Block-Structured Adaptive Mesh Refinement

Lecture 3

- Incompressible AMR
- Linear solvers
- Multiphysics applications
  - LMC (Low Mach Number Combustion)
  - AMAR (Adaptive Mesh and Algorithm Refinement)

Incompressible Navier-Stokes equations

Recall the incompressible Navier-Stokes equations, with an additional advected/diffused scalar quantity s

$$U_t + U \cdot \nabla U + \nabla p = \epsilon \Delta U$$

$$s_t + \nabla \cdot sU = \kappa \Delta s$$

$$\nabla \cdot U = 0$$

Review of projection method

Fractional step scheme

Advection / Diffusion step:

$$\frac{U^* - U^n}{\Delta t} = -[U^{ADV} \cdot \nabla U]^{n+\frac{1}{2}} - \nabla p^{n-\frac{1}{2}} + \epsilon \Delta \frac{U^n + U^*}{2}$$

$$\frac{s^{n+1} - s^n}{\Delta t} + [\nabla \cdot sU]^{n+1/2} = \kappa \Delta \frac{s^n + s^{n+1}}{2}$$

Projection step:

Solve $Lp^{n+\frac{1}{2}} = DV \quad \text{where} \quad V = \frac{U^*}{\Delta t} + Gp^{n-\frac{1}{2}}$

and set

$$U^{n+1} = \Delta t(V - Gp^{n+\frac{1}{2}})$$

Algorithm components

- Construction of explicit hyperbolic advection terms for U and s
- Cell-centered elliptic solve to enforce constraint at half-time level
- Crank-Nicolson discretization of diffusion
- Nodal projection to enforce constraint

AMR for projection method

Goal: Combine basic methodology for solving hyperbolic, parabolic and elliptic PDE's on hierarchically refined grids to develop efficient adaptive projection algorithm

Design Issues: Desireable properties of the adaptive projection algorithm

- Second-order accurate in both space and time
- Use subcycling in time (i.e. $\Delta t^c = r\Delta t^f$)
- Conservative
- "Free-stream" preserving (i.e. constant fields stay constant)

Adaptive projection time step

Advance (level ℓ)

- Predict normal velocities
- Do MAC projection to define advection velocities
- Compute advection terms for U, s, etc., using upwind methodology
- Solve for diffusive transport using Crank-Nicholson discretization
- Perform nodal projection to enforce divergence constraint
- if $(\ell < \ell_{max})$
  - **Advance**$(\ell + 1)$ r times using Dirichlet boundary conditions from level ℓ at coarse / fine boundary
  - **Synchronize** levels ℓ and $\ell + 1$

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

- Coarse grid cells covered by fine grid cells don't have the most accurate data

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

- Coarse grid cells covered by fine grid cells don't have the most accurate data
  - Average down

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

- Coarse grid cells covered by fine grid cells don't have the most accurate data
  - Average down
- Fluxes (advective and diffusive) at coarse-fine interface are inconsistent

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

- Coarse grid cells covered by fine grid cells don't have the most accurate data
  - Average down
- Fluxes (advective and diffusive) at coarse-fine interface are inconsistent
  - Explicit reflux correction → **conservative**

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

- Coarse grid cells covered by fine grid cells don't have the most accurate data
  - Average down
- Fluxes (advective and diffusive) at coarse-fine interface are inconsistent
  - Explicit reflux correction $\rightarrow$ **conservative**
- Composite advection velocity (U^{ADV}) is not divergence-free on composite grid

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

- Coarse grid cells covered by fine grid cells don't have the most accurate data
  - Average down
- Fluxes (advective and diffusive) at coarse-fine interface are inconsistent
  - Explicit reflux correction → **conservative**
- Composite advection velocity (U^{ADV}) is not divergence-free on composite grid
  - MAC sync correction → **free-stream-preserving**

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

- Coarse grid cells covered by fine grid cells don't have the most accurate data
  - Average down
- Fluxes (advective and diffusive) at coarse-fine interface are inconsistent
  - Explicit reflux correction → **conservative**
- Composite advection velocity (U^{ADV}) is not divergence-free on composite grid
  - MAC sync correction → **free-stream-preserving**
- Reflux corrections and re-advection corrections have not been diffused.

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

- Coarse grid cells covered by fine grid cells don't have the most accurate data
  - Average down
- Fluxes (advective and diffusive) at coarse-fine interface are inconsistent
  - Explicit reflux correction $\rightarrow$ **conservative**
- Composite advection velocity (U^{ADV}) is not divergence-free on composite grid
  - MAC sync correction $\rightarrow$ **free-stream-preserving**
- Reflux corrections and re-advection corrections have not been diffused.
  - Diffusion correction $\rightarrow$ **stable for low** Re

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

- Coarse grid cells covered by fine grid cells don't have the most accurate data
  - Average down
- Fluxes (advective and diffusive) at coarse-fine interface are inconsistent
  - Explicit reflux correction $\rightarrow$ **conservative**
- Composite advection velocity (U^{ADV}) is not divergence-free on composite grid
  - MAC sync correction $\rightarrow$ **free-stream-preserving**
- Reflux corrections and re-advection corrections have not been diffused.
  - Diffusion correction $\rightarrow$ **stable for low** Re
- Composite solution was not divergence-free on composite grid
- Refluxing corrections are not divergence-free

Synchronization for IAMR

Errors result from calculating solution on coarse and fine levels independently.

- Coarse grid cells covered by fine grid cells don't have the most accurate data
  - Average down
- Fluxes (advective and diffusive) at coarse-fine interface are inconsistent
  - Explicit reflux correction $\rightarrow$ **conservative**
- Composite advection velocity (U^{ADV}) is not divergence-free on composite grid
  - MAC sync correction $\rightarrow$ **free-stream-preserving**
- Reflux corrections and re-advection corrections have not been diffused.
  - Diffusion correction $\rightarrow$ **stable for low** Re
- Composite solution was not divergence-free on composite grid
- Refluxing corrections are not divergence-free
  - Sync Projection $\rightarrow$ **Satisfy composite constraint, 2nd order accurate**

Explicit Reflux

This step is analogous to the hyperbolic case. Form

- $\delta \mathbf{F}_U = \mathbf{F}_U^f - \mathbf{F}_U^c$
- $\delta \mathbf{F}_s = \mathbf{F}_s^f - \mathbf{F}_s^c$

For $\kappa = 0$, use $\delta \mathbf{F}_s$ to correct the coarse cells immediately outside the union of fine grids.

$$s^{c,n+1} := s^{c,n+1} + \Delta t^c / \Delta x^c \delta \mathbf{F}_s$$

For $\kappa \neq 0$ (δF_s contains diffuive fluxes) form

$$s^{sync} = \Delta t^c / \Delta x^c \delta \mathbf{F}_s$$

For velocity we cannot update yet. Instead form

$$V^{sync} = \Delta t^c / \Delta x^c \delta \mathbf{F}_U$$

This corrections make the scheme **conservative.**

MAC Sync Correction

To compute advective fluxes we computed $U^{ADV,c}$ and $U^{ADV,f}$

In general, at coarse / fine boundary

$$U^{ADV,c} \neq \sum_t \sum_{faces} U^{ADV,f}$$

As a results, if we consider them as a composite $\overline{U}^{ADV,c-f}$ then

$$D^{MAC,c-f}\ \overline{U}^{ADV,c-f} \neq 0$$

In a composite sense, we did not advect with a divergence free field

To correct this mismatch we define

$$\delta U = \sum_t \sum_{faces} U^{ADV,f} - U^{ADV,c}$$

MAC Sync Correction (p2)

