Detonation Modeling in OpenFOAM Using Adaptive Mesh Refinement

Duncan A. McGough

April 29, 2020

University of Colorado, Boulder

Ann and H.J. Smead Aerospace Engineering Sciences

Acknowledgements

Many thanks to Dr. Hamlington, Caelan Lapointe, and Colin Towery for the invaluable assistance and guidance.

Additional thanks to Dr. Diego Arias, and Julie, Jeff, and Vismaya for advice and support.

Overview

Introduction

AMR-Based Detonation Solver in OpenFOAM

Solver Parameter Testing

Summary

Introduction

Motivation

- Many challenges exist regarding modern combustion systems
- Better modeling can improve performance, efficiency, and reliability
- Detonation engines are a growing area of interest
    - Aircraft turbine propulsion
    - Rocket aerospike-based propulsion
- Current CFD models are computationally expensive
- Adaptive meshing can reduce computational expense

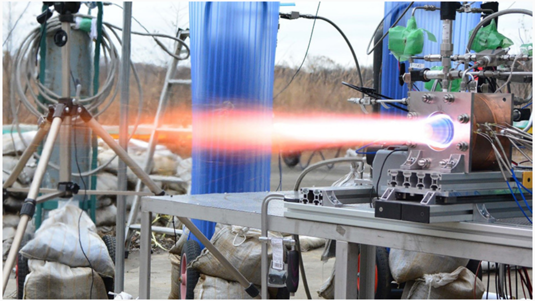


Figure 1: Nagoya University 900 N Rotating Detonation Engine [1]

Objectives

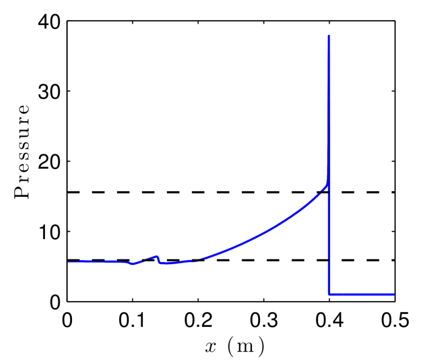
- Implement and test AMR techniques for simulations of detonations found within RDEs and PDEs within OpenFOAM [2], which has:
  - geometric flexibility
  - consistent input structure (easy to transition between solvers)
  - good documentation
  - open source, GNU GPL
  - industry use
  - existing infrastructure with the high-performance computing clusters used by TESLa
- Build a framework on previous work by Towery [3] and TESLa at CU Boulder for further RDE and PDE research

Previous Work

- PDE and RDE numerical modeling on static grids
- Detonation modeling with AMR using in-house codes
- Flame and combustion modeling with AMR in OpenFOAM
- OpenFOAM detonation modeling with AMR without parallel computing or open source

Previous Work: Towery [3]

- Linear detonation tube modeling with in-house Fortran code
- Compared numerical models to Chapman-Jouguet [4] (CJ) detonation model
- Able to produce ZND von Neumann pressure spike [5, 6, 7]
- Simulated idealized and discrete injection, unrolled RDEs



Solver Motivation

Many combustion solvers exist, why do we need a new one?

Open-source	OpenFOAM	Detonations	Adaptive Meshing	Parallelized
	X	X		X
X	X	X		X
	X	X	X	
X	X	X		X
X		X	X	X
X	X		X	X
X	X	X	X	X

AMR-Based Detonation Solver in OpenFOAM

Tested Solvers

Solvers tested for their capability to model shocks and detonations:

- **rhoReactingFoam**: included with OpenFOAM, a density-based combustion solver
- **rhoCentralFoam**: included with OpenFOAM and developed by Greenshields *et. al.* [8], a density-based solver that uses the central-upwind schemes of Kurganov and Tadmor [9]
- **rhoReactingCentralFoam**: a solver combined by Caelan Lapointe with previous work done by Nakul [10], with AMR support

AMR-Based Detonation Solver in OpenFOAM

Governing Equations

Detonations were modeled using the reacting Navier-Stokes equations [11, 12]:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{1}$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) + \nabla p - \mu \nabla^2 \mathbf{u} = \mathbf{0}\,, \tag{2}$$

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot [(\rho E + p)\,\mathbf{u}] - \alpha \nabla^2 e = \dot{q}\,, \tag{3}$$

$$\frac{\partial \rho Y_i}{\partial t} + \nabla \cdot (\rho Y_i \mathbf{u}) - \mu \nabla^2 Y_i = \dot{\omega}_i\,, \tag{4}$$

where

$$\dot{q} = \sum_{i=1}^{N} \dot{\omega}_i \Delta h^0_{f,i}\,, \tag{5}$$

Notes About Governing Equations

- Specific heat $C_{p,i} = C_{p,i}(T)$ from NIST JANAF [13] lookup tables.
- No explicit turbulence modeling, but turbulence can form
    - subgrid-scale turbulence structures averaged out numerically
    - akin to implicit LES
- Not modeling inviscid Euler equations; viscosity is accounted for with Sutherland [14] model:

$$\mu = \frac{A_s\sqrt{T_s}}{1 + \frac{T_s}{T}}, \tag{6}$$

AMR-Based Detonation Solver in OpenFOAM

Chemistry Modeling

Chemical Reactions

Single-step stoichiometric hydrogen-air utilized for simplicity, which follows the following expression [11]:

$$2\,H_2 + 2\,(O_2 + 3.76\,N_2) \longrightarrow 2\,H_2O + 7.52\,N_2$$

with

Species	Mass Fraction
H_2	0.02851
H_2O	0
N_2	0.745
O_2	0.226
Total	0.99951

OpenFOAM allows for easy transitions between chemical models.

Reaction Rate Modeling

Arrhenius equation [15] takes the form [16]

$$\dot{\omega}_i = AT^{\beta} \exp\left(\frac{-E_a}{RT}\right) , \tag{7}$$

where $Ta = Ea/R$. Simulation values were explored, but we settled on

$$A = 1.4 \times 10^{13}\ \mathrm{m^3 mol^{-1} s^{-1}}, \qquad Ta = 12996\ \mathrm{K}, \qquad \beta = 0\,. \tag{8}$$

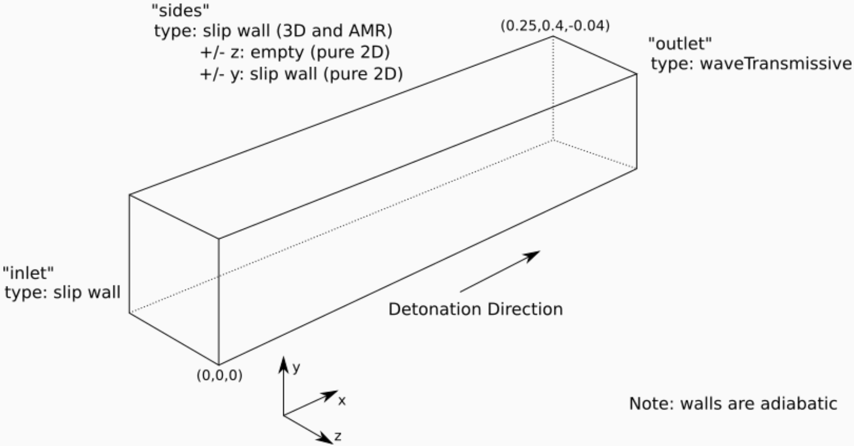
with $R = 368.9$ J/Kg-K. As shown later these reasonably match Chapman-Jouguet detonation theory [4] along with other published values [3, 17].

AMR-Based Detonation Solver in OpenFOAM

Domain Setup

Simulation Domain Setup

Besides ignition region, domain is at 1 atm and 300 K



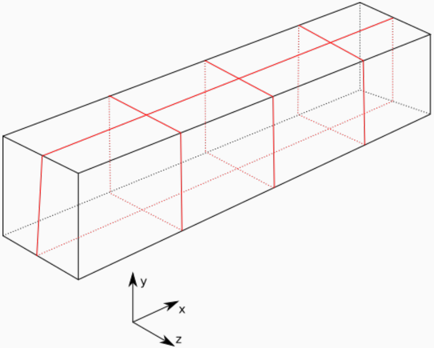
AMR-Based Detonation Solver in OpenFOAM

Parallel Computing

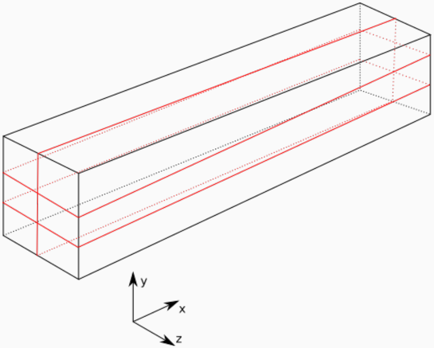
Parallel Computing: Decomposition

- Domain is decomposed into chunks which are independently processed in parallel, communicating with MPI [18]
- Decomposition defined in `decomposeParDict`
- Several methods for decomposing domain in OpenFOAM, but `simple` method was utilized which simply splits number of times in each direction

Parallel Computing: Load Balancing



(a) Domain decomposed into typical chunks, bad for detonation and AMR load balancing



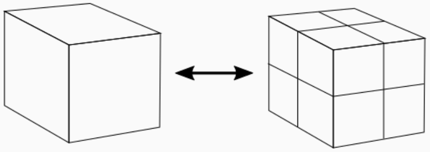
(b) Domain decomposed into long chunks, better for detonation and AMR load balancing

AMR-Based Detonation Solver in OpenFOAM

Adaptive Mesh Refinement

Adaptive Mesh Refinement: Splitting

Adaptive meshing splits the cells using an octree splitting method:



This makes the AMR inherently three-dimensional.

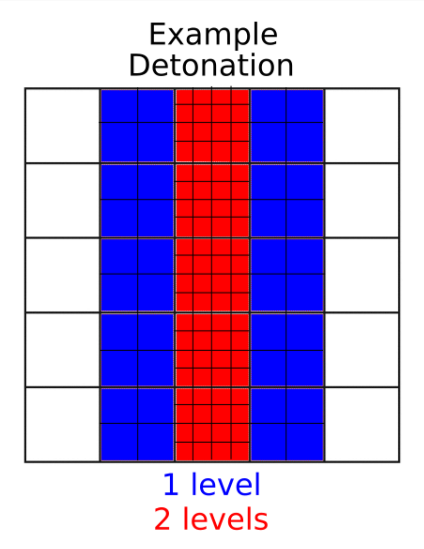
Adaptive Mesh Refinement Parameters

Allows for the mesh to refine and unrefine based on set parameters:

- `refineInterval`: frequency when active, based on time steps
- `field`: which parameter to track
- `lowerRefineLevel`: lower bound of active refinement
- `upperRefineLevel`: upper bound of active refinement
- `unrefineLevel`: upper bound of unrefinement
- `maxCells`: maximum cell count to trigger AMR update

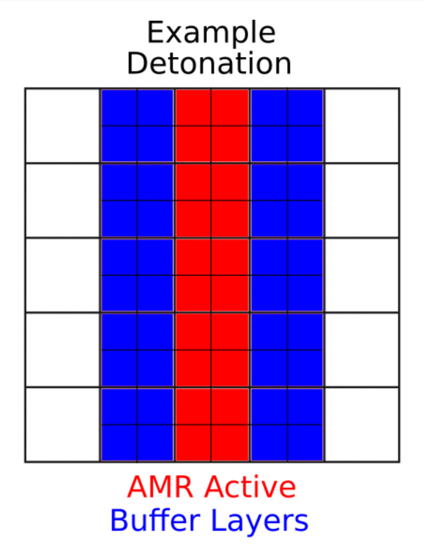
Adaptive Mesh Refinement: Refinement Levels

`maxRefinement`: Number of additional recursive refinement levels



Adaptive Mesh Refinement: Buffer Layers

`nBufferLayers`: Number of buffer cell layers between levels of refinement



Adaptive Mesh Refinement: Detonation Wave

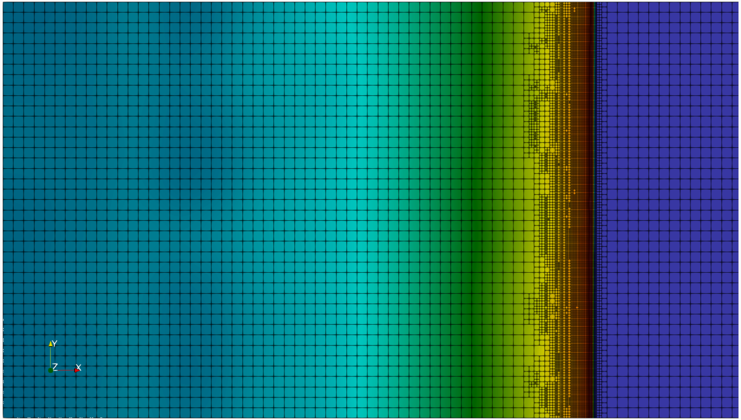
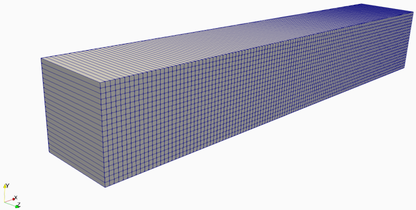
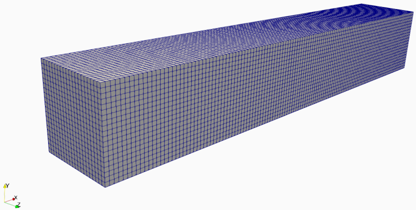


