Numerical simulation of reacting flows using the unstructured adaptive mesh refinement code HAMISH

Stewart Cant, Jian Fang, Umair Ahmed, Girish Nivarti, Charles Moulinec, David Emerson, Nilanjan Chakraborty

UNIVERSITY OF CAMBRIDGE
DEPARTMENT OF ENGINEERING
Science & Technology Facilities Council
Newcastle University

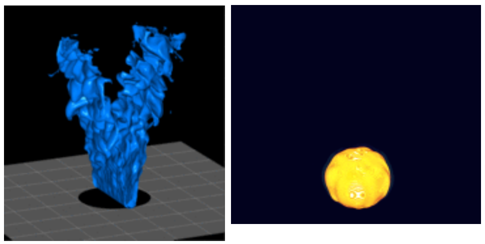
Outline

- Background
- Introduction to the HAMISH Code
- Code Tests
  - 1-D planar flame
  - 1-D Head-on quench
  - 2-D flame propagation
  - 2-D and 3-D channel flow
  - 2-D thermal conduction
  - 2-D and 3-D Rayleigh-Taylor instability
  - 3-D Taylor-Green vortex
  - 3-D isotropic decaying turbulence
- Scalability and Code Profiling
- Summary and Perspectives

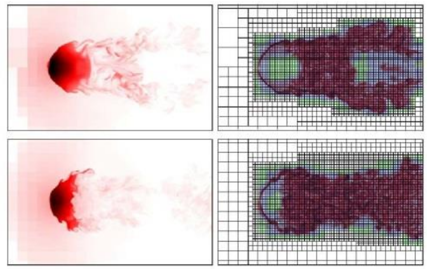
Background

- **Adaptive Mesh Refinement**
  - Dynamic adaption of the mesh based on the solution
  - Local in space and time
- **Advantages of AMR**
  - Higher accuracy and lower cost compared with a static mesh
  - CPU time and memory savings
  - Full control of the local mesh resolution
  - More detailed physics for the same number of cells
- **Main Applications**
  - Problems with large dynamic range of scales
  - Flames, two-phase flow, boundary layers, shock waves

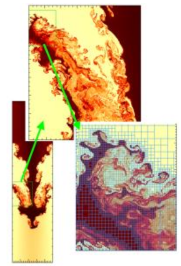
AMR in practice



Flames (Boxlib)

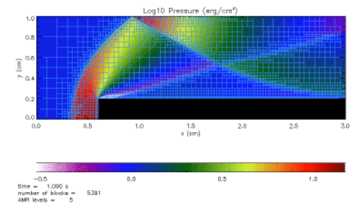


R-T instability (ENZO Code)

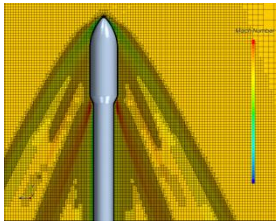


(FLASHCode)

Problems with interfaces

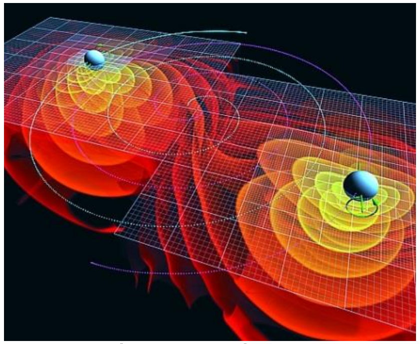


Moving Shock-Wave (PARAMESH)

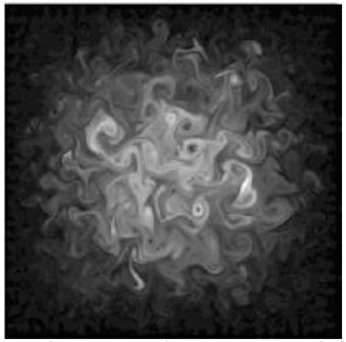


Supersonic Vehicle

Problems with discontinuities

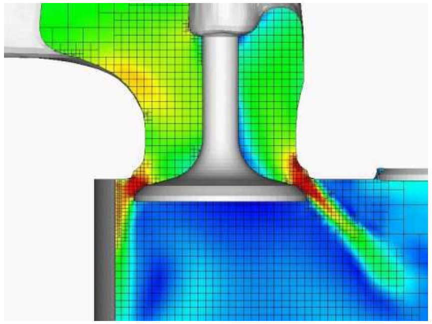


Computing Cosmic Cataclysms

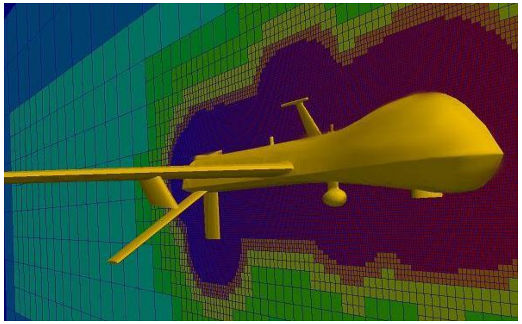


Turbulence (FLASH Code)