To account for the fact that the mismatch arose from δU we compute a correction U^{corr} to the composite advection velocity by solving

$$\begin{aligned} D^{MAC} G^{MAC} e^{corr} &= D^{MAC}(\delta U) \\ U^{corr} &= -G e^{corr} \end{aligned}$$

and explicitly re-advecting with this "correction velocity"

Namely, we compute $(U^{corr} \cdot \nabla) U$ $D^{MAC} F_s^{corr} = \nabla \cdot (U^{corr} s)$. These corrections are defined at *every* coarse grid cell and interpolated to finer grids

For velocity we still cannot update

$$V^{sync} := V^{sync} + (U^{corr} \cdot \nabla) U$$

If $\kappa = 0$ we can add $\Delta t D^{MAC} F_s^{corr}$ to s and it's synchronization is complete
If $\kappa \neq 0$ set

$$s^{sync} := s^{sync} + \Delta t D^{MAC} F_s^{corr}$$

MAC Sync Correction (p3)

What does this correction do?

Consider the case with $s = 1$ throughout the domain at t^n.

Then $s^{n+\frac{1}{2}} = 1$ at all edges at $t^{n+\frac{1}{2}}$, and the correct solution is $s^{c,n+1} = s^{f,n+1} = 1$.

But explicit refluxing would replace the value $s^{c,n+1} = 1$ (computed during the coarse advance) by $s^{c,n+1} = 1 + \Delta t^c / \Delta x^c \delta \mathbf{F}_s$.

As a result, in making the scheme conservative we have introduced spurious signal in s

The MAC sync correction removes this signal:

$$s^{n+1} := \overline{s}^{n+1} + \Delta t^c / \Delta x^c \delta \mathbf{F}_s + \Delta t D^{MAC} F_s^{corr} \equiv 1$$

in our example

This makes the scheme **free-stream preserving**

Diffusing the correction

When viscosity/diffusivity is non-zero:

Recall the parabolic synchronization step from the 1-d example.

Instead of adding the refluxing corrections directly to the solution, we must solve

$$(I - \frac{\epsilon \Delta t}{2} \Delta^h) e_U^{n+1} = \frac{\Delta t^c}{\Delta x_c} (\delta \mathbf{F}_U) + \Delta t^c D^{MAC} (\mathbf{F}_U^{corr}) (U^{corr} \cdot \nabla) U \equiv V^{sync}$$

$$(I - \frac{\kappa \Delta t}{2} \Delta^h) e_s^{n+1} = \frac{\Delta t^c}{\Delta x_c} (\delta \mathbf{F}_s) + \Delta t^c D^{MAC} (\mathbf{F}_s^{corr}) \equiv s^{sync}$$

This keeps the scheme **stable** at low Re.

Then

$$s^{n+1} := s^{n+1} + e_s^{n+1}$$

Here we perform a coarse level solve and interpolate to finer grids

We still are not ready to update U

Sync Projection

Remaining problem:

- Composite solution was not divergence-free on composite grid
- Corrections due to refluxing, etc, are not divergence-free

These mismatches can be combined in the solution of

$$DG\phi^{SP} = RHS_{elliptic} + RHS_{reflux}$$

$RHS_{elliptic} =$ contribution to $D^c(U_t - G\phi)^c$ coming from coarse side
$+\sum_{time}$ contribution from $D^f(U_t - G\phi)^f$ coming from fine side

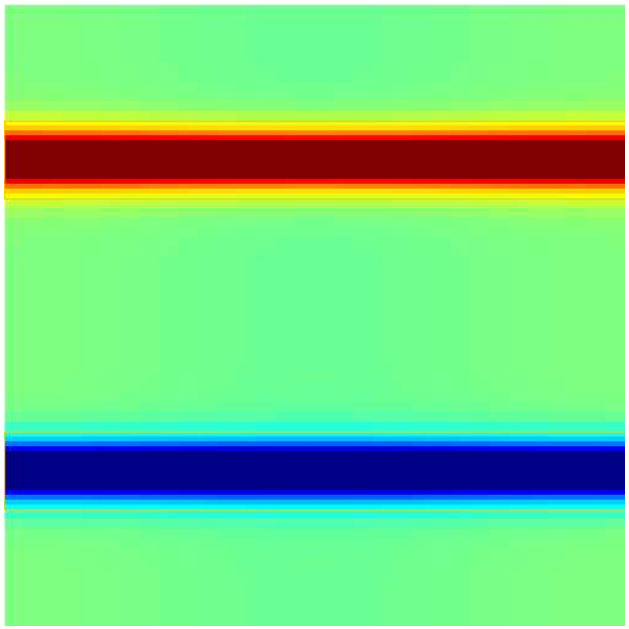
and is non-zero only at coarse nodes on coarse-fine interface

$RHS_{reflux} = D(e_U^{n+1})$ is defined at all nodes with contributions from all coarse grid cells

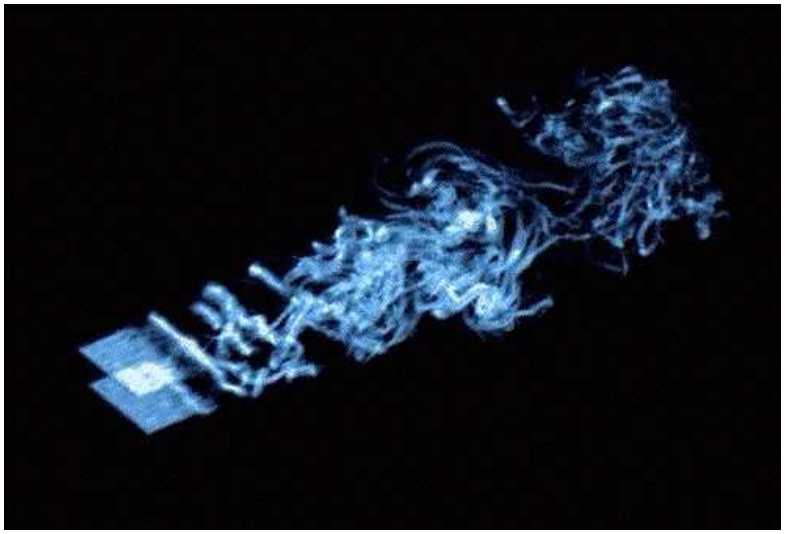
Then

$$p^{n+1/2,c} := p^{n+1/2,c} + \phi^{SP} \qquad U^{n+1,c} := U^{n+1,c} - G\phi^{SP}$$
$$p^{n+3/4,f} := p^{n+3/4,f} + \phi^{SP} \qquad U^{n+1,f} := U^{n+1,f} - G\phi^{SP}$$

Examples



2D shear layer



3D planar jet

Additional software support for IAMR

What operations must be supported for IAMR that did not exist for hyperbolic AMR?

Additional software support for IAMR

What operations must be supported for IAMR that did not exist for hyperbolic AMR?

