 [29]. Since then, discontinuous Galerkin methods have become popular in computational fluid dynamics. DG methods can be interpreted as spectral element methods which do not enforce strict $C^0$ continuity across interfaces. Instead, neighboring elements are coupled together through numerical fluxes, as in finite volume methods. DG methods share most of the advantages of spectral element methods, including a natural parallelizable structure due to the locality of most operations [30]. DG methods also simplify the implementation of adaptive strategies, since the refinement and unrefinement of grids and polynomial degrees

---

\(^1\)Note that the spectral element method here refers to the mass lumped scheme based on Gauss-Lobatto points, and not to general spectral/hp element methods.
can be performed locally over each element [31, 32]. Last but not least, appropriate choices of numerical fluxes and slope limiters in DG methods make them suitable for convection dominated problems and solutions with shocks and discontinuities [4].

For linear problems, it is common to compute numerical fluxes by solving the Riemann problem analytically [33]. However, for nonlinear conservation laws the Riemann problem is more complex and can only be numerically approximated. Furthermore, discontinuities in the solutions together with Gibbs phenomenon near shocks in high-order approximations can result in numerical densities and pressure which are small or even negative, resulting in non-physical solutions. Slope limiters, which are extensively used in finite volume methods, can also be used to remove oscillations in DG methods. However, this approach will restrict DG methods to low order accuracy where limiters are applied [4]. Moreover, not all slope limiters guarantee stability for simulations.

2.2 Entropy Stable Discontinuous Galerkin methods

As discussed in the previous section, it is challenging to solve nonlinear conservation laws in the presence of shocks with high order methods. High order methods can develop oscillations, which eventually blow up and cause the simulation to crash. For linear wave propagation problems, semi-discretely energy stable numerical methods can be constructed [5], and high order methods can be applied in a stable manner while keeping the discretizations robust. However, for nonlinear conservation laws, it may not be possible to derive an
energy inequality. Instead, the concept of a mathematical entropy generalizes the notion of
the “energy” to nonlinear systems. Entropy conditions also provide a way to characterize
physically realizable solutions when shocks occur [6, 7].

Solutions to inviscid conservation laws are defined as limit solutions to viscous conserva-
tion laws as viscosity goes to zero. It can be shown that the vanishing viscosity solution
satisfies an entropy inequality. This motivates the development of numerical methods that
satisfy a discrete entropy inequality.

The proof of the entropy inequality in the continuous setting relies on the chain rule [8].
Due to inexactness of quadrature rules, the chain rule does not hold at the discrete level, and
thus the proof of the entropy inequality in the continuous setting cannot be reproduced after
discretization. For low order methods, this loss of stability is not an issue in practice since the
large numerical dissipation stabilizes the method. However, the low numerical dissipation of
high order methods makes them more sensitive to instability. Therefore, restoring a provable
entropy inequality at the discrete level will make high order methods more robust.

First and second-order accurate entropy stable finite volume methods based on entropy
conservative fluxes were introduced by Tadmor in [9] and provide the framework for entropy
stable methods in general. A general procedure to construct arbitrarily high order accurate
entropy conservative fluxes is given in [10], which forms the building blocks of arbitrarily

Another approach to constructing high-order entropy stable schemes is through summation-
by-parts (SBP) finite difference operators with simultaneous approximation terms (SAT),
commonly referred to as SBP-SAT methods. SBP operators satisfy a discrete version of integration by parts, even under inexact quadrature rules. SATs refer to boundary and interface terms used to couple neighboring elements together and impose boundary conditions weakly [34]. A characterization of finite-difference SBP operators via quadrature rules is given in [35]. In [12], general procedures for deriving SBP operators with SATs are illustrated for linear PDEs.

A typical SBP operator includes a mass matrix (also known as a norm matrix) and a differentiation matrix. When the mass matrix is diagonal, we refer to the differencing operators as diagonal-norm SBP operators. Using such operators, Fisher and Carpenter [13] construct high-order accurate discretizations of nonlinear flux derivatives, such that the resulting scheme is discretely entropy conservative. Their technique is commonly referred to as flux differencing and can be interpreted as a subcell based high-order finite volume formulation. Diagonal norm SBP-SAT discretizations were connected to discontinuous Galerkin collocation spectral element methods using Gauss-Lobatto points by Gassner [36]. Carpenter et al. then utilized this connection to construct discretely entropy stable spectral collocation schemes on tensor product elements [14].

Entropy stable DG methods were then generalized from tensor product elements to simplex elements in [15, 16, 37, 38]. An entropy stable DG-like method was constructed using special quadrature rules with collocated surface quadrature points on triangles to mimic the structure of SBP operators on tensor product elements [15]. Another approach using a separate set of face cubature nodes with an extension to curvilinear meshes was proposed
CHAPTER 2. LITERATURE REVIEW

in [16]. Entropy stable SBP discretizations have also been applied to the continuous framework, which eliminates interface penalties and reduces the computational cost of residual evaluations compared to entropy stable SBP-SAT discretizations [38].

However, all of the approaches above rely on diagonal-norm SBP operators. Diagonal norm matrices correspond to mass lumping using quadrature. Mass lumping can retain high order accuracy on tensor product elements, but due to under-integration errors, it tends to lose high order accuracy on simplex elements [28]. Under-integration errors can be avoided using more accurate quadrature rules, but the resulting norm matrices become dense and the approaches described above are not applicable. Using over-integration with entropy projection during flux evaluation, Chan proposed a modal entropy stable DG formulation [17] that extends to general quadrature rules. Hybridized SBP operators introduced in [17] enable efficient interface coupling compared to previous approaches in [16], which required non-compact interface terms between elements coupling. The modal entropy stable DG formulation was then extended to curvilinear meshes using weight-adjusted mass matrices in [18]. Finally, an alternative skew-symmetric modal entropy stable DG formulation which relaxed conditions on quadrature for discrete entropy stability was presented in [20]. These papers form the foundations of our work, and the proposed method is an extension of Chan’s work to 3D domains through Fourier extrusion.
Chapter 3

Entropy Stable Discontinuous
Galerkin-Fourier Method

3.1 Entropy stability for systems of nonlinear conservation laws

In this work, we focus on the numerical approximation of solutions to systems of nonlinear conservation laws in $d$ dimensions with $n$ variables

$$\frac{\partial u}{\partial t} + \sum_{i=1}^{d} \frac{\partial f_i(u)}{\partial x_i} = 0,$$

$$u(x,t) = (u_1(x,t), \ldots, u_n(x,t)), \quad f_i : \mathbb{R}^n \rightarrow \mathbb{R}^n$$
where \( u \) denotes the conservative variables and \( f_i \) denote nonlinear fluxes, which are continuous functions of the conservative variables. In addition, we focus on nonlinear conservation laws with a convex scalar entropy \( \eta(u) \) which satisfies

\[
\eta''(u) A_i(u) = (\eta''(u) A_i(u))^T, \quad (A_i(u))_{mn} = \left( \frac{\partial f_i(u)}{\partial u_n} \right)_m
\]

where \( A_i(u) \) are Jacobian matrices for the fluxes \( f_i(u) \). Then, we can define the entropy variables \( v(u) \) as the gradient of the entropy \( \eta(u) \) with respect to the conservative variables \( u \)

\[
v(u) = \frac{\partial \eta(u)}{\partial u}
\]

For \( \eta(u) \) convex, the mapping between conservative and entropy variables \( v(u) \) is invertible, and we denote its inverse by \( u(v) \). By multiplying (3.1) on the left by entropy variables \( v(u)^T \) and applying the chain rule, (3.1) admits an extension

\[
\frac{\partial \eta(u)}{\partial t} + \sum_{i=1}^{d} \frac{\partial F_i(u)}{\partial x_i} = 0, \quad \left( \frac{\partial F_i(u)}{\partial u} \right)^T = v^T A_i(u)
\]

where the scalar valued \( F_i \) are referred to as entropy flux functions.

As a consequence of the Lax entropy condition \([7]\), vanishing viscosity limit solutions to (3.1) satisfy an entropy inequality

\[
\frac{\partial \eta(u)}{\partial t} + \sum_{i=1}^{d} \frac{\partial F_i(u)}{\partial x_i} \leq 0
\]

Integrating (3.6) over the domain \( \Omega \) with boundary \( \partial \Omega \) with outward unit normals \( n = (n_1, \ldots, n_d) \) and applying integration by parts, we get an integrated form of this entropy inequality.
CHAPTER 3. ENTROPY STABLE DISCONTINUOUS GALERKIN-FOURIER METHOD

inequality

\[ \int_{\Omega} \frac{\partial \eta(u)}{\partial t} + \sum_{i=1}^{d} \int_{\partial \Omega} n_i(v^T f_i(u) - \psi_i(u)) \leq 0, \quad \psi_i(u) = v^T f_i(u) - F_i(u) \] (3.7)

where \( \psi_i(u) \) are scalar-valued functions referred to as entropy potentials.

