A COMPUTATIONAL APPROACH TO FLAME HOLE DYNAMICS

BY

ROBERT C. KNAUS

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Mechanical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2014

Urbana, Illinois

Doctoral Committee:

Associate Professor Carlos Pantano, Chair and Director of Research
Dr. John Hewson
Associate Professor Tonghun Lee
Professor Moshe Matalon
Associate Professor Matthew West

Abstract

Turbulent diffusion flames at low strain rates sustain a spatially continuous flame surface. However, at high strains, which may be localized in a flow or not, the flame can be quenched due to the increased heat loss away from the reaction zone. These quenched regions are sometimes called flame holes. Flame holes reduce the efficiency of combustion, can increase the production of certain pollutants (e.g. carbon monoxide, soot) as well as limit the overall stability of the flame. We present a numerical algorithm for the calculation of the dynamics of flame holes in diffusion flames. The key element is the solution of an evolution equation defined on a general moving surface. The low-dimensional manifold (the surface) can evolve in time and it is defined implicitly as an iso-level set of an associated Cartesian scalar field. An important property of the method described here is that the surface coordinates or parameterization does not need to be determined explicitly; instead, the numerical method employs an embedding technique where the evolution equation is extended to the Cartesian space, where well-known and efficient numerical methods can be used. In our application of this method, the field defined on the surface represents the chemical activity state of a turbulent diffusion flame. We present a formulation that describes the formation, propagation, and growth of flames holes using edge-flame modeling in laminar and turbulent diffusion flames. This problem is solved using a high-order finite-volume WENO method and a new extension algorithm defined in terms of propagation PDEs. The complete algorithm is demonstrated by tracking the dynamics of flame holes in a turbulent reacting shear layer. The method is also implemented in a generalized unstructured low-Mach number fluid solver (Sandia's SIERRA low Mach Module "Nalu") and applied to simulate local extinction in a piloted jet diffusion flame configuration.

To my mother and father, Nancy and William Knaus.

Acknowledgements

I would like to thank my advisor Prof. Carlos Pantano for his help, support and guidance through the years of my graduate study. His exceptional patience, mentorship and insight have been invaluable to me.

I would like to express my gratitude to the past and current members of The High Performance Turbulence and Combustion Modeling Laboratory—Kai-Pin Liao, Reetesh Ranjan, Hasib Uddin, Jaeseung Byun, Richard Kramer, Timothy Smith, and Sicong Wu—for their feedback, support and friendship through the years. In particular, I would like to acknowledge Kai-Pin Liao, Reetesh Ranjan, and Hasib Uddin for encouragement, assistance, and camaraderie over the years.

I thank Dr. John Hewson and Dr. Stefan Domino for helping with this work during my summer at Sandia National Laboratories. I am also grateful to the members of my doctoral committee—Prof. Moshe Matalon, Prof. Matthew West, Prof. Tonghun Lee, and Dr. John Hewson—for their time and guidance in the development of this dissertation.

I thank the National Science Foundation for partially supporting this work under grant CBET-1236164. I also thank the Department of Energy, National Nuclear Security Administration, for partially supporting this work under Award Number DE-NA0002374.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Turbulent diffusion flames experience velocity gradients (strain rates) that vary in space and time, increasing with increasing Reynolds number. Since the combustion conversion rate is limited by the fixed chemical time scales of the elementary reactions that describe the combustion of each fuel-oxidizer mixture, eventually combustion is unable to adjust to fast flow time scales and the reaction is partially and locally quenched. The quenching or extinction process starts at those locations where the rate of heat release, which sustains the high temperature of the combustion, is unable to balance the rate of heat loss from the reaction zone [5, 6]. The flame ceases to exist once the temperature drops below the extinction limit temperature and all remnant temperature and chemical products diffuse quickly away and radicals recombine into stable, practically inert, species. If the Reynolds number keeps increasing, for example by increasing the velocity in a jet flame, extinction will propagate as a global instability, called flame blow off, which will quench the flame completely (not just locally or transitorily). The process by which high strain rates quench the flame is called extinction while the healing of a quenched zone, by advection or heating to more favorable regions is termed reignition. These locally quenched regions of a flame are called here “flame holes” in accordance with [7]. Although we call them “holes”, the extinction zones can have arbitrary shapes even resembling strips or islands [8, 9]. Flame holes are an important finite-chemistry effect which can affect the rate at which certain pollutants (e.g. carbon monoxide, soot, etc.) are produced and reduce the amount of fuel burned by the flame.

The main contribution of my thesis is the development of a method of modeling the dynamics of flame holes and the presentation of a novel numerical framework to solve the governing equation.

1.1 Flame hole dynamics

After the development of the flame hole, the quenched region grows, shrinks, merges, splits, and changes shape depending on the evolution of the flame rim. Computationally modeling all these flow/chemistry interactions at high Reynolds numbers is prohibitively expensive using first-principle methods, i.e. resolving all fields. In the absence of extinction, one well-established modeling approach is to represent the mixture as an ensemble of thin reaction zones called flamelets [10, 11, 12]. The reaction takes place near the stoichiometric surface, where fuel and oxidizer meet in stoichiometric proportions. But, the flamelet concept is inapplicable when the flame is quenched because there is no flame at those locations and, furthermore, the flame hole boundary propagates at its own speed, here called the edge-flame speed V_e. As demonstrated in experiments [13] and in direct numerical simulations [2, 14], the controlling process determining the growth or shrinkage of a hole is the velocity of propagation of the boundary separating the burning flame form the quenched region. Generally, this boundary (or flame rim) is a nominally two-dimensional time-dependent flame front, which can possibly have some degree of three-dimensional structure. To leading order of approximation, the flame boundary can be understood as the proper flame structure resulting from the balance of heat conduction upstream towards a stratified fuel-oxidizer mixture propagating (as in a premixed flame) with a trailing diffusion flame [15, 16, 17]. The flame front (or tip) is generally not planar but curved towards the diffusion flame with lean and rich wings burning the excess fuel or oxidizer. The diffusion flame trailing behind the edge flame approaches quickly the nominal position determined by the stoichiometric surface of the otherwise unperturbed diffusion flame. This description of extinction/reignition flame dynamics was originally introduced by Dold et al. [7, 18] and used preliminary by [19] and [20]; see Figure 2.1. Extinction of the edge flame is given by the extinction limit of the corresponding canonical diffusion flame, parameterized by a scalar dissipation value χ_{q}.

The present modeling approach removes the need for detailed calculation of the advection-diffusion-reaction problem that is tightly coupled in the edge flame. The only information required in this modeling approach is the edge flame velocity given as a function of the local strain rate as well as the orientation of the flame boundary curve, since the edge flame is supposed to propagate normal to this curve. Time-dependent effects, such as unstationary response of the edge flame to time variations of mixture composition, strain rate, etc., are not incorporated at this level of closure, but they could be included in future, extended models, if deemed necessary. In the end, one needs to solve an evolution equation for a flame state field on the moving stoichiometric surface.

1.2 Numerical solution of partial differential equations on surfaces

The numerical solution of partial differential equations on surfaces is an area of research with a variety of applications such as image processing, geometry, physiology, solidification, gravitation, and fluid mechanics (see Ratz and Voigt [21]). The problem of evolution of a field constrained to a surface can be described in terms of surface differential operators (gradient, divergence, etc.) if a parametric description of the surface (e.g. z_1, z_2) is available. Unfortunately, it is difficult and computationally expensive to employ this approach for a general surface, where even creating a clean surface mesh might be difficult [21]. Additionally, frequent re-meshing may be needed if the surface evolves in time, and particularly if the surface experiences topological changes. An alternative approach to solving PDEs on surfaces using the parametric coordinates is to embed the surface in the three-dimensional Cartesian space. This requires a transformation of the surface PDE to an equivalent volumetric PDE that allows one to solve the latter using customary Cartesian discretizations.

The heart of an embedding method is an extension operator by which the surface field is extended smoothly throughout the Cartesian space. Surface data is propagated normal to isosurfaces of an embedding function that implicitly defines the surface. These methods can be classified into: geometrical and differential equation-based methods. Introduced by Bertalmıo et al. [22], differential equation-based methods construct a PDE defining the extension operation in the embedding space. The solution of this equation has the property that the extended surface data in the Cartesian domain is normal to the isosurfaces of the embedding function. Cartesian discretizations can then be applied in the embedding space to solve the surface PDE on the implicit surface. This method has been applied to a wide variety of problems [21, 22, 23] and has been used for solving surface advection problems on evolving surfaces [24, 25]. The method nonetheless has a number of drawbacks. The addition of an extra dimension to the surface problem increases the amount of computational work required to solve the problem. To minimize this effect, the embedding PDE is typically only solved in a narrow band around the surface. However, solving the embedding PDE requires the imposition of artificial boundary conditions at the boundary of the band in which the embedding PDE is solved [23]. If instead we seek to solve the embedding PDE in a much larger band around the surface, the artificial imposition of boundary conditions is not an issue and this type of method does not have difficulty with extending surface data far away from the surface. In the geometrical "closest point method" [26, 27, 28], surface data is propagated along the local normal direction by use of a closest point extension. Values of the extended surface field at grid points in the embedding Cartesian space are set to be equal to the value of the surface field evaluated at the point on the surface that is closest to the Cartesian grid point. This is implemented by determining the closest point on the surface relative to a grid point through a closest point transform and then interpolating the value of the variable at that point from the data in the Cartesian grid surrounding it. The closest point extension has the desired property that the intrinsic surface gradient operator (and divergence operator) is the same as the Cartesian gradient of the extended field when evaluated at the surface and does not require the imposition of artificial boundary conditions. The method can also be constructed to be high-order accurate. For instance, the orthogonal gradient radial basis function method of Piret [29] uses a closest point representation with the surface approximated using a radial basis function approach to obtain spectral accuracy. It has been successful in solving diffusion and advection equations [27] and has been modified to efficiently solve problems on moving surface by Leung et al. [30]. The grid-based particle method by Leung et al. [30] reconstructs the interface through a local least-squares approximation. This allows the computation of the closest point transform, which is used to apply the closest point extension. The method tracks the interface motion by using a Lagrangian particle tracking method, using the movement of the surface to update the closest point extension. This type of method allows for the efficient computation of the closest point method for the case of an evolving surface. Finding the closest point to a surface, however, becomes increasingly difficult as the distance from the surface increases. The closest point function can be costly to compute and multivalued. As previously stated, PDE-based methods do not have problematic behavior far from the surface. For this reason, PDE-based methods based on an elliptic-type differential operator and an anisotropic diffusion operator are implemented using the closest point extension as an internal Dirichlet boundary condition.

1.3 Research objectives

The thesis has several core research objectives:

Development of a flame hole dynamics model equation

> The essential development of this thesis is the description of flame hole dynamics. Through the research activity, a model equation describing the physical phenomena associated with local extinction in turbulent diffusion flames was developed.

Development of a method for the solution of PDE's on complex, moving surfaces

> As the model equation is defined nominally on a complex, moving surface, the numerical solution of the resulting equation is difficult to construct. A novel numerical method was created to efficiently solve the flame hole dynamics model equation. The method was implemented in parallel multiprocessor clusters in both a structured and unstructured setting.

Application of the method toward turbulent reacting systems of interest

> Further work is done in order to take the solution of the model equation and use it in a realistic scenario. A high-order method was demonstrated on the stoichiometric surface from a turbulent reacting shear layer simulation. The implementation into a parallel unstructured solver allows the model to be used in large scale parallel simulations with domains involving complex geometry.

Chapter 2

Flame Hole Modeling

2.1 Kinematic and dynamic aspects of flame hole modeling

The flame hole dynamics description of extinction/reignition processes in non-premixed combustion assumes a thin reaction zone residing near the stoichiometric surface; sufficiently thin that it can be considered as a mathematical surface from the point of view of the hydrodynamics coupling. This surface is defined implicitly in terms of the mixture fraction field, $Z(\mathbf{x}, t)$, according to

$$\Sigma_{\mathrm{st}} = \{\mathbf{x} : Z(\mathbf{x}, t) = Z_{\mathrm{st}}\}, \tag{2.1}$$

where t denotes time, Z_{st} is a constant that depends on the stoichiometry of the system and $Z(\mathbf{x}, t)$, which is bounded between zero and one, obeys a partial differential equation independent of the combustion chemistry details to leading order, given by

$$\rho \frac{DZ}{Dt} = \nabla \cdot (\rho D_{\mathrm{mix}} \nabla Z), \tag{2.2}$$

where D/Dt denotes the material derivative, ρ is the density and D_{mix} denotes the average diffusivity of the primary reactants and products; see Williams [31]. In a leading order description, where unstationary effects are neglected, the flame strength responds instantaneously according to the value of the scalar dissipation, defined by

$$\chi(\mathbf{x}, t) = 2D_{\mathrm{mix}} \nabla Z \cdot \nabla Z. \tag{2.3}$$

The flame will locally extinguish and form a flame hole if the scalar dissipation exceeds the quenching value, denoted by the constant χ_{q} (which is well defined for each particular reacting system).

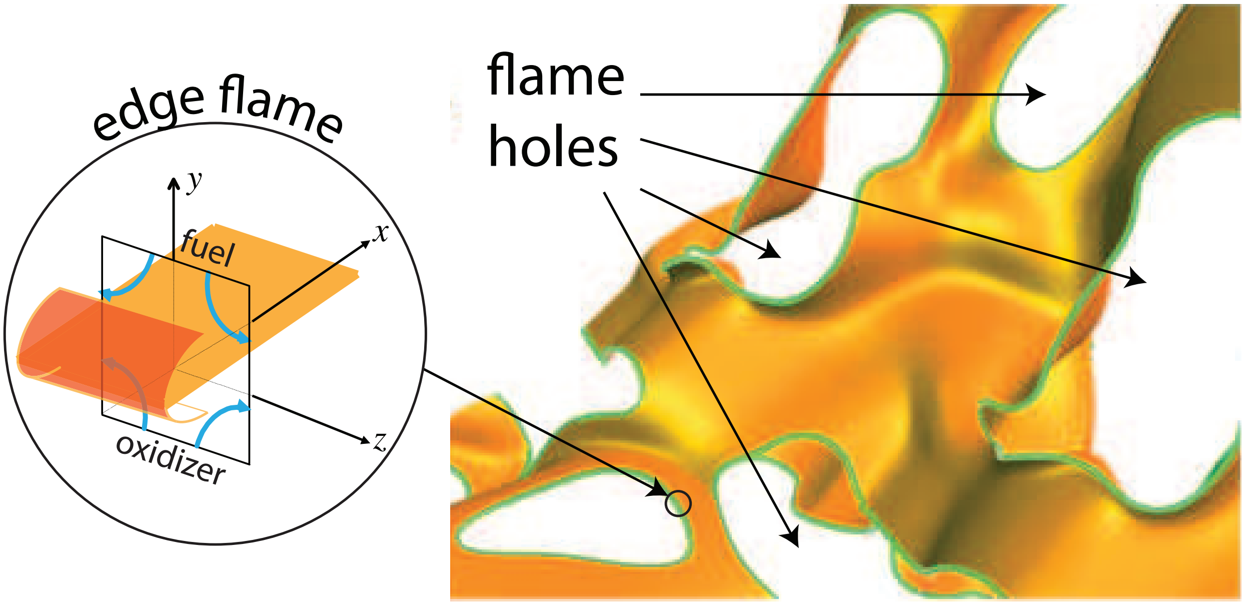


Figure 2.1: Flame holes in a turbulent diffusion flame [2].

The term "flame hole" here is used generally to denote local extinction regions and it is used to refer to general non-circular holes with an arbitrarily shaped boundary, Γ_i, where i is a hole index number (assuming the holes are countable entities). The location of an isolated flame hole boundary (FHB) is denoted by its position vector $\mathbf{r}_i(s,t)$ with s being an arc-length parameterization of the FHB. The tangent vector to the FHB is defined through $\mathbf{r}_i(s,t)$, according to

$$\mathbf{t} = \frac{\partial \mathbf{r}_i}{\partial s}, \tag{2.4}$$

and the normal to the FHB curve is the cross product, $\mathbf{m} = \mathbf{n} \times \mathbf{t}$; by convention we chose s such that $\mathbf{m}$ points in the direction of the quenched region in the corresponding tangent plane to the stoichiometric surface; where the direction normal to Σ_{st} is defined by

$$\mathbf{n} = \left. \frac{\nabla Z}{|\nabla Z|} \right|_{\Sigma_{\mathrm{st}}}. \tag{2.5}$$

The evolution in space and time of the FHB is then governed by the velocity of incoming gases $\mathbf{u}$ projected on the FHB, $\mathbf{u} \cdot \mathbf{m}$, and the displacement speed of the FHB relative to the flow, which is modeled in flame hole dynamics as an edge flame with velocity $V_e(\chi)$, according to

$$\frac{d\mathbf{r}_i}{dt} = [\mathbf{u} \cdot \mathbf{m} + V_e(\chi)]\, \mathbf{m}, \tag{2.6}$$

see Pantano and Pullin [19]. Therefore, the expansion or contraction of the flame hole involves two components: the flow velocity that can advect the FHB and the intrinsic edge-flame speed that couples growth or shrinkage of the flame hole depending on the magnitude of χ. So far, the physical process of FHB propagation has been described in as general terms as possible with the only assumption that $V_e(\chi)$ exists. Further explicit closure for this quantity is required to implement this framework concretely.

The other aspect of flame hole dynamics that needs modeling is the initial formation of a quenched region; the sudden extinction that takes place when $\chi > \chi_{\mathrm{q}}$, which affects the region

$$\Sigma_q = \{\mathbf{x} : \chi(\mathbf{x}, t) > \chi_{\mathrm{q}} \cap \Sigma_{\mathrm{st}}\}. \tag{2.7}$$

It is acknowledged that the interval over which and the degree by which χ exceeds χ_{q} plays a role in determining the unsteady extinction dynamics [32, 33]. Immediately after chemical extinction, there are residual processes that take place and lead to the actual quenching of the flame. During this phase, after the temperature has been reduced below the cross-over temperature for extinction, complete extinction is regulated by diffusion of the remaining thermal energy from the flame hole region [32]. Adopting a more accurate criterion requires simply a change in the first condition in Eq. (2.7).

As can be imagined, the modeling of flame holes using Eq. (2.6) is exceedingly complicated since it requires tracking the evolution and the generation of new holes throughout space in a proper curvilinear coordinate system attached to the stoichiometric surface. Alternatively, we can define a flame state field $\varphi(\mathbf{z}, t)$ that denotes the extent of reactivity at a particular point on the stoichiometric surface, where $\mathbf{z}$ denotes the intrinsic coordinates of the surface (which for the time being do not need to be precisely defined). φ takes values between zero and one, according to

$$\varphi = \begin{cases} \to 1 & \notin \Sigma_q, \\ \frac{1}{2} & \in \partial\Sigma_q = \mathbf{r}_i, \\ \to 0 & \in \Sigma_q. \end{cases} \tag{2.8}$$

In terms of φ, whose equation will be defined more precisely in Section 2.3, extinction can be incorporated by introducing a sink in the governing equation for φ, of the form

$$\left.\frac{D\varphi}{Dt}\right|_{\Sigma_{\text{st}}} = -\dot{Q}(\varphi, \chi), \tag{2.9}$$

where the derivative operator is an appropriate semi-Lagrangian derivative following the flame surface,which will be described next, but always attached to it and $\dot{Q}$ is a sink law that accounts for extinction and needs to be provided; see A.2. In a more precise analysis, extinction can be defined (approximately) in terms of the more physically realistic flame impulse, see Hewson [33]. The criteria establishes that the flame will quench if χ exceeds χ_{q} for a sufficiently large amount of time (precisely defined through an integral); this requires a slightly more general form of the sink in Eq. (2.9) but it can be accommodated without difficulties.

The concepts described above involve a Lagrangian-like dynamic phenomena, Eq. (2.6), with an Eulerian-like process, Eq. (2.9). Furthermore, attempting to solve Eq. (2.6) embedded in a turbulent flow environment quickly becomes impractical due to the bookkeeping complexity of evolution, creation and destruction of FHB. Therefore, it is desirable to recast Eq. (2.6) in a purely Eulerian framework that is compatible with Eq. (2.9) and, more importantly, that can be implemented alongside the hydrodynamics. The main mathematical difficulty here is that φ is really defined on a moving (and usually multiply connected) surface that is defined implicitly and whose total area is not constant. The re-formulation of the flame hole dynamics method allows for a modeled (finite) FHB propagation speed as well as taking into account the effect of the flow and extinction through the use of a level-set-like field defined on a moving curvilinear surface. This flame state field, φ, obeys a PDE describing the flame hole evolution, as described next. The formulation of this PDE and the efficient numerical implementation of such framework are the main goals of the thesis.

2.2 Evolution equations on general surfaces

Let us consider a scalar field φ_s that obeys an evolution equation of the form

$$\frac{\partial \varphi_s}{\partial t} + \mathbf{u}_s \cdot \nabla_s \varphi_s = 0, \tag{2.10}$$

where all quantities are defined on an arbitrary surface Σ (in this section the subscript "st" is not used since we discuss properties valid on arbitrary surfaces), $\mathbf{u}_s$ is a vector defined on the tangent space of Σ, denoted by T_Σ. The differential operator ∇_s represents the proper surface gradient on Σ. The surface gradient can be defined as the orthogonal projection of the Cartesian gradient onto T_S, $\nabla_s = P\nabla$, where P is a projection operator to be defined shortly [21, 25, 34, 35]; also, it can be defined in terms of the directional derivative on the surface [36, 37] or the intrinsic coordinate system [38]. Following Simon [36], let Σ be at least a $\mathcal{C}^1$ two-dimensional submanifold of $\mathbb{R}^3$. Let us introduce the embedding defined by the intrinsic coordinate system $\mathbf{z} \in U \subset \mathbb{R}^2$, $\mathbf{z} \mapsto \mathbf{x} \in \Sigma$ and assume that $\varphi_s(\mathbf{z}, t)$ is a $\mathcal{C}^2 \times \mathcal{C}^1$ function, where the first coordinate denotes the spatial coordinates of the surface and the second coordinate denotes time. By definition, φ_s is a $\mathcal{C}^2$ function on $\mathbf{z}$ only if it is the restriction to Σ of a $\mathcal{C}^2$ function $\varphi : M \times \mathbb{R} \to \mathbb{R}$, with M being an open subset $M \subset \mathbb{R}^3$ such that $\Sigma \subset M$. φ_s is related to φ throughout

$$\varphi_s(\mathbf{z}, t) = \varphi(\mathbf{x}_s, t), \tag{2.11}$$

with $\mathbf{x}_s = \mathbf{x}(\mathbf{z}, t)$ denoting the position vector in the embedding space. Here, it is not assumed that a single coordinate system $\mathbf{z}$ is uniformly valid over all Σ. Instead, as it is done customarily, it is presumed that a set of differentiable coordinate systems or mappings, $\mathbf{z}_k$, exist on overlapping subsets U_k of Σ; these local charts $(\mathbf{z}_k, U_k)$ cover all Σ through an atlas [39]. For notationally simplicity, we denote the coordinate system as $\mathbf{z}$, implying that the corresponding $\mathbf{z}_k$ on the appropriate U_k is being used locally.

2.2.1 Spatial considerations

Let $\mathbf{b}_\alpha$, with $\alpha = 1, 2$, denote the covariant base vectors of the surface, where

$$\mathbf{b}_\alpha = \frac{\partial \mathbf{x}}{\partial z^\alpha}, \tag{2.12}$$

and the normalized counterparts

$$\mathbf{t}_\alpha = \frac{\mathbf{b}_\alpha}{|\mathbf{b}_\alpha|}. \tag{2.13}$$

First, we wish to define an orthogonal projection tensor $P : \mathbb{R}^3 \to T_S$, i.e., that takes an arbitrary vector in $\mathbb{R}^3$ and projects it to T_S (still defined in the coordinates of $\mathbb{R}^3$). Strictly speaking P is defined at each point $\mathbf{x}$ and a notation such as $P_\mathbf{x}$ would be more explicit, but unless it is required to avoid ambiguity we will not denote the explicit dependence of P on $\mathbf{x}$. The orthogonal projection tensor has the properties that $P = P^\intercal$, $P = P^2$ and that it transforms every vector in T_S into itself [38]. Consider now an arbitrary vector $\mathbf{v}$ at a point $\mathbf{x}$ in $\mathbb{R}^3$ and split the vector, according to

$$\mathbf{v} = \mathbf{v}_T + v_n \mathbf{n}, \tag{2.14}$$

where $\mathbf{n} = \mathbf{t}_1 \times \mathbf{t}_2$ is the surface normal and v_n and $\mathbf{v}_T$ denote the normal and tangential components of $\mathbf{v}$ to the surface, respectively. Eq. (2.14) can also be written using $\mathbf{v}_T = P\mathbf{v}$ as

$$\mathbf{v} = (P + \mathbf{n} \otimes \mathbf{n})\mathbf{v}. \tag{2.15}$$

Rearranging terms in Eq. (2.15) and noting that $\mathbf{v}$ is arbitrary leads to

$$P = I - \mathbf{n} \otimes \mathbf{n}. \tag{2.16}$$

One can verify that this projection has the required properties $P = P^\intercal$, $P = P^2$, and so is an orthogonal projection of a Cartesian vector into T_S.

Now, the surface gradient will be defined using the method that employs the directional derivative on the surface in the intrinsic coordinates of Σ [38]. The gradient operator is a linear transformation that given a function φ_s and a particular direction, say an arbitrary tangent space vector

$\mathbf{a} = (a^1, a^2)$, defines the derivative of φ_s along such direction by

$$\mathbf{a} \cdot \nabla_s \varphi_s(\mathbf{z}) \equiv \frac{d\varphi_s}{ds}(\mathbf{z} + s\mathbf{a})\bigg|_{s=0} = \lim_{s\to 0} \frac{\varphi_s(z^1 + s\ a^1, z^2 + s\ a^2) - \varphi_s(z^1, z^2)}{s}, \tag{2.17}$$

where the dependence of φ on time has been omitted for clarity (for the time being). This limit can be expressed in terms of partial derivatives with respect to the surface's intrinsic coordinate system,

$$\begin{aligned} \mathbf{a} \cdot \nabla_s \varphi_s(\mathbf{z}) &= \lim_{s\to 0} \frac{\varphi_s(z^1 + sa^1, z^2 + sa^2) - \varphi_s(z^1, z^2 + sa^2)}{s} \\ &\qquad + \lim_{s\to 0} \frac{\varphi_s(z^1, z^2 + sa^2) - \varphi_s(z^1, z^2)}{s} \\ &= a^1 \lim_{sa^1\to 0} \frac{\varphi_s(z^1 + s\ a^1, z^2 + s\ a^2) - \varphi_s(z^1, z^2 + s\ a^2)}{s\ a^1} \\ &\qquad + a^2 \lim_{sa^2\to 0} \frac{\varphi_s(z^1, z^2 + s\ a^2) - \varphi_s(z^1, z^2)}{s\ a^2} \\ &= a^1 \frac{\partial \varphi_s}{\partial z^1}(z^1, z^2) + a^2 \frac{\partial \varphi_s}{\partial z^2}(z^1, z^2) = a^\beta \frac{\partial \varphi_s}{\partial z^\beta}(\mathbf{z}). \end{aligned} \tag{2.18}$$

In terms of the smoothly extended φ, Eq. (2.11), the definition of the surface gradient for the scalar φ_s can be extended to φ at a point $\mathbf{x}$ on the surface, according to

$$\nabla_s \varphi_s(\mathbf{z}) = \frac{\partial \varphi_s}{\partial z^\beta} \mathbf{b}^\beta = \frac{\partial \varphi}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial z^\beta} \mathbf{b}^\beta = \left(\frac{\partial \varphi}{\partial \mathbf{x}} \cdot \mathbf{b}_\beta \right) \mathbf{b}^\beta = P \nabla \varphi(\mathbf{x}), \tag{2.19}$$

since the scalar product with $\mathbf{b}_\beta$ will take only the part of $\nabla\varphi$ that projects along the covariant direction vectors, and these will then be expressed in the corresponding basis function of $\mathbb{R}^3$, which is the same as the action of P.

2.2.2 Moving surface considerations

In our case, the surface defined in terms of an embedding function, $Z(\mathbf{x}, t)$, according to Eq. (2.1), depends on time. Now, the projection of the regular gradient of φ on T_S can be simplified if φ has the following property:

$$\nabla Z \cdot \nabla \varphi = 0, \quad \forall\ \mathbf{x}_s, \tag{2.20}$$

since this implies

$$P\nabla\varphi = \nabla\varphi. \tag{2.21}$$

We denote by $E(t)$ the extension operator that enforces Eq. (2.20), or equivalently

$$\varphi(\mathbf{x}, t) = E(t)\varphi_s(\mathbf{z}, t) = E(t)\varphi(\mathbf{x}_s, t). \tag{2.22}$$

With these new definitions, we can rewrite Eq. (2.10) according to

$$\frac{\partial \varphi_s}{\partial t} + \mathbf{u}_s \cdot \nabla_s \varphi_s = \frac{\partial \varphi_s}{\partial t} + \mathbf{u}_s \cdot \nabla\varphi = 0, \tag{2.23}$$

and the time-derivative term can be expanded using Eq. (2.11) to

$$\frac{\partial \varphi_s}{\partial t} = \frac{\partial \varphi}{\partial t} + \nabla\varphi \cdot \frac{\partial \mathbf{x}_s}{\partial t}, \tag{2.24}$$

giving

$$\frac{\partial \varphi}{\partial t} + \left(\mathbf{u}_s + \frac{\partial \mathbf{x}_s}{\partial t}\right) \cdot \nabla\varphi = 0. \tag{2.25}$$

It is important to note that the solution of Eq. (2.25) is not guaranteed to match exactly the solution of Eq. (2.10), on Σ at all times; even if φ satisfies Eq. (2.22) at $t = 0$. To ensure consistency, one would require appending Eq. (2.22) to Eq. (2.25). This can be accomplished computationally in several ways. One strategy is to employ a fractional step method where an intermediate φ obtained from Eq. (2.25) applied to the whole three-dimensional domain is followed by enforcement of Eq. (2.22). Another approach is to fuse Eq. (2.22) and Eq. (2.25) using a penalty term, according to

$$\frac{\partial \varphi}{\partial t} + \left(\mathbf{u}_s + \frac{\partial \mathbf{x}_s}{\partial t}\right) \cdot \nabla\varphi = \lambda[E(t)\varphi(\mathbf{x}_s, t) - \varphi(\mathbf{x}, t)], \tag{2.26}$$

where λ is a strength parameter that controls how close is the solution of Eq. (2.26) to Eq. (2.22). This can now be integrated in time using a semi-implicit time integration method, although this has only been demonstrated with stationary surfaces, see Yoh and Zhong [40], Chen and Macdonald [41].

To close Eq. (2.25) one needs the surface speed, $\frac{\partial \mathbf{x}_s}{\partial t}$, which is defined implicitly by Eq. (2.1). One feature of implicitly defined surfaces is that one cannot define a unique velocity at each point of the surface. It is possible, nevertheless, to assign a vector magnitude for a given preselected direction [42]. Assume a surface point moves in direction $\mathbf{d}$ (assumed normalized), then by definition

$$\frac{\partial \mathbf{x}_s}{\partial t} = \mathbf{d}\left|\frac{\partial \mathbf{x}_s}{\partial t}\right|. \tag{2.27}$$

By differentiating Eq. (2.1) with respect to time we have

$$\frac{\partial Z}{\partial t} + \nabla Z \cdot \frac{\partial \mathbf{x}_s}{\partial t} = 0 \quad \Rightarrow \quad \left|\frac{\partial \mathbf{x}_s}{\partial t}\right| = -\frac{\frac{\partial Z}{\partial t}}{\mathbf{d} \cdot \nabla Z}, \tag{2.28}$$

where $\frac{\partial Z}{\partial t}$ must be determined from the governing equation for Z evaluated at Σ_{st}. Now, since the velocity of the flame hole boundary is already decomposed into a component along Σ_{st}, Eq. (2.6), the only meaningful choice for $\mathbf{d}$ is $\mathbf{n}$; note that φ does not change in that direction. This is consistent with the value of φ being dragged to a new location simply due to the deformation of the surface. Introducing $\mathbf{d} = \mathbf{n}$ in Eq. (2.28), one can simplify Eq. (2.25) to

$$\frac{\partial \varphi}{\partial t} + \left(\mathbf{u}_s - \frac{1}{|\nabla Z|}\frac{\partial Z}{\partial t}\mathbf{n}\right) \cdot \nabla \varphi = 0, \tag{2.29}$$

which can be further simplified by account of Eq. (2.20) into the final equation

$$\frac{\partial \varphi}{\partial t} + \mathbf{u}_s \cdot \nabla \varphi = 0. \tag{2.30}$$

Now, the surface equation is expressed completely in terms of the extended φ in the usual three-dimensional space and its gradients and it can be solved using standard numerical techniques. Note that Eq. (2.22) still needs to be imposed. The solution of Eq. (2.30) together with Eq. (2.22) satisfies the surface evolution equation everywhere on Σ. The main achievement here is that the surface evolution problem, Eq. (2.10), has been replaced by a three-dimensional problem in the usual Cartesian space and it does not require a parameterization and discretization of the surface Σ. The implementation details of the extension operator, Eq. (2.22), are further discussed in Section 3.1.

