

Structure of the lectures

Block-structured Adaptive Finite Volume Methods in C++

The AMROC Framework for Parallel AMR and Shock-Induced Combustion Simulation

Short course at Xiamen University, 18th July to 22nd July 2016

Ralf Deiterding
University of Southampton
Engineering and the Environment
Highfield Campus, Southampton SO17 1BJ, UK
E-mail: r.deiterding@soton.ac.uk

1. Fundamentals (18th)
 - ▶ Conservation laws
 - ▶ Finite volume methods
 - ▶ Upwind schemes
2. Structured AMR for hyperbolic problems (19th)
 - ▶ Meshes and adaptation
 - ▶ Presentation of all algorithmic components
 - ▶ Parallelization
3. Hyperbolic AMROC solvers (20th)
 - ▶ Higher-order methods
 - ▶ AMROC design
 - ▶ Clawpack and WENO solvers in AMROC
4. Numerical methods for combustion research (20th)
 - ▶ Consideration of non-Cartesian geometries
 - ▶ Numerical methods for the inviscid reactive equations

Structure of the lectures - II

5. Detonation simulation (21th)
 - ▶ Examples of ignition and detonation structure simulation
 - ▶ Extensions to viscous reactive equations
6. Fluid-structure interaction (FSI) simulation (22th)
 - ▶ Examples of detonation-driven FSI
7. Lattice Boltzmann methods (22th)
 - ▶ Adaptive LBM
 - ▶ Aerodynamics computations
8. Further topics and software demo of AMROC (22th)
 - ▶ Adaptive multigrid method
 - ▶ Installation of AMROC on Linux
 - ▶ Running examples

Useful references I

Finite volume methods for hyperbolic problems

- ▶ LeVeque, R. J. (2002). *Finite volume methods for hyperbolic problems*. Cambridge University Press, Cambridge, New York.
- ▶ Godlewski, E. and Raviart, P.-A. (1996). *Numerical approximation of hyperbolic systems of conservation laws*. Springer Verlag, New York.
- ▶ Toro, E. F. (1999). *Riemann solvers and numerical methods for fluid dynamics*. Springer-Verlag, Berlin, Heidelberg, 2nd edition.
- ▶ Laney, C. B. (1998). *Computational gasdynamics*. Cambridge University Press, Cambridge.

Structured Adaptive Mesh Refinement

- ▶ Berger, M. and Colella, P. (1988). Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82:64–84.
- ▶ Bell, J., Berger, M., Saltzman, J., and Welcome, M. (1994). Three-dimensional adaptive mesh refinement for hyperbolic conservation laws. *SIAM J. Sci. Comp.*, 15(1):127–138.
- ▶ Berger, M. and LeVeque, R. (1998). Adaptive mesh refinement using wave-propagation algorithms for hyperbolic systems. *SIAM J. Numer. Anal.*, 35(6):2298–2316.

Useful references II

- ▶ Deiterding, R. (2011). Block-structured adaptive mesh refinement - theory, implementation and application, *Series in Applied and Industrial Mathematics: Proceedings*, 34: 97–150.

Combustion, detonations and shockwave theory

- ▶ Williams, F. A. (1985). *Combustion theory*, Addison-Wesley, Reading.
- ▶ Fickett, W. and Davis, W. C. (1979). *Detonation*, University of California Press, Berkeley and Los Angeles, California.
- ▶ Ben-Dor, G. (2007). *Shock wave reflection phenomena*, Springer, Berlin.

Shock-capturing schemes for combustion

- ▶ Grossmann, B. and Cinella, P. (1990). Flux-split algorithms for flows with non-equilibrium chemistry and vibrational relaxation. *J. Comput. Phys.*, 88:131–168.
- ▶ Fedkiw, R. P., Merriman, B. and Osher, S. (1997). High accuracy numerical methods for thermally perfect gas flows with chemistry. *J. Comput. Phys.*, 132:175–190.
- ▶ Deiterding, R. (2003). *Parallel adaptive simulation of multi-dimensional detonation structures*. PhD thesis, Brandenburgische Technische Universität Cottbus.

Useful references III

- ▶ Deiterding, R. (2009). A parallel adaptive method for simulating shock-induced combustion with detailed chemical kinetics in complex domains. *Computers & Structures*, 87:769–783.
- ▶ Ziegler, J. L., Deiterding, R., Shepherd, J. E. and Pullin, D. I. (2011). An adaptive high-order hybrid scheme for compressive, viscous flows with detailed chemistry. *J. Comput. Phys.*, 230(20): 7598–7630.

Lattice-Boltzmann methods

- ▶ Succi, S. (2001). *The Lattice Boltzmann Equation for Fluid Dynamics and Beyond*. Oxford Science Publications.
- ▶ Guo, Z., Shu, C. (2013). *Lattice Boltzmann Method and Its Applications in Engineering*, World Scientific.
- ▶ Hähnel, D. (2004). *Molekulare Gasdynamik*, Springer.
- ▶ Aidun, C. K., Clausen, J. A. (2010). Lattice-Boltzmann method for complex flows. *Annu. Rev. Fluid Mech.*, 42: 439–472.

Adaptive multigrid (finite difference and finite element based in textbooks)

- ▶ Hackbusch, W. (1985). *Multi-Grid Methods and Applications*. Springer Verlag, Berlin, Heidelberg.

Useful references IV

- ▶ Briggs, W. L., Henson, V. E., and McCormick, S. F. (2001). *A Multigrid Tutorial*. Society for Industrial and Applied Mathematics, 2nd edition.
- ▶ Trottenberg, U., Oosterlee, C., and Schüller, A. (2001). *Multigrid*. Academic Press, San Antonio.
- ▶ Martin, D. F. (1998). *A cell-centered adaptive projection method for the incompressible Euler equations*. PhD thesis, University of California at Berkeley.

Fluid-structure interaction and further applications (from my own work only)

- ▶ Deiterding, R. and Wood, S (2013). Parallel adaptive fluid-structure interaction simulation of explosions impacting on building structures. *Computers & Fluids*, 88: 719–729.
- ▶ Deiterding, R., Radovitzky, R., Mauch, S. P., Noels, L., Cummings, J. C., and Meiron, D. I. (2006). A virtual test facility for the efficient simulation of solid materials under high energy shock-wave loading. *Engineering with Computers*, 22(3-4):325–347.
- ▶ Pantano, C., Deiterding, R., Hill, D. J., and Pullin, D. I. (2007). A low-numerical dissipation patch-based adaptive mesh refinement method for large-eddy simulation of compressible flows. *J. Comput. Phys.*, 221(1):63–87.

Useful references V

- ▶ Barton, P. T., Deiterding, R. and Meiron, D. I. and Pullin, D. I. (2013). Eulerian adaptive finite-difference method for high-velocity impact and penetration problems, *J. Comput. Phys.*, 240: 76–99.
- ▶ Perotti, L. E., Deiterding, R., Inaba, D. K., Shepherd, J. E. and Ortiz, M. (2013). Elastic response of water-filled fiber composite tubes under shock wave loading, *Int. J. Solids and Structures*, 50: 473–486.
- ▶ Gomes, A. K. F., Domingues, M. O., Schneider, K., Mendes, O., Deiterding, R. (2015). An adaptive multiresolution method for ideal magnetohydrodynamics using divergence cleaning with parabolic-hyperbolic correction. *Applied Numerical Mathematics* 95: 199–213.
- ▶ Deiterding, R. and Wood, S. L. (2015). A dynamically adaptive lattice Boltzmann method for predicting wake phenomena in fully coupled wind engineering problems. *IV Int. Conf. on Coupled Problems in Science and Engineering* 489–500.

Implementation, parallelization

- ▶ Hornung, R. D., Wissink, A. M., and Kohn, S. H. (2006). Managing complex data and geometry in parallel structured AMR applications. *Engineering with Computers*, 22:181–195.