Problems with great variety of scales

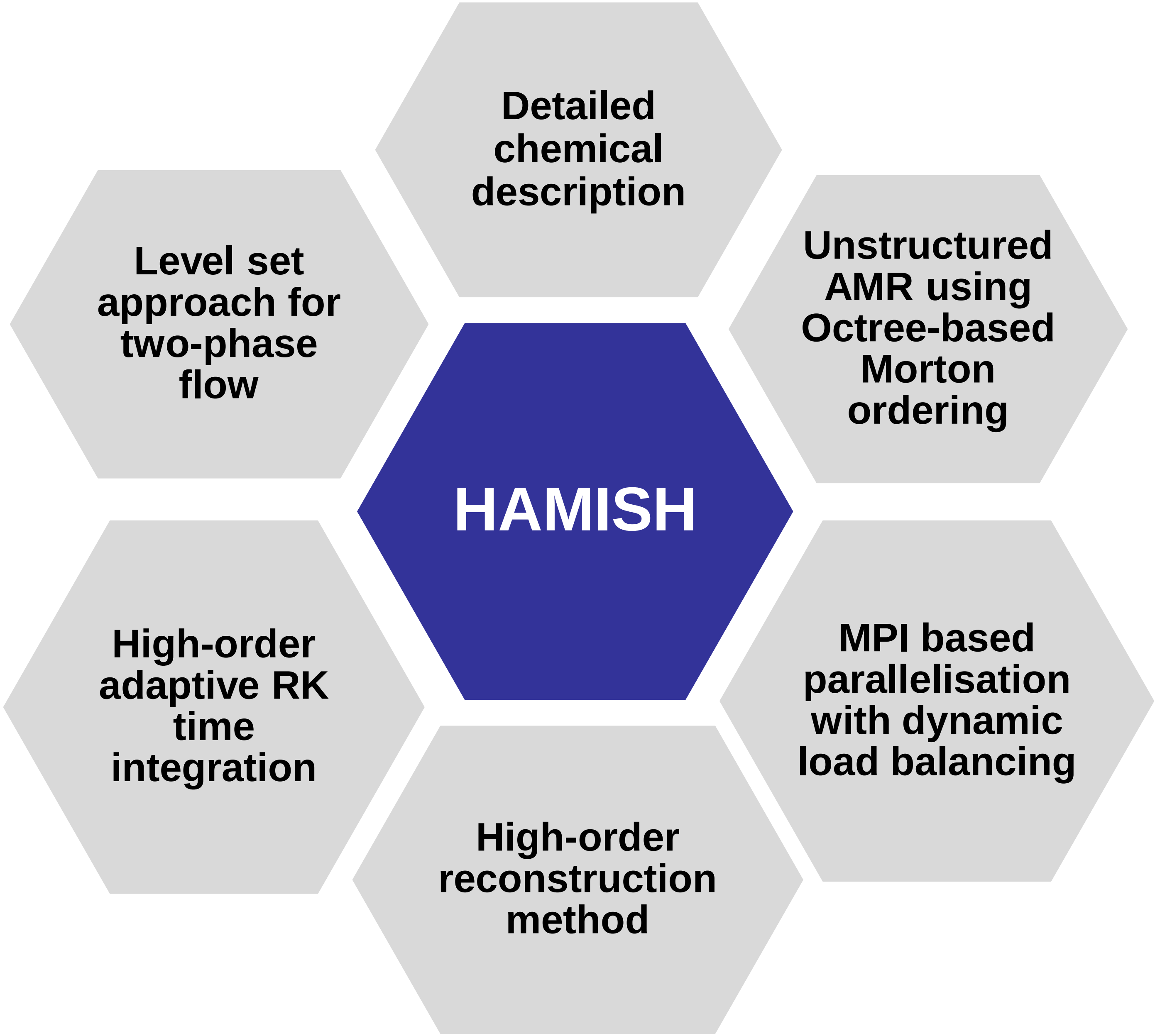


Engine Combustion



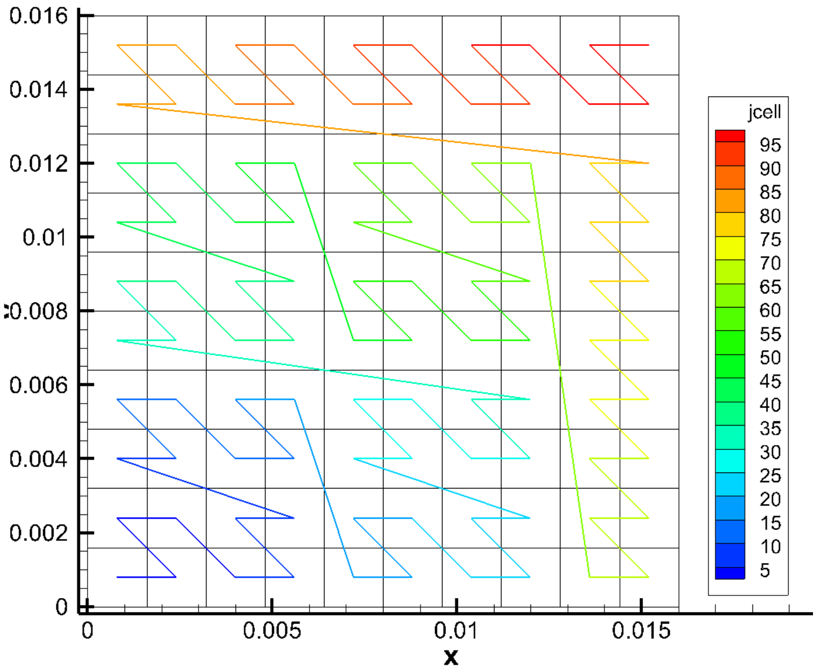
Drone

Problems with complex geometries


Detailed chemical description
Level set approach for two-phase flow
Unstructured AMR using Octree-based Morton ordering
HAMISH
High-order adaptive RK time integration
MPI based parallelisation with dynamic load balancing
High-order reconstruction method

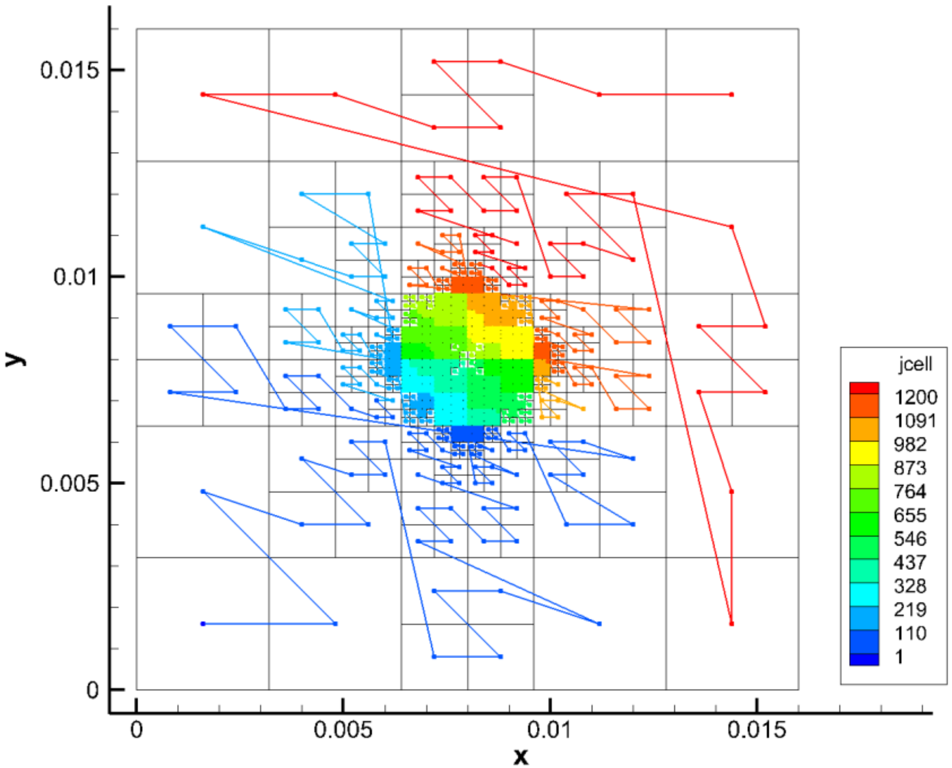
AMR in HAMISH

- Cartesian unit-cell mesh - unstructured

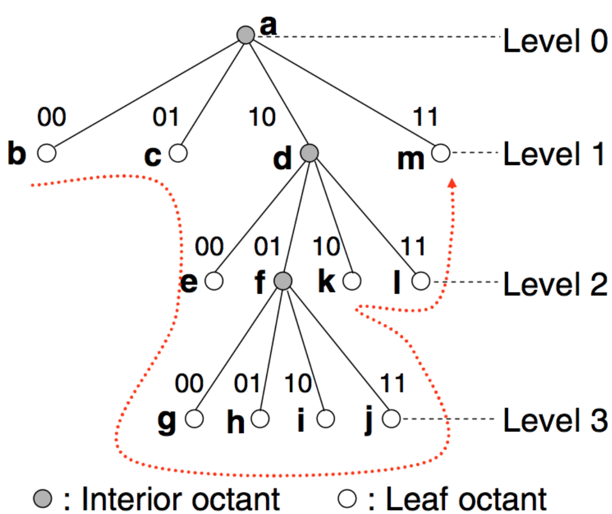


Morton ordering

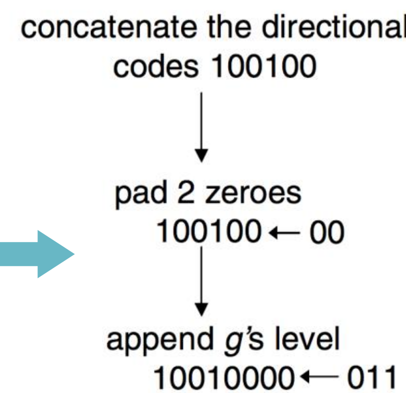
AMR cell heirarchy



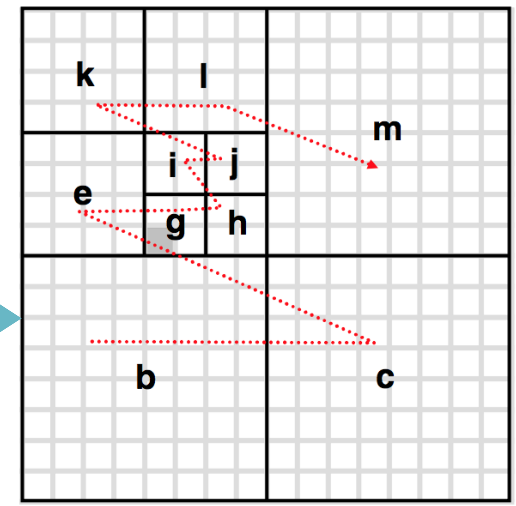
Data structures in HAMISH



(Quad)tree



Spatial index
(Morton code)



Space-filling curve

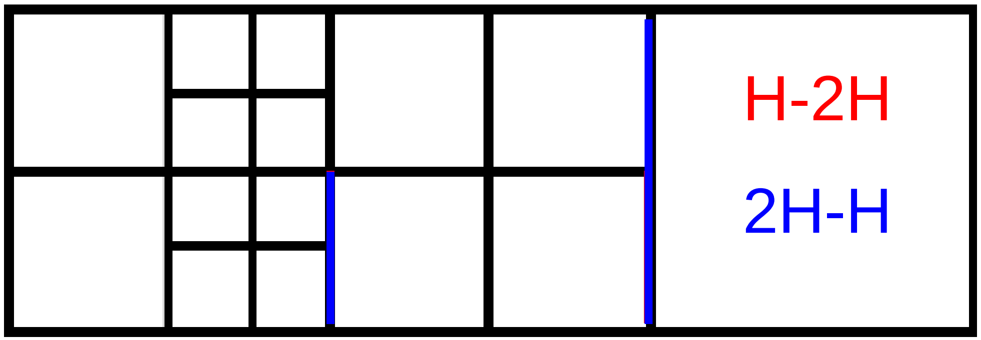
Refinement criterion based on the Euclidean norm of the local Laplacian
Tree balancing ensures that (at most) h-2h transitions exist

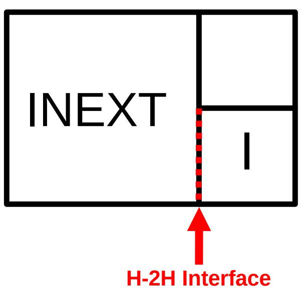
Partition Interval Table stores the highest local Morton code on each processor

Ghattas et al (2006)

Conservation of fluxes

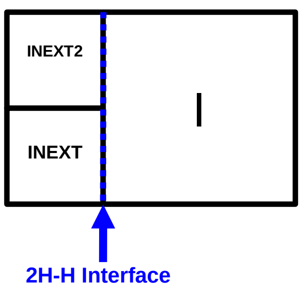
- **Flux calculation (linear scheme: 2nd order)**





$$f_{Interface} = \frac{1}{2}\left(f_I + f_{INEXT}\right)$$

$$\left.\frac{\partial f}{\partial x}\right|_{Interface} = \frac{\left(f_I - f_{INEXT}\right)}{0.5\left(L_I + L_{INEXT}\right)}$$



$$f_{Interface} = \frac{1}{2}\left(f_I + 0.5\left(f_{INEXT} + f_{INEXT2}\right)\right)$$

$$\left.\frac{\partial f}{\partial x}\right|_{Interface} = \frac{\left(f_I - 0.5\left(f_{INEXT} + f_{INEXT2}\right)\right)}{0.5\left(L_I + L_{INEXT}\right)}$$