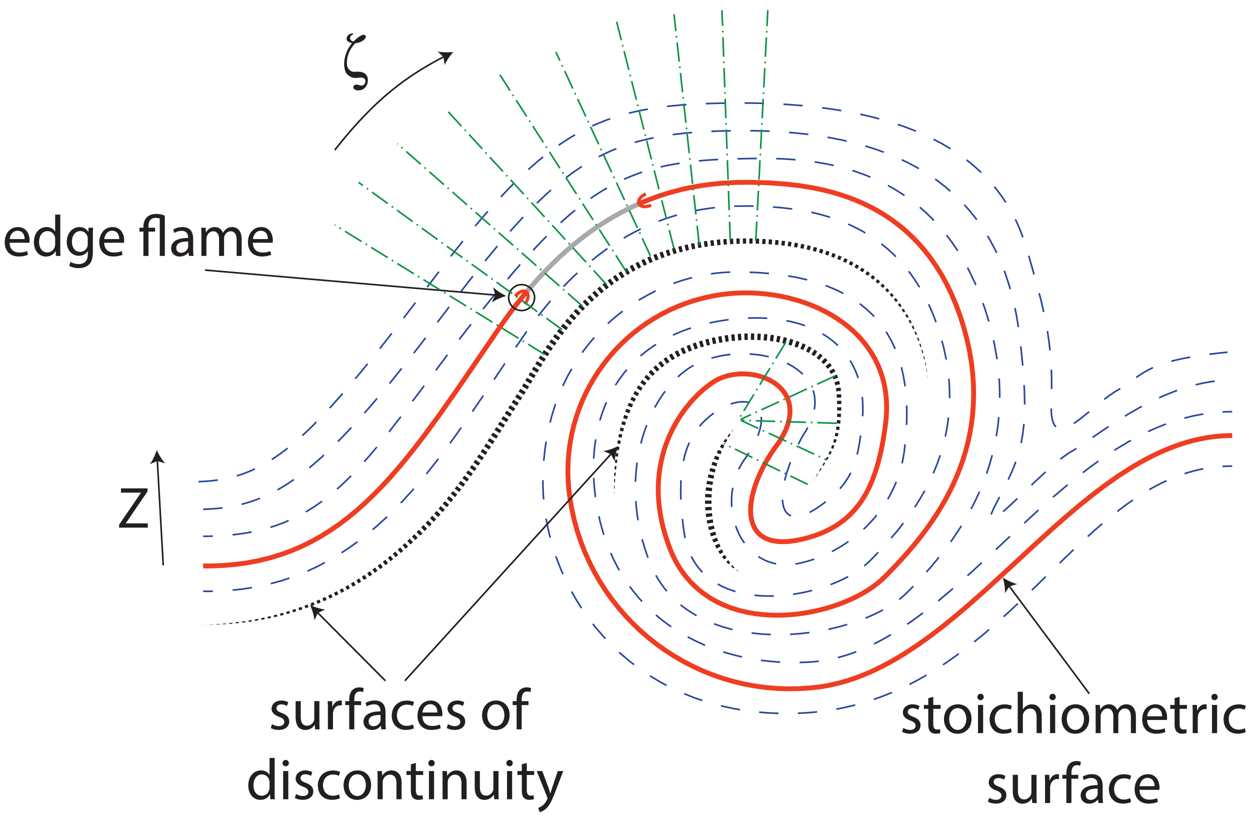


Figure 2.2: Sketch of the coordinate transformation used in the flame hole dynamics formulation. φ is constructed in the orthogonal coordinate system. The surfaces of discontinuity are manifolds where the extension in the orthogonal coordinate becomes multi-valued.

2.3 Transformation of the kinematic flame hole dynamics formulation

Using the final result of the previous section, Eq. (2.30), it is possible to re-cast the Lagrangian formulation, Eq. (2.6), describing the evolution of flame hole boundary into a purely Eulerian formulation in the fixed Cartesian space. The kinematic equation for φ is obtained from Eq. (2.6), by observing that the characteristic curves of Eq. (2.30) must equal dr_i/dt, if the level set $\varphi = 1/2$ is to move according to Eq. (2.6). Now, combining this observation with the flame hole boundary normal relationship to φ, given by

$$\mathbf{n}_\varphi = \frac{\nabla\varphi}{|\nabla\varphi|} = -\mathbf{m}, \tag{2.31}$$

according to the earlier convention of $\mathbf{m}$ pointing into the quenched region, and incorporating Eq. (2.9), we have

$$\mathbf{u}_s = (\mathbf{u} \cdot \mathbf{n}_\varphi)\, \mathbf{n}_\varphi - V_e(\chi)\mathbf{n}_\varphi, \tag{2.32}$$

which can be simplified using $\nabla\varphi \cdot \nabla\varphi = |\nabla\varphi|^2$, leading to the master flame hole dynamics Eulerian equation

$$\frac{\partial\varphi}{\partial t} + \mathbf{u} \cdot \nabla\varphi = V_e(\chi)|\nabla\varphi| - \dot{Q}(\varphi, \chi). \tag{2.33}$$

Note that, to be precise, one must understand the right-hand side of Eq. (2.33) as $E(t)(V_e|\nabla\varphi| - \dot{Q})$ because χ can (and generally does) vary along $\mathbf{n}$. Computationally, this projection is unnecessary when using a fractional-step approach because Eq. (2.22) is always enforced at the end of each time-step. Furthermore, related equations of true interfacial dynamics are different from Eq. (2.33) because the flame state field is not a physically conserved quantity. In the former, additional terms must appear due to the requirements of conservation of mass or momentum on the surface [43]. Figure **??** is a sketch describing the coordinate transformation used in the construction of φ. The flame state variable is constructed such that it is extended orthogonally in the "Z" direction of the sketch, with the edge flame dynamics defined in the "ζ" direction.

Solving Eq. (2.33) while enforcing Eq. (2.22) in an efficient manner requires a careful numerical strategy to resolve well all the physical processes that this equation describes: flame hole formation, propagation and advection on the moving Σ_{st}; and it is the main contribution of this thesis.

Chapter 3

Algorithms

3.1 The extension operator

The extension operator, $E(t)$, takes information on Σ_{st} and propagates it smoothly into the surrounding three-dimensional domain along the direction defined by ∇Z; the basic condition that must be enforced is Eq. (2.20). We choose a hybrid method for the extension. The extension is done in two bands: a closest point extension is applied in a band near the surface and a PDE-type extension (either based on an elliptic-type operator or an anisotropic diffusion-type operator) is applied outside of that band, using values from the closest point extension [26, 27, 28, 30] as the interior Dirichlet condition. This mixture of methods allows us to use a high-order extension close to the surface, where the accuracy of the surface evolution is affected, and a faster but less accurate extension where the solution does not affect the accuracy of the surface evolution.

3.1.1 Closest point extension

For the closest point extension, a band is formed around the surface such that all points are within a distance of $n_b \max(\Delta x, \Delta y, \Delta z) = n_b \Delta_{\max}$ from the surface in any direction, where n_b, discussed later in this section, is some positive integer that sets the width of the band. Within this band, the closest point on the surface to a grid point $x_{i,j,k}$ is calculated by use of a local, quadratic approximation to Σ_{st} as in Leung et al. [30], given by

$$Z_{LSQ,2}(\mathbf{x}) = a + \mathbf{b}^{\intercal}\mathbf{x} + \frac{1}{2}\mathbf{x}^{\intercal}\mathbf{H}\mathbf{x}, \tag{3.1}$$

where the coefficients of the approximation—a, $\mathbf{b}$, and $\mathbf{H}$—are obtained through a local least squares approximation using the neighboring points; i.e., for a point $\mathbf{x}_j$, a cell of neighboring points $\mathbf{x}_j + n(\Delta x, \Delta y, \Delta z)$, with n ranging between -1 and 1, are used to form the least squares system,

$$\mathbf{A}_{\text{LSQ}}\ell = \mathbf{z}. \tag{3.2}$$

Equivalently, the least squares system is

$$\begin{pmatrix} 1 & x_0 & y_0 & z_0 & \frac{1}{2}x_0^2 & x_0y_0 & x_0z_0 & \frac{1}{2}y_0^2 & y_0z_0 & \frac{1}{2}z_0^2 \\ 1 & x_1 & y_1 & z_1 & \frac{1}{2}x_1^2 & x_1y_1 & x_1z_1 & \frac{1}{2}y_1^2 & y_1z_1 & \frac{1}{2}z_1^2 \\ & & & & \vdots & & & & & \\ 1 & x_n & y_n & z_n & \frac{1}{2}x_n^2 & x_ny_n & x_nz_n & \frac{1}{2}y_n^2 & y_nz_n & \frac{1}{2}z_n^2 \end{pmatrix} \begin{pmatrix} \ell_0 \\ \ell_1 \\ \vdots \\ \ell_n \end{pmatrix} = \begin{pmatrix} Z(x_0, y_0, z_0) \\ Z(x_1, y_1, z_1) \\ \vdots \\ Z(x_n, y_n, z_n) \end{pmatrix}. \tag{3.3}$$

The least squares cell is weighted, at a point $\mathbf{x}_j$ in the least squares system, with an approximation of the distance using the mixture fraction,

$$\mathbf{W}_{Z,j} = \frac{\min(|Z(\mathbf{x}_j) - Z_{\text{st}}|)}{|Z(\mathbf{x}_j) - Z_{\text{st}}|}. \tag{3.4}$$

The weighted least squares is then

$$\mathbf{W}_{\text{LSQ}}\mathbf{A}_{\text{LSQ}}\ell = \mathbf{W}_{\text{LSQ}}\mathbf{z}, \tag{3.5}$$

where

$$\mathbf{W}_{\text{LSQ}} = \begin{pmatrix} \sqrt{w_{Z,0}} & 0 & \cdots & 0 \\ 0 & \sqrt{w_{Z,1}} & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & 0 & \sqrt{w_{Z,n}} \end{pmatrix}. \tag{3.6}$$

With the coefficients of the least squares approximation defined, the distance between the grid point and the surface is then minimized by moving along the coordinate normal to the quadratic approximation of the surface,

$$\mathbf{cp}(\mathbf{x}_j) = \mathbf{x}_j - \eta \nabla Z_{LSQ}, \tag{3.7}$$

with η chosen such that $Z(\mathbf{cp}(\mathbf{x})) = Z_{\text{st}}$. To find η, we use a Newton iteration to solve

$$f(\eta) = \det(\mathbf{I} + \eta\mathbf{H})^2(a - Z_{\text{st}}) + \det(\mathbf{I} + \eta\mathbf{H})\mathbf{b}^\intercal\mathbf{B}\zeta + \frac{1}{2}(\mathbf{B}\zeta)^\intercal\mathbf{H}\left(\mathbf{B}\zeta\right) = 0, \tag{3.8}$$

with $\zeta = \mathbf{x}_j - \eta\mathbf{b}$ and $\mathbf{B} = \det(\mathbf{I} + \eta\mathbf{H})(\mathbf{I} + \eta\mathbf{H})^{-1}$. The derivative of this function can be written in matrix-vector form as

$$f'(\eta) = 2\det(\mathbf{I} + \eta\mathbf{H})\det(\mathbf{I} + \eta\mathbf{H})'(a - Z_{\text{st}}) + \det(\mathbf{I} + \eta\mathbf{H})\mathbf{b}^\intercal\mathbf{q}' + \det(\mathbf{I} + \eta\mathbf{H})'\mathbf{b}^\intercal\mathbf{q} + \mathbf{q}^\intercal\mathbf{H}\mathbf{q}', \tag{3.9}$$

with $q = \mathbf{B}\zeta$ and $q' = \mathbf{B}'\zeta - \mathbf{B}\mathbf{b}$. The value of η is then computed iteratively to convergence by,

$$\eta_{n+1} = \eta_n - \alpha_D\frac{f(\eta)}{f'(\eta)}, \tag{3.10}$$

where α_D is a damping coefficient chosen to be 0.5. In practice, the coefficients of $f(\eta)$, although lengthy and cumbersome, can be calculated analytically and the derivative can be evaluated by taking the derivative of each term of the resulting sixth-order polynomial. A third-order method, while not requiring a larger cell (as the number of coefficients is 20, which is still less than the 27-point cell used), does not have an analytical form for the coefficients or even for the coordinate transformation into the normal coordinate. A third-order method would require solving a direct minimization problem to determine the minima of the least squares polynomial instead of the polynomial root-finding required for the quadratic local least squares (or, a polynomial root solving algorithm can be constructed with numerically determined coefficients that now depend on η). The root-finding can be done by the aforementioned Newton method, which in practice converges quickly to what is typically the closest point. However, in some cases, the Newton method fails to converge or converges to a root of the polynomial that does not correspond to the closest point. For this reason, a eigenvalue-based polynomial root-finding calculation was also added. We introduce a matrix, termed the companion matrix, whose eigenvalues are the roots of the polynomial. The coefficients of the matrix, c_n, are the coefficients of the 6^{th}-order polynomial and are calculated analytically by expanding Eq. (3.8). The form of the characteristic matrix is then

$$C = \begin{pmatrix} -c_1/c_0 & -c_2/c_0 & -c_3/c_0 & -c_4/c_0 & -c_5/c_0 & -c_6/c_0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \tag{3.11}$$

The eigenvalues of the companion matrix correspond with the zeros of the polynomial, and so solving for the eigenvalues of C is equivalent to finding the roots of the polynomial. This technique, while much slower than the Newton method, allows for robust checking of multiple roots for the closest point extension. The direction toward the surface is calculated using the gradient of the local least squares (parameterized by η) and the signum function,

$$a_{\alpha,\mathrm{LSQ}} = \mathrm{sgn}(Z - Z_{\mathrm{st}})\eta_{\alpha}, \tag{3.12}$$

where α denotes a root of the polynomial. The smallest root such that $a_{\alpha,\mathrm{LSQ}} > 0$ is chosen for the closest point.

For a general surface, the closest point is almost never a grid point of the computational domain, and therefore an estimate of $\varphi(\mathbf{cp}(\mathbf{x}_j))$ is also required. We use a weighted least squares approximation of φ centered around the nearest point to $\mathbf{cp}(\mathbf{x}_j)$ on the grid,

$$\varphi_{LSQ,2}(\mathbf{cp}(\mathbf{x}_j)) = \hat{a} + \hat{\mathbf{b}}^{\intercal}\mathbf{cp}(\mathbf{x}_j) + \frac{1}{2}\mathbf{cp}(\mathbf{x}_j)^{\intercal}\hat{\mathbf{H}}\,\mathbf{cp}(\mathbf{x}_j), \tag{3.13}$$

where the hat indicates that the coefficients are calculated based on values of φ with weighting,

$$\mathbf{W}_{\varphi,j} = \frac{\min(|\mathbf{x}_j - \mathbf{cp}(\mathbf{x}_j)|)}{|\mathbf{x}_j - \mathbf{cp}(\mathbf{x}_j)|}. \tag{3.14}$$

Since we are interested in the application of the algorithm to very complicated surfaces, we find it necessary to incorporate certain regularizations or numerical entropy solution strategies to obtain a robust functioning of the method. These corrections are not often activated in a typical domain, but when they are, they enable calculation on otherwise impossible surface configurations that are not statistically significant but must be treated satisfactorily. As this approximation is not total variation diminishing, we limit value produced by this approximation to be within the range of values used in the least squares system. The right-hand side of flame hole dynamics equation, Eq. (2.33), is then computed on each point used in the least squares approximation to $\varphi(\mathbf{cp}(\mathbf{x}))$ in order to advance the solution in time. The width of the band is chosen such that the closest point extension is computed at all points used to evaluate the righthand side of Eq. (2.33). For cases with highly varying Z fields, like the shear layer in figure 4.11), the Newton method Eq. (3.10) can sometimes fail to converge. At these points, a linear approximation to the surface is employed instead, given by

$$Z_{LSQ,1}(\mathbf{x}) = a_1 + \mathbf{b}_1^{\intercal}\mathbf{x}, \tag{3.15}$$

where a_1 and b_1 represent the least square coefficients for the linear approximation. The closest point can be found exactly as

$$\mathbf{cp}(\mathbf{x}_j) = \mathbf{x}_j - (a - Z_{\mathrm{st}})\frac{\mathbf{b}_1}{\mathbf{b}_1^{\intercal}\mathbf{b_1}} \tag{3.16}$$

and the resulting estimate for the closest point can be used to evaluate $\varphi_{LSQ,2}(\mathbf{cp}(\mathbf{x}))$. We additionally test whether the linear approximation gives a reasonable solution by checking if $|\mathbf{cp}(\mathbf{x}) - \mathbf{x}_j| > n_b\Delta_{\mathrm{max}}$. If it fails this test, then value of φ at that point is set equal to the average of neighboring points for which the closest point extension passes this test.

The width of the band, n_b, is chosen based on the width of the cell used in the least squares approximation of φ and the width of the stencil used to solve Eq. (2.33). The flame hole dynamics equation is solved at every point used in the estimation of $\varphi_{LSQ,2}(\mathbf{cp}(\mathbf{x}_j))$, and the band width is chosen such that every point used has been extended through the closest point method. The stencil width is therefore the maximum distance a point in the least squares cell for φ can be from the surface ($2\Delta_{\mathrm{max}}$) in any direction, plus the width of stencil discretizing Eq. (2.33) ($3\Delta_{\mathrm{max}}$), resulting in a value of $n_b = 5$. This choice of band width is consistent with the analysis of band width for the closest point method in Macdonald and Ruuth [27].

3.1.2 Elliptic-type extension operator

The closest point extension, while higher order, is both expensive and inaccurate at points far away from the surface. In order to extend the field into a large band in the domain, a PDE-based extension is used. This type of extension has desirable properties [44, 45] for solving the extension in a very large band around the surface. They rely on solving a PDE that satisfies the extension condition Eq. (2.20), achieved by the steady solution of

$$\frac{\partial \varphi}{\partial \tau} + \operatorname{sgn}(Z - Z_{\mathrm{st}})\mathbf{n} \cdot \nabla \varphi = 0, \tag{3.17}$$

where τ is a pseudo-time variable that has only a transitory role as the equation is solved until the solution is stationary. In this method, there is no difficulty at the boundaries of the computational domain and simple outflow conditions for the scalar advection equation can be used. However, the computational effort associated with this method is significant because one needs to take many pseudo-time steps for the solution to Eq. (3.17) to become stationary. This can be avoided solving directly for the modified steady-state version of Eq. (3.17), given by

$$\operatorname{sgn}(Z - Z_{\mathrm{st}})\tilde{\mathbf{n}} \cdot \nabla \varphi = \nabla \cdot (\varepsilon(\mathbf{x}) \nabla \varphi), \tag{3.18}$$

where

$$\tilde{\mathbf{n}} = \frac{\nabla Z}{|\nabla Z| + g_\epsilon}, \tag{3.19}$$

is a regularized approximation of the Σ_{st} normal required to ensure good behavior when approaching critical points in Z or the free streams (where Z is constant); further discussion on this regularization follows below. The parameter $\varepsilon(\mathbf{x})$ is introduced to ensure stability and it scales with $\Delta_{\mathrm{avg}} = (\Delta x \Delta y \Delta z)^{1/3}$ so that Eq. (3.18) and Eq. (3.17) are consistent and Eq. (3.18) converges to the same solution under refinement. The implementation of the method is a generalization of a recent implicit extension algorithm developed in Uddin et al. [46], modified to ensure the idempotence of the extension operator; that is $E^2 = E$.

Problems with this method can occur in practical applications when there may be regions of the domain where $|\nabla Z|$ approaches zero (a case not considered in figure 4.2). There are two typical scenarios when this can occur: in unmixed regions of a flow such as at the edge of a shear layer or at the outer extent of a jet flow, or at critical points in a turbulent flow. Each scenario is handled differently. To ensure stability of the method for Z fields containing critical points, $|\nabla Z| \approx 0$, a proper choice of ε, related to the regularization used in Eq. (3.19), must be used in Eq. (3.18). Our numerical experiments were successful using the following strategy. In the first phase, an indicator function, I_ε, was defined according to

$$I_\varepsilon = \varepsilon_0 \left[\Delta_{\text{avg}} + \ell \bigg(\frac{g_\epsilon}{g_\epsilon + |\nabla Z|} \bigg) \right], \tag{3.20}$$

where ℓ denotes the characteristic lengthscale of variation of ∇Z near critical points and g_ϵ denotes the characteristic cutoff scale for good behavior of $\mathbf{n}$, a small fraction of $|\nabla Z|$ overall. The intention of Eq. (3.20) is to ensure that ε behaves as Δ_{avg} for large values of ∇Z, while it approaches ℓ when ∇Z is very small (it is assumed that $\Delta_{\text{avg}} \ll \ell$ if Z is well-resolved by the mesh). The constant ε_0 is independent of the grid spacing or Z, while g_ϵ and ℓ depend on the degree of variation of ∇Z in the domain (these parameters can be selected reasonably well by an initial analysis of Z throughout the domain). In the second phase, the indicator function is smoothed over several grid cells by the application of a filter of width $2\Delta_{\text{avg}}$, giving

$$\varepsilon(\mathbf{x}) = \left(\frac{1}{8\pi\Delta_{\text{avg}}^2} \right)^{3/2} \iiint I_\epsilon(\boldsymbol{x} - \boldsymbol{x}') \exp\left(\frac{-|\boldsymbol{x} - \boldsymbol{x}'|^2}{8\Delta_{\text{avg}}^2} \right) d\boldsymbol{x}', \tag{3.21}$$

The resulting ε field is smooth and localized around critical points. This ensures the stability of the method while still limiting the effect diffusion has on φ.

The previous selection of ε resolves the treatment of critical points. For unmixed regions, where Z is 0 or 1, there is no flame and the value of φ can be set to zero safely because it does not affect the calculation of the state of the gas mixture. Therefore, the extension is only performed on regions where mixing is present: $0 + Z_\epsilon = Z_- < Z < Z_+ = 1 - Z_\epsilon$, where we have used $Z_\epsilon = 0.001$ satisfactorily in all our simulations. A Robin condition is used at this boundary, $Z = Z_\pm$, to ensure stability of the extension algorithm when disconnected regions of slightly mixed fluid near the exterior boundary are present or form during a simulation (multiply connected case), a scenario that occurs in practice in turbulence simulations when detachment of mixture fraction blobs from the main turbulence region takes place. Now, the Robin boundary condition is given by

$$\alpha_R \varphi + \frac{\partial \varphi}{\partial n} = 0, \tag{3.22}$$

where n is the normal direction to the boundary, and α_R is a small parameter that determines the relative weight of the exterior boundary data in the boundary condition; note that for α_R sufficiently small, the boundary condition $\varphi = 0$ is effective only when the normal gradient is negligible (as intended).

3.1.3 Anisotropic diffusion-type extension operator

Information on the stoichiometric surface is propagated along the normal to the mixture fraction field. The elliptic-type extension, Eq. (3.17), extends the flame state field into the three-dimensional domain instantaneously. The instantaneous extension might not be realistic in some flows because the state of the flame near the stoichiometric surface takes a finite time to reach other regions far removed from Z_{st}. This limitation can be improved by an alternative approach to extension.

An anisotropic diffusion equation can also be developed to extend the flame state field beyond the band where the closest point extension is used. The anisotropic diffusion is the gradient descent equation minimizing the L_2 norm of the extension condition,

$$F(\varphi) = \frac{1}{2} \int_\Omega |\mathbf{n}_Z \cdot \nabla \varphi|^2 dx. \tag{3.23}$$

The Gâteaux derivative, with respect to $\nu \in H^1(\Omega)$ is

$$\begin{aligned} D_\nu F(\varphi) &= \lim_{\epsilon \to 0} \frac{F(\varphi + \epsilon \nu) - F(\varphi)}{\epsilon} & (3.24)\\ &= \lim_{\epsilon \to 0} \frac{1}{2\epsilon} \left(\int_\Omega \mathbf{n}_Z \cdot \nabla(\varphi + \epsilon \nu) d\Omega - \int_\Omega \mathbf{n}_Z \cdot \nabla \varphi d\Omega \right) & (3.25)\\ &= \lim_{\epsilon \to 0} \frac{1}{2\epsilon} \int_\Omega 2\epsilon (\mathbf{n}_Z \cdot \nabla \varphi)(\mathbf{n}_Z \cdot \nabla \nu) d\Omega & (3.26)\\ &= \int_\Omega \nabla \nu \cdot (\mathbf{n}_{\mathbf{Z}} \cdot \nabla \varphi) \mathbf{n}_Z d\Omega. & (3.27) \end{aligned}$$

Using the Gauss divergence theorem, we have

$$D_\nu F(\varphi) = \int_S \nu(\nabla\varphi \cdot \mathbf{n}_Z)\mathbf{n}_Z \cdot \mathbf{n}_S dS - \int_\Omega \nu[\nabla \cdot \big((\mathbf{n}_Z \otimes \mathbf{n}_Z) \cdot \nabla\varphi\big)]d\Omega. \tag{3.28}$$

Taking the domain Ω to be arbitrarily large and noting that $D_\nu F(\varphi)$ is linear in ν, the gradient descent equation that minimizes $F(\varphi)$ is

$$\frac{\partial\varphi}{\partial t} = \nabla \cdot \big((\mathbf{n}_Z \otimes \mathbf{n}_Z) \cdot \nabla\varphi\big). \tag{3.29}$$

The dyadic product $\mathbf{n}_Z \otimes \mathbf{n}_Z$ is a symmetric positive semidefinite tensor,

$$\mathbf{u}^\intercal(\mathbf{n}_Z\mathbf{n}_Z^\intercal)\mathbf{u} = (\mathbf{n}_Z^\intercal\mathbf{u})^2 \geq 0, \tag{3.30}$$

and therefore the anisotropic diffusion is well-posed. However, to avoid ambiguities with degenerate regions where $\mathbf{n}_Z \approx \mathbf{0}$, we add a small isotropic diffusion correction to the gradient descent. Additionally, we chose the anisotropic diffusion to have a large diffusivity coefficient D,

$$\frac{\partial\varphi}{\partial t} = \nabla \cdot \left[D\bigg(\epsilon_D\mathbf{I} + \mathbf{n}_Z \otimes \mathbf{n}_Z\bigg) \cdot \nabla\varphi\right], \tag{3.31}$$

with ϵ_D small (chosen always to be $\epsilon_D = 0.01$ coefficient for the purposes of this thesis) added to ensure that the diffusion tensor is symmetric positive definite.

The anisotropic diffusion extension was implemented into the unstructured solver (see Section 6.6).

3.2 Anti-dissipative correction

To prevent the artificial smearing of φ on the stoichiometric surface, due to numerical dissipation, we add an antidiffusive regularization to the governing equation, Eq. (2.33)—which we call a compression term [47, 48]—based on the idea that the equilibrium solution of

$$\frac{\partial\varphi}{\partial\tau} = \mathcal{U}_0\mathcal{M}(\varphi), \tag{3.32}$$

where

$$\mathcal{M}(\varphi) = \mathbf{n}_{\varphi} \cdot \nabla\bigg(\epsilon_c|\nabla\varphi| - \varphi(1-\varphi)\bigg). \tag{3.33}$$

with $\mathcal{U}_0$ large, converges to

$$\varphi = \frac{1}{2}\bigg(1 + \tanh\left(\frac{n}{2\epsilon_c}\right)\bigg), \tag{3.34}$$

with n denoting the spatial coordinate normal to the $\varphi = 1/2$ surface and ϵ_c is a parameter that determines the interface thickness (usually chosen proportional to the grid spacing) and the value of $\mathcal{U}_0$ chosen as the maximum of the norm of the velocity,

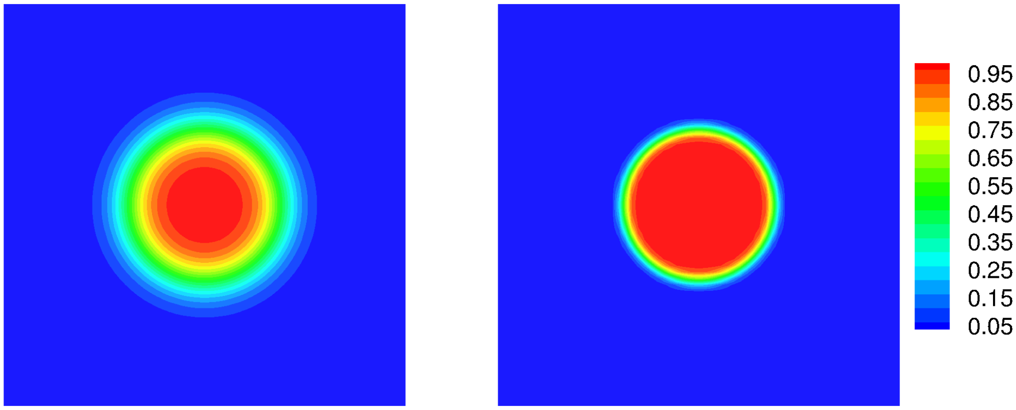
$$\mathcal{U}_0 = \max_{i,j,k}\left(\sqrt{u_{i,j,k}^2 + v_{i,j,k}^2 + w_{i,j,k}^2}\right). \tag{3.35}$$

Figure 3.1 shows an example of the effect of adding compression regularization to the advection of an initially diffuse cylindrical flame hole in a flat surface. The diffuse interface is compressed to a specified width (here, two grid cells wide) by adding the right-hand side of Eq. (3.32) to Eq. (4.2). Note that the initially very diffuse flame hole becomes sharp after the first time-step since Eq. (3.32) is allowed to take enough sub-steps to fully compress the interface. This strategy yields sharp but mesh-representable transitions between the burning and quenched states. Since the compression is acting all along the evolution of the governing equation, we find that it is typically only necessary to perform one or two compression sub-cycles at each time step to keep the flame hole boundary sharp. Furthermore, compression does not seem to affect merging or spliting of flame holes in any measurable way in all the tests we conducted.

When ϵ_c is chosen to be small, such as in our case, the compression of the interface leads to difficulty in resolving the normal. To improve the computation of the normal in this case, we use a variable ψ that as has identical normal to φ but is stretched and allows a better estimate of the gradient:

$$\psi = \frac{\varphi^{\alpha}}{\varphi^{\alpha} + (1-\varphi)^{\alpha}}. \tag{3.36}$$

The parameter, α, is a value between 0 and 1 that controls the thickness of the ψ interface. If φ is a hyperbolic tangent, then ψ is also a hyperbolic tangent with a thickness $1/\alpha$ that of φ. In accordance with Shukla et al. [48], a value of 0.1 is chosen in order to balance the enhanced



(a) Initial condition

(b) After compression

Figure 3.1: Compression algorithm effect in the structured implementation (Section 4.4) on a circular flame disk on a rectangular 64 × 64 × 2 domain, with parameters $\mathcal{U}_0 = 1$ and $\epsilon_c = \max(\Delta x, \Delta y, \Delta z) = \frac{1}{63}$, and no flow velocity. A sub-CFL number of 0.025 is used until a convergence of $\times 10^{-2}$ is obtained. The thickness of the initial profile is reduced from $8\Delta x$ to $2\Delta x$ through the compression. A consistent color scale from 0 to 1 is used for future images of φ.

resolution of the interface with the increased round-off errors associated with the introduction of the auxiliary variable ψ. The gradient $\nabla\varphi$ can then be calculated by

$$\nabla\varphi = \frac{\nabla\psi}{\alpha}(\varphi(1-\varphi))^{1-\alpha}(\varphi^\alpha + (1-\varphi)^\alpha)^2, \tag{3.37}$$

and the normal can be calculated as

$$\mathbf{n}_\varphi = \frac{\nabla\varphi}{|\nabla\varphi|} = \frac{\nabla\psi}{|\nabla\psi|}. \tag{3.38}$$

This procedure is implemented in both the WENO (see Section 4.4)and unstructured (see Section 6.8) implementations.

Chapter 4

High-Order Method for Solving the Flame Hole Dynamics Equation

4.1 Numerical scheme

The numerical implementation of the equations describing flame hole dynamics utilizes several methods. The advection-like term in Eq. (2.33) is split into a divergence-like and a dilatational-like term which are treated with two different numerical methods: a conservative finite volume scheme is applied to the divergence-like term and a modified Gaussian quadrature integration is applied to the dilatational term (Section 4.2.3). A high-order WENO finite-volume formulation is used to discretize Eq. (2.33). WENO interpolation is used to evaluate the surface and volume integrals in the discretization (Section 4.2) and the discrete equations are integrated in time using a 3$^{\text{rd}}$-order Runge-Kutta (RK3) scheme with a special antidiffusive term (or compression) that ensures that the thickness of φ is not smeared over time because of numerical diffusion; see Olsson and Kreiss [47], Shukla et al. [48]. The flame hole formation term (Section 2.1) is treated using a standard splitting method. Finally, the enforcement of Eq. (2.22) uses the extension algorithm, $E(t)$, described in Section 3.1.

4.2 Discretization of the governing equation

A finite-volume discretization is formulated by integrating the governing equation over a control volume, I_j, which is always taken to be a cuboid in our current implementation but could be generalized without much difficulty to other mesh geometries using already available techniques. In the finite-volume approach one solves for the cell-averaged variable, defined by

$$\bar{\varphi}_j = \frac{1}{\mathrm{V}_j} \iiint_{I_j} \varphi d\mathbf{x}, \tag{4.1}$$

where V_j is the volume of cell I_j. The cell-averaged form of Eq. (2.33) with the advection term split into a divergence and a dilatational term is then given by

$$\frac{d\bar{\varphi}_j}{dt} + \frac{1}{\mathrm{V}_j}\iint_{S_j} \varphi\ \mathbf{u}\cdot\mathbf{n}_j dS = \frac{1}{\mathrm{V}_j}\iiint_{I_j} \varphi\ \nabla\cdot\mathbf{u} d\mathbf{x} + \frac{1}{\mathrm{V}_j}\iiint_{I_j} V_e(\chi)|\nabla\varphi| d\mathbf{x} - \frac{1}{\mathrm{V}_j}\iiint_{I_j} \dot{Q}(\varphi,\chi) d\mathbf{x}, \tag{4.2}$$

where S_j denotes all the surfaces of the cell I_j and $\mathbf{n}_j$ the corresponding surface normal vector. The approach used here whereby the advection term is casted into divergence and dilatational terms is by no means unique but it has been used successfully by many others in the past, see for example Johnsen and Colonius [49]. Another approach is to use a numerical method for non-conservation laws, in general formulated in terms of path integrals, see LeFloch [50], but we found that this alternative strategy (possibly more mathematically elegant) was not strictly necessary to get good results. Therefore, we proceed as follows: the divergence-like term is approximated by a 5^{th}-order accurate WENO scheme with 2-point Gaussian quadrature [51] while the dilatational-like term employs a specially conceived bilinear finite-volume integration scheme using a 4^{th}-order WENO reconstruction and a modified Gaussian quadrature, described below. The volume-averaged flame state field, $\bar{\varphi}_j$, is advanced in time using a 3^{rd}-order TVD Runge-Kutta method, developed by Gottlieb and Shu [52].