Figure 3: Three-level adaptive mesh refinement over a pressure field surface contour, with the detonation wave traveling from the -x wall to +x exit

Initial Meshing



(a) Two-dimensional mesh



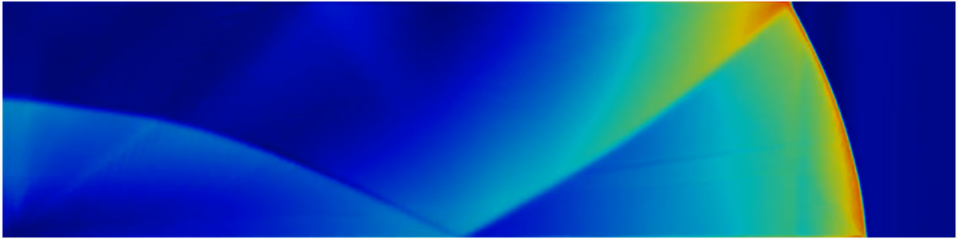
(b) Three-dimensional mesh

AMR-Based Detonation Solver in OpenFOAM

Initial Work and Testing

Initial Detonation Attempts

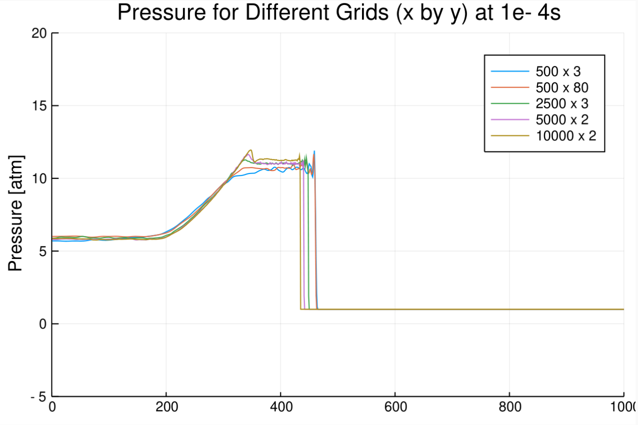
First testing `rhoReactingFoam`, methane-oxygen without nitrogen was used, with corner ignition to test shock reflection and exit



Testing the detonation tube setup performed by Towery *et. al.* [3] was next, to be used as comparison.

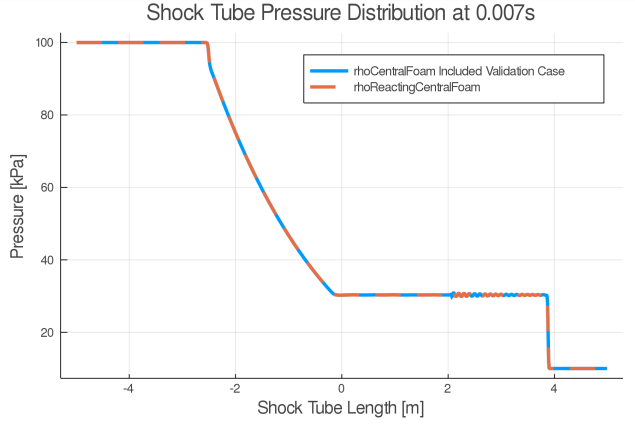
rhoReactingFoam Problems

- Different static mesh resolutions were tested with rhoReactingFoam, but noise and instability in the solution was seen.
- likely due to solver being more pressure-based than density-based



Pressure vs. Grid Location

rhoReactingCentralFoam Selection and Validation



- turned towards the solvers utilizing central-upwind schemes of Kurganov and Tadmor [9]
- used the shock tube test to validate shock-capturing capability

Solver Parameter Testing

Solver Parameter Testing

- Certain parameters can be varied during the modeling of detonations
- Plots shown are pressure distributions, which agree in regards to convergence to temperature and velocity distribution plots
- Grid resolutions will be referred to as their number of cells in that dimension, or primarily in tube length which is 0.25 m long.

Solver Parameter Testing

Detonation Ignition Testing

Ignition Tests

- Detonations can be initialized several ways
  - high T and P reactant block
  - high T and P inert (e.g. helium) block
  - gradient ignition, shown by Towery *et. al.* [19]
- Necessary to test before AMR testing began

Block Ignition

Block of reactants at 3000 K and 20 atm spanning 0.001 m:

Abandoned due to due instability in detonation initiation.

Gradient Ignition

Gradient spanning from 1200 K and 4 atm to 300 K and 1 atm over 0.01 m:

This produced smoother and more consistent detonations.

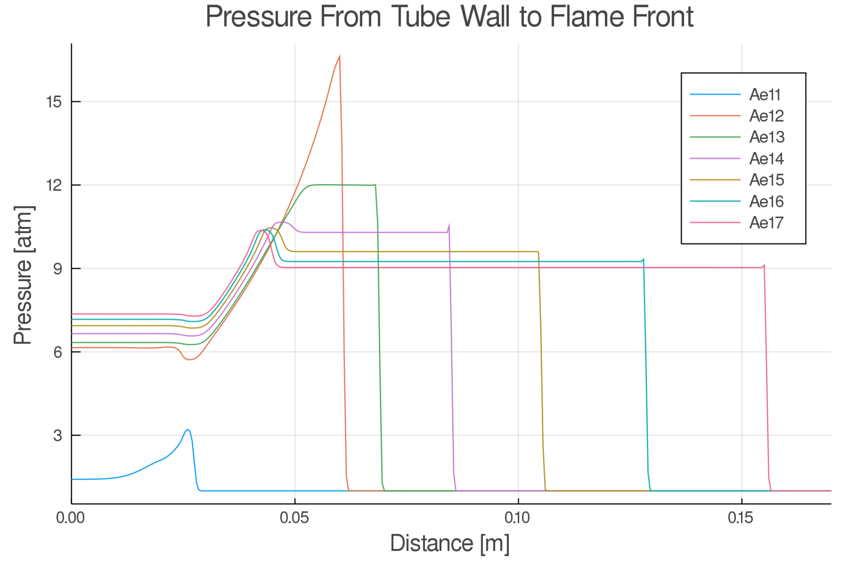
Solver Parameter Testing

Arrhenius Pre-exponential Factor Variation

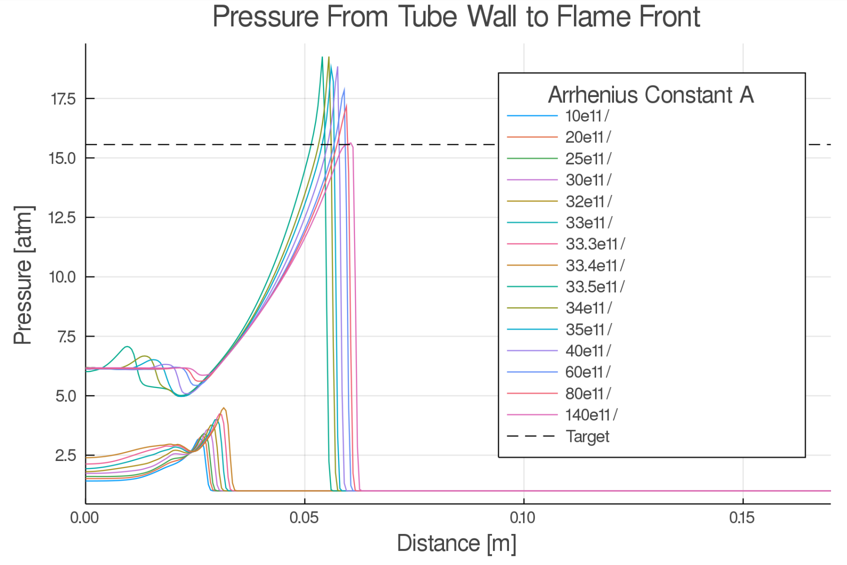
Arrhenius Pre-exponential Factor A Variation

- Wanted to gauge sensitivity to Arrhenius pre-exponential factor as well as get a sense of how values compare to published values
- Swept exponent between 10^{11} and 10^{17}
- Plotted CJ target from similar setup by Towery *et. al.* [3]

Arrhenius A Variation: Pressure Distribution



Arrhenius A Variation: Detonation Decoupling



Solver Parameter Testing

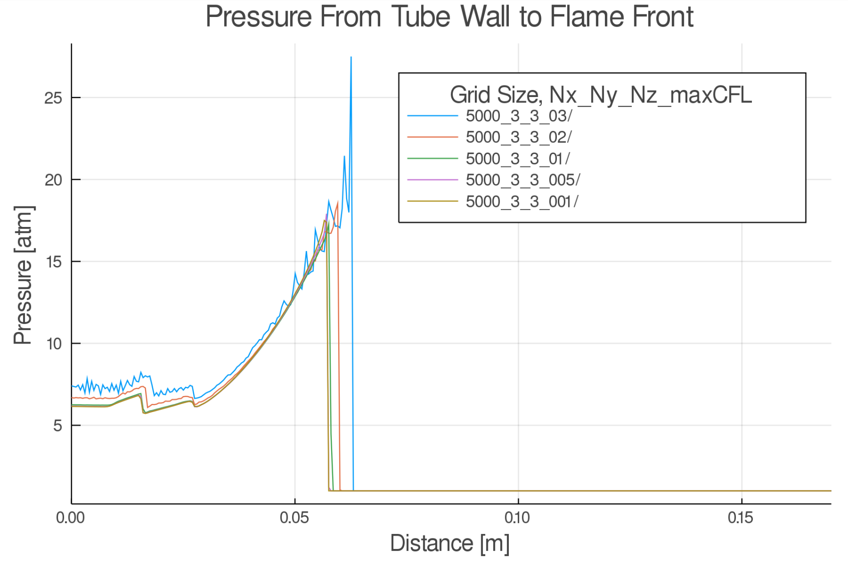
Time Step Variation

Time Step Variation

- Larger timesteps reduce computational expense, but can increase solution instability
- `rhoReactingCentralFoam` can change timestep based on both max-desired central and acoustic Courant numbers
  - most existing OpenFOAM solvers track solely the central courant number
- To characterize solution behavior, the max-allowed CFL was varied

Max CFL	Pressure Error (%)	Temperature Error (%)	Velocity Error (%)
0.3	57.18	19.54	27.62
0.2	5.89	8.72	8.66
0.1	1.28	2.71	12.92
0.05	2.34	1.29	8.96
0.01	0	0	0

Time Step Variation: Pressure Distribution



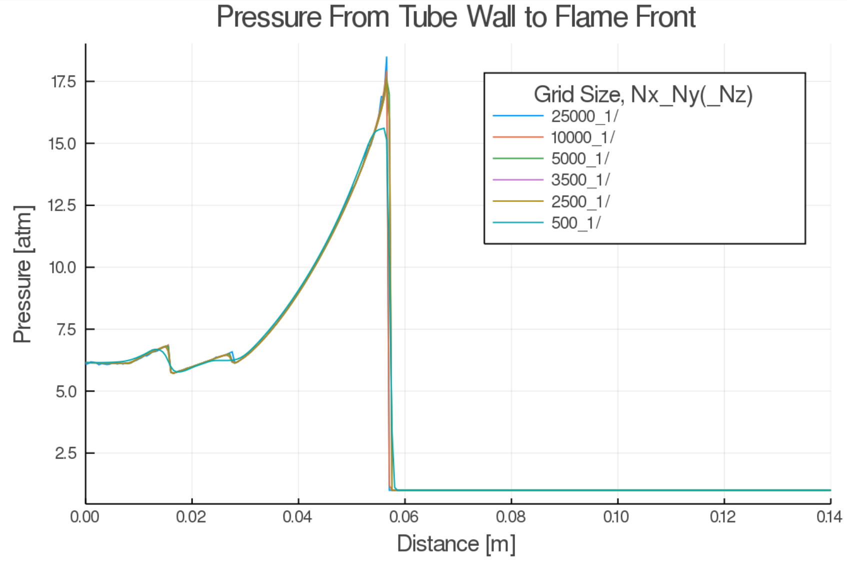
Solver Parameter Testing

Static Mesh Variation

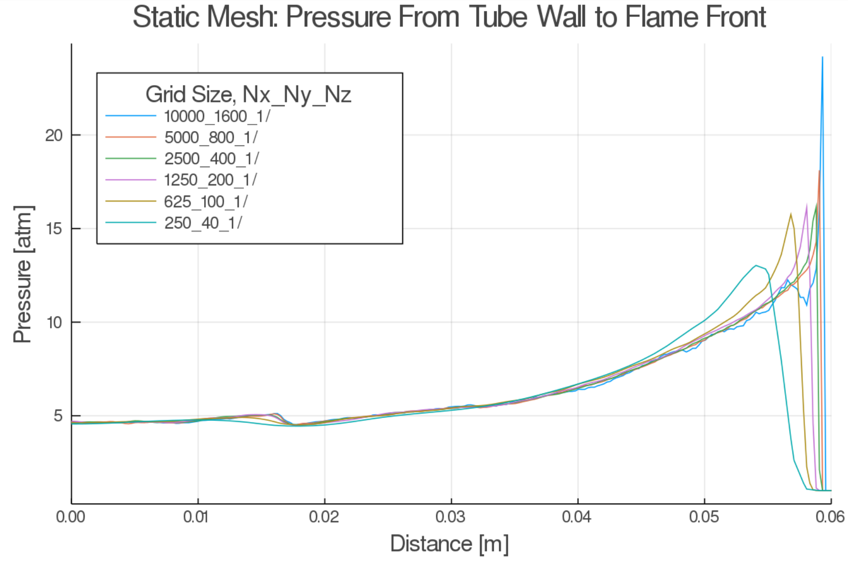
Static Mesh Variation

- Before comparing AMR, need to have baseline static meshes
- Start with one-dimensional, move to two-dimensional
- Static mesh resolutions will help determine convergence criteria for:
  - overall detonation shape convergence
  - ZND Von Neumann pressure spike convergence
  - fine detonation structure convergence