Conservation is ensured

RENO scheme

Arbitrarily high-order reconstruction scheme

Solution is reconstructed within each cell using polynomial basis functions ϕ

$$u(x,y,z) = \bar{u}_0 + \sum_{k=1}^{K} a_k^{(u)} \phi_k(x,y,z) \qquad \phi_k = \psi_k - \int_{-1/2}^{1/2}\int_{-1/2}^{1/2}\int_{-1/2}^{1/2} \psi_k dxdydz$$

Fourth order sweet-spot: monomials ψ are:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
x	x^2	x^3	y	yx	y^2	yx^2	y^2x	y^3	z	zx	zy	z^2	zx^2	zxy	z^2x	zy^2	z^2y	z^3

Integrate over a cell:

$$\bar{u}_j = \bar{u}_0 + \sum_{k=1}^{K} a_k^{(u)} A_{jk} \qquad A_{jk} = \frac{1}{\hat{h}_j}\int_{-\hat{h}_j/2}^{\hat{h}_j/2}\int_{-\hat{h}_j/2}^{\hat{h}_j/2}\int_{-\hat{h}_j/2}^{\hat{h}_j/2} \phi_k(x-\hat{x}_j, y-\hat{y}_j, z-\hat{z}_j)dxdydz$$

Solve the linear system:

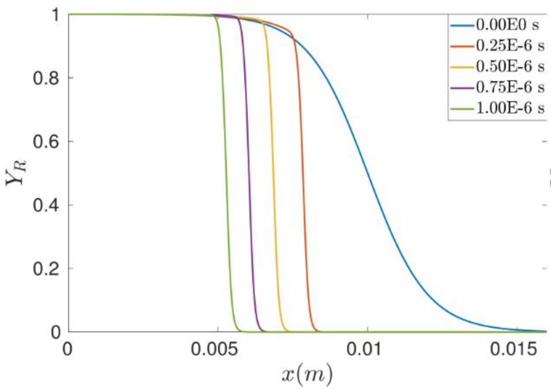
$$A_{jk} a_k^{(u)} = b_j^{(u)}$$

using Singular Value Decomposition, producing the Moore-Penrose Pseudoinverse A_{kj}^*
Note that A_{jk} (and A_{kj}^*) depend only on the local geometric configuration of the stencil.

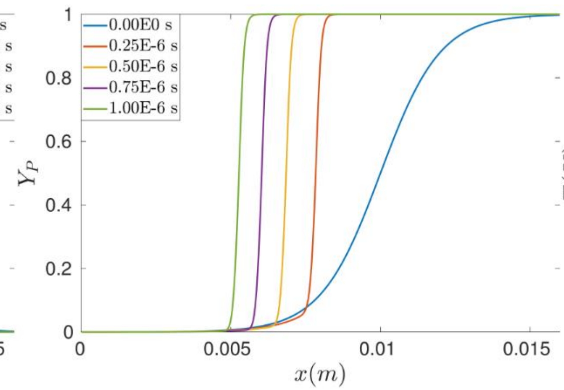
Fluxes obtained from the polynomials evaluated at Gauss integration points on each cell face
Fluxes calculated for the same face in adjacent cells reconciled using a Riemann solver