4.2.1 Divergence-like advection terms

The divergence part of the advection term is approximated following the method of Titarev and Toro [51], which is reproduced here partially because the new discretization method for the dilatational term—first term in the right-hand side of Eq. (4.2)—employs elements of the WENO formulation. The surface integral term in Eq. (4.2) is decomposed into integrals for each of the six faces of the finite-volume cell, according to

$$\frac{1}{\mathrm{V}_j}\iint_{S_j} \varphi\ \mathbf{u}\cdot\mathbf{n}_j dS = \frac{F_{i+\frac{1}{2},j,k} - F_{i-\frac{1}{2},j,k}}{\Delta x} + \frac{G_{i,j+\frac{1}{2},k} - G_{i,j-\frac{1}{2},k}}{\Delta y} + \frac{H_{i,j,k+\frac{1}{2}} - H_{i,j,k-\frac{1}{2}}}{\Delta z}. \tag{4.3}$$

where the x-flux term $F_{i+\frac{1}{2},j,k}$ denotes the surface integral

$$F_{i+\frac{1}{2},j,k} = \frac{1}{\Delta y \Delta z}\iint u(x_{i+\frac{1}{2}},y,z)\varphi(x_{i+\frac{1}{2}},y,z) dydz, \tag{4.4}$$

with u denoting the x component of the vector $\mathbf{u}$; similarly for the fluxes G and H along the other directions. This flux is approximated numerically through a four-point, two-dimensional Gaussian quadrature on the cell face, given by

$$F_{i+\frac{1}{2},j,k} \approx \sum_{\alpha=0}^{1}\sum_{\beta=0}^{1} w_\alpha w_\beta F^*\left(\varphi^L(x_{i+\frac{1}{2}},y_\alpha,z_\beta),\varphi^R(x_{i+\frac{1}{2}},y_\alpha,z_\beta);u^L(x_{i+\frac{1}{2}},y_\alpha,z_\beta),u^R(x_{i+\frac{1}{2}},y_\alpha,z_\beta)\right), \tag{4.5}$$

where the weights of the Gaussian quadrature are denoted by w_α and w_β and y_α and z_β denote the Gaussian quadrature collocation points on the cell face in the y and z coordinates, respectively. The quantities $\varphi^L(x_{i+1/2}, y_\alpha, z_\beta)$ and $\varphi^R(x_{i+\frac{1}{2}}, y_\alpha, z_\beta)$ denote the 'left' and 'right' extrapolations to the cell face located at $x_{i+1/2}$, respectively. The extrapolations for φ^L and φ^R are obtained through WENO reconstruction whereas those of u^L and u^R use WENO interpolation; see B.3. The numerical flux, F^*, is obtained from the multistage upwind method (MUSTA), an iterative approximate Riemann solver that is discussed in greater detail in B.1 [51, 53, 54, 55]; see Section 4.6 for test cases involving the WENO-MUSTA method.

4.2.2 Propagation and source terms

The discretization for the flame hole boundary propagation term, $\mathcal{N} = V_e(\chi)|\nabla\varphi|$, is handled by a Hamilton-Jacobi WENO procedure, see Fedkiw et al. [56]. We approximate the edge term by evaluating a volume integral through a Gauss quadrature,

$$\frac{1}{V_j}\iiint_{I_j} \mathcal{N} d\mathbf{x} = \sum_{\alpha=0}^{1}\sum_{\beta=0}^{1}\sum_{\gamma=0}^{1} w_\alpha w_\beta w_\gamma \mathcal{N}^{\text{num}}(x_\alpha, y_\beta, z_\gamma), \tag{4.6}$$

where

$$\mathcal{N}^{\text{num}}(x_\alpha, y_\beta, z_\gamma) = V_e(\chi_{\alpha\beta\gamma})\sqrt{(\varphi_x^{\text{up}})^2_{\alpha\beta\gamma} + (\varphi_y^{\text{up}})^2_{\alpha\beta\gamma} + (\varphi_z^{\text{up}})^2_{\alpha\beta\gamma}}. \tag{4.7}$$

Right- and left-sided approximations to the derivative (φ_x^+ and φ_x^- respectively) are calculated for use in an upwinding procedure to numerically estimate φ_x^{up}. We use a fourth-order, dimensionally split WENO reconstruction to evaluate the derivative (see B.3) at the Gauss quadrature points. In the x-direction, for example, a stencil based around $x_{i,j,k}$ is used to calculate the right-sided approximation of the derivative at $\varphi_x^+(x_i + \Delta x\alpha_1, y_\beta, z_\gamma)$ (where $x_i + \Delta x\alpha_1$ is the x-coordinate of a Gauss quadrature point for the cell I_j) and a stencil based around $x_{i+1,j,k}$ is used to calculate the left-sided approximation, $\varphi_x^-(x_{i+1} - \Delta x(1-\alpha_1), y_\beta, z_\gamma)$. The values of scalar dissipation are also evaluated at the Gauss quadrature points through WENO interpolation.

An upwinding procedure, described in Fedkiw et al. [56], is then applied to the left- and right-sided approximations at the Gauss quadrature points. If $V_e(\chi)\varphi_x^+ \geq 0$ and $V_e(\chi)\varphi_x^- \geq 0$, then $\varphi_x^{\text{up}} = \varphi_x^-$ is chosen. If $V_e(\chi)\varphi_x^+ \leq 0$ and $V_e(\chi)\varphi_x^- \leq 0$, then $\varphi_x^{\text{up}} = \varphi_x^+$. If $V_e(\chi)\varphi_x^+ \geq 0$ and $V_e(\chi)\varphi_x^- \leq 0$, then $\varphi_x^{\text{up}} = 0$. Finally, if $V_e(\chi)\varphi_x^+ \leq 0$ and $V_e(\chi)\varphi_x^- \geq 0$, then an intermediary variable is calculated,

$$s = \frac{|\varphi_x^+| - |\varphi_x^-|}{\varphi_x^+ - \varphi_x^-}, \tag{4.8}$$

and if $s > 0$, then $\varphi_x^{\text{up}} = \varphi_x^+$; otherwise, $\varphi_x^{\text{up}} = \varphi_x^-$. The upwind derivatives are then used in Eq. (4.6) to calculate the FHB propagation term's contribution to the right-hand side of Eq. (2.33).

The algebraic source term, Eq. (A.2), is computed directly at the Gauss quadrature points, according to

$$\frac{1}{V_j}\iiint_{I_j} \dot{Q} d\mathbf{x} = \sum_{\alpha=0}^{1}\sum_{\beta=0}^{1}\sum_{\gamma=0}^{1} w_\alpha w_\beta w_\gamma \dot{Q}(\varphi(x_\alpha, y_\beta, z_\gamma), \chi(x_\alpha, y_\beta, z_\gamma)), \tag{4.9}$$

where $\varphi(x_\alpha, y_\beta, z_\gamma)$ and $\chi(x_\alpha, y_\beta, z_\gamma)$ are obtained through WENO reconstruction. The flame hole formation term implemented in this manner produces flame holes with smooth boundaries; see Section 4.6.4.

4.2.3 Discretization of the dilatational-like term

There are several strategies to integrate the dilatational-like term. One approach is described by Xing and Shu [57] for the case of the shallow water equations using a WENO scheme. Other WENO-based three-dimensional nonconservative finite-volume methods include the approach of Dumbser et al. [58, 59], which use a $P_N P_M$ method in concert with a Roe-type or modified FORCE flux approximation [60]. The present approach is to use a modified Gauss quadrature rule to directly evaluate the integrals appearing in the first-term of the right-hand side of Eq. (4.2). The functions in this new modified Gauss quadrature are interpolated or reconstructed by the WENO method. The basic idea is to use a specific quadrature for integrals of the form

$$\int_{-1}^{1} \varphi(x) \frac{du}{dx}(x) dx. \tag{4.10}$$

We follow the classical approach [61] and approximate the function φ in a Hermite polynomial, according to

$$\varphi(x) \approx \tilde{\varphi}(x) = \sum_{\alpha=1}^{m} h_\alpha(x)\varphi(x_\alpha) + \sum_{\alpha=1}^{m} \bar{h}_\alpha(x)\varphi'(x_\alpha), \tag{4.11}$$

where prime denote the derivative with respect to x, and

$$h_\alpha(x) = [1 - 2l_j^{m\prime}(x_\alpha)(x - x_\alpha)][l_\alpha^m(x)]^2, \quad \bar{h}_\alpha = (x - x_\alpha)[l_\alpha^m(x)]^2, \tag{4.12}$$

where

$$l_\alpha^m(x) = \frac{(x - x_1)\cdots(x - x_{\alpha-1})(x - x_{\alpha+1})\cdots(x - x_m)}{(x_\alpha - x_1)\cdots(x_\alpha - x_{\alpha-1})(x_\alpha - x_{\alpha+1})\cdots(x_\alpha - x_m)}, \tag{4.13}$$

and x_α denote the interpolation points, yet to be determined. The function u is expanded in the same manner but with a degree $m+1$ polynomial, since we need to take the derivative of the interpolating polynomial, given by

$$\tilde{u}(x) = \sum_{b=1}^{m+1} g_\beta(x)u(x_\beta) + \sum_{b=1}^{m+1} \bar{g}_\beta(x)u'(x_\beta). \tag{4.14}$$

Here, g_β and $\bar{g}_\beta$ are analogous to h_α and $\bar{h}_\alpha$ from Eq. (4.12)-(4.13) except that their interpolating polynomials have degree $m+1$. The modified Gaussian quadrature seeks to estimate $\varphi(x)u'(x)$ by multiplying their Hermite polynomials together and integrating. The derivative of u is given by

$$\tilde{u}'(x) = \sum_{b=1}^{m+1} g'_\beta(x)u(x_\beta) + \sum_{b=1}^{m+1} \bar{g}'_\beta(x)u'(x_\beta), \tag{4.15}$$

with

$$g'_\beta(x) = [-2l_\beta^{m+1}(x_\beta)][l_\beta^{m+1}(x)]^2 + [1 - 2l_\beta^{m+1\prime}(x_\beta)(x - x_\beta)][2l_\beta^{m+1}(x)l_\beta^{m+1\prime}(x)], \tag{4.16}$$

$$\bar{g}'_\beta(x) = 2(x - x_\beta)l_\beta^{m+1}(x)l_\beta^{m+1\prime}(x) + l_\beta^{m+1}(x)^2. \tag{4.17}$$

The combined Hermite polynomial expansion for the integrand of Eq. (4.10) is

$$\tilde{u}'(x)\tilde{\varphi}(x) = \sum_{\alpha=1}^{m}\sum_{b=1}^{m+1}\Big(h_\alpha(x)g'_\beta(x)\varphi(x_\alpha)u(x_\beta) + h_\alpha(x)\bar{g}'_\beta(x)\varphi(x_\alpha)u'(x_\beta) + \bar{h}_\alpha(x)g'_\beta(x)\varphi'(x_\alpha)u(x_\beta) + \bar{h}_\alpha(x)\bar{g}'_\beta(x)\varphi'(x_\alpha)u'(x_\beta)\Big). \tag{4.18}$$

The integral of Eq. (4.18) is given by

$$\int_{-1}^{1}\tilde{u}'(x)\tilde{\varphi}(x)dx = \sum_{\alpha=1}^{m}\sum_{b=1}^{m+1}\left[I^0_{\alpha\beta}\varphi(x_\alpha)u(x_\beta) + I^1_{\alpha\beta}\varphi(x_\alpha)u'(x_\beta) + I^2_{\alpha\beta}\varphi'(x_\alpha)u(x_\beta) + I^3_{\alpha\beta}\varphi'(x_\alpha)u'(x_\beta)\right], \tag{4.19}$$

with

$$\begin{aligned} I^0_{\alpha\beta} &= \int_{-1}^{1} h_\alpha(x)g'_\beta(x)dx, \qquad & I^1_{\alpha\beta} &= \int_{-1}^{1} h_\alpha(x)\bar{g}'_\beta(x)dx, \\ I^2_{\alpha\beta} &= \int_{-1}^{1} \bar{h}_\alpha(x)g'_\beta(x)dx, \qquad & I^3_{\alpha\beta} &= \int_{-1}^{1} \bar{h}_\alpha(x)\bar{g}'_\beta(x)dx. \end{aligned} \tag{4.20}$$

In order for the quadrature to not involve the derivatives of either u or φ (which are unknown), $I^1_{\alpha\beta}$, $I^2_{\alpha\beta}$, and $I^3_{\alpha\beta}$ must be zero for all a and b, a task accomplished by choosing the abscissae x_α for φ and x_β for u such that the integrals vanish. The weights and abscissae can be derived by solving a nonlinear system of equations for both the values of the weights and locations of the abscissae, designed by requiring exact solutions to polynomials of a certain degree. For polynomials of combined degrees $n_\alpha + n_\beta < 2m+1$, we have

$$\sum_{\alpha}^{n_\alpha}\sum_{\beta}^{n_\beta} w_{\alpha\beta}x^j_\alpha x^k_\beta = \int_{-1}^{1}\xi^{n_\alpha}\frac{d(\xi^{n_\beta})}{d\xi}d\xi = \frac{(1+(-1)^{n_\alpha+n_\beta})n_\beta}{n_\alpha+n_\beta}. \tag{4.21}$$

The solution of this polynomial system of equations provides both the weights and abscissae of the modified Gaussian quadrature. We have implemented this method using both $m=2$ and $m=3$. The weights and abscissae are given in B.2.

The modified Gauss quadrature is extended to three dimensions using the regular Gaussian quadrature rule in the directions orthogonal to the derivative direction (tensor product). The values of the functions at the modified Gaussian quadrature points are estimated by WENO interpolation, e.g., evaluation for the x-derivative term gives

$$\frac{1}{\mathrm{V}_j}\iiint_{\mathrm{I}_j}\frac{\partial u}{\partial x}\varphi d\mathbf{x} = \frac{1}{\Delta x}\sum_{\alpha}\sum_{\beta}\sum_{\kappa}\sum_{\lambda} w_\kappa w_\lambda w_{\alpha\beta}\varphi(x_\alpha, y_\kappa, z_\lambda)u(x_\beta, y_\kappa, z_\lambda), \tag{4.22}$$

with analogous terms in the y and z directions. The integration involves a significant amount of WENO interpolations and reconstructions to perform, with 48 evaluations required to evaluate the $m = 3$ quadrature in one of three directions for each finite-volume cell. The method however directly evaluates the dilatational-like term using a WENO scheme consistent with the divergence-like term and since it is only applied to one scalar equation, its overall computational costs is not overwhelming; note that in the use of the whole coupled method one is also solving the Navier-Stokes equations and the extension operator described in next section.

We can estimate the numerical truncation errors of the divergence and dilatational terms generated by both the interpolation (reconstruction) error of the WENO method and the integration error of the Gaussian quadratures. Consider the surface integration of the divergence term, with a Gaussian integration of order p and a WENO interpolation of order q. For equal grid spacing $\Delta x = \Delta y = \Delta z = \Delta$, we have

$$\begin{aligned}\frac{1}{\Delta^3}\iint f(y,z)dydz &= \frac{1}{\Delta^3}\left(\iint \tilde{f}(y,z)dydz + \iint O(\Delta^q)dydz\right)\\ &= \sum_{\alpha}\sum_{\beta} w_\alpha w_\beta \tilde{f}(y_\alpha, z_\beta) + O(\Delta^{p-3}) + O(\Delta^{q-1}),\end{aligned} \tag{4.23}$$

where $\tilde{f}$ denotes the interpolated value of f. The error in one of the flux terms is order $\min(p-3, q-1)$. Since these fluxes appear always in pairs, as a difference between neighboring points, one actually gains one order of accuracy above the individual flux approximations, giving a final error of $\min(p-2, q)$. A similar analysis applies to the nonconservative term, giving

$$\begin{aligned}\frac{1}{\Delta^3}\iiint \nabla\cdot\mathbf{u}(x,y,z)\varphi(x,y,z)dxdydz =\\ \frac{1}{\Delta^3}\iiint [\nabla\cdot\tilde{\mathbf{u}}(x,y,z) + O(\Delta^{q-1})][\tilde{\varphi}(x,y,z) + O(\Delta^q)]dxdydz =\\ \frac{1}{\Delta^3}\iiint \nabla\cdot\tilde{\mathbf{u}}(x,y,z)\tilde{\varphi}(x,y,z)dxdydz + O(\Delta^{q-1}) =\\ \sum\sum\sum w_{\alpha,\beta,\gamma}\nabla\cdot\tilde{\mathbf{u}}(x_{\alpha'}, y_{\beta'}, z_{\gamma'})\tilde{\varphi}(x_\alpha, y_\beta, z_\gamma) + O(\Delta^{r-3}) + O(\Delta^{q-1}).\end{aligned} \tag{4.24}$$

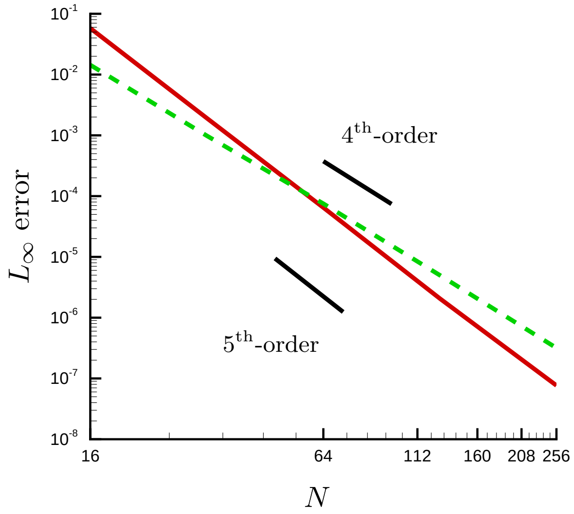


Figure 4.1: Scaling of the truncation error for the divergence- and dilatational-like terms for two-point quadratures.

were r denotes the order of the modified Gaussian quadrature introduced above, $w_{\alpha,\beta,\gamma}$ denotes the corresponding combined quadrature weights in all directions and the primed Greek index denote the collocation points of the derivative term in the modified Gaussian quadrature. The order of accuracy for this term is therefore $\min(r-3, q-1)$.

Figure 4.1 shows the scaling of the calculated truncation error for the whole divergence and dilatational terms using the quadrature rules described above for $u = 1 + a\sin(x)$, $v = 1 + a\sin(y)$, $w = 1 + a\sin(z)$, $\varphi = \sin(x)\sin(y)\sin(z)$ with $a = 0.8$, $x, y, z \in [0, 2\pi]$ using grids with 16^3, 32^3, 64^3, 128^3, and 256^3 points. For the two-point Gaussian quadrature rules we use throughout the paper, it was found that the dilatational term follows an approximately 4^{th}-order slope with increasing resolution, and the divergence term has a 5^{th}-order error, as expected by the previous truncation error estimates since $p = 2 \times 3 + 1 = 7$, $r = 2 \times 3 + 1 = 7$, and $q = 5$.

4.3 Discretization of the elliptic-type extension operator

First, a band of grid points that are within $\Delta_{\max} = \max(\Delta x, \Delta y, \Delta z)$ from the stoichiometric surface are identified as the boundary points φ_{bc} (Dirichlet boundary condition data) of Eq. (3.18).

Second, discretization of Eq. (3.18) is done through a first-order upwind method for the advection term, where the x derivative (similarly for the y and z derivatives) is given by

$$D_{\mathrm{up},x}\varphi = m_x \frac{\varphi_{i,j,k} - \varphi_{i-m_x,j,k}}{\Delta x}, \tag{4.25}$$

where $m_x = \mathrm{sgn}(a_x)$, and

$$\mathbf{a} = (a_x, a_y, a_z) = \mathrm{sgn}(Z - Z_{\mathrm{st}})\tilde{\mathbf{n}}, \tag{4.26}$$

with $\tilde{\mathbf{n}}$ given by Eq. (3.19) approximated by central differences. The x derivatives of the diffusion terms are discretized according to

$$D_x(\varepsilon D_x \varphi) = \frac{1}{2\Delta x^2}\Big[(\varepsilon_{i+1,j,k} + \varepsilon_{i,j,k})(\varphi_{i+1,j,k} - \varphi_{i,j,k}) - (\varepsilon_{i,j,k} + \varepsilon_{i-1,j,k})(\varphi_{i,j,k} - \varphi_{i-1,j,k})\Big], \tag{4.27}$$

which in the case of constant coefficient diffusion reduces to the standard central stencil for the second-order derivative; similarly for the other coordinate directions. After rearrangement of the variables, the extension algorithm requires the solution of

$$\mathcal{L}\varphi = b_{\mathrm{bc}}, \tag{4.28}$$

where

$$\mathcal{L} = a_x D'_{\mathrm{up},x} + a_y D'_{\mathrm{up},y} + a_z D'_{\mathrm{up},z} - D'_x \varepsilon D'_x - D'_y \varepsilon D'_y - D'_z \varepsilon D'_z, \tag{4.29}$$

denotes the advection-diffusion operator stripped of the points affecting φ_{bc} (denoted with primes) and b_{bc} denotes the parts of the stencils involving φ_{bc}.

Figure 4.2 shows the result of solving Eq. (4.28) when discretized in the Cartesian grid for the case of a circular flame disk on a curved sinusoidal stoichiometric surface. The flame state field φ is extended smoothly into the domain, perpendicular to the iso-surfaces of the mixture fraction and along the normal $\mathbf{n}$, as depicted with translucent iso-surfaces of constant φ. Note that the extended φ iso-surfaces ($\varphi = 0.25$, 0.5 and 0.75) concentrate as they move away from Σ_{st} due to the curvature of this surface and the interaction with the flame state values on the other sections of this surface.

The resulting equations are solved using an algebraic multigrid method (BoomerAMG) provided

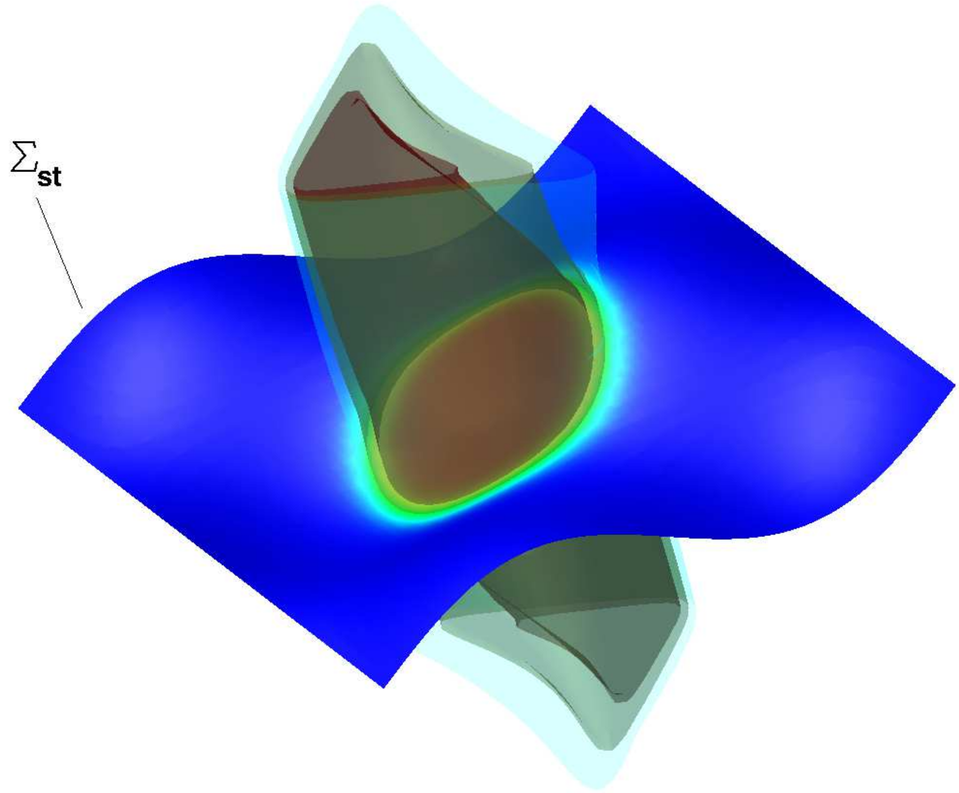


Figure 4.2: Example of extension of φ defined on a sinusoidal surface using the Cartesian grid-based method, with $Z_{\mathrm{st}} = 0.5$. Surface is given by $Z_{\mathrm{st}} = asin(2\pi/L_x x)cos(2\pi/L_y y)$, with $a = 0.1L_x$. Color scale used is consistent with scale in figure 3.1.

by the library *hypre* [62]. Finally, note that the extension only depends on the value of φ at points neighboring the stoichiometric surface, over a narrow band of grid points, and it is performed at every Runge-Kutta substage. Therefore, the advection, source, and FHB propagation terms need to be only evaluated at these boundary points, which allows high-order methods to be applied without significantly increasing the computational cost of the method since most points in the domain are skipped.

4.4 Discretization of the anti-dissipative correction

The antidiffusive correction is discretized according to Shukla et al. [48],

$$\frac{\partial\varphi}{\partial\tau} = -\left[n^x_{ijk}\left(\frac{f_{i+1,j,k} - f_{i-1,j,k}}{\Delta x}\right) + n^y_{ijk}\left(\frac{g_{i,j+1,k} - g_{i,j-1,k}}{\Delta y}\right) + n^z_{ijk}\left(\frac{h_{i,j,k+1} - h_{i,j,k-1}}{\Delta z}\right)\right], \quad (4.30)$$

where f is the x-directed flux,

$$f_{i+1/2,j,k} = (\varphi(1-\varphi))_{i+1/2,j,k} - |\nabla\varphi|_{i+1/2,j,k}, \quad (4.31)$$

and g and h are the y- and z-directed fluxes respectively. The gradient term, $\nabla\varphi$, is computed through Eq. (3.37) using the discretization

$$\nabla\psi = (\frac{\psi_{i+1/2,j,k} - \psi_{i-1/2,j,k}}{\Delta x}, \frac{\psi_{i,j+1/2,k} - \psi_{i,j-1/2,k}}{\Delta y}, \frac{\psi_{i,j,k+1/2} - \psi_{i,j,k-1/2}}{\Delta z}). \quad (4.32)$$

The pseudo-time integration uses an explicit forward Euler discretization as proceeds as

$$\frac{\partial\varphi}{\partial\tau} \approx \frac{\varphi^n - \varphi^{n-1}}{\Delta\tau}, \quad (4.33)$$

with $\Delta\tau$ being specified from the subiteration CFL number,

$$\mathrm{CFL}_{\mathrm{comp}} = \frac{\mathcal{U}_0\Delta\tau}{4\epsilon_c}. \quad (4.34)$$

Figures 3.1 shows the effect the anti-dissipative correction on a flame disc for a flat stoichiometric surface. Table 4.2 shows the effect that the anti-dissipative correction has on the error of the solution for a flat stoichiometric surface test case at several different resolutions.

4.5 Parallelization strategy

The parallelization strategy for the structured implementation uses a block decomposition to assign a subset of the computational domain to each processor. An aura of ghosts is added to each domain subset and field data is communicated by the SENDRECV routine of OpenMPI to populate data in the ghost cells. For instance, for a (N, N, N) domain, a decomposition onto N_p processors results with each processor dealing with a subset of size $(N/N_p + 2N_G)^3$, with N_G being the number of ghosts used in the domain. The number of ghosts is set to be 3 in our cases, as the WENO involves points between $i - 3, \ldots, i + 3$ in each dimension. The addition of the closest point transform, however, requires additional ghosts be added to the domain. As the closest point extension injects points based on the values of φ at in a cell around $\mathbf{cp}(x_j)$, which can be a maximum distance of $5 \max(\Delta x, \Delta y, \Delta z)$ away from the grid point being processed. As the maximum distance of the closest point transform is fixed, the number of the ghost cells can be set to be one greater than the width of the band of the closest point transform.

The closest point transform is a relatively expensive $\mathcal{O}(N^2)$ operation. As such, with increased resolution, the percentage of time used by the closest point transform reduces, *ceteris paribus*. In practice, load balancing in parallel can cause the closest point transform to be a more expensive part of the computation than would otherwise be expected. Load balancing typically is done on a volumetric basis, with the domain split into parts of an equal number of total grid points, with a cubic decomposition limiting the surface area of the domain partitions (and, thus, the size of the communication calls). For the closest point transform, however, the most effective decomposition would be to decompose the stoichiometric surface into equally sized partitions. For the shear layer simulation, used as an example here, the domain was divided in periodic directions x and z but not in the inhomogeneous dimension y. This is not an optimal decomposition in terms of the number of ghost cells, but it does approximately divide the stoichiometric surface into equal parts as well as the volume.

The library *hypre*[62] is used to solve the elliptic-type extension efficiently in parallel.

Resolution	L^1	L^2	L^∞	Order
$16 \times 16 \times 2$	2.67×10^{-4}	1.22×10^{-3}	7.17×10^{-3}	
$32 \times 32 \times 2$	4.87×10^{-6}	3.18×10^{-5}	2.96×10^{-4}	5.26
$64 \times 64 \times 2$	1.05×10^{-7}	9.42×10^{-7}	1.22×10^{-5}	5.08
$128 \times 128 \times 2$	3.45×10^{-9}	4.33×10^{-8}	7.42×10^{-7}	4.44

Table 4.1: Advection of a smooth, sinusoidal φ in the x-direction on a flat surface. Error comparison is between the initial condition and the solution after one flow through time. Order of accuracy is estimated based on the 2-norm.

4.6 Tests of the complete algorithm

The performance of the algorithms was studied using several test cases of increasing degree of complexity. These tests were conducted in addition to several elementary tests that verified that the individual discretization components functioned according to design. For example, advection, extension, compression, etc, were all tested individually. Here, we detail the coupled behavior of the complete method for increasing degree of complexity of Σ_{st}.

4.6.1 Flat Σ_{st}

The convergence rate of the method is first evaluated using a very smooth initial condition, given by $\varphi = \sin(2\pi x/L_x)$ (where L_x is the length of the domain in the x-direction) advected at $u = 1$ over a flat Σ_{st} without compression, see Section 3.2. Table 4.1 shows better than fourth-order convergence rate at lower resolution and convergence to fourth order at the highest resolution in all the norms, as expected for a smooth field. This test verifies the convergence rate of the WENO method.

The second test corresponds to propagation of a circular flame hole in a flat stationary surface. Here, extension is not required because the solution is everywhere parallel to the surface plane (taken as the horizontal plane). The hole is advected at uniform velocity $u = 1$ and in consequence the dilatational term of advection is identically zero. This allows us to verify the accuracy of the conservative WENO algorithm when the flame hole formation and propagation are also disabled. The compression algorithm used $\epsilon_c = L_x/64$ and applied only two subiterations with pseudo-time CFL of 0.0025. Two subiterations of the compression algorithm are quite effective at preventing smearing of φ due to numerical dissipation over long time integrations. Figure 4.3 shows results

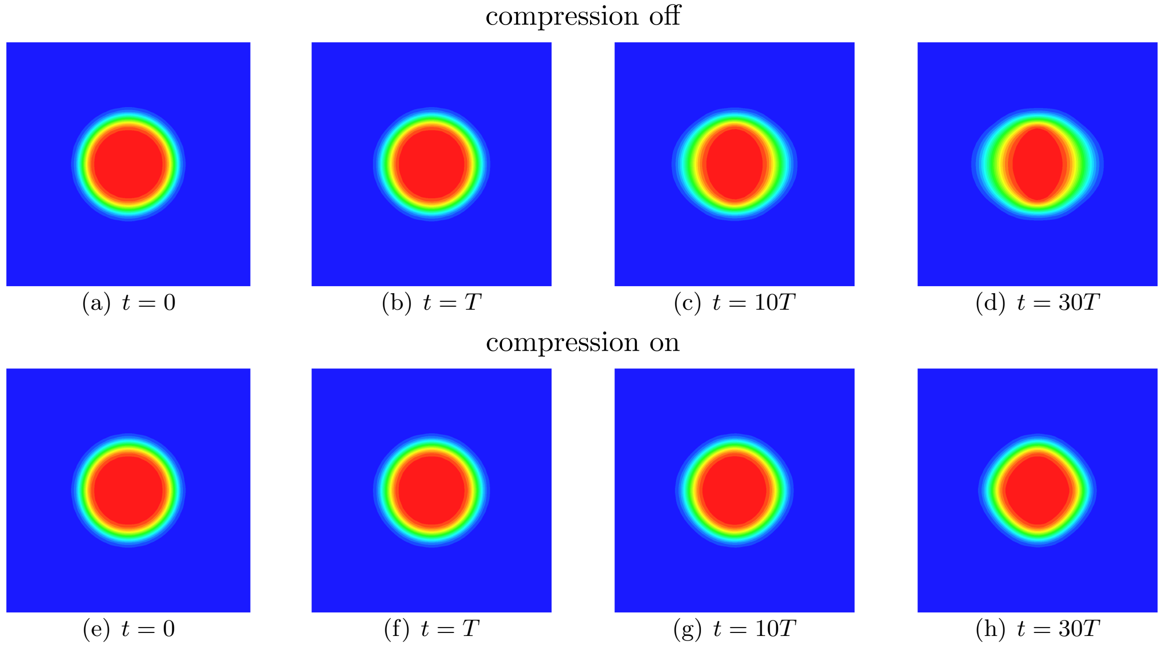


Figure 4.3: Flame state φ after one, ten, and thirty flow-through times T for a flat Σ_{st}, computed on a $64 \times 48 \times 2$ grid in a domain of size $L_x = 1.0$, $L_y = 0.75$, $L_z = 0.03125$. Color scale used is consistent with scale in figure 3.1.

from this test case, comparing the solution after 1, 10, and 30 flow-through times $T = L_x/u$, without and with compression. As can be seen, the compression algorithm limits the effect of diffusion, keeping the interface thin even after a very long time. Since the whole algorithm is a Cartesian-based discretization method, there is some deformation of the shape of the interface over long periods of time which has its roots in the advection algorithm itself, due to grid anisotropy errors, but observe that the thickness of φ is unchanged even at late times, as desired.