The entropy inequalities (3.5),(3.6),(3.7) imply the dissipation of entropy \( \eta(u) \). Note that the proof of this inequality relies on the chain rule and integration by parts, which holds at the continuous level. However, the chain rule generally does not hold at the discrete level for numerical discretizations. This work proposes high order DG-Fourier methods which circumvent the chain rule and satisfy a semi-discrete version of the entropy inequality (3.7).

3.2 Discretely Entropy stable DG-Fourier method

3.2.1 Mathematical assumptions and notations

We denote the computational domain by \( \Omega \subseteq \mathbb{R}^d \). In this work, we assume that the geometry is homogeneous and periodic in one direction. This allows us to approximate the solution in the homogeneous direction using a Fourier basis. In this work, we focus on 3D domains, though the theory applies in \( d \) dimensions. Without loss of generality, the homogeneous direction will be ordered as the last dimension. We will also refer to the homogeneous direction as the spanwise direction.

For notation, vector and matrix quantities are denoted using lower and upper case bold font respectively, for example \( \mathbf{A} \) and \( \mathbf{u} \). Spatially discrete quantities are written in bold
sans serif font, for example $x$. For clarity, continuous real functions evaluated over spatially
discrete quantities are taken to mean point-wise evaluations. For example,

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad u : \mathbb{R} \to \mathbb{R}, \quad u(x) = \begin{bmatrix} u(x_1) \\ \vdots \\ u(x_n) \end{bmatrix}$$

The $L^2$ inner products over an element $D$ and the boundary of element $\partial D$ is denoted by

$$(u, v)_D = \int_D u \cdot v \, dx, \quad \langle u, v \rangle_{\partial D} = \int_{\partial D} u \cdot v \, dx$$

For systems of conservation laws, since there are multiple scalar components, when $A \in \mathbb{R}^{m \times m}, u \in \mathbb{R}^{mn}$, we will regard $Au$ as the kronecker product $(A \otimes I_n)u$. In this paper, we will adapt the lexicographical ordering $(x, y, z, \ldots)$ of nodes and basis functions, so that a multi-index is replaced by a lexicographical single index for clarity of notation.

### 3.2.2 Choice of reference element, approximation space and quadrature rules

#### 3.2.2.1 Reference element

We start by introducing a $d$-dimensional reference element $\widehat{D} = \widehat{D}^P \times \widehat{D}^F$, which is the direct product of a $d - 1$ dimensional reference element $\widehat{D}^P$ and a 1D reference element $\widehat{D}^F$ in spanwise direction. In this work, $\widehat{D}^P$ is simplicial (interval or triangle), though it can also
be a tensor product element (quadrilateral or hexahedron). \( \widehat{D}^F \) is the interval \([0, 2\pi]\). Then, the reference element \( \widehat{D} \) is the extrusion of \( \widehat{D}^P \) along the spanwise direction, for example a rectangle when \( d = 2 \) and a wedge or hexahedron when \( d = 3 \), as shown in Figure 3.1.

### 3.2.2.2 Approximation space

We define the approximation space on \( \widehat{D}^P \) as the space of degree \( N^P \) polynomials. Depending on our choice of reference element \( \widehat{D}^P \), the polynomial approximation space is defined differently. If \( \widehat{D}^P \) is simplicial, the natural polynomial space consists of polynomials with total monomial degrees less than or equal to \( N^P \)

\[
P^{N^P}(\widehat{D}^P) = \left\{ p_{k_1, \ldots, k_{d-1}}(\bar{x}) = \bar{x}_1^{k_1} \cdots \bar{x}_{d-1}^{k_{d-1}}, \quad \bar{x} \in \widehat{D}^P, \quad 0 \leq \sum_{i=1}^{d-1} k_i \leq N^P \right\}
\]

If \( \widehat{D}^P \) is a tensor product element, the natural polynomial space consists of polynomials with highest monomial degree less than or equal to \( N^P \). \[20\]

In the spanwise direction, assuming periodicity, it is natural to define an approximation space on \( \widehat{D}^F \) using Fourier basis. By the boundness and periodicity of \( \widehat{D}^F \), given \( N^F \) Fourier
modes with equally spaced nodes $\hat{z}_n = \frac{2\pi n}{N_F}, k = 1, \ldots, N_F$, the periodic sinc basis (or Fourier nodal basis) is defined as \cite{22}

$$S_i(\hat{z}) = S(\hat{z} - \hat{z}_i), \quad i = 1, \ldots, N_F, \quad S(\hat{z}) = \frac{\sin(N_F \hat{z})}{N_F \tan(\hat{z})}$$ (3.8)

The approximation space on $\hat{D}_F$ is then defined as

$$F^{N_F}(\hat{D}_F) = \left\{ S_i(\hat{z}), \quad \hat{z} \in \hat{D}_F, \quad i = 1, \ldots, N_F \right\}$$ (3.9)

The approximation space on the reference element $\hat{D}$ is then defined as the tensor product of approximation spaces on $\hat{D}_P$ and $\hat{D}_F$.

$$V_h(\hat{D}) = P^{N_P}(\hat{D}_P) \otimes F^{N_F}(\hat{D}_F) = \{ \phi_i(\hat{x}, \hat{z}) = p_{k_1, \ldots, k_{d-1}}(\hat{x})S_i(\hat{z}), \quad \hat{x} \in \hat{D}_P, \hat{z} \in \hat{D}_F \}$$ (3.10)

We denote the dimension of the above approximation spaces as $N_p = \dim(P^{N_P}(\hat{D}_P))$, $N_F = \dim(F^{N_F}(\hat{D}_F)) = N_F$, and $N_P = \dim(V_h(\hat{D})) = N_P N_F$.

### 3.2.2.3 Quadrature rules

For the reference elements presented above, we also construct quadrature rules on the reference element in a tensor product fashion. On $\hat{D}_P$, we use $\{ (\hat{x}_i, w_i) \}_{i=1}^{N_P}$ and $\{ \hat{x}_i^{f}, w_i^{f} \}_{i=1}^{N_F}$ to denote volume and surface quadrature rules on the reference element. In order to reproduce discrete version of integration by parts

$$\left( \frac{\partial u}{\partial x_n}, v \right)_{\hat{D}_P} = \langle u, v \rangle_{\partial \hat{D}_P} - \left( u, \frac{\partial v}{\partial x_n} \right)_{\hat{D}_P}$$

Since $u, v \in P^{N_P}$, we require volume quadratures exact for degree $2N_P - 1$ polynomials and surface quadrature rule exact for degree $2N_P$ polynomials.
CHAPTER 3. ENTROPY STABLE DISCONTINUOUS GALERKIN-FOURIER METHOD

Next, because we use spectral collocation to approximate the solution in the spanwise direction, we use an interpolatory quadrature rule based on equally spaced nodes \( \{ \tilde{z}_i \}_{i=1}^{N_p^F} \) and a periodic sinc nodal basis. By explicit calculations, it can be shown that this interpolatory quadrature reduces to the periodic trapezoidal rule

\[
\int_0^{2\pi} f(z) \, dz \approx \frac{2\pi}{N_p^F} \sum_{i=1}^{N_p^F} f(\tilde{z}_i), \quad \tilde{z}_i = \frac{2\pi}{N_p^F} i, \quad i = 1, \ldots, N_p^F
\]

with equally spaced quadrature nodes and uniform quadrature weights \( h := w_i^F = \frac{2\pi}{N_p^F} \). It is also well known that, in contrast to the non-periodic case, the trapezoid rule converges exponentially fast for smooth periodic functions [25].

The quadrature rule on the reference element \( \hat{D} \) can then be constructed via tensor product. The volume and surface quadrature rule will be

\[
\{(\hat{x}_i, \tilde{z}_j), hw_{ij}\}_{i=1...N_{Pq}^q, j=1...N_{Fq}^f}, \quad \{(\hat{x}_i^f, \tilde{z}_j), hw_{ij}^f\}_{i=1...N_{Pf}^q, j=1...N_{Ff}^q}
\]

Denote the vector of quadrature nodes on \( \hat{D}_P \) and \( \hat{D}_F \) by \( \hat{x}_q, \hat{x}_q^f \) and \( \tilde{z}_q \), and define \( N_q = N_{Pq}^q N_{Fq}^f, N_{f,q} = N_{Pf}^q N_{Ff}^f \) and number of hybridized quadrature points \( N_{hP}^q = N_{P}^q + N_{Pf}^q, N_{h} = N_{hP}^q N_{F}^f \).