- ▶ Rendleman, C. A., Beckner, V. E., Lijewski, M., Crutchfield, W., and Bell, J. B. (2000). Parallelization of structured, hierarchical adaptive mesh refinement algorithms. *Computing and Visualization in Science*, 3:147–157.
- ▶ Deiterding, R. (2005). Construction and application of an AMR algorithm for distributed memory computers. In Plewa, T., Linde, T., and Weirs, V. G., editors, *Adaptive Mesh Refinement - Theory and Applications*, volume 41 of *Lecture Notes in Computational Science and Engineering*, pages 361–372. Springer.

Lecture 1 Fundamentals

Course *Block-structured Adaptive Finite Volume Methods in C++*

Ralf Deiterding

University of Southampton

Engineering and the Environment

Highfield Campus, Southampton SO17 1BJ, UK

E-mail: r.deiterding@soton.ac.uk

Block-structured Adaptive Finite Volume Methods in C++
Conservation laws
ooooooooooooooo

Finite volume methods
ooooo

Upwind schemes
oooooooooooo

9
References
oo

Fundamentals
Conservation laws
ooooooooooooooo

Finite volume methods
ooooo

Upwind schemes
oooooooooooo

1
References
oo

Outline

Conservation laws

- Mathematical background
- Characteristic information
- Weak and entropy solutions
- Characteristic form of the Euler equations
- Navier-Stokes equations

Finite volume methods

- Basics of finite difference methods
- Splitting methods, second derivatives

Upwind schemes

- The linear Riemann problem
- Flux-difference splitting
- Flux-vector splitting

Outline

Conservation laws

- Mathematical background
- Characteristic information
- Weak and entropy solutions
- Characteristic form of the Euler equations
- Navier-Stokes equations

Finite volume methods

- Basics of finite difference methods
- Splitting methods, second derivatives

Upwind schemes

- The linear Riemann problem
- Flux-difference splitting
- Flux-vector splitting

Hyperbolic Conservation Laws

$$\frac{\partial}{\partial t} \mathbf{q}(\mathbf{x}, t) + \sum_{n=1}^d \frac{\partial}{\partial x_n} \mathbf{f}_n(\mathbf{q}(\mathbf{x}, t)) = \mathbf{s}(\mathbf{q}(\mathbf{x}, t)), \quad D \subset \{(\mathbf{x}, t) \in \mathbb{R}^d \times \mathbb{R}_0^+\} \quad (1)$$

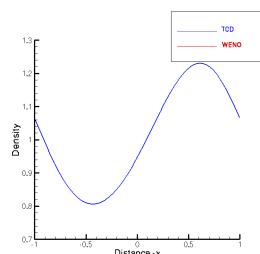
$\mathbf{q} = \mathbf{q}(\mathbf{x}, t) \in S \subset \mathbb{R}^M$ - vector of state, $\mathbf{f}_n(\mathbf{q}) \in C^1(S, \mathbb{R}^M)$ - flux functions, $\mathbf{s}(\mathbf{q}) \in C^1(S, \mathbb{R}^M)$ - source term

Definition (Hyperbolicity)

$\mathbf{A}(\mathbf{q}, \nu) = \nu_1 \mathbf{A}_1(\mathbf{q}) + \dots + \nu_d \mathbf{A}_d(\mathbf{q})$ with $\mathbf{A}_n(\mathbf{q}) = \partial \mathbf{f}_n(\mathbf{q}) / \partial \mathbf{q}$ has M real eigenvalues $\lambda_1(\mathbf{q}, \nu) \leq \dots \leq \lambda_M(\mathbf{q}, \nu)$ and M linear independent right eigenvectors $\mathbf{r}_m(\mathbf{q}, \nu)$.

If $\mathbf{f}_n(\mathbf{q})$ is nonlinear, classical solutions $\mathbf{q}(\mathbf{x}, t) \in C^1(D, S)$ do not generally exist, not even for $\mathbf{q}_0(\mathbf{x}) \in C^1(\mathbb{R}^d, S)$ [Majda, 1984], [Godlewski and Raviart, 1996], [Kröner, 1997]

Example: Euler equations



Characteristic variables - II

Multiplying (2) with \mathbf{R}^{-1} gives

$$\mathbf{R}^{-1} \frac{\partial \mathbf{q}}{\partial t} + \mathbf{R}^{-1} \mathbf{A} \frac{\partial \mathbf{q}}{\partial x} = 0$$

with $\mathbf{R}^{-1} d\mathbf{q} = d\mathbf{v}$ this becomes

$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{R}^{-1} \mathbf{A} \mathbf{R} \frac{\partial \mathbf{v}}{\partial x} = 0$$

or

$$\frac{\partial \mathbf{v}}{\partial t} + \Lambda \frac{\partial \mathbf{v}}{\partial x} = 0$$

which is just a set of decoupled independent advection equations for the components, i.e.,

$$\frac{\partial \mathbf{v}_m}{\partial t} + \lambda_m \frac{\partial \mathbf{v}_m}{\partial x} = 0 \quad \text{for } m = 1, \dots, M \quad (3)$$

(3) is a wave equation but note that in the general quasi-linear case the eigenvalues can depend on all v_m , i.e. $\lambda_m = \lambda_m(v_1, \dots, v_M)$. Nevertheless, an analysis as for the wave equations shows

$$\mathbf{v}_m = \text{const.} \quad \text{for } \frac{dx}{dt} = \lambda_m$$

Characteristic variables

Consider the first-order partial differential equation

$$\frac{\partial \mathbf{q}}{\partial t} + \mathbf{A}(\mathbf{q}) \frac{\partial \mathbf{q}}{\partial x} = 0 \quad (2)$$

For $\mathbf{A} = \text{const.}$ Eq. (2) is called linear, for $\mathbf{A} = \mathbf{A}(\mathbf{q}(x, t))$ it is called quasi-linear. For a hyperbolic system, \mathbf{A} is diagonalizable as

$$\mathbf{R}^{-1} \mathbf{A} \mathbf{R} = \Lambda$$

\mathbf{R} is the matrix of right eigenvectors (column-wise)

$$\mathbf{R} = (\mathbf{r}_1 | \dots | \mathbf{r}_M)$$

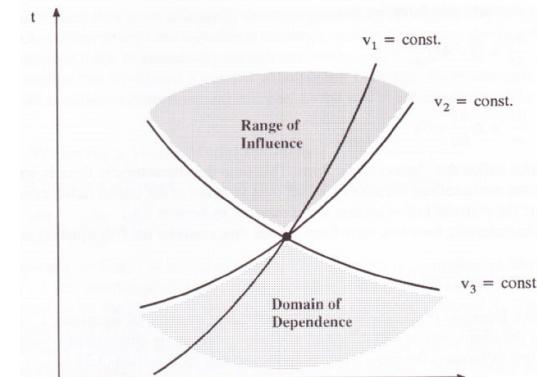
and Λ the diagonal matrix of eigenvalues

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_M \end{pmatrix}$$

Wavefronts

The curves $dx = \lambda_m dt$ are called wavefronts or characteristics, v_m are the characteristic variables.

The characteristics define how influence spreads in the $x - t$ plane. A point in the $x - t$ plane is only influenced by points at earlier times in a finite domain of dependence and influences only points in a finite range of influence.



Typical wave diagram for vector model problem.