Table 4.2 shows the convergence rate of the method. Here, since a very sharp initial condition is used, the WENO method on the coarser meshes exhibits a lower convergence rate of third-order. As the mesh is refined, the method has a higher, third- to fourth-order convergence. As seen in Table 4.2(b), compression does not significantly affect the convergence when a sufficiently high resolution is used.

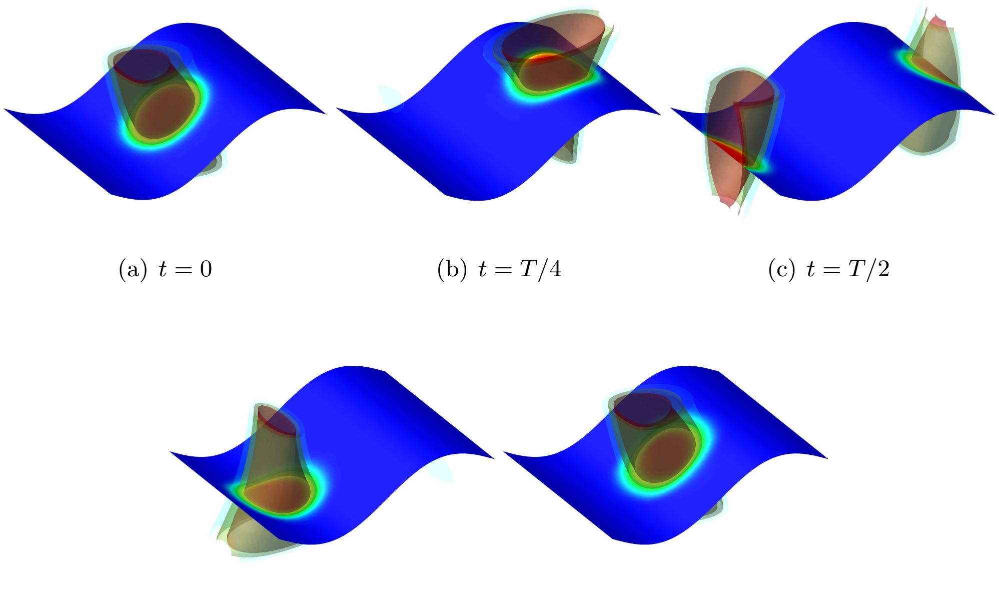
Resolution	L^1	L^2	L^∞	Order
$32 \times 32 \times 2$	3.49×10^{-4}	5.27×10^{-3}	1.33×10^{-1}	
$48 \times 48 \times 2$	8.15×10^{-5}	1.93×10^{-3}	7.69×10^{-2}	2.48
$64 \times 64 \times 2$	2.82×10^{-5}	7.75×10^{-4}	4.24×10^{-2}	3.15
$96 \times 96 \times 2$	5.03×10^{-6}	2.15×10^{-4}	1.75×10^{-2}	3.16
$128 \times 128 \times 2$	1.77×10^{-6}	9.25×10^{-5}	9.52×10^{-3}	2.93
$192 \times 192 \times 2$	3.22×10^{-7}	2.25×10^{-5}	3.31×10^{-3}	3.49

(a)

Resolution	L^1	L^2	L^∞	Order
$32 \times 32 \times 2$	3.26×10^{-4}	4.98×10^{-3}	1.25×10^{-1}	
$48 \times 48 \times 2$	7.99×10^{-5}	1.69×10^{-3}	6.79×10^{-2}	2.66
$64 \times 64 \times 2$	2.30×10^{-5}	6.23×10^{-4}	3.41×10^{-2}	3.48
$96 \times 96 \times 2$	3.55×10^{-6}	1.26×10^{-4}	9.30×10^{-3}	3.93
$128 \times 128 \times 2$	1.18×10^{-6}	4.71×10^{-5}	3.80×10^{-3}	3.43
$192 \times 192 \times 2$	3.16×10^{-7}	1.47×10^{-5}	1.36×10^{-3}	2.89

(b)

Table 4.2: Advection of a flame disc in the x-direction on a flat surface without (a) and with (b) compression. Error comparison is between the initial condition and the solution after one flow through time. Order of accuracy is estimated based on the 2-norm.



(a) $t = 0$

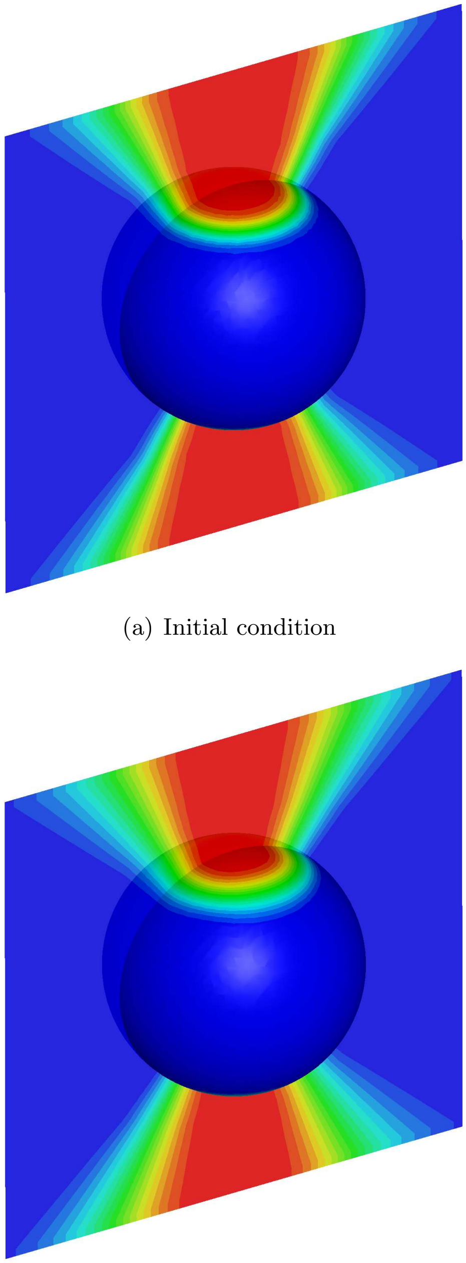
(b) $t = T/4$

(c) $t = T/2$

(d) $t = 3/4T$

(e) $t = T$

Figure 4.4: Solution after one flow-through time, T, for the advection of a flame disc on a sinusoidal surface with a mesh resolution of $64 \times 48 \times 32$. Color scale used is consistent with scale in figure 3.1.



(a) Initial condition

(b) After one rotation

Figure 4.5: Solution after one flow-through time of a flame disc rotated on a moving sphere by an irrotational vortex with a mesh resolution of $64 \times 64 \times 64$. Color scale used is consistent with scale in figure 3.1.

Resolution	L^1	L^2	L^∞	Order
$32 \times 24 \times 16$	9.38×10^{-4}	9.92×10^{-3}	2.47×10^{-1}	
$48 \times 36 \times 24$	3.48×10^{-4}	5.17×10^{-3}	1.62×10^{-1}	1.61
$64 \times 48 \times 32$	1.56×10^{-4}	2.87×10^{-3}	1.09×10^{-1}	2.05
$96 \times 72 \times 48$	4.42×10^{-5}	1.03×10^{-3}	5.17×10^{-2}	2.53
$128 \times 96 \times 64$	1.84×10^{-5}	4.78×10^{-4}	2.98×10^{-2}	2.66
$192 \times 144 \times 96$	4.03×10^{-6}	1.26×10^{-4}	1.08×10^{-2}	3.29

Table 4.3: Advection of a flame disc in the x-direction on a sinusoidal surface. Error comparison is between the initial condition and the solution after one flow through time. Order of accuracy is estimated based on the 2-norm.

4.6.2 Sinusoidal Σ_{st}

The third test considers advection of a flame hole in a periodic domain over a Σ_{st} of sinusoidal shape along the direction of a flow with velocity components

$$\begin{aligned} u &= \frac{1}{\sqrt{1+(ak_x\cos(k_x x))^2}} \\ v &= 0 \\ w &= \frac{-ak_x\cos(k_x x)}{\sqrt{1+(ak_x\cos(k_x x))^2}}, \end{aligned} \tag{4.35}$$

with $a = 0.1$ and $k_x = 2\pi/L_x$ referring to the amplitude and wavenumber of the sinusoidal-wave surface. The flow-through time is calculated as the arc-length along the sine wave and is $T = 1.09L_x/u_0$, $u_0 = 1$, for the chosen parameters. The initial value of φ is given by the intersection of a cylinder of radius $0.1875L_x$ with Σ_{st}, creating a "flame disc" on the surface. Advection, compression and extension subalgorithms are enabled in this calculation and the dilatational-like term discretization is also active since the velocity field is not divergence free, making this is a more realistic test. After one T, the shape of the hole should return unmodified to the initial condition because self-propagation of the flame hole boundary and hole formation are disabled. Figure 4.4 shows the evolution of the extended φ field at several instants in $0 \leq t \leq T$. The initial condition and the solution after one flow-through time are compared quantitatively in terms of the one-, two-, and infinity norms of the difference between the initial condition and the solution after one flow-through time around Σ_{st}. These error norms are tabulated in Table 4.3.

Resolution	L^1	L^2	L^∞	Order
$32 \times 32 \times 32$	1.40×10^{-3}	1.28×10^{-2}	2.39×10^{-1}	
$48 \times 48 \times 48$	5.42×10^{-4}	6.43×10^{-3}	1.44×10^{-1}	1.69
$64 \times 64 \times 64$	2.33×10^{-4}	3.19×10^{-3}	9.11×10^{-2}	2.44
$96 \times 96 \times 96$	7.04×10^{-5}	1.19×10^{-3}	4.52×10^{-2}	2.43
$128 \times 128 \times 128$	2.63×10^{-5}	5.10×10^{-4}	2.08×10^{-2}	2.95
$192 \times 192 \times 192$	6.15×10^{-6}	1.36×10^{-4}	6.60×10^{-3}	3.26

Table 4.4: Advection of a flame disc by an irrotational vortex on a spherical surface. Error comparison is between the initial condition and the solution after one rotation. Order of accuracy is estimated based on the 2-norm.

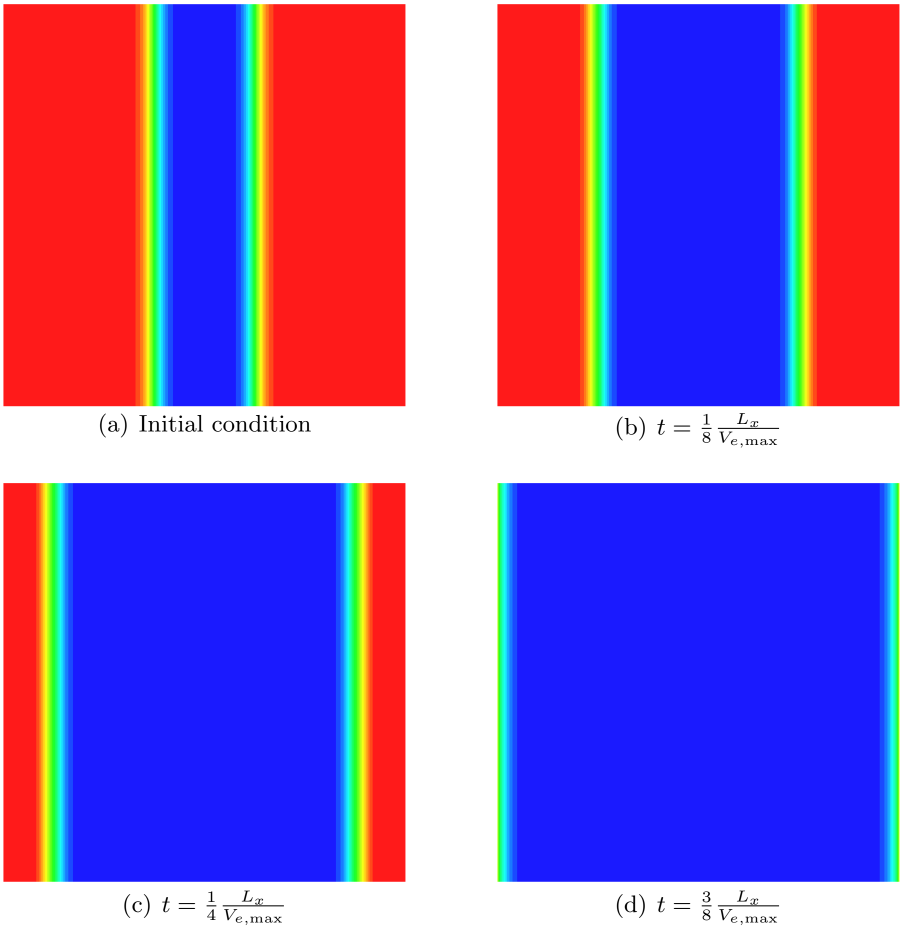
Resolution	L^1	L^2	L^∞	Order
$32 \times 32 \times 32$	1.80×10^{-3}	1.53×10^{-2}	2.49×10^{-1}	—
$48 \times 48 \times 48$	6.39×10^{-4}	7.47×10^{-3}	1.65×10^{-1}	1.77
$64 \times 64 \times 64$	2.81×10^{-4}	3.83×10^{-3}	1.08×10^{-1}	2.32
$96 \times 96 \times 96$	9.11×10^{-5}	1.52×10^{-3}	5.64×10^{-2}	2.27
$128 \times 128 \times 128$	3.77×10^{-5}	7.49×10^{-3}	3.52×10^{-2}	2.47
$192 \times 192 \times 192$	8.99×10^{-6}	2.14×10^{-4}	1.40×10^{-2}	3.09

Table 4.5: Advection of a flame disc by an irrotational vortex on a moving spherical surface. Error comparison is between the initial condition and the solution after one rotation. Order of accuracy is estimated based on the 2-norm.

4.6.3 Spherical Σ_{st}

The rotation of flame discs on a spherical Σ_{st} is tested to ensure the accuracy on a closed surface with variable flow velocity. Circular flame disc regions ($\varphi = 1$) are initialized on Σ_{st} and are then extended to create the initial condition for this test case. φ is advected by an irrotational vortex with velocity given by $\mathbf{u}^* = (u, v, w) = (0, -z, y)$, which rotates the flame disc on the surface. The complete algorithm is used to evolve φ but, as before, without self-propagation of flame hole boundary or flame hole formation. After one rotation, the solution is compared with the initial condition, which should be ideally identical, as shown in Figure 4.5. The accuracy results are given in Table 4.4.

To test the algorithm on a moving Σ_{st}, the previous test is modified such that the sphere moves laterally in a sinusoidal motion in time. The center of the sphere, initially at the origin of the domain, moves in the x-direction as $x_c(t) = a\sin(2t)$, making the surface velocity $\mathbf{V}_{\text{st}} = (2a\cos(2t), 0, 0)$. The velocity of the flow is then determined according to $\mathbf{u} = \mathbf{u}^* + \mathbf{V}_{\text{st}}$, where $\mathbf{u}^*$ is the velocity in the flow in the stationary sphere case used above. The amplitude of the


(a) Initial condition

(b) $t = \frac{1}{8}\frac{L_x}{V_{e,\max}}$

(c) $t = \frac{1}{4}\frac{L_x}{V_{e,\max}}$

(d) $t = \frac{3}{8}\frac{L_x}{V_{e,\max}}$

Figure 4.6: Evolution of a flame hole strip expanding at maximum negative edge speed, $-V_{e,\max}$, at different instants of time. Color scale used is consistent with scale in figure 3.1.

displacement used in the test is $a = 0.125L_x$. The movement of Σ_{st} makes it such that the closest point transform and extension matrix must be recreated at every subiteration of the Runge-Kutta, including the determination of the boundary points of the surface. The accuracy results are given in Table 4.5.

4.6.4 Flame hole propagation and creation tests

To test the FHB propagation properties, a flame hole strip was initialized in the domain and allowed to move without a background flow velocity. As described in A.1, the scalar dissipation was set to $\chi = 2\chi_{\text{q}}$, such that the maximum negative edge flame speed, $-V_{e,\max}$, is reached. The strip is initialized to have a half-width of $\frac{1}{8}L_x$ on a $128 \times 128 \times 2$ domain. Under these circumstances,

Resolution	L^1	L^2	L^∞	Order
$32 \times 32 \times 2$	8.22×10^{-4}	8.87×10^{-3}	1.66×10^{-1}	
$48 \times 48 \times 2$	3.16×10^{-4}	4.33×10^{-3}	1.10×10^{-1}	1.77
$64 \times 64 \times 2$	1.38×10^{-4}	2.38×10^{-3}	7.85×10^{-2}	2.08
$96 \times 96 \times 2$	2.98×10^{-5}	7.72×10^{-4}	4.18×10^{-2}	2.78
$128 \times 128 \times 2$	1.06×10^{-5}	3.64×10^{-4}	2.32×10^{-2}	2.61
$192 \times 192 \times 2$	2.40×10^{-6}	1.10×10^{-4}	1.06×10^{-2}	2.95

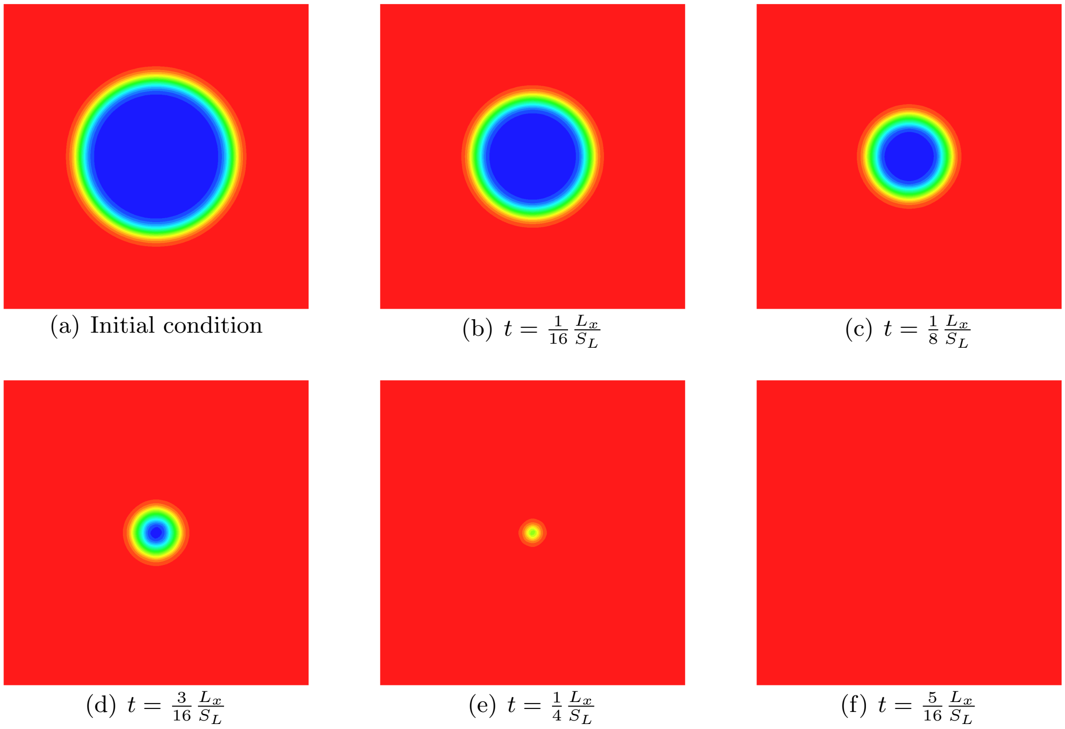
Table 4.6: Test of self-advection for the closing flame hole test case, compared with the exact solution at $t = 0.125\frac{L_x}{S_L}$. Order of accuracy is estimated based on the 2-norm.

we expect the $\varphi = 0.5$-level to reach the edge of the computational domain at $\frac{3}{8}\frac{L_x}{V_{e,max}}$, which corresponds to the observed behavior in figure 4.6. The opposite case, where the flame hole heals under a low scalar dissipation, was also tested. A flame hole is initialized in a domain of zero scalar dissipation with zero flow velocity and allowed to heal. Now, according to the FHB propagation model Eq. (A.1), the edge flame travels at a speed of S_L. The hole is initialized to be $0.25L_x$ in radius and at a constant speed is expected to heal over a time $0.25L_x/S_L$ on a $128 \times 128 \times 2$ domain. The flame is seen to be closing at the appropriate time, with the $\varphi = 0.5$ level reaching the center of the domain at the expected time, the hole closing and the shape being nearly circular throughout the evolution, see figure 4.7. The solution is compared with the exact solution at $t = 0.125L_x/S_L$, where the hole has reduced to a radius of $0.125L_x$. Accuracy results are tabulated in Table 4.6.

Finally, a test of flame hole formation was considered where a cylindrical region of high scalar dissipation, greater than χ_q, was set in a circular region of $0.25L_x$ radius at the center of a $64 \times 64 \times 2$ domain with φ set initially to one. Figure 4.8 shows the formation of a cylindrical hole in approximately τ_q corresponding to the high scalar dissipation region and the permanence of this hole in time.

4.7 Application to realistic turbulent flow

In more realistic applications of flame hole dynamics, the stoichiometric surface is highly distorted by the stirring and mixing effects of the flow. To assess the applicability of the flame hole dynamics algorithms to more realistic conditions, we tested the performance of the method on the stoichiometric surface obtained from a DNS of a reacting turbulent shear layer at a fixed instant of time,


(a) Initial condition

(b) $t = \frac{1}{16} \frac{L_x}{S_L}$

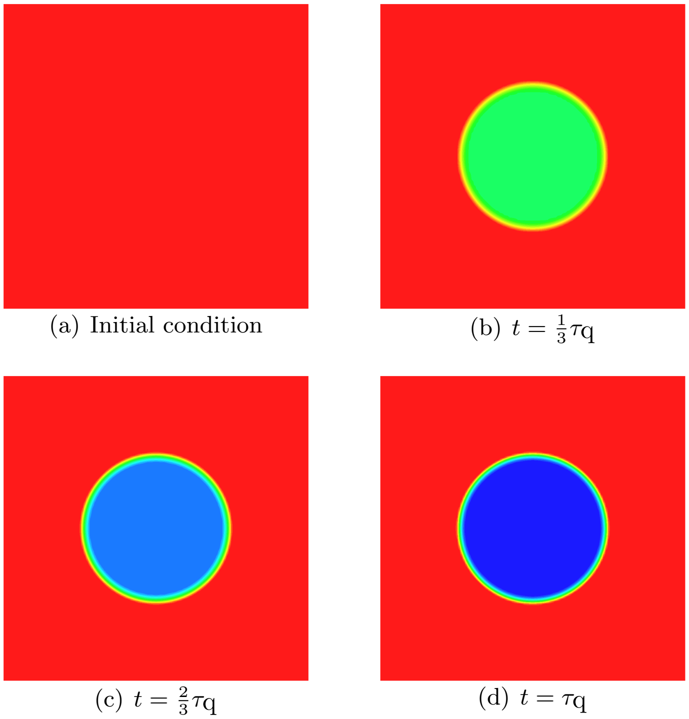
(c) $t = \frac{1}{8} \frac{L_x}{S_L}$

(d) $t = \frac{3}{16} \frac{L_x}{S_L}$

(e) $t = \frac{1}{4} \frac{L_x}{S_L}$

(f) $t = \frac{5}{16} \frac{L_x}{S_L}$

Figure 4.7: A flame hole closing in a domain with zero scalar dissipation. The hole is expected to close at $0.25 L_x / S_L$. Color scale used is consistent with scale in figure 3.1.



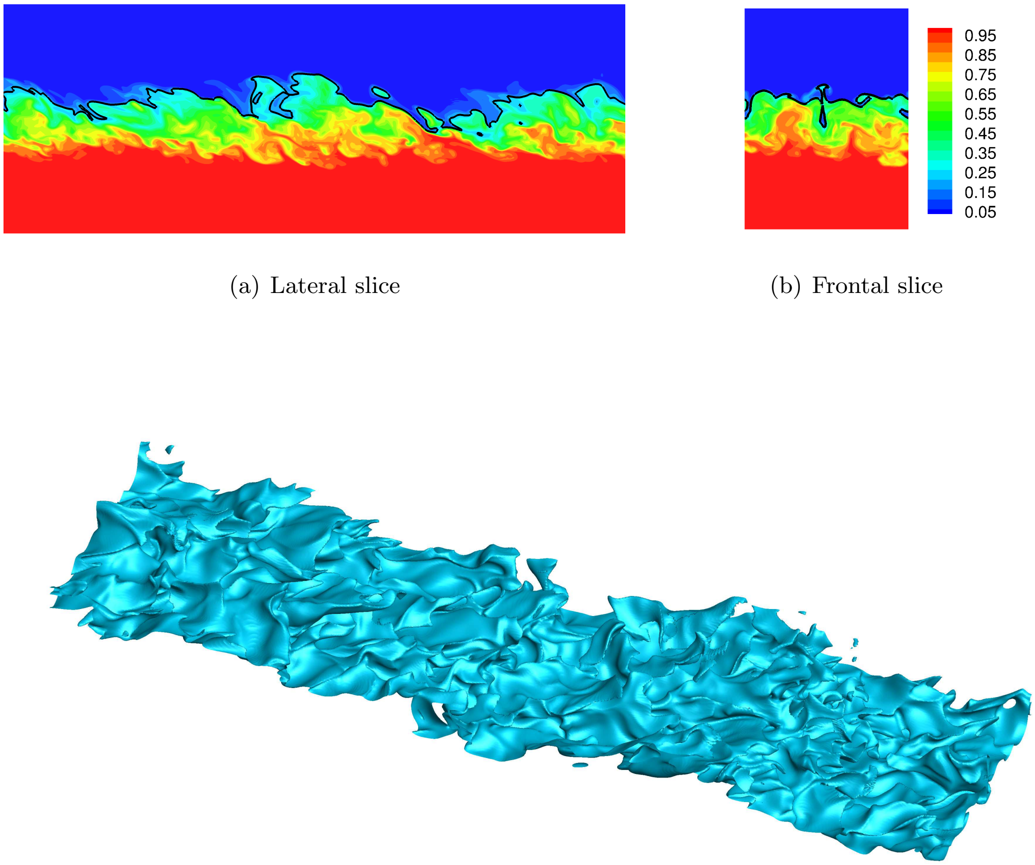
(a) Initial condition

(b) $t = \frac{1}{3}\tau_\mathrm{q}$

(c) $t = \frac{2}{3}\tau_\mathrm{q}$

(d) $t = \tau_\mathrm{q}$

Figure 4.8: A hole forming in the presence of a cylindrical region of high scalar dissipation beyond the quenching limit, χ_q. Color scale used is consistent with scale in figure 3.1.

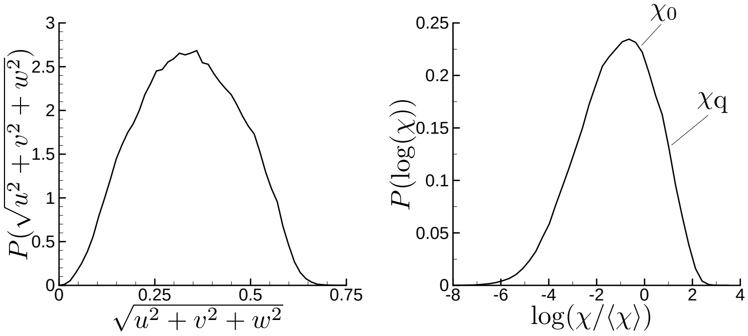


(a) Lateral slice

(b) Frontal slice

(c) Stoichiometric surface

Figure 4.9: Views of mixture fraction in a turbulent reacting shear layer obtained from DNS.



(a) Kinetic energy

(b) Scalar dissipation

Figure 4.10: Statistics of the shear layers on the stoichiometric surface.

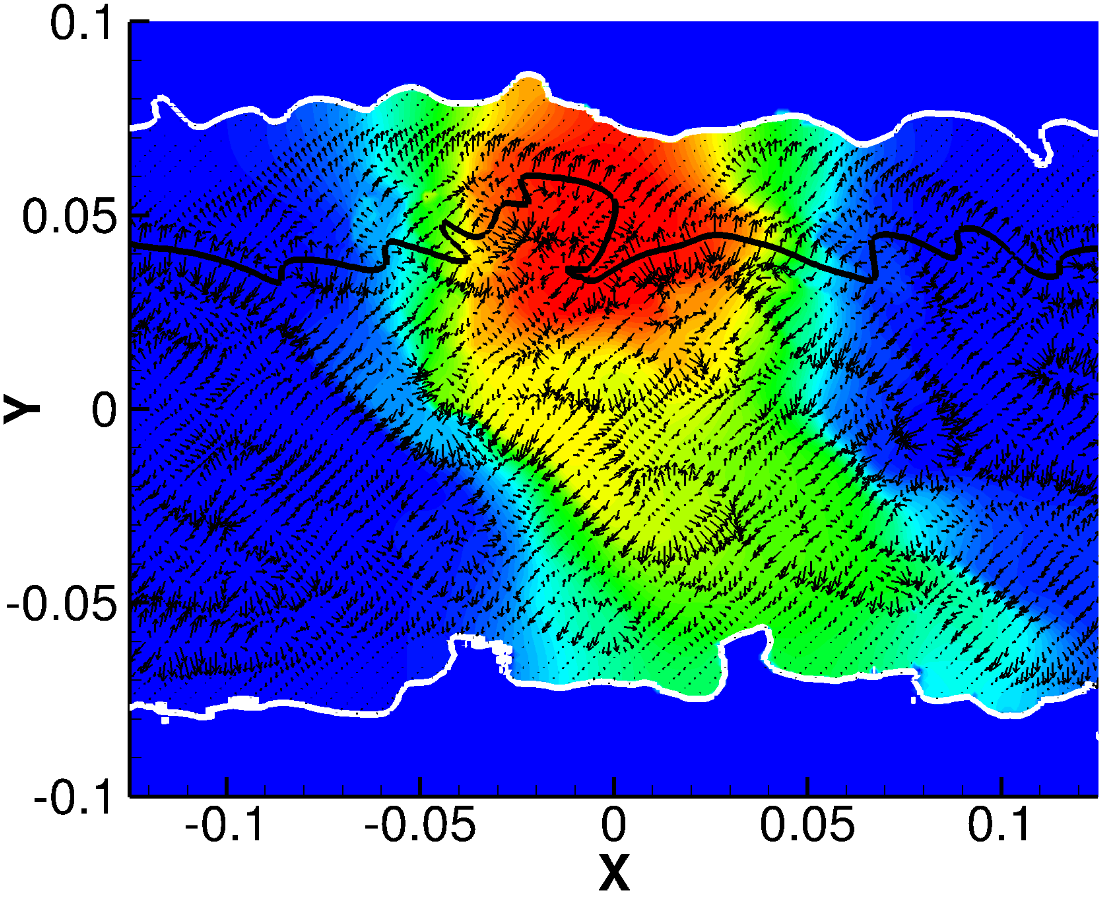
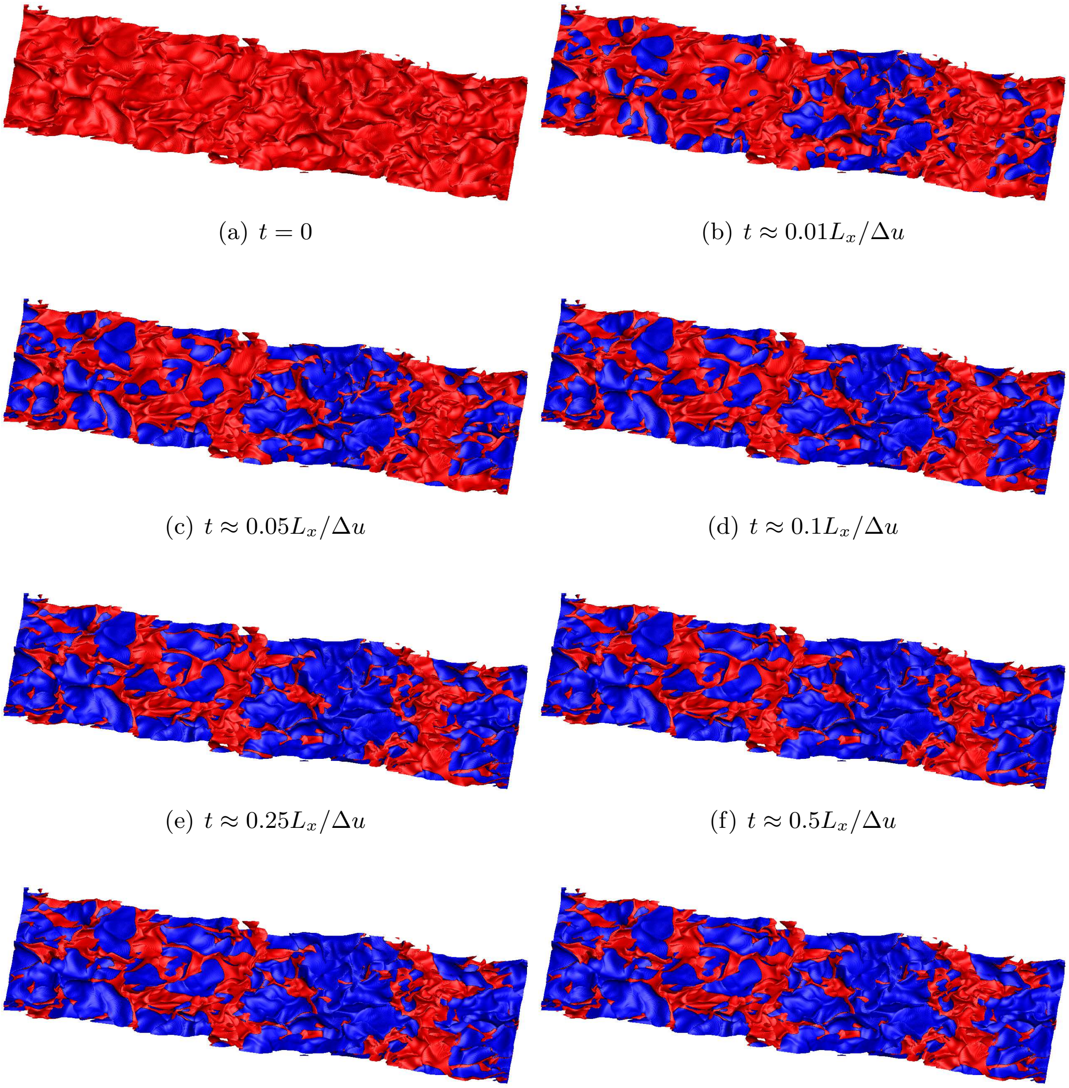


Figure 4.11: A slice of the extended φ applied to a turbulent shear layer DNS [3], with the stoichiometric surface shown as a thick continuous black curve and the direction field $\mathbf{a}$ throughout the domain. The exterior boundary is shown through a thick, continuous white curve. The vector field represents the advection velocity $\mathbf{a}$ of the mixture fraction field. Color scale used is consistent with scale in figure 3.1.