### 3.2.3 Reference element operators

In this work we develop matrix formulations of discontinuous Galerkin methods. In this section, necessary linear operators in our formulation are presented. We assume functions
in approximation spaces in previous sections can be written in terms of modal expansions

\[ u_N(\vec{x}, t) = \sum_{i=1}^{N^P} u_{h,i}(t)p_i(\vec{x}), \quad w_N(\vec{z}, t) = \sum_{i=1}^{N^F} w_{h,i}(t)S_i(\vec{z}) \]

We denote the vector of modal coefficient by \( u_h \) and \( w_h \) respectively. Since we focus on semi-discrete discretizations, we drop the argument \( t \) from \( u_{h,i}(t) \) for simplicity of notation.

### 3.2.3.1 Interpolation and Differentiation

Explicit formulas for interpolation are defined in terms of the Vandermonde matrix. Using the fact that \( \{S_i\}_{i=1}^{N^F} \) is a nodal basis, the interpolation matrix on \( \hat{D}^F \) is an identity matrix. Explicit formulas for quadrature interpolation matrices are then given by

\[
(V^P_q)_{ij} = p_j(\vec{x}_i), \quad (V^F_q)_{ij} = p_j(\vec{x}^f_i), \quad V^F_q = I_{N^F}
\]

Quadrature interpolation matrices map modal coefficients of approximants to their evaluation at quadrature points. In particular, volume and surface quadrature interpolation on \( \hat{D}_P \) and \( \hat{D}_F \) satisfies the relations

\[
V^P_q u_h = u_N(\vec{x}_q), \quad V^P_f u_h = u_N\left(\vec{x}^f_q\right), \quad V^F_q w_h = w_N(\vec{z}_q)
\]

Because \( \varphi_i(\vec{x}, \vec{z}) \) is constructed using a tensor product, the volume and surface quadrature interpolation matrices on reference element \( \hat{D} \) can be expressed as

\[
V_q = V^F_q \otimes V^P_q = I_{N^F} \otimes V^P_q, \quad V_f = V^F_q \otimes V^F_f = I_{N^F} \otimes V^F_f
\]
In addition, we introduce hybridized quadrature interpolation matrices on $\hat{D}^P$ and $\hat{D}$ as

$$
V_h^P = \begin{bmatrix} V_q^P \\ V_f^P \end{bmatrix}, \quad V_h = I_{NP} \otimes V_h^P
$$

(3.12)

which map modal coefficient of approximants to their evaluation at both volume and face quadrature points.

Next, let $D_n^P$ be the differentiation matrix with respect to $n$-th coordinate on $\hat{D}^P$, and let $D^F$ denote the modal differentiation matrix on $\hat{D}^F$ such that

$$
\frac{\partial u_N}{\partial \hat{x}_n} = \sum_{i=1}^{NP} (D_n^P u_h)_{i} \delta_i(\hat{x}), \quad \frac{\partial w_N}{\partial \hat{z}} = \sum_{i=1}^{NP} (D^F w_h)_{i} S_i(\hat{z})
$$

The matrix $D_n^P$ takes modal coefficients of an approximant and outputs modal coefficients of the $n$-th derivative of the approximant. Because the spanwise basis is the nodal sinc basis, $D^F$ is also the nodal differentiation matrix (also known as the spectral differentiation matrix).

An explicit formula for $\hat{D}^F$ can be derived using derivatives of periodic sinc functions [22]

$$
D^F = \begin{bmatrix}
0 & -\frac{1}{2} \cot \frac{h}{2} \\
-\frac{1}{2} \cot \frac{h}{2} & \ddots & \ddots & \frac{1}{2} \cot h \\
\frac{1}{2} \cot h & \ddots & \ddots & -\frac{1}{2} \cot \frac{3h}{2} \\
\vdots & \ddots & \ddots & \ddots \\
\frac{1}{2} \cot \frac{h}{2} & \ddots & \ddots & 0
\end{bmatrix},
$$

(3.13)

from which one can observe the following properties

$$
D^F + (D^F)^T = 0, \quad D^F 1 = 0
$$

(3.14)
We note that the skew-symmetry of $D^F$ is crucial for proving entropy stability in the following section.

Now we extend differentiation operators to the reference element $\hat{D}$ through a Kronecker product. Regarding the accuracy of differentiation operators, $D^n_P$ is $N^n_P$-th order accurate (exactly differentiates polynomial of degree less than or equal to $N^n_P$) by construction, and the spectral differentiation matrix $D^F$ possesses spectral accuracy for smooth solutions.

$$D_i = I_{N^F} \otimes D^n_i, \quad i = 1, \ldots, d - 1, \quad D_d = D^F \otimes I_{N^P}$$ \hfill (3.15)

### 3.2.3.2 Quadrature based operators: mass and projection matrices

For the purpose of our matrix formulation, we define volume and surface quadrature weight matrices as diagonal matrices with quadrature weights as diagonal entries

$$W^P = \begin{bmatrix} w_1 & \cdots & w_{N^P} \end{bmatrix}, \quad W^F = \begin{bmatrix} w^F_1 & \cdots & \end{bmatrix}$$ \hfill (3.16)

$$W^P_f = \begin{bmatrix} W^P \\ W^P_f \end{bmatrix}, \quad W^F = hI_{N^F}. \hfill (3.17)$$

Quadrature weight matrices on $\hat{D}$ can then be written as Kronecker products

$$W = W^F \otimes W^P = hI_{N^F} \otimes W^P, \quad W_f = W^F \otimes W^P_f = hI_{N^F} \otimes W^P_f$$ \hfill (3.18)

$$W_h = W^F \otimes W^P_h = hI_{N^F} \otimes W^P_h$$ \hfill (3.19)
CHAPTER 3. ENTROPY STABLE DISCONTINUOUS GALERKIN-FOURIER METHOD

If integrals are computed using quadrature, then for two functions $u, v$ defined on $\hat{D}^P \cup \partial \hat{D}^P$, we can write the $L^2$ inner product in matrix form as

$$\int_{\hat{D}} uv \approx \sum_{i=1}^{N^P} w_i u(\hat{x}_i) v(\hat{x}_i), \quad \int_{\partial \hat{D}} uv \approx \sum_{i=1}^{N^P} w_i f_i u(\hat{x}_i) v(\hat{x}_i)$$

With the matrix form of quadrature in mind, we next introduce some operators that are defined using inner products in the continuous setting and their discretized counterparts.

Mass matrices on $\hat{D}^P$, $\hat{D}^F$, and $\hat{D}$ are defined respectively as

$$(M^P)_{ij} = \int_{\hat{D}^P} \mathcal{P}_i(\hat{x}) \mathcal{P}_j(\hat{x}) \, d\hat{x} \approx (V^P_q)^T W^P V^P_q$$

$$(M^F)_{ij} = \int_{\hat{D}^F} \cal{S}_i(\hat{z}) \cal{S}_j(\hat{z}) \, d\hat{z} \approx (V^F_q)^T W^F V^F_q = h I_{N^F}$$

$$M = M^F \otimes M^P = h I_{N^F} \otimes M^P$$

It is important to note that $M^P$ is symmetric and positive definite when volume quadratures exact for degree $2N^P$ polynomials [17]. As a result, the mass matrix on reference element $\hat{D}$ is also symmetric and positive definite.