Weak solutions

Integral form (Gauss's theorem):

$$\int_{\Omega} \mathbf{q}(\mathbf{x}, t + \Delta t) d\mathbf{x} - \int_{\Omega} \mathbf{q}(\mathbf{x}, t) d\mathbf{x} + \sum_{n=1}^d \int_t^{t+\Delta t} \int_{\partial\Omega} \mathbf{f}_n(\mathbf{q}(\mathbf{o}, t)) \sigma_n(\mathbf{o}) d\mathbf{o} dt = \int_t^{t+\Delta t} \int_{\Omega} \mathbf{s}(\mathbf{q}(\mathbf{x}, t)) d\mathbf{x}$$

Theorem (Weak solution)

$\mathbf{q}_0 \in L_{loc}^\infty(\mathbb{R}^d, S)$. $\mathbf{q} \in L_{loc}^\infty(D, S)$ is weak solution if \mathbf{q} satisfies

$$\int_0^\infty \int_{\mathbb{R}^d} \left[\frac{\partial \varphi}{\partial t} \cdot \mathbf{q} + \sum_{n=1}^d \frac{\partial \varphi}{\partial x_n} \cdot \mathbf{f}_n(\mathbf{q}) - \varphi \cdot \mathbf{s}(\mathbf{q}) \right] d\mathbf{x} dt + \int_{\mathbb{R}^d} \varphi(\mathbf{x}, 0) \cdot \mathbf{q}_0(\mathbf{x}) d\mathbf{x} = 0$$

for any test function $\varphi \in C_0^1(D, S)$

Entropy solutions

Select physical weak solution as $\lim_{\varepsilon \rightarrow 0} \mathbf{q}_\varepsilon = \mathbf{q}$ almost everywhere in D of

$$\frac{\partial \mathbf{q}_\varepsilon}{\partial t} + \sum_{n=1}^d \frac{\partial \mathbf{f}_n(\mathbf{q}_\varepsilon)}{\partial x_n} - \varepsilon \sum_{n=1}^d \frac{\partial^2 \mathbf{q}_\varepsilon}{\partial x_n^2} = \mathbf{s}(\mathbf{q}_\varepsilon), \quad \mathbf{x} \in \mathbb{R}^d, \quad t > 0$$

Theorem (Entropy condition)

Assume existence of entropy $\eta \in C^2(S, \mathbb{R})$ and entropy fluxes $\psi_n \in C^1(S, \mathbb{R})$ that satisfy

$$\frac{\partial \eta(\mathbf{q})}{\partial \mathbf{q}}^T \cdot \frac{\partial \mathbf{f}_n(\mathbf{q})}{\partial \mathbf{q}} = \frac{\partial \psi_n(\mathbf{q})}{\partial \mathbf{q}}^T, \quad n = 1, \dots, d$$

then $\lim_{\varepsilon \rightarrow 0} \mathbf{q}_\varepsilon = \mathbf{q}$ almost everywhere in D is weak solution and satisfies

$$\frac{\partial \eta(\mathbf{q})}{\partial t} + \sum_{n=1}^d \frac{\partial \psi_n(\mathbf{q})}{\partial x_n} \leq \frac{\partial \eta(\mathbf{q})}{\partial \mathbf{q}}^T \cdot \mathbf{s}(\mathbf{q})$$

in the sense of distributions. Proof: [Godlewski and Raviart, 1996]

Rankine-Hugoniot relations

Consider the 1d version of (1), $\mathbf{s}(\mathbf{q}) = 0$ integrated over interval $[x, x + dx] \times [t, t + dt]$

$$\int_x^{x+dx} \mathbf{q}(x', t + dt) dx' - \int_x^{x+dx} \mathbf{q}(x', t) dx' = - \int_t^{t+dt} [\mathbf{f}(\mathbf{q}(x + dx, t')) - \mathbf{f}(\mathbf{q}(x, t'))] dt'$$

Assume a discontinuity traveling with speed

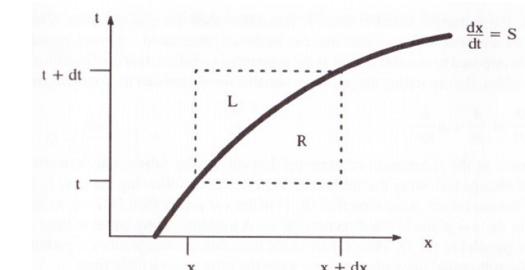
$$S = \frac{dx}{dt}$$

State on the left of discontinuity is index with L, on the right with R Inserting the states into (8) gives

$$(\mathbf{q}_L - \mathbf{q}_R) dx = -[\mathbf{f}(\mathbf{q}_R) - \mathbf{f}(\mathbf{q}_L)] dt$$

Or using the above speed definition

$$S(\mathbf{q}_R - \mathbf{q}_L) = \mathbf{f}(\mathbf{q}_R) - \mathbf{f}(\mathbf{q}_L)$$



This is called Rankine-Hugoniot jump relation. Note the form $\mathbf{f}(\mathbf{q}_R) = \mathbf{f}(\mathbf{q}_L)$ for $S = 0$ from which, for instance, the shock relations for Euler equations are derived.

Entropy solutions II

Definition (Entropy solution)

Weak solution \mathbf{q} is called an entropy solution if \mathbf{q} satisfies

$$\int_0^\infty \int_{\mathbb{R}^d} \left[\frac{\partial \varphi}{\partial t} \eta(\mathbf{q}) + \sum_{n=1}^d \frac{\partial \varphi}{\partial x_n} \psi_n(\mathbf{q}) - \varphi \frac{\partial \eta(\mathbf{q})}{\partial \mathbf{q}}^T \cdot \mathbf{s}(\mathbf{q}) \right] d\mathbf{x} dt + \int_{\mathbb{R}^d} \varphi(\mathbf{x}, 0) \eta(\mathbf{q}_0(\mathbf{x})) d\mathbf{x} \geq 0$$

for all entropy functions $\eta(\mathbf{q})$ and all test functions $\varphi \in C_0^1(D, \mathbb{R}_0^+)$, $\varphi \geq 0$

Theorem (Jump conditions)

An entropy solution \mathbf{q} is a classical solution $\mathbf{q} \in C^1(D, S)$ almost everywhere and satisfies the Rankine-Hugoniot (RH) jump condition

$$(\mathbf{q}^+ - \mathbf{q}^-) \sigma_t + \sum_{n=1}^d (\mathbf{f}_n(\mathbf{q}^+) - \mathbf{f}_n(\mathbf{q}^-)) \sigma_n = \mathbf{0}$$

and the jump inequality

$$(\eta(\mathbf{q}^+) - \eta(\mathbf{q}^-)) \sigma_t + \sum_{n=1}^d (\psi_n(\mathbf{q}^+) - \psi_n(\mathbf{q}^-)) \sigma_n \leq 0$$

along discontinuities. Proof: [Godlewski and Raviart, 1996]

Euler equations

$$\begin{aligned}\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_n} (\rho u_n) &= 0 \\ \frac{\partial}{\partial t} (\rho u_k) + \frac{\partial}{\partial x_n} (\rho u_k u_n + \delta_{kn} p) &= 0, \quad k = 1, \dots, d \\ \frac{\partial}{\partial t} (\rho E) + \frac{\partial}{\partial x_n} (u_n (\rho E + p)) &= 0\end{aligned}$$

with polytrope gas equation of state

$$p = (\gamma - 1)(\rho E - \frac{1}{2} \rho u_n u_n)$$

have structure

$$\partial_t \mathbf{q}(\mathbf{x}, t) + \nabla \cdot \mathbf{f}(\mathbf{q}(\mathbf{x}, t)) = 0$$

Characteristic form of the Euler equations

The Jacobian can be written in different forms, using

$$a^2 = \gamma \frac{p}{\rho}, \quad h = e + \frac{p}{\rho}, \quad H = h + \frac{1}{2} u^2 \Rightarrow H = \frac{a^2}{\gamma - 1} + \frac{1}{2} u^2$$

For

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ \frac{\gamma - 3}{2} u^2 & (3 - \gamma) u & \gamma - 1 \\ -uH + \frac{1}{2} (\gamma - 1) u^3 & H - (\gamma - 1) u^2 & \gamma u \end{bmatrix}$$