(a) $t = 0$

(b) $t \approx 0.01 L_x/\Delta u$

(c) $t \approx 0.05 L_x/\Delta u$

(d) $t \approx 0.1 L_x/\Delta u$

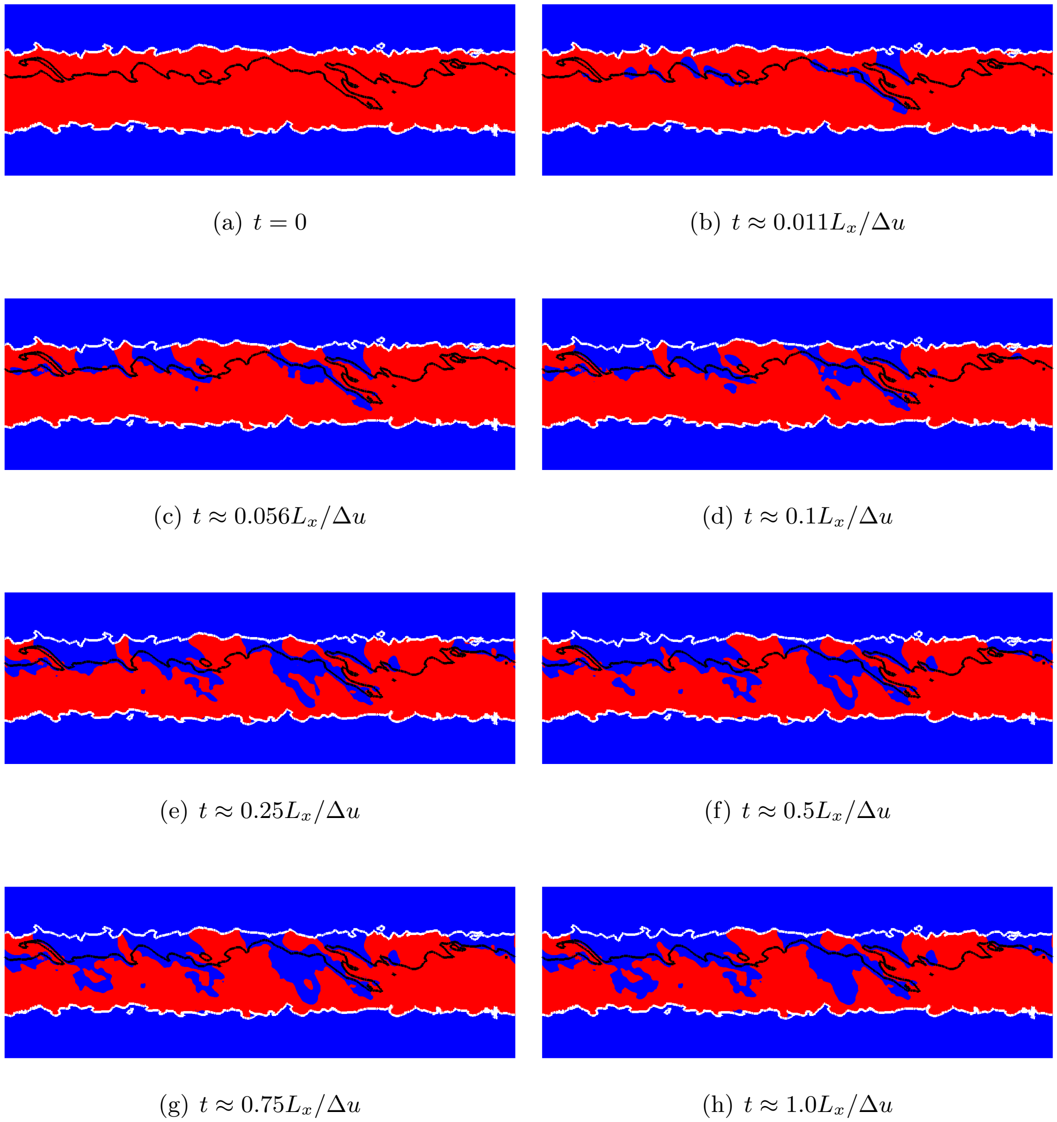
(e) $t \approx 0.25 L_x/\Delta u$

(f) $t \approx 0.5 L_x/\Delta u$

(g) $t \approx 0.75 L_x/\Delta u$

(h) $t \approx 1.0 L_x/\Delta u$

Figure 4.12: Flame hole formation and evolution on Σ_{st} from a shear layer DNS. Blue regions correspond to "flame holes". The stoichiometric mixture fraction ($Z_{\mathrm{st}} = 0.2$) is shown in black.



(a) $t = 0$

(b) $t \approx 0.011 L_x/\Delta u$

(c) $t \approx 0.056 L_x/\Delta u$

(d) $t \approx 0.1 L_x/\Delta u$

(e) $t \approx 0.25 L_x/\Delta u$

(f) $t \approx 0.5 L_x/\Delta u$

(g) $t \approx 0.75 L_x/\Delta u$

(h) $t \approx 1.0 L_x/\Delta u$

Figure 4.13: Spanwise cut through shear layer showing φ extended into the domain. Blue regions correspond to “flame holes”. The stoichiometric mixture fraction ($Z_{\mathrm{st}} = 0.2$) is shown as a black curve and the exterior boundaries ($Z = 0.001$ and $Z = 0.999$) are shown as a pair of white curves.

see Pantano et al. [63]. Figure 4.9 shows side, spanwise cuts and a three-dimensional rendering of the Z from the DNS with the stoichiometric location highlighted by a black curve. This simulation is described in detail in the reference, but its key properties are: turbulence Reynolds number of $Re_{\lambda,peak} = 82$ at the time we applied flame hole dynamics, grid size of $768 \times 258 \times 192$, model combustion of methane with air at $Z_{\rm st} = 0.2$. Figure 4.10 shows the probability-density-function (PDF) of velocity and scalar dissipation at $\Sigma_{\rm st}$, highlighting the broad variation of different fields around the flame surface. Note that the PDF of scalar dissipation, in particular, is shown in logarithmic coordinates due to the large variation of extreme high (and low) values of χ on the stoichiometric surface (since it is a highly intermittent quantity). The flame hole dynamics algorithm was used on this case with the flow velocity and mixture fraction provided by the DNS. The particular values of the crossover, χ_0 and quenching, $\chi_{\rm q}$, scalar dissipation used in the edge flame velocity model, Eq. (A.1), were chosen to trigger partial extinction of the flame thoughout the domain. These are indicated in figure 4.10(b), where $\chi_0/\langle\chi\rangle = 1$, and $\chi_{\rm q}/\langle\chi\rangle = 2.86$ (resulting in roughly a third of the stoichiometric surface being below the quenching dissipation, with a ramp parameter of $\Delta_\chi/\langle\chi\rangle = 0.2$, see Eq. (A.2)), where angle brackets denote the mean value of χ on $\Sigma_{\rm st}$. A flame hole quenching time of $\tau_{\rm q} = 4.0 \times 10^{-3} L_x/\Delta u$, where Δu is the difference between the free stream velocities in the shear layer, is used so that flame holes form quickly relative to the advection and FHB propagation time scales. The FHB propagation uses a laminar flame speed of $S_L = 0.1\Delta u \approx 0.3\langle u\rangle$ and a maximum negative flame speed of $V_{e,\max} = -3S_L$ so that the advection and FHB propagation occur at similar time scales. The flame state, φ, at the stoichiometric surface is propagated into the domain by the extension operation, giving information about the gas composition at any point between the upper and lower exterior boundaries of the shear layer. Through numerical experimentation, a reference value of extension dissipation of $\varepsilon_0 = 1.0 \times 10^{-4}$ with characteristic length scale $\ell = 10(\delta_\omega/\Delta u)^{1/2}$, defined in Section 3.1 where δ_ω is the vorticity thickness, see Pantano et al. [63], was found to be sufficient for rapid convergence of the multigrid solver of the extension algorithm. Critical points are detected by comparing with a value of $g_\epsilon = 0.05|\nabla Z|_{\rm avg}$ (averaged over the domain of extension), with the Robin boundary condition parameterized by $\alpha_R = 0.25/(\varepsilon_0(1 + \ell/\Delta))$. The interface is compressed with a subiteration CFL of 0.0025 and $\epsilon_c = 2.0 \min(\Delta x, \Delta y, \Delta z)$. Figure 4.11 shows a two-dimensional cut of the extended

φ for a section of 3D turbulent shear layer containing both critical points and exterior boundaries. The values on Σ_{st} were initialized as a flame disc through the center of the shear layer and they are extended normal to the iso-surfaces of Z by the extension algorithm. Note that the extension algorithm does not always propagate the value of φ all the way to the shear layer boundaries because of the three-dimensional nature of Σ_{st} (∇Z has a component off the plane shown in figure 4.11).

Figure 4.12 shows the solution of the flame hole dynamics model Eq. (2.33) at several instants in time for the shear layer, with φ initialized to be 1 everywhere on Σ_{st}. Flame holes form in regions where the scalar dissipation exceeds the quenching dissipation and expand due to FHB propagation and the flow velocity until an equilibrium is reached where the FHB propagation is counter balanced by the flow velocity ($|d\mathbf{r}_i/dt| = \mathbf{u}\cdot\mathbf{m} + V_e = 0$; see Eq. (2.6)). Flame holes, flame islands, and flame strips all develop and propagate on the complicated and realistic Σ_{st}. The φ field is extended into the domain (Figure 4.13) and is available to be used for evaluation of gas composition by flamelet/mixing models.

Chapter 5

Reacting Flow Formulation

A goal of this research is to apply the flame hole dynamics model to increasingly realistic flow scenarios. To this end, our goal for the model is to apply it to a fully coupled simulation a piloted jet flame configuration. We chose to construct our simulation to model the Sandia "F" flame [1]. The Sandia "F" flame is a highly turbulent piloted jet flame that exhibits high levels of local extinction (and is close to global extinction). Our intent is to perform a massively parallel simulation and compare the results with the experimental data.

In order to apply the model, the method needs to be coupled with a fluid dynamics solver (see Chapter 6) and needs formulation to describe the reaction and its effect on the flow. Two reacting flow models are developed for the simulation of the Sandia "F" flame, a flamelet model and a Burke-Schumann model. Both are linked with the flame state variable to describe the gas composition and temperature of the mixture. Both the flamelet and Burke-Schumann approximations rely on the advection of conserved variables, the mixture fraction (or, in the case of multiple streams, the mixture fraction variables), to describe the extent of mixing in the turbulent flow. The flame hole dynamics model itself makes no particular assumption about the reacting flow model, and the assumptions made in this section do not necessarily need to be made to implement Eq. (2.33) (see Section 2.1).

5.1 Conserved Scalar Approach

A one-step reaction model is chosen to describe the chemical process,

$$\sum_{i=1}^{N_{\text{spec}}} M_i \nu_i' = \sum_{i=1}^{N_{\text{spec}}} M_i \nu_i'', \tag{5.1}$$

where M_i denotes the chemically active species in the reacting flow. Assuming species Fickian diffusion with equal species diffusivities, a transport equation for the species i can be written as

$$\rho \frac{DY_i}{Dt} - \nabla \cdot (\rho D_{\text{mix}} \nabla Y_i) = W_i(\nu_i' - \nu_i'')\dot{\omega}, \tag{5.2}$$

where $\dot{\omega}$ is a species-independent reaction rate, with an overall activation energy E and a pre-exponential factor $\mathcal{B}$ (considered, for the purposes of this discussion, to be constant), is defined here as, $\dot{\omega} = \mathcal{B}\big(\prod_j^{N_{\text{spec}}}(\rho Y_j/W_j)^{\nu_j'}\big)e^{-E/RT}$. Additionally defining $(\nu_i' - \nu_i'') = \nu_i$ and a linear advection-diffusion operator $L_D(\cdot) = \rho D(\cdot)/Dt - \nabla \cdot (\rho D_{\text{mix}} \nabla(\cdot))$, the equation can be written in terms a modified mass fraction $\tilde{Y}_i = Y_i/W_i$ as

$$L_D \tilde{\mathbf{Y}} = \boldsymbol{\nu}\dot{\omega}, \tag{5.3}$$

and linear combinations of $\tilde{\mathbf{Y}}$ with a constant weight vector $\mathbf{b}$ satisfy the equation,

$$L_D(\mathbf{b}^\intercal \tilde{\mathbf{Y}}) = (\mathbf{b}^\intercal \boldsymbol{\nu})\dot{\omega}. \tag{5.4}$$

For N_{spec} independent species, given that $\boldsymbol{\nu}$ is constant, one can construct $N_{\text{spec}} - 1$ linearly independent vectors orthogonal to $\boldsymbol{\nu}$ (ignoring the trivial case where $\boldsymbol{\nu} = \mathbf{0}$), and thus there are $N_{\text{spec}} - 1$ linearly independent conserved scalars, $\boldsymbol{\beta} = \mathbf{b}^\intercal \tilde{\mathbf{Y}}$, with the condition that $\mathbf{b}^\intercal \boldsymbol{\nu} = 0$. These conserved scalars are known as the "Shvab-Zeld'ovich coupling functions" ([64]) and follow passive advection-diffusion relations,

$$L_D \boldsymbol{\beta} = 0. \tag{5.5}$$

Additionally, following Williams [31] with the assumptions of equal binary mass diffusivities, negligible viscous heating, Fourier heat conduction, and constant pressure, the total enthalpy equation can be written as,

$$\rho \frac{Dh}{Dt} = \frac{Dp}{Dt} + \nabla \cdot \left(\frac{\lambda}{c_p} \nabla h + \left(\rho D - \frac{\lambda}{c_p} \right) \sum_i h_i^\circ \nabla Y_i \right), \tag{5.6}$$

with $\overline{c_p}$ representing the average specific heat of the mixture, $\sum_i c_{p,i} Y_i$. In the event that the Lewis number,

$$\mathrm{Le} = \frac{\lambda/\bar{c}_p}{\rho D}, \tag{5.7}$$

of each of the species is unity, then the total enthalpy is also a conserved scalar,

$$L_D h = 0. \tag{5.8}$$

The conserved variables will be used, in Section 5.4 and Section 5.5 to construct the composition of the gas in the reacting flow as well as the temperature.

5.2 Mixture fraction definition for N streams

For the case where the conserved scalar equation, Eq. (5.5), is specified at N distinct boundary surfaces on a domain Ω, ($S^1, \ldots, S^N$ and $\partial\Omega = \bigcup_i^N S^i$), we can develop a mixture fraction formulation describing the mixing process as $N-1$ mixture fraction variables. Consider an arbitrary set of N linearly independent conserved scalars,

$$L_D \boldsymbol{\phi} = 0 \quad \text{and} \quad \boldsymbol{\phi}(\mathbf{x} \in S^i) = \boldsymbol{\phi}^i = \mathit{const.} \quad 1 \leq i \leq N. \tag{5.9}$$

Subtracting the state of the conserved scalars at a reference surface, $\boldsymbol{\phi}^N$, the system becomes

$$L_D(\boldsymbol{\phi} - \boldsymbol{\phi}^N) = 0 \quad \text{and} \quad \boldsymbol{\phi}(\mathbf{x} \in S^i) = (\boldsymbol{\phi}^i - \boldsymbol{\phi}^N) = \mathit{const.} \quad 1 \leq i \leq N-1. \tag{5.10}$$

$$\boldsymbol{\phi}(\mathbf{x} \in S^N) = \mathbf{0}. \tag{5.11}$$

Instead of solving this set of equations, we seek a specially defined set of $N-1$ conserved scalars that obey a normalized equation set,

$$L_D \boldsymbol{Z} = 0 \quad \text{and} \quad Z_j(\mathbf{x} \in S^i) = \delta_{i,j} \quad 1 \leq i \leq N-1,\ 1 \leq j \leq N-1, \tag{5.12}$$

where $\delta_{i,j}$ is the Kronecker delta ($\delta_{i,j} = 1$ if $i = j$ and 0 otherwise). This is now effectively a two boundary problem, with $Z_j = 1$ at the j^{th} boundary and zero at the other boundaries collectively.

As it is a passive scalar advection-diffusion equation, the maximum/minimum principle is obeyed and the value of Z_j remains bounded by its values at the boundaries, implying that $0 \leq Z_j \leq 1$ everywhere. Sums of the mixture variables—i.e., $\sum_{j=1}^{M} Z_j$ where M is an arbitrary number between 1 and $N-1$—also have boundary conditions that are between 0 and 1, and so also take values in the range of 0 and 1, $0 \leq \sum_{j=1}^{M} Z_j \leq 1$.

The transformation from the arbitrarily chosen set of conserved variables $\boldsymbol{\phi}$ to $\boldsymbol{Z}$ can be constructed in terms of a linear combination of the state of the conserved variables at the boundaries,

$$\phi_i^N + \sum_{j=1}^{N-1} Z_j(\phi_i^j - \phi_i^N) = \phi_i. \tag{5.13}$$

Evaluating Eq. (5.13) at the boundaries implies that $Z(\mathbf{x} \in S^k)_j = \delta_{i,j}$, and so satisfies the boundary condition of Eq. (5.12). Expressing this as a linear system produces the relationship that

$$\sum_{j=1}^{N-1} B_{i,j} Z_j = \phi_i - \phi_i^N \quad 1 \leq i \leq N-1 \tag{5.14}$$

where

$$\mathbf{B} = \begin{pmatrix} \phi_1^1 - \phi_1^N & \phi_1^2 - \phi_1^N & \cdots & \phi_1^{N-1} - \phi_1^N \\ \phi_2^1 - \phi_2^N & \phi_2^2 - \phi_2^N & \cdots & \phi_2^{N-1} - \phi_2^N \\ \vdots & \vdots & \vdots & \vdots \\ \phi_{N-1}^1 - \phi_{N-1}^N & \phi_{N-1}^2 - \phi_{N-1}^N & \cdots & \phi_{N-1}^{N-1} - \phi_{N-1}^N \end{pmatrix}. \tag{5.15}$$

Given the set of $\boldsymbol{\phi}^i - \boldsymbol{\phi}^N$ are linearly independent (if not, the number of surfaces and mixture fraction variables can be reduced), then $\mathbf{B}^{-1}$ exists and the mixture fraction variables can be defined as

$$Z_i \equiv \sum_{j=1}^{N-1} B_{i,j}^{-1}(\phi_i - \phi_i^N) \quad 1 \leq i \leq N-1. \tag{5.16}$$

Given that the boundary conditions are constant, then it's also true that $L_D \boldsymbol{Z} = 0$ as L_D is linear and $\mathbf{B}^{-1}$ is constant. The mixture fraction variables, $\boldsymbol{Z}$, with this definition, satisfy Eq. (5.12); furthermore, the number of mixture fraction variables is one less than the number of boundary surfaces. While the definition of the mixture fraction depends on the particular choice of conserved scalars, the mixture fraction will follow the same governing equation (Eq. (5.12)) irrespective of the set of conserved scalars used to define it. Because of this, any conserved scalar can be constructed from $\boldsymbol{Z}$ by the mixing relationship, Eq. (5.13).

5.3 Three-stream methane-air system

For a methane-air reaction with a one-step chemistry approximation, there are four chemically active species,

$$\mathrm{CH_4} + 2\mathrm{O_2} \rightarrow 2\mathrm{H_2O} + \mathrm{CO_2}. \tag{5.17}$$

Three linearly independent, conserved Shvab-Zeld'ovich coupling functions can be constructed as

$$\beta_O = 2\frac{Y_{\mathrm{O_2}}}{W_{\mathrm{O_2}}} + \frac{Y_{\mathrm{H_2O}}}{W_{\mathrm{H_2O}}} + 2\frac{Y_{\mathrm{CO_2}}}{W_{\mathrm{CO_2}}} \tag{5.18}$$

$$\beta_C = \frac{Y_{\mathrm{CH_4}}}{W_{\mathrm{CH_4}}} + \frac{Y_{\mathrm{CO_2}}}{W_{CO_2}} \tag{5.19}$$

$$\beta_H = 4\frac{Y_{\mathrm{CH_4}}}{W_{\mathrm{CH_4}}} + 2\frac{Y_{\mathrm{H_2O}}}{W_{\mathrm{H_2O}}}. \tag{5.20}$$

One particularly useful combination of these the conserved scalars is

$$\hat{\beta} = 2\beta_C + \frac{1}{2}\beta_H - \beta_O = 4\frac{Y_{\mathrm{CH_4}}}{W_{\mathrm{CH_4}}} - 2\frac{Y_{\mathrm{O_2}}}{W_{\mathrm{O_2}}}, \tag{5.21}$$

which specifies the stoichiometry by the condition $\hat{\beta} = 0$. Without differential diffusion, this conserved scalar yields an identical mixture fraction to the Bilger mixture fraction when normalized between the jet and coflow,

$$\hat{Z} = \frac{\hat{\beta} - \hat{\beta}^{\mathrm{coflow}}}{\hat{\beta}^{\mathrm{jet}} - \hat{\beta}^{\mathrm{coflow}}}. \tag{5.22}$$

One can also define a normalized scalar based on the total enthalpy between the pilot and the coflow as

$$Z_h = \frac{h - h^{\mathrm{coflow}}}{h^{\mathrm{pilot}} - h^{\mathrm{coflow}}}. \tag{5.23}$$

As the choice of conserved scalar used to define the normalized mixture fractions is arbitrary, we choose Z and Z_h and construct the mixture fraction definitions implicitly as

$$\begin{pmatrix} 1 & \hat{Z}^{\text{pilot}} \\ Z_h^{\text{jet}} & 1 \end{pmatrix} \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} = \begin{pmatrix} \hat{Z} \\ Z_h \end{pmatrix}.$$

The normalized mixture fractions are therefore

$$Z_1 = \hat{Z} - \frac{\hat{Z}^{\text{pilot}}}{\Delta_Z} Z_h \quad \text{and} \quad Z_2 = Z_h - \frac{Z_h^{\text{jet}}}{\Delta_Z} \hat{Z} \tag{5.24}$$

where $\Delta_Z = 1 - \hat{Z}^{\text{pilot}} Z_h^{\text{jet}}$. We maintain that $Z_1 + Z_2 \leq 1$ as indicated in Section 5.2. If for instance both Z_1 and Z_2 are equal to 1, then it would indicate that the state of the conservative is instantaneously that of the pilot and the jet, which is not consistent. From this relationship, the composition of any conserved scalar can be computed through $\phi_i = \phi_i^{\text{coflow}} + (\phi_i^{\text{jet}} - \phi_i^{\text{coflow}})Z_1 + (\phi_i^{\text{pilot}} - \phi_i^{\text{coflow}})Z_2$. The stoichiomeric mixture is defined when $2Y_{\text{CH}_4}/W_{\text{CH}_4} = Y_{\text{O}_2}/W_{\text{O}_2}$. In the case of Z, the stoichiometric fraction is a constant value, given by $\hat{\beta} = 0$,

$$\hat{Z}_{\text{st}} = \frac{-\hat{\beta}^{\text{coflow}}}{\hat{\beta}^{\text{jet}} - \hat{\beta}^{\text{coflow}}} \tag{5.25}$$

and so the stoichiometric mixture fraction for the normalized mixture fraction "1" (which marks the stream containing fuel) is

$$Z_{1,\text{st}} = \hat{Z}_{\text{st}} - \hat{Z}^{\text{pilot}} Z_2. \tag{5.26}$$

When there is no reaction, $\dot{\omega} = 0$, and therefore each of the chemical species follows a passive scalar-advection equation $L_D Y_i = 0$. The mass fraction of any of the species as well as the total enthalpy (given the assumptions in Section 5.1) can then be constructed from Z_1 and Z_2 as

$$Y_i^{\text{mix}} = Y_i^{\text{coflow}} + (Y_i^{\text{jet}} - Y_i^{\text{coflow}})Z_1 + (Y_i^{\text{pilot}} - Y_i^{\text{coflow}})Z_2 \tag{5.27}$$

$$h = h^{\text{coflow}} + (h^{\text{jet}} - h^{\text{coflow}})Z_1 + (h^{\text{pilot}} - h^{\text{coflow}})Z_2. \tag{5.28}$$

Any conserved scalar quantity can be constructed in the same manner as well out of the two mixture fraction variables.

5.4 Flamelets with multiple mixture fractions

One issue with the three-stream system is that one cannot define a simple extinction condition based on one mixture fraction variable. However, we can proceed with a transformation into the mixture coordinates. In a two-stream system, the flame thickness is small around a particular location, the stoichiometric surface. With a three-stream problem, this is not so obvious. However, noting that there is no fuel in the pilot, we can assume the flame is concentrated in a thin region around $Z_{1,\mathrm{st}}(Z_2)$ and proceed systematically with an asymptotic expansion around this surface. There is no evidence in the Sandia flames that the flame is of a structure different from this, and it is assumed in Barlow and Frank [1] that the system can be described in terms of $\hat{Z}$.

Noting the mass fraction equation with a species-dependent reaction-rate term $w_i = W_i \nu_i \dot{\omega}$ (see Section 5.1 for the definitions of these terms), the species equation can be written as

$$\rho \frac{DY_i}{Dt} = \nabla \cdot (\rho D_{\mathrm{mix}} \nabla Y_i) + \dot{w}_i. \tag{5.29}$$

We anticipate that the left-hand side $\rho DY_i/Dt$ is small and introduce a variable transformation $Z_1 = Z_{1,\mathrm{st}}(\eta) + \epsilon\zeta$ and $Z_2 = \eta$,

$$\begin{pmatrix} \frac{\partial \zeta}{\partial Z_1} & \frac{\partial \zeta}{\partial Z_2} \\ \frac{\partial \eta}{\partial Z_2} & \frac{\partial \eta}{\partial Z_2} \end{pmatrix} = \begin{pmatrix} \frac{1}{\epsilon} & -\frac{Z'_{1,\mathrm{st}}}{\epsilon} \\ 0 & 1 \end{pmatrix}. \tag{5.30}$$

Another method of creating two-dimensional laminar flamelet equations has been developed by Hasse and Peters [65] using an asymptotic expansion with both Z_1 and Z_2 scaled around a constant reference mixture fraction value designed for a two fuel, one oxidizer feed system. However, we chose to expand the flamelet equations around Z_1, the stream with fuel, relative to the stoichiometric mixture fraction, which is a function of Z_2. Applying this transformation, using Einstein summation, the mass fraction equations results in

$$\frac{\partial}{\partial x_j} \left[\rho D_{\mathrm{mix}} \left(\frac{1}{\epsilon} \frac{\partial Y_i}{\partial \zeta} \left(\frac{\partial Z_1}{\partial x_j} - Z'_{1,\mathrm{st}} \frac{\partial Z_2}{\partial x_j} \right) + \frac{\partial Y_i}{\partial \eta} \frac{\partial Z_2}{\partial x_j} \right) \right] + w_i = 0. \tag{5.31}$$

Given ϵ as a small parameter, the $1/\epsilon$ terms dominate the order one terms. Expanding the last derivative results in

$$\frac{1}{\epsilon}\frac{\partial}{\partial x_j}\left[\rho D_{\text{mix}}\frac{\partial Y_i}{\partial \zeta}\left(\frac{\partial Z_1}{\partial x_j} - Z'_{1,\text{st}}\frac{\partial Z_2}{\partial x_j}\right)\right] + w_i = \tag{5.32}$$

$$\frac{1}{\epsilon}\left\{\frac{\partial}{\partial x_j}\left[\rho D_{\text{mix}}\left(\frac{\partial Z_1}{\partial x_j} - Z'_{1,\text{st}}\frac{\partial Z_2}{\partial x_j}\right)\right]\frac{\partial Y_i}{\partial \zeta} + \left[\rho D_{\text{mix}}\left(\frac{\partial Z_1}{\partial x_j} - Z'_{1,\text{st}}\frac{\partial Z_2}{\partial x_j}\right)\right]\frac{\partial}{\partial x_j}\frac{\partial Y_i}{\partial \zeta}\right\} + \dot{w}_i = 0. \tag{5.33}$$

The only term expected to change on a scale of order ϵ is Y_i, so we can neglect the first term and arrive at

$$\frac{1}{\epsilon^2}\left[\rho D_{\text{mix}}\left(\frac{\partial Z_1}{\partial x_j} - Z'_{1,\text{st}}\frac{\partial Z_2}{\partial x_j}\right)^2\right]\frac{\partial Y_i}{\partial \zeta^2} + \dot{w}_i = 0. \tag{5.34}$$

Reverting back from the change of variables, retaining the most significant term, produces a flamelet-type equation

$$-\frac{1}{2}\rho\hat{\chi}\frac{\partial^2 Y_i}{\partial Z_1^2} = \dot{w}_i. \tag{5.35}$$

with a specially defined scalar dissipation

$$\hat{\chi} = 2D_{\text{mix}}|\nabla Z_1 - Z'_{1,\text{st}}\nabla Z_2|^2 = 2D_{\text{mix}}|\nabla \hat{Z}|^2. \tag{5.36}$$

The second mixture fraction appears parametrically in the flamelet equations and is present in the boundary conditions. The flamelet treatment is consistent in the description of the three-feed system for the piloted Sandia “F” flames, and other three-feed systems with similar conditions. Note that the proper Z_1 and Z_2 remove problems that others have encountered when attempting to define mixture fractions in three-feed systems [66, 67].