Linear solvers for parabolic and elliptic equations.

Additional software support for IAMR

What operations must be supported for IAMR that did not exist for hyperbolic AMR?

Linear solvers for parabolic and elliptic equations.

- Single-level solvers
- Multi-level solvers

Additional software support for IAMR

What operations must be supported for IAMR that did not exist for hyperbolic AMR?

Linear solvers for parabolic and elliptic equations.

- Single-level solvers
- Multi-level solvers

- Cell-centered (for MAC and diffusive solves)
- Node-centered (for projection)

Linear Solvers

- Even for single-level solvers, the distribution of points on multiple grids results in irregular data layout.
- Off-the-shelf and black box methods have been inadequate
- Iterative methods have proven to be the most efficient solvers.
- Multigrid methods, in particular, are desirable
  - Multigrid naturally respects the AMR hierarchy present in multi-level solves
  - Typically require many fewer iterations than other iterative methods (conjugate gradient)
  - $O(N \log N)$ as opposed to $O(N^2)$

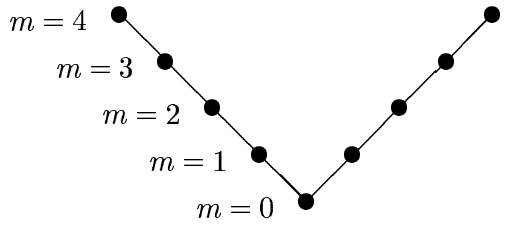
Multigrid : Introduction

Multigrid is an iterative method in which a multigrid hierarchy of successively coarser (by a factor of 2) levels is created and each iteration, starting at the level where the problem is specified, recursively calls

MG(m)

- smooth the data at level m
- if m is not the coarsest level
  - coarsen the residual data to level $m-1$
  - call MG($m-1$)
  - interpolate the corrected coarse data back to level m
- smooth the data at level m

This recursive coarsening to the coarsest level then interpolation back to the finest level is known as a V-cycle.



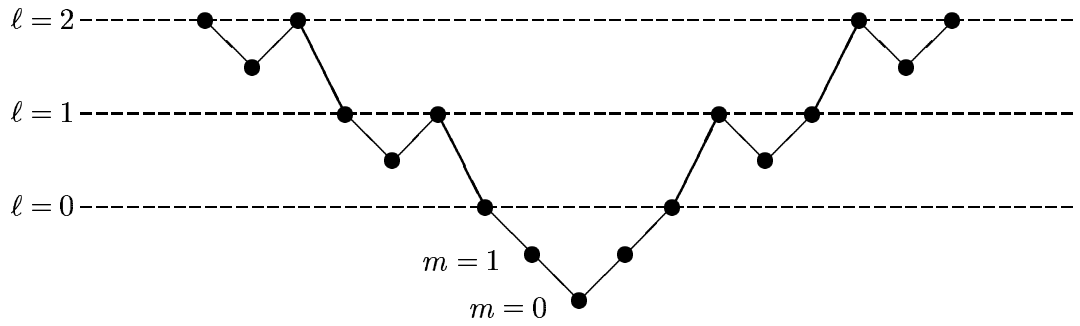
Multigrid for AMR

We distinguish between AMR levels, where we require a solution to the problem, and multigrid levels, which exist only to facilitate the solution of the linear system.

Dotted lines show AMR levels, other levels are used only by multigrid.

Note that multigrid levels can exist between AMR levels if $r = 4$.

Multiple-AMR-level V-cycle

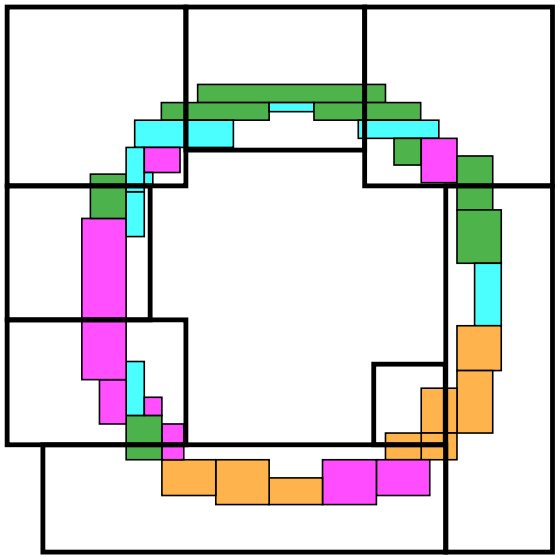


Multigrid: Implementation Issues

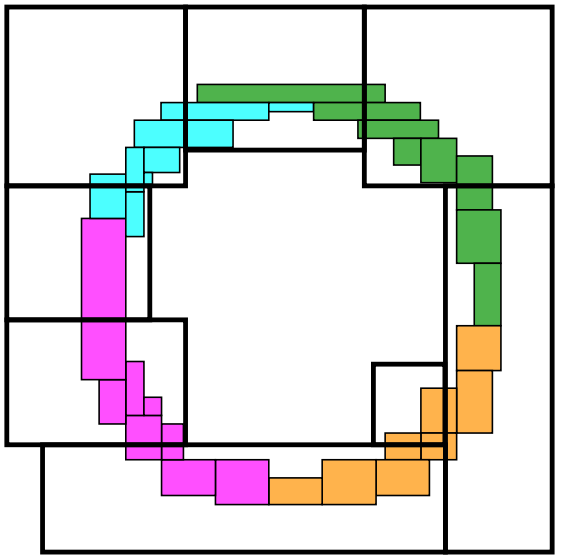
- Multigrid operations and AMR operations use comparable inter-level operations, such as interpolation and restriction.
- Because a single solve requires multiple iterations and each iteration requires significant inter-grid and inter-level communication, multigrid methods are problematic in parallel computing environments: naive implementations result in excessive message passing.
  - Specialized restriction and interpolation operations are used, since there is a one-to-one correspondence between fine grids and coarse grids;
  - the coarsening of each fine grid can be put on the same processor as the fine grid itself, resulting in locality for restriction and interpolation.
- For single level solves, the degree of coarsening that can be achieved within a multigrid V-cycle is limited by the smallest grid. Overall efficiency dictates that grids for IAMR calculations be divisible by multiple factors of 2. This can be achieved using a *blocking-factor* requirement during regridding.

Load Balance for Data Locality

- Recall that a *Knapsack* algorithm can be used for dynamic load-balancing in AMR calculations
  - effectively balances computational work across processors
  - often results in excessive off-processor communications
- Can improve communication while maintaining load balance
  - grids of the same size on different processors are exchanged if communication is improved.



Initial grid distribution



Grid distribution for data locality

Multiphysics applications

- Shock physics
- Incompressible flow
- MHD
- Radiation Hydrodynamics
- Compressible Navier Stokes
- Low mach number models
  - Combustion
  - Nuclear flames
  - Atmospheric flows
- Biology
- Multiphase flow
- Porous media flow
- Hybrid methods

Low Mach Number Combustion

Low Mach number model, $M = U/c \ll 1$ (Rehm & Baum 1978, Majda & Sethian 1985)

Start with the compressible Navier-Stokes equations for multicomponent reacting flow, and expand in the Mach number, $M = U/c$.