The matrices

$$\mathbf{R} = \begin{bmatrix} 1 & 1 & 1 \\ u - a & u & u + a \\ H - ua & \frac{1}{2} u^2 & H + ua \end{bmatrix}$$

$$\mathbf{R}^{-1} = \frac{1}{2a^2} \begin{bmatrix} \frac{1}{2}(\gamma - 1)u^2 + ua & (1 - \gamma)u - a & \gamma - 1 \\ 2a^2 - (\gamma - 1)u^2 & 2(\gamma - 1)u & 2(1 - \gamma) \\ \frac{1}{2}(\gamma - 1)u^2 - ua & (1 - \gamma)u + a & \gamma - 1 \end{bmatrix}$$

diagonalize \mathbf{A} as

Characteristic form of the Euler equations - II

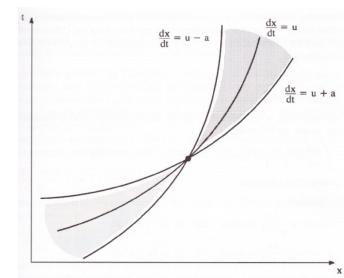
$$\mathbf{R}^{-1} \mathbf{A} \mathbf{R} = \Lambda = \begin{pmatrix} u - a & 0 & 0 \\ 0 & u & 0 \\ 0 & 0 & u + a \end{pmatrix}$$

The transformation $\mathbf{R}^{-1} d\mathbf{q} = \mathbf{R}^{-1} (d\rho, d(\rho u), d(\rho E))^T$ into characteristic variables therefore leads to

$$\begin{aligned}\frac{\partial \mathbf{v}^-}{\partial t} + (u - a) \frac{\partial \mathbf{v}^-}{\partial x} &= 0 \\ \frac{\partial \mathbf{v}_0}{\partial t} + u \frac{\partial \mathbf{v}_0}{\partial x} &= 0 \\ \frac{\partial \mathbf{v}^+}{\partial t} + (u + a) \frac{\partial \mathbf{v}^+}{\partial x} &= 0\end{aligned}$$

with

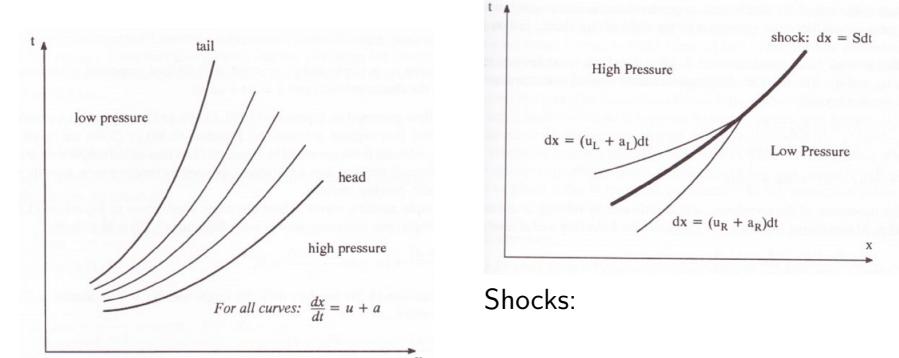
$$\begin{aligned}dv^- = du - \frac{dp}{\rho a} &= 0 \quad \text{for } dx = (u - a)dt \\ dv_0 = d\rho - \frac{dp}{a^2} &= 0 \quad \text{for } dx = u dt \\ dv^+ = du + \frac{dp}{\rho a} &= 0 \quad \text{for } dx = (u + a)dt\end{aligned}$$



The crossing of characteristics causes a shock wave.

Rarefaction and shock waves in the $x - t$ plane

Consider the two enclosing characteristics $b_1(t) \leq x \leq b_2(t)$



Shocks:

Rarefaction:

$$u(x, t) \pm a(x, t) \geq u(y, t) \pm a(y, t)$$

which gives for the shock speed

$$u_L \pm a_L \geq S \geq u_R \pm a_R$$

Navier-Stokes equations

$$\begin{aligned}\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_n} (\rho u_n) &= 0 \\ \frac{\partial}{\partial t} (\rho u_k) + \frac{\partial}{\partial x_n} (\rho u_k u_n + \delta_{kn} p - \tau_{kn}) &= 0, \quad k = 1, \dots, d \\ \frac{\partial}{\partial t} (\rho E) + \frac{\partial}{\partial x_n} (u_n (\rho E + p) + q_n - \tau_{nj} u_j) &= 0\end{aligned}$$

with stress tensor

$$\tau_{kn} = \mu \left(\frac{\partial u_n}{\partial x_k} + \frac{\partial u_k}{\partial x_n} \right) - \frac{2}{3} \mu \frac{\partial u_i}{\partial x_j} \delta_{kn}$$

and heat conduction

$$q_n = -\lambda \frac{\partial T}{\partial x_n}$$

have structure

$$\partial_t \mathbf{q}(\mathbf{x}, t) + \nabla \cdot \mathbf{f}(\mathbf{q}(\mathbf{x}, t)) + \nabla \cdot \mathbf{h}(\mathbf{q}(\mathbf{x}, t), \nabla \mathbf{q}(\mathbf{x}, t)) = 0$$

Type can be either hyperbolic or parabolic

Outline

Conservation laws

- Mathematical background
- Characteristic information
- Weak and entropy solutions
- Characteristic form of the Euler equations
- Navier-Stokes equations

Finite volume methods

- Basics of finite difference methods
- Splitting methods, second derivatives

Upwind schemes

- The linear Riemann problem
- Flux-difference splitting
- Flux-vector splitting

Navier-Stokes equations for multiple species

For multiple species with chemical reaction, the Navier-Stokes equations would be extended to

$$\begin{aligned}\frac{\partial \rho_i}{\partial t} + \frac{\partial}{\partial x_n} (\rho_i u_n + \rho \nu_{in}) &= W_i \dot{\omega}_i, \quad i = 1, \dots, N \\ \frac{\partial}{\partial t} (\rho u_k) + \frac{\partial}{\partial x_n} (\rho u_k u_n + \delta_{kn} p - \tau_{kn}) &= 0, \quad k = 1, \dots, d \\ \frac{\partial}{\partial t} (\rho E) + \frac{\partial}{\partial x_n} (u_n (\rho E + p) + q_n + \rho \sum_j h_j \nu_{jn} - \tau_{nj} u_j) &= 0\end{aligned}$$

with diffusivities

$$\nu_{in} = D_i \frac{\partial Y_i}{\partial x_n}$$

of species i into the mixture (note difference to binary diffusivities). The equation of state

$$p = \sum_i \rho_i R_i T$$

still contains the temperature, which complicates the analysis. The structure is

$$\partial_t \mathbf{q}(\mathbf{x}, t) + \nabla \cdot \mathbf{f}(\mathbf{q}(\mathbf{x}, t)) + \nabla \cdot \mathbf{h}(\mathbf{q}(\mathbf{x}, t), \nabla \mathbf{q}(\mathbf{x}, t)) = \mathbf{s}(\mathbf{q}(\mathbf{x}, t))$$

Derivation

Assume $\partial_t \mathbf{q} + \partial_x \mathbf{f}(\mathbf{q}) + \partial_x \mathbf{h}(\mathbf{q}, \partial_x \mathbf{q}) = \mathbf{s}(\mathbf{q})$

Time discretization $t_n = n \Delta t$, discrete volumes
 $I_j = [x_j - \frac{1}{2} \Delta x, x_j + \frac{1}{2} \Delta x] =: [x_{j-1/2}, x_{j+1/2}]$