1D Static Mesh Variation: Pressure Distribution



2D Static Mesh Variation: Pressure Distribution



Solver Parameter Testing

Adaptive Meshing

Adaptive Mesh Variation and Comparison

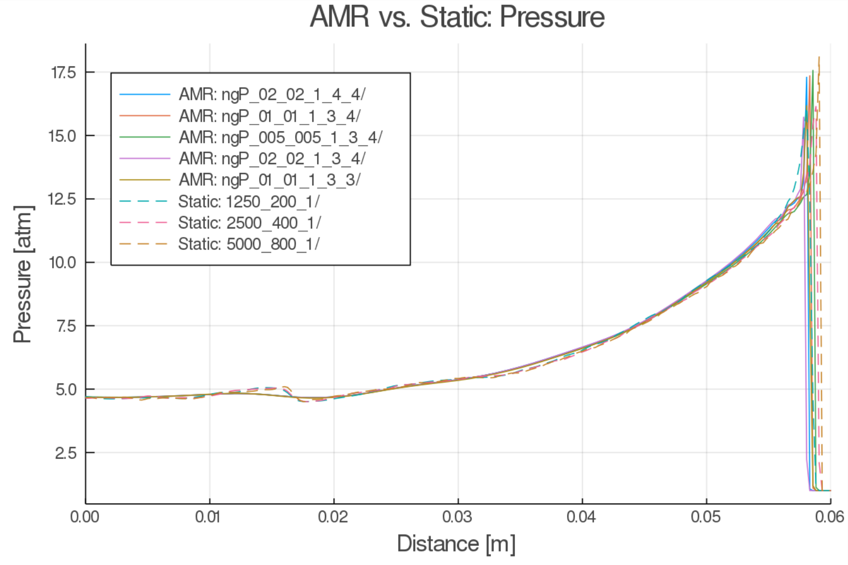
Some AMR comparisons and parameter variations were made next, namely:

- AMR compared to static meshes
- AMR refinement level variation
- AMR buffer layer variation
- AMR normalized pressure gradient bound variation

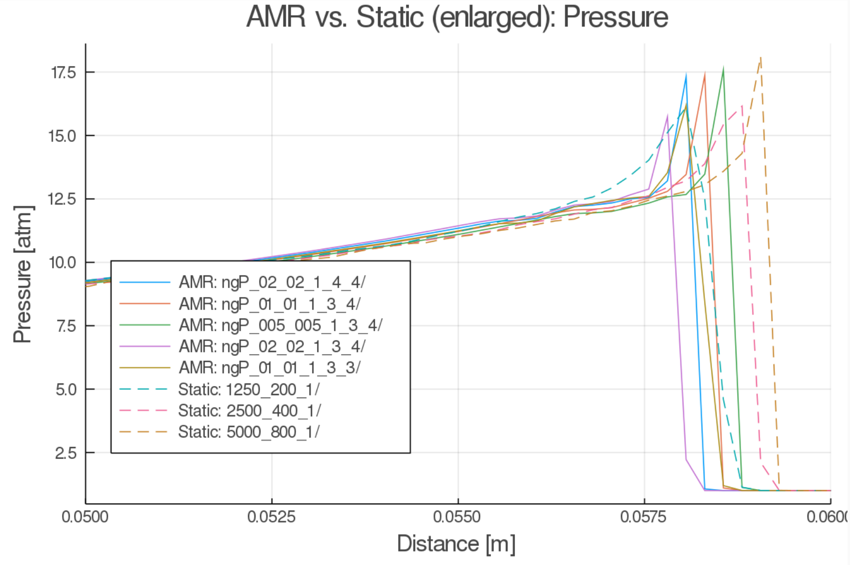
Solver Parameter Testing

AMR and Static Mesh Comparison

AMR vs. Static: Pressure Distribution



AMR vs. Static: Pressure Distribution (enlarged)



AMR and Static Mesh Comparison: Summary

- 240-40-1 base mesh, AMR tracking $||\nabla(p)||$
- Good agreement between 1250-200-1 static mesh and AMR with low/unrefine bound of 0.1 and 3 refine and buffer layers; 85% cell count reduction with exact peak solution reproduction
- AMR with low/unrefine bound of 0.2 and 4 refine and buffer layers can model peak conditions of 5000-800-1 mesh with 96% reduction in cell count ($\Delta x \approx 0.00125$ m)
- Interesting observation: purple AMR run has more refinement levels than brown but brown has a lower AMR trigger bound and is more accurate overall, while being 3.8x smaller

AMR and Static Mesh Comparison: Summary ii

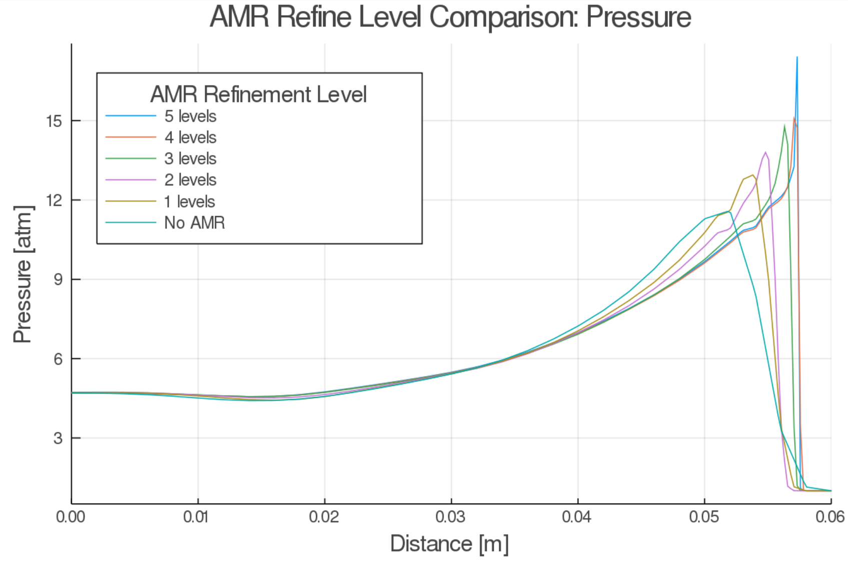
Type colors match static targets to AMR:

Mesh	Type	Unrefine/ lowrefine	Upper refine	Buffer layers	Refine levels	Cell Count
1250-200-1	Static	-	-	-	-	250,000
2500-400-1	Static	-	-	-	-	1,000,000
5000-800-1	Static	-	-	-	-	4,000,000
250-40-1	AMR	0.2	1	4	4	148,950
250-40-1	AMR	0.1	1	3	4	149,762
250-40-1	AMR	0.05	1	3	4	160,570
250-40-1	AMR	0.2	1	3	4	146,710
250-40-1	AMR	0.1	1	3	3	38,343

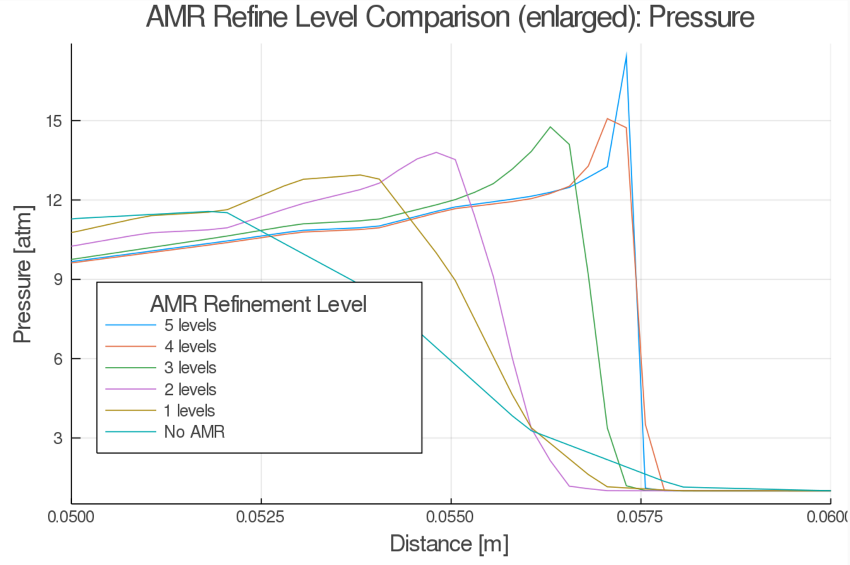
Solver Parameter Testing

AMR Refinement Level Variation

AMR Refinement Level Variation: Pressure Distribution



AMR Refinement Level Variation: Pressure Distribution (enlarged)



AMR Refinement Level Variation Summary

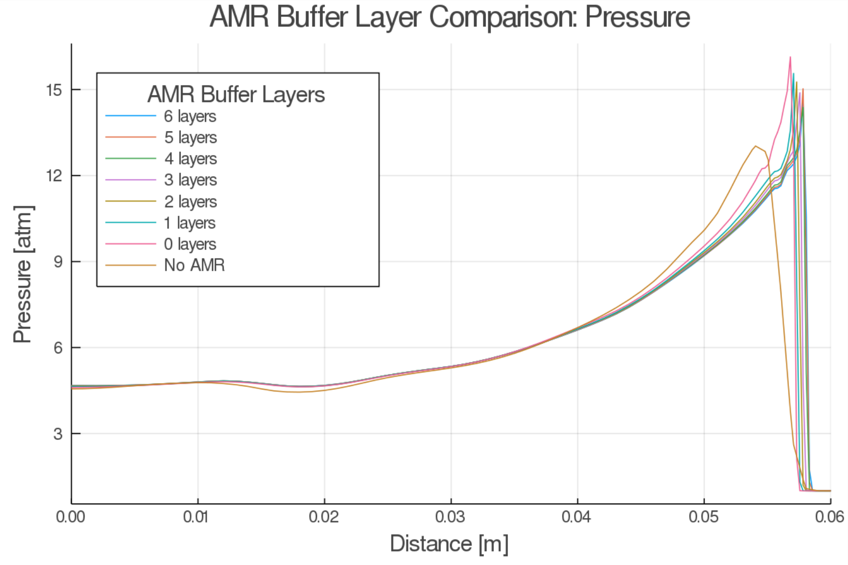
- 125-20-1 base mesh
- tracking $||\nabla(p)||$ from 0.2 to 1, unrefining below 0.2, with 3 buffer layers
- 4 levels (2000-320-1) to begin resolving von Neumann spike, agrees with static trends
- 5 levels nearly matches peak pressure of 5000-800-1 static mesh with 3.7m fewer cells
- cells scale exponentially with refinement level

Refinement Level	Cells
5	228,607
4	54,678
3	13,028
2	5,300
1	2,920
None	2500

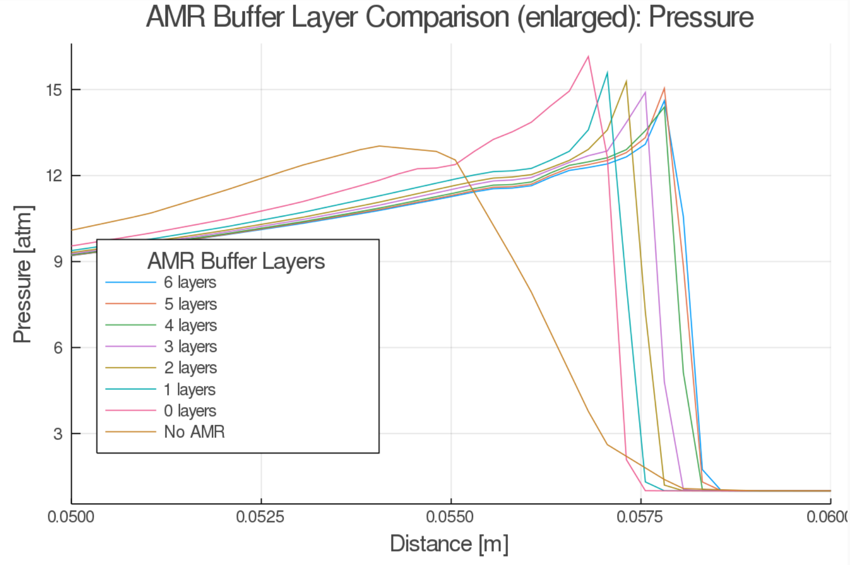
Solver Parameter Testing

AMR Buffer Layer Variation

AMR Buffer Layer Variation: Pressure Distribution



AMR Buffer Layer Variation: Pressure Distribution (enlarged)