Asymptotic analysis shows that:

$$p(\vec{x}, t) = p_0(t) + \pi(\vec{x}, t) \quad \text{where} \quad \pi/p_0 \sim \mathcal{O}(M^2)$$

- p_0 does not affect local dynamics, π does not affect thermodynamics
- For open containers p_0 is constant
- Acoustic waves analytically removed (or, have been "relaxed" away)

Low Mach number combustion

$$\textbf{Momentum} \quad \rho\frac{DU}{Dt} = -\nabla\pi + \nabla\cdot\left[\mu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} - \frac{2}{3}\delta_{ij}\nabla\cdot U\right)\right]$$

$$\textbf{Species} \quad \frac{\partial(\rho Y_m)}{\partial t} + \nabla\cdot(\rho U Y_m) = \nabla\cdot(\rho D_m \nabla Y_m) + \dot{\omega}_m$$

$$\textbf{Mass} \quad \frac{\partial\rho}{\partial t} + \nabla\cdot(\rho U) = 0$$

$$\textbf{Energy} \quad \frac{\partial\rho h}{\partial t} + \nabla\cdot\left(\rho h\vec{U}\right) = \nabla\cdot(\lambda\nabla T) + \sum_m \nabla\cdot(\rho h_m D_m \nabla Y_m)$$

Equation of state $p_0 = \rho\mathcal{R}T\sum_m \frac{Y_m}{W_m}$

System contains four evolution equations for U, Y_m, ρ, h, with a constraint given by the EOS.

Constraint for reacting flows

We differentiate the EOS along particle paths and use the evolution equations for ρ and T to define a constraint on the velocity:

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

Constraint expresses compressibiity arising from thermal processes

Variable coefficient projection

Generalized vector field decomposition

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

where $\nabla \cdot U_d = 0$ and $U \cdot n = 0$ on the boundary

Then U_d and $\frac{1}{\rho}\nabla\phi$ are orthogonal in a density weighted space.

$$\int \frac{1}{\rho}\nabla\phi \cdot U \ \rho \ dx = 0$$

Defines a projection $\mathbf{P}_\rho = I - \frac{1}{\rho}\nabla((\nabla \cdot \frac{1}{\rho}\nabla)^{-1})\nabla\cdot$ such that $\mathbf{P}_\rho V = U_d$.

$\mathbf{P}_\rho$ is idempotent and $||\mathbf{P}_\rho|| = 1$

Variable coefficient projection method

We can use this projection to define a projection scheme for the variable density system

$$\rho_t + \nabla \cdot \rho U = 0$$

$$U_t + U \cdot \nabla U + \frac{1}{\rho} \nabla \pi = 0$$

$$\nabla \cdot U = 0$$

Advection step

$$\rho^{n+1} = \rho^n - \Delta t \nabla \cdot \rho U$$

$$U^* = U^n - \Delta t \, U \nabla \cdot U \frac{1}{\rho} - \Delta t \nabla \pi^{n-1/2}$$

Projection step

$$U^{n+1} = \mathbf{P}_\rho U^*$$

Recasts system as initial value problem

$$U_t + \mathbf{P}_\rho (U \cdot \nabla U) = 0$$

Projection method with sources

Low Mach number models introduce an inhomogeneous constraint

$$U_t + U \cdot \nabla U + \frac{1}{\rho}\nabla\pi = \frac{1}{\rho}F_U$$

$$\nabla \cdot U = S$$

Advection step defines an intermediate velocity, U^*

We want decomposition

$$U^* = U^{n+1} + \frac{1}{\rho}\nabla\phi \quad \text{with} \quad \nabla \cdot \mathrm{U}^{\mathrm{n+1}} = \mathrm{S}$$

Projection step

$$U^{n+1} = U_d + \nabla\xi$$

where

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

$$U^{n+1} = \mathbf{P}_\rho(U^* - \nabla\xi) + \nabla\xi$$

Fractional Step Approach

1. Advance velocity from $\vec{U}^n$ to $\vec{U}^{n+1,*}$ using explicit advection terms, Crank-Nicolson diffusion, and a lagged pressure gradient.
2. Update the species and enthalpy equations
3. Use the updated values to compute S^{n+1}
4. Decompose $\vec{U}^{n+1,*}$ to extract the component satisfying the divergence constraint.

This decomposition is achieved by solving

$$\nabla \cdot \left(\frac{1}{\rho} \nabla \phi \right) = \nabla \cdot \left(\frac{\vec{U}^{n+1,*}}{\Delta t} + \frac{1}{\rho} \nabla \pi^{n-1/2} \right) - S^{n+1}$$

for ϕ, and setting $\pi^{n+1/2} = \phi$ and

$$\vec{U}^{n+1} = \vec{U}^{n+1,*} - \frac{\Delta t}{\rho} \nabla \phi$$

Exploits linearity to represent the compressible component of the velocity

Species equation advance

$$\frac{\partial(\rho Y_m)}{\partial t} + \nabla \cdot (\rho U Y_m) = \nabla \cdot (\rho D_m \nabla Y_m) + \dot{\omega}_m$$

$$\frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho U h) = \nabla \cdot (\lambda \nabla T) + \sum_m \nabla \cdot (\rho h_m D_m \nabla Y_m)$$

Stiff kinetics relative to fluid dynamical time scales

Operator split approach

- Chemistry $\Rightarrow \Delta t/2$
- Advection – Diffusion $\Rightarrow \Delta t$
- Chemistry $\Rightarrow \Delta t/2$

Properties of the methodology

Overall operator-split projection formulation is 2^{nd}-order accurate in space and time.

Godunov-type discretization of advection terms provides a robust 2^{nd}-order accurate treatment of advective transport.

Formulation conserves species, mass and energy.

Equation of state is only approximately satisfied

$$p_o \neq \rho RT \sum_m \frac{Y_m}{W_m}$$

but modified constraint minimizes drift from equation of state.

Model problems

2-D Vortex flame interactions

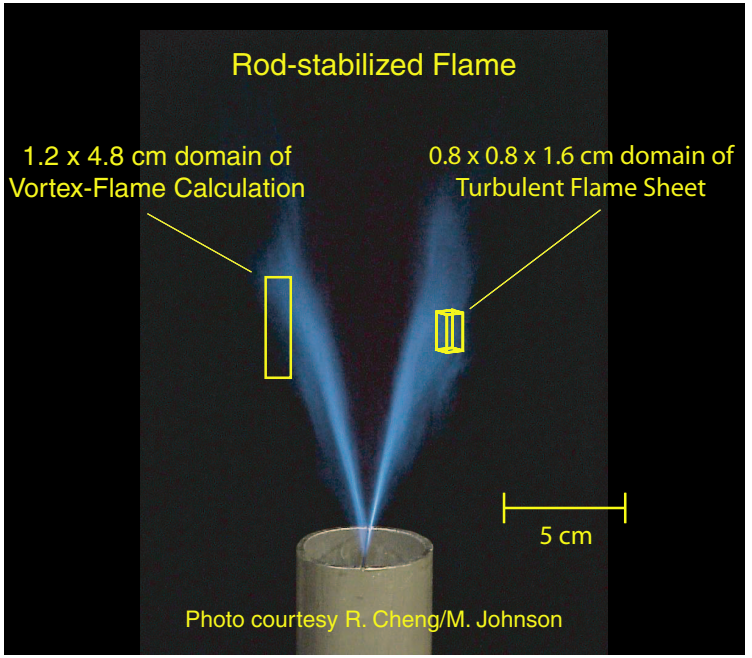
(28^{th} International Combustion Sympsium, 2000)

