Adaptive Methods for Simulation of Turbulent Combustion

John Bell and Marcus Day
Center for Computational Sciences and Engineering
Lawrenece Berkley National Laboratory

Introduction

We have entered a new era in turbulent combustion calculations; we can now simulate laboratory-scale turbulent reacting flows with sufficient fidelity that the computed time-dependent multi-dimensional solution should be expected to agree with experimental measurements. Such calculations are possible only through the effective use of modern high-performance computing architectures, and even then only by exploiting many of the multi-scale aspects of the reacting flow system.

Adaptive mesh refinement provides a mechanism for exploiting spatial and temporal variability in resolution requirements for given problem. With an adaptive algorithm, resolution can be changed dynamically to match local features of the flow. For example, finer resolution can be employed in a neighborhood of a flame to resolve the details of the flame structure. Fine resolution can also be used around complex fluid dynamical features such as shear layers or regions of intense turbulence while using coarse resolution where the flow has little structure such as in the products region of a premixed flame.

This chapter discusses the construction of structured, hierarchical adaptive mesh refinement schemes for modeling reacting flows. We will consider both a fully compressible formulation that is applicable to generic reacting flow simulations and a low Mach number formulation that exploits the temporal separation of scales between acoustic wave propagation and fluid motion typical of broad range of turbulent combustion scenarios. The discussion will focus on issues related to AMR for the low Mach number model; developing an adaptive compressible flow solver requires only a subset of the ideas. We will restrict our consideration to single-phase gaseous combustion and, for the sake of exposition, assume an ideal gas equation of state. The generalization to more complex equations of state is discussed in Bell et al. [1]

The objective here is to provide a pedagogical overview of the basic design concepts used to develop block-structured AMR algorithms. An outline the remainder of the chapter is given by:

1. Mathematical formulation
   (a) Compressible Navier Stokes
   (b) Low Mach number formulation
2. AMR concepts
   (a) Creating and managing the grid hierarchy
   (b) Hyperbolic conservation laws
   (c) Elliptic equations
   (d) Parabolic transport
3. Low Mach number AMR
   (a) Projection methods
   (b) Adaptive projection algorithm
4. Implementation issues and software design
5. Application

1 Mathematical formulation

Gas phase combustion problems can be modeled by the multicomponent reacting compressible Navier-Stokes equations. Ignoring external body forces and assuming an ideal gas law, the governing equations express conservation of species mass, momentum and total energy are

$$\begin{aligned}\frac{\partial \rho Y_m}{\partial t} + \nabla \cdot \rho Y_m U &= \nabla \cdot \mathcal{F}_m + \omega_m,\\ \frac{\partial \rho U}{\partial t} + \nabla \cdot \rho U U + \nabla p &= \nabla \cdot \tau,\\ \frac{\partial \rho E}{\partial t} + \nabla \cdot (\rho E + p)\, U &= \nabla \cdot (\mathcal{Q} + \tau \cdot U)\end{aligned}$$

where ρ is the density, U is the velocity, $E = \sum e_m(T) Y_m + {}^1\!/_2 U \cdot U$ is the total energy, Y_m is the mass fraction of species m, T is the temperature, and ω_m is the net mass production rate for species m due to chemical reactions. Also, $\mathcal{Q}$ is the heat flux, τ is the stress tensor, and $\mathcal{F}_m$ is the diffusion flux of the m^{th} species. Note that the internal energy, e_m, incorporates the potential chemical energy that is released by exothermic reactions in the flame. For these equations, we have $\sum Y_m = 1$, $\sum \mathcal{F}_m = 0$ and $\sum \omega_m = 0$, so that the sum of the species transport equations gives the conservation of total mass

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho U = 0$$

These evolution equations are supplemented by an equation of state, $p = p(\rho, Y_m, T)$. For the mixture of ideal gases considered here the equation of state is given by

$$p = \rho R T = \rho \mathcal{R} T \sum_m \frac{Y_m}{W_m}$$

where W_m is the molecular weight of species m. We note that the numerical solution approaches discussed here have been applied to a much more general equation of state (e.g., see [1]).

The compressible flow equations admit two types of waves, material waves that propagate at the fluid velocity U, and acoustic waves that propagate through the moving fluid at the speed of sound, c. One possible approach to simulating reacting flow is to discretize the compressible system directly and numerically resolve all of the time scales. However, for a wide range of reacting flow phenomena, including most practical combustion systems, the fluid velocity is considerably smaller than the sound speed, and this disparity in scales can be exploited to compute much more efficiently. For typical laboratory flame experiments $U \approx 3{-}30$ m/s, while the sound speed in the hot product gases is about 1000 m/s. Flows in this regime are referred to as low Mach number since the Mach number, $M = |U|/c \ll 1$.

The low Mach number combustion formulation was first introduced by Rehm and Baum [2] and was later derived rigorously from low Mach number asymptotic analysis by Majda and Sethian [3]. The basic steps of the analysis are to first nondimensionalize the system with respect to the time and length scales of the flow and then to expand the resulting terms in powers of the Mach number M. Examining the behavior as $M \to 0$, one can show that in an unconfined domain, the pressure can be decomposed as

$$p(x,t) = p_0 + \pi(x,t)$$

where p_0 is the ambient thermodynamic pressure and π is a perturbational pressure field that satisfies $\pi/p_0 \sim \mathcal{O}\left(M^2\right)$. (In a more general setting, p_0 may be a function of time.) With this decomposition, p_0 defines the thermodynamic state; thermodynamic quantities are independent of π. The flow model in this regime becomes

$$\begin{aligned}\frac{\partial \rho Y_m}{\partial t} + \nabla \cdot \rho U Y_m &= \nabla \cdot \mathcal{F}_m + \omega_m,\\ \frac{\partial \rho U}{\partial t} + \nabla \cdot \rho U U + \nabla \pi &= \nabla \cdot \tau,\\ \frac{\partial \rho h}{\partial t} + \nabla \cdot \rho U h &= \nabla \cdot \mathcal{Q},\end{aligned} \tag{1}$$

where the equation of state constrains the evolution. Note that the energy variable used in this formulation is the mass-averaged enthalpy, defined as $h(T, Y_m) = \sum_m Y_m\, h_m(T) Y_m$, where

$$h_m(T) = \int_{298K}^{T} c_{p,m}(T) dT + h_m(298K). \tag{2}$$

Here, $h_m(298K)$ is the standard heat of formation of species m at 298 K, and $c_{p,m}$ is its heat capacity at constant pressure. Empirical fits for both $h_m(T)$ and $c_{p,m}(T)$ are readily available. By combining the enthalpy equation and the species conservation equation, we can derive an alternative form of the energy equation in terms of T given by

$$\rho c_p \Big(\frac{\partial T}{\partial t} + U \cdot \nabla T \Big) = \nabla \cdot \mathcal{Q} - \sum_m h_m \Big(\nabla \cdot \mathcal{F}_m + \omega_m \Big)$$

where $c_p = \sum_m Y_m c_{pm}$ is the specific heat of the mixture at constant pressure. Although we advance the energy equation in terms of enthalpy, the temperature equation is useful in defining the constraint.

This low Mach number model retains compressibility effects due to chemical heat release and other thermal processes, but eliminates acoustic wave propagation entirely. The perturbation pressure, π, plays the role of a Lagrange multiplier to constrain the evolution so that the thermodynamic pressure is equilibrated everywhere instantaneously. Note that the form of these equations is no longer an initial value problem; the constrained system form a differential algebraic equation (DAE) system that is considerably more difficult to evolve numerically.

For the low Mach number combustion model, we do not work with the constraint give by the equation of state directly. Instead, we differentiate the equation of state along particle paths and use the evolution equations for ρ, Y_m and the auxiliary equation for T to define a constraint on the velocity:

$$\begin{aligned}
\nabla \cdot U &= -\frac{1}{\rho}\frac{D\rho}{Dt} = \frac{1}{T}\frac{DT}{Dt} + \frac{\mathcal{R}}{R}\sum_m \frac{1}{W_m}\frac{DY_m}{Dt} \\
&= \frac{1}{\rho c_p T}\left(\nabla \cdot \mathcal{Q} - \sum_m h_m \nabla \cdot Y_m \cdot \nabla h_m \right) + \frac{1}{\rho}\sum_m \frac{W}{W_m}\nabla(\rho D_m \nabla Y_m) \\
&\qquad + \frac{1}{\rho}\sum_m \left(\frac{W}{W_m} - \frac{h_m(T)}{c_p T} \right) \dot{\omega_m} \\
&\equiv S
\end{aligned}$$

In section 3 will outline an approach to solving the low Mach number system based on a projection formulation. However, we first present a number of basic concepts inherent to our approach to adaptive-grid discretizations.

2 AMR basic concepts

There are several distinct approaches to developing adaptive mesh refinement algorithms. The approach taken here uses a block-structured hierarchical form of refinement, hereafter referred to simply as AMR. AMR was first developed by Berger and Oliger [4] for hyperbolic partial differential equations. A conservative version of this methodology for gas dynamics was developed by Berger and Colella [5] and extended to three dimensions by Bell et al. [6]. This approach was extended to variable-density incompressible flow by Almgren et al. [7]. Pember et al. [8] generalized the approach to low Mach number combustion with simplified chemistry and transport. Day and Bell [9] extended the method to treat detailed chemistry and transport.