Discretizing continuous $L^2$ projection operators through quadrature yields the following projection matrices on $\hat{D}^P$, $\hat{D}^F$, and $\hat{D}$

$$P^P_q = (M^P)^{-1}(V^P_q)^T W^P, \quad P^F_q = (M^F)^{-1}(V^F_q)^T W^F = I_{N^F}$$

$$P_q = P^F_q \otimes P^P_q = I_{N^F} \otimes P^P_q$$

Furthermore, we introduce an extrapolation operator $E^P = V^P_q P^P_q$ on $\hat{D}^P$, which maps volume quadrature values to face quadrature values through $L^2$ projection and interpolation to
face quadrature points. We also introduce a boundary integration matrix $B_i$, which is a discretization of $\int_{\partial D^P} u w \tilde{n}_i \approx u_f^T B_i v_f$

$$B_i^P = W_f^P \text{diag}(\tilde{n}_i), \quad B_i = h I_{N^P} \otimes B_i^P$$

where $\tilde{n}_i$ is the vector of $i$-th component of normals on the reference element evaluated at face quadrature points, and $u_f, v_f$ are vectors of values of $u, v$ at face quadrature points. Since volume quadrature points don’t contribute to surface integration, we extend the boundary integration matrix to hybridized quadrature points by appending zeros:

$$B_{h,i}^P = \begin{bmatrix} 0 \\ B_i^P \end{bmatrix}, \quad B_{h,i} = h I_{N^P} \otimes B_{h,i}^P$$

### 3.2.3.3 Hybridized SBP operators

Given quadrature-based $L^2$ projection, we can now construct quadrature-based differentiation matrices by discretizing the operators $\frac{\partial}{\partial x_n} \Pi_{N^P}$ and $\frac{\partial}{\partial x_n} \Pi_N$, and evaluating them at quadrature points:

$$D_{q,n}^P = V_q^P D_{n}^P P_q^P, \quad D_{q,n} = V_q D_{n} P_q = I_{N^P} \otimes D_{q,n}^P$$

We note that $D_{q,n}^P$ is a $N^P$-th order accurate nodal differentiation matrix by construction.

Next, the hybridized operator introduced by Chan on $\tilde{D}^P$ is defined as [17]

$$Q_{h,n}^P = \begin{bmatrix} Q_{q,n}^P - \frac{1}{2} (E^P)^T B_n^P E^P & \frac{1}{2} (E^P)^T B_n^P \\ -\frac{1}{2} B_n^P E^P & \frac{1}{2} B_n^P \end{bmatrix}, \quad Q_{q,n}^P = W^P D_{q,n}^P, \quad n = 1, \ldots, d - 1$$
which satisfies a SBP-like property. The hybridized operator is the integrated version of hybridized differentiation, which is a high order accurate derivative approximation which incorporates a boundary correction term \[17\]. We then extend this hybridized operator to \( \hat{D} \) through a Kronecker product:

\[
Q_{h,n} = h I_{NF} \otimes Q_{h,n}^P, \quad n = 1, \ldots, d - 1
\]

The spanwise domain is periodic, so no surface term appear in differentiation matrices in the spanwise direction. The hybridized differentiation matrix with respect to the spanwise coordinate direction is written as:

\[
Q_{h,d} = h D_F \otimes \begin{bmatrix}
W^P \\
0
\end{bmatrix}
\]

The key to proving discrete entropy stability is the SBP-like property of the hybridized operators, as described below:

**Theorem 3.2.1. (SBP-like properties of hybridized operators)** Hybridized operators defined above satisfy SBP like properties

\[
Q_{h,n} + Q_{h,n}^T = h I_{NF} \otimes B^P_{h,n}, \quad n = 1, \ldots, d - 1, \quad Q_{h,d} + Q_{h,d}^T = 0 \quad (3.21)
\]

\[
Q_{h,n} 1 = 0, \quad n = 1, \ldots, d \quad (3.22)
\]

**Proof.** By the proof in \[17\], for \( n = 1, \ldots, d - 1 \), \( Q_{h,n} + (Q_{h,n})^T = \begin{bmatrix}
0 \\
B^P_{h,n}
\end{bmatrix} \).

Through linearity of the Kronecker product and skew-symmetry of \( D_F \), the SBP-like prop-
properties of $Q_{h,n}$ follows from direct calculations.

Finally, $Q_{h,n}1 = 0$ by properties of the Kronecker product and $Q_{h,n}^P1 = 0$ and $D^F1 = 0$. \qed

Table 3.1 summarizes the operators on the reference element

<table>
<thead>
<tr>
<th>Operator</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_q$</td>
<td>Interpolation (Volume quadrature points)</td>
</tr>
<tr>
<td>$V_f$</td>
<td>Interpolation (Surface quadrature points)</td>
</tr>
<tr>
<td>$V_h$</td>
<td>Interpolation (Volume and surface quadrature points)</td>
</tr>
<tr>
<td>$D_n$</td>
<td>Modal differentiation along $x_n$</td>
</tr>
<tr>
<td>$W$</td>
<td>Volume quadrature weights</td>
</tr>
<tr>
<td>$W_f$</td>
<td>Surface quadrature weights</td>
</tr>
<tr>
<td>$W_h$</td>
<td>Volume and surface quadrature weights</td>
</tr>
<tr>
<td>$M$</td>
<td>Mass matrix</td>
</tr>
<tr>
<td>$P_q$</td>
<td>$L^2$ projection</td>
</tr>
<tr>
<td>$B_i$</td>
<td>Boundary integration along normals $\hat{n}_i$ (Surface quadrature points)</td>
</tr>
<tr>
<td>$B_{h,i}$</td>
<td>Boundary integration along normals $\hat{n}_i$ (Volume and surface quadrature points)</td>
</tr>
<tr>
<td>$D_{q,n}$</td>
<td>Quadrature based differentiation along $x_n$</td>
</tr>
<tr>
<td>$Q_{q,n}$</td>
<td>SBP-like operator along $x_n$ (Volume quadrature points)</td>
</tr>
<tr>
<td>$Q_{h,n}$</td>
<td>SBP-like operator along $x_n$ (Volume and surface quadrature points)</td>
</tr>
</tbody>
</table>

Table 3.1: Operators on the reference element
3.2.4 Flux differencing and entropy projection

In this section we briefly introduce flux differencing and entropy projection techniques. Flux differencing can be interpreted as a subcell based high order finite volume formulation using entropy conservative numerical fluxes, and entropy projections evaluate entropy conservative fluxes via the $L^2$ projection of entropy variables.

3.2.4.1 Flux Differencing

Entropy conservative numerical fluxes as introduced by Tadmor [9] are key components in entropy stable schemes:

**Definition 3.2.1.** Let $f_{i,S}(u_L, u_R)$ be a bivariate function which is symmetric and consistent with the $i$-th coordinate flux function $f_i(u)$

$$f_{i,S}(u_L, u_R) = f_{i,S}(u_R, u_L), \quad f_{i,S}(u, u) = f_i(u)$$

We refer to a numerical flux $f_{i,S}(u_L, u_R)$ as entropy conservative if, for entropy variables $v_L = v(u_L), v_R = v(u_R)$, the following properties hold:

$$\begin{align*}
(v_L - v_R)^T f_{i,S}(u_L, u_R) &= (\psi_{i,L} - \psi_{i,R}) \\
\psi_{i,L} &= \psi_i(v(u_L)), \quad \psi_{i,R} = \psi_i(v(u_R)), \quad i = 1, \ldots, d
\end{align*} \quad (3.23)$$

Adopting the continuous interpretation of flux differencing introduced by Gassner, Winters, and Kopriva [39], the derivative of the flux $f_i(u)$ can be reformulated using entropy
conservative fluxes as
\[ \frac{\partial f_i(u(x))}{\partial x_i} = 2 \frac{\partial f_i, S(u(x), u(y))}{\partial x_i} \bigg|_{y=x} \]

Our goal is to evaluate derivatives of the flux at quadrature points using the above formula. Recall that the quadrature based differentiation matrix \( D_{q,n}^P \) on \( \tilde{D}^P \) (which evaluates derivatives at quadrature points) is a high order approximation of the derivative. Combining the above interpretation of flux differencing with quadrature-based differentiation, we can discretize the flux as
\[ 2 \frac{\partial f_{n,S}(u(x), u(y))}{\partial x_n} \bigg|_{y=x_i} \approx 2 \frac{\Pi_N f_{n,S}(u(x_i), u(y))}{\partial x_n} \bigg|_{y=x_i} = 2(D_{q,n} F_n)_{ii} = 2((D_{q,n} \circ F_n) 1)_i \]

(3.25)
\[ (F_n)_{ij} = f_{n,S}(x_i, x_j) \] (3.26)

where \( \circ \) denotes the Hadamard product of two matrices. The last equality follows from the property of Hadamard product \((AB)_{ii} = ((A \circ B)1)_i \) [17].