Code Tests

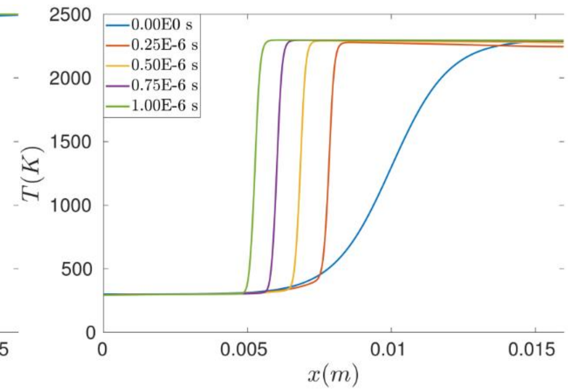
- **1-D planar flame results**



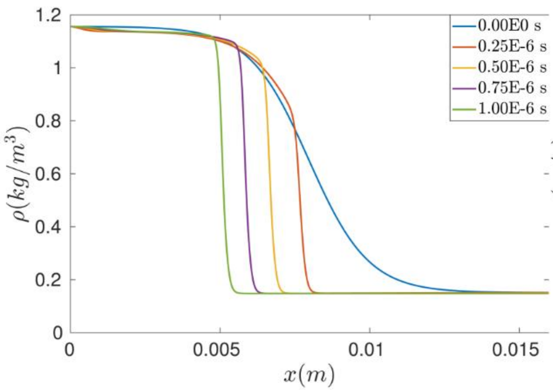
(a) Reactant mass fraction



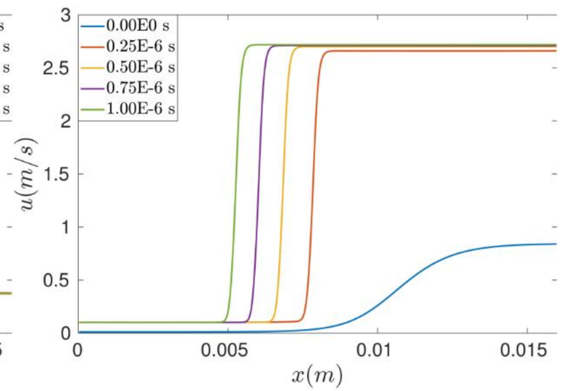
(b) Product mass fraction



(c) Temperature



(d) Density

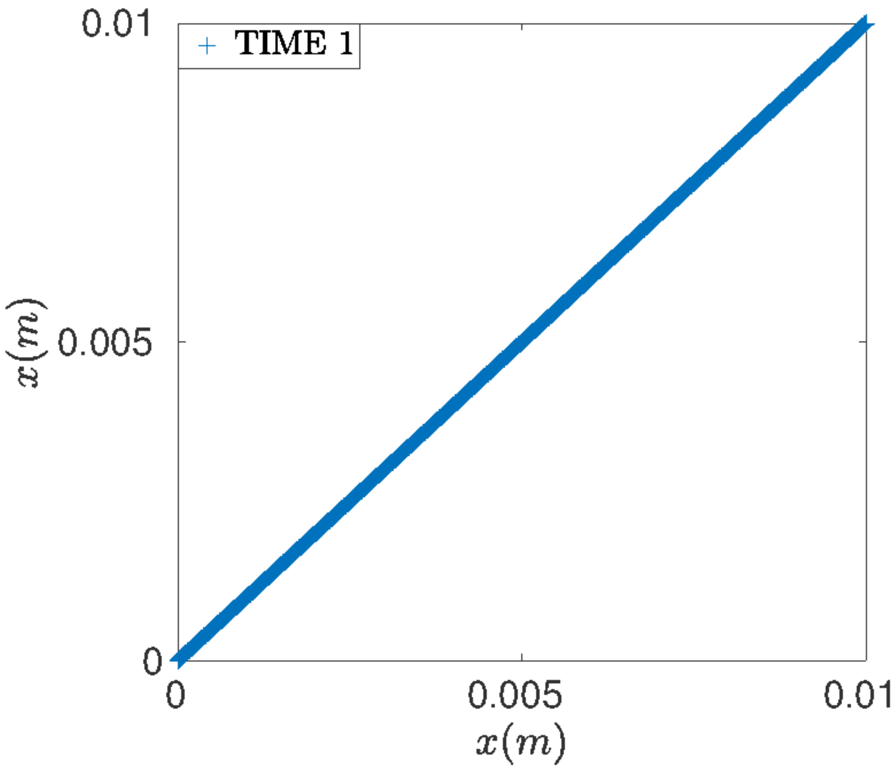


(e) Velocity

Code Tests

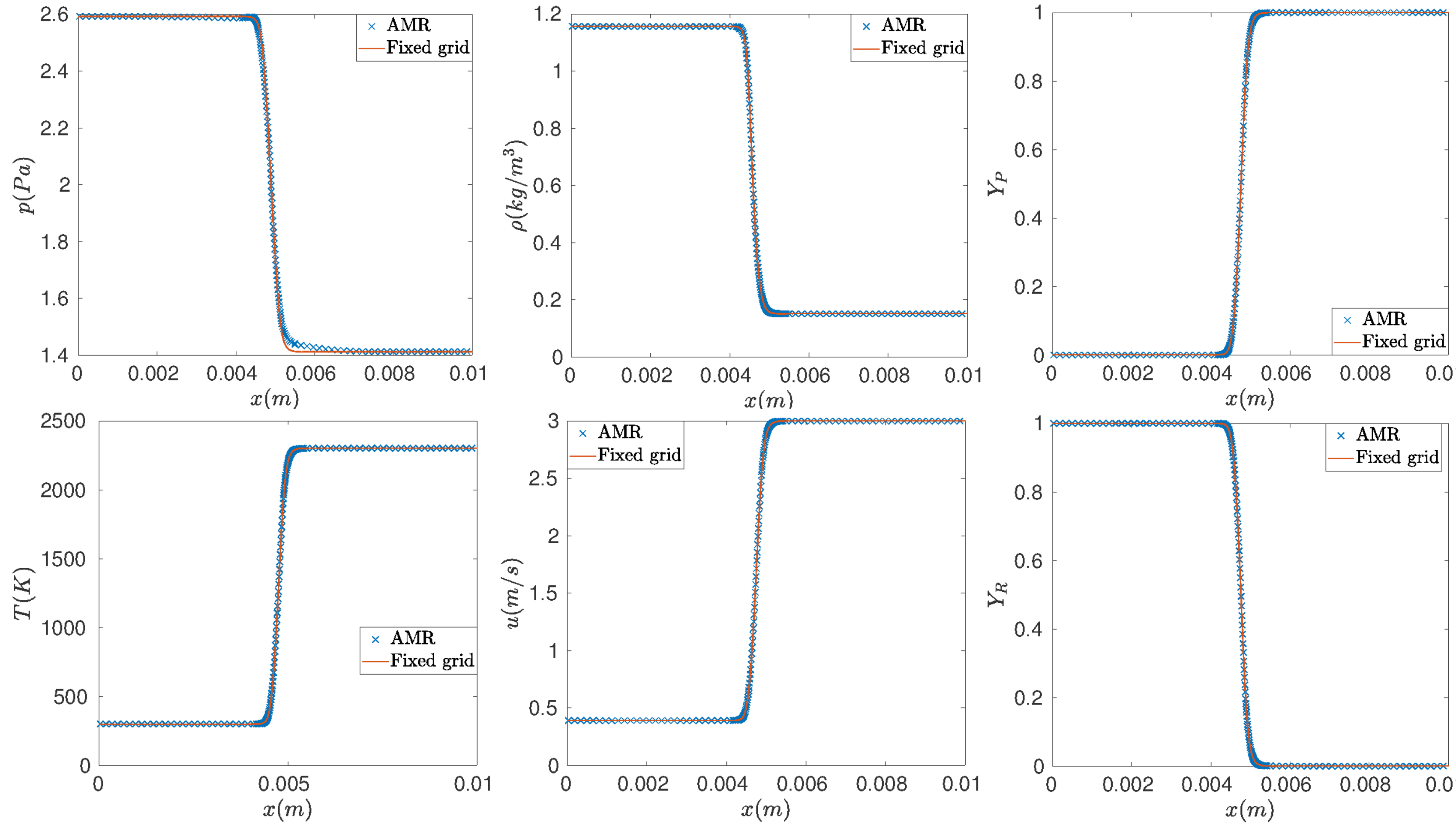
- **1-D planar flame results with AMR**

[redacted]



Code Tests

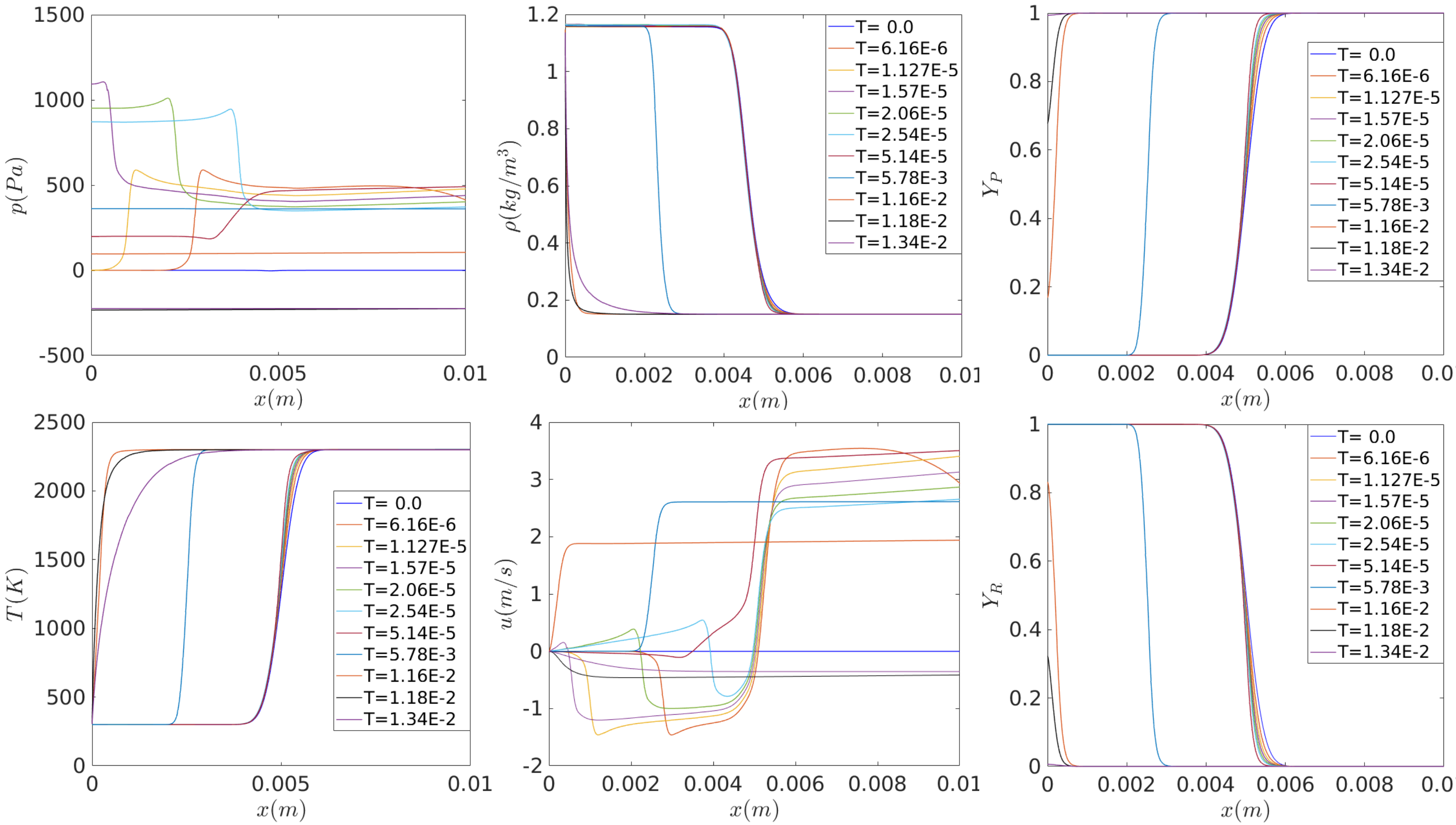
- **1-D planar flame results with AMR**



- Fixed grid simulation with 2048 cells
- AMR simulation finished with 157 cells

Code Tests

- **1-D HOQ results with AMR**

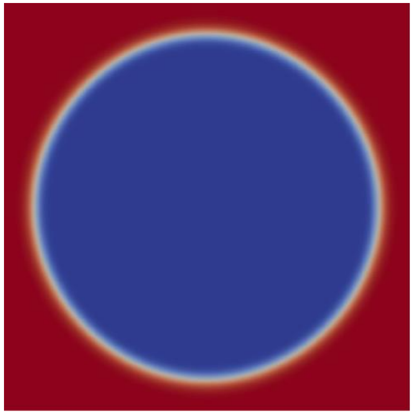


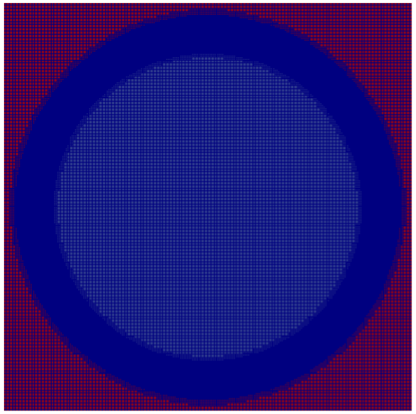
- AMR simulation started with 400 cells, finished with 160 cells