5.5 Burke-Schumann approximation

If the reaction rate is infinitely fast, then the reaction occurs at an infinitesimally thin surface where all fuel and air is consumed, $Y_{CH_4}(Z_1 = Z_{1,\text{st}}, Z_2) = Y_{O_2}(Z_1 = Z_{1,\text{st}}, Z_2) = 0$. For an infinitely thin sheet, assuming the scalar dissipation is nowhere 0, away from the reaction zone we have, noting

Eq. (5.35),

$$\frac{\partial^2 Y_{CH_4}}{\partial Z_1^2} = 0 \quad \text{and} \quad Y_{CH_4}(Z_1 \to Z_{1,\text{st}}^-, Z_2) = 0 \quad Z_1 < Z_{1,\text{st}}, \tag{5.37}$$

$$\frac{\partial^2 Y_{CH_4}}{\partial Z_1^2} = 0 \quad \text{and} \quad Y_{CH_4}(Z_1 \to Z_{1,\text{st}}^+, Z_2) = 0 \quad Z_1 > Z_{1,\text{st}}. \tag{5.38}$$

The solution is then

$$Y_{CH_4}(Z_1, Z_2) = \begin{cases} f(Z_2)(Z_{1,\text{st}} - Z_1) & Z_1 \leq Z_{1,st}, \\ g(Z_2)(Z_1 - Z_{1,\text{st}}) & Z_1 > Z_{1,st}, \end{cases} \tag{5.39}$$

with two functions $f(Z_2)$ and $g(Z_2)$ that describe the boundary condition at $Z_1 = 0$ and $Z_1 = 1 - Z_2$ respectively. $f(Z_2)$ and $g(Z_2)$ can be determined considering the equation along the $Z_1 = 0$—in our case, $Y_{CH_4}(Z_1 = 0, Z_2) = 0$ as both the oxidizer and pilot do not contain fuel—and $Z_1 = 1 - Z_2$ lines,

$$\frac{\partial^2 Y_{\text{CH}_4}}{\partial (1 - Z_2)^2} = 0 \quad \text{and} \quad Y_{CH_4}(1 - Z_2 \to Z_{1,\text{st}}^-, Z_2) = 0 \quad 1 - Z_2 < Z_{1,\text{st}} \tag{5.40}$$

$$\frac{\partial^2 Y_{\text{CH}_4}}{\partial (1 - Z_2)^2} = 0 \quad \text{and} \quad Y_{\text{CH}_4}(1 - Z_2 \to Z_{1,\text{st}}^+, Z_2) = 0 \quad 1 - Z_2 > Z_{1,\text{st}} \tag{5.41}$$

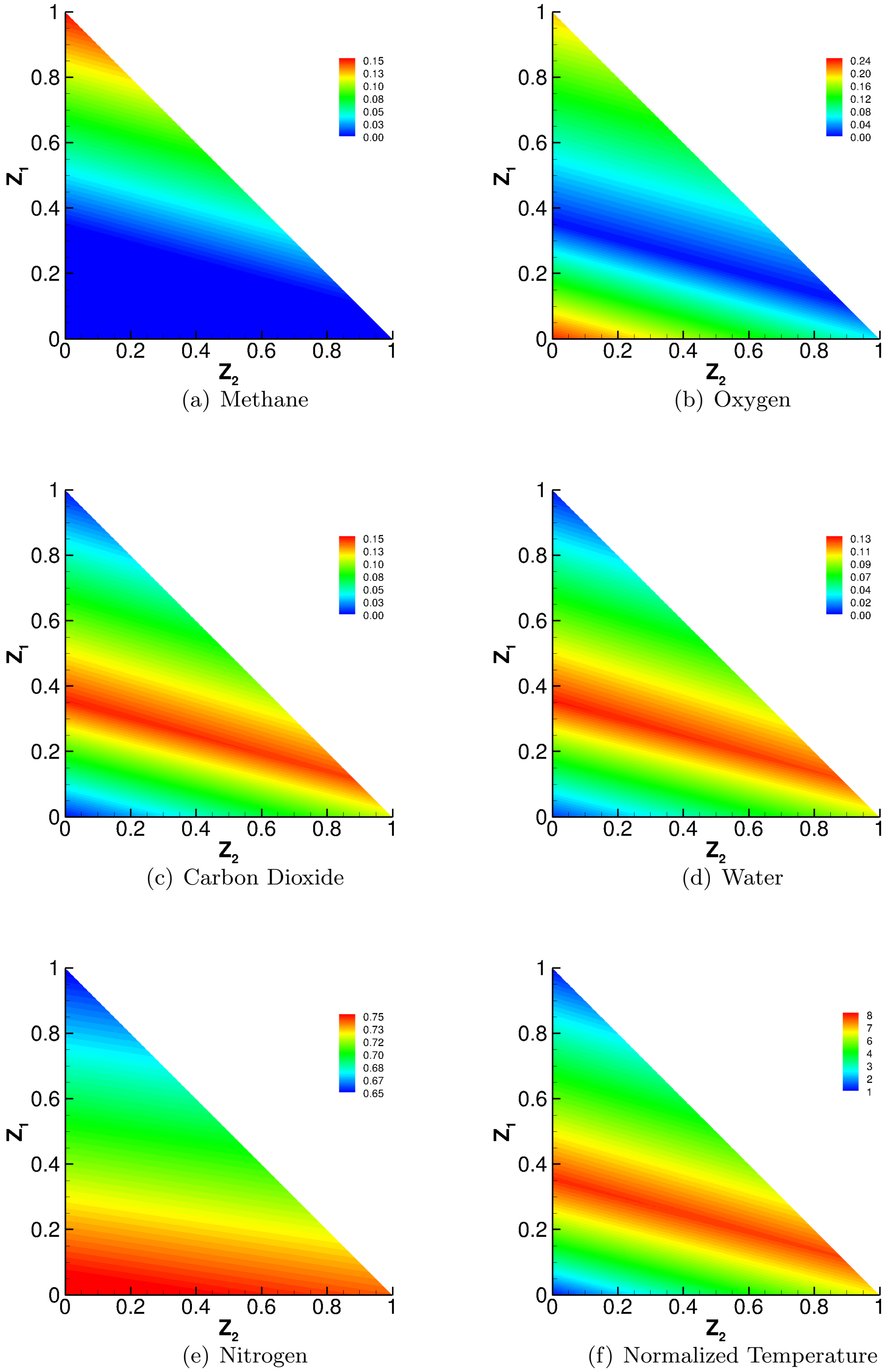
The boundary conditions for the equation on each side of the stoichiometric value are given by the composition of the pilot and jet streams as

$$Y_{\text{CH}_4}(0, 1) = Y_{\text{CH}_4}^{\text{pilot}} = 0 \quad \text{and} \quad Y_{\text{CH}_4}(1, 0) = Y_{\text{CH}_4}^{\text{jet}}. \tag{5.42}$$

Using these boundary conditions, the mass fraction of methane can be reduced to

$$Y_{\text{CH}_4}(Z_1, Z_2) = \begin{cases} Y_{\text{CH}_4}^{\text{jet}} \frac{Z_1 - Z_{1,\text{st}}}{1 - \hat{Z}_{\text{st}}} & Z_1 > Z_{1,\text{st}} \\ 0 & Z_1 \leq Z_{1,\text{st}}, \end{cases} \tag{5.43}$$

which is exactly the result for the univariate Burke-Schumann based on $\hat{Z}$, noting that $\hat{Z} = Z_1 - \hat{Z}^{\text{pilot}} Z_2$. The mass fractions of the remaining species can be constructed by use the coupling



(a) Methane

(b) Oxygen

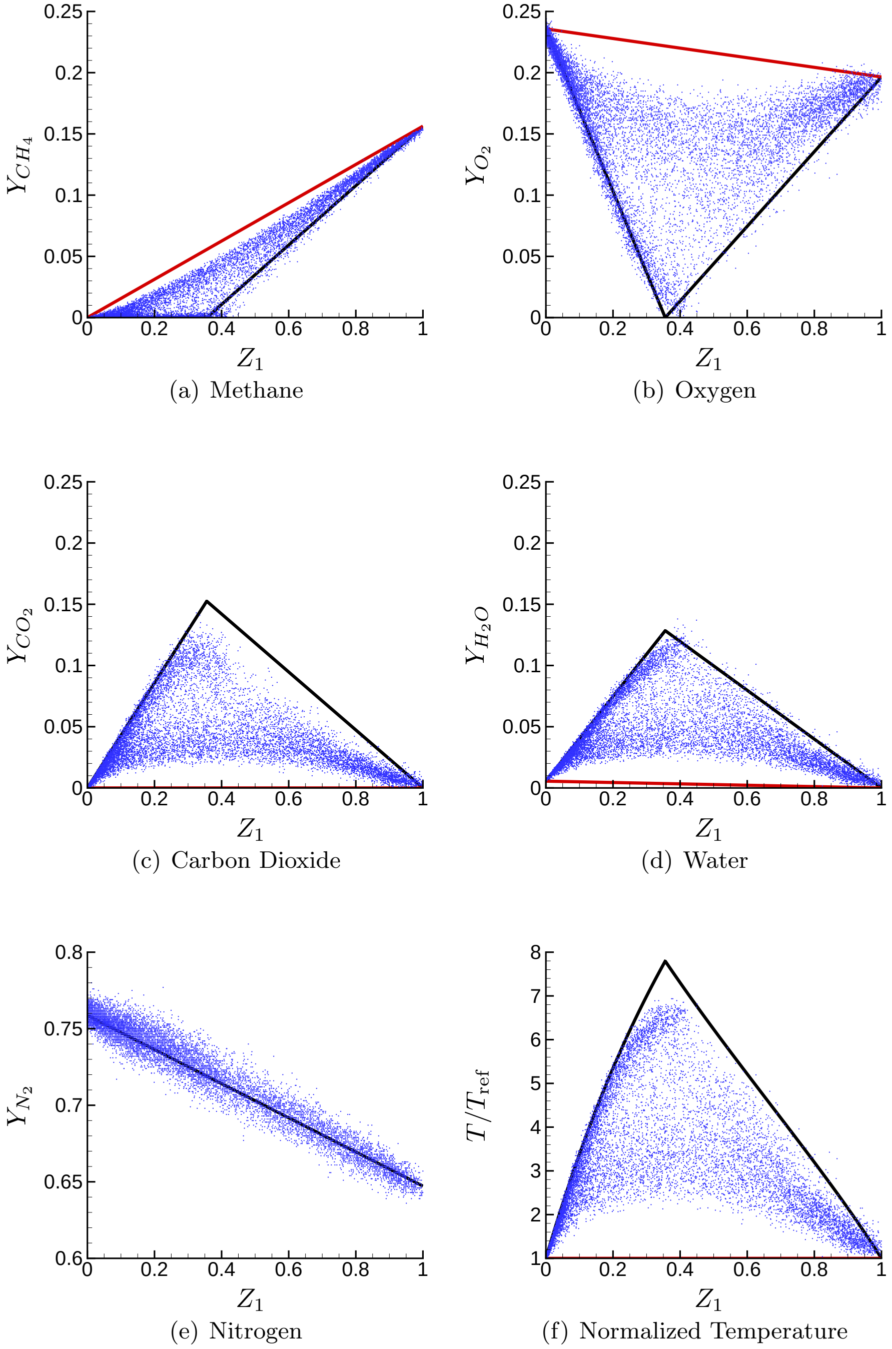
(c) Carbon Dioxide

(d) Water

(e) Nitrogen

(f) Normalized Temperature

Figure 5.1: Mass fractions and temperature using the Burke-Schumann approximation for two mixture fraction variables. The temperature is normalized by the temperature of the main jet (and oxidizer), $T_{\text{ref}} = 298K$



(a) Methane

(b) Oxygen

(c) Carbon Dioxide

(d) Water

(e) Nitrogen

(f) Normalized Temperature

Figure 5.2: Mass fractions and temperature for the Burke-Schumann approximation at $Z_2 = 0$ (black line) and the mixing solution (red line) against experimental data (blue dots). The reference temperature is that of the main jet and oxidizer streams, $T_{\text{ref}} = 298K$.

Species	$c_p^0/c_{p,\mathrm{O}_2}$	$c_p^1/c_{p,\mathrm{O}_2}T_{\mathrm{ref}}$	W_i	$\Delta h^\circ/c_{p,\mathrm{O}_2}T_{\mathrm{ref}}$
Methane	2.466	0.8743	16.04	−17.15
Oxygen	1.000	0.0639	32.00	0
Water	2.011	0.2187	18.02	−49.81
Carbon Dioxide	0.9347	0.1321	44.01	−33.17
Nitrogen	1.140	0.0553	28.01	0

Table 5.1: Specific heats, molecular weights, and formation enthalpies for the species. Reference specific heat is that of air at $T_{\mathrm{ref}} = 298K$, $c_{p,\mathrm{O}_2} = 905\mathrm{J\ kg}^{-1}\ \mathrm{K}^{-1}$

.

Stream	Y_{CH_4}	Y_{O_2}	$Y_{\mathrm{H}_2\mathrm{O}}$	Y_{CO_2}	T/T_{ref}
Jet	0.1563	0.1965	0	0	1.000
Pilot	0	0.0560	0.0920	0.1100	6.309
Coflow	0	0.2357	0.0056	0	1.000

Table 5.2: Mass fractions of species and temperature at the three streams used in the Burke Schumann and mixing solutions. Calculated from the elemental mass fractions and species mass fractions provided in Barlow and Frank [1]. All remaining mass is considered to be Nitrogen.

relationships and the mass fraction of methane, as

$$\hat{\beta} = \hat{\beta}^{\mathrm{coflow}} + (\hat{\beta}^{\mathrm{jet}} - \hat{\beta}^{\mathrm{coflow}})Z_1 + (\hat{\beta}^{\mathrm{pilot}} - \hat{\beta}^{\mathrm{coflow}})Z_2 = 4\frac{Y_{\mathrm{CH}_4}}{W_{\mathrm{CH}_4}} - 2\frac{Y_{\mathrm{O}_2}}{W_{\mathrm{O}_2}} \tag{5.44}$$

$$\beta_C = \beta_C^{\mathrm{coflow}} + (\beta_C^{\mathrm{jet}} - \beta_C^{\mathrm{coflow}})Z_1 + (\beta_C^{\mathrm{pilot}} - \beta_C^{\mathrm{coflow}})Z_2 = \frac{Y_{\mathrm{CH}_4}}{W_{\mathrm{CH}_4}} + \frac{Y_{\mathrm{CO}_2}}{W_{\mathrm{CO}_2}} \tag{5.45}$$

$$\beta_H = \beta_H^{\mathrm{coflow}} + (\beta_H^{\mathrm{jet}} - \beta_H^{\mathrm{coflow}})Z_1 + (\beta_H^{\mathrm{pilot}} - \beta_H^{\mathrm{coflow}})Z_2 = 4\frac{Y_{\mathrm{CH}_4}}{W_{\mathrm{CH}_4}} + 2\frac{\mathrm{Y}_{\mathrm{H}_2\mathrm{O}}}{W_{\mathrm{H}_2\mathrm{O}}} \tag{5.46}$$

Rearranging, the mass fractions of oxygen, carbon dioxide, and water can be obtained using Eq. (5.43) and Eq. (5.44),

$$Y_{\mathrm{O}_2} = -\frac{1}{2}W_{\mathrm{O}_2}\hat{\beta}(Z_1, Z_2) + 2\frac{W_{\mathrm{O}_2}}{W_{\mathrm{CH}_4}}Y_{\mathrm{CH}_4}(Z_1, Z_2) \tag{5.47}$$

$$Y_{\mathrm{CO}_2} = W_{\mathrm{CO}_2}\beta_C(Z_1, Z_2) - \frac{W_{\mathrm{CO}_2}}{W_{\mathrm{CH}_4}}Y_{\mathrm{CH}_4}(Z_1, Z_2) \tag{5.48}$$

$$Y_{\mathrm{H}_2\mathrm{O}} = \frac{1}{2}W_{\mathrm{H}_2\mathrm{O}}\beta_H(Z_1, Z_2) - 2\frac{W_{\mathrm{H}_2\mathrm{O}}}{W_{\mathrm{CH}_4}}Y_{\mathrm{CH}_4}(Z_1, Z_2), \tag{5.49}$$

and the composition of the gas is completely determined with $Y_{\mathrm{N}_2} = 1 - \sum_j^{N_{\mathrm{spec}}} Y_j$. Figure 5.1 shows the Burke-Schumann as a function of Z_1 and Z_2 for the applicable domain $Z_1 + Z_2 \leq 1$.

A comparison with the experimental mass fractions and temperature is shown in Figure 5.2, with $Z_2 = 0$. The experimental data is obtained from the *International Workshop on Measurement and Computation of Turbulent Nonpremixed Flames* (Barlow and Frank [1], see Table 5.1), from which all the parameters are obtained. As the experimental data is taken at 15 jet diameters downstream from the jet, the pilot mixture fraction is expected to be exponentially small, justifying the choice of $Z_2 = 0$ for the comparison. The temperature can be calculated using the total enthalpy by inverting the relationship

$$h(Z_1, Z_2) = \sum_{i}^{N_{\text{spec}}} Y_i h_i, \quad \text{with} \quad h_i = \Delta h_i^\circ + \int_{T^{\text{coflow}}}^{T} c_p(T')dT', \tag{5.50}$$

where the specific heats are allowed to vary linearly with temperature, $c_p = c_p^0 + c_p^1(T/T_{\text{ref}} - 1)$ according to Table 5.1. If other physical effects such as radiative heat loss is included in the enthalpy equation, then the enthalpy equation would need to be solved and the mixture fractions would need to be defined based on a different conserved variable (e.g.β_C).

The density is calculated from the ideal gas relationship,

$$p = \frac{\rho R T}{\overline{W}}, \tag{5.51}$$

with R being the universal gas constant and $\overline{W}$ denoting the average molecular weight of the mixture

$$\overline{W} = \left(\sum_{i=1}^{N_{\text{spec}}} \frac{Y_i}{W_i} \right)^{-1}. \tag{5.52}$$

The density is directly coupled into the dynamics of the flow through the momentum and mass conservation equations.

5.6 Modification of the gas state due to the flame state

In the case where the flame state is active, the gas composition and temperature are also a function of the extended flame state—which, by construction, agrees with the flame state on the surface and is extended along the $\hat{Z}$ coordinate. For the Burke-Schumann case, the composition is a blend of the mixing solution ("m") and the Burke-Schumann model ("f")

$$Y_i^{\varphi}(Z_1, Z_2, \varphi) = Y_i^{f}(Z_1, Z_2)\varphi + Y_i^{m}(Z_1, Z_2)(1 - \varphi). \tag{5.53}$$

The temperature can then be determined through the enthalpy relationship, Eq. (5.50). The affect of the flame state is coupled to the dynamics of the flow through the state equation,

$$p_0 = \rho R T^{\varphi}(Z_1, Z_2, \varphi) \sum_{i}^{N} \frac{Y_i^{\varphi}(Z_1, Z_2, \varphi)}{W_i}. \tag{5.54}$$

In a two stream configuration, only one mixture fraction Z would be required to describe the state. Additionally, if a flamelet calculation was used for the burning solution, then there would be an explicit dependence of the scalar dissipation for the state of the gas instead of the current dependence on the scalar dissipation only through the flame state variable.

Chapter 6

Implementation into an Unstructured Fluid Dynamics Solver

6.1 Details of Sierra's low Mach-number module "Nalu"

The method was implemented into Sandia National Laboratories' low-Mach number solver "Nalu". Nalu is a generalized unstructured fluid solver designed to be used for "extreme scale" turbulence simulations and has been tested in simulations of more than 9 billion nodes; see Lin et al. [68], Domino [4].

Nalu supports two discretizations: a control-volume finite element method (CVFEM) [69] and an edge-based, vertex-centered discretization. Two implicit time-integration schemes, a backward Euler and a second-order backward difference method (BDF2), are also implemented.

Nalu uses a generalized unstructured mesh in its discretization procedure. Implementing the flame hole dynamics model in this setting provides its own challenges as certain operations become more difficult to perform in an unstructured setting. The mesh, "Sierra toolkit mesh" (part of Trilinos [70]), provides tools for working with elements of the mesh. The mesh framework is parallel distributed and parallel consistent. Any modification to the mesh made by a local process is communicated to the necessary processes to ensure that the mesh is consistent. The mesh also provides upward and downward connectivity relations between nodes and elements (node to element being "upward" and element to node being "downward"). This is ensured for multiple element topologies and even for hybrid meshes containing multiple element topologies (tetrahedral and hexahedral, for instance).

The mesh framework is broken into essentially two major structures: the "meta data" structure and the "bulk data" structure. The meta data structure describes the problem being solved: it contains information about the partitions of the mesh and any fields defined on those mesh partitions. Examples of a mesh partition would be the boundary surfaces of the mesh, where one

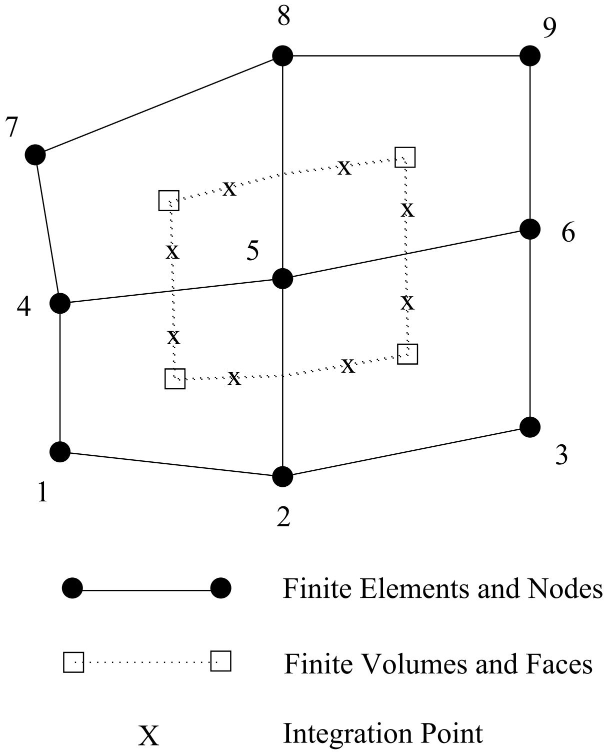


Figure 6.1: CVFEM mesh (Domino [4]).

may define boundary condition fields. This type of information is held in the meta data structure. The bulk data structure contains information regarding the mesh itself: entities (e.g.node, surface, or element), entity relationships (the aforementioned "upward" and "downward" connectivities), and the element topology associated with each section ("bucket") of the mesh. Parallel mesh consistency is also handled through this structure, and in particular interest for the flame hole dynamics implementation, the bulk data structure handles modifications to the ghosting aura of non-locally owned mesh elements. The combination of these structures allows the mesh transversal operations necessary to implement the closest point extension in an unstructured environment as well as the discretization of the partial differential equations in the method.

A scalar variable, ψ can be evaluated at the integration points (see Figure 6.1) as a weighted sum of the values at the nodes ψ_k with the basis functions N_k evaluated at the integration point α; that is,

$$\psi_\alpha \approx \sum_k N_{\alpha,k}\psi_k. \tag{6.1}$$

The derivatives can also be evaluated at the CVFEM integration points by evaluating the derivatives of the basis functions,

$$\frac{\partial \psi_\alpha}{\partial x_j} \approx \sum_k \frac{\partial N_{\alpha,k}}{\partial x_j} \varphi_k. \tag{6.2}$$

Projected nodal gradients (that is, gradients evaluated at the mesh nodes) are used extensively in the implementation and are defined, for a scalar variable φ, the lumped projected nodal gradient is

$$G_j \psi = \frac{\int_S \psi_\alpha n_j dS}{\int_V dV}, \tag{6.3}$$

where α is an integration point of the CVFEM dual volume (or, in the case of the edge-based scheme, the mid-point of the edge).

6.1.1 Low Mach number expansion used in Nalu

Nalu accounts for variable density through a asymptotic low-Mach number formulation (based on, for instance, Majda and Sethian [71]). The compressible Navier-Stokes equation set is, in dimensional form (denoted by the tilde),

$$\frac{\partial \tilde{\rho}}{\partial \tilde{t}} + \tilde{\nabla} \cdot (\tilde{\rho} \tilde{\mathbf{u}}) = 0 \tag{6.4}$$

$$\tilde{\rho} \frac{D\tilde{\mathbf{u}}}{D\tilde{t}} = -\tilde{\nabla}\tilde{p} + \tilde{\nabla} \cdot \tilde{\boldsymbol{\tau}} + \tilde{\rho}\tilde{\mathbf{g}} \tag{6.5}$$

$$\tilde{\rho} \frac{D\tilde{h}}{D\tilde{t}} = \frac{D\tilde{p}}{D\tilde{t}} + \tilde{\Phi} - \tilde{\nabla} \cdot \tilde{\mathbf{q}}, \tag{6.6}$$

with the shear stress tensor $\tilde{\boldsymbol{\tau}}$, and the viscous heating $\tilde{\Phi}$,

$$\tilde{\boldsymbol{\tau}} = \mu\left(\frac{\partial \tilde{u}_i}{\partial \tilde{x}_j} + \frac{\partial \tilde{u}_j}{\partial \tilde{x}_i}\right) + \left(\kappa - \frac{2}{3}\mu\right)\frac{\partial \tilde{u}_k}{\partial \tilde{x}_k}\delta_{ij}. \tag{6.7}$$

$$\tilde{\Phi} = \frac{1}{2}\mu\left(\frac{\partial \tilde{u}_i}{\partial \tilde{x}_j} + \frac{\partial \tilde{u}_j}{\partial \tilde{x}_i}\right)^2 + \left(\kappa - \frac{2}{3}\mu\right)\left(\frac{\partial \tilde{u}_k}{\partial \tilde{x}_k}\right)^2. \tag{6.8}$$

The heat flux vector will be defined precisely later in this section (Eq. (6.18)). The equation is scaled with a reference length, velocity, pressure, density, thermal diffusivity, specific heat, ratio of specific heats, mean molecular weight, viscosity, and temperature: L, U_0, P_0, ρ_0, λ_0, γ_0, $\overline{W}_0$, and T_0. The time scale is constructed from the velocity and length scales. In Cartesian coordinates, using Einstein notation, the nondimensional equations are

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_j}{\partial x_j} = 0, \tag{6.9}$$

$$\frac{\partial(\rho u_j)}{\partial t} + \frac{\partial(\rho u_i u_j)}{\partial x_j} = -\frac{1}{\gamma \mathrm{Ma}^2}\frac{\partial p}{\partial x_i} + \frac{1}{\mathrm{Re}}\frac{\partial \tau_{ij}}{\partial x_j} + \frac{\rho}{Fr_i}, \tag{6.10}$$

$$\frac{\partial(\rho h)}{\partial t} + \frac{\partial(\rho u_j h)}{\partial x_j} = \frac{\gamma - 1}{\gamma}\frac{DP}{Dt} - \frac{1}{\mathrm{Re}\,\mathrm{Pr}}\frac{\partial q_j}{\partial x_j} + \frac{\gamma-1}{\gamma}\frac{\mathrm{Ma}^2}{\mathrm{Re}}\frac{\partial \Phi}{\partial x_j}. \tag{6.11}$$

The equation set has four nondimensional parameters,

$$\mathrm{Re} = \frac{\rho_0 U_0 L}{\mu_0}, \quad \mathrm{Pr} = \frac{\mu_0 c_{p,0}}{\lambda_0}, \quad \mathrm{Fr}_i = \frac{U_0^2}{g_i L}, \quad \text{and} \quad \mathrm{Ma}^2 = \frac{u_0^2}{\gamma_0 R T_0/\overline{W}_0}, \tag{6.12}$$

with R being the universal gas constant. These equations are expanded with a small term $\epsilon_M = \gamma \mathrm{Ma}^2$, keeping the ϵ^{-1} and ϵ^0 order terms. The viscous heating term is proportional to ϵ and therefore dropped. The pressure term in the momentum equation contains an order ϵ^{-1} term in the expansion and requires special treatment. The pressure is split two components: a thermodynamic component and an order ϵ dynamic component,

$$P = p_{\mathrm{th}} + \epsilon p_{\mathrm{dyn}} \quad \text{and} \quad \frac{\partial p_{\mathrm{th}}}{\partial x_i} = 0. \tag{6.13}$$

With this splitting, the ϵ^{-1} term in the momentum equation vanishes and the equation set is written

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_j}{\partial x_j} = 0, \tag{6.14}$$

$$\frac{\partial(\rho u_j)}{\partial t} + \frac{\partial(\rho u_i u_j)}{\partial x_j} = -\frac{\partial p_{\mathrm{dyn}}}{\partial x_i} + \frac{1}{\mathrm{Re}}\frac{\partial \tau_{ij}}{\partial x_j} + \frac{\rho}{Fr_i}, \tag{6.15}$$

$$\frac{\partial \rho h}{\partial t} + \frac{\partial \rho u_j h}{\partial x_j} = \frac{\gamma - 1}{\gamma}\frac{\partial p_{\mathrm{th}}}{\partial t} - \frac{1}{\mathrm{Re}\,\mathrm{Pr}}\frac{\partial q_j}{\partial x_j}, \tag{6.16}$$

with the thermodynamic pressure being calculated from the ideal gas relationship,

$$p_{\mathrm{th}} = \frac{1}{\gamma \mathrm{Ma}^2}\frac{\rho T}{\overline{W}/\overline{W}_0}. \tag{6.17}$$

The heat flux $\mathbf{q}$ remains to be precisely defined. Nalu supports a Boltzmann radiative transport equation through which it can account for the radiative losses. However, for purposes of the flame hole dynamics modeling, the radiative contribution to $\mathbf{q}$ is ignored. We additionally assume Fourier law heat diffusion and equal binary diffusivities of the mass fractions. With these assumptions, the heat flux vector can be expressed as

$$\mathbf{q} = -\lambda \nabla T + \sum_i h_i (1 - \mathrm{Le}^{-1}) \nabla Y_i. \tag{6.18}$$

As discussed in Section 5.1, with the assumption of equal binary species diffusivities and that the Lewis numbers of all the components is unity, the heat flux vector, $\mathbf{q}$, can be further reduced. Given the application is an open jet flame, the pressure term in the enthalpy equation is small, and the enthalpy can be treated as a passive scalar. Many of these assumptions are not necessary for the flame hole dynamics model equation itself but are instead made in preparation to use the model to describe the highly turbulent Sandia "F" flame.

6.1.2 Time integration scheme

Nalu uses implicit time stepping schemes to integrate equations in time. The two time-stepping schemes that are incorporated into Nalu are the backward Euler and a second-order backwards difference formula method (BDF2). Both methods can be written as a sum of states between $n+1$ and $n-1$ as

$$\int_V \frac{\partial \rho \psi}{\partial t} dV = \int_V \frac{\alpha_1 \rho^{n+1} \psi^{n+1} + \alpha_0 \rho^n \psi^n + \alpha_{-1} \rho^n \psi^{n-1}}{\Delta t} dV. \tag{6.19}$$

The backward Euler has coefficients $\alpha_1 = 1,\ \alpha_0 = -1$, and $\alpha_{-1} = 0$ whereas the BDF2 method has coefficients $\alpha_1 = 1,\ \alpha_0 = -4/3$, and $\alpha_{-1} = 1/3$. Nalu uses a lumped mass scheme where the scalar and density are evaluated at the node and integrated over the dual volume. For the flame state equation Eq. (2.33), the time integration reduces further as

$$\int_V \frac{\partial \varphi}{\partial t} dV = \int_V \frac{\alpha_1 \varphi^{n+1} + \alpha_0 \varphi^n + \alpha_{-1} \varphi^{n-1}}{\Delta t} dV. \tag{6.20}$$

6.2 Implementation of the flame hole dynamics equation

The flame hole dynamics equation Eq. (2.33) is implemented into Nalu using the vertex-centered edge-based discretization. A CVFEM discretization is also possible but has not been implemented. For meshes with limited non-orthogonal faces but potentially high aspect ratios, the edge-based method is considered to be ideal over the CVFEM discretization [4]. We intend to use a hexahedral mesh with orthogonal faces in our simulation of the Sandia "F" flame.

As in the structured implementation, the advection term is split into a divergence-free and dilatational-like component as

$$\mathbf{u} \cdot \varphi = \nabla \cdot (\mathbf{u}\varphi) - (\nabla \cdot \mathbf{u})\varphi, \tag{6.21}$$

and the two terms are handled separately. The finite-volume discretization of the equation is, with Einstein summation (which is used throughout this chapter),

$$\int_V \frac{\partial \varphi}{\partial t} dV + \int_S \varphi u_j n_j dS = \int_V \left(\frac{\partial u_k}{\partial x_k} \varphi + V_e(\chi) \left(\frac{\partial \varphi}{\partial x_k} \frac{\partial \varphi}{\partial x_k} \right)^{1/2} - \dot{Q}(\varphi, \chi) \right) dV. \tag{6.22}$$

The term in divergence form is treated using a stabilized upwind advection algorithm blended with a higher, fourth-order method. The advection term is discretized as

$$\int (\rho u_j \varphi) n_j dS \approx \dot{v}_\alpha \varphi_\alpha, \tag{6.23}$$

where φ_α is the flame state at an integration point α (the midpoint of the edge in the edge-based scheme) and $\dot{v}$ is the advection velocity rate, which in the edge-based scheme is

$$\dot{v}_\alpha = \frac{1}{2} \left(u_j^L + u_j^R \right) A_j, \tag{6.24}$$

where A_j is j^{th} component of the area vector. The value of the scalar at the integration point is reconstructed through a blending of a higher-order upwind method and general unstabilized central interpolation, $\varphi_{g,0}$ (see Eq. (6.32)),

$$\varphi_\alpha = \eta \varphi_{\text{up}} + (1 - \eta) \varphi_{g,0}. \tag{6.25}$$

φ_0 is the central interpolation of φ at the integration point α; in the edge-based scheme, this is the average of the value at the "left" and "right" end-point nodes of the edge,

$$\varphi_0 = \frac{1}{2}\left(\varphi^L + \varphi^R\right). \tag{6.26}$$

The upwinding φ is decided based on the advection velocity rate,

$$\varphi_{\mathrm{up}} = \begin{cases} \omega_{\mathrm{up}}\varphi^L_{\mathrm{up}} + (1-\omega_{\mathrm{up}})\varphi_0 & \dot{v} > 0 \\ \omega_{\mathrm{up}}\varphi^R_{\mathrm{up}} + (1-\omega_{\mathrm{up}})\varphi_0 & \dot{v} < 0, \end{cases} \tag{6.27}$$

with the *left* and *right* upwind interpolated states are determined by the L and R edge nodes as

$$\varphi^L_{\mathrm{up}} = \varphi^L + d^L_j\left(\frac{\partial\varphi}{\partial x_j}\right)^L \tag{6.28}$$

$$\varphi^R_{\mathrm{up}} = \varphi^R - d^R_j\left(\frac{\partial\varphi}{\partial x_j}\right)^R, \tag{6.29}$$

where the distance vectors are defined as the distance from the left and right nodes to the integration point, $\mathbf{x}_\alpha$,

$$d^L_j = x_{\alpha,j} - x^L_j \tag{6.30}$$

$$d^R_j = x^R_j - x_{\alpha,j}, \tag{6.31}$$

and the partial derivative $(\partial\varphi/\partial x_j)^L$ is the projected nodal gradient at the "left" edge node (the "R" partial derivative term is defined similarly at "right" edge node). The general central difference operator is simply the average of the "left" and "right" extrapolated upwind states,

$$\varphi_{g,0} = \frac{1}{2}\left(\hat{\varphi}^L_{\mathrm{up}} + \hat{\varphi}^R_{\mathrm{up}},\right) \tag{6.32}$$

where the hat $(\hat{\cdot})$ in Eq. (6.32) is used to indicate that the upwind interpolated states used in the general unstabilized central interpolation optionally use a different upwind coefficient, $\omega_{g,0}$, than what is used to define the upwind term in Eq. (6.27),

$$\hat{\varphi}_{\mathrm{up}}^{L} = \omega_{g,0}\varphi_{\mathrm{up}}^{L} + (1-\omega_{g,0})\varphi_0 \tag{6.33}$$

$$\hat{\varphi}_{\mathrm{up}}^{R} = \omega_{g,0}\varphi_{\mathrm{up}}^{R} + (1-\omega_{g,0})\varphi_0. \tag{6.34}$$

$\omega_{g,0}$ is specified in the input and determines the degree of blending between the central discretization and the higher-order upwind extrapolation. The order of accuracy of the discretization, asymptotically, is second order due to the presence of a second-order discretization in the blending. At the finite resolutions typically used to perform simulations, however, some higher-order behavior is expected and observed due to the blending of a higher-order method into the discretization. The blending coefficient η is considered to be a specified constant for the implementation of Eq. (2.33) into Nalu, but for other algorithms, a Peclet blending is used [4].