2.1 Creating and managing the grid hierarchy

In AMR, the local mesh refinement strategy is designed to exploit the advantages of uniform-grid PDE discretizations (uniform well-characterized errors, high accuracy per floating-point operation, memory-efficient

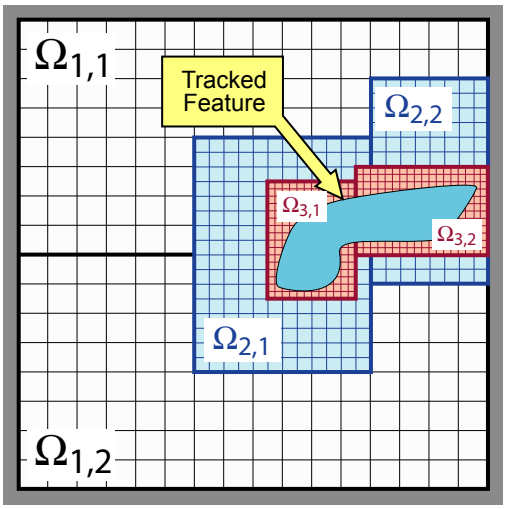

Figure 1: Typical AMR grid structure in 2D surrounding a representative feature. The computational domain (gray box) is tiled by the union of the coarsest (black) grids, $\ell_1 = \cup\, \Omega_{1,i}$. The finest level, $\ell_3 = \cup\, \Omega_{3,i}$ encloses the feature of interest, and is properly nested within ℓ_2.

stencil evaluations, etc.). The fundamental data object is a logically rectangular subdomain of uniformly spaced grid cells. A given level of refinement, ℓ, in the AMR hierarchy is represented as a union of such subdomains with a common Δx_ℓ. The refinement ratio, $r_\ell = \Delta x_\ell / \Delta x_{\ell+1}$ (typically 2 or 4) defines the reduction in grid spacing with increasing levels in the AMR hierarchy. The computational domain for an AMR calculation is rectangular and is completely tiled by the coarsest AMR level, ℓ_1. Finer levels tile successively smaller regions of the domain and satisfy the following *proper nesting* requirements (see Figure 1):

- Each subdomain at level ℓ_j, $\Omega_{j,i}$, starts and ends at the corners of a cell in the next coarser level, ℓ_{j-1}
- There must be at least n_{buf} level ℓ_j cells between the bounding edge of level ℓ_{j+1} and any uncovered ℓ_{j-1} cell.

where a cell in level ℓ_j is said to be "uncovered" if there are no cells in level ℓ_{j+1} at that location. Note that fine levels may extend to the domain boundary and that there is no requirement that level ℓ_{j+1} grids be fully contained within a single level ℓ_j grid (e.g., see $\Omega_{3,2}$ in Figure 1).

A key feature of the AMR approach is that interfaces between adjacent levels in the hierarchy are aligned with the coordinate directions *and* coincide with cell boundaries on both levels. As we will see, this greatly simplifies the construction of composite (multi-level) solution methods. Also, the subdomains, $\Omega_{\ell,j}$ at each refinement level typically contain a large number of cells, typically 16-64 cells in each direction. Compared with cell-by-cell local refinement strategies, AMR affords significant advantages in terms of access to a wide range of well-characterized discretizations for uniform grids over most of the computational domain. Nonuniform regions are limited to the substantially smaller codimension-one regions at the interfaces between levels. Finally, the AMR grid structure presents a natural parallelization strategy for distributed-data computing systems, which we discuss briefly later in this chapter.

Although the AMR grid hierarchy does not "move" with respect to the Eulerian frame, the structure is dynamic in time. At discrete intervals between time steps, subsets of the hierarchy are destroyed and regenerated in response to the evolving solution (see Figure 2). During such a *re-grid* operation, an error estimation procedure identifies individual cells at level ℓ_j requiring additional refinement (blue cells in Figure 2, for example). These cells are clustered together and surrounded with a buffer layer (pink cells in Figure 2), and then grouped into rectangular boxes and refined to generate new grids at level ℓ_{j+1}. Data on the new patches is filled by a copy-on-intersect operation from data on the previous level ℓ_{j+1} grids. In regions where the procedure generates entirely new cells at that level, data is filled by interpolation from next coarser level. This re-grid procedure must simultaneously recompute the grid structure at all finer levels, $\ell_{j+2} \ldots \ell_{max}$ in order to guarantee that the new hierarchy does not violate proper nesting requirements. In

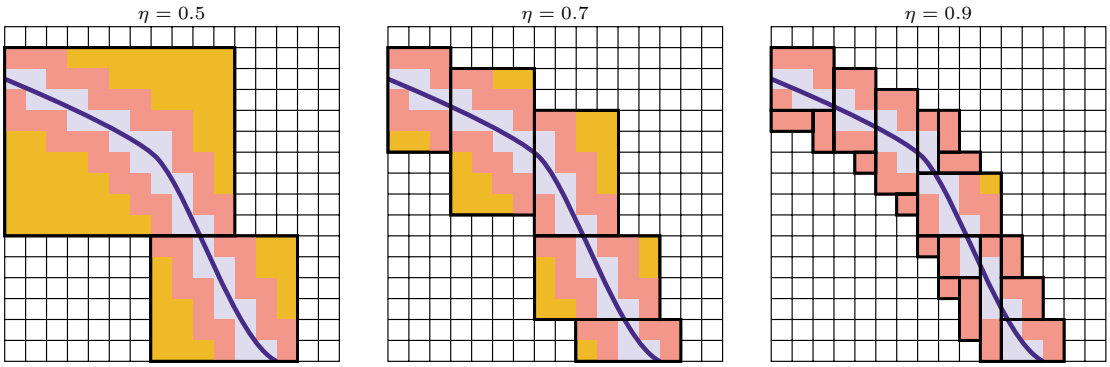

Figure 2: *Grid generation to refine a structure (blue line). Cells tagged for refinement include those containing the structure (blue cells) and those in a buffer zone surrounding them (pink cells). New rectangular grid boxes are constructed to enclose the tagged cells with a gridding efficiency, η=(# cells tagged)/ (# cells newly boxed).*

a typical application regeneration of the AMR grid hierarchy accounts for less than 1% of the total run time. The initial grid hierarchy is created using a similar strategy, except that the state data is filled by user-supplied functions for initial data at each level. The re-grid frequency required for a specific application is determined by the dynamics of refined features on the Eulerian grid, and is controlled by the error tagging procedure and through a number of adjustable parameters, including buffer zone thickness, nesting buffer width, n_{buf}, and grid efficiencies (η in Figure 2).

2.2 AMR Discretization

There are several approaches to solving partial differential equations on a hierarchical AMR grid structure. One approach would be to simply write discrete approximations that operate directly on the irregular grids. Here, we consider a different paradigm in which the levels are advanced independently with a uniform-grid scheme, and then synchronized to account for the irregular interface between levels.

In the remainder of this section, we will discuss how to develop finite-volume discretization schemes on AMR grids. The key ideas will be discussed for two-level systems in one spatial dimension with and without subcycling in time. We will briefly discuss the issues related to generalizing to multiple space dimensions and extensions to more than two levels.

Hyperbolic conservation laws

We first consider a system of hyperbolic conservation laws

$$U_t + F_x = 0$$

discretized with a time-explicit finite volume scheme:

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} = \frac{F_{i-1/2}^{n+1/2} - F_{i+1/2}^{n+1/2}}{\Delta x} \tag{3}$$

where the numerical fluxes are explicitly computed from the solution at time t^n, i.e., $F^{n+1/2} = F(U^n, \Delta t)$. The explicit treatment of fluxes implies that Δt is restricted by the CFL limit. The ideas presented here can easily be generalized to any other discretized PDE systems that are based on flux differencing. Conservative flux-based methods for the compressible Navier-Stokes equations, for example, fit within this framework.

We want to consider how to modify the basic uniform-grid scheme to update U^n on a locally refined grid, as depicted in Figure 3. The goal is to define a "composite" (multi-level) solution using a numerical flux algorithm that is unaware of the multi-level nature of the data. Initially, we will assume that both the

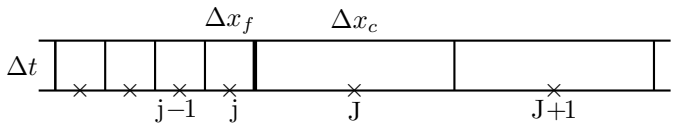

Figure 3: *A two-level 1D system with a coarse/fine interface, $c-f$, between fine cell j, and coarse cell J.*

coarse and fine grid use the same time step. In this case, we need to specify how to compute the numerical flux at each interface in Figure 3.

We define an averaging (or restriction) operator $\mathcal{A}$ that maps data at the fine grid resolution to the coarse grid resolution. We also define an interpolation operator $\mathcal{I}$ that interpolates data from the coarse resolution to fine resolution. With these operators, we can define the flux on the coarse grids away from the coarse / fine grid interface by using $\mathcal{A}(U^{f,n})$ to define data needed to compute the coarse fluxes in regions covered by fine grid. For example, if the numerical flux uses two values on each side of the interface, then to compute the flux at edge $J+\frac{1}{2}$ we define an effective coarse value at a fictitious coarse cell $J-1$ and use that value to compute the numerical flux. Similarly, we can use $\mathcal{I}(U^{c,n})$ to construct data at the fine resolution from the coarse data near the boundary of the fine grid as needed to compute fluxes at the fine resolution. To complete the specification of a composite grid solution, we only need to specifiy the flux at the coarse / fine boundary. We define that flux to be the flux computed using the fine grid data and interpolated coarse grid data. This completes the specification of a well-defined flux at each interface that enable us to advance the composite solution.