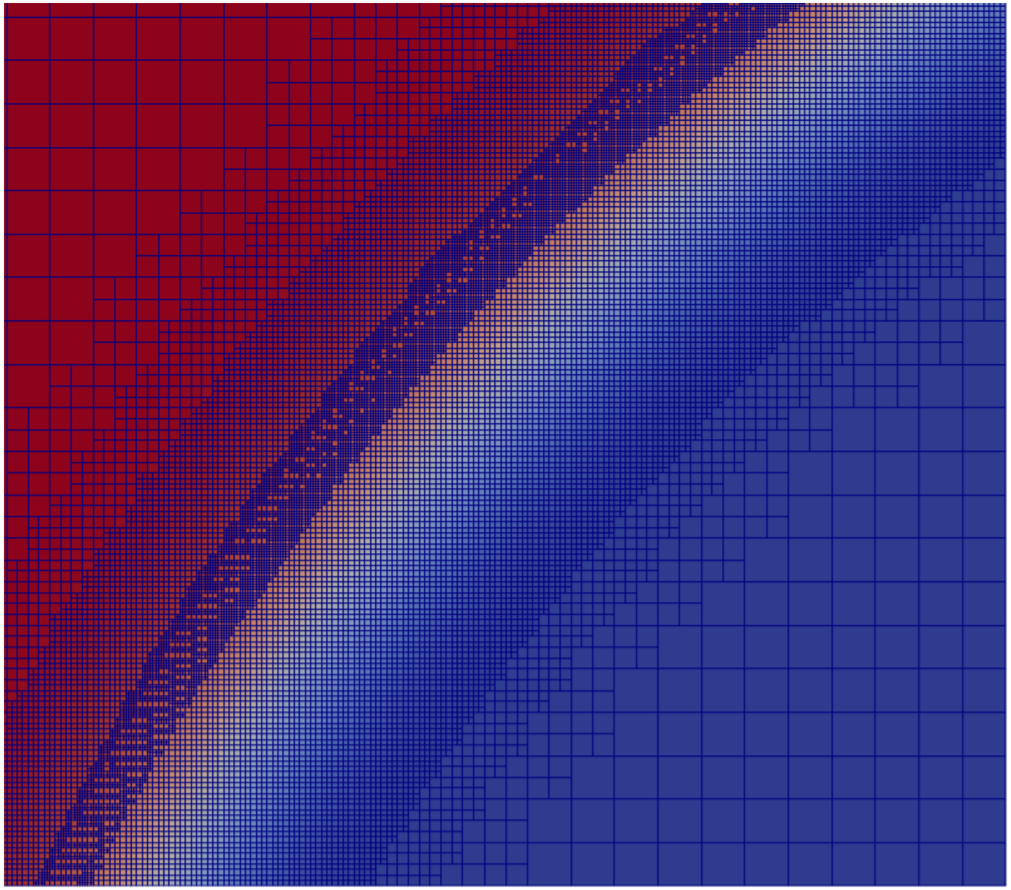
Code Tests

2D laminar flame propagation

- NSCBC gradient outflow
- Single-step chemistry
- AMR + parallel
- Base mesh 128x128
- Circular laminar flame
- Inward propagation