AMR Buffer Layer Variation Summary

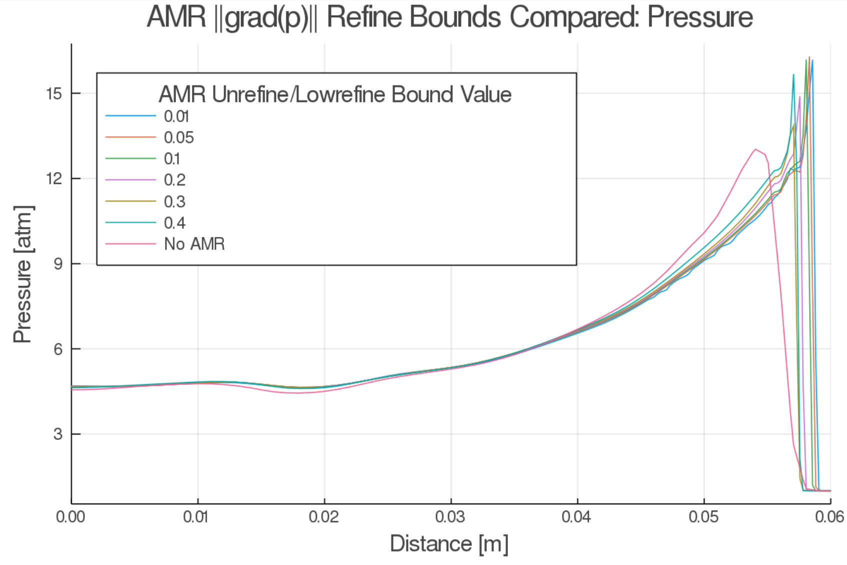
- 240-40-1 base mesh
- tracking $||\nabla(p)||$ from 0.2 to 1, unrefining below 0.2, with 3 refinement levels
- past 4 buffer layers, solution is "converged"
- each layer adds around 0.2 mm to wave progression until "converged", peak solution largely unchanged
- cells scale linearly with buffer layers

Buffer Layers	Cells
6	37,405
5	34,353
4	32,960
3	37,678
2	16,173
1	10,840
None	10,000

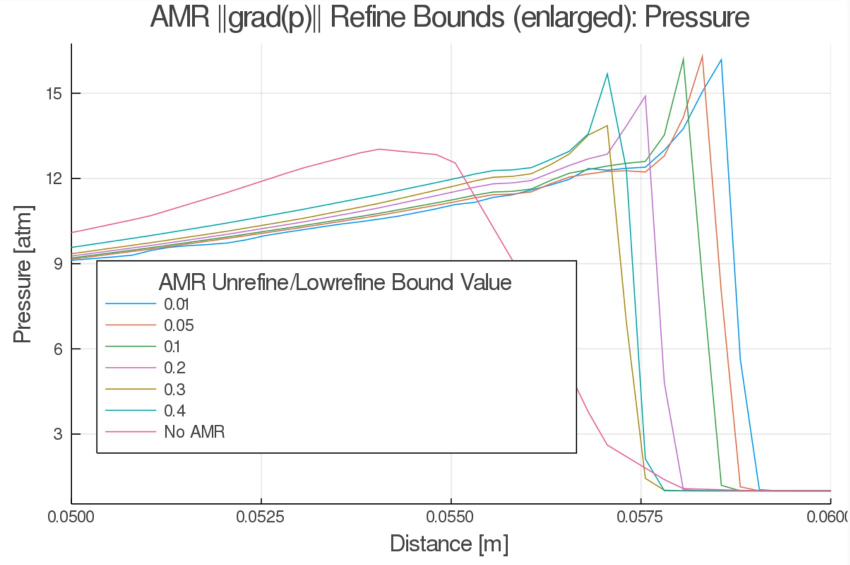
Solver Parameter Testing

AMR $||\nabla(p)||$ Refinement Range Variation

AMR $||\nabla(p)||$ Refine Range Variation: Pressure Distribution



AMR $||\nabla(p)||$ Refine Range Variation: Pressure Distribution (enlarged)



AMR $||\nabla(p)||$ Refinement Range Variation Summary

- 240-40-1 base mesh
- tracking $||\nabla(p)||$ with 3 refinement and buffer layers
- Solution peak values plateau at 0.1 and lower
- Below 0.05 computational expense is exponential
- Above 0.05 computational expense is largely unchanged

Lower un/refinement bound	Cells
0.01	139,136
0.05	39,799
0.1	38,343
0.2	37,678
0.3	37,692
0.4	36,355
No AMR	10,000

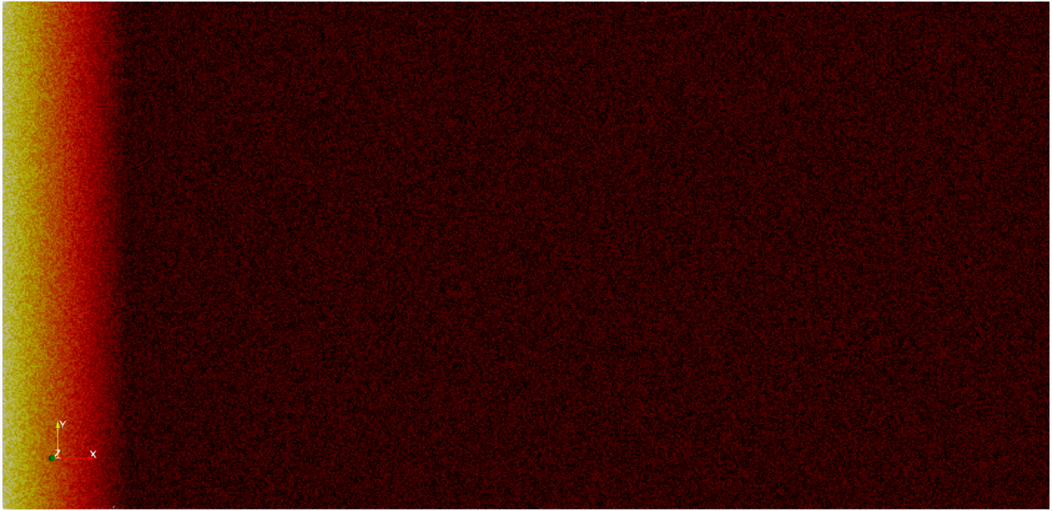
Solver Parameter Testing

Cellular Detonation Modeling

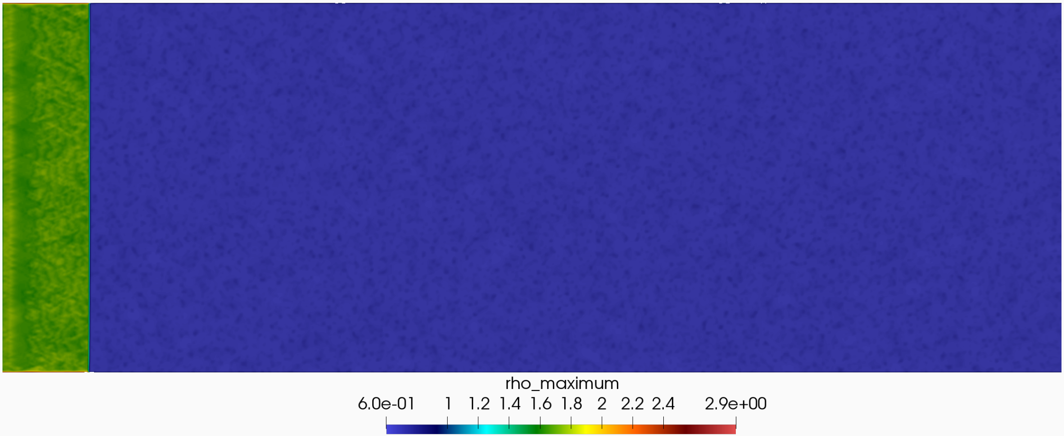
Cellular Detonation Modeling

- Cellular “fishscale” patterns appear on smoked foils placed in detonation tube experiments
- Can be used to verify numerical detonation modeling
- Numerical simulations can replicate this with maximum pressure and density traces over time for each cell
- Temperature randomization was seeded throughout the domain, at up to 20% of the maximum gradient temperature to assist cellular detonation formation

Cellular Detonation Modeling: Initial T Randomization



Cellular Detonation Modeling: Maximum ρ Traces



Summary

Summary i

- `rhoReactingFoam` and `rhoCentralFoam` were tested and found to be incapable of detonation modeling
- `rhoReactingCentralFoam` can model detonations
- Showed how to effectively model detonations in OpenFOAM, with parallel computing and adaptive meshing considerations
- Gradient ignition produces more stable detonations
- Randomly-seeded temperature fields solved with `rhoReactingCentralFoam` have potential to produce cellular detonation structure
- Results are sensitive to Arrhenius pre-exponential factor order, caution must be taken to not decouple shock from flame

Summary ii

- CFL number must be lower than typical high-speed flows
- PDE/detonation tube meshes are seen to:
  - resolve wave structure at 0.1 mm resolution
  - begin to resolve von Neumann spike at 0.05 mm resolution
  - resolve finer detonation structures at 0.025 mm resolution
- AMR refinement levels are exponential in computational expense, and levels agree with static trends for von Neumann spike resolving
- AMR buffer layers are linear in computational expense, and:
  - additional layers add 0.2 mm to wave progression
  - past 4 buffer layers, solution is "converged"
  - peak solution values remain largely unchanged with additional layers

Summary iii

- AMR refinement range variation has:
  - solution values that plateau for lower bound of ≤ 0.1
  - exponential increase of computational expense for lower bound of ≤ 0.05
  - largely unchanged computational expense for upper bound ≥ 0.05
- **AMR can reproduce static mesh detonation simulations with up to 96% reduction in cell count**

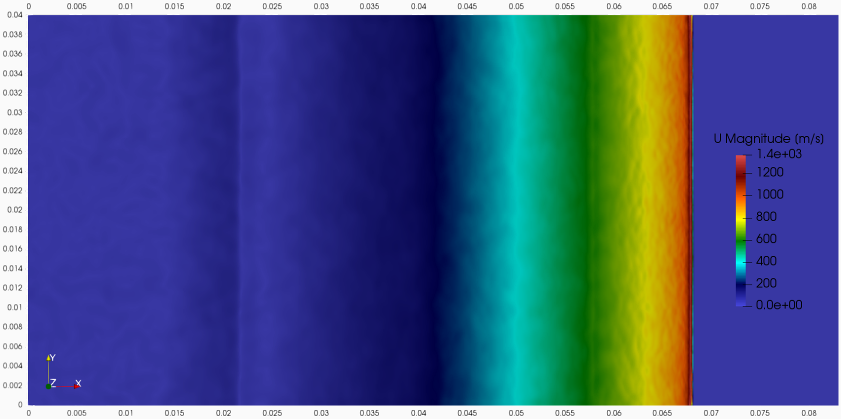
Areas to Improve

- Further research and comparison with three-dimensional cases
- Exploration into further alternate AMR tracking parameters, or combining them
- Better parallel load balancing improvement
- Base mesh characterization on AMR results, with particular effort on unrefinement effects
- AMR can be unstable in parallel

Future Work

- Bringing this AMR work to RDEs
- Deflagration to detonation transition modeling
- Utilizing solver for other propulsion technologies, such as rocket engines

Thank You!



Backup

OpenFOAM Overview

- Open-source, free, computational fluid dynamics toolbox
- Written in C++, has many solvers
- No GUI, file-based input and control

Total energy [11] can be written as

$$E = h - \frac{p}{\rho} + \frac{1}{2}\left(\mathbf{u} \cdot \mathbf{u}\right) , \tag{9}$$

where the total summed enthalpy is written [11] as

$$h = \sum_{i=1}^{N} h_{s,i} Y_i , \tag{10}$$

and the species total enthalpy [11] is given by

$$h_i = \Delta h_{f,i}^0 + h_{s,i} , \tag{11}$$

Governing Equations ii

with the sensible enthalpy for the ith species expressed as

$$h_{s,i} = \int_{T_0}^{T} C_{p,i} \mathrm{d}T \, , \tag{12}$$

Here $C_{p,i}$ is the specific heat for the ith species, T is the temperature, and T_0 is an initial, or reference, temperature. The equation of state is expressed as

$$p = \rho R T \, , \tag{13}$$

where $R = R_u/W$. Specific heat $C_{p,i} = C_{p,i}(T)$ from NIST JANAF [13] lookup tables.

Backup

OpenFOAM Numerical Setup

OpenFOAM Finite Volume Numerical Schemes

Term	OpenFOAM Variable	Numerical Scheme
Flux Scheme	`fluxScheme`	Kurganov
Time Scheme	`ddtSchemes`	Euler
Gradient Schemes	`gradSchemes`	Gauss linear
Divergence Schemes	`divSchemes`	none by default
	`div(tauMC)`	linear
	`div(phi,specie)`	van Leer
Laplacian Schemes	`laplacianSchemes`	Gauss linear uncorrected
Interpolation Schemes	`interpolationSchemes`	default linear
	`reconstruct(rho)`	Minmod
	`reconstruct(U)`	MinmodV
	`reconstruct(T)`	Minmod
	`reconstruct(Yi)`	Minmod
Surface Normal Gradient Schemes	`snGradSchemes`	uncorrected

OpenFOAM Finite Volume Numerical Solvers

Variable	Solver	Parameter	Value
`e`, `Y`	`PBiCGStab`	Preconditioner	`DILU`
		Tolerance	`1e-17`
		Relative tolerance	0
`U`	`PBiCGStab`	Preconditioner	`DIC`
		Tolerance	`1e-15`
		Relative tolerance	0
`rho`	`diagonal`		

Backup

OpenFOAM Case Setup

OpenFOAM Directory Structure

An OpenFOAM case is divided into:

- `0/`: holds initial conditions and boundary conditions for quantities like pressure, temperature, velocity, etc.
- `constant/`: holds thermophysical quantities and some mesh information
- `system/`: contains numerical settings, mesh setup, and simulation settings

OpenFOAM `constant/` Directory

Contains:

- `chemistryProperties`
- `combustionProperties`
- `dynamicMeshDict`
- `reactions`
- `thermo.compressibleGas`
- `thermophysicalProperties`
- `turbulenceProperties`