### 3.2.4.2 Entropy Projection

To derive the continuous statement of entropy conservation, we test the DG formulation with entropy variables. At the discrete level, since the approximation space is the space of polynomials, we instead test the DG formulation with the \( L^2 \) projection of entropy variables, which we refer to as the “projected entropy variables”. Recall also that the proof of entropy stability needs the equality (3.23) to hold. Note that the conservative variables can be written as entropy variables via \( u_L = u(v_L), u_R = u(v_R) \). We can then evaluate the fluxes
and conservative variables in terms of projected entropy variables. We define projected entropy variables \( v_h \) and entropy-projected conservative variables \( \tilde{\mathbf{u}} \) as

\[
\begin{align*}
\mathbf{u}_q &= V_q \mathbf{u}_h, \\
v_h &= P_q v_q(
\mathbf{u}_q)
\end{align*}
\]

\[
\begin{align*}
\tilde{v}_q &= V_q v_h, \\
\tilde{\mathbf{v}}_f &= V_f v_h, \\
\tilde{\mathbf{v}} &= V_h v_h \\
\tilde{\mathbf{u}}_q &= u(V_q v_h), \\
\tilde{\mathbf{u}}_f &= u(V_f v_h), \\
\tilde{\mathbf{u}} &= u(\tilde{\mathbf{v}})
\end{align*}
\]

Given modal coefficients \( \mathbf{u}_h, v_h \) is the degree \( N_p \) polynomial \( L^2 \) projection of the entropy variables evaluated using volume quadrature. We then use \( v_h \) to define \( \tilde{\mathbf{u}} \), the conservative variables evaluated at volume and face quadrature points. We can view \( v_h \) as the discretization of \( \Pi_N \mathbf{v} \) and \( \tilde{\mathbf{u}} \) as the discretization of \( \mathbf{u}(\Pi_N \mathbf{v}) \).

### 3.2.5 An entropy stable DG-Fourier method on the reference element

We now discuss the construction of entropy stable DG-Fourier schemes. We wish to approximate the solution to a system of nonlinear conservation law by

\[
\mathbf{u}_N(\mathbf{x}, t) = \sum_{j=1}^{N_q} (\mathbf{u}_h(t))_j \varphi_j(\mathbf{x})
\]

Motivated by the variational formulation of nonlinear conservation laws on \( \hat{D} \), let \( f^*_m \) be a numerical flux which imposes boundary conditions.

\[
\left( \frac{\partial \mathbf{u}_N}{\partial t}, \varphi_i \right)_{\hat{D}} + \sum_{m=1}^{d} \left( \frac{\partial f_m(\mathbf{u}_N)}{\partial x_m}, \varphi_i \right)_{\hat{D}} + (\hat{n}_m \cdot (f^*_m - f_m(\mathbf{u}_N)), \varphi_i)_{\partial \hat{D}} = 0, \quad i = 1, \ldots, N_p
\]

(3.27)
CHAPTER 3. ENTROPY STABLE DISCONTINUOUS GALERKIN-FOURIER
METHOD

30

By periodicity in the spanwise direction, \( \langle \hat{n}_d \cdot (f^*_d - f_d(u_N)), \varphi_i \rangle_{\partial \hat{D}} \) vanishes. Discretizing the spatial derivative with flux differencing and using entropy-projected conservative variables \( \tilde{u} \), we arrive at the semi-discrete entropy stable DG-Fourier formulation

\[
M \frac{\partial \mathbf{u}_h}{\partial t} = -V^T_h \left( \sum_{i=1}^{d} 2(Q_{h,n} \circ \mathbf{F}_i) \mathbf{1} \right) - V^T_f \sum_{i=1}^{d-1} \mathbf{B}_i(f^*_i - f_i(\tilde{u}_f)) \tag{3.28}
\]

\[
(F_n)_{ij} = f_{n,S}(\tilde{u}_i, \tilde{u}_j), \quad 1 \leq i, j \leq N_h \tag{3.29}
\]

The following theorem describes the discrete entropy conservation of the scheme:

**Theorem 3.2.2.** Let \( f_{i,S} \) be an entropy conservative flux defined in Definition 3.2.1. Assuming continuity in time, physically meaningful \(^1\) solutions \( u_h \) of (3.28) satisfy a semi-discrete conservation of entropy

\[
1^T W \frac{d\eta(u_h)}{dt} = \sum_{i=1}^{d-1} 1^T W_f(\text{diag}(\hat{n}_i)(\psi_i(\tilde{u}_f) - \tilde{v}_f f^*_i)) \tag{3.30}
\]

which is the quadrature approximation of the statement of entropy conservation

\[
\int_{\hat{D}} \frac{\partial \eta(u_N)}{\partial t} \, d\hat{x} = \sum_{i=1}^{d} \int_{\partial \hat{D}} (\psi_i(\Pi_N \mathbf{v}) - (\Pi_N \mathbf{v})^T f^*_i) \hat{n}_i \tag{3.31}
\]

**Proof.** We test the variational formulation with the projected entropy variables \( \mathbf{v}_h \). The time derivative part can be simplified as

\[
\mathbf{v}_h^T M \frac{\partial \mathbf{u}_h}{\partial t} = 1^T W \frac{d\eta(u_h)}{dt} \approx \int_{\hat{D}} \frac{\partial \eta(u_N)}{\partial t} \, d\hat{x} \tag{3.32}
\]

\(^1\)A solution is referred to as “physically meaningful” if the entropy \( \eta(u_h) \) is convex.
CHAPTER 3. ENTROPY STABLE DISCONTINUOUS GALERKIN-FOURIER METHOD

Since $W$ is diagonal, the boundary terms on the right hand side can be simplified as

$$-V_h^T V_f^T \sum_{i=1}^{d-1} B_i (f_i^* - f_i(\tilde{u}_f)) = -V_f^T \sum_{i=1}^{d-1} B_i (f_i^* - f_i(\tilde{u}_f))$$

$$\approx -\sum_{i=1}^{d-1} \int_{\partial D} (\Pi_N \nu)^T (f_i (\Pi_N \nu) - f_i^*) \hat{n}_i$$ (3.33)

The nonlinear terms on right hand side can be decomposed into

$$-V_h^T V_h^T \left( \sum_{i=1}^{d} 2(\mathbf{Q}_{h,n} \circ \mathbf{F}_i) \mathbf{1} \right) = -V_h^T \left( \sum_{i=1}^{d} 2((\mathbf{Q}_{h,n} - \mathbf{Q}_{h,n}^T) \circ \mathbf{F}_i) \mathbf{1} \right) - V_h^T \left( \sum_{i=1}^{d-1} 2(\mathbf{B}_{h,n} \circ \mathbf{F}_i) \mathbf{1} \right)$$

By direct calculation, the surface contributions can be combined as

$$-V_h^T \left( \sum_{i=1}^{d} 2(\mathbf{B}_{h,n} \circ \mathbf{F}_i) \mathbf{1} \right) - V_f^T \sum_{i=1}^{d-1} B_i (f_i^* - f_i(\tilde{u}_f)) = -V_f^T \sum_{i=1}^{d-1} B_i f_i^*$$

$$\approx -\sum_{i=1}^{d-1} \int_{\partial D} (\Pi_N \nu)^T f_i^* \hat{n}_i$$ (3.36)

Finally, the volume contributions are simplified through the SBP-like property of the nonlinear operator and properties of entropy conservative fluxes

$$-V_j^T \left( \sum_{n=1}^{d} 2((\mathbf{Q}_{h,n} - \mathbf{Q}_{h,n}^T) \circ \mathbf{F}_i) \mathbf{1} \right) = -\sum_{n=1}^{d} \sum_{i,j=1,\ldots,N_h} (\mathbf{Q}_{h,n})_{ij} (\tilde{v}_j - \tilde{v}_j) f_n, S(\tilde{u}_i, \tilde{u}_j)$$

$$= -\sum_{n=1}^{d} \psi_n (\tilde{v})^T (\mathbf{Q}_{h,n} - \mathbf{Q}_{h,n}^T) \mathbf{1}$$ (3.38)

$$= -\sum_{n=1}^{d} \psi_n (\tilde{v})^T (2\mathbf{Q}_{h,n} - \mathbf{B}_{h,n}^T) \mathbf{1} = \sum_{n=1}^{d-1} \psi_n (\tilde{v})^T \mathbf{B}_{h,n}^T \mathbf{1}$$ (3.39)

$$\approx \sum_{i=1}^{d-1} \int_{\partial D} \psi_i (\Pi_N \nu) \hat{n}_i$$ (3.40)

The result follows by combining this with previous equalities, and noting that $\int_{\partial D} (\psi_d (\Pi_N \nu) - (\Pi_N \nu)^T f_d^*) \hat{n}_d = 0$ by periodicity on the spanwise direction.
The scheme is also locally conservative, as stated in the following theorem.