- 1.2×4.8 mm domain
- 32 species, 177 reactions

3-D Turbulent flame sheet

(29^{th} International Combustion Sympsium, 2002)

- $.8 \times .8 \times 1.6$ cm domain
- 20 species, 84 reactions

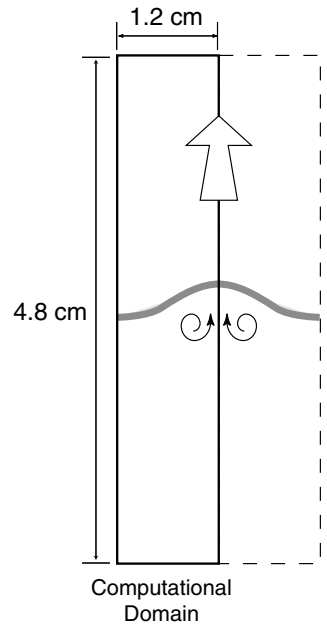


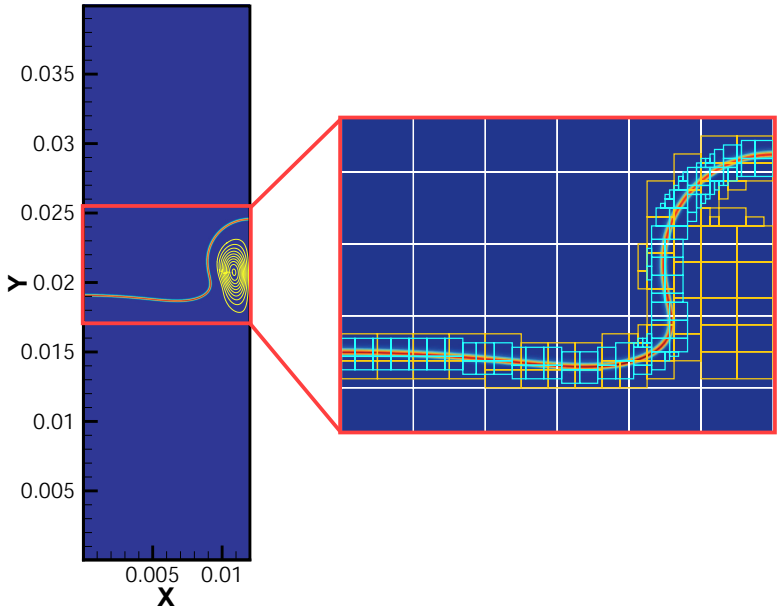
Laboratory-scale V-flame

(19^{th} International Colloquium on the Dynamics of Explosions and Reactive Systems, 2003)

- $12 \times 12 \times 12$ cm domain
- 20 species, 84 reactions

Vortex flame interaction

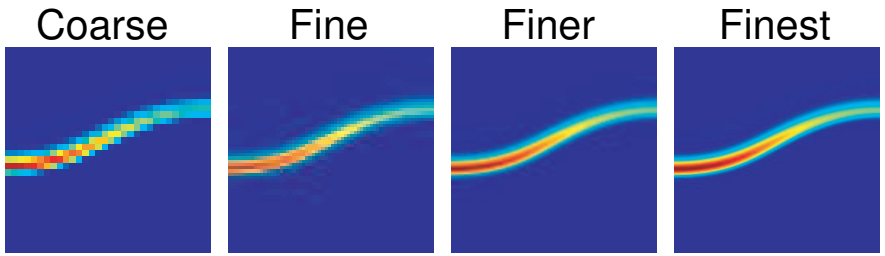


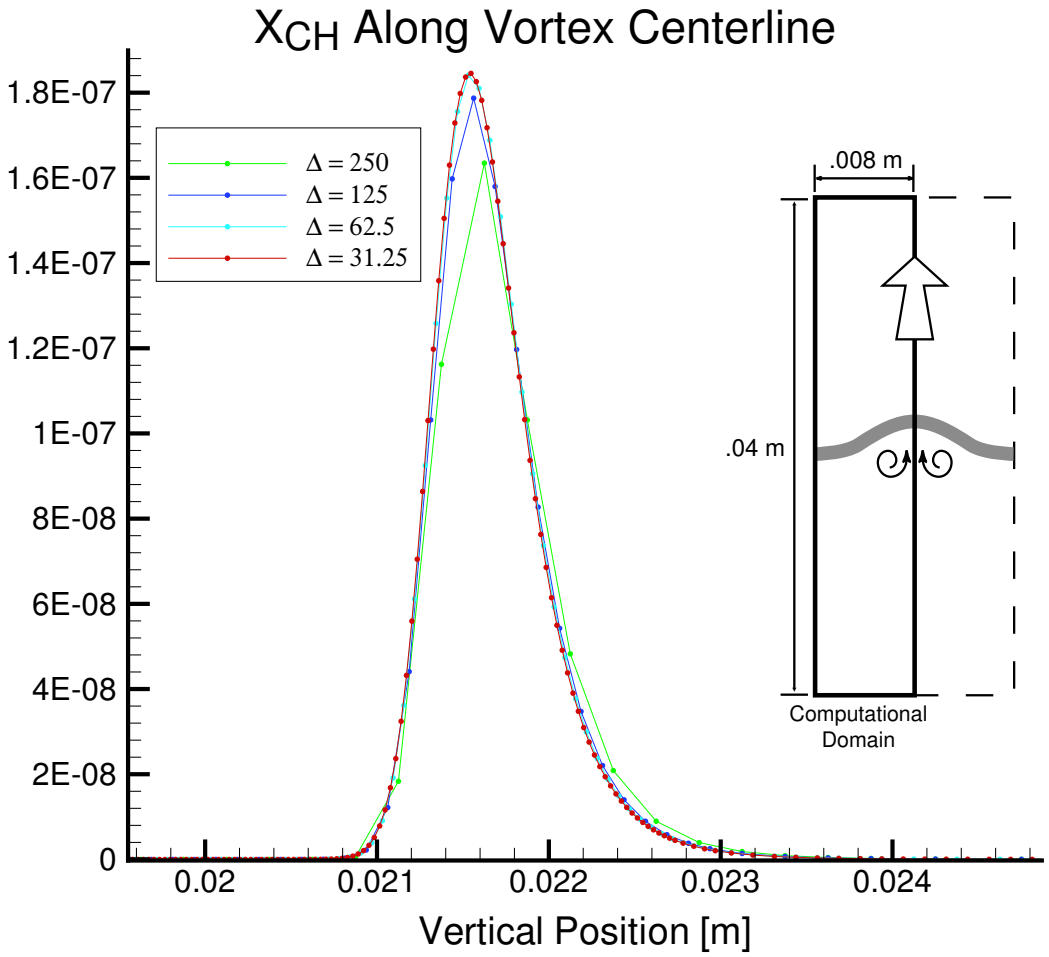


- Fuel: N_2-diluted CH_4/air
  - $\phi = 0.8$
- Mech: GRI-Mech 1.2
  - 32 species, 177 reactions