OpenFOAM `system/` Directory

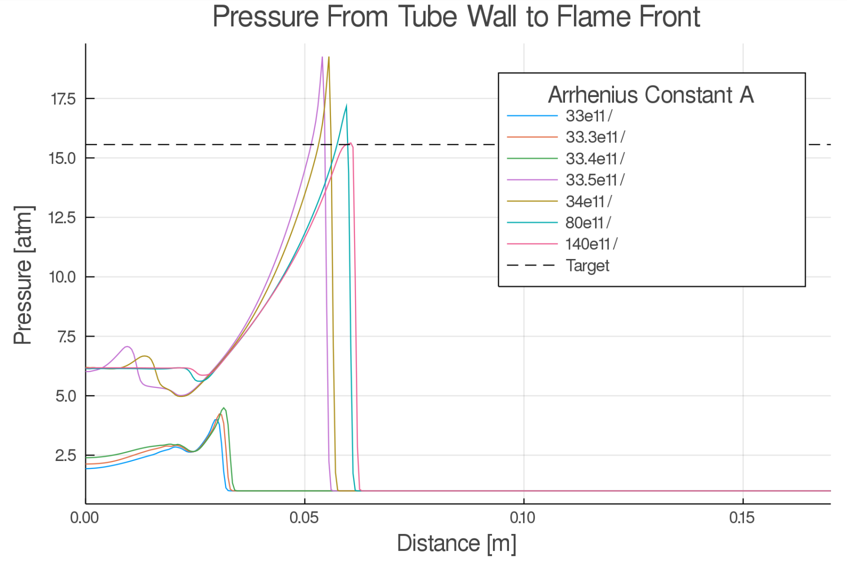
Contains:

- `blockMeshDict`
- `controlDict`
- `decomposeParDict`
- `setFieldsDict`/`funkySetFieldsDict`
- `fvSchemes`
- `fvSolution`
- files defining post-processing line sampling

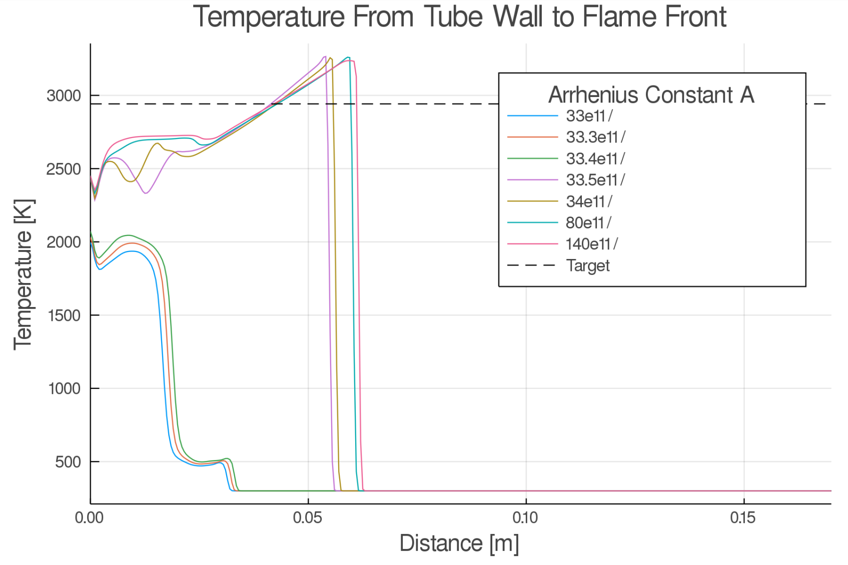
Backup

Arrhenius Pre-exponential Factor Sweep

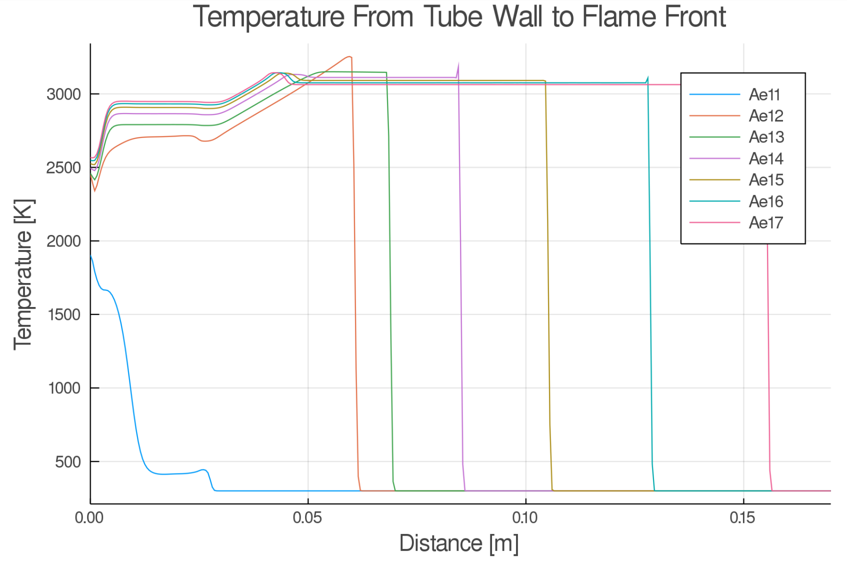
Arrhenius Pre-exponential Factor Sweep, Refined: Pressure Distribution



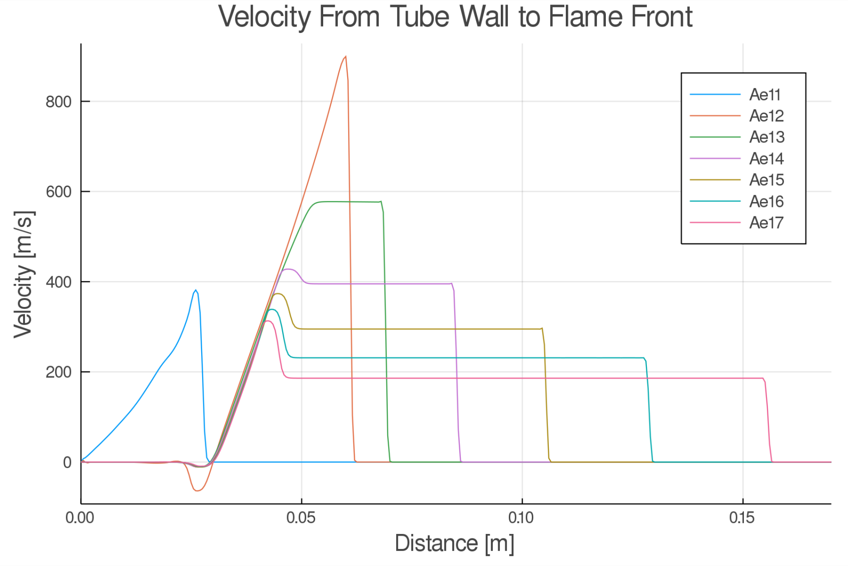
Arrhenius Pre-exponential Factor Sweep, Refined: Temperature Distribution



Arrhenius A Variation: Temperature Distribution



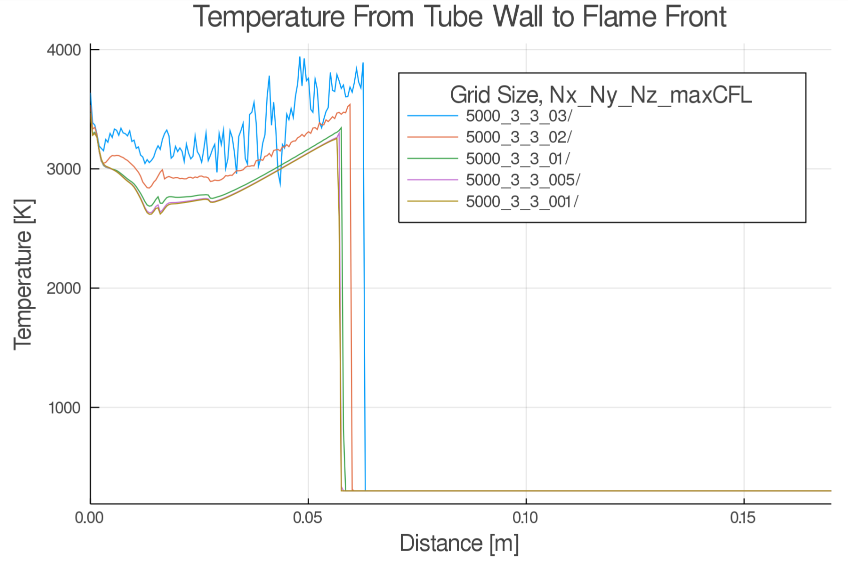
Arrhenius A Variation: Velocity Distribution



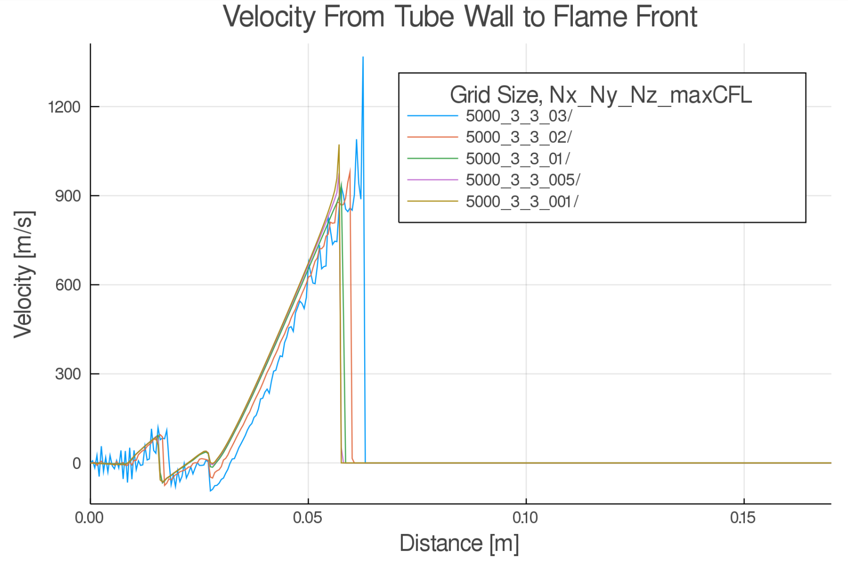
Backup

Time Step Variation

Time Step Variation: Temperature Distribution



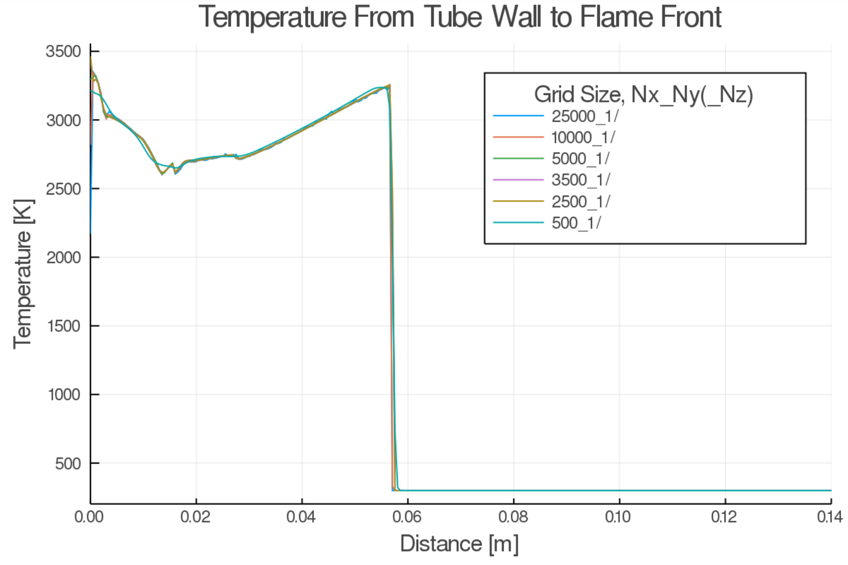
Time Step Variation: Velocity Distribution



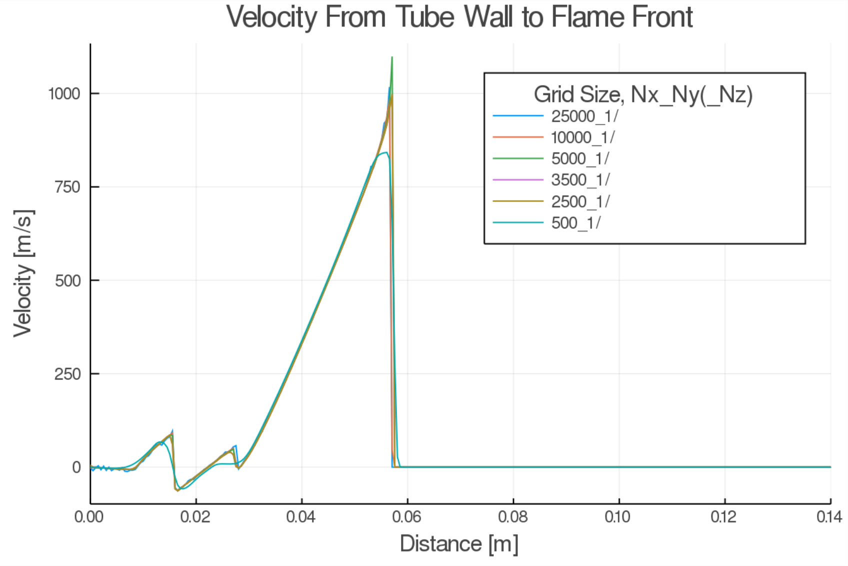
Backup

Static Mesh Variation

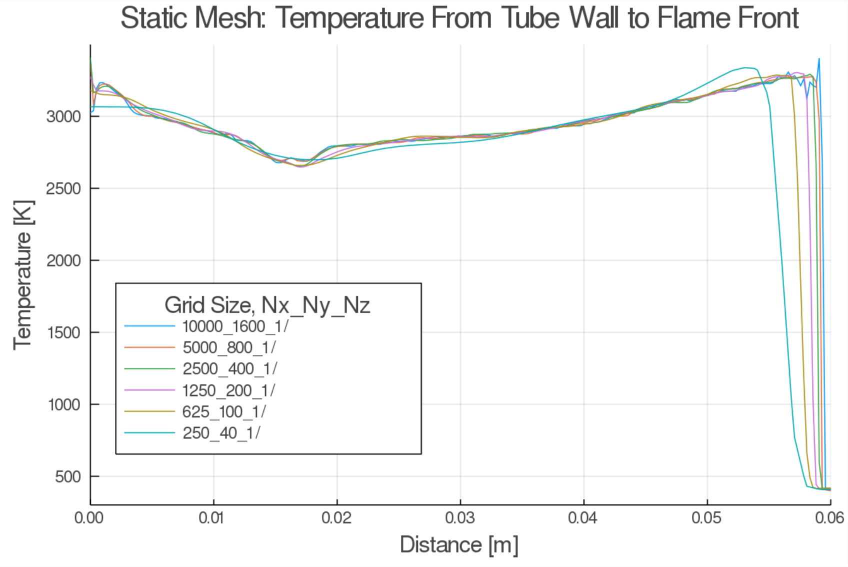
1D Static Mesh Variation: Temperature Distribution