**Theorem 3.2.3.** Under the same assumption as Theorem 3.2.2, the semi-discrete formulation satisfies

\[
1^T W \frac{du_q}{dt} + \sum_{i=1}^{d-1} 1^T B_i f_i^* = 0
\]

which is the discretization of the statement of local conservation

\[
\int_{\hat{D}} \frac{\partial u_h}{\partial t} + \sum_{i=1}^{d} \int_{\partial \hat{D}} f_i^* \text{diag}(\hat{n}_i) = 0 \tag{3.41}
\]

**Proof.** We test the variational formulation with \(P_q 1\), and proceed similarly as in Theorem 3.2.2. \qed

An equivalent skew-symmetric formulation, which loosens requirements on quadrature strength [20], is presented below. The equivalency is established through the SBP-like property of the nonlinear operators. This formulation is also used in the implementation.

\[
M \frac{\partial u_h}{\partial t} = -V_h^T \left( \sum_{i=1}^{d-1} ((Q_{h,n} - Q_{h,n}^T) \circ F_i) 1 + (Q_{h,d} \circ F_{d,S}) 1 \right) - V_f^T \sum_{i=1}^{d-1} B_i f_i^* \tag{3.42}
\]

\[
(F_n)_{ij} = f_{n,S}(\hat{u}_i, \tilde{u}_j), \quad 1 \leq i, j \leq N_h \tag{3.43}
\]

### 3.2.6 An entropy stable DG-Fourier method on mapped elements

We can now extend the scheme to multiple mapped elements. We denote the physical domain by \(\Omega = \Omega^P \times \Omega^F\) with dimension \(d\), \(\Omega^P\) and \(\Omega^F\) are dimension \(d-1\) and 1 respectively. Based on our choice of reference element, we discretize the physical space \(\Omega\) into \(K\) disjoint
elements $\Omega = \bigcup_{k=1}^{K} D^k$, where each element can be written as the product $D^k = D^P,k \times \Omega^F$.

Throughout this work, we assume physical elements are images of the reference element through affine mappings

$$\Phi^P,k(\hat{D}^P) = D^P,k, \quad \Phi^F(\hat{D}^F) = \Omega^F, \quad \Phi^k(\hat{D}) = D^k, \quad \Phi^k(r,s) = \Phi^P,k(r)\Phi^F(s)$$

Affine transformations imply that Jacobian determinants are constant on each element. For simplicity, we denote constant Jacobian determinants as scalar values

$$J^P,k = \left| \frac{\partial \Phi^P,k(r)}{\partial r} \right|, \quad J^F = \left| \frac{\partial \Phi^F(s)}{\partial s} \right|, \quad J = \left| \frac{\partial \Phi^k(r,s)}{\partial (r,s)} \right| = J^P,k J^F \quad (3.44)$$

Similarly, we denote the vector of face Jacobians as $\mathbf{J}_f^k$ and denote by $\hat{\mathbf{J}}_f$ the vector of Jacobians of mappings from faces of the reference element to the reference face. Note that $\hat{\mathbf{J}}_f$ is constant on each face. We define the geometric terms $\mathbf{G}^P = \frac{\partial (\Phi^P,k)^{-1}(x)}{\partial x}$ and can establish the relationship between $\mathbf{G}^P$ and physical normals $\mathbf{n}_i^k$ through [4]

$$\sum_{j=1}^{d-1} J^P,k \mathbf{G}_{ij}(\mathbf{n}_i \circ \hat{\mathbf{J}}_f) = \mathbf{n}_i^k \circ \mathbf{J}_f^k \quad (3.45)$$

The local approximation space on each element $D^k$ is defined as

$$P^{NP,k}(D^P,k) = P^{NP} \circ (\Phi^P,k)^{-1}, \quad F^{NF,k}(\Omega^F) = F^{NF} \circ (\Phi^F)^{-1}$$

$$V_h^k(D^k) = P^{NP,k}(D^P,k) \otimes F^{NF,k}(\Omega^F)$$

We wish to approximate the solution to a system of nonlinear conservation laws on $D^k$ by

$$\mathbf{u}_N^k(\mathbf{x},t) = \sum_{j=1}^{N_u} (\mathbf{u}_h^k(t))_j \varphi_j^k(\mathbf{x})$$
By construction, local interpolation matrices are the same as reference interpolation matrices.

Through a change of variables, integration on physical elements is now a scaling of integration on the reference element

\[
\int_{D^k} f^k(x) \, dx = \int_{\hat{D}} f(\hat{x}) J^k \, d\hat{x}, \quad \int_{\partial D^k} f^k(x) \, dx = \int_{\partial \hat{D}} f(\hat{x}) J^k_1 \, d\hat{x},
\]

In the matrix formulation, we take the Jacobian determinant into account by weighting the volume and surface quadrature rules accordingly.

\[
W^{P, k} = J^{P, k} W^P, \quad W^{P, k}_f = \text{diag}(J^k_f) W^P_f, \quad W^{P, k}_h = \begin{bmatrix} W^{P, k} \\ W^P_f \end{bmatrix}
\]

\[
W^{F, k} = J^F W^F
\]

\[
W^k = J^k W, \quad W^k_f = W^{F, k} \otimes W^P_f, \quad W^k_h = W^{F, k} \otimes W^P_h
\]

Through (3.45) and the chain rule, we can map reference operators to physical operators via

\[
B^k_i = J^k \sum_{j=1}^d G^P_{ij} B_i, \quad D^{P, k}_{q, i} = \sum_{j=1}^d G^P_{ij} D^P_{q, i}
\]

\[
Q^k_{h, i} = J^k \sum_{j=1}^d G^P_{ij} Q_{h, i}, \quad Q^{k}_{h, d} = J^{P, k} Q_{h, d}, \quad i = 1, \ldots, d - 1
\]

Guided by the entropy conservative DG-Fourier formulation on the reference element, the semi-discrete formulation over each \( D^k \) is then

\[
M^k \frac{\partial \mathbf{u}_h}{\partial t} = - (V^k_h)^T \left( \sum_{i=1}^d 2(Q^k_{h, n} \circ F^k_i) 1 \right) - (V^k_f)^T \sum_{i=1}^{d-1} B^k_i (f^*_i - f_i(\tilde{u}^k_f)) \quad (3.46)
\]

\[
(F^k_n)_{ij} = f_{n,S}(\tilde{u}^k_i, \tilde{u}^k_j), \quad 1 \leq i, j \leq N_h, \quad f^*_n = f_{n,S}(\tilde{u}_f, \tilde{u}^+_f) \quad (3.47)
\]
where \( \tilde{u}_j^+ \) denotes exterior values of the entropy-projected conservative variable, which can correspond to either exterior values on neighboring elements or an exterior state for enforcing boundary conditions.

**Theorem 3.2.4.** Define the boundary quadrature weight matrix as

\[
(W_{\partial\Omega})_{ii} = \begin{cases} 
(W_f^k)_{ii} & \text{if } x_i^{f,k} \in \partial \Omega \\
0 & \text{otherwise} 
\end{cases}
\]

Then, (3.46) satisfies both a semi-discrete conservation of entropy and local conservation

\[
\sum_{k=1}^{K} 1^T W^k \frac{d\eta(u^k_q)}{dt} = \sum_{k=1}^{K} \sum_{i=1}^{d-1} 1^T W_{\partial\Omega}^k (\text{diag}(n_i^k) (\psi_i(\tilde{u}_f^k) - (\tilde{v}_f^k)^T f_i^*)) ,
\]

\[
\sum_{k=1}^{K} 1^T W^k \frac{du^k_q}{dt} + \sum_{i=1}^{d-1} 1^T B_i^k f_i^* = 0,
\]

which are the quadrature approximations to the following

\[
\int_{\Omega} \frac{\partial \eta(u_N)}{\partial t} \, dx = \sum_{i=1}^{d} \int_{\partial\Omega} (\psi_i(\Pi_N v) - (\Pi_N v)^T f_i^*) n_i^k,
\]

\[
\int_{\Omega} \frac{\partial u_h}{\partial t} + \sum_{i=1}^{d} \int_{\partial\Omega} f_i^* \text{diag}(n_i^k) = 0.
\]

**Proof.** Since the SBP-like properties of \( Q_{h,n}^k \) are preserved for linear combinations of reference SBP-like operators \( Q_{h,n} \), the proof is the same as that of Theorem 3.2.2.

Note that the schemes introduced above are entropy conservative, which holds only when solutions are smooth. When shocks are present in the solution, entropy should be dissipated.
Our scheme should mimic this dissipation. Following [17, 14], we can add a local Lax-Friedrichs penalization term $-\frac{\lambda}{2}[\tilde{u}]$ to the formulation, where $\lambda$ is an estimation of maximum eigenvalue of the flux Jacobian matrix. For appropriate boundary conditions, the addition of Lax-Friedrichs dissipation yields a global entropy inequality [17]

$$\int_{\Omega} \frac{\partial \eta(u_N)}{\partial t} \, dx \leq \sum_{i=1}^{d} \int_{\partial \Omega} (\psi_i(\Pi_N v) - (\Pi_N v)^T f^*_i) n_i \leq 0$$

3.3 GPU acceleration

In this section, we briefly discuss the GPU implementation of the ESDG-Fourier scheme in 3D. The implementation is written in the programming language Julia 1.4 and utilizes the package CUDA.jl, the main interface for writing CUDA kernels natively in Julia. The code is hosted on Github [40]. We used Google Cloud platform as the environment in which to run the code. All experiments below are run on Nvidia Tesla V100 GPUs using double precision arithmetic.