We can reinterpret this algorithm in such a way that the coarse and fine grids can be advanced independently and subsequently synchronized. This interpretation provide an algorithm to advance the composite solution in terms of an AMR grid hierarchy as defined earlier. Recall that the coarsest AMR level tiles the entire computational domain, including region that is tiled by the finer grids. Although this region is "covered" and thus not formally part of the composite solution, it provides a convenient location to store $\mathcal{A}(U^{f,n})$. Operationally, we store $\mathcal{A}(U^{f,n})$ over the entire covered portion of the coarse grid so that the coarse grid integration need not be aware of where the fine grid is located. We can then apply the uniform grid algorithm to advance the solution on the entire coarse grid in Figure 4.

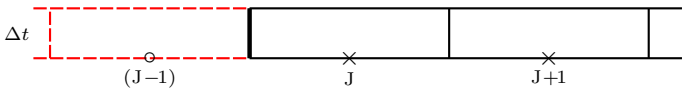

Figure 4: *The coarse integration. Covered cells (indexed in parentheses) are first filled using $\mathcal{A}(U^{f,n})$, and then become part of the uniform-grid data at the coarse level.*

In order to advance the fine level, we use $\mathcal{I}$ to generate ficticious fine data in a buffer zone along the boundary of the fine grids. These cells are often referred to as "ghost" cells (see Figure 5). The ghost cells provide sufficient data to compute fluxes needed advance the solution on the fine grid, including fluxes at the coarse / fine boundary needed to advance fine cells adjacent to the boundary.

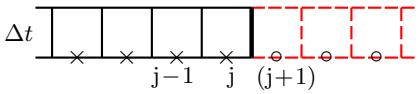

Figure 5: *The fine grid integration. Ghost cells (indexed in parentheses) are filled using $\mathcal{I}(U^{c,n})$.*

At this point we have values for U^{n+1} that are defined on both the coarse and fine levels. The only difference between this provisional solution and our desired composite solution is that the two levels have not been computed with the same flux at the $c-f$ interface. In particular, the coarse grid cell U_J^n was advanced with the coarse flux $F^{n+\frac{1}{2}}_{J-\frac{1}{2}}$ instead of the fine grid flux $F^f_{j+\frac{1}{2}}$ as specified in the definition of the composite solution given above. We can correct this discrepancy by modifying U_J^{n+1} by

$$\Delta x_c U_J^{n+1} := \Delta x_c U_J^{n+1} - \Delta t^f F^c_{J-\frac{1}{2}} + \Delta t^f F^f_{j+\frac{1}{2}}.$$

This inter-level synchronization step is referred to as *refluxing*.

Sub-cycling

We can extend the notion of refluxing to enable us to subcycle in time, provided that Δt^c is an integer multiple of Δt^f. First we advance the coarse grid with Δt^c using the uniform grid algorithm as shown in Figure 6. Next, we advance the fine grid r steps with $\Delta t^f = \Delta t^c / r$. Ghost data (in parentheses) is required

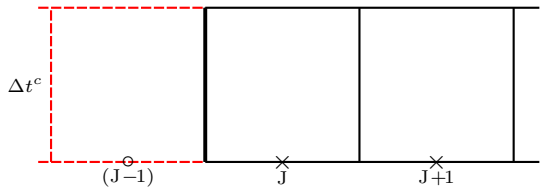

Figure 6: *Time advance of the coarse level near a coarse/fine interface in a subcycled integration. Ghost data (in parentheses) is obtained by the restriction operator,* $\mathcal{A}(U^{f,n})$.

on the boundary of the fine grid for each subcycled step. Here the data must be interpolated in time and space in order to provide consistent boundary conditions for the fine levels at intermediate times, as shown in Figure 7. The only modification to the algorithm that is necessary for sub-cyling is that the refluxing

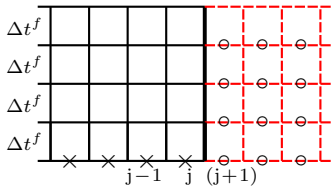

Figure 7: *Time advance of the fine level near a coarse/fine interface in a subcycled integration. Ghost data (in parentheses) is obtained by the interpolation operator,* $\mathcal{I}(U^{c,n}, U^{c,n+1})$*, and interpolation in space* and *time.*

correction to the coarse cell at the boundary of the fine grid is now summed over the r fine grid time steps

$$\Delta x_c U_J^{n+1} := \Delta x_c U_J^{n+1} - \Delta t^c F^c_{J-\frac{1}{2}} + \sum \Delta t^f F^f_{j+\frac{1}{2}} \tag{4}$$

From this discussion, we can abstract out a process for developing PDE discretizations on a locally refined grid using a uniform-grid flux-based discretization method, and arrive at a set of discretization design principles:

1. Define what is meant by the *solution* on the composite grid hierarchy.
2. Identify the errors that result from solving the equations on each level of the hierarchy "independently".
3. Solve correction equation(s) to "fix" the solution.
4. For subcycling, average the correction in time.

Within this setting, the coarse grid supplies Dirichlet data as boundary conditions for the fine grids. The errors then take the form of a flux mismatch at the coarse/fine interface. These ideas can be extended to multiple dimensions simply by defining suitable operators for $\mathcal{A}$ and $\mathcal{I}$. Because of the locality property of the refluxing operation, the algorithms discussed above can be applied recursively when there are more than two levels.

Elliptic

Next we illustrate how to apply the design principles described above to construct an algorithm to solve an elliptic equation on an AMR grid. We first consider how to define an appropriate composite solution to

$$-\phi_{xx} = \rho$$

on the AMR grid shown in Figure 3. A second-order discretization at all fine grid points $i \neq j$ can be written based on centered difference approximations as

$$-\frac{1}{\Delta x_f}\left(\frac{(\phi_{i+1}-\phi_i)}{\Delta x_f}-\frac{(\phi_i-\phi_{i-1})}{\Delta x_f}\right)=\rho_i,$$

and at all coarse grid points $I \neq J$ as

$$-\frac{1}{\Delta x_c}\left(\frac{(\phi_{I+1}-\phi_I)}{\Delta x_c}-\frac{(\phi_I-\phi_{I-1})}{\Delta x_c}\right)=\rho_I.$$

These discretizations are analogous to the flux form used in Equation 3 with ϕ_x playing the role of the numerical flux. At the $c-f$ boundary, the computation ϕ_x^{c-f} is slightly more complicated. The strategy is to build a polynomial interpolant, $I^e(\phi)$ using both coarse and fine data neighboring the $c-f$ interface to define an effective value for ϕ_{I^e} at $j+1$. Then approximate

$$\phi_x^{c-f}=\frac{\phi_{I^e}-\phi j}{\Delta x_f} \tag{5}$$

We can then discretize the equation using

$$-\frac{1}{\Delta x_f}\left(\phi_x^{c-f}-\frac{(\phi_j-\phi_{j-1})}{\Delta x_f}\right)=\rho_j$$

at $i=j$ and

$$-\frac{1}{\Delta x_c}\left(\frac{(\phi_{J+1}-\phi_J)}{\Delta x_c}-\phi_x^{c-f}\right)=\rho_J$$

at $I=J$. Provided the interpolant $I^e(\phi)$ is sufficiently accurate, this set of expressions over the composite grid defines a suitable discrete approximation to the elliptic equation. In particular, in the case of the second-order stencil used away from the $c-f$ boundary, the interpolant I_e needs to be sufficiently accurate that the local truncation error at the coarse fine boundary is first-order accurate to ensure second-order accuracy of the composite solution.

As before, we now consider what happens when we discretize the system on the coarse and fine grids in two separate steps, and we use our composite discretization to construct the appropriate synchronization. In particular, we first solve the coarse-grid problem

$$-\frac{1}{\Delta x_c}\left(\frac{(\overline{\phi}_{I+1}-\overline{\phi}_I)}{\Delta x_c}-\frac{(\overline{\phi}_I-\overline{\phi}_{I-1})}{\Delta x_c}\right)=\rho_I$$

at *all* coarse grid points I, including those covered by the fine cells. We then solve

$$-\frac{1}{\Delta x_f}\left(\frac{(\overline{\phi}_{i+1}-\overline{\phi}_i)}{\Delta x_f}-\frac{(\overline{\phi}_i-\overline{\phi}_{i-1})}{\Delta x_f}\right)=\rho_i$$

at all fine grid points, $i \neq j$. At $i=j$, we use the "correct" stencil defined using the Eq. 5.

The composite solution, $\overline{\phi}^c$ and $\overline{\phi}^f$, obtained from solving the levels separately represents the "provisional" composite solution, which satisfies the composite equations everywhere except at J. As before, the error is manifest in the difference between ϕ_x^{c-f} and $-\frac{(\overline{\phi}_J-\overline{\phi}_{J-1})}{\Delta x_c}$. If we let $e=\phi-\overline{\phi}$, then

$$-\Delta^{c-f}e=0$$

except at $I=J$, where

$$-\Delta^c e=\frac{1}{\Delta x_c}\left(\phi_x^{c-f}-\frac{(\overline{\phi}_J-\overline{\phi}_{J-1})}{\Delta x_c}\right)$$

where Δ^{c-f} is the composite discrete Laplacian defined above.