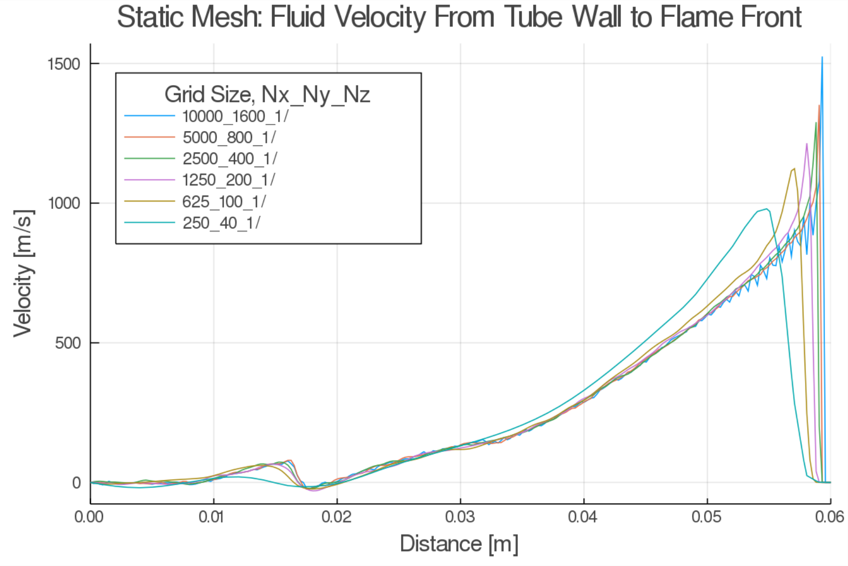
1D Static Mesh Variation: Velocity Distribution



2D Static Mesh Variation: Temperature Distribution



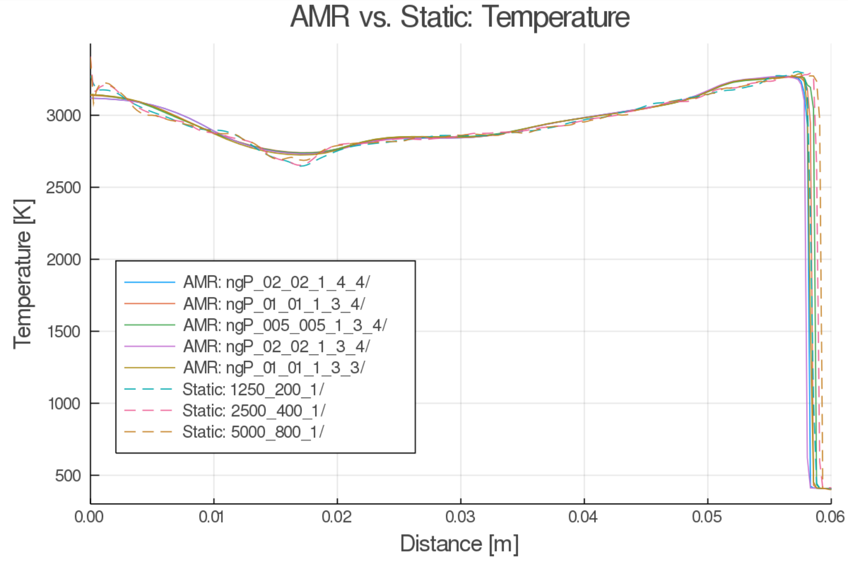
2D Static Mesh Variation: Velocity Distribution



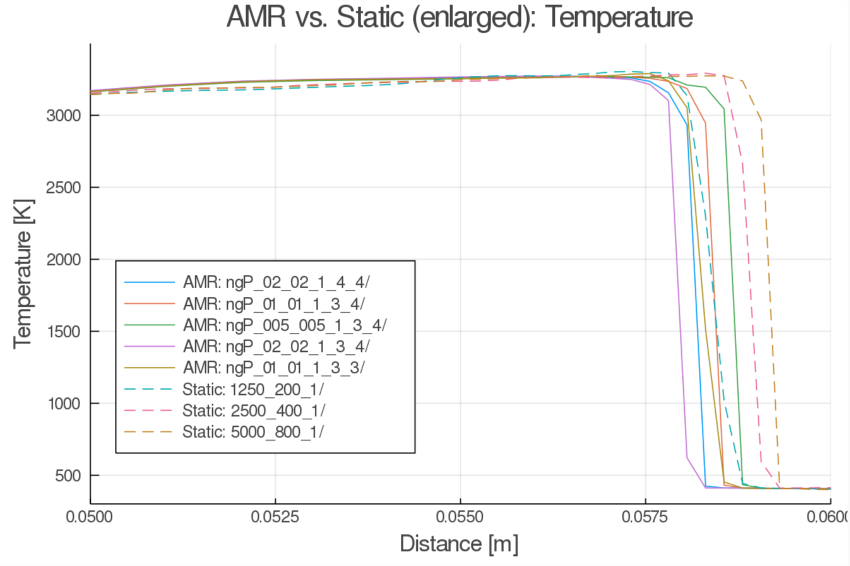
Backup

AMR and Static Mesh Comparison

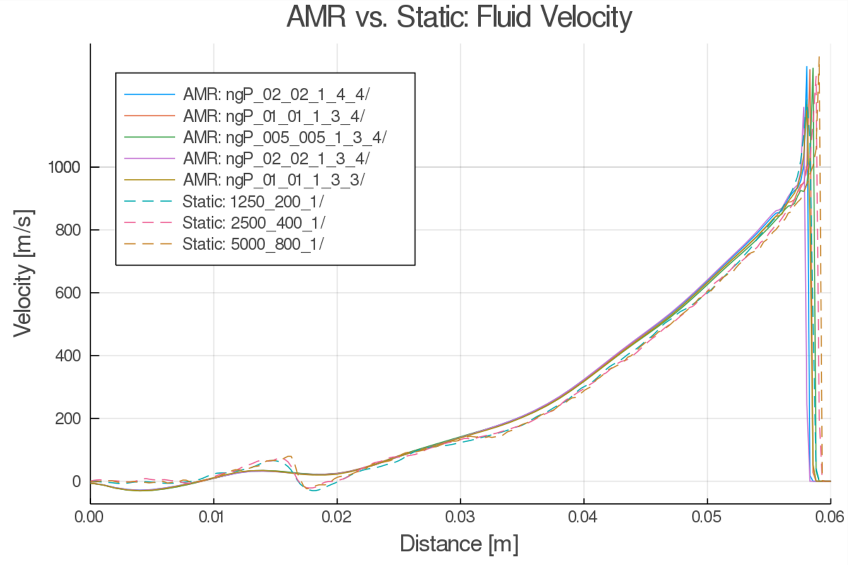
AMR vs. Static: Temperature Distribution



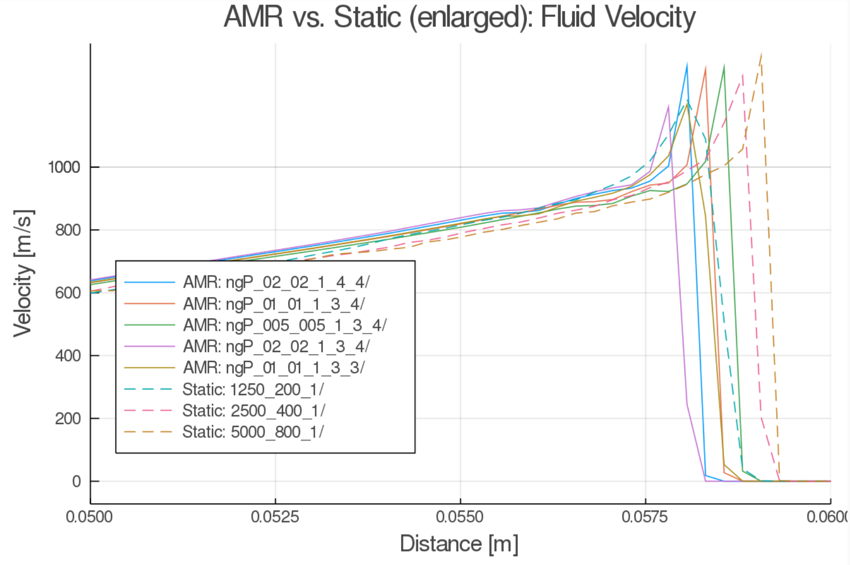
AMR vs. Static: Temperature Distribution (enlarged)



AMR vs. Static: Velocity Distribution



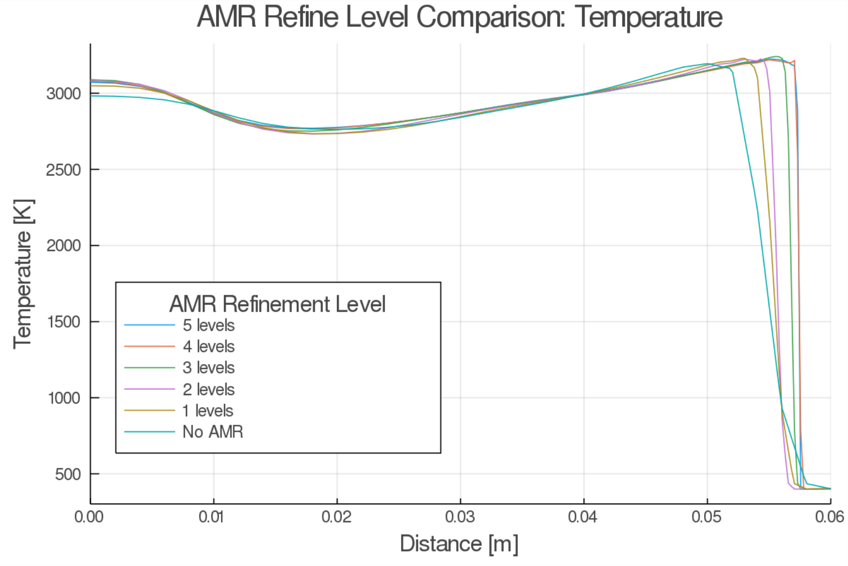
AMR vs. Static: Velocity Distribution (enlarged)



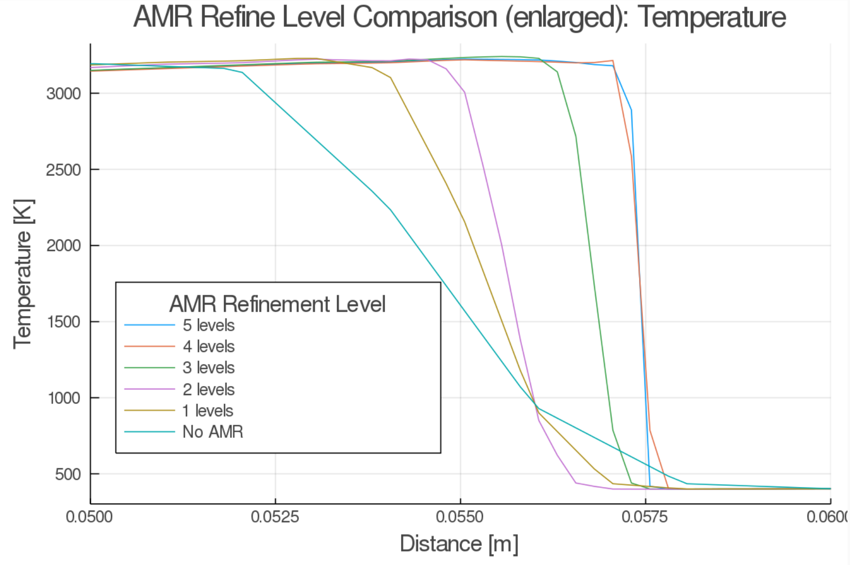
Backup

AMR Refinement Level Variation

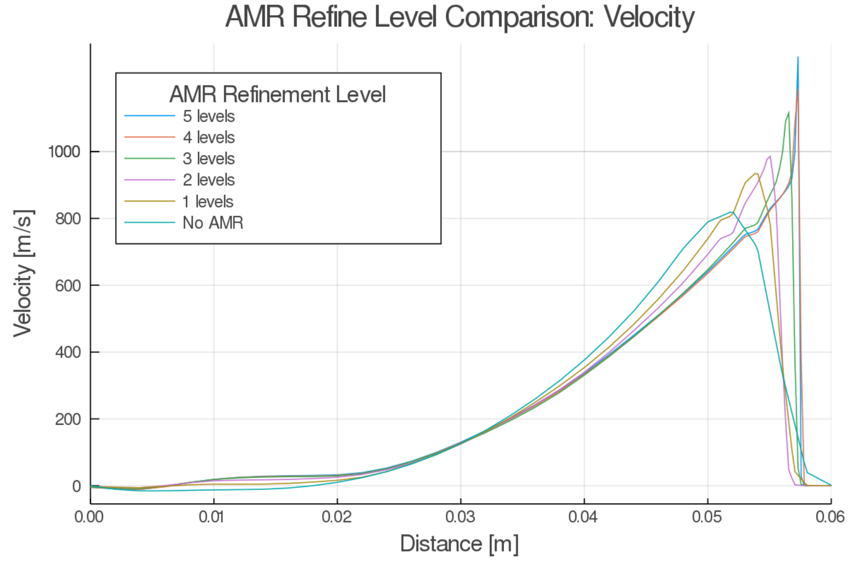
AMR Refinement Level Variation: Temperature Distribution