Using approximations $\mathbf{Q}_j(t) \approx \frac{1}{|I_j|} \int_j \mathbf{q}(\mathbf{x}, t) dx$, $\mathbf{s}(\mathbf{Q}_j(t)) \approx \frac{1}{|I_j|} \int_j \mathbf{s}(\mathbf{q}(\mathbf{x}, t)) dx$

and numerical fluxes

$\mathbf{F}(\mathbf{Q}_j(t), \mathbf{Q}_{j+1}(t)) \approx \mathbf{f}(\mathbf{q}(x_{j+1/2}, t))$, $\mathbf{H}(\mathbf{Q}_j(t), \mathbf{Q}_{j+1}(t)) \approx \mathbf{h}(\mathbf{q}(x_{j+1/2}, t), \nabla \mathbf{q}(x_{j+1/2}, t))$
yields after integration (Gauss theorem)

$$\begin{aligned}\mathbf{Q}_j(t_{n+1}) &= \mathbf{Q}_j(t_n) - \frac{1}{\Delta x} \int_{t_n}^{t_{n+1}} [\mathbf{F}(\mathbf{Q}_j(t), \mathbf{Q}_{j+1}(t)) - \mathbf{F}(\mathbf{Q}_{j-1}(t), \mathbf{Q}_j(t))] dt - \\ &\quad \frac{1}{\Delta x} \int_{t_n}^{t_{n+1}} [\mathbf{H}(\mathbf{Q}_j(t), \mathbf{Q}_{j+1}(t)) - \mathbf{H}(\mathbf{Q}_{j-1}(t), \mathbf{Q}_j(t))] dt + \int_{t_n}^{t_{n+1}} \mathbf{s}(\mathbf{Q}_j(t)) dt\end{aligned}$$

For instance:

$$\begin{aligned}\mathbf{Q}_j^{n+1} &= \mathbf{Q}_j^n - \frac{\Delta t}{\Delta x} \left[\mathbf{F}(\mathbf{Q}_j^n, \mathbf{Q}_{j+1}^n) - \mathbf{F}(\mathbf{Q}_{j-1}^n, \mathbf{Q}_j^n) \right] - \\ &\quad \frac{\Delta t}{\Delta x} \left[\mathbf{H}(\mathbf{Q}_j^n, \mathbf{Q}_{j+1}^n) - \mathbf{H}(\mathbf{Q}_{j-1}^n, \mathbf{Q}_j^n) \right] + \Delta t \mathbf{s}(\mathbf{Q}_j^n) dt\end{aligned}$$

Some classical definitions

($2s+1$)-point difference scheme of the form

$$\mathbf{Q}_j^{n+1} = \mathcal{H}^{(\Delta t)}(\mathbf{Q}_{j-s}^n, \dots, \mathbf{Q}_{j+s}^n)$$

Definition (Stability)

For each time τ there is a constant C_S and a value $n_0 \in \mathbb{N}$ such that
 $\|\mathcal{H}^{(\Delta t)}(\mathbf{Q}^n)\| \leq C_S$ for all $n\Delta t \leq \tau$, $n < n_0$

Definition (Consistency)

If the local truncation error

$$\mathcal{L}^{(\Delta t)}(\mathbf{x}, t) := \frac{1}{\Delta t} [\mathbf{q}(\mathbf{x}, t + \Delta t) - \mathcal{H}^{(\Delta t)}(\mathbf{q}(\cdot, t))]$$

satisfies $\|\mathcal{L}^{(\Delta t)}(\cdot, t)\| \rightarrow 0$ as $\Delta t \rightarrow 0$

Definition (Convergence)

If the global error $\mathcal{E}^{(\Delta t)}(\mathbf{x}, t) := \mathbf{Q}(\mathbf{x}, t) - \mathbf{q}(\mathbf{x}, t)$ satisfies $\|\mathcal{E}^{(\Delta t)}(\cdot, t)\| \rightarrow 0$ as $\Delta t \rightarrow 0$ for all admissible initial data $\mathbf{q}_0(\mathbf{x})$

Splitting methods

Solve homogeneous PDE and ODE successively!

$$\begin{aligned} \mathcal{H}^{(\Delta t)} : \quad & \partial_t \mathbf{q} + \nabla \cdot \mathbf{f}(\mathbf{q}) = 0, \quad \text{IC: } \mathbf{Q}(t_m) \xrightarrow{\Delta t} \tilde{\mathbf{Q}} \\ \mathcal{S}^{(\Delta t)} : \quad & \partial_t \mathbf{q} = \mathbf{s}(\mathbf{q}), \quad \text{IC: } \tilde{\mathbf{Q}} \xrightarrow{\Delta t} \mathbf{Q}(t_m + \Delta t) \end{aligned}$$

1st-order Godunov splitting: $\mathbf{Q}(t_m + \Delta t) = \mathcal{S}^{(\Delta t)} \mathcal{H}^{(\Delta t)}(\mathbf{Q}(t_m))$,

2nd-order Strang splitting: $\mathbf{Q}(t_m + \Delta t) = \mathcal{S}^{(\frac{1}{2}\Delta t)} \mathcal{H}^{(\Delta t)} \mathcal{S}^{(\frac{1}{2}\Delta t)}(\mathbf{Q}(t_m))$

1st-order dimensional splitting for $\mathcal{H}^{(\cdot)}$:

$$\begin{aligned} \mathcal{X}_1^{(\Delta t)} : \quad & \partial_t \mathbf{q} + \partial_{x_1} \mathbf{f}_1(\mathbf{q}) = 0, \quad \text{IC: } \mathbf{Q}(t_m) \xrightarrow{\Delta t} \tilde{\mathbf{Q}}^{1/2} \\ \mathcal{X}_2^{(\Delta t)} : \quad & \partial_t \mathbf{q} + \partial_{x_2} \mathbf{f}_2(\mathbf{q}) = 0, \quad \text{IC: } \tilde{\mathbf{Q}}^{1/2} \xrightarrow{\Delta t} \tilde{\mathbf{Q}} \end{aligned}$$

[Toro, 1999]

Some classical definitions II

Definition (Order of accuracy)

$\mathcal{H}(\cdot)$ is accurate of order o if for all sufficiently smooth initial data $\mathbf{q}_0(\mathbf{x})$, there is a constant C_L , such that the local truncation error satisfies
 $\|\mathcal{L}^{(\Delta t)}(\cdot, t)\| \leq C_L \Delta t^o$ for all $\Delta t < \Delta t_0$, $t \leq \tau$

Definition (Conservative form)

If $\mathcal{H}(\cdot)$ can be written in the form

$$\mathbf{Q}_j^{n+1} = \mathbf{Q}_j^n - \frac{\Delta t}{\Delta x} (\mathbf{F}(\mathbf{Q}_{j-s+1}^n, \dots, \mathbf{Q}_{j+s}^n) - \mathbf{F}(\mathbf{Q}_{j-s}^n, \dots, \mathbf{Q}_{j+s-1}^n))$$

A conservative scheme satisfies

$$\sum_{j \in \mathbb{Z}} \mathbf{Q}_j^{n+1} = \sum_{j \in \mathbb{Z}} \mathbf{Q}_j^n$$

Definition (Consistency of a conservative method)

If the numerical flux satisfies $\mathbf{F}(\mathbf{q}, \dots, \mathbf{q}) = \mathbf{f}(\mathbf{q})$ for all $\mathbf{q} \in S$

Conservative scheme for diffusion equation

Consider $\partial_t q - c \Delta q = 0$ with $c \in \mathbb{R}^+$, which is readily discretized as

$$Q_{jk}^{n+1} = Q_{jk}^n + c \frac{\Delta t}{\Delta x_1^2} (Q_{j+1,k}^n - 2Q_{jk}^n + Q_{j-1,k}^n) + c \frac{\Delta t}{\Delta x_2^2} (Q_{j,k+1}^n - 2Q_{jk}^n + Q_{j,k-1}^n)$$

or conservatively

$$Q_{jk}^{n+1} = Q_{jk}^n + c \frac{\Delta t}{\Delta x_1} \left(H_{j+\frac{1}{2},k}^1 - H_{j-\frac{1}{2},k}^1 \right) + c \frac{\Delta t}{\Delta x_2} \left(H_{j,k+\frac{1}{2}}^2 - H_{j,k-\frac{1}{2}}^2 \right)$$