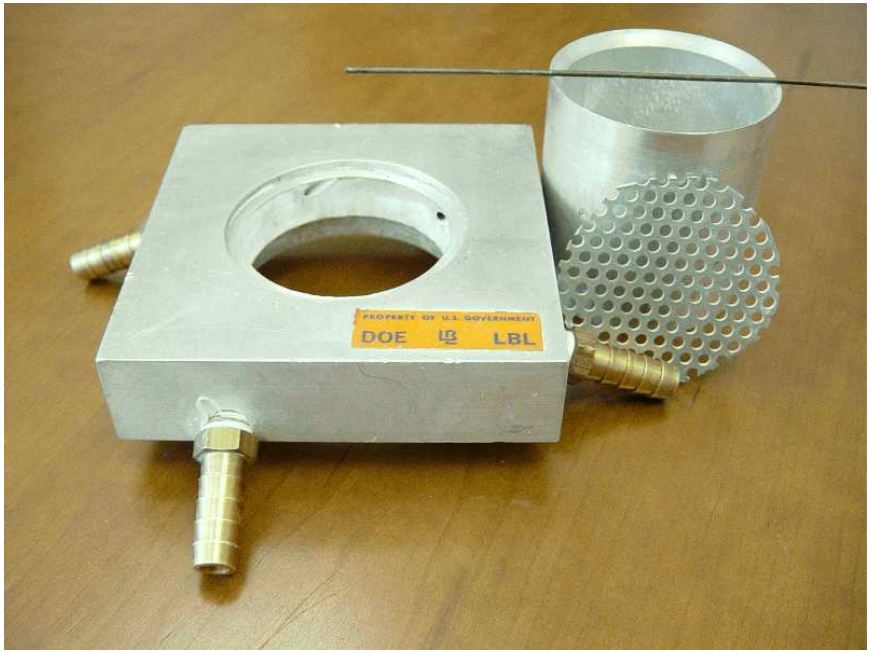
Representative adaptive solution

Convergence Behavior

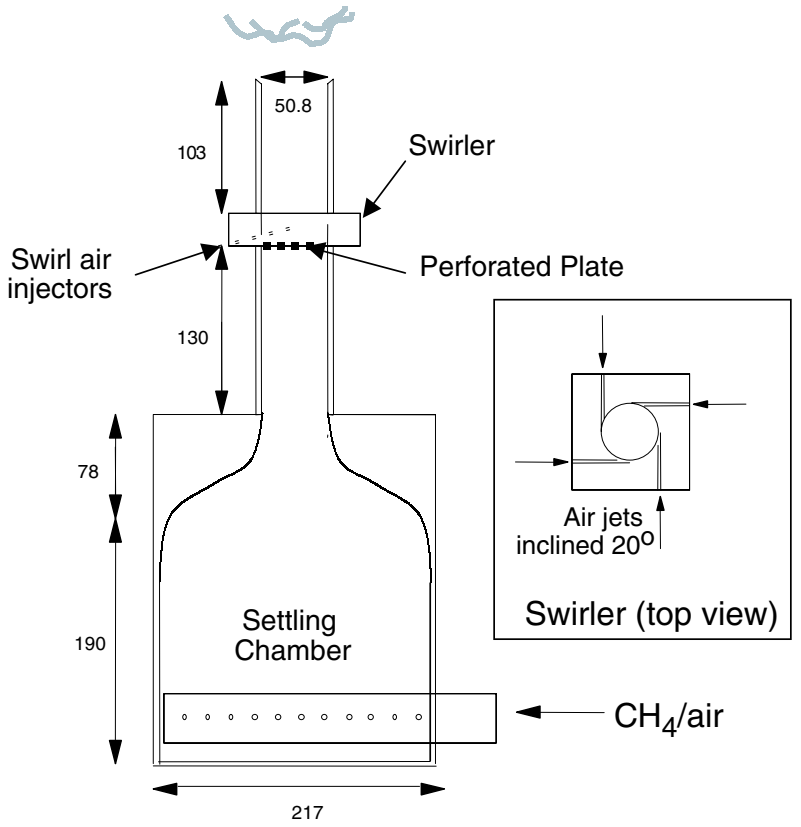




Configuration



Burner assembly

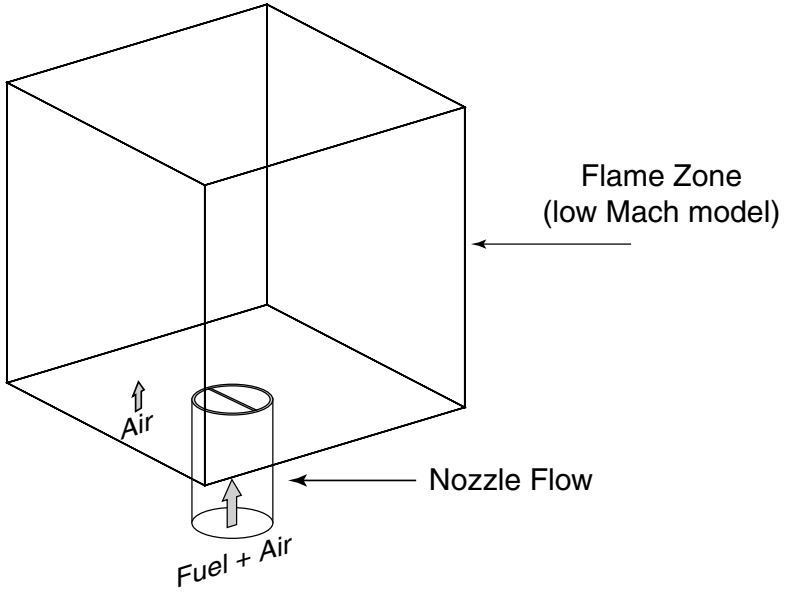


Experiment schematic

- V-flame ($\dot{m}_{air} \equiv 0$): rod $\sim$ 1 mm
- Turbulence plate: 3 mm holes on 4.8 mm center

V-flame Setup

Strategy - Treat nozzle exit as inflow boundary condition for combustion simulation



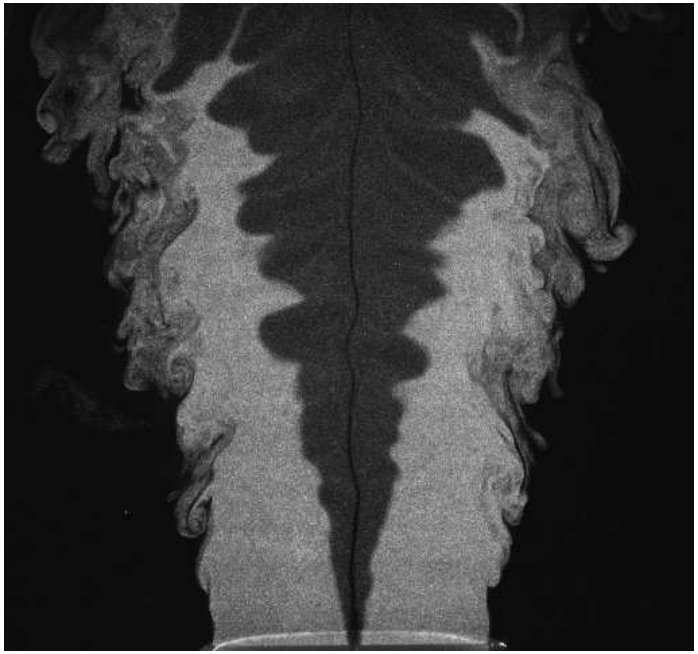
- 12cm x 12cm x 12cm domain
- DRM-19: 20 species, 84 reactions
- Mixture model for differential diffusion