We solve this linear system for the error, e, and correct the provisional solution by setting

$$\phi^c = \overline{\phi}^c + e^c$$

$$\phi^f = \overline{\phi}^f + e^f$$

Note that while the flux mismatch is localized to the $c-f$ interface, the correction, e is not. However, because our model problem is linear, this multi-step approach exactly recovers the solution to the original composite system. Also note that the correction equation is an elliptic system on the composite grid, and is in fact no simpler to solve than the original composite problem. The benefit to this formulation however, is that it allows us to use subcycling in time for algorithms that solve an elliptic equation as part of the more complex discretization. An example of this will be discussed in the next section when we construct a semi-implicit discretization of a parabolic PDE. Also, in many circumstances we appeal to the properties of *elliptic regularity* to reduce the complexity/cost of the composite approximation. In particular, the right hand side of the synchronization equation vanishes on the region covered by the fine grid to that on the fine grid e is a diecrete harmonic. Thus, e is very smooth in the fine grid region and often will be well-represented on the coarse grid. In practice, this means that the synchronization need only be computed at the coarse level and interpolated to the fine resolution.

These options also extend to more than two levels of refinement. As above one can form and solve a composite correction equation over the entire hierarchy or one can solve only on the coarse grid and interpolate to the finer grids. An additional option is to solve the composite operator on levels ℓ and $\ell+1$ and interpolate to finer levels.

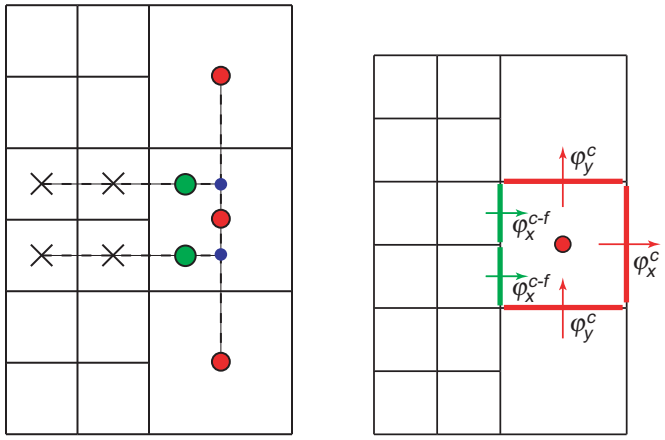

Figure 8: *The left image illustrates the interpolation stencils used to interpolate data to the location needed to evaluate ϕ_x on the fine grid, denoted by the green dot. The image on the right shows how a composite elliptic operator is defined in terms of a flux integral at the coarse / fine boundary*

In defining the composite solution for the elliptic case, we have used a slightly different paradigm than in the explicit hyperbolic case. In particular, the composite solution does not have any dependence on the coarse data covered by the fine grid. Although developing a discretization in this form in possible, the implicit coupling would require that the dependencies associated with averaging operator $\mathcal{A}$ be reflected in the overall discretization, making for extremely large stencils, particularly in multiple dimensions. In multiple dimensions, we use the interplation scheme depicted in Figure 8. We first interpolate values from coarse grid cells indicated in red to define values at the blue points using quadratic interpolation. We then interpolate from the blue points and the fine cells marked $\times$ to define values at the green points needed to evaluate the stencil, again quadratically. We note that in this form, the boundary flux ϕ_x^{c-f} is a function of coarse value at the red points and the fine values at the $\times$ points on the fine grid. The multidimensional composite discretization at the coarse cell adjacent to the fine grid is then define by the divergence of the composite flux at the boundary and standard fluxes at the other interfaces as shown in Figure 8. With these definitions, the local truncation error is uniformly first order accurate and the solution is uniformly second order.

Parabolic Systems

In this section we discuss how to combine AMR discretizations for hyperbolic and elliptic systems in order to develop an AMR discretization of the parabolic model equation

$$u_t + f_x = \varepsilon u_{xx}$$

on the AMR grid depicted in Figure 3. We consider a semi-implicit algorithm in which we combine a time-explicit treatment advective terms and Crank-Nicolson treatment of diffusion. In particular,

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + \frac{f_{i+1/2}^{n+1/2} - f_{i-1/2}^{n+1/2}}{\Delta x_{loc}} = \frac{\varepsilon}{2}\left((\Delta^{c-f} u^{n+1})_i + (\Delta^{c-f} u^n)_i\right)$$

where Δx_{loc} is the local value of Δx. Here the hyperbolic flux $f^{n+1/2} = f^{n+1/2}(u^n)$ is computed using the composite hyperbolic flux defined above and Δ^{c-f} denotes the composite discrete Laplacian. This defines a suitable definition of a composite solution.

As before, we consider what happens if we advance the coarse and fine grids separately. We first advance the coarse solution, and then use the result to generate Dirichlet conditions for the fine grid advance. Let $\overline{u}^{n+1}$ denote the provisional composite solution formed in this way. Let $e^{n+1} = u^{n+1} - \overline{u}^{n+1}$ represent the difference between the provisional and exact composite solutions. Substituting $\overline{u}^{n+1}$ into the composite discretization, we see that e^{n+1} satisfies

$$(I - \frac{\varepsilon \Delta t}{2}\Delta^{c-f})\; e^{n+1} = \frac{\Delta t}{\Delta x_c}(\delta f + \delta D)$$

where

$$\begin{aligned}
\Delta t\; \delta f &= \Delta t\; (-\overline{f}_{J-1/2} \;+\; f_{j+1/2}) \\
\Delta t\; \delta D &= \frac{\varepsilon \Delta t}{2}\left((\overline{u}^{c,n}_{x,J-1/2} + \overline{u}^{c,n+1}_{x,J-1/2}) - (u_x^{c-f,n} + u_x^{c-f,n+1})\right)
\end{aligned}$$

As before the source term is localized to the coarse cell at $c-f$ boundary and takes the form of a mismatch in coarse and fine fluxes. Updating $u^{n+1} = \overline{u}^{n+1} + e^{n+1}$ recovers the exact composite solution. As in the elliptic case, the implicit coupling in the parabolic equation leads to an implicit coupling in the synchronization equation.

The parabolic algorithm can also be extended to incorporate subcycling in time. To define the provisonal solution we first advance coarse grid over the time interval Δt^c. We then advance fine grid over the interval Δt^f a total of r times, using Dirichlet boundary data interpolated from the coarse grid. As in the explicit hyperbolic case discussed above, this interpolation must be in space and time in order to provide boundary consistent data for the fine grid at intermediate time levels during the sub-cycling.

The synchronization equation for the subcycled case now corrects the solution over the entire coarse interval; the source terms for the refluxing are summed over the r fine grid time steps. The parabolic syncrhonization equation is given by

$$(I - \frac{\varepsilon \Delta t^c}{2}\Delta^c)\; e^{n+1} = \frac{\Delta t^c}{\Delta x_c}(\delta f + \delta D)$$

where Δ^c is the coarse grid Laplacian.

$$\begin{aligned}
\Delta t^c\; \delta f &= -\Delta t^c\; \overline{f}_{J-1/2} \;+\; \sum \Delta t^f f_{j+1/2} \\
\Delta t^c\; \delta D &= \frac{\varepsilon \Delta t^c}{2}(\overline{u}^{c,n}_{x,J-1/2} + \overline{u}^{c,n+1}_{x,J-1/2}) \\
&- \sum \frac{\varepsilon \Delta t^f}{2}(u_x^{c-f,n} + u_x^{c-f,n+1})
\end{aligned}$$

In this case, for simplicity, we are only solving the correction equation on the coarse grid; corrections to the fine grid are interpolated from the coarse data. This reflects the notion that how we define the composite solution is not so straightforward and the refluxing equation does not exactly recover an "exact" composite solution. However, it does provide a second-order accurate, conservative and stable solution when the underlying basic operators are themselves second-order accurate and stable. Extension of these ideas to multiple dimensions and more than two levels of refinement are a straightforward extension of idea discussed earlier.

Summarizing the construction of the AMR discretization, there are a couple of key design goals. For the first-order hyperbolic and second-order elliptic and parabolic equations, the coarse grids provide Dirichlet boundary conditions to the fine grids. The synchronization steps ensure proper mathcing of the fluxes, and take the mathematical form of the original equation; e.g., a simple hyperbolic scheme leads to a simple local reflux corrections; an implicit parabolic equation leads to an implicit parabolic correction.

3 AMR for low Mach number combustion

An AMR implementation for an explicit compressible Navier-Stokes solver is a straightforward application of some of the ideas discussed above. Developing an AMR algorithm for the low Mach number equations is considerably more complex. Here we will only sketch the basic ideas and refer the reader to Almgren et al. [7] and Day and Bell [9] for the details. We first discuss a single grid integration algorithm and then discuss some of the key issues associated with an AMR implementation.

Our basic discretization strategy is based on a projection formulation in which we evolve the system without strictly enforcing the constraint and then project the resulting solution back onto the constraint. Structurally, the low Mach number equations evolve momentum, density, species and enthalpy subject to a divergence constraint on the velocity field, which constrains the evolution of the thermodynamics variables so that the equation of state is satisfied. Thus, the constrained evolution in the low Mach number model is similar in structure to the incompressible Navier Stokes equations. For incompressible flows, projection-based fractional step methods, which parallel standard DAE methodologies [10, 11], have proven to provide an efficient discretization strategy [12–14] Our goal then is to define a generalized projection methodology for low Mach number reacting flows. This generalization requires that we address two key differences between the incompressible flow equations and the low Mach number system. First, the low Mach number system includes finite amplitude density variations; and, second, the constraint on the velocity field is inhomogeneous.

Two different projection-based sequential algorithms have been proposed. One of these approaches, developed by McMurtry *et al.* [15] and Rutland and Ferziger [16], advances the thermodynamic variables and then uses the conservation of mass equation to constrain the evolution. Imposing the constraint in this form requires the solution of a Poisson equation. Although this approach does not fit within a mathematical projection framework, is has been successfully used by a number of authors to model reacting flows. See, for example, [17–21].