The right-hand size of Eq. (6.22) is evaluated at the nodes using projected nodal gradients to express the derivative terms. The volume is evaluated over the dual volume as a nodal source term in the algorithm. The algorithm is integrated in time using either a backwards Euler or BDF2 time integration described in Section 6.1.2.

In principle, the flame hole dynamics master equation Eq. (2.33) needs only to be evaluated on the band of surface points (see Section 6.4 for the definition in an unstructured context). However, given that Nalu uses implicit time-stepping, the matrix would need to be recreated every time-step and the pre-conditioning algorithm reapplied. It is not clear that there is any benefit to restricting the domain of the flame hole dynamics equation in Nalu. For this reason, the master equation is solved on the entire domain. The closest point extension (Section 3.1.1) is only evaluated at nodes in the surface band, however, and only points in the surface band are used as a boundary condition for the anistropic diffusion extension (Section 6.6).

6.3 Closest point extension in Nalu

The closest point extension was implemented into Nalu using the surface identification algorithm (Section 6.4), the nearest node search algorithm (Section 6.5), and the local least squares approximation described in Section 3.1.1.

Within the set of surface points constructed through the surface identification algorithm (analogous to the band of surface points in the structured algorithm), each node calculates a minimum distance vector to the stoichiometric surface using the local least-squares approximation, Eq. (3.3). At each computational node, a local least-squares cell is created using the upward connectivity relations in the sierra toolkit's mesh [70]. For a hexahedral mesh, this results in a 27×10 linear system at nodes in the interior of the domain. Near the boundaries, the number of points in the local least squares cell can be less than 27. However, as one individual element contains 8 computational nodes, the number of points in the local least squares cell will always be greater than 10 except in the trivial case where an element is completely disconnected from all other elements. The linear system is solved using LAPACK's DGELS least squares solver. The resulting least squares coefficients are then used to calculated the coefficients of the polynomial in Eq. (3.8). The polynomial root finding problem is solved either by using LAPACK's DGEEV algorithm applied to the polynomial's companion matrix (Eq. (3.11)) or through a Newton iteration (Eq. (3.10)). As the solution of the eigensystem is relatively slow, a hybrid method where the Newton iteration is allowed to compute for a limited number of iterations (e.g. 5) to test for rapid convergence before switching to the slower and more robust companion matrix formulation is also implemented. For all of the tests of the algorithm, the companion matrix formulation is used in order to select the appropriate polynomial root through the process described in Section 3.1.1.

After the closest point is calculated, the nearest node search algorithm is used to determine the nearest node to each closest point vector. Each node is mapped to its corresponding nearest node, and a local least squares cell is formed using the upward node-to-element relationships in the mesh framework. The least squares system is solved (again, using LAPACK's DGELS algorithm) and Eq. (3.13) is used to inject the value into node that is being processed.

Figure 6.2 shows the relative closest point vectors ($\mathbf{cp}(\mathbf{x}_j) - \mathbf{x}_j$, with tails centered at the node) for a complicated stoichiometric surface generated by artificially perturbing a jet flow. The mesh and domain used in this calculation is discussed in Section 6.10.

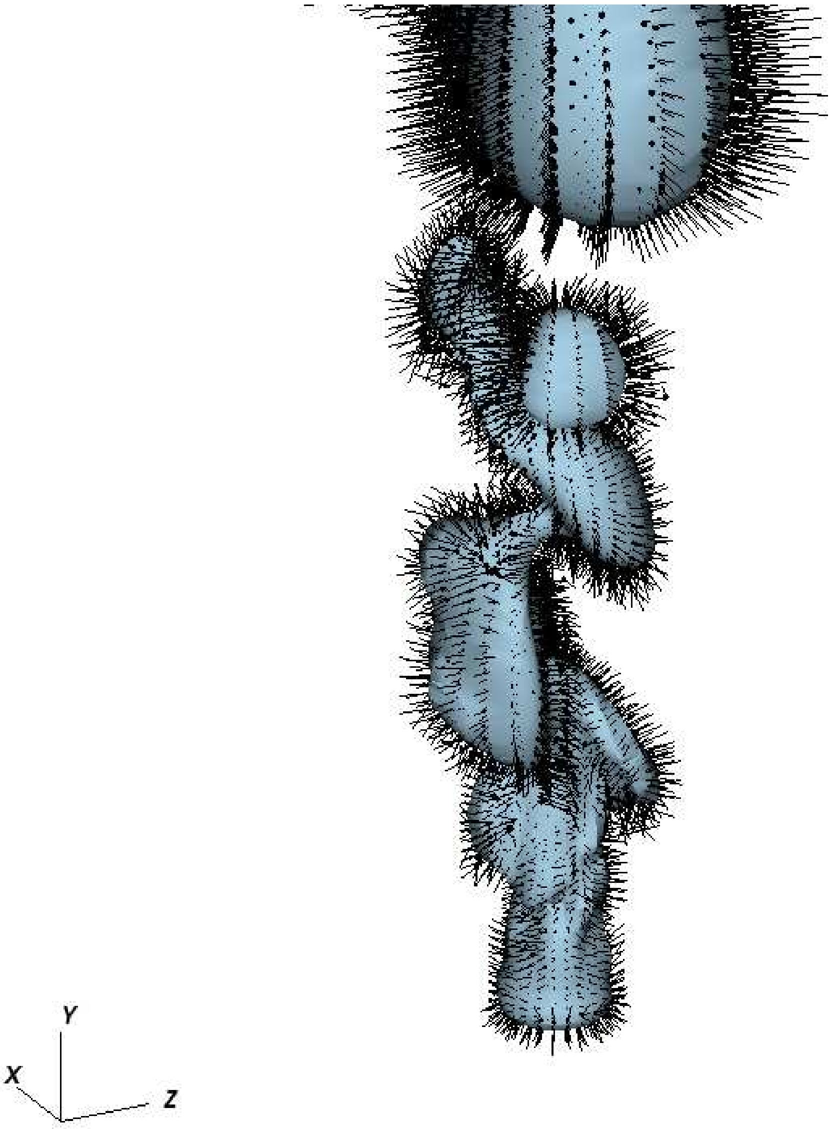


Figure 6.2: Closest point vectors (black) with a distorted stoichiometric surface (light blue) originating from a perturbed jet simulation with 8 processors on a 220k node mesh at with $U_{c,o,\text{test}} = 1/100 U_{c,o} = 1.267$ m s^{-1}.

6.4 Surface identification

The identification of the surface is done in multiple stages. In the first stage, the nodes of each control volume surface are iterated over to determine whether a crossing of the stoichiometric mixture fraction has occurred according to

$$(Z_L - Z_{\mathrm{st}})(Z_R - Z_{\mathrm{st}}) \leq 0, \tag{6.35}$$

where Z_L and Z_R are the “left” and “right” node pairs of an element edge. Nodes satisfying this condition (or nodes that have a value exactly equal to the stoichiometric value) are herein referred to as “surface nodes”. A masking field marks all of the surface nodes with an integer value (e.g.“1”). Then a communication stage occurs where the masking field that marks the surface nodes is communicated to all processors. All nodes in all elements containing a surface node are then also labeled (“2”) by use of the upward connectivity relationships in the mesh toolkit used by Nalu (Sierra toolkit mesh [70]). Another communication stage occurs and this last process is repeated once more to label all elements that are used in the discretization of nodes that are in the local least-squares cell of the surface nodes. The element bandwidth is chosen such that surface nodes are not influenced by points outside of the band given the recentering of the local least squares cell associated with the interpolation of φ at $\mathbf{cp}(\mathbf{x}_j)$ (see Section 6.5).

The masking field is also used for several consistency checks in the method. If the closest point transform does not identify a node in the band of surface nodes, then the value of the masking field changes to indicate that the point failed to find a closest point. The anisotropic diffusion-type extension, discussed in Section 3.1.3, is used instead of the closest point transform for such a point. Additionally, if the distance indicated by the closest point vector $|\mathbf{cp}(\mathbf{x}_j) - \mathbf{x}_j|$ is too large; that is, it is larger than

$$|\mathbf{cp}(\mathbf{x}_j) - \mathbf{x}_j| > \sum_{i=0}^{N_{\mathrm{band}}} (1 + \delta_e)^k V_j^{1/3}, \tag{6.36}$$

where N_{band} is the number of elements the node is from the surface (“1”, “2”, or “3”), V_j is the dual nodal volume, and δ_e is an expansion factor added to account for variation in element size. If the point fails this check, its extended value is computed by the anisotropic diffusion extension

(Section 6.6).

6.5 Nearest node search

Once the closest point on the surface, $\mathbf{cp}(\mathbf{x}_j)$, is obtained for a node j, the closest point extension centers the local least-squares cell on the node nearest to $\mathbf{cp}(\mathbf{x}_j)$. In the structured version of the algorithm, the closest cell center can be calculated directly; however, with a generalized unstructured mesh, this is no longer possible. Instead, a search for the nearest node is performed.

The search is performed using a coarse, parallel bounding box search available in the Sierra Toolkit Mesh (STK mesh). The coarse search algorithm maps domain bounding boxes to range bounding boxes, if the domain and range bounding boxes intersect. The range bounding boxes are potentially not on the processor owning the domain bounding box, and parallel communication is done to ensure that the search returns all possible intersections using the Boost library's "rtree" search algorithm. Three types of bounding box structures are possible: a "point", "box" and "sphere" bounding box. The nearest node search implemented into Nalu uses the "point" and "sphere" structures.

"Point" bounding boxes are created at the domain's "surface nodes" (Section 6.4) and are specified by the node's global coordinates. A "sphere" structure bounding box is used to describe location of the closest point vectors on the mesh. For any node j within the entire surface band, "sphere" bounding boxes are centered at $\mathbf{cp}(\mathbf{x}_j)$ with a radius proportional to the dual nodal volume, $r_{\text{sphere}} = (1 + \delta_{\text{nns}})V_j^{1/3}$, are created. The expansion factor, δ_{nns}, is included to account for potential stretching of the mesh. The coarse search algorithm returns a vector containing paired identifiers for domain nodes and range nodes ("candidate nearest nodes") as well as their processor ranks. The global indices of the candidate nearest nodes as well as their processor ranks are used to modify the parallel ghosting aura of the mesh. For a particular domain node, all nodes that share an upward relationship (i.e. nodes that are within the same element) with range nodes are added to the ghosting aura of the domain node's process. After the ghosting has been modified, field data is communicated between processors so that a fine search for the nearest node can be executed by computing the distance of each candidate node to the closest point vector.

The result is a map between a domain node j in the surface band and a node k in the global set of surface points, such that $|\mathbf{cp}(\mathbf{x}_j) - \mathbf{x}_k|$ is minimized. The map structure is iterated over in order to form the local least-squares cell used to compute the injection $\varphi(\mathbf{cp}(\mathbf{x}_j))$, Eq. (3.13).

6.6 Discretization of the anisotropic diffusion extension operator

A vertex-centered, edge-based discretization was implemented for the anisotropic diffusion extension (Section 3.1.3). The gradient in the anisotropic diffusion operator is discretized by edge using a projected nodal gradient with an over-relaxed correction [72] applied to account for skewed elements,

$$G_{\mathrm{orth},j}\varphi \approx \overline{G_j\varphi} + [(\varphi_R - \varphi_L) - \overline{G_\ell\varphi} dx_\ell]\frac{A_j}{A_k dx_k}, \tag{6.37}$$

where $\overline{G_j\varphi}$ is the average of the projected nodal gradient of φ between the left and right nodes of the edge. The isotropic part of the diffusion equation, Eq. (3.31), is computed as

$$\int_S \epsilon_D D_{\mathrm{mix}}\frac{\partial\varphi}{\partial x_i} n_i dS \approx \epsilon_D \overline{D_{\mathrm{mix}}}(G_{\mathrm{orth},i}\varphi)A_i, \tag{6.38}$$

where $\overline{D_{\mathrm{mix}}}$ denotes the diffusivity coefficient averaged between the left and right nodes of the edge. The discretization of the anisotropic part uses the normal of the mixture fraction $\mathbf{n}_Z$ and is computed as

$$\int_S D_{\mathrm{mix}}\left(n_{Z,i}n_{Z,j}\frac{\partial\varphi}{\partial x_j}\right) n_i dS \approx \overline{D_{\mathrm{mix}}}\frac{(G_iZ)A_i}{(G_kZ)(G_kZ)}(G_{\mathrm{orth},j}\varphi)G_jZ. \tag{6.39}$$

Edges are iterated over and the resulting discretized system is solved using either the backward Euler or BDF2 implicit time integration schemes.

6.7 Internal Dirichlet condition

The internal Dirichlet condition is implemented using Trilinos's “TPetra” linear solver library [70]. For each entry of the nearest surface band (labeled “1”, as discussed in Section 6.4) that is locally owned by the computational process, the corresponding row of the anisotropic diffusion (see Section 6.6) matrix is set to zero except for the diagonal component, which is set to 1. The right-hand side vector entry corresponding to that row is modified to be the value of the flame state at the corresponding node. This method enforces the internal Dirichlet condition without requiring a restructuring of the matrix overall.

6.8 Anti-dissipative correction in Nalu

The anti-dissipative correction (Section 3.2) is implemented into Nalu as a subiteration using an an explicit first-order Euler time integration for the pseudo-time variable τ,

$$\int_V \frac{\partial \varphi}{\partial \tau} dV = \int_V (\mathcal{U}_0 \epsilon_c \mathbf{n}_\varphi \cdot \nabla K) dV \tag{6.40}$$

where

$$K = |\nabla \varphi| - \frac{\varphi(1-\varphi)}{\epsilon_c}. \tag{6.41}$$

The algorithm is split into several steps. The auxiliary variable, ψ, Eq. (3.36), is calculated at each node. The algorithm then computes the projected nodal gradient of ψ at each node and uses it to calculate the term K of the right-hand side of the anti-dissipative correction given using Eq. (3.37). The projected nodal gradient of K is then calculated and the right-hand side can be fully formed,

$$\frac{\partial \varphi}{\partial \tau} \approx \left(\frac{(G_j \psi)(G_j K)}{\sqrt{(G_k \psi)(G_k \psi)}} \right), \tag{6.42}$$

with the time derivative being discretized with a forward Euler discretization as in Eq. (4.33). The pseudo-time step, $\Delta\tau$, is set by specifying the subiteration CFL number as in Eq. (4.34). ϵ_c and $\mathcal{U}_0$ are chosen for each particular simulation as an input. In our simulations, ϵ_c is chosen to be proportional to the size of the smallest mesh elements and $\mathcal{U}_0$ is set to a constant reference value (e.g. the inlet velocity of a jet). The subiteration is performed for a fixed number of iterations (typically 2), with a communication stage occurring after the computation of each subiteration. As the compression subiteration is explicit, the domain over which it is applied is restricted to the surface band for computational efficiency.

6.9 Entrainment

For open jet flows, the momentum of the jet entrains fluid from the surrounding air. While the momentum of the jet is conserved in a cold jet, the mass flux increases due to this entrainment. For a numerical simulation, the extent of the domain is finite, and the entrainment of the jet at the boundaries of the domain needs to be accounted for numerically. Our jet simulation uses two boundary conditions at the "open" boundaries of the domain (a boundary that is not an inflow nor a wall, essentially).

6.9.1 Nalu's open boundary condition

Nalu's open boundary condition allows the specification of the normal stress at the domain's open boundaries. It is imposed by iterating over the faces of the mesh matching the open boundary condition. The nearest node to the boundary integration point is saved and used in coordination with the opposing node to compute the gradients for required derivative terms at the boundaries. For the momentum equation, the tangential force to the boundary is expressed as

$$F_i^T = F_i(1 - n_i n_i) - \sum_{i \neq j} F_j n_i n_j \tag{6.43}$$

and the normal stress contribution is specified directly as

$$F_i^N = p^{\text{specified}} A_i. \tag{6.44}$$

In our simulation, $p^{\text{specified}}$, is set to be the ambient pressure. This boundary condition is used for the "top" boundary of the simulation (the boundary that is normal to the major flow direction of the jet).

At the side boundary, however, a special entrainment condition that is tuned to the conditions of the Sandia piloted jet flame "F" is used instead.

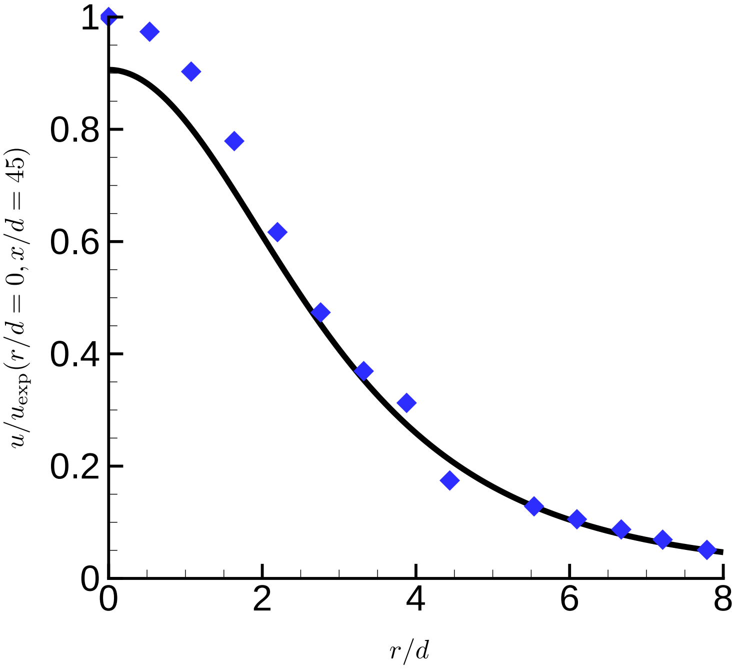


Figure 6.3: Comparison of experimental axial velocity with model at $x/d = 45$. The Batchelor-type model is represented by the black line while the experimental data is given by blue diamonds. The velocity is normalized by the experimental centerline velocity of main jet at $x/d = 45$, $u_{\exp}(r = 0) = 53.98\ \mathrm{m\ s^{-1}}$.

6.9.2 Entrainment boundary condition based on Batchelor's jet solution

Another possibility for the entrainment is to set the values according to theoretical jet solutions. The similarity solution of an open, turbulent jet (e.g. Pope [73] is one possibility), but it requires the definition of a "virtual jet origin". In contrast, the jet solution of Batchelor, where the jet is modeled as a momentum source at the jet exit, does not require the definition of a virtual origin. For the entrainment condition for a fully developed jet, we chose to implement a Dirichlet condition on the velocity based on Batchelor's jet solution.

Batchelor's jet solution, given in Eq. (6.50), has two free parameters: ν_T and c, a parameterization of the jet's momentum. The value of ν_T is estimated using values of the half-radius that are obtained from the data published in Schneider et al. [74]. From Pope [73], the jet is expected to be self-similar after around $30d$ (the inner diameter of the main jet, d, is 7.2[mm] from Barlow and Frank [1]) with S obtaining a constant value. The value of spreading rate was obtained by finding the average slope of $r_{1/2}$ using the three data points with $x \geq 30d$ available in Schneider et al. [74]. The spreading rate is calculated directly by its definition in Pope [73] (Eqn. 5.7):

$$S = \frac{dr_{1/2}(x)}{dx} = 0.05738. \tag{6.45}$$

From the spreading rate, the turbulent viscosity is obtained through equation 5.84 in Pope [73]:

$$\hat{\nu}_T = \frac{S}{8(\sqrt{2}-1)} = 0.01732 \tag{6.46}$$

and by equation 5.85 in Pope [73],

$$\nu_T = \hat{v}_T U_0(x = 45d) r_{1/2}(x = 45d) = 0.01799 \text{ m}^2 \text{ s}^{-1}, \tag{6.47}$$

where the jet velocity $U_0(x = 45d)$ is obtained from Schneider et al. [74] as well. The value of $x = 45d$ is chosen because the velocity profile in Schneider et al. [74] is measured at that value of x. The momentum flux is calculated using the axial component of the velocity, as in Pope [73] (eqtn

5.64) at the inlet of the jet (denoted with a J subscript),

$$\frac{F_P}{2\pi\bar{\rho}\nu_T^2} = \frac{1}{\nu_T^2}\int_0^{R_{\max}} \rho_J \langle U_J\rangle^2 r dr = 1207.7, \tag{6.48}$$

where $\bar{\rho}$ is the average density at the plane of comparison, computed as

$$\bar{\rho} = \frac{1}{\pi r_{1/2}^2}\int_0^{r_{1/2}} 2\pi\rho r dr, \tag{6.49}$$

and ρ_J is the average density of the jet inlet flow (1/4 methane and 3/4 air at temperature of $294K$). The velocity profile is taken from the experimental results of Schneider et al. [74]. This is compared with the Batchelor model,

$$\psi = r_s \nu_T f(\theta) \tag{6.50}$$

$$f(\theta) = \frac{2(1-\cos^2\theta)}{1+c-\cos\theta} \tag{6.51}$$

$$u_{r_s} = \frac{1}{r_s^2\sin\theta}\frac{\partial\psi}{\partial\theta} \tag{6.52}$$

$$u_\theta = \frac{1}{r_s\sin\theta}\frac{\partial\psi}{\partial r_s}, \tag{6.53}$$

with r_s being the radial coordinate in the spherical coordinate system (r_s, θ, ϕ). The x-directed velocity is

$$U_b = u_{r_s}\cos\theta - u_\theta\sin\theta \tag{6.54}$$

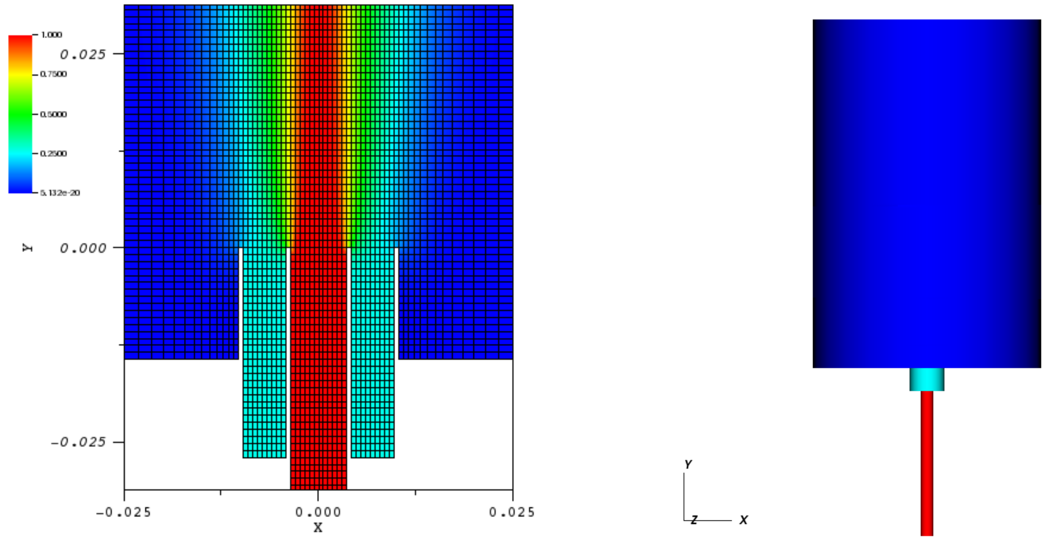
The axial velocity is used for comparison with the experimental data by evaluating the x-directed velocity at $x = 45d$, ($r_s = \sqrt{x^2+r^2}$ and $\cos\theta = x/\sqrt{x^2+r^2}$). The flux integral, from Batchelor, is

$$F_B = 2\pi\bar{\rho}\int_0^{\pi} u_{r_s}(u_{r_s}\cos\theta - u_\theta\sin\theta) - (\sigma_{r_s r_s}\cos\theta - \sigma_{r_s\theta}\sin\theta) r_s^2 \sin\theta d\theta \tag{6.55}$$

which can be integrated analytically, resulting in

$$\frac{F_B}{2\pi\bar{\rho}\nu_T^2} = \frac{32}{3}\frac{1+c}{c(2+c)} + 4(1+c)^2\log\left(\frac{c}{2+c}\right) + 8(1+c). \tag{6.56}$$

The value of c such that $F_B = F_P$ is then obtained numerically through Brent's method for a value



(a) Mixture fraction and mesh

(b) Computational domain

Figure 6.4: Domain and mixture fraction field (overlayed with the mesh) for the low resolution mesh. Note: for the conditions of the Sandia "F" flame, the required extent of the domain is estimated to be much larger (about 100 jet diameters in length, about 30 jet diameters in span).

of $c = 0.004543$. U_B with this value of c is plotted against the experimental data for comparison.

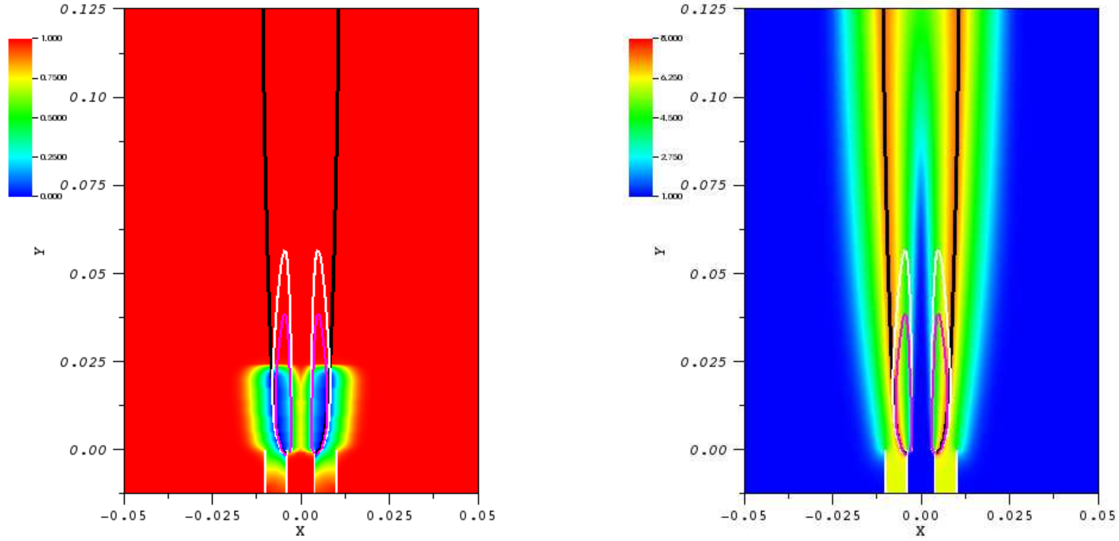
6.10 Uncoupled flame hole dynamics tests in a piloted jet geometry

Running a realistic flame hole dynamics model with realistic parameters for the Sandia "F" flame requires a large simulation run at a high Reynolds number,

$$\mathrm{Re}_{\mathrm{jet,F}} = \frac{U_{0,\mathrm{bulk}} d_{\mathrm{jet}}}{\nu} \approx 45,000. \qquad (6.57)$$

Working towards the Sandia "F" flame, a low resolution mesh with reduced domain extents (about 25 jet diameters in length and 10 jet diameters in span—the Sandia "F" flame has a visible length of around 70 jet diameters) was created and uncoupled flow simulations were run in parallel with up to 16 processors. The domain and mesh (as well as the mixture fraction) of a test simulation is shown in Figure 6.4. The test was run at a much smaller jet Reynolds number, $Re_{\mathrm{jet,test}} \approx 450$, with the main jet and pilot velocities scaled to 1/100-th of that of the Sandia “F” flame; that is, the main jet bulk velocity was set to 0.992 m s^{-1} with a pilot jet bulk velocity of 0.228 m s^{-1} for the purposes of this test.

A one mixture fraction Burke-Schumann approximation was used to describe the chemistry with the parameters of the pilot adjusted to coincide exactly with the Burke-Schumann approximation at $\hat{Z} = 0.27$. Otherwise, the composition of the jet and oxidizer streams were set to be the same as in the Sandia “F” flame (see Table 5.1 and Table 5.2). In the coupled simulation, this would not be sufficient as we also require the mixing solution of the pilot and jet. In this configuration, the crossover scalar dissipation was chosen to be $\chi_0 = 0.75\chi_q$ and χ_q was chosen such that there is partial extinction at the base of jet by setting $\chi_q = 4.0\ \mathrm{s}^{-1}$. The laminar flame speed was set to $S_L^0 = 2.0\,\mathrm{m\,s}^{-1}$ and the quenching time was set to be 4 times the constant timestep of 0.0001 seconds. The anti-dissipative correction parameters were set to $\epsilon_c = 0.001$ (similar to the element size in the main pipe of the jet), $\mathcal{U}_0 = 1.0\,\mathrm{m\,s}^{-1}$ (similar to the jet bulk velocity), and the sub-CFL was set to 0.0001. Two compression iterations were performed each time step. The entrainment condition was set with the parameters described in Section 6.9.2, with $c = 0.004543$ and $\nu_T = 0.0001799\ \mathrm{m}^2\ \mathrm{s}^{-1}$ consistent with the reduction of the main jet velocity by a factor of 1/100; see Eq. (6.47). Figure 6.5 shows the temperature and extended flame state contours through a slice through the center of the computational domain. Figure 6.6 shows the stoichiometric surface colored by the flame state variable. The flame hole dynamics algorithm in this test case mimics a lifted jet diffusion flame, with the flame “quenched” near the burner (though, in this test, the flame state variable has no effect on the temperature or dynamics of the flow). The quenched region extends slightly beyond the crossover scalar dissipation. The equilibrium height of the “quenched” region in the simulation is at the point where the flame hole boundary propagation is counter balanced by the flow velocity (that is, $|d\mathbf{r}_i/dt| = \mathbf{u}\cdot\mathbf{m} + V_e = 0$; see Eq. (2.6)). As the laminar flame speed is roughly twice that of the main jet speed in the simulation, the equilibrium point is only slightly beyond the crossover dissipation.



(a) Extended flame state

(b) Temperature

Figure 6.5: Extended flame state and temperature (using a Burke-Schumann model on a coarse (around 220k node) mesh of a piloted jet flame at $U_{c,o,\text{test}} = 1/100 U_{c,o} = 1.267$ m s^{-1}

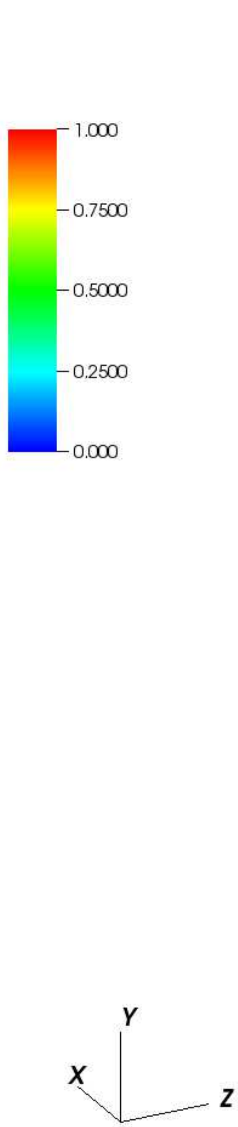


Figure 6.6: flame state on the stoichiometric surface for a coarse (220k node) mesh of the piloted jet flame at $U_{c,o,\text{test}} = 1/100U_{c,o} = 1.267\ \mathrm{m\,s^{-1}}$ The magenta line is a contour of the quenching dissipation χ_q and the white line is a contour of the critical dissipation χ_0.

Chapter 7

Conclusions

The thesis describes the development of a model for extinction and reignition in turbulent diffusion flames and a new numerical approach for the resulting equation. The main challenge is the need to solve an evolution equation, of the flame state field, defined on a moving complex surface. Here, a Cartesian embedding technique is preferred because it renders the algorithm very efficient, especially when the underlying grid is structured.

Because the flame state field does not represent a material property, it is a dynamical description of the flame burning conditions, the advection part of the governing equation is not in conservation form. This necessitates a specialized treatment where advection is split into divergence- and dilatational-like terms. In the structured implementation, the divergence-like term is discretized using an existing high-order finite-volume WENO method while the dilatational-like term uses a new high-order Gaussian quadrature method developed specifically for this application, which is made consistent with the WENO method at the level of the reconstruction of the field. The choice of high-order method is dictated by the need to minimize anisotropic errors tied to the logically Cartesian mesh, since flame hole dynamics is a highly geometrical process. Another key component of the algorithm is an extension procedure whereby the flame state field that is nominally defined around the stoichiometric surface is extended to cover the whole three-dimensional domain. This allows the determination of the gas mixture composition anywhere in space. The extension algorithm is formulated by determining the stationary solution of a propagation equation that is solved using a multigrid technique. Furthermore, we demonstrate the combined integration of the different mechanisms present in flame hole dynamics (advection, flame boundary propagation, and flame quenching) and their successful numerical coupling. The method is shown to be accurate on several test geometries and was employed to solve for the evolution of flame holes on a realistic and complex turbulent stoichiometric surface taken from a DNS of a shear layer with a mesh involving millions of grid points. The structured implementation is efficient and high-order accurate.

The method is also implemented into the unstructured fluid dynamics solver “Nalu”. In this framework, the model can be used in parallel simulations using complex geometry. Development is made toward applying the flame hole dynamics model to massively parallel simulation of a highly turbulent, piloted jet flame, Sandia’s “F” flame. A two mixture fraction approach is adopted to define a flamelet reacting flow formulation as well as a Burke-Schumann approximation in order to properly represent the main jet and pilot of the burner. Additionally, an entrainment condition is developed using Batchelor’s analytical jet solution with parameters tuned to the Sandia “F” flame. Preliminary tests have been conducted on a pilot jet flame geometry and the implementation into Nalu is shown to produce results mimicking a lifted jet flame.