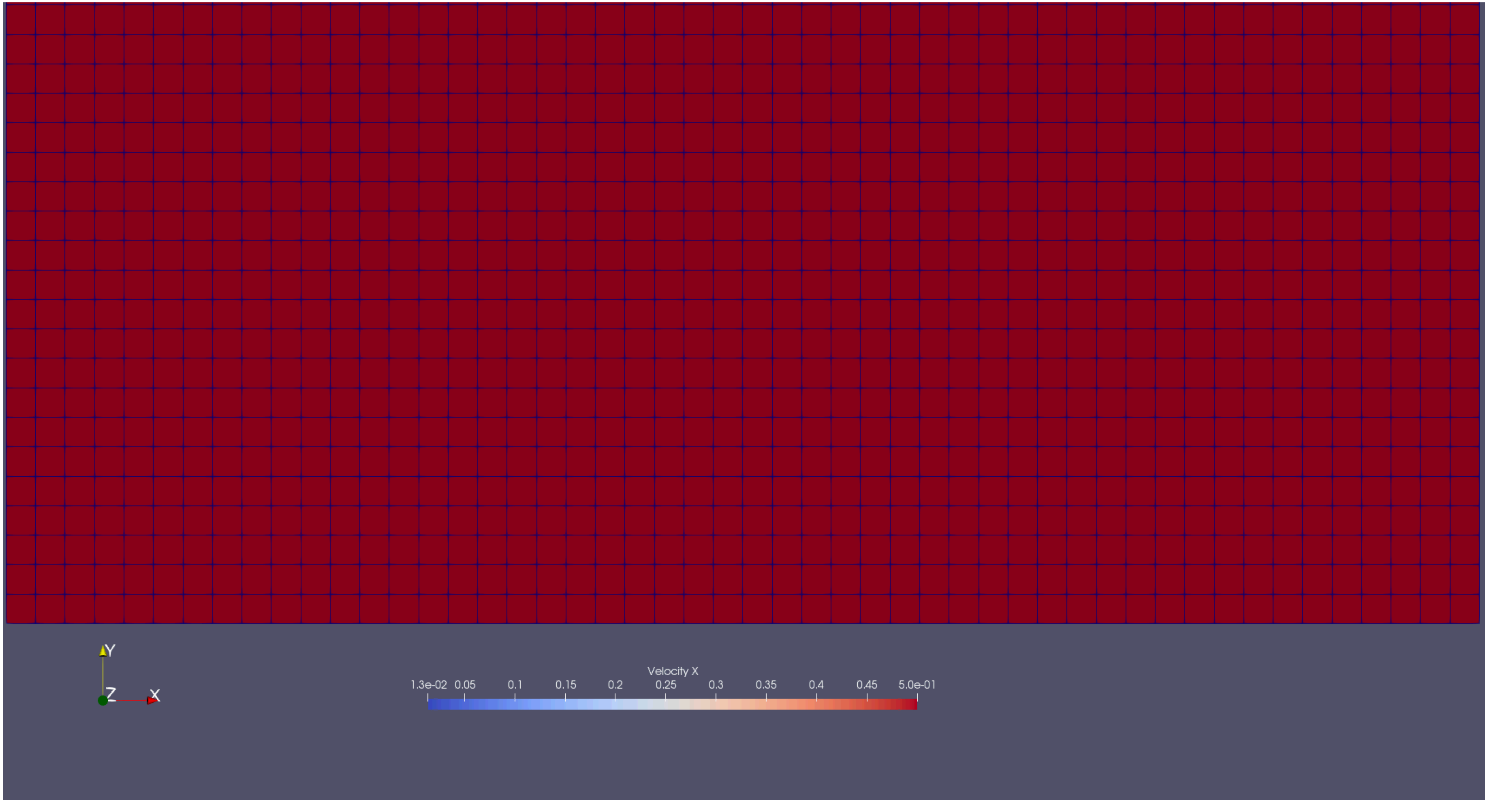




Code Tests

- **2-D Periodic Channel flow: results with AMR**

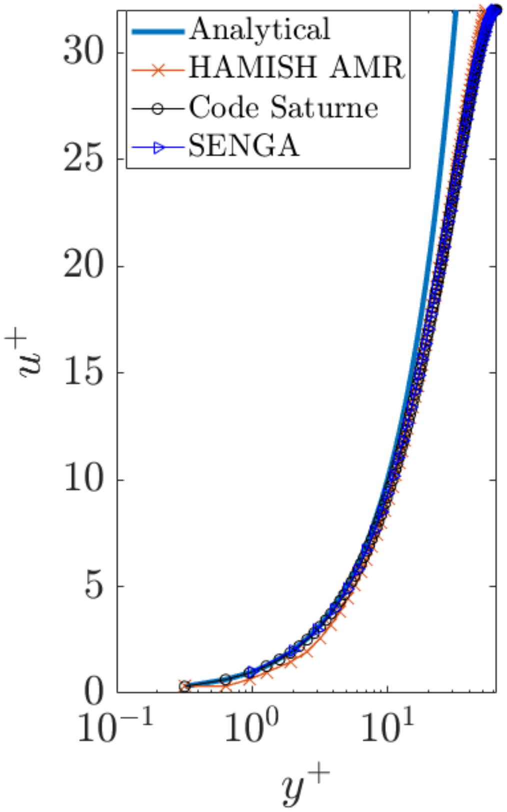
Non-reacting viscous flow

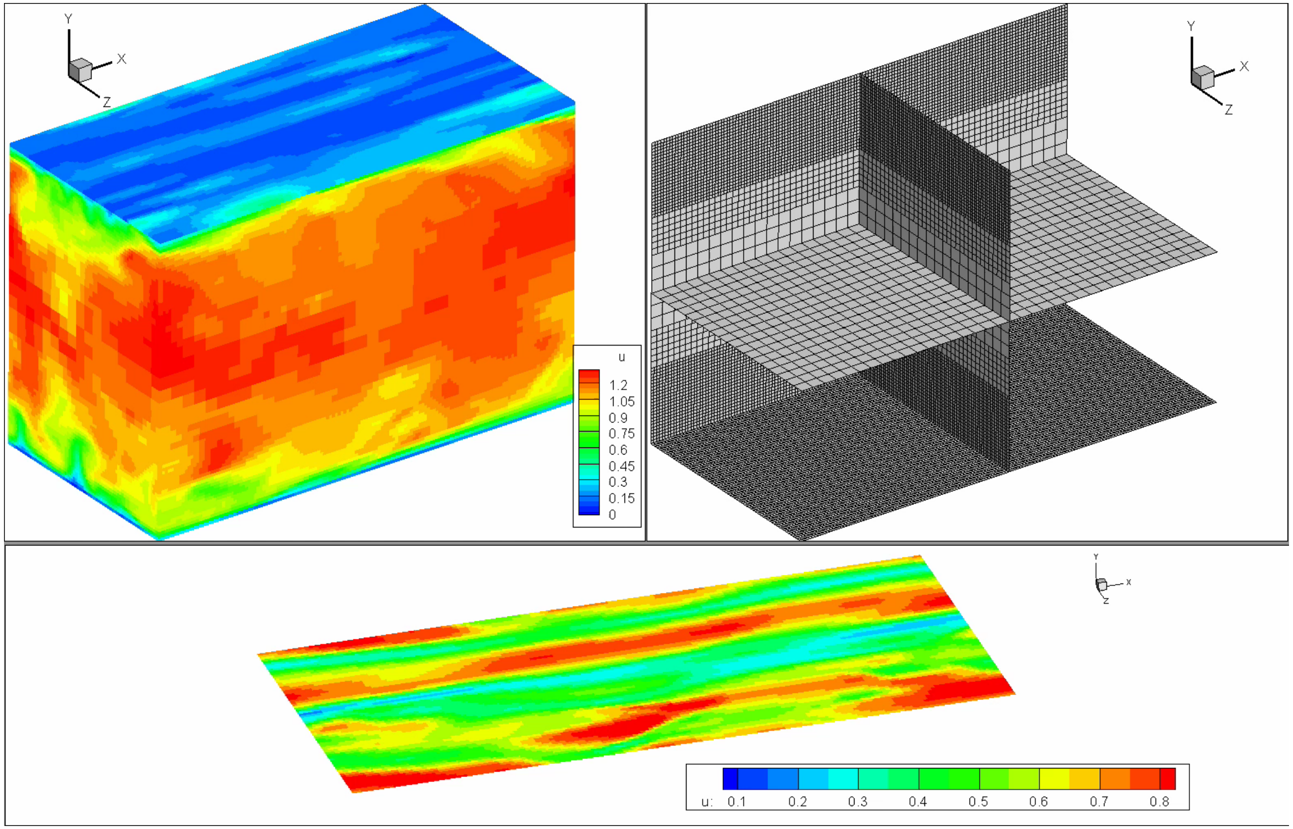


Code Tests

laminar and turbulent channel flow

2D comparisons





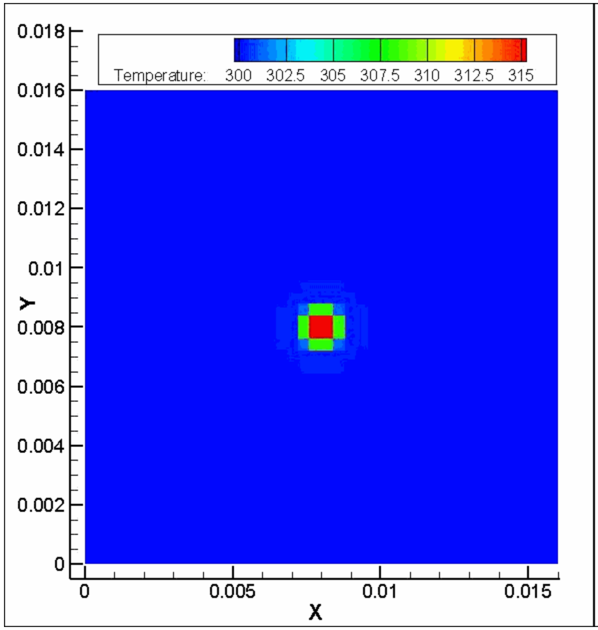
3D simulations with AMR

Code Tests

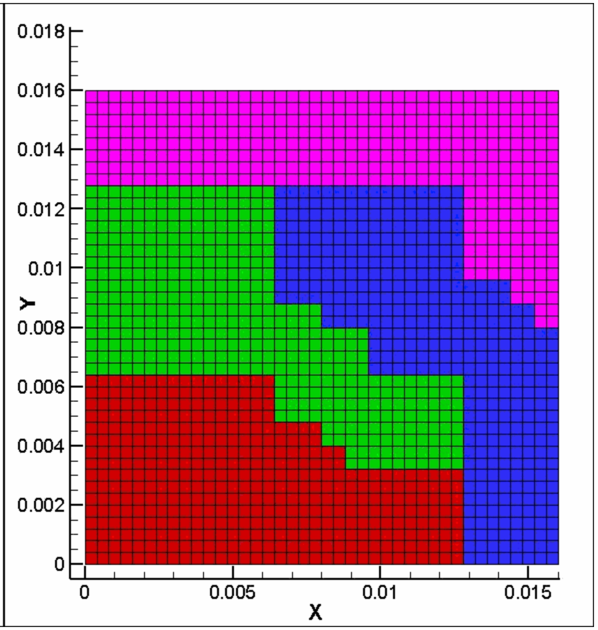
- **2-D thermal conduction problem**

  - Pure thermal conduction case
  - Periodic boundary condition for all sides
  - No chemical reaction
  - Initial condition