We use the skew-symmetric formulation in our implementation because the formulation loosens requirements on quadrature strength and results in simpler surface terms. We divide
the computations into four kernels

\[
\frac{\partial \mathbf{u}_h^k}{\partial t} = -\frac{1}{J_k} \mathbf{M}^{-1} \mathbf{V}_h^T \left( \sum_{n=1}^{2} \left( (\mathbf{Q}_{h,n}^k - (\mathbf{Q}_{h,n}^k)^T) \circ \mathbf{F}_n^k \right) \right) + 2(\mathbf{Q}_{h,3}^k \circ \mathbf{F}_3^k) + \frac{1}{J_k} \mathbf{M}^{-1} \mathbf{V}_f^T \sum_{m=1}^{d-1} \mathbf{B}_m^k \mathbf{f}_m^k,
\]

where

- **update kernel**
- **xy flux-differencing kernel**
- **z flux-differencing kernel**
- **surface kernel**
- **entropy projection kernel**

The GPU implementations of the update, surface and entropy projection kernels are described in [41, 42], and are similar to implementations of nodal DG methods on GPUs. We focus here on the flux differencing step \((\mathbf{Q}_{h,n}^k \circ \mathbf{F}_n^k)\mathbf{1}\), and restrict our discussion to the implementation of the flux differencing kernel, which differs from standard volume kernels for nodal DG methods.

We parallelize computations by assigning each thread a row of \((\mathbf{Q}_{h,n}^k \circ \mathbf{F}_n^k)\mathbf{1}\) to compute, Entropy-projected variables are loaded into shared memory for fast memory access during flux evaluation. To avoid overuse of shared memory which can lead to a reduction in achieved occupancy, we adopt a two kernel splitting strategy. Instead of evaluating rows of \(\sum_{n=1}^{d} (\mathbf{Q}_{h,n} \circ \mathbf{F}_n^k)\mathbf{1}\) in one pass, we split the summation in two passes, one through triangles, another one through Fourier slices. Figure 3.2 shows the sparsity pattern and the corresponding nodes on the reference element. The procedure is described in pseudocode below.
CHAPTER 3. ENTROPY STABLE DISCONTINUOUS GALERKIN-FOURIER METHOD

Algorithm 1: Flux Differencing on triangles

for each triangular element do
  load entropy-projected variables on current triangle into shared memory;
prefetch $Q^P_{h,1}, Q^P_{h,2}$, geometric factors into local memory;
synchronize threads;
  for each hybridized point do
    compute the current hybridized point’s contribution to
    \[ \sum_{n=1}^{2} ((Q^k_{h,n} - (Q^k_{h,n})^T) \circ F_n) 1, \]
    where $F_n$ is computed on the fly;
    store results into global memory;
  end
end

Figure 3.2: Sparsity plot of $\sum_{n=1}^{d-1} Q^k_{h,n} - (Q^k_{h,n})^T$ and $Q_{h,d}$, and corresponding flux evaluations in the physical space. $N^F = 4$ and $N^P = 4$

Due to the sparsity pattern of $Q^k_{h,d}$, the number of Fourier slices is $N^P_q$ instead of $N^P_h$. 
Algorithm 2: Flux Differencing on Fourier slices

for each Fourier slice do
    load entropy-projected variables evaluated on current Fourier slice into shared memory;
    prefetch \( Q^F \), geometric factors into local memory;
    synchronize threads;
    for each Fourier node do
        compute the current Fourier node’s contribution to \( (Q_h^k \circ F_3)I \), where \( F_h \) is computed on the fly;
        store results into global memory;
    end
end

3.4 Numerical Experiments: 3D Compressible Euler Equations

In this section we present numerical examples for the compressible Euler equations in 3D:

\[
\frac{\partial U}{\partial t} + \sum_{i=1}^{3} \left( \frac{\partial f_i(U)}{\partial x_i} \right) = 0
\]

\[
U = \begin{bmatrix}
\rho \\
\rho u \\
\rho v \\
\rho w \\
E
\end{bmatrix}, \quad f_1(U) = \begin{bmatrix}
\rho u \\
\rho u^2 + p \\
\rho u v \\
\rho u w \\
u(E + p)
\end{bmatrix}, \quad f_2(u) = \begin{bmatrix}
\rho v \\
\rho v^2 + p \\
\rho v w \\
\rho v w \\
v(E + p)
\end{bmatrix}, \quad f_3(U) = \begin{bmatrix}
\rho w \\
\rho w^2 + p \\
\rho w v \\
\rho w w \\
w(E + p)
\end{bmatrix}
\]
CHAPTER 3. ENTROPY STABLE DISCONTINUOUS GALERKIN-FOURIER

where \( \rho \) is density, \((u, v, w)\) are velocity components in \((x, y, z)\) direction, and \(E\) is the total energy. The vector of five variables \( \mathbf{U} \) is referred as the conservative variables. In addition, the pressure \( p \) and specific internal energy \( \rho e \) are related to the conservative variables through the constitutive relations:

\[
p = (\gamma - 1) \left( E - \frac{1}{2} \rho (u^2 + v^2 + w^2) \right), \quad \rho e = E - \frac{1}{2} \rho (u^2 + v^2 + w^2)
\]

where \( \gamma = 1.4 \) is the ratio of specific heat for a diatomic gas.

The unique entropy \( \eta(\mathbf{u}) \) for the viscous compressible Navier-Stokes equation is

\[
\eta(\mathbf{u}) = -\frac{\rho s}{\gamma - 1}
\]

where \( s = \log\left(\frac{p}{\rho} \gamma \right) \) is the physical specific entropy [43]. In this work, entropy stability and conservation are based on this choice of mathematical entropy.

The mappings between the entropy variables \( \mathbf{V} \) and the conservative variables \( \mathbf{U} \) is given by

\[
\mathbf{V}(\mathbf{U}) = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{bmatrix} = \begin{bmatrix} \frac{\rho e(\gamma+1-s)-E}{\rho e} \\ \frac{\rho u}{\rho e} \\ \frac{\rho v}{\rho e} \\ \frac{\rho w}{\rho e} \\ -\frac{\rho}{\rho e} \end{bmatrix}, \quad \mathbf{U}(\mathbf{V}) = \begin{bmatrix} -(\rho e)v_5 \\ -(\rho e)v_2 \\ -(\rho e)v_3 \\ -(\rho e)v_4 \\ \rho e \left(1 - \frac{v_2^2 + v_3^2 + v_4^2}{2v_5}\right) \end{bmatrix}
\]

with specific internal energy and physical specific entropy related to the entropy variables through

\[
\rho e = \left(\frac{\gamma - 1}{(-v_5)^{\gamma}}\right)^{\frac{1}{\gamma-1}} e^{-\frac{s}{\gamma-1}}, \quad s = \gamma - v_1 + \frac{v_2^2 + v_3^2 + v_4^2}{2v_5}
\]
We use Chandrashekar’s entropy conservative flux [44] that satisfies Tadmor’s definition 3.2.1

\[
\begin{bmatrix}
\{\rho\}\log\{u\} \\
\{\rho\}\log\{u\}^2 + p_{\text{avg}} \\
\{\rho\}\log\{u\}\{v\} \\
\{\rho\}\log\{u\}\{w\} \\
(E_{\text{avg}} + p_{\text{avg}})\{u\}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\{\rho\}\log\{v\} \\
\{\rho\}\log\{u\}\{v\} \\
\{\rho\}\log\{u\}^2 + p_{\text{avg}} \\
\{\rho\}\log\{u\}\{w\} \\
(E_{\text{avg}} + p_{\text{avg}})\{v\}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\{\rho\}\log\{w\} \\
\{\rho\}\log\{u\}\{w\} \\
\{\rho\}\log\{v\}\{w\} \\
\{\rho\}\log\{w\}^2 + p_{\text{avg}} \\
(E_{\text{avg}} + p_{\text{avg}})\{w\}
\end{bmatrix}
\]

where we have introduced averages, logarithmic averages, and intermediate variables:

\[
\{f\} = \frac{f^+ + f^-}{2}, \quad \{f\}^{\log} = \frac{f^+ - f^-}{\log f^+ - \log f}
\]

\[
p_{\text{avg}} = \frac{\{\rho\}}{2\{\beta\}}, \quad E_{\text{avg}} = \frac{\{\rho\}\log\{\rho\}}{2\{\beta\}\log(\gamma - 1)} + \frac{u_{\text{avg}}^2}{2}, \quad u_{\text{avg}}^2 = uu^+ + vv^+ + w^+ w^+ , \quad \beta = \frac{\rho}{2p}
\]

The logarithmic average is evaluated using the numerically stable expansion in [45].