We use a different approach based on a generalized projection framework first introduced in Bell and Marcus [22]. The basic idea is that, subject to boundary conditions, any vector field, V can be decomposed as

$$V = U_d + \frac{1}{\rho}\nabla\phi$$

where U_d is divergence free and U_d and $\frac{1}{\rho}\nabla\phi$ are orthogonal with respect to a suitable inner product. This decomposition is an exact analog to the standard Hodge decomposition in a ρ-weighted inner product; e.g.,

$$\int (U_d \cdot \frac{1}{\rho}\nabla\phi)\rho \; dm = 0$$

Using this inner product, we can define a ρ-based projection, $\mathbf{P}_\rho$ such that $\mathbf{P}_\rho V = U_d$ with $||\mathbf{P}_\rho|| = 1$ and $\mathbf{P}_\rho^2 = \mathbf{P}_\rho$.

This projection operator allows us to address the finite amplitude density variations in the low Mach number system. The Majda and Sethian analysis [3] shows the flow compressibility, S, can be represented in terms of the gradient of a potential,

$$\nabla \cdot \nabla\xi = S$$

Using this form, we can generalize the ρ-weighted vector field decomposition to write any velocity field as

$$V = U_d + \nabla\xi + \frac{1}{\rho}\nabla\phi$$

We can then define

$$U = \mathbf{P}_\rho(V - \nabla\xi) + \nabla\xi$$

so that $\nabla \cdot U = S$ and $\mathbf{P}_\rho(\frac{1}{\rho}\nabla\phi) = 0$. This construction, which specifically addresses finite amplitude density variations, provides the basis for a robust projection algorithm to evolve the low Mach number equations.

The basic idea of the variable-ρ projection algorithm is to advance the thermodynamic variables using discretized conservation equations, and generate a provisional velocity field with lagged approximation to the constraint. We then use the vector field decomposition to extract the component of the velocity update that satisfies the modified constraint on the velocity divergence based on the time-advanced thermodynamic variables. More specifically, we advance the species mass densities and enthalpy equations using a flux-based conservative discretization,

$$\frac{\rho^{n+1}\chi^{n+1} - \rho^n\chi^n}{\Delta t} + \nabla \cdot (\rho U^{ADV}\chi)^{n+1/2} = D_\chi + R_\chi \quad \text{for } \chi = h, Y_m \quad .$$

Here we use a Crank-Nicolson discretization of the diffusion terms, and a specialized second-order Godunov algorithm to compute the advective derivatives. As part of the Godunov algorithm we compute an advective velocity field U^{ADV} on cell interfaces that has been projected so that it satisfies the constraint as well. The chemical rate equations are decoupled in an operator split form. In particular, we first advance the chemistry over the interval, $\Delta t/2$. We then advance the advection and diffusion components over Δt, and then finish with a second advance of the chemistry by $\Delta t/2$. This operator-split strategy allows us to decouple the pointwise chemical kinetics so that we can use stiff ODE integration methodologies to advance the kinetics equations, while preserving the second-order accuracy of the integration.

In a similar way, we compute a provisional velocity based on a lagged approximation to the dynamic pressure, π

$$\frac{U^* - U^n}{\Delta t} = -[U^{ADV} \cdot \nabla U]^{n+1/2} - \frac{1}{\rho^{n+1/2}}\nabla\pi^{n-1/2} + \frac{1}{\rho^{n+1/2}}\nabla \cdot \frac{\tau^n + \tau^*}{2} \quad .$$

The updated thermodynamic variables are then used to compute the constraint at the new time level S^{n+1}. To extract the component satisfying the divergence constraint we solve the variable-coefficient linear system

$$\nabla \cdot \left(\frac{1}{\rho}\nabla\phi\right) = \nabla \cdot \vec{V}^* - S^{n+1}$$

for ϕ, where $\vec{V}^* = \vec{U}^* + (\Delta t/\rho^{n+1/2})\nabla\pi^{n-1/2}$, and set

$$\pi^{n+1/2} = \phi \qquad \text{and} \qquad \vec{U}^{n+1} = \vec{V}^* - \frac{\Delta t}{\rho^{n+1/2}}\nabla\phi \tag{6}$$

This procedure implements the generalized vector field decomposition discussed above but exploits linearity to perform only a single elliptic solve to enforce the constraint. This final projection is based on node-centered ϕ data and a conformal bilinear finite element construction on a locally refined grid. As with the finite-volume elliptic solvers discussed above, the nodal solver can be decomposed into a solution operator on each of the refinement levels independently, followed by a correction procedure. See see Reference [7] for details. The projection operator defined in this way is not a discrete projection; such approaches are referred to as approximate projection algorithms. For approximate projections, the projection step can be constructed using a number of analytically equivalent forms for V^*. The choice identified in Equation 6 were demonstrated to be the best choice form a perspective of robustness and accuracy (see Reference [23] for details).

The algorithm presented above discretely conserve density, species (up to reactions) and enthalpy. However, because we use a linearized form of the constraint, the solution can drift off the constraint surface, $p_0 = constant$. To correct for this drift, we include an relaxation term to the compressiblity expression used to compute U^{ADV}. This relaxation term is constructed to force the solution back toward the constraint without violating discrete conservation.

The adaptive time-step algorithm advances grids at different levels using time steps appropriate to that level based on CFL considerations. The procedure can most easily be thought of as a recursive algorithm, in which to advance level ℓ, $1 \le \ell \le \ell_{max}$ the following steps are taken:

- Advance level ℓ in time as if it is the only level. Supply boundary conditions for U, ρ, Y_m, h and π from level $\ell - 1$ if level $\ell > 1$, and from the physical domain boundaries.

- Compute U^{ADV} including projection to enforce the constraint
- Advance ρ, Y_m, h
- Compute provisional U^*
- Evaluate constraint, S^{n+1}
- Apply generalized projection to compute U^{n+1} and $\pi^{n+1/2}$

- If $\ell < \ell_{max}$
  - Advance level $(\ell+1)$ r times with time step $\Delta t^{\ell+1} = \frac{1}{r}\Delta t^{\ell}$ as indicated above.
  - Synchronize the data between levels ℓ and $\ell+1$, and interpolate corrections to higher levels if $\ell + 1 < \ell_{max}$.
    * Solve elliptic synchronization equation for U^{ADV}
    * Compute changes for advection terms from explicit reflux and changes to U^{ADV}
    * Solve parabolic synchronization with right hand side representing reflux corrections from adjection and diffusion
    * Apply synchronization projection to correct final velocity field. The right hand side includes terms representing the discrepancy from projecting levels independently and corrections to the velocity from the advection and diffusion synchronization.

4 Implementation issues and software design

The combination of adaptive mesh refinement and the projection-based low Mach number formulation can considerably reduce the computational cost of reacting flow simulations; however, the computational demand can still be significant, particularly when modeling turbulent flame phenomena with detailed chemistry and transort. Consequently, we must be able to implement the adaptive low Mach number algorithm described above so that we can effectively utilize high-performance parallel computers. Before discussing the implementation in detail, we first comment on the impact of some of the choices we made in developing the basic algorithm on the design of the software. Our basic discretization strategy decomposes the problem into different mathematical components to treat advection, diffusion, chemical reactions, and projections. We use an explicit treatment of advection so that the implicit solves needed for diffusion and the projection represent discrete approximations to self-adjoint elliptic partial differential equations. Consequently, we can solve the requisite linear systems using geometric multigrid iterative methods. Also, we have decomposed the dynamics so that the chemistry is advanced independent of the other processes. As a result the chemistry can be treated locally on a point-by-point basis.

Our choice of AMR strategy has a significant impact on software design. By adopting a block-structured form of AMR, the solution at each level in the hierarchy is naturally represented in terms of data defined on a collection of logically rectangular grid patches each containing a large number of points. Thus, the data is represented by a modest collection of relatively large regular data objects as compared to a point-by-point refinement strategy. This type of approach allows us to amortize the irregular aspects of an adaptive algorithm over large regular operations on the grid patches. This organization of data into large aggregate grid patches also provides a model for parallelization of the AMR methodology.

Our adaptive methodology is embodied in a hybrid C++/`FORTRAN` software system. In this framework, memory management and flow control are expressed in the C++ portions of the program and the numerically intensive portions of the computation are handled in `FORTRAN`. The software is written using a layered approach, with a foundation library, BoxLib, that is responsible for the basic data container abstractions at the lowest levels, and a framework library, AMRLib, that marshals the components of the AMR discretization. Support libraries built on BoxLib are used as necessary to implement utility components, such as interpolation of cell and interface data between levels, and linear solvers used in the projections and diffusion solves.

In BoxLib, the fundamental parallel data abstraction is a `MultiFab`. A `MultiFab` is the union of a set of distributed blocks of `FORTRAN`-compatible data arrays. Each block is defined on a `Box`, which is defined by a pair of integer coordinate tuples that identify the lower and upper bounds of the region in a global index

space. `MultiFab`'s at each level of refinement are distributed independently. The software supports two data distribution schemes, as well as a dynamic switching scheme that decides which approach to use based on the number of grids at a level and the number of processors. The first scheme is based on a heuristic knapsack algorithm as described in Crutchfield [24] and in Rendleman [25]. The second is based on the use of a Morton-ordering space-filling curve. MultiFab operations are performed with an *owner computes* rule with each processor operating independently on its local data. For operations that require data owned by other processors, the MultiFab operations are preceded by a data exchange between processors.