Inflow characteristics

- Mean flow
  - 3 m/s mean inflow
  - Boundary layer profile at edge
  - Noflow condition to model rod
  - Weak co-flow air
- Turbulent fluctuations
  - $\ell_t = 3.5mm$, $u' = 0.18m/sec$
  - Slightly anisotropic (axial > radial)
  - Estimated $\eta = 220\mu m$
- Use synthetic turbulence field shaped to match nozzle flow characteristics to specify turbulent fluctuations

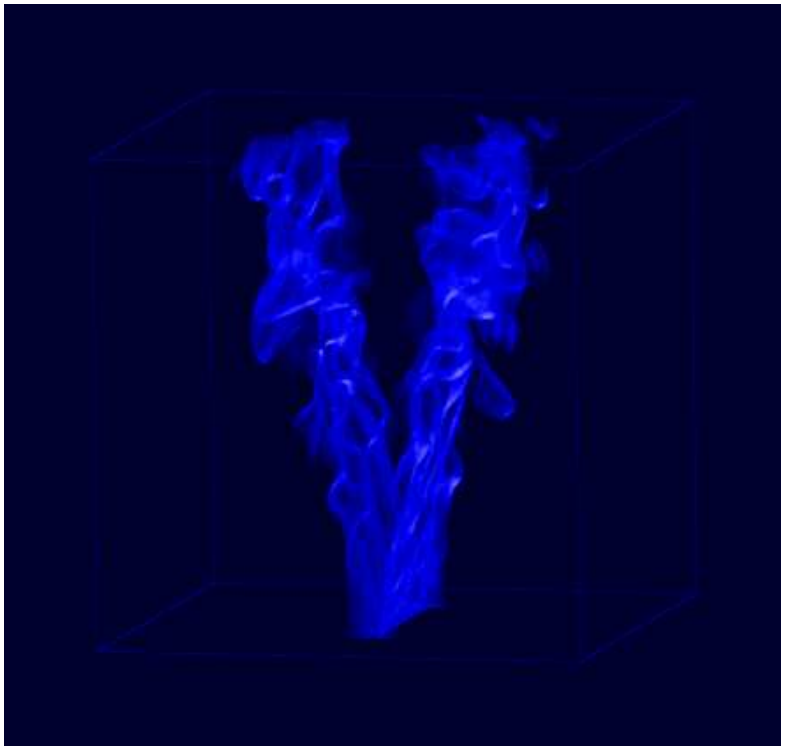
Results: Computation vs. Experiment

CH_4 from simulation



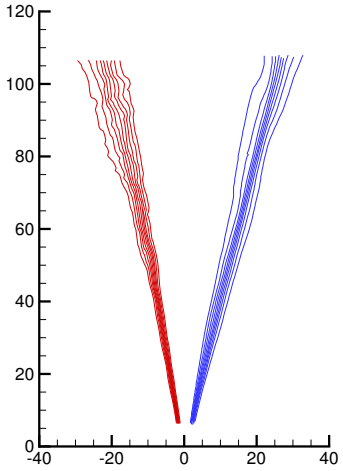
Single image from experimental PIV

Flame Surface

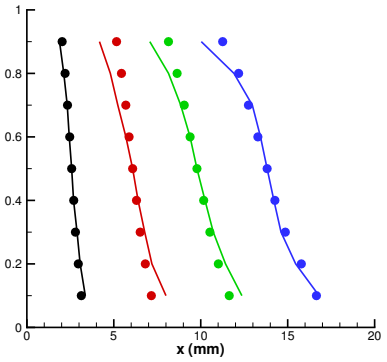


Instantaneous flame surface

Flame brush



Internal flame brush structure



Nuclear flames

Characterization of stellar material

Timmes equation of state provides:

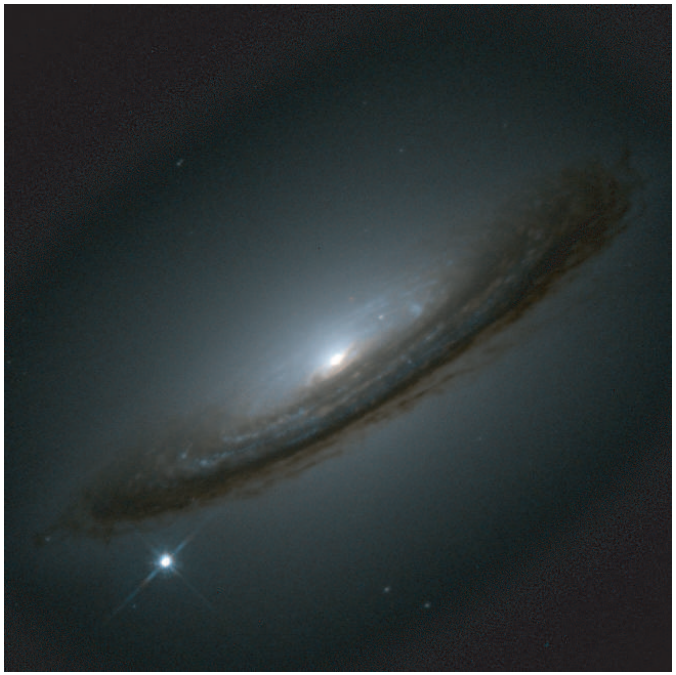
$$e(\rho, T, X_k) = e_{ele} + e_{rad} + e_{ion}$$

$$e_{ele} = \text{fermi}$$
$$e_{rad} = aT^4/\rho$$
$$e_{ion} = \frac{3kT}{2m_p} \sum_m X_k/A_m$$