3.4.1 Vortex Propagation and Sine wave

In this section, we examine the convergence of the proposed method in 3D using the isentropic vortex problem and a convecting wave solution. The convergence of high order
entropy stable DG has been thoroughly studied in previous works [17], so we focus on illustrating exponential convergence as the number of Fourier modes in the spanwise direction are increased. The vortex propagates in the $z$ direction, and the solution has the following analytic expression [46]:

$$
\begin{bmatrix}
\rho(x, t) \\
u(x, t) \\
v(x, t) \\
w(x, t) \\
E(x, t)
\end{bmatrix} =
\begin{bmatrix}
(1 - \frac{\gamma - 1}{2} \Pi^2)^{\frac{1}{\gamma - 1}} \\
\Pi r_1 \\
\Pi r_2 \\
\Pi r_3 + 1 \\
\frac{\rho_0}{\gamma - 1} (1 - \frac{\gamma - 1}{2} \Pi^2)^{\frac{1}{\gamma - 1}} + \frac{\rho}{2} (u^2 + v^2 + w^2)
\end{bmatrix}
\begin{bmatrix}
r_1 \\
r_2 \\
r_3
\end{bmatrix} =
\begin{bmatrix}
0 \\
-(z - c_3 - t) \\
y - c_2 - t
\end{bmatrix}, \quad \Pi = \Pi_{\text{max}} e^{\frac{1-r_1^2-r_2^2-r_3^2}{2}}
$$

where the vortex is centered in the $yz$ plane at $(c_2, c_3) = (5.0, 4.5)$. For those experiments, we set $\rho_0 = \frac{1}{\gamma}$ and $\Pi_{\text{max}} = 0.4$.

A cubic computational domain $\Omega = [0, 10]^3$ is discretized into wedges by decomposing the $xy$ plane into uniform quadrilateral elements, then subdividing each quadrilateral element into two uniform triangles. Each uniform triangle is then extruded in the $z$ direction.

We also test an additional diagonally convecting wave solution

$$
\begin{align*}
\rho(x, t) &= 1 + 0.2 \exp (\sin(\pi (x + y + z - t(u + v + w))))), \\
u(x, t) &= 1, \quad v(x, t) = -\frac{1}{2}, \quad w(x, t) = 1, \quad p(x, t) = 1
\end{align*}
$$
over a cubic domain $\Omega = [0, 2]^3$, which is discretized similarly as above.

We use an explicit low storage RK-45 time stepper for convergence experiments, and estimate the time step by

$$dt = \frac{C_{CFL} h}{C_N}, \quad h = \frac{L}{2K_{1D}}, \quad C_N = \frac{3(N_P + 1)(N_P + 2)}{2}$$

where $L$ is the length of the cubic domain, $K_{1D}$ is the number of uniform subintervals along $x$ or $y$ direction, $h$ estimates the mesh size, $C_N$ is the constant in the hp inverse trace inequality on a triangle [47], and $C_{CFL}$ is the user defined CFL constant.

In both cases, we assume the domain is periodic in all directions, and Lax-Friedrichs dissipation is introduced through the numerical flux. We run the sine wave case with polynomial degree $N_P = 7$ and the vortex propagation test case with $N_P = 3$. Both cases use meshes with a total number of elements $K = 800$ ($K_{1D} = 20$) until final time $T = 1.0$ with $C_{CFL} = 0.5$. We compute the $L^2$ error using a higher degree $N_P + 2$ quadrature rule on triangles composed with the periodic trapezoidal rule in the spanwise direction.

The semilog plots on Figures 3.3 and 3.4 show the convergence behaviour of both test cases. For both cases, as we increase the number of Fourier modes, the error decreases exponentially, which verifies spectral accuracy in the spanwise direction when the solution is sufficiently smooth.
3.4.2 Doubly periodic shear layers

Finally, we present a numerical simulation of doubly periodic shear layers in 3D using the proposed method. The initial condition is a slight modification of Brown and Minion’s doubly periodic pairs of shear layer [48] for the 3D case, where the shear layer width varies continuously with respect to $z$. The initial condition is given as

$$u(x, t) = \begin{cases} \tanh (\epsilon (1 + z^2) (y + 0.25)) & \text{if } y < 0 \\ \tanh (\epsilon (1 + z^2) (0.25 - y)) & \text{otherwise} \end{cases}$$

$$v(x, t) = \delta \cos(2\pi x), \quad w(x, t) = \delta \cos(2\pi x),$$

$$\rho(x, t) = 1, \quad p(x, t) = \frac{1}{\gamma M^2},$$

where $M$ is the Mach number, $\epsilon = 30$ is the parameter controlling shear layer width, and $\delta = 0.05$ is the strength of the initial perturbation to the $y$ and $z$ velocities.
We solve the equation in cubic domain $[-1,1]^3$, discretized into wedges similarly as in previous section. We utilize the adaptive 5th order Dormand-Prince method to advance in time. We assume the domain is periodic in all directions and again introduce Lax-Friedrichs dissipation through the numerical flux. In order to compare the difference between the proposed method when shocks are and are not present in the solution, we run the simulation with two different Mach numbers 0.3 and 0.7. We use a refined mesh, with a total of $K = 32768$ ($K_{1D} = 128$) elements until $T = 2.0$ and approximate the solution with degree $N^P = 4$ polynomials and $N^F = 8$ Fourier modes. We plot the squared norm of velocity, $u^2 + v^2 + w^2$, and the $z$ component of vorticity $\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}$ at the first, fourth and eighth Fourier slices ($z = -0.75, 0.0$ and $1.0$) in Figure 3.5. From Figure 3.7, we observe that shocks form after time $T \approx 2.0$ when the Mach number is 0.7. However, the proposed entropy stable method remains robust in the presence of shocks.
Figure 3.6: Doubly periodic shear layer, $Ma = 0.3$, $K_{1D} = 128$
CHAPTER 3. ENTROPY STABLE DISCONTINUOUS GALERKIN-FOURIER METHOD

(a) Squared norm of velocity, Fourier mode 1
(b) $z$ component of vorticity, Fourier mode 1

c) Squared norm of velocity, Fourier mode 4
d) $z$ component of vorticity, Fourier mode 4

e) Squared norm of velocity, Fourier mode 8
f) $z$ component of vorticity, Fourier mode 8

Figure 3.7: Doubly periodic shear layer, $Ma = 0.7$, $K_{1D} = 128$
Finally, we present the numerical simulation of doubly periodic shear layers in a under-resolved case with Mach number 0.7. The equation is discretized on a coarser mesh with a total of $K = 800$ ($K_{1D} = 20$) elements. We approximate the solution with degree $N^P = 7$ polynomials and $N^F = 8$ Fourier modes. Under this configuration, from Figure 3.8, the proposed scheme does not blow up in presence of shocks, and resolves the shocks up to oscillations introduced by the coarse mesh. We should note that the solution is qualitatively different when $K_{1D} = 20$. This may be due to the sensitivity of the Euler equations [49].
Figure 3.8: Doubly periodic shear layer, $Ma = 0.7$, $K_{1D} = 20$
Chapter 4

Conclusion

In this work, we proposed an entropy stable discontinuous Galerkin-Fourier method. The construction is based on modal entropy stable discontinuous Galerkin methods as proposed by Chan [17]. We extend this formulation to higher dimensions through Fourier extrusion and a tensor product formulation. Discrete entropy conservation of the scheme follows from the SBP properties of spectral differentiation operators. We present a two-kernel split GPU acceleration strategy. Numerical experiments show “spectral convergence” in the spanwise direction, and we present numerical results for a modification of the doubly periodic shear layer problem in 3D which show the robustness of the proposed method for underresolved and shock solutions. Future work will include extending the method to non-conforming and moving meshes.
Bibliography


[43] Thomas JR Hughes, LP Franca, and M Mallet. “A new finite element formulation for computational fluid dynamics: I. Symmetric forms of the compressible Euler and