A model for flame hole dynamics has been developed, and a numerical framework for solving the resulting equation has been presented. The method has been implemented in both a structured and unstructured setting.

Appendix A

Constitutive models

A.1 Edge flame velocity model

The edge flame velocity depends primarily on the rate of strain of the flow (here parameterized by the mixture fraction rate of dissipation) and the cold-to-hot mixture density ratio. It is customary to normalize the edge flame velocity by S_L^0, the laminar premixed speed of a mixture at the stoichiometric conditions for the mixture just ahead of the edge flame [16]. Owing to this dependence on strain rate, the main diffusion flame can undergo reignition by the healing of the flame hole boundary through a positive edge-flame velocity (and correspondingly a low scalar dissipation) or further extinction by growth of the flame hole, when the edge-flame velocity is negative due to large local values of scalar dissipation. It is also possible, but not considered here, for a quenched zone to become reactive due to the folding of burning regions near non-burning regions of the stoichiometric surface [6, 75].

The edge flame velocity is provided through a model as suggested in Pantano and Pullin [19]. To date, there are limited theoretical studies providing precise calculations of the edge flame velocity as a function of the relevant local flow conditions and mixture conditions for real chemistry and realistic levels of heat release, (most studies concentrate on the thermo-diffusive limit where density is constant [76]). This is a consequence of the complicated and difficult-to-solve nature of the two-dimensional fully coupled reactive Navier-Stokes boundary value problem. Despite these limitations, and only for the purpose of a formulation of a flame hole dynamics framework, one can approximate the edge flame velocity, according to

$$\frac{V_e}{S_L^0} = 1 - \frac{1 + \frac{1}{\frac{\chi}{\chi_\mathrm{q}} - 1}}{1 + \frac{1}{\frac{\chi_0}{\chi_\mathrm{q}} - 1}}, \tag{A.1}$$

where χ_0 is the value of the scalar dissipation for which the edge speed is zero.

A.2 Flame extinction model

Flame hole creation will be modeled (to leading order) as an instantaneous process that pokes a hole in the flame at every point in Σ_q. Computationally, it is beneficial to weaken this very fast process by an extinction process that takes place over a finite amount of time, τ_{q}, as explained in Hewson [33]. If we impose that complete extinction is reached at $t = \tau_{\mathrm{q}}$ (starting from $t = 0$ when condition Σ_q is attained) following the time-dependent law, $\varphi(t) = (1 - t/\tau_{\mathrm{q}})^2$, then one can determine the form of the sink, $\dot{Q}(\varphi, \chi)$, to be

$$\dot{Q}(\varphi, \chi) = R(\chi - \chi_{\mathrm{q}}) \frac{2\sqrt{\varphi}}{\tau_{\mathrm{q}}}, \tag{A.2}$$

where $R(\chi - \chi_{\mathrm{q}})$ is a ramp function,

$$R(\chi - \chi_{\mathrm{q}}) = \begin{cases} 0 & \chi < \chi_{\mathrm{q}}, \\ \frac{\chi - \chi_{\mathrm{q}}}{\Delta_\chi} & \chi_{\mathrm{q}} \leq \chi < \chi_{\mathrm{q}} + \Delta_\chi, \\ 1 & \chi \geq \chi_{\mathrm{q}} + \Delta_\chi, \end{cases} \tag{A.3}$$

and Δ_χ is a parameter set to be small relative to χ_{q}. This model can be enhanced, and made more accurate, by replacing $R(\chi - \chi_{\mathrm{q}})$ with the scalar dissipation impulse model of Hewson [33].

Appendix B

Discretization details

B.1 MUSTA

The multistage upwind method (called MUSTA, for "**mu**lti **sta**ge" [53]) is an iterative approximate Riemann solver used in the conservative part of the finite-volume technique. It solves the Riemann problem between cell interfaces on a local, coarse space-time mesh $(d - \tau)$ using the "**F**irst-**OR**der **CE**ntered (FORCE) flux". The FORCE flux is an average of the Lax-Friedrichs flux,

$$F_{1/2}^{LF}(\varphi^L, \varphi^R) = \frac{1}{2}[F(\varphi^L) + F(\varphi^R)] - \frac{1}{2}\frac{\Delta d}{\Delta \tau}[\varphi^R - \varphi^L], \tag{B.1}$$

and the two-stage Lax-Wendroff flux,

$$F_{1/2}^{LW}(\varphi^L, \varphi^R) = \varphi_{1/2} = \frac{1}{2}[\varphi^L + \varphi^R] - \frac{1}{2}\frac{\Delta \tau}{\Delta d}[F(\varphi^L) - F(\varphi^R)]. \tag{B.2}$$

The method allows for the use of a local Courant number, based on the local speed in the direction normal to the face. The local timestep is

$$\Delta \tau = \text{CFL}\frac{\Delta d}{S}, \tag{B.3}$$

where S is the local wavespeed decided in our case by the velocity u as $|(max(u^L, u^R))|$. Transmissive boundary conditions are used in the local mesh. For instance, with two cells, the transmissive boundary is

$$F_{-1/2} = \varphi^L \qquad F_{3/2} = \varphi^R. \tag{B.4}$$

and the intercell flux is

$$F_{1/2} = \frac{1}{2}F^{LF}_{1/2}(\varphi^L, \varphi^R, \Delta\tau, \Delta d) + \frac{1}{2}F^{LW}_{1/2}(\varphi^L, \varphi^R, \Delta\tau, \Delta d). \tag{B.5}$$

As reported by Titarev and Toro [51], two cells produce accuracy similar to the HLL (Harten, Lax, van Leer) flux in the case of the Euler equations. Greater detail into this method is provided in a series of papers [53, 54, 55].

B.2 Weights and abscissae of the modified Gaussian quadrature

The nonconservative weights and abscissae were calculated for a two-point ($m = 2$) and three-point ($m = 3$) quadrature of φ (three and four-point quadrature of the velocity, correspondingly). For $m = 2$, the weights are

$$W = \begin{pmatrix} -\frac{1}{2} - \frac{1}{\sqrt{3}} & \frac{2}{\sqrt{3}} & \frac{1}{2} - \frac{1}{\sqrt{3}} \\ -\frac{1}{2} + \frac{1}{\sqrt{3}} & -\frac{2}{\sqrt{3}} & \frac{1}{2} + \frac{1}{\sqrt{3}} \end{pmatrix}$$

and the abscissae are $x_\alpha = \{-1/\sqrt{3}, 1/\sqrt{3}\}$ for the scalar and $x_b = \{-1, 0, 1\}$ for the velocity. For $m = 3$,

$$\begin{aligned}
W_{11} &= -\frac{5\left(1849 + \sqrt{43\left(69437 - 1520\sqrt{15}\right)}\right)}{14706}, & W_{12} &= \frac{5\left(1849 + \sqrt{43\left(69437 + 1520\sqrt{15}\right)}\right)}{14706} \\
W_{13} &= \frac{5\left(-1849 + \sqrt{43\left(69437 - 1520\sqrt{15}\right)}\right)}{14706}, & W_{14} &= -\frac{5\left(-1849 + \sqrt{43\left(69437 + 1520\sqrt{15}\right)}\right)}{14706} \\
W_{21} &= -\frac{4\left(67\sqrt{215} - 1247\right)}{7353}, & W_{22} &= -\frac{4\left(1247 + 67\sqrt{215}\right)}{7353} \\
W_{23} &= \frac{4\left(1247 + 67\sqrt{215}\right)}{7353} & W_{24} &= \frac{4\left(67\sqrt{215} - 1247\right)}{7353} \\
W_{31} &= \frac{5\left(-1849 + \sqrt{43\left(69437 + 1520\sqrt{15}\right)}\right)}{14706} & W_{32} &= -\frac{5\left(-1849 + \sqrt{43\left(69437 - 1520\sqrt{15}\right)}\right)}{14706} \\
W_{33} &= -\frac{5\left(-1849 + \sqrt{43\left(69437 - 1520\sqrt{15}\right)}\right)}{14706} & W_{34} &= \frac{5\left(1849 + \sqrt{43\left(69437 - 1520\sqrt{15}\right)}\right)}{14706}
\end{aligned}$$

with abscissae

$$\begin{aligned} x_\alpha &= \left\{-\sqrt{\frac{3}{5}}, 0, \sqrt{\frac{3}{5}}\right\} \quad \text{and} \\ x_\beta &= \left\{-\frac{1}{20}\left(5+\sqrt{215}\right), -\frac{1}{20}\left(-5+\sqrt{215}\right), \frac{1}{20}\left(-5+\sqrt{215}\right), \frac{1}{20}\left(5+\sqrt{215}\right)\right\}. \end{aligned} \tag{B.6}$$

B.3 WENO interpolation and reconstruction

A combined WENO interpolation and reconstruction is used to determine the flux across each face of the conservative term of the governing equation [77]. The interpolation and reconstruction steps are dimensionally-split, using a one-dimensional interpolation in each cardinal direction to evaluate the variable of interest at the specified Gaussian quadrature collocation point. The conservative method requires the evaluation of the scalar field, φ, and the velocity, $\mathbf{u}$, at Gauss points located on the faces of the finite-volume cell. The nonconservative method requires the evaluation of $\mathbf{u}$ and φ at separate points prescribed by a modified Gaussian quadrature discussed in Section 4.2. The scalar field is volume-averaged as part of the finite-volume procedure and requires WENO reconstruction whereas the velocity field is assumed to be given as pointwise values from some exterior flow solver and requires WENO interpolation. The derivative of the pointwise values is also constructed for use in evaluating the nonlinear FHB propagation speed term in the governing equation.

B.3.1 WENO interpolation

A 5th-order WENO interpolation is developed by defining an interpolating polynomial of degree 4 through a centered stencil of five points, $S = \{u_{i-2}, \ldots, u_{i+2}\}$. The stencil is broken into three substencils: $S_{-1} = \{u_{i-2}, u_{i-1}, u_i\}$, $S_0 = \{u_{i-1}, u_i, u_{i+1}\}$, and $S_{+1} = \{u_i, u_{i+1}, u_{i+2}\}$. Interpolating polynomials are constructed for each of these stencils. Weights are chosen such that the value of subpolynomials matches the wide polynomial at a point ξ. These linear weights are then modified such that a greater importance is given on subpolynomials with lower total variation, providing robustness for problems containing sharp gradients. The subpolynomials are

$$\begin{aligned}
p_{-1} &= u_{i-2}\left[\frac{1}{2}\xi(1+\xi)\right] - u_{i-1}\left[\xi(2+\xi)\right] + u_i\left[\frac{1}{2}(1+\xi)(2+\xi)\right]\\
p_0 &= u_{i-1}\left[\frac{1}{2}\xi(-1+\xi)\right] + u_i\left[1-\xi^2\right] + u_{i+1}\left[\frac{1}{2}(1+\xi)(2+\xi)\right]\\
p_{+1} &= u_i\left[\frac{1}{2}(-2+\xi)(-1+\xi)\right] - u_{i+1}\left[\xi(-2+\xi)\right] + u_{i+2}\left[\frac{1}{2}\xi(-1+\xi)\right],
\end{aligned} \tag{B.7}$$

and the wide stencil is

$$\begin{aligned}
p_{\text{wide}} = u_{i-2}&\left[\frac{1}{24}(\xi+1)\xi(\xi-1)(\xi-2)\right] - u_{i-1}\left[\frac{1}{6}(\xi+2)\xi(\xi-1)(\xi-2)\right]\\
&+ u_i\left[\frac{1}{4}(\xi+2)(\xi+1)(\xi-1)(\xi-2)\right]\\
-u_{i+1}&\left[\frac{1}{6}(\xi+2)(\xi+1)\xi(\xi-2)\right] + u_{i+2}\left[\frac{1}{24}(\xi+2)(\xi+1)(\xi-1)\right].
\end{aligned} \tag{B.8}$$

The linear weights are chosen such that, at the desired point ξ, the sum of subpolynomials matches the wide polynomial:

$$p_{\text{wide}}(\xi) = \sum_{j=-1}^{1} \gamma_j p_j(\xi). \tag{B.9}$$

Solving the resulting linear system, the linear weights, γ, are defined at the location ξ:

$$\begin{aligned}
\gamma_{-1} &= \frac{1}{12}(\xi-2)(\xi-1),\\
\gamma_0 &= -\frac{1}{6}(\xi-2)(\xi+2), \quad \text{and}\\
\gamma_{+1} &= \frac{1}{12}(\xi+1)(\xi+2).
\end{aligned} \tag{B.10}$$

Nonlinear weights are then constructed with the intent to approximate the values of the linear weights in regions of the domain that are smooth, achieving maximal 5th-order of accuracy in the interpolation, while increasing dissipation in regions where the field contains sharp gradients and, correspondingly, the decreasing the order of accuracy of the method in such regions to at worst 2nd. The nonlinear weights are developed based on the spatial variation of points in the stencil,

$$\beta_l = \sum_{m=1}^{r-1} \Delta\xi^{2m-1} \int_{\xi_{i-1/2}}^{\xi_{i+1/2}} \left(\frac{d^m}{dx^m} p_l(\xi)\right)^2 d\xi. \tag{B.11}$$

and so the nonlinear weight ω_k corresponding to the k subpolynomial is

$$\alpha_k = \frac{\gamma_j}{(\epsilon_{\text{WENO}} + \beta_j)^2}, \quad \text{and} \quad \omega_k \quad = \frac{\alpha_k}{\sum_{j=-1}^{1} \alpha_j}. \tag{B.12}$$

B.3.2 WENO reconstruction

The WENO reconstruction follows a very similar derivation to the WENO interpolation. With WENO reconstruction, it is assumed that the variable is known only on a cell-averaged basis, like what is given by solving for the variable through a finite-volume discretization. The WENO interpolation procedure is undertaken with a surrogate primitive variable,

$$U(x_{i+1/2}) = \int_{x_{\min}}^{x_{i+1/2}} u(\xi) d\xi = \sum_{l=0}^{i} \Delta x \bar{u}_l. \tag{B.13}$$

Here, $x_{\min}$ is a fixed number chosen to correspond with the left-most boundary of the domain. As we know the pointwise values of U, we can perform the WENO interpolation, with the value of u obtained through the relationship $u(x) = U'(x)$. The subpolynomials are then derived for the 5th-order WENO at a point $x = \xi$,

$$\begin{aligned}
p_{-1} &= u_{i-2}\left[\frac{1}{24}(-1 + 12\xi(1+\xi))\right] + u_{i-1}\left[\frac{1}{12} - \xi(2+\xi)\right] + u_i\left[\frac{1}{24}(23 + 12\xi(3+\xi))\right] \\
p_0 &= u_{i-1}\left[\frac{1}{24}(-1 + 12\xi(-1+\xi))\right] + u_i\left[\frac{13}{12} - \xi^2\right] + u_{i+1}\left[\frac{1}{24}(-1 + 12\xi(1+\xi))\right] \\
p_{+1} &= u_i\left[\frac{1}{24}(23 + 12\xi(-3+\xi))\right] + u_{i+1}\left[\frac{1}{12} - \xi(-2+\xi)\right] + u_{i+2}\left[\frac{1}{24}(-1 + 12\xi(-1+\xi))\right]
\end{aligned} \tag{B.14}$$

The wide polynomial, in the general case of ξ, is defined similarly,

$$\begin{aligned} p_{\text{wide}} = u_{i-2}&\left[\frac{1}{1920}(9+40(-1+\xi)\xi(-5+2(-1+\xi)\xi))\right] \\ +u_{i-1}&\left[\frac{1}{480}(-29+20\xi(-17+2\xi(9-2(-1+\xi)\xi)))\right] \\ +u_{i}&\left[\frac{1067}{960}-\frac{11}{8}\xi^2+\frac{1}{4}\xi^4\right] \\ +u_{i+1}&\left[\frac{1}{480}(-29+20\xi(17-2\xi(-9+2(1+\xi)\xi)))\right] \\ +u_{i+2}&\left[\frac{1}{1920}(9+40(1+\xi)\xi(-5+2(1+\xi)\xi))\right]. \end{aligned} \tag{B.15}$$

The wide polynomial is constructed in the same manner as the subpolynomials, for a degree 4 polynomial interpolant, yielding the same 5^{th}-order accuracy as the interpolation procedure. The linear weights are

$$\begin{aligned} \gamma_{-1} &= \frac{9+200\xi-120\xi^2-160\xi^3+80\xi^4}{80(-1+12\xi(1+\xi))} \\ \gamma_{0} &= \frac{49-4548\xi^2+5360\xi^4-960\xi^6}{40-6720\xi^2+5760\xi^4} \\ \gamma_{+1} &= \frac{9-200\xi-120\xi^2+160\xi^3+80\xi^4}{80(-1+12\xi(-1+\xi))} \end{aligned} \tag{B.16}$$

B.3.3 WENO derivative

To reconstruct the derivative, we use the relationship that $u'(x) = U''(x)$ to construct the subpolynomials,

$$\begin{aligned} p_{-1} &= u_{i-2}\left[\frac{1}{2}+\xi\right]+u_{i-1}\left[-2(1+\xi)\right]+u_i\left[\frac{3}{2}+\xi\right] \\ p_{0} &= u_{i-1}\left[-\frac{1}{2}+\xi\right]+u_i\left[-2\xi\right]+u_{i+1}\left[\frac{1}{2}+\xi\right] \\ p_{+1} &= u_{i}\left[-\frac{3}{2}+\xi\right]+u_{i+1}\left[2(1-\xi)\right]+u_{i+2}\left[-\frac{1}{2}+\xi\right]. \end{aligned} \tag{B.17}$$

The wide polynomial, in the general case of ξ, is defined similarly,

$$\begin{aligned} p_{\text{wide}} = u_{i-2}&\left[\frac{1}{48}(1-2\xi)(5+4\xi(1-\xi)\right] \\ +u_{i-1}&\left[\frac{1}{24}(-17+4\xi(9+(3-4\xi)\xi))\right] \\ +u_{i}&\left[-\frac{11}{4}+\xi^3\right] \\ +u_{i+1}&\left[\frac{1}{24}(17-4\xi(-9+(3+4\xi)\xi))\right] \\ +u_{i+2}&\left[\frac{1}{48}(1+2\xi)(-5+4\xi(1+\xi))\right]. \end{aligned} \tag{B.18}$$

The wide polynomial is constructed in the same manner as the subpolynomials, for what is now a degree 3 polynomial interpolant, yielding a reduced 4$^{\text{th}}$-order accuracy. The linear weights for this reconstruction are

$$\begin{aligned} \gamma_{-1} &= \frac{5-6\xi-12\xi^2+8\xi^3}{24(1+2\xi)} \\ \gamma_0 &= \frac{7-48\xi^2+16\xi^4}{12(1-4\xi^2)} \\ \gamma_{+1} &= \frac{5+6\xi-12\xi^2-8\xi^3}{24(1-2\xi)}. \end{aligned} \tag{B.19}$$

The nonlinear weights require the smoothness indicators of the first-order subpolynomials. The smoothness indicators are

$$\begin{aligned} \beta_{-1} &= (u_{i-2}-2u_{i-1}+u_i)^2 \\ \beta_0 &= (u_{i-1}-2u_i+u_{i+1})^2 \\ \beta_{+1} &= (u_i-2u_{i+1}+u_{i+2})^2. \end{aligned} \tag{B.20}$$

B.4 Treatment of negative weights

In this reconstruction, the possibility exists that some of the linear weights are negative. The presence of the negative linear weights in the discretization can cause numerical instability [78]. In order to avoid instability, a special treatment is applied to cases that have negative linear weights. One example would be the reconstruction of the value of variable at the center of the finite volume cell, which appears in the modified Gaussian quadrature technique. The treatment of the negative weights is to split thee reconstruction,

$$\tilde{\gamma}_i^+ = \frac{1}{2}(\gamma_i + \theta|\gamma_i|), \quad \tilde{\gamma}_i^- = \tilde{\gamma}_i^+ - \gamma_i, \tag{B.21}$$

with i ranging between -1 and 1. The value of θ is taken as $\theta = 3$ in accordance to [78]. In the case that a linear weight is positive, this results in value of $\gamma_i^+ = 2\gamma_i$ and in the case of a negative weight, $\gamma_i^+ = \gamma_i$.

The split linear weights are scaled are with the sum of either the positive or negative weights,

$$\sigma^\pm = \sum_{j=-1}^{1} \tilde{\gamma}_j^\pm, \quad \gamma_i^\pm = \tilde{\gamma}_i^\pm / \sigma^\pm. \tag{B.22}$$

We can then construct the wide polynomial as a sum of the positively weighted and negatively weighted polynomials,

$$p_{\text{wide}}(x) = \sigma^+ p^+ - \sigma^- p^-, \tag{B.23}$$

with the split polynomials

$$p_{\text{wide}}^\pm(x) = \sum_{j=-1}^{1} \gamma_j^\pm p_j(x). \tag{B.24}$$

The split wide polynomials, $p_{\text{wide}}^\pm(x)$, can then be used to define linear weights individually, yielding positive linear weights for both of the two split polynomials. Thus, all of the linear weights are positive at the cost of an additional polynomial evaluation for the discretization.

References

[1] R. Barlow, J. Frank, Piloted CH_4/Air Flames C, D, E, and F Release 2.1, 2007. Available at `http://www.sandia.gov/TNF/DataArch/FlameD.html`.

[2] C. Pantano, Direct simulation of non-premixed flame extinction in a methane-air jet with reduced chemistry, Journal of Fluid Mechanics 514 (2004) 231–270.

[3] R. Knaus, C. Pantano, On the effect of heat release in turbulence spectra of non-premixed reacting shear layers, Journal of Fluid Mechanics 626 (2009) 67.

[4] S. Domino, SIERRA Low Mach Module Nalu: Theory manual, 2014. Available at `github.com/spdomin/Nalu`.

[5] A. Liñán, The asymptotic structure of counterflow diffusion flames for large activation energies, Acta Astronautica 1 (1974) 1007–1039.

[6] J. Hewson, A. Kerstein, Local extinction and reignition in nonpremixed turbulent CO/H_2/N_2 jet flames, Combustion science and technology 174 (2002) 35–66.

[7] J. W. Dold, L. J. Hartley, D. Green, Dynamics of laminar triple-flamelet structures in non-premixed turbulent combustion, Dynamical Issues in Combustion Theory, The IMA Volumes in Mathematics and its Applications (1991).

[8] V. S. Santoro, A. Liñán, A. Gomez, Propagation of edge flames in counterflow mixing layers: experiments and theory, Proceedings of the Combustion Institute 28 (2000) 2039–2046.

[9] G. Amantini, J. H. Frank, A. Gomez, Experiments on standing and traveling edge flames around flame holes, Proceedings of the Combustion Institute 30 (2005) 313–321.

[10] F. Williams, S. Murthy, Turbulent mixing in nonreactive and reactive flows, Complex Chemical Reaction Systems," SNB Murphy, Ed.(Plenum, New York, 1975) (1975) 189.

[11] N. Peters, Laminar diffusion flamelet models in non-premixed turbulent combustion, Progress in Energy and Combustion Science 10 (1984) 319–339.

[12] N. Peters, Laminar flamelet concepts in turbulent combustion, Proc. Combust. Inst. 21 (1986) 1231–1250.

[13] M. Cha, P. D. Ronney, Propagation rates of nonpremixed edge flames, Combustion and Flame 146 (2006) 312 – 328.

[14] L. Wang, E. R. Hawkes, J. H. Chen, Flame edge statistics in turbulent combustion, Proceedings of the Combustion Institute 33 (2011) 1439–1446.

[15] V. Favier, L. Vervisch, Edge flames and partially premixed combustion in diffusion flame quenching, Combustion and flame 125 (2001) 788–803.

[16] J. Buckmaster, Edge-flames, Progress in Energy and Combustion Science 28 (2002) 435–475.

[17] M. Matalon, Flame dynamics, Proceedings of the Combustion Institute 32 (2009) 57–82.

[18] J. W. Dold, L. J. Hartley, D. Green, Dynamics of laminar triple-flamelet structures in non-premixed turbulent combustion, Dynamics of hetergeneous combustion and reacting systems 152 (1993) 70–86.

[19] C. Pantano, D. I. Pullin, A statistical description of turbulent diffusion flame holes, Combustion and Flame 137 (2004) 295 – 305.

[20] J. Kim, S. H. Chung, K. Y. Ahn, J. S. Kim, Simulation of a diffusion flame in turbulent mixing layer by the flame hole dynamics model with level-set method, Combustion Theory and Modelling 10 (2006) 219–240.

[21] A. Ratz, A. Voigt, PDE's on surfaces—a diffuse interface approach, Commun. Math. Sci. 4 (2006) 575–590.

[22] M. Bertalmıo, L. Cheng, S. Osher, G. Sapiro, Variational problems and partial differential equations on implicit surfaces, Journal of Computational Physics 174 (2001) 759–780.

[23] J. Greer, An improvement of a recent Eulerian method for solving PDEs on general geometries, J. Sci. Comp. 29 (2006) 321–352.

[24] D. Adalsteinsson, J. Sethian, Transport and diffusion of material quantities on propagating interfaces via level set methods, Journal of Computational Physics 185 (2003) 271–288.

[25] J. Xu, H. Zhao, An Eulerian formulation for solving partial differential equations along a moving interface, Journal of Scientific Computing 19 (2003) 573–594.

[26] S. Ruuth, B. Merriman, A simple embedding method for solving partial differential equations on surfaces, J. Comput. Phys. 227 (2008) 1943–1961.

[27] C. Macdonald, S. Ruuth, Level set equations on surfaces via the closest point method, Journal of Scientific Computing 35 (2008) 219–240.

[28] C. Macdonald, J. Brandman, S. J. Ruuth, Solving eigenvalue problems on curved surfaces using the closest point method, Journal of Computational Physics 230 (2011) 7944–7956.

[29] C. Piret, The orthogonal gradients method: A radial basis functions method for solving partial differential equations on arbitrary surfaces, Journal of Computational Physics 231 (2012) 4662–4675.

[30] S. Leung, J. Lowengrub, H. Zhao, A grid based particle method for solving partial differential equations on evolving surfaces and modeling high order geometrical motion, Journal of Computational Physics 230 (2011) 2540–2561.

[31] F. Williams, Combustion theory, Addison-Wesley Pub., Reading, MA, 1985.

[32] B. Cuenot, F. Egolfopoulos, T. Poinsot, An unsteady laminar flamelet model for non-premixed combustion, Combustion Theory and Modelling 4 (2000) 77–97.

[33] J. C. Hewson, An extinction criterion for nonpremixed flames subject to brief periods of high dissipation rates, Combustion and Flame (2013).

[34] A. Dziubek, Equations for two-phase flows: a primer, Meccanica (2011) 1–18.

[35] T. März, C. Macdonald, Calculus on surfaces with general closest point functions, arXiv preprint arXiv:1202.3001 (2012).

[36] L. Simon, Lectures on geometric measure theory, 1984.

[37] M. P. Do Carmo, Differential geometry of curves and surfaces, volume 2, Prentice-Hall Englewood Cliffs, NJ, 1976.

[38] J. C. Slattery, L. Sagis, E. Oh, Interfacial transport phenomena, Springer, 2006.

[39] E. Kreyszig, Differential geometry, 1991.

[40] J. J. Yoh, X. Zhong, New hybrid Runge-Kutta methods for unsteady reactive flow simulation, AIAA journal 42 (2004) 1593–1600.

[41] Y. Chen, C. Macdonald, The closest point method and multigrid solvers for elliptic equations on surfaces, arXiv preprint arXiv:1307.4354 (2013).

[42] C. H. Gibson, Fine structure of scalar fields mixed by turbulence. i. zero-gradient points and minimal gradient surfaces, Physics of Fluids 11 (1968) 2305.

[43] L. Scriven, Dynamics of a fluid interface equation of motion for Newtonian surface fluids, Chemical Engineering Science 12 (1960) 98–108.

[44] D. Peng, B. Merriman, S. Osher, H. Zhao, M. Kang, A PDE-based fast local level set method, Journal of Computational Physics 155 (1999) 410–438.

[45] F. Memoli, G. Sapiro, S. Osher, Solving variational problems and partial differential equations mapping into general target manifolds, J. Comput. Phys 195 (2002) 263–292.

[46] H. Uddin, R. Kramer, C. Pantano, A Cartesian-based embedded geometry technique with adaptive high-order finite differences for compressible flow around complex geometries, Journal of Computational Physics (2014).

[47] E. Olsson, G. Kreiss, A conservative level set method for two phase flow, Journal of computational physics 210 (2005) 225–246.

[48] R. K. Shukla, C. Pantano, J. B. Freund, An interface capturing method for the simulation of multi-phase compressible flows, Journal of Computational Physics 229 (2010) 7411 – 7439.

[49] E. Johnsen, T. Colonius, Implementation of WENO schemes in compressible multicomponent flow problems, Journal of Computational Physics 219 (2006) 715–732.

[50] P. LeFloch, Shock waves for nonlinear hyperbolic systems in nonconservative form, Institute for Math. and its Appl., Minneapolis 593 (1989) 1989.

[51] V. A. Titarev, E. F. Toro, Finite-volume WENO schemes for three-dimensional conservation laws, J. Comput. Phys. 201 (2004) 238–260.

[52] S. Gottlieb, C. Shu, Total variation diminishing Runge-Kutta schemes, Math. Comp 67 (1998) 73–85.

[53] V. Titarev, E. Toro, MUSTA schemes for multi-dimensional hyperbolic systems: analysis and improvements, International journal for numerical methods in fluids 49 (2005) 117–147.

[54] E. Toro, V. Titarev, MUSTA fluxes for systems of conservation laws, Journal of Computational Physics 216 (2006) 403–429.

[55] E. F. Toro, Musta: A multi-stage numerical flux, Applied Numerical Mathematics 56 (2006) 1464–1479.

[56] R. P. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), Journal of Computational Physics 152 (1999) 457–492.

[57] Y. Xing, C.-W. Shu, High-order finite volume WENO schemes for the shallow water equations with dry states, Advances in Water Resources 34 (2011) 1026–1038.

[58] M. Dumbser, M. Castro, C. Parés, E. F. Toro, ADER schemes on unstructured meshes for nonconservative hyperbolic systems: Applications to geophysical flows, Computers & Fluids 38 (2009) 1731–1748.

[59] M. Dumbser, A. Hidalgo, M. Castro, C. Parés, E. F. Toro, FORCE schemes on unstructured meshes II: Non-conservative hyperbolic systems, Computer Methods in Applied Mechanics and Engineering 199 (2010) 625–647.

[60] M. Dumbser, D. S. Balsara, E. F. Toro, C.-D. Munz, A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes, Journal of Computational Physics 227 (2008) 8209–8253.

[61] F. B. Hildebrand, Introduction to numerical analysis, Courier Dover Publications, 1956.

[62] Lawrence Livermore National Laboratory, *hypre*, high performance preconditioners, `http://acts.nersc.gov/hypre/`, 2012.

[63] C. Pantano, S. Sarkar, F. A. Williams, Mixing of a conserved scalar in a turbulent reacting shear layer, Journal of Fluid Mechanics 481 (2003) 291–328.

[64] C. K. Law, Combustion physics, Cambridge University Press, 2006.

[65] C. Hasse, N. Peters, A two mixture fraction flamelet model applied to split injections in a DI diesel engine, Proceedings of the Combustion Institute 30 (2005) 2755–2762.

[66] E. M. Doran, A multi-dimensional flamelet model for ignition in multi-feed combustion systems, Ph.D. thesis, 2011.

[67] M. Ihme, Y. C. See, LES flamelet modeling of a three-stream MILD combustor: Analysis of flame sensitivity to scalar inflow conditions, Proceedings of the Combustion Institute 33 (2011) 1309–1317.

[68] P. Lin, M. Bettencourt, S. Domino, T. Fisher, M. Hoemmen, J. Hu, E. Phipps, A. Prokopenko, S. Rajamanickam, C. Siefert, E. Cyr, S. Kennon, A numerical study of reignition induced by a diffusion flame, Submitted to Parallel Processing Letters (2014).

[69] G. Schneider, M. Raw, Control volume finite-element method for heat transfer and fluid flow using colocated variables1. Computational procedure, Numerical Heat Transfer, Part A Applications 11 (1987) 363–390.

[70] M. Heroux, R. Bartlett, R. Howle, V. Hoekstra, J. Hu, T. Kolda, R. Lehoucq, K. Long, R. Pawlowski, E. Phipps, A. Salinger, H. Thornquist, R. Tuminaro, J. Willenbring, A. Williams, An Overview of Trilinos, Technical Report SAND2003-2927, Sandia National Laboratories, 2003.

[71] A. Majda, J. Sethian, The derivation and numerical solution of the equations for zero mach number combustion, Combustion science and technology 42 (1985) 185–205.

[72] H. Jasak, Error analysis and estimation for the finite volume method with applications to fluid flows, Ph.D. thesis, Imperial College London (University of London), 1996.

[73] S. B. Pope, Turbulent flows, Cambridge university press, 2000.

[74] C. Schneider, A. Dreizler, J. Janicka, E. Hassel, Flow field measurements of stable and locally extinguishing hydrocarbon-fueled jet flames, Combustion and Flame 135 (2003) 185–190.

[75] K.-P. Liao, C. Pantano, A numerical study of reignition induced by a diffusion flame, Combustion Theory and Modelling 13 (2009) 1025–1052.

[76] R. Daou, J. Daou, J. Dold, Effect of heat loss on flame edges in a non-premixed counterflow within a thermo-diffusive model, Combustion Theory and Modelling 8 (2004) 683 – 699.

[77] C. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, II, Journal of Computational Physics 83 (1989) 32 – 78.

[78] J. Shi, C. Hu, C. Shu, A technique of treating negative weights in WENO schemes, Journal of Computational Physics 175 (2002) 108 – 127.