Each processor contains *meta-data* that is needed to fully specify the geometry and processor assignments of the `MultiFab`s. At a minimum, this requires the storage of an array of boxes for each AMR level, the refinement ratio between levels, and the physical location of the origin of the base grid coordinate indices. In the parallel implementation, meta-data also includes the processor distribution of data, which is used to dynamically evaluate the necessary communication patterns to share data amongst the processors.

Performance of Adaptive Projection

Compared to a time-explicit numerical integration of the compressible reacting flow equations, the adaptive projection methodolgy for low Mach number recting flows is enormously complex. The integration algorithm is a multi-stage procedure involving a number of elliptic/parabolic solves each time step. Dynamic regridding operations track developing features in the flow, and require frequent redistribution of the data and workload, and the workload itself is non-uniform due to the inhomogeneous nature of the reaction chemistry. Not only is it interesting to assess the parallel scalability of this agorithm to thousands of processors, it is also important to gauge the effectiveness of the low Mach number algorithm with respect to time-explicit simulation approaches directly, which historically have shown nearly ideal scaling behavior.

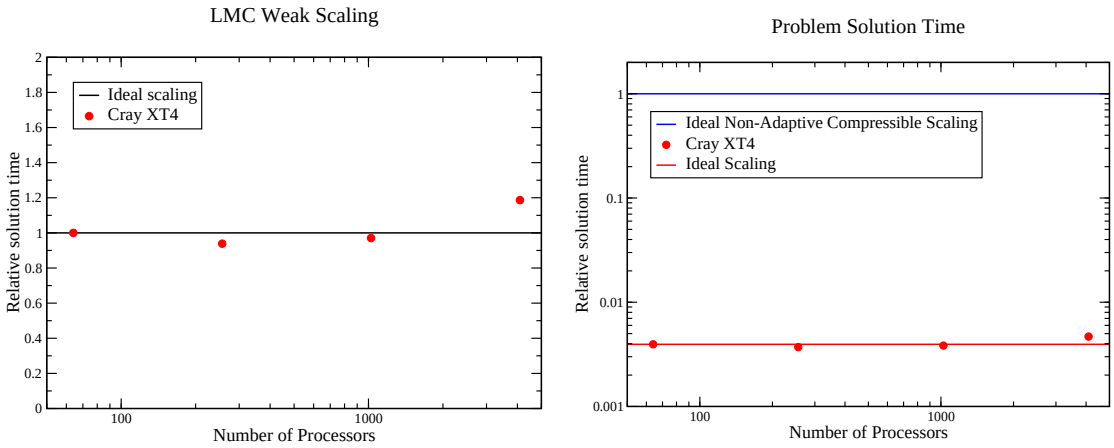

Figure 9: *Parallel performance of the low Mach number adaptive algorithm. (a) Weak scaling behavior of the adaptive low Mach number algorithm. (b) Scaling of the time-to-solution, relative to a compressible solver.*

Here we consider a weak scaling study in which the relative amount of work per processor remains constant as we increase the number of processors. The study here is based on the propagation of a doubly-periodic wrinkled premixed methane-air flame in three dimensions using the GRI-Mech 3.0 [26] chemical mechanism for methane combustion. By replicating the problem in the periodic directions, we are able to scale the problem size without modifying the problem characteristics so that we can reliably test the behavior of the full AMR algorithm. Normalize computational times versus number of processors is presented in Figure 9(a). This figure shows a modest increase in execution time of approximately 20% as we increase the number of processors from 64 to 4096 on the XT4-type architecture. A more interesting metric than simple scalability is the relative performance of the low Mach number methodology to that of a time-explicit scheme. Figure 9(b) shows the time-to-solution for the same range of processor counts using our low Mach number algorithm compare to the idealized performance of a non-adaptive compressible reacting flow solver [27]. Note that

for this study, we did not run the compressible code at all of the resolutions; we simply extrapolated the performance from 64 processors of assuming ideal scaling. The data shows that the low Mach number methodology is more than a factor of 200 faster than the compressible method. It is important to point out additionally that this estimate is quite conservative, since for many interesting 3D problems the finest level typically covers a far smaller fraction of the domain (the example presented below, for example, required the highest resolution on less than 4% of the domain).

5 Application – Lean premixed hydrogen flames

The AMR algorithm design principles and BoxLib software library have been used to build a wide variety of PDE integration schemes, with applications in compressible and incompressible turbulence (jets, shear layers), astrophysics, porous media, fluctuating hydrodynamics, and reacting flows. The adaptive projection scheme has been particularly successful in studies of low Mach number reacting flows, such as astrophysical explosions and terrestrial combustion. In this section, we summarize a recent combustion study that serves to highlight the applicability of this AMR scheme to real systems with complex transport and chemistry models at the laboratory scale.

Background

Within the combustion community there is considerable interest in developing fuel-flexible burners that can be used to stabilize lean premixed flames in a stationary turbine designed for power generation. Low-swirl burner technology, originally introduced by Bedat and Cheng [28] as tool for studying the fundamental properties of lean, premixed turbulent flames, has the potential for meeting this need. Burners based on modifications of the original design have been used by a number of research groups [29–32], and have the potential for use in the design of next-generation, lean premixed combustion systems, including those burning lean hydrogen at both at atmospheric and elevated pressures [33].

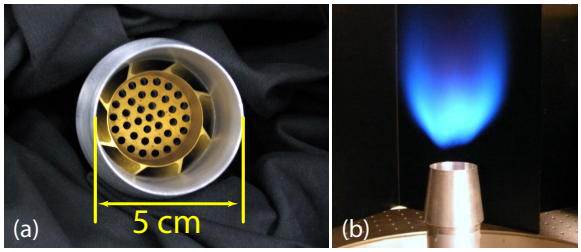

Figure 10: *(a) Low-swirl nozzle, showing vanes and turbulence generation plate. (b) typical turbulent low-swirl methane flame stabilized in divergent flow above the low-swirl nozzle.*

The low-swirl burner concept is extremely simple: premixed fuel exits a pipe after passing through a turbulence generation plate and an annular set of curved vanes as shown in Figure 10. The vanes impart a swirl component to the flow over a narrow layer near the pipe wall, and a detached premixed flame anchors in the diverging flow above the pipe exit. Turbulence in fuel stream wrinkles the flame, which enhances the overall rate of combustion in the device; the flames stabilize where the mean burning speed matches the axial flow velocity. Application of these types of burners, particularly for alternative fuels, depends on improving our understanding of basic flame structure, stabilization mechanisms, emissions and response to changes in fuel. Numerical modeling has the potential to address some of these issues, but simulation of these types of burners has proven to be difficult because of the large range of spatial and temporal scales in the system; the bulk of the analysis to date has been experimental.

As noted in Bell et al. [34], the detailed structure of lean premixed flames becomes particularly important and difficult to simulate when burning hydrogen. Lean hydrogen-air flames burn in cellular structures—localized regions of intense burning, separated by regions of local extinction. In this regime, the flame surface is broken into discontinuous segments. This type of structure introduces severe difficulties in applying standard turbulence/chemistry interaction models, which are based on the presence of a highly wrinkled but continuous flame surface that propagates locally as an idealized laminar flame structure. In the absence of a suitably general model for the turbulent combustion of lean hydrogen-air mixtures, numerical simulations must incorporate sufficient detail in the chemical kinetics, differential species transport and turbulent fluid dynamics to capture all the important couplings in these flows.

Models and Setup

For this problem, we treat the fluid as a mixture of perfect gases. We use a mixture-averaged model for differential species diffusion, ignoring Soret, Dufour and radiative transport processes (see [35] for a complete discussion of this approximation). In this case, the diffusive fluxes in Equations 1 can be written

$$\mathcal{F}_m = \rho D_m \nabla Y_m, \quad \tau = 2\eta_{mix}\mathcal{S} - \frac{2}{3}\eta_{mix}\nabla \cdot U, \quad \mathcal{Q} = \lambda_{mix}\nabla T - h_m \mathcal{F}_m$$

where $\mathcal{S}$ is the symmetric part of strain tensor, D_m is the mixture-averaged mass diffusion coefficient for species m, and η_{mix} and λ_{mix} are the mixture-averaged viscosity and thermal conductivity, respectively. A lean hydrogen-air inlet fuel mixture (ϕ=0.37) was modeled with the hydrogen sub-mechanism of GRI-Mech 2.11. There are 9 chemical speices and 27 fundamental Arrhenius chemical reactions. The transport coefficients and thermodynamic relationships are obtained from EGLib [36].

An idealized flat unstretched steady 1D (the so-called "laminar flame") configuration provides scale factors that characterize this hydrogen-air flame: the thermal thicknesses is approximately 800 μm; the full half-width maximum of the fuel consumption layer is approximately 0.5 mm. The computational domain for this study measures 25 cm^3 (we aassume that this will place the computational boundaries sufficiently far from the flame as to not significantly affect its dynamics). The base mesh for the simulation is a uniform grid of 256^3 cells and we use 3 additional levels of factor-of-two grid refinement to track regions of high vorticity (turbulence) and reactivity (combustion). The flame is contained entirely within the finest level with an effective resolution of 2048^3. In previous work, we demonstrated that this level of resolution is adequate to capture the detailed structure of the flame including the peak fuel consumption, the thermal field and major species. Note that this level occupies less than 4% of the entire computational domain. Although evolution of the flow outside of the fine grid region certainly impacts on the flame dynamics, there are no features in the flow that require high resolution.