Von Neumann stability analysis: Insert single eigenmode $\hat{Q}(t)e^{ik_1 x_1} e^{ik_2 x_2}$ into discretization

$$\hat{Q}^{n+1} = \hat{Q}^n + C_1 \left(\hat{Q}^n e^{ik_1 \Delta x_1} - 2\hat{Q}^n + \hat{Q}^n e^{-ik_1 \Delta x_1} \right) + C_2 \left(\hat{Q}^n e^{ik_2 \Delta x_2} - 2\hat{Q}^n + \hat{Q}^n e^{-ik_2 \Delta x_2} \right)$$

with $C_\nu = c \frac{\Delta t}{\Delta x_\nu^2}$, $\nu = 1, 2$, which gives after inserting $e^{ik_\nu x_\nu} = \cos(k_\nu x_\nu) + i \sin(k_\nu x_\nu)$

$$\hat{Q}^{n+1} = \hat{Q}^n (1 + 2C_1(\cos(k_1 \Delta x_1) - 1) + 2C_2(\cos(k_2 \Delta x_2) - 1))$$

Stability requires

$$|1 + 2C_1(\cos(k_1 \Delta x_1) - 1) + 2C_2(\cos(k_2 \Delta x_2) - 1)| \leq 1$$

i.e.

$$|1 - 4C_1 - 4C_2| \leq 1$$

from which we derive the stability condition

$$0 \leq c \left(\frac{\Delta t}{\Delta x_1^2} + \frac{\Delta t}{\Delta x_2^2} \right) \leq \frac{1}{2}$$

Outline

Conservation laws

- Mathematical background
- Characteristic information
- Weak and entropy solutions
- Characteristic form of the Euler equations
- Navier-Stokes equations

Finite volume methods

- Basics of finite difference methods
- Splitting methods, second derivatives

Upwind schemes

- The linear Riemann problem
- Flux-difference splitting
- Flux-vector splitting

The Riemann problem in the linear case

Consider the linear hyperbolic equation (i.e. $\mathbf{A} = \text{const.}$)

$$\frac{\partial}{\partial t} \mathbf{q}(x, t) + \mathbf{A} \frac{\partial}{\partial x} \mathbf{q}(x, t) = 0$$

Assume (for simplicity) that \mathbf{A} has M distinct real eigenvalues $\lambda_1 < \dots < \lambda_M$ with M linear independent right eigenvectors \mathbf{r}_m .

We can readily apply the characteristic transformation $\mathbf{R}^{-1} \mathbf{q} = \mathbf{v}$ to obtain

$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{\Lambda} \frac{\partial \mathbf{v}}{\partial x} = 0$$

or

$$\frac{\partial v_m}{\partial t} + \lambda_m \frac{\partial v_m}{\partial x} = 0 \quad \text{for all } m = 1, \dots, M$$

Each characteristic variable v_m changes only across the characteristic line associated to λ_m .

Since the entire problem is linear, we can simply sum up all these jumps Δv_m successively to connect the RP states $\mathbf{v}_L, \mathbf{v}_R$ in characteristic variables.

The Riemann problem in the linear case - II

Example of a linear 3 PDE system:

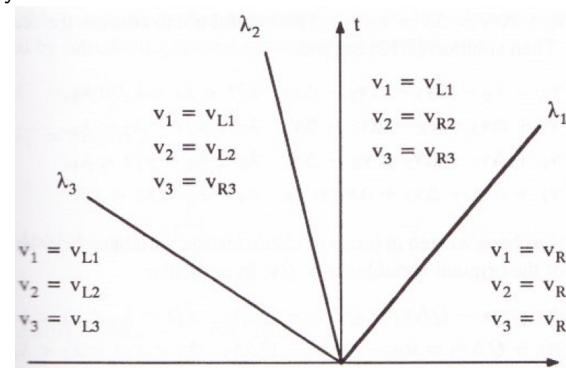
Introducing the jumps

$$\Delta \mathbf{v}_1 = [v_{R1} - v_{L1}, 0, 0]^T$$

$$\Delta \mathbf{v}_2 = [0, v_{R2} - v_{L2}, 0]^T$$

$$\Delta \mathbf{v}_3 = [0, 0, v_{R3} - v_{L3}]^T$$

the solution reads



$$\mathbf{v}\left(\frac{x}{t}\right) = \begin{cases} \mathbf{v}_L = \mathbf{v}_R - \Delta \mathbf{v}_3 - \Delta \mathbf{v}_2 - \Delta \mathbf{v}_1 & x/t < \lambda_3 \\ \mathbf{v}_L + \Delta \mathbf{v}_3 = \mathbf{v}_R - \Delta \mathbf{v}_2 - \Delta \mathbf{v}_1 & \lambda_3 < x/t < \lambda_2 \\ \mathbf{v}_L + \Delta \mathbf{v}_3 + \Delta \mathbf{v}_2 = \mathbf{v}_R - \Delta \mathbf{v}_1 & \lambda_2 < x/t < \lambda_1 \\ \mathbf{v}_L + \Delta \mathbf{v}_3 + \Delta \mathbf{v}_2 + \Delta \mathbf{v}_1 = \mathbf{v}_R & \lambda_1 < x/t \end{cases}$$

The Riemann problem in the linear case - III

or using the transformation $\mathbf{q} = \mathbf{R}\mathbf{v}$ and $\mathbf{R}\Delta\mathbf{v}_m = \mathbf{r}_m\Delta v_m$

$$\mathbf{q}\left(\frac{x}{t}\right) = \begin{cases} \mathbf{q}_L = \mathbf{q}_R - \mathbf{r}_3\Delta v_3 - \mathbf{r}_2\Delta v_2 - \mathbf{r}_1\Delta v_1 & x/t < \lambda_3 \\ \mathbf{q}_L + \mathbf{r}_3\Delta v_3 = \mathbf{q}_R - \mathbf{r}_2\Delta v_2 - \mathbf{r}_1\Delta v_1 & \lambda_3 < x/t < \lambda_2 \\ \mathbf{q}_L + \mathbf{r}_3\Delta v_3 + \mathbf{r}_2\Delta v_2 = \mathbf{q}_R - \mathbf{r}_1\Delta v_1 & \lambda_2 < x/t < \lambda_1 \\ \mathbf{q}_L + \mathbf{r}_3\Delta v_3 + \mathbf{r}_2\Delta v_2 + \mathbf{r}_1\Delta v_1 = \mathbf{q}_R & \lambda_1 < x/t \end{cases}$$

Multiplying with \mathbf{A} and using $\mathbf{A}\mathbf{r}_m = \lambda_m \mathbf{r}_m$ gives

$$\mathbf{A}\mathbf{q}\left(\frac{x}{t}\right) = \begin{cases} \mathbf{A}\mathbf{q}_L = \mathbf{A}\mathbf{q}_R - \mathbf{r}_3\lambda_3\Delta v_3 - \mathbf{r}_2\lambda_2\Delta v_2 - \mathbf{r}_1\lambda_1\Delta v_1 & x/t < \lambda_3 \\ \mathbf{A}\mathbf{q}_L + \mathbf{r}_3\lambda_3\Delta v_3 = \mathbf{A}\mathbf{q}_R - \mathbf{r}_2\lambda_2\Delta v_2 - \mathbf{r}_1\lambda_1\Delta v_1 & \lambda_3 < x/t < \lambda_2 \\ \mathbf{A}\mathbf{q}_L + \mathbf{r}_3\lambda_3\Delta v_3 + \mathbf{r}_2\lambda_2\Delta v_2 = \mathbf{A}\mathbf{q}_R - \mathbf{r}_1\lambda_1\Delta v_1 & \lambda_2 < x/t < \lambda_1 \\ \mathbf{A}\mathbf{q}_L + \mathbf{r}_3\lambda_3\Delta v_3 + \mathbf{r}_2\lambda_2\Delta v_2 + \mathbf{r}_1\lambda_1\Delta v_1 = \mathbf{A}\mathbf{q}_R & \lambda_1 < x/t \end{cases}$$

This allows direct evaluation of the flux at $x=0$.