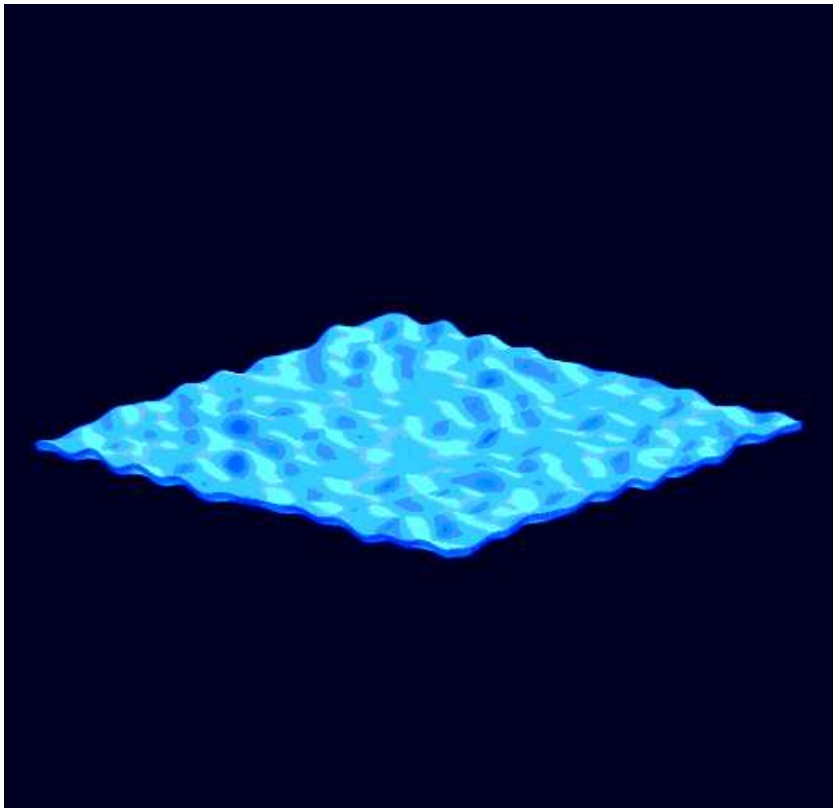
$$p(\rho, T, X_k) = p_{ele} + p_{rad} + p_{ion}$$

$$p_{ele} = \text{fermi}$$
$$p_{rad} = aT^4/3$$
$$p_{ion} = \frac{\rho kT}{m_p} \sum_m X_k/A_m$$

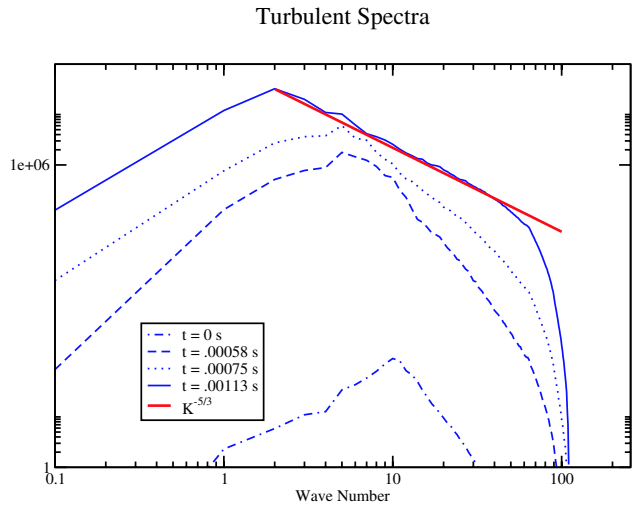
Type Ia supernovae



3D Rayleigh Taylor



Flame surface



Multi-scale fluid flows

Most computations of fluid flows use a continuum representation (density, pressure, etc.) for the fluid.

Dynamics described by set of partial differential equations (PDEs).

Well-established numerical methods (finite difference, finite elements, etc.) for solving these PDEs.

The hydrodynamic PDEs are accurate over a broad range of length and time scales.

But at some scales the continuum representation breaks down and more physics is needed

When is the continuum description of a gas no accurate?

- Knudsen number = Mean Free Path / System Length
- $Kn < 0.1$ continuum description is not accurate
- Discreteness of collisions and fluctuations are important
  - Rarefied gases
  - Low-pressure manufacturing
  - Micro-scale flows

Hybrid methods

To capture Knudsen effects and fluctuations we need a particle description of the flow; however, these types of method are computationally expsensive

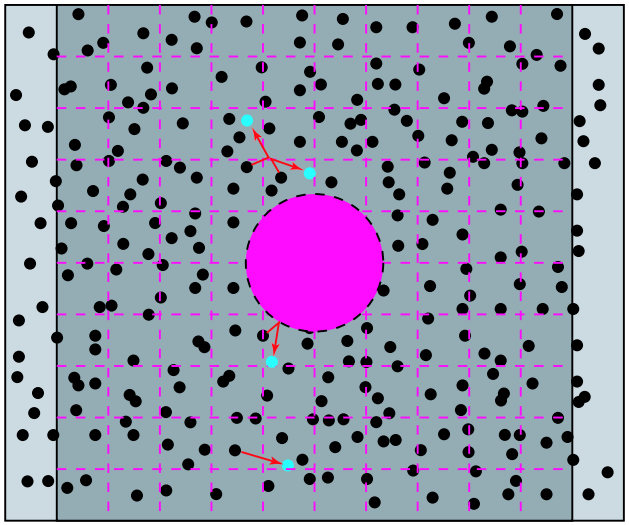
Can we hybridize a particle method with a continuum solver

- AMR provides a framework for such a coupling
  - AMR for fluids except change to a particle description at the finest level of the heirarchy
- Use basic AMR design paradigm for development of a hybrid method
  - How to integrate a level
  - How to synchronize levels

Discrete Simulation Monte Carlo (DSMC) is the dominant numerical method for molecular simulations of dilute gases

DSMC

- Initialize system with particles
- Loop over time steps
  - Create particles at open boundaries
  - Move all the particles
  - Process particle/boundary interactions
  - Sort particles into cells
  - Select and execute random collisions
  - Sample statistical values



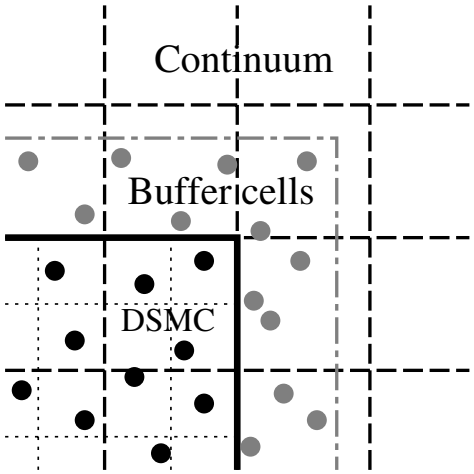
Adaptive mesh and algorithm refinement

Continuum solver – Compressible Navier Stokes

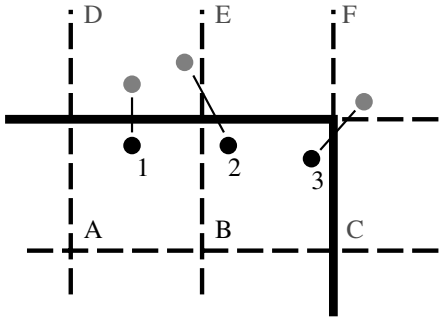
- Unsplit Godunov advection
- Crank-Nicolson diffusion (nonlinear multigrid)

Algorithm – 2 level

- Advance continuum
  - $F_T = F_A + F_D$ at DSMC boundary
- Advance DMSC region
  - Interpolation – Sampling from Chapman-Enskog distribution
  - Fluxes are given by particles crossing boundary of DSMC region
- Synchronize
  - Average down – moments
  - Reflux $\delta F = -\Delta t A F_T + \sum_p F_p$
  - Nonlinear diffusion to distribute reflux δU
  - δU corrects particles to preserve moments



DSMC boundary conditions



DSMC flux

Example

Flow past a microscopic sphere

- $M_a = 1$, $K_n = 0.2$
- Sphere radius 312 nm
- Domain 8000 nm in each direction
- Base grid 32^3
- One continuum level of refinment
- DSMC around sphere
  - 4×10^5 particles
  - DSMC region $< 0.2\%$ of domain