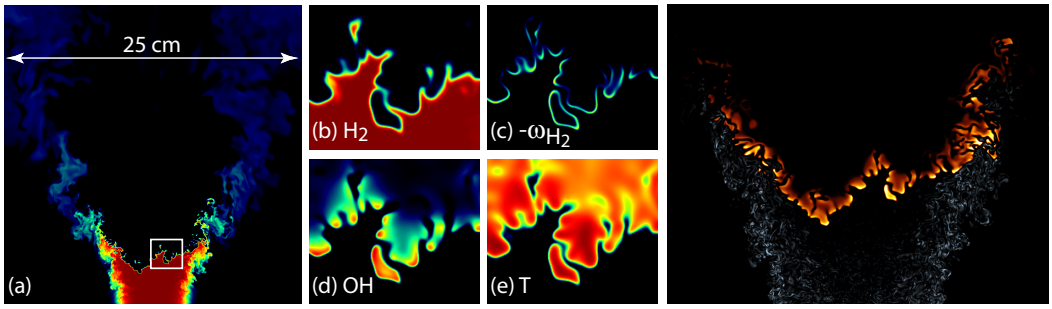

Figure 11: *Vertical slice of solution centered on nozzle axis (Frames (b)-(e) over 2.5 cm white box in (a)). (a-b) Mole fraction of H_2, (c) H_2 Consumption rate, (d) Mole fraction of OH, (e) temperature, and (f) Composite image (width = 8cm) depicts OH concentration (in orange) and the vorticity magnitude (in grey).*

Flow from the low-swirl nozzle enters our computational domain centered on the bottom boundary. Profiles for the velocity components at the nozzle incorporate an experimental characterization provided by Petersson et al. [29]. On the inlet face outside the nozzle, a 35 cm/s upward coflow of cold air is specified. Turbulent fluctuations in the nozzle flow were prepared in an auxiliary simulation to have the experimentally measured intensity and integral length scale, and were added to the mean flows as a time-dependent boundary function. The remaining boundaries are outflow. The domain was intitially filled with air at standard conditions, except for a small volume of hot air above the nozzle. As the simulation progresss, the flame ignites, and propagates downstream in the flow until reaching a quasi-stationary position in the radially divergent flow field. The additional levels of grid refinement were added and the simulation continued until reaching a new quasi-steady configuration.

Simulation Results

In Figure 11, we show a cross-section through the middle of the simulation that provides a picture of the overall structure of the flame. In the upper part of the slice plane in Figure 11(a), one sees a faint cloud of unburnt fuel at low concentrations (blue). This represents fuel that has been sufficiently diluted with air that it is below the flamability limit. Figure 11(b-e) show various other fields near the flame surface. Turbulent fluctuations from two primary sources interact with the flame (see Figure 11(f)): plate turbulence from the inlet advects downstream and wrinkles the flame in the central core; a shear layer from the swirl entrains coflowing air from the sides and reduces fuel concentrations below flammability.

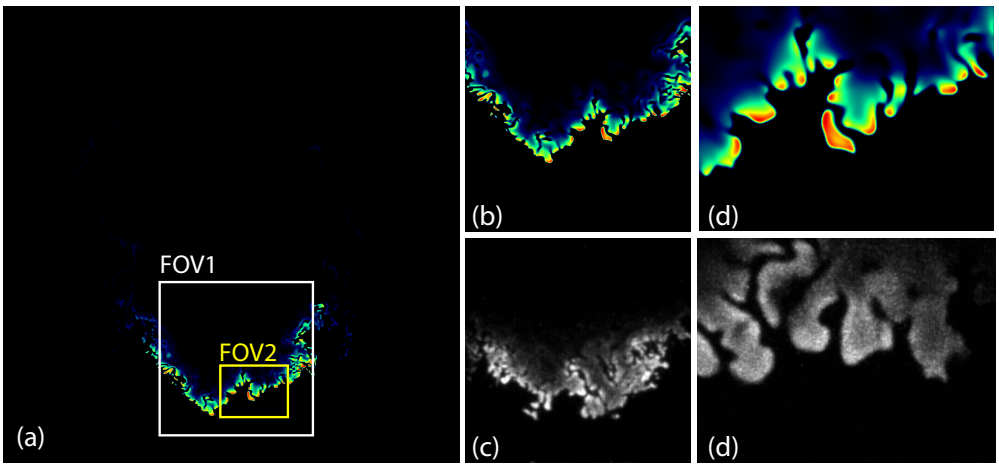

Figure 12: *(a) Typical slice of OH concentration on vertical midplane from the simulation. Red indicates high relative values, and are correlated with the flame front. The white box ("FOV1") represents the field of view of OH-PLIF measurements from the representative low swirl experiment focused on the large-scale flame structure. The smaller yellow box ("FOV2") represents the experimental field-of-view OH-PLIF measurements focused more tightly on the local structure of the flame surface (width of FOV2 is approximately 26 mm). (b) Zoom of profile in (a) over FOV1, (c) typical experimental OH-PLIF image over FOV1, (d) Zoom of profile in (a) over FOV2, and (e) typical experimental OH-PLIF image over FOV2.*

One of the principal diagnostics used in the experiments is OH-PLIF (planar laser-induced fluorescence) based on imaging the flouresence of OH radicals excited by a tuned laser sheet. In Figure 12, we show a typical vertical slice of the OH concentration from the simulation alongside typical PLIF images from the low-swirl burner experiment at similar conditions. Figure 12(a) shows the profile of OH over the $(25\ \mathrm{cm})^2$ slice, while Figures 12(b) and (d) show progressive enlargements of the data corresponding to the field of view (FOV) of the typical OH-PLIF data shown in Figures 12(c) and (e). The figure shows that the simulation captures with remarkable fidelity the primary features of these flames, including their distribution of sizes, shapes and global structure. The simulation also captures the observed variability of the OH signal (brightness on the experimental images) along the flame surface. Previous work on turbulent hydrogen flames shows that the high diffusivity of H_2 can lead to local enrichment of the fuel mixture along the front. This local enrichment can lead to intensification of local burning, which leads to increased OH that can be seen experimentally as a brigher signal in the OH-PLIF (and red regions in the simulation slice data).

From analysis of the simulation data, we can obtain a detailed characterization of the local flame structure that is not feasible from the experimental data. We begin with an examination of the fuel consumption rate. As with the OH-PLIF data, the fuel consumption shows considerable variability along the flame front. To quantify this behavior, we extract the T=1144 K surface, which corresponds to that of the peak fuel consumption in the flat laminar flame at ϕ=0.37. In Figure 13, this isotherm is colored by local fuel consumption in the core of the burner. For comparison, we also show the analogous image for a freely propagating hydrogen flame (taken from the study in [37]). Both images are based on the same color map, and the length scales are as indicated in the figure. Comparing these images, one can see that in the turbulent flame, there are finer structures than in the non-turbulent case. The turbulent flame shows more ridge-like,

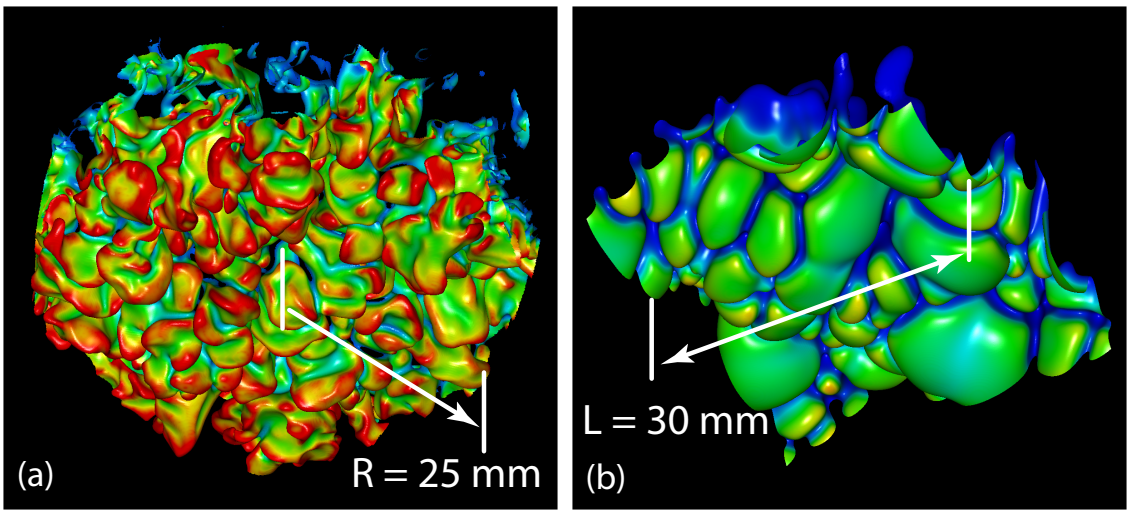

Figure 13: *Flame isotherm, $T = 1144$ K, colored by the local rate of H_2 consumption. (a) Low-swirl flame isotherm, conditioned on $r < 25mm$, $z < 8mm$. (b) Representative isotherm of an idealized ϕ=0.37 H_2-air flame propagating freely in uniform flow.*

cylindrical structures compared to the freely propagating flame in which the features are more spherical. Finally, the burning rate over most of the flame surface is considerably higher in the turbulent configuration, particularly where the flame is tightly folded.

To quantify the relationship between flame curvature and local burning speed we compute a local consumption-based flame speed along the isotherm. For this construction, we triangulate the isotherm, trimming away sections whether the fuel consumption is less than half the laminar flame value, and construct a local coordinate system in a neighbor hood of the flame. We can then define a local consumption-based flame speed on the triangulated isotherm surface by integrating fuel consumption through a region normal to the flame. (See [37] for a detailed description of this construction.) In Figure 14 we show a joint PDF of mean curvature and s_c, normalized the laminar burning speed, s_L, of a flat steady hydrogen-air flame at ϕ=0.37. This figure shows a strong correlation between local burning speed and curvature. Regions of intense burning correspond to regions of positive curvature (positive curvature corresponds to regions where the center of curvature is on the products side of the flame interface). Also note that the dominant behavior corresponds to local burning speeds that range from 2 to 4 times the laminar flame speed. Even at zero curvature, the most probable local burning speed is three times the laminar flame speed. This type of behavior illustrates the challenges in developing suitable turbulence / chemistry interaction models for lean hydrogen flames and underscores the need to perform simulations of laboratory-scale turbulent flames.