$$T = 300 + 100exp\left(-\frac{(x-x_0)^2+(y-y_0)^2}{4\delta^2}\right)$$



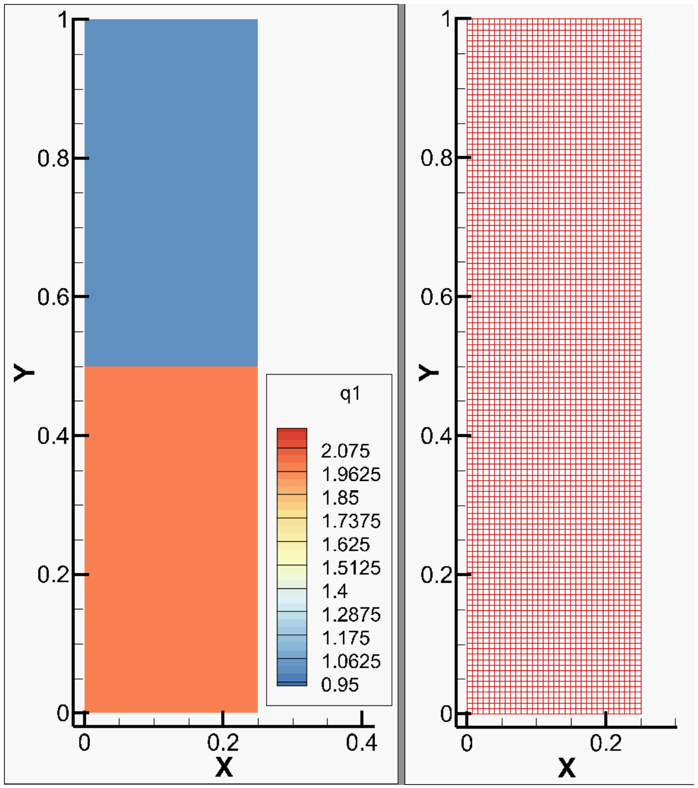
Temperature



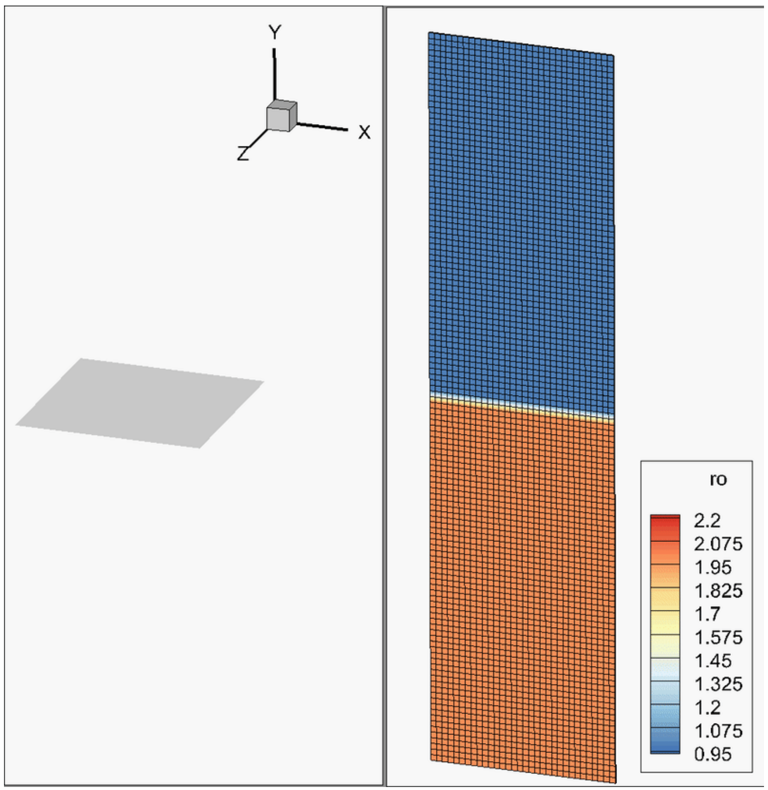
Dynamic load balance

Code Tests

Rayleigh-Taylor instability



2D

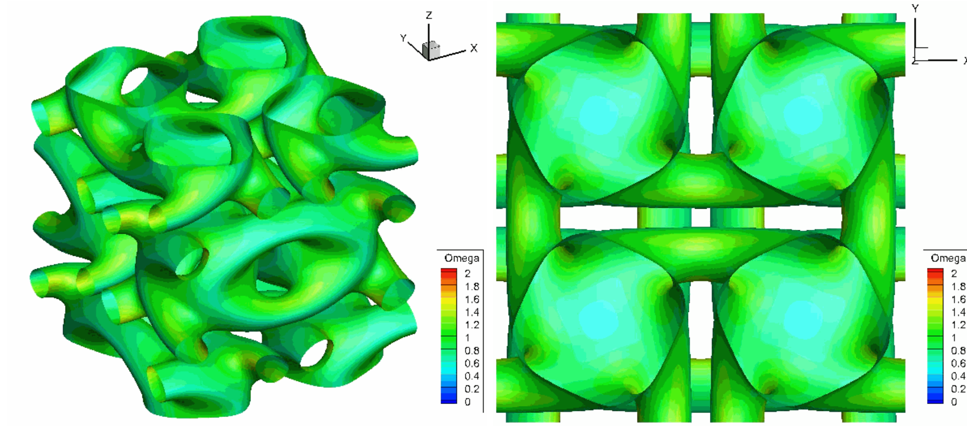


3D

Code Tests

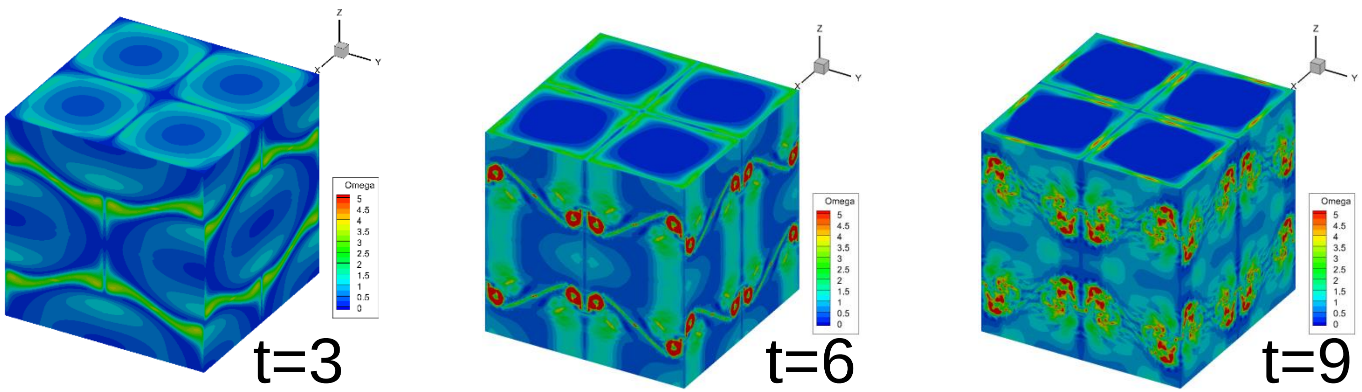
- **3-D Taylor-Green vortex**

$$u = U_0 \sin(x/L)\cos(y\,/\,L)\cos(z\,/\,L)$$
$$v = -U_0 \cos(x/L)\sin(y\,/\,L)\cos(z\,/\,L)$$
$$w = 0$$
$$p = p_0 + \frac{\rho_0 U_0^2}{16}[\cos(2x/L) + \cos(2y/L)][\cos(2z\,/\,L) + 2]$$
$$\rho = \rho_0$$
$$T = \frac{p}{\rho R}$$



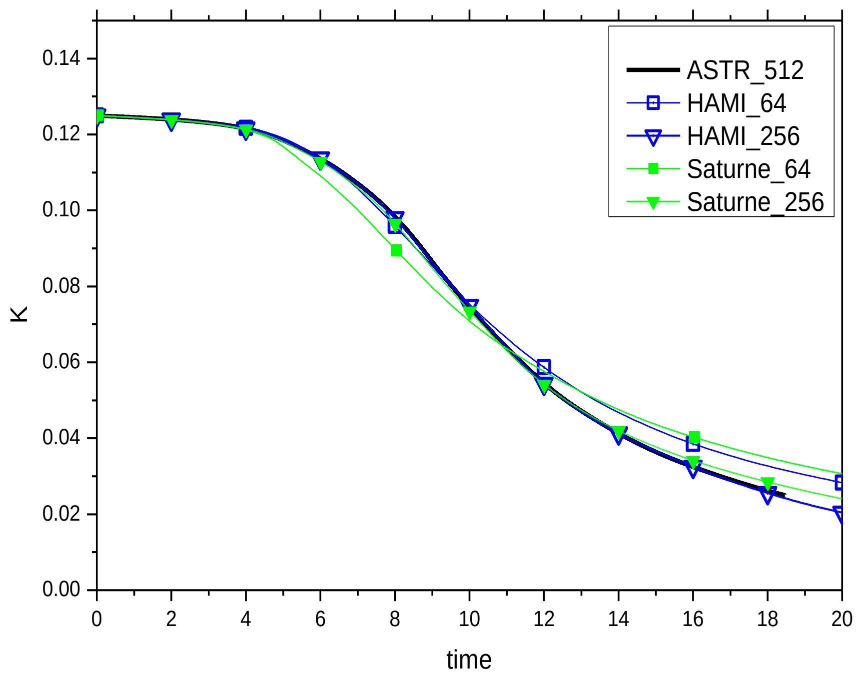
$2\pi \times 2\pi \times 2\pi$, Re=1600, Ma=0.1

J. R. Bull and A. Jameson. "Simulation of the Taylor–Green Vortex Using High-Order Flux Reconstruction Schemes", AIAA Journal, Vol. 53, No. 9 (2015), pp. 2750-2761.

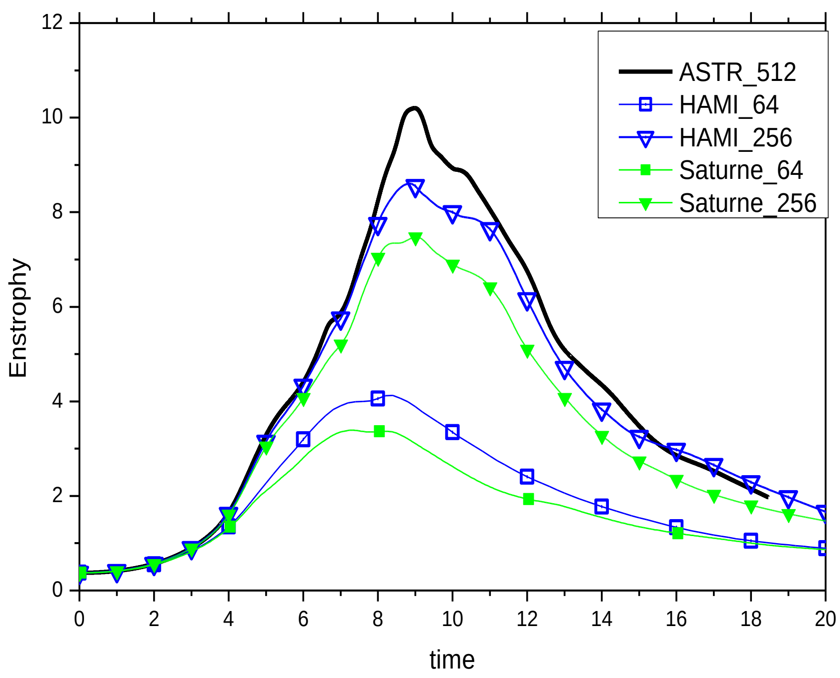


Code Tests

- **3-D Taylor-Green vortex**



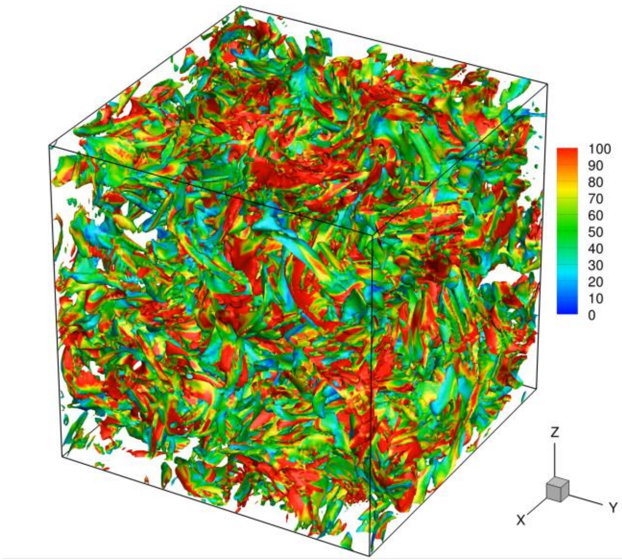
KE temporal evolution



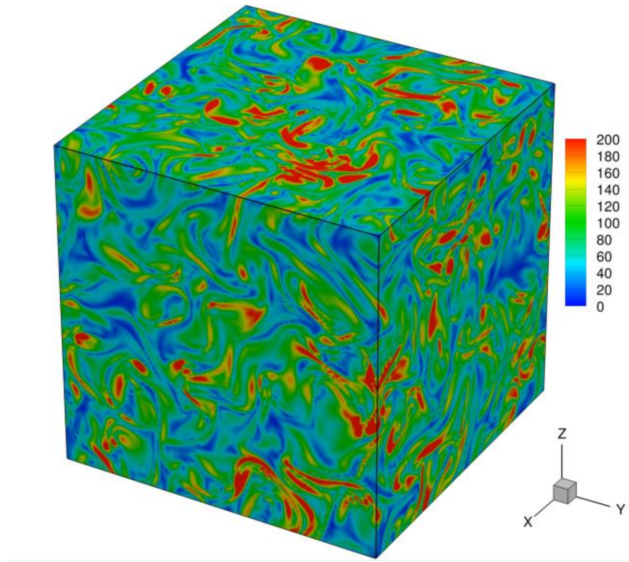
Enstrophy temporal evolution

Code Tests

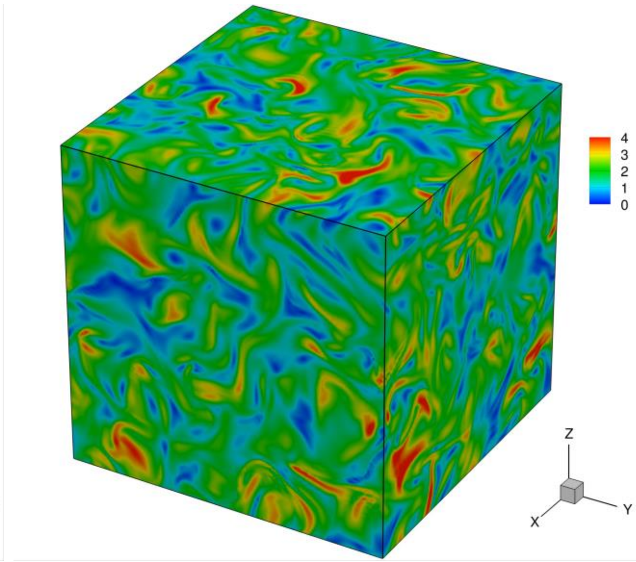
- **3-D Isotropic decaying grid turbulence**