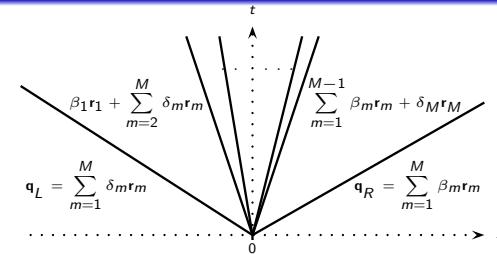
$$\mathbf{F}(\mathbf{q}_L, \mathbf{q}_R) := \mathbf{f}(\mathbf{q}(0, t)) = \mathbf{A}\mathbf{q}(0, t)$$

Linear upwind schemes

Consider Riemann problem

$$\frac{\partial}{\partial t} \mathbf{q}(x, t) + \mathbf{A} \frac{\partial}{\partial x} \mathbf{q}(x, t) = \mathbf{0}, \quad x \in \mathbb{R}, \quad t > 0$$

Has exact solution



Use Riemann problem to evaluate numerical flux $\mathbf{F}(\mathbf{q}_L, \mathbf{q}_R) := \mathbf{f}(\mathbf{q}(0, t)) = \mathbf{A}\mathbf{q}(0, t)$ as

$$\mathbf{F}(\mathbf{q}_L, \mathbf{q}_R) = \mathbf{A}\mathbf{q}_L + \sum_{\lambda_m < 0} a_m \lambda_m \mathbf{r}_m = \mathbf{A}\mathbf{q}_R - \sum_{\lambda_m \geq 0} a_m \lambda_m \mathbf{r}_m = \sum_{\lambda_m \geq 0} \delta_m \lambda_m \mathbf{r}_m + \sum_{\lambda_m < 0} \beta_m \lambda_m \mathbf{r}_m$$

Use $\lambda_m^+ = \max(\lambda_m, 0)$, $\lambda_m^- = \min(\lambda_m, 0)$

to define $\Lambda^+ := \text{diag}(\lambda_1^+, \dots, \lambda_M^+)$, $\Lambda^- := \text{diag}(\lambda_1^-, \dots, \lambda_M^-)$

and $\mathbf{A}^+ := \mathbf{R} \Lambda^+ \mathbf{R}^{-1}$, $\mathbf{A}^- := \mathbf{R} \Lambda^- \mathbf{R}^{-1}$ which gives

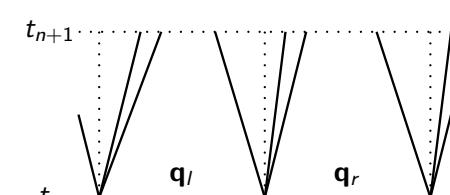
$$\mathbf{F}(\mathbf{q}_L, \mathbf{q}_R) = \mathbf{A}\mathbf{q}_L + \mathbf{A}^- \Delta \mathbf{q} = \mathbf{A}\mathbf{q}_R - \mathbf{A}^+ \Delta \mathbf{q} = \mathbf{A}^+ \mathbf{q}_L + \mathbf{A}^- \mathbf{q}_R$$

with $\Delta \mathbf{q} = \mathbf{q}_R - \mathbf{q}_L$

Roe's approximate Riemann solver

Choosing $\hat{\mathbf{A}}(\mathbf{q}_L, \mathbf{q}_R)$ [Roe, 1981]:

- (i) $\hat{\mathbf{A}}(\mathbf{q}_L, \mathbf{q}_R)$ has real eigenvalues
- (ii) $\hat{\mathbf{A}}(\mathbf{q}_L, \mathbf{q}_R) \rightarrow \frac{\partial \mathbf{f}(\mathbf{q})}{\partial \mathbf{q}}$ as $\mathbf{q}_L, \mathbf{q}_R \rightarrow \mathbf{q}$
- (iii) $\hat{\mathbf{A}}(\mathbf{q}_L, \mathbf{q}_R) \Delta \mathbf{q} = \mathbf{f}(\mathbf{q}_R) - \mathbf{f}(\mathbf{q}_L)$



Wave decomposition: $\Delta \mathbf{q} = \mathbf{q}_r - \mathbf{q}_l = \sum_m a_m \hat{\mathbf{r}}_m$

$$\begin{aligned} \mathbf{F}(\mathbf{q}_L, \mathbf{q}_R) &= \mathbf{f}(\mathbf{q}_L) + \sum_{\hat{\lambda}_m < 0} \hat{\lambda}_m a_m \hat{\mathbf{r}}_m = \mathbf{f}(\mathbf{q}_R) - \sum_{\hat{\lambda}_m \geq 0} \hat{\lambda}_m a_m \hat{\mathbf{r}}_m \\ &= \frac{1}{2} \left(\mathbf{f}(\mathbf{q}_L) + \mathbf{f}(\mathbf{q}_R) - \sum_m |\hat{\lambda}_m| a_m \hat{\mathbf{r}}_m \right) \end{aligned}$$

Flux difference splitting

Godunov-type scheme with $\Delta \mathbf{Q}_{j+1/2}^n = \mathbf{Q}_{j+1}^n - \mathbf{Q}_j^n$

$$\mathbf{Q}_j^{n+1} = \mathbf{Q}_j^n - \frac{\Delta t}{\Delta x} \left(\mathbf{A}^- \Delta \mathbf{Q}_{j+1/2}^n + \mathbf{A}^+ \Delta \mathbf{Q}_{j-1/2}^n \right)$$

Use linearization $\bar{\mathbf{f}}(\bar{\mathbf{q}}) = \hat{\mathbf{A}}(\mathbf{q}_L, \mathbf{q}_R) \bar{\mathbf{q}}$ and construct scheme for nonlinear problem as

$$\mathbf{Q}_j^{n+1} = \mathbf{Q}_j^n - \frac{\Delta t}{\Delta x} \left(\hat{\mathbf{A}}^-(\mathbf{Q}_j^n, \mathbf{Q}_{j+1}^n) \Delta \mathbf{Q}_{j+\frac{1}{2}}^n + \hat{\mathbf{A}}^+(\mathbf{Q}_{j-1}^n, \mathbf{Q}_j^n) \Delta \mathbf{Q}_{j-\frac{1}{2}}^n \right)$$

stability condition

$$\max_{j \in \mathbb{Z}} |\hat{\lambda}_{m,j+\frac{1}{2}}| \frac{\Delta t}{\Delta x} \leq 1, \quad \text{for all } m = 1, \dots, M$$

[LeVeque, 1992]

The Roe solver for Euler equations

For Euler equations, the following non-apparent average defines the Roe method:
The average for u and H read

$$\hat{u} := \frac{\sqrt{\rho_L} u_L + \sqrt{\rho_R} u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \quad \hat{H} := \frac{\sqrt{\rho_L} H_L + \sqrt{\rho_R} H_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}$$