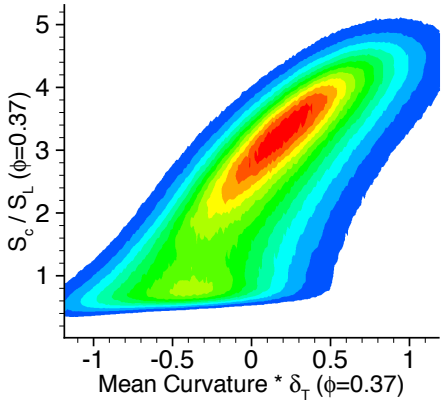

Figure 14: *Joint probability distribution function of s_c, with mean curvature.*

6 Summary

In this chapter, we have discussed the development of parallel, adaptive solvers for reacting flows. The particular class of adaptive methods we consider are hierarchical block-structured methods that support subcycling in time. We have discussed the key ideas needed to design adaptive methods within this context

and illustrated how those approaches are instantiated for various simple classical partial differential equations. We then showed how those basic discretization concepts can be interwoven to construct an adaptive low Mach number simulation capability suitable. The basic algorithmic constructs associated with block-structured AMR motivate the design of a software framework to support parallel implementation of these algorithms. The overall synthesis of these ideas leads to an enhanced simulation capability that has made it possible to simulate realistic turbulent flames without using explicit models for turbulence or turbulence / chemistry interaction.

Acknowledgments

This work was support by the DOE Office of Advanced Scientific Computing Research under the U.S. Department of Energy under contract No. DE-AC02-05CH11231. The computations presented were performed on Franklin at NERSC as part of an INCITE award.

References

[1] J. B. Bell, M. S. Day, C. A. Rendleman, S. E. Woosley, and M. A. Zingale. Adaptive low mach number simulations of nuclear flame microphysics. *Journal of Computational Physics*, 195(2):677–694, 2004.

[2] R. G. Rehm and H. R. Baum. The equations of motion for thermally driven buoyant flows. *N. B. S. J. Res.*, 83:297–308, 1978.

[3] A. Majda and J. A. Sethian. The derivation and numerical solution of the equations for zero Mach number combustion. *Combust. Sci. Technol.*, 42:185–205, 1985.

[4] M. J. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53:484–512, March 1984.

[5] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82(1):64–84, May 1989.

[6] J. Bell, M. Berger, J. Saltzman, and M. Welcome. A three-dimensional adaptive mesh refinement for hyperbolic conservation laws. *SIAM Journal on Scientific and Statistical Computing*, 15(1):127–138, 1994.

[7] A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell, and M. L. Welcome. A conservative adaptive projection method for the variable density incompressible Navier-Stokes equations. *J. Comput. Phys.*, 142:1–46, May 1998.

[8] R. B. Pember, L. H. Howell, J. B. Bell, P. Colella, W. Y. Crutchfield, W. A. Fiveland, and J. P. Jessee. An adaptive projection method for unsteady, low-Mach number combustion. *Combust. Sci. Technol.*, 140:123–168, 1998.

[9] M. S. Day and J. B. Bell. Numerical simulation of laminar reacting flows with complex chemistry. *Combust. Theory Modelling*, 4:535–556, 2000.

[10] K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical Solution of Initial-Value problems in Differential-Algrebraic Equations*. SIAM, Philadelphia, PA, 1996.

[11] U. Ascher and L. R. Petzold. Projected implicit Runge Kutta methods for differential algebraic systems. *SIAM J. Num. Anal.*, 28:1097–1120, 1991.

[12] A. J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. Comp.*, 22:745–762, 1968.

[13] J. B. Bell, P. Colella, and H. M. Glaz. A second-order projection method for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 85(2):257–283, December 1989.

[14] A. S. Almgren, J. B. Bell, and W. G. Szymczak. A numerical method for the incompressible Navier-Stokes equations based on an approximate projection. *SIAM J. Sci. Comput.*, 17(2):358–369, March 1996.

[15] P.A. McMurtry, W.-H. Jou, J.J. Riley, and R.W. Metcalfe. Direct numerical simulations or a reacting mixing layer with chemical heat release. *AIAA J.*, 24:962, 1986.

[16] C.J. Rutland and J.H. Ferziger. Simulations of flame-vortex interactions. *Combust. Flame*, 84:343, 1991.

[17] S. Zhang and C. J. Rutland. Premixed flame effects on turbulence and pressure-related terms. *Combust. Flame*, 102:447–461, 1995.

[18] H. N. Najm and P. S. Wyckoff. Premixed flame response to unsteady strain rate and curvature. *Combust. Flame*, 110(1–2):92–112, 1997.

[19] H. N. Najm, O. M. Knio, P. H. Paul, and P. S. Wyckoff. A study of flame observables in premixed methane-air flames. *Combust. Sci. Technol.*, 140:369–403, 1998.

[20] H. N. Najm, P. S. Wyckoff, and O. M. Knio. A semi-implicit numerical scheme for reacting flow. I. Stiff chemistry. *J. Comput. Phys.*, 143:381–402, 1998.

[21] J. Qian, G. Tryggvason, and C. K. Law. Front tracking method for the motion of premixed flames. *J. Comput. Phys.*, 144:52–69, 1988.

[22] J. B. Bell and D. L. Marcus. A second-order projection method for variable density flows. *J. Comput. Phys.*, 101(2):334–348, August 1992.

[23] A. S. Almgren, J. B. Bell, and W. Y. Crutchfield. Approximate projection methods: Part I. inviscid analysis. *SIAM J. Sci. Comput.*, 22(4):1139–1159, 2000.

[24] W. Y. Crutchfield. Load balancing irregular algorithms. Technical Report UCRL-JC-107679, Lawrence Livermore National Laboratory, July 1991.

[25] Charles A. Rendleman, Vincent E. Beckner, Mike Lijewski, William Y. Crutchfield, and John B. Bell. Parallelization of structured, hierarchical adaptive mesh refinement algorithms. *Computing and Visualization in Science*, 3(3):147–157, 2000.

[26] M. Frenklach, H. Wang, M. Goldenberg, G. P. Smith, D. M. Golden, C. T. Bowman, R. K. Hanson, W. C. Gardiner, and V. Lissianski. GRI-Mech—an optimized detailed chemical reaction mechanism for methane combustion. Technical Report GRI-95/0058, Gas Research Institute, 1995. `http://www.me.berkeley.edu/gri_mech/`.

[27] J. Bell, M. Day, and A. L. Kuhl. Numerical simulations of shock-induced mixing and combustion. In *19th ICDERS*, Hakone, Japan, July 27 – August 1, 2003.

[28] C. K. Chan, K. S. Lau, W. K. Chin, and R. K. Chang. Freely propagating open premixed turbulent flames stabilized by swirl. *Proc. Combust. Inst.*, 24:511–518, 1992.

[29] P. Peterson, J. Olofsson, C. Brackman, H. Seyfried, J. Zetterberg, M. Richter, M. Alden, M. Linne, R. Cheng, A. Nauert, D. Geyer, and A. Dreizler. Simultaneous PIV/OH PLIF, Rayleigh thermometry/ OH PLIF and stereo PIV measurements in a low-swirl flame. *Appl. Opt*, 46:3928–3936, 2007.

[30] R. K. Cheng. Velocity and scalar characteristics of premixed turbulent flames stabilized by weak swirl. *Combust. Flame*, 101(1-2):1–14, 1991.

[31] K. Nogenmyr, P. Peterson, X. S. Bai, A. Nauert, J. Olofsson, C. Brackman, H. Seyfried, J. Zetterberg, Z-S. Li, M. Richter, A. Dreizler, M., Linne, and M. Alden. Large eddy simulation and experiments of stratified lean premixed methane/air turbulent flames. *Proc. Combust. Inst*, 31:1467–1475, 2007.

[32] M Mansour and Y.-C. Chen. *Experimental Thermal Fluid Sci.*, 2007. *in press.*

[33] R. K. Cheng, D. Littlejohn, P. A. Strakey, and T. Sidwell. Laboratory investigations of a low-swirl injector with h_2 and ch_4 at gas turbine conditions. *Proc. Combust. Inst.*, 32:21–46, 2009.

[34] J. B. Bell, R. K. Cheng, M. S. Day, and I. G. Shepherd. Numerical simulation of Lewis number effects on lean premixed turbulent flames. *Proc. Combust. Inst.*, 31:1309–1317, 2007.

[35] A. Ern and V. Giovangigli. *Multicomponent Transport Algorithms*, volume m24 of *Lecture Notes in Physics*. Springer-Verlag, Berlin, 1994.

[36] A. Ern and V. Giovangigli. EGLIB: A General-Purpose Fortran Library for Multicomponent Transport Property Evaluations. *J. Comput. Phys.*, 120:105–116, 2005.

[37] M. S. Day, J. B. Bell, P.-T. Bremer, V. Pascucci, and V. E. Beckner. Turbulence effects on cellular burning structures in lean premixed hydrogen flames. *Combustion and Flame*, 156:1035–1045, 2009.