Q-Criterion



Vorticity magnitude



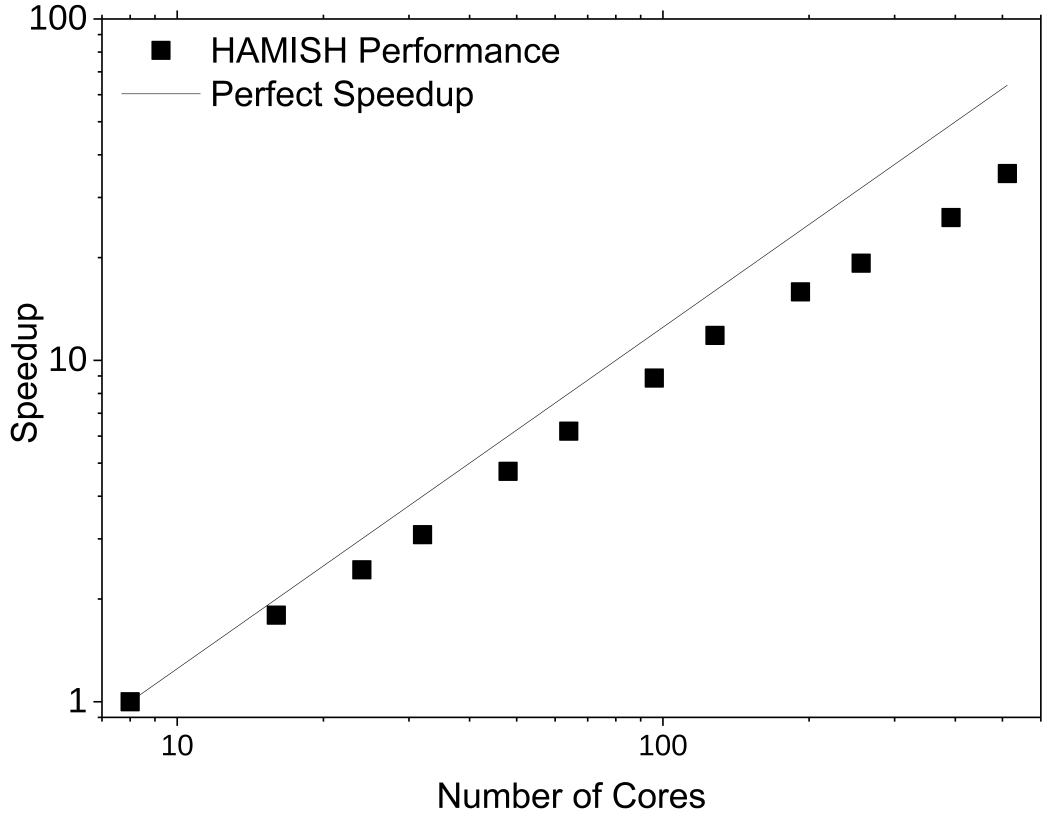
Velocity magnitude

- Fixed grid 128x128x128
- Best AMR criterion remains uncertain
- Criterion based on enstrophy currently being tested

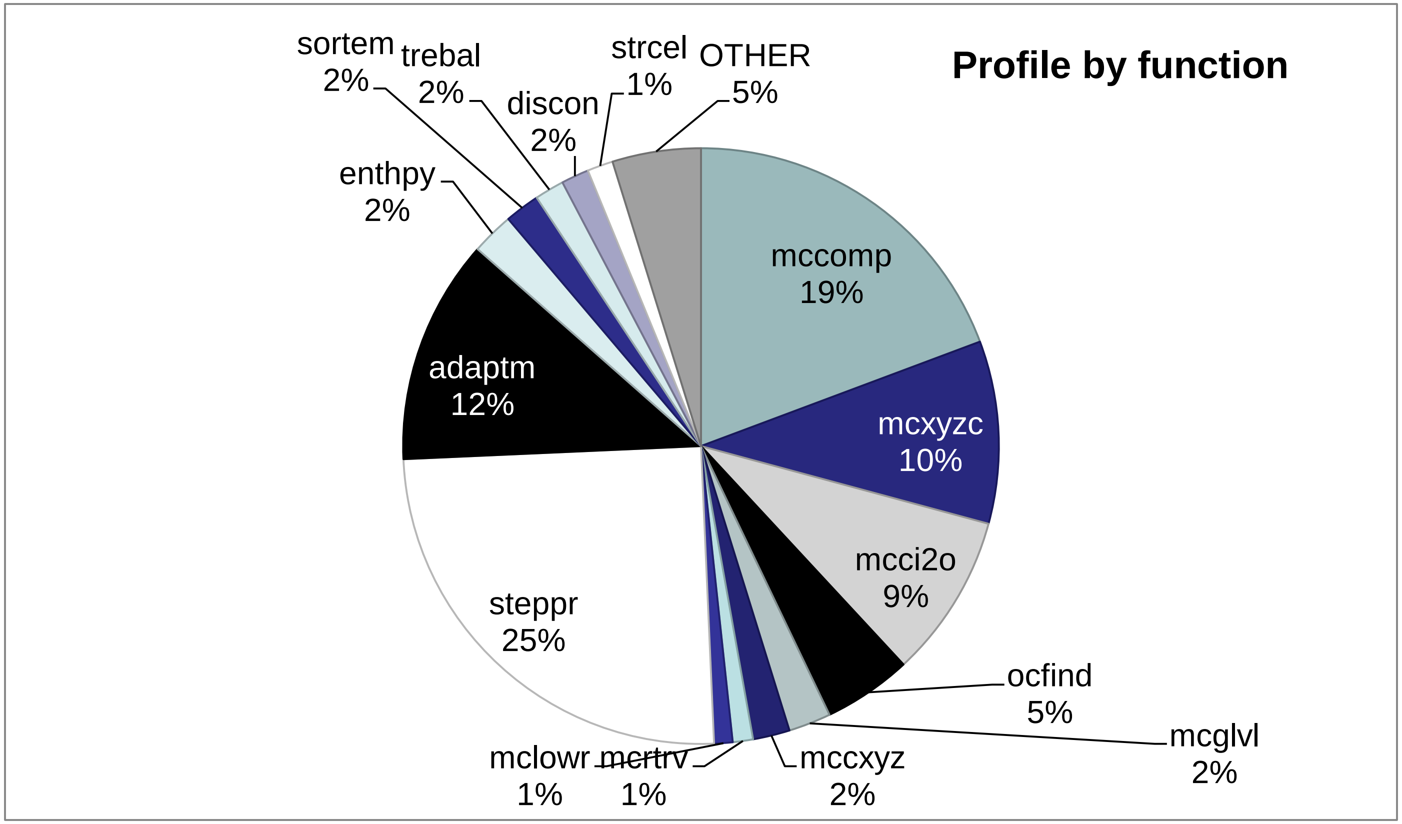
Scalability so far

- Scalability of HAMISH without AMR (128^3 cells)

Cores	Runtime / s	Speedup
8	1632.8	1.00
16	910.6	1.79
24	670.5	2.44
32	529.1	3.09
48	345.1	4.73
64	263.2	6.20
96	184.0	8.87
128	138.0	11.83
192	102.9	15.87
256	84.82	19.26
392	62.3	26.20
512	46.3	35.26



Code Profiling



Pie chart shows relative CPU costs when AMR is active at every solver step. Cost of AMR is about 60% of the total - i.e. a single AMR step costs about 1.5 times as much as a single solver step.

MCCOMP	Compares two Morton codes in their entirety
MCXYZC	Converts x-y-z coordinates into a Morton code at the specified level
MCCI2O	Converts an encoded integer array to an octal string
OCFIND	Searches the local octree using a given Morton code
STEPPR	Time stepping of the solution, including calculating RHS
ADAPTM	Adapts the spatial mesh

Summary and Perspectives

- HAMISH code is being tested and accuracy has been assessed
- Good performance and scalability are observed
- Adaptive Mesh Refinement is working and offers good local resolution

  - 1D+2D flames, HOQ, 2D+3D channels, R-T instability, TGV, 3D HIT
- Significant savings in total mesh requirement
- AMR step costs about 50% more than a solver step

  - but AMR step required only every 10 solver steps or fewer
- Current HAMISH code demonstrates its capability in capturing small-scale structures and interfaces in turbulent reacting flows.

Next:

- Further code optimisation
- Improved level set formulation for two-phase flow
- Post-processing tools for turbulence simulation
- OpenMP support

Acknowledgements

The project is supported by the UKCTRF under EPSRC grants EP/K025163/1 and EP/R029369/1, and by EPSRC Software Flagship Project EP/P022286/1