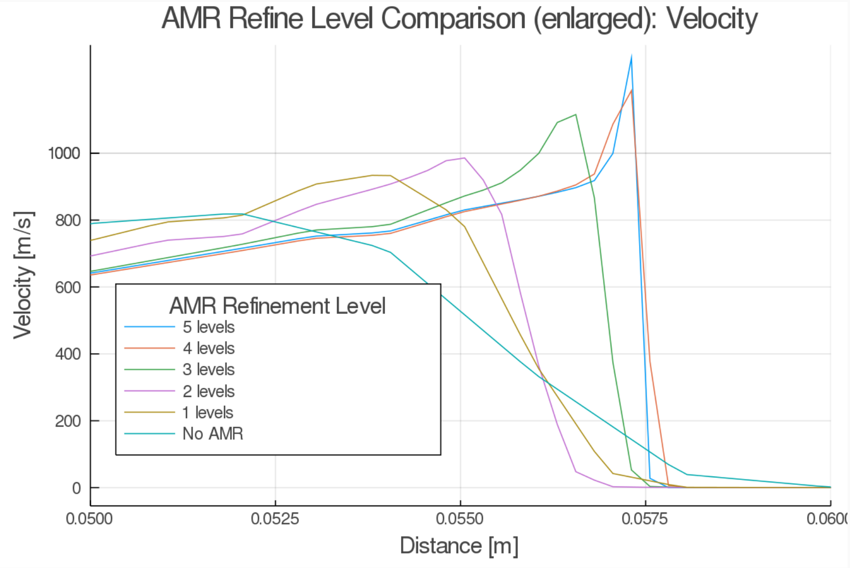
AMR Refinement Level Variation: Temperature Distribution (enlarged)



AMR Refinement Level Variation: Velocity Distribution



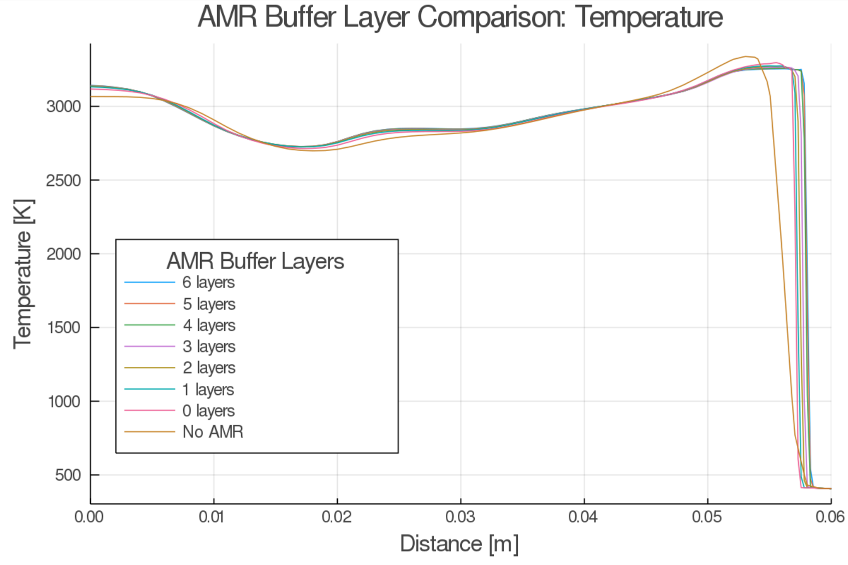
AMR Refinement Level Variation: Velocity Distribution (enlarged)



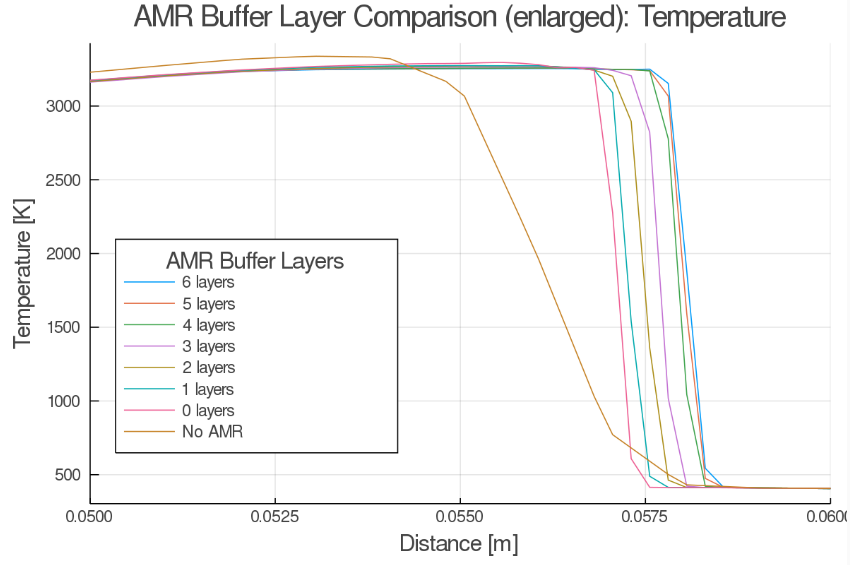
Backup

AMR Buffer Layer Variation

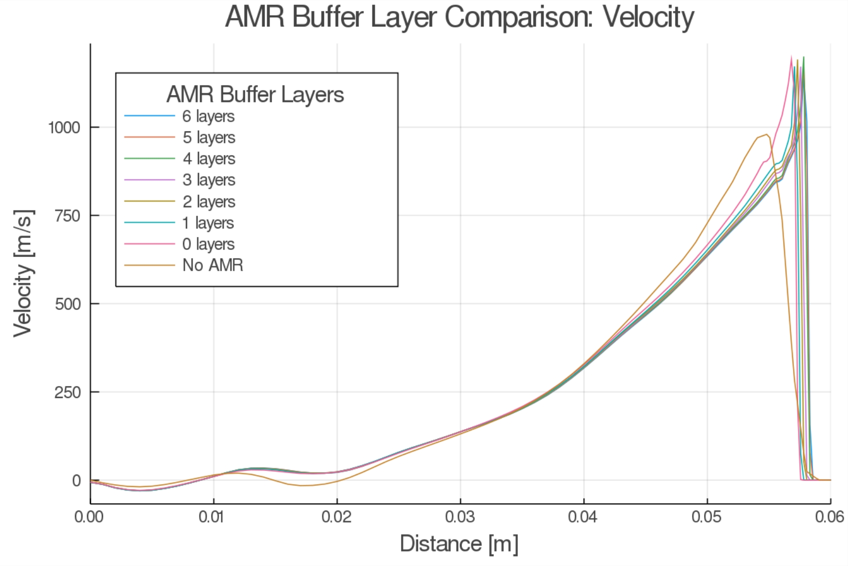
AMR Buffer Layer Variation: Temperature Distribution



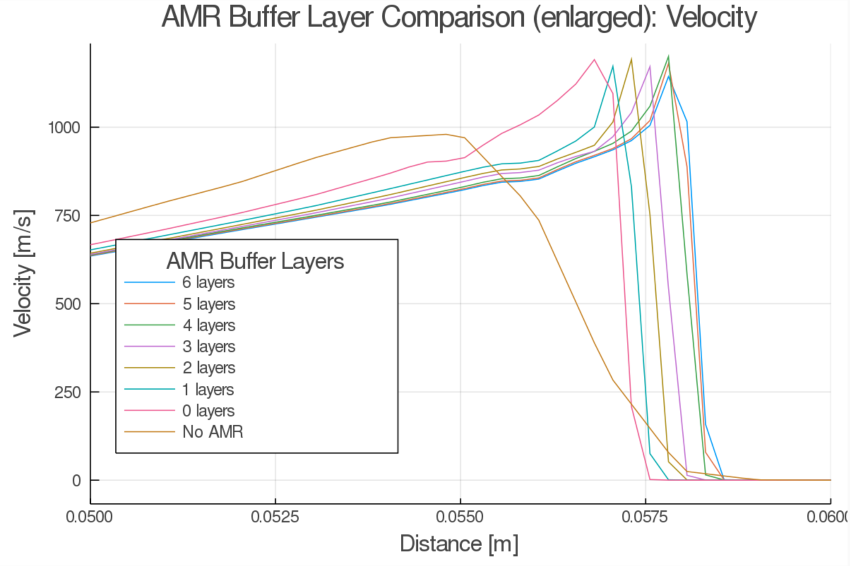
AMR Buffer Layer Variation: Temperature Distribution (enlarged)



AMR Buffer Layer Variation: Velocity Distribution



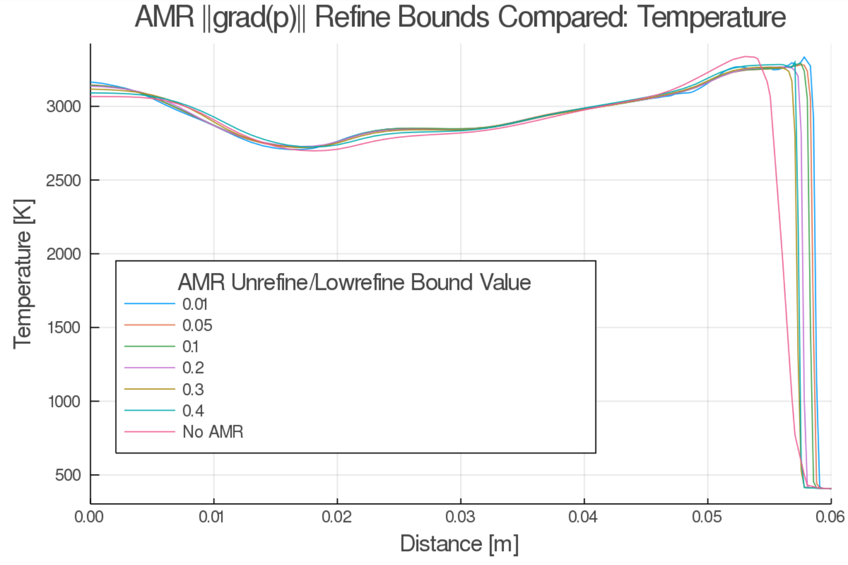
AMR Buffer Layer Variation: Velocity Distribution (enlarged)



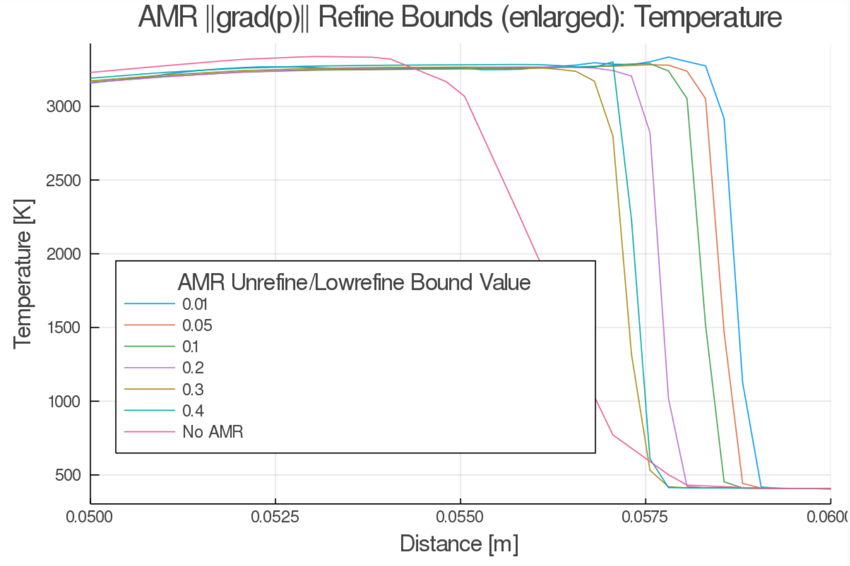
Backup

Refinement Range Variation

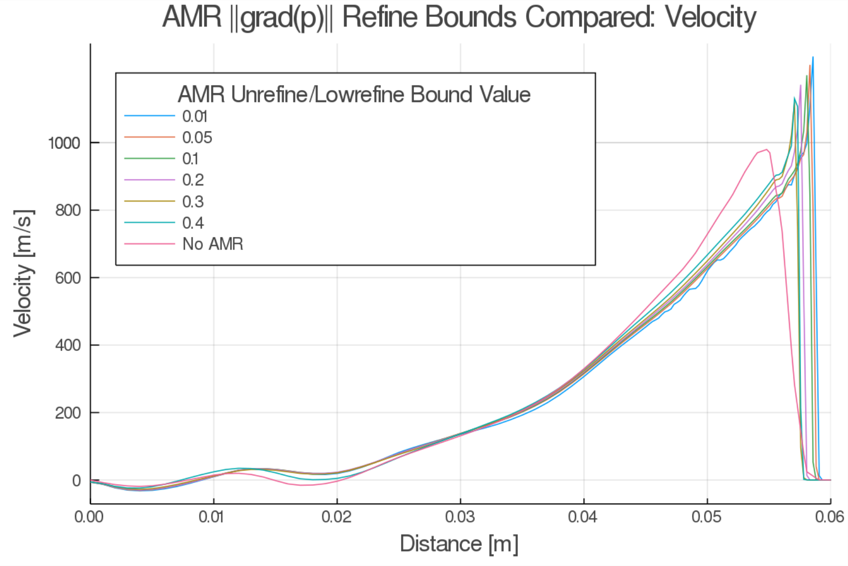
AMR $||\nabla(p)||$ Refine Range Variation: Temperature Distribution



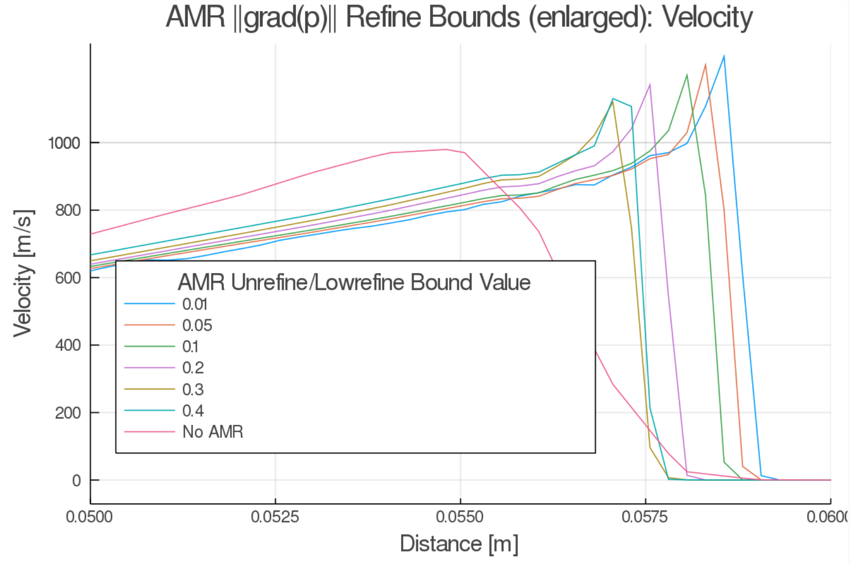
AMR $||\nabla(p)||$ Refine Range Variation: Temperature Distribution (enlarged)



AMR $||\nabla(p)||$ Refine Range Variation: Velocity Distribution



AMR $||\nabla(p)||$ Refine Range Variation: Velocity Distribution (enlarged)



Previous Work

- Towery *et. al.* [3] examined PDEs and RDEs with static grids with inviscid Euler equations using an in-house Fortran parallelized code, also focusing on turbulence modeling
- Marcantoni *et. al.* [20] developed `rhoReactingRfFoam`, a non-AMR OpenFOAM solver for detonations
- Berger and Jameson [21] used and developed adaptive mesh refinement routines to solve the Euler equations
- Deiterding [22] used the Berger and Colella [23] AMR routines to simulate detonations with the Euler equations, developing and implementing parallelization and distributed computing

Previous Work (ii)

- Yi *et. al.* [24] explored numerical simulations of RDEs in three dimensions, exploring overall flow-field and comparing to one- and two-dimensional simulations
- Schwer and Kailasanath [25] examined effects of different hydrocarbon fuels in RDEs as well as numerical methods for simulating them
- Li *et. al.* [26] used the Berger-Oliger AMR routines [27] and simulated detonations, noting the presence of cellular detonation structure
- Kim and Kim [28] simulated confined detonations in OpenFOAM with seven-step chemistry and compared to experimental results

Bibliography

[1] Nagoya University.
900-n ethelene-oxygen rotating detonation engine.
http://www.mae.nagoya-u.ac.jp/flight/research1.pdf.

[2] H Weller, C Greenshields, M Janssens, A Heather, and S Ferraris.
Cj, openfoam®-the open source computational fluid dynamics (cfd) toolbox, 2004.

[3] Colin A. Towery, Katherine M. Smith, Prateek Shrestha, Peter E. Hamlington, and Marthinus Van Schoor.
Examination of Turbulent Flow Effects in Rotating Detonation Engines.

[4] David Leonard Chapman.
Vi. on the rate of explosion in gases.
The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 47(284):90–104, 1899.

[5] Ya. B. Zel'dovich.
On the theory of the propagation of detonations on gaseous system.
Zhurnal Eksperimental'noi i Teoreticheskoi Fiziki, pages 542 – 568, 1940.

[6] J. von Neumann.
Theory of detonation waves. progress report to the national defense research committee div. b, osrd-549 (pb 31090).
John von Neumann: Collected Works, 1903–1957, pages 178 – 218, 1942.

[7] Werner Doring.
On detonation processes in gases.
Annalen der Physik, pages 421 – 436, 1943.

[8] Christopher J Greenshields, Henry G Weller, Luca Gasparini, and Jason M Reese.
Implementation of semi-discrete, non-staggered central schemes in a colocated, polyhedral, finite volume framework, for high-speed viscous flows.
International journal for numerical methods in fluids, 63(1):1–21, 2010.

[9] Alexander Kurganov and Eitan Tadmor.
New high-resolution central schemes for nonlinear conservation laws and convection-diffusion equations.
Journal of Computational Physics, 160:241–282, 2000.

[10] Nakul.
rhocentralspeciesfoam, a new solver for supersonic combustion.
`https://www.cfd-online.com/Forums/openfoam-community-contributions/82851-new-solver-supersonic-combustion.html`.

[11] Kenneth Kuo.
Principles of Combustion.
Wiley, 2nd edition, 2005.

[12] George Gabriel Stokes.
On the steady motion of incompressible fluides.
Transactions of the Cambridge Philosophical Society, 7:439 – 453, 1842.

[13] Jr. M.W. Chase, C.A. Davies, Jr. J.R. Downey, D.J. Frurip, R.A. McDonald, and A.N. Syverud.
Nist-janaf thermochemical tables.
`https://janaf.nist.gov/`, 1998.

[14] William Sutherland.
Lii. the viscosity of gases and molecular force.
The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 36(223):507–531, 1893.

[15] Svante Arrhenius.
Uber die dissociationswarme und den einfluss der temperatur auf den dissociationsgrad der elektrolyte.
Zeitschrift fur Physikalische Chemie, 4U(1):96 – 116, 1889.

[16] Dominik Christ.
Simulating the combustion of gaseous fuels.
`https://openfoamwiki.net/images/8/8a/Christ_OFW6_slides.pdf`.

[17] Jung Rai, Sudeep Parajuli, Arash Abbasi, and Sayed Hashemi.
Investigation of different chemical kinetics reaction rates and its relevant modelling for h2 -o2 combustion.
05 2016.

[18] D W Walker.
Standards for message-passing in a distributed memory environment.
1992.

[19] Colin A.Z. Towery, Alexei Y. Poludnenko, and Peter E. Hamlington.
Detonation initiation by compressible turbulence thermodynamic fluctuations.
Combustion and Flame, 213:172 – 183, 2020.

[20] L.F. Gutiérrez Marcantoni, J. Tamagno, and S. Elaskar.
rhocentralrffoam: An openfoam solver for high speed chemically active flows – simulation of planar detonations –.
Computer Physics Communications, 219:209 – 222, 2017.

[21] Marsha J Berger and Antony Jameson.
Automatic adaptive grid refinement for the euler equations.
AIAA journal, 23(4):561–568, 1985.

[22] Ralf Deiterding.
Parallel adaptive simulation of multi-dimensional detonation structures.
Dissertation. de, 2003.

[23] Marsha J Berger, Phillip Colella, et al.
Local adaptive mesh refinement for shock hydrodynamics.
Journal of computational Physics, 82(1):64–84, 1989.

[24] Tae-Hyeong Yi, Cary Turangan, Jing Lou, Piotr Wolanski, and Jan Kindracki.
A Three-Dimensional Numerical Study of Rotational Detonation in an Annular Chamber.

[25] Douglas Schwer and K. Kailasanath.
Fluid dynamics of rotating detonation engines with hydrogen and hydrocarbon fuels.
Proceedings of the Combustion Institute, 34(2):1991 – 1998, 2013.

[26] Jian Li, Jianguo Ning, Charles B. Kiyanda, and Hoi Dick Ng.
Numerical simulations of cellular detonation diffraction in a stable gaseous mixture.
Propulsion and Power Research, 5(3):177 – 183, 2016.

[27] MJ Berger and J Oliger.
The amr technique.
J. Computat. Phys, 53:484–512, 1984.

[28] D. Kim and J. Kim.
Numerical method to simulate detonative combustion of hydrogen-air mixture in a containment.
Engineering Applications of Computational Fluid Mechanics, 13(1):938–953, 2019.

[29] Chibuikem Ajaero.
Validation of the kurganov-tadmor/kurganov-noelle petrova scheme for rotating detonation engine simulations using openfoam.
https://spectrum.library.concordia.ca/984878/, January 2019.

[30] The OpenFOAM Foundation.
Openfoam v7 user guide.
https://cfd.direct/openfoam/user-guide.

[31] J. Ahrens, Berk Geveci, and Charles Law.
Paraview: An end-user tool for large data visualization.
Visualization Handbook, 01 2005.

[32] Thien Xuan Dinh, Masatake Yoshida, and Shuichi Ishikura.
Simulation of rotating detonation engine by openfoam.
Sci. Tech Energetic Materials, 80(2):68 – 71, 2019.

[33] Alexander Kurganov, Sebastian Noelle, and Guergana Petrova.
Semidiscrete central-upwind schemes for hyperbolic conservation laws and hamilton–jacobi equations.
SIAM Journal on Scientific Computing, 23(3):707–740, 2001.

[34] NASA.
Chemical equilibrium with applications (cea).
https://www.grc.nasa.gov/WWW/CEAWeb/.

[35] swak4foam.
`https://openfoamwiki.net/index.php/Contrib/swak4Foam`.

[36] E. Fehlberg.
Klassische runge-kutta-formeln vierter und niedrigerer ordnung mit schrittweiten-kontrolle und ihre anwendung auf wärmeleitungsprobleme.
Computing, 6(1):61–71, 1970.

[37] B. van Leer.
Towards the ultimate conservative difference scheme, v. a second order sequel to godunov's method.
Journal of Computational Physics, 32:101 – 136, 1979.

[38] H. A. van der Vorst.
Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems.
SIAM Journal on Scientific and Statistical Computing, 13(2):631–644, 1992.

[39] S. K. Godunov.
Eine Differenzenmethode für die Näherungsberechnung unstetiger Lösungen der hydrodynamischen Gleichungen.
Mat. Sb., Nov. Ser., 47:271–306, 1959.

[40] Krzysztof Michalak and Carl Ollivier-Gooch.
Limiters for Unstructured Higher-Order Accurate Solutions of the Euler Equations.

[41] Yousef Saad.
Iterative method for sparse linear systems.
PWS publishing company, a division of international Thomson publishing Inc, 2nd edition, 1996.

[42] A. Schwarzenberg-Czerny.
On matrix factorization and efficient least squares solution.
Astronomy and Astrophysics Supplement,, 110:405, April 1995.

[43] G.H. Golub and C.F. Van Loan.
Matrix Computations.
Johns Hopkins Studies in Atlantic History & Culture. Johns Hopkins University Press, 1983.

[44] T.J Poinsot and S.K Lelef.
Boundary conditions for direct simulations of compressible viscous flows.
Journal of Computational Physics, 101(1):104 – 129, 1992.

[45] R. Courant, K. Friedrichs, and H. Lewy.
Über die partiellen differenzengleichungen der mathematischen physik.
Mathematische Annalen, 100(1):32–74, Dec 1928.

[46] L. Euler.
Institutionum calculi integralis.
Number v. 1 in Institutionum calculi integralis. imp. Acad. imp. Saènt., 1768.

[47] William D Henshaw and Donald W Schwendeman.
An adaptive numerical scheme for high-speed reactive flow on overlapping grids.
Journal of Computational Physics, 191(2):420–447, 2003.

[48] D.A. Kessler, V.N. Gamezo, and E.S. Oran.
Multilevel detonation cell structures in methane-air mixtures.
Proceedings of the Combustion Institute, 33(2):2211 – 2218, 2011.

[49] Emile Jouguet.
Sur la propagation des réactions chimiques dans les gaz.
J. Maths. Pure Appl., 7:347, 1905.

[50] Simscale: Cloud-based hpc simulation platform. `https://www.simscale.com/`, **2020.**

[51] SIMFLOW Technologies. **simflow: Computational fluid dynamics software.** `https://sim-flow.com/`, 2020.

[52] SIMFLOW Technologies. **Rapidcfd: Openfoam running on a gpu.** `https://github.com/Atizar/RapidCFD-dev`, 2020.

[53] Nvidia. **Cuda: a parallel computing platform and programming model.** `https://developer.nvidia.com/cuda-zone`, 2020.

[54] Carl Friedrich Gauss.
Methodus nova integralium valores per approximationem inveniendi.
apvd Henricvm Dieterich, 1815.