The average of the density is

$$\hat{\rho} = \frac{\sqrt{\rho_L} \rho_R + \sqrt{\rho_R} \rho_L}{\sqrt{\rho_L} + \sqrt{\rho_R}} = \sqrt{\rho_L \rho_R}$$

and the averaged speed of sound is

$$\hat{a} := \left((\gamma - 1)(\hat{H} - \frac{1}{2} \hat{u}^2) \right)^{1/2}$$

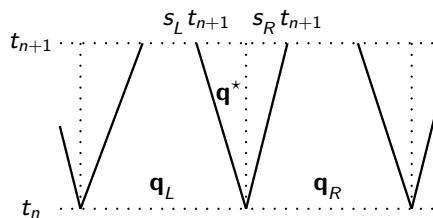
The eigenvectors read

$$\hat{\mathbf{r}}_1 = \begin{bmatrix} 1 \\ \hat{u} - \hat{a} \\ \hat{H} - \hat{u}\hat{a} \end{bmatrix}, \quad \hat{\mathbf{r}}_2 = \begin{bmatrix} 1 \\ \hat{u} \\ \hat{u}^2/2 \end{bmatrix}, \quad \hat{\mathbf{r}}_3 = \begin{bmatrix} 1 \\ \hat{u} + \hat{a} \\ \hat{H} + \hat{u}\hat{a} \end{bmatrix}$$

and the characteristic wave strengths are

$$\Delta v_1 = a_1 = \frac{\Delta p - \hat{\rho}\hat{a}\Delta u}{2\hat{a}^2}, \quad \Delta v_2 = a_2 = \Delta \rho - \frac{\Delta p}{\hat{a}^2}, \quad \Delta v_3 = a_3 = \frac{\Delta p + \hat{\rho}\hat{a}\Delta u}{2\hat{a}^2}.$$

Harten-Lax-Van Leer (HLL) approximate Riemann solver



$$\bar{\mathbf{q}}(x, t) = \begin{cases} \mathbf{q}_L, & x < s_L t \\ \mathbf{q}^*, & s_L t \leq x \leq s_R t \\ \mathbf{q}_R, & x > s_R t \end{cases}$$

$$\mathbf{F}_{HLL}(\mathbf{q}_L, \mathbf{q}_R) = \begin{cases} \mathbf{f}(\mathbf{q}_L), & 0 < s_L, \\ \frac{s_R \mathbf{f}(\mathbf{q}_L) - s_L \mathbf{f}(\mathbf{q}_R) + s_L s_R (\mathbf{q}_R - \mathbf{q}_L)}{s_R - s_L}, & s_L \leq 0 \leq s_R, \\ \mathbf{f}(\mathbf{q}_R), & 0 > s_R, \end{cases}$$

Euler equations:

$$s_L = \min(u_{1,L} - c_L, u_{1,R} - c_R), \quad s_R = \max(u_{1,L} + c_L, u_{1,R} + c_R)$$

[Toro, 1999], HLLC: [Toro et al., 1994]

Steger-Warming

Required $\mathbf{f}(\mathbf{q}) = \mathbf{A}(\mathbf{q}) \mathbf{q}$

$$\lambda_m^+ = \frac{1}{2} (\lambda_m + |\lambda_m|) \quad \lambda_m^- = \frac{1}{2} (\lambda_m - |\lambda_m|)$$

$$\mathbf{A}^+(\mathbf{q}) := \mathbf{R}(\mathbf{q}) \Lambda^+(\mathbf{q}) \mathbf{R}^{-1}(\mathbf{q}), \quad \mathbf{A}^-(\mathbf{q}) := \mathbf{R}(\mathbf{q}) \Lambda^-(\mathbf{q}) \mathbf{R}^{-1}(\mathbf{q})$$

Gives

$$\mathbf{f}(\mathbf{q}) = \mathbf{A}^+(\mathbf{q}) \mathbf{q} + \mathbf{A}^-(\mathbf{q}) \mathbf{q}$$

and the numerical flux

$$\mathbf{F}(\mathbf{q}_L, \mathbf{q}_R) = \mathbf{A}^+(\mathbf{q}_L) \mathbf{q}_L + \mathbf{A}^-(\mathbf{q}_R) \mathbf{q}_R$$

Jacobians of the split fluxes are identical to $\mathbf{A}^\pm(\mathbf{q})$ only in linear case

$$\frac{\partial \mathbf{f}^\pm(\mathbf{q})}{\partial \mathbf{q}} = \frac{\partial (\mathbf{A}^\pm(\mathbf{q}) \mathbf{q})}{\partial \mathbf{q}} = \mathbf{A}^\pm(\mathbf{q}) + \frac{\partial \mathbf{A}^\pm(\mathbf{q})}{\partial \mathbf{q}} \mathbf{q}$$

Further methods: Van Leer FVS [Toro, 1999], AUSM [Wada and Liou, 1997]

Flux vector splitting

Splitting

$$\mathbf{f}(\mathbf{q}) = \mathbf{f}^+(\mathbf{q}) + \mathbf{f}^-(\mathbf{q})$$

derived under restriction $\hat{\lambda}_m^+ \geq 0$ and $\hat{\lambda}_m^- \leq 0$ for all $m = 1, \dots, M$ for

$$\hat{\mathbf{A}}^+(\mathbf{q}) = \frac{\partial \mathbf{f}^+(\mathbf{q})}{\partial \mathbf{q}}, \quad \hat{\mathbf{A}}^-(\mathbf{q}) = \frac{\partial \mathbf{f}^-(\mathbf{q})}{\partial \mathbf{q}}$$

plus reproduction of regular upwinding

$$\begin{aligned} \mathbf{f}^+(\mathbf{q}) &= \mathbf{f}(\mathbf{q}), & \mathbf{f}^-(\mathbf{q}) &= \mathbf{0} & \text{if } \lambda_m \geq 0 & \text{for all } m = 1, \dots, M \\ \mathbf{f}^+(\mathbf{q}) &= \mathbf{0}, & \mathbf{f}^-(\mathbf{q}) &= \mathbf{f}(\mathbf{q}) & \text{if } \lambda_m \leq 0 & \text{for all } m = 1, \dots, M \end{aligned}$$

Then use

$$\mathbf{F}(\mathbf{q}_L, \mathbf{q}_R) = \mathbf{f}^+(\mathbf{q}_L) + \mathbf{f}^-(\mathbf{q}_R)$$

Steger-Warming FVS for Euler equations

For Euler equations, $\mathbf{f}(\mathbf{q}) = \mathbf{A}(\mathbf{q}) \mathbf{q}$ holds true. We also know all matrices $\mathbf{R}^{-1} \mathbf{A} \mathbf{R} = \mathbf{\Lambda}$.

Approach 1: Introduce $\lambda_m^+ = \frac{1}{2} (\lambda_m + |\lambda_m|)$, $\lambda_m^- = \frac{1}{2} (\lambda_m - |\lambda_m|)$ and compute \mathbf{A}^+ and \mathbf{A}^- directly.

Approach 2: Analyze sign of eigenvalues:

$u < -a$: All eigenvalues are negative: $u - a < u < u + a < 0$

$$\mathbf{f}^-(\mathbf{q}) = [\rho u, \rho u^2 + p, \rho u H]^T, \quad \mathbf{f}^+(\mathbf{q}) = \mathbf{0}$$

$u > a$: All eigenvalues are positive: $0 < u - a < u < u + a$

$$\mathbf{f}^-(\mathbf{q}) = \mathbf{0}, \quad \mathbf{f}^+(\mathbf{q}) = [\rho u, \rho u^2 + p, \rho u H]^T$$

$-a \leq u \leq a$: We find $u - a \leq 0$ and $u + a \geq 0$ are always satisfied. For $u < 0$, we need to evaluate

$$\mathbf{f}^-(\mathbf{q}) = \mathbf{R} \begin{bmatrix} u - a & 0 & 0 \\ 0 & u & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{R}^{-1} \mathbf{q}, \quad \mathbf{f}^+(\mathbf{q}) = \mathbf{R} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & u + a \end{bmatrix} \mathbf{R}^{-1} \mathbf{q}$$

For $u \geq 0$, we need to evaluate

$$\mathbf{f}^-(\mathbf{q}) = \mathbf{R} \begin{bmatrix} u - a & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{R}^{-1} \mathbf{q}, \quad \mathbf{f}^+(\mathbf{q}) = \mathbf{R} \begin{bmatrix} 0 & 0 & 0 \\ 0 & u & 0 \\ 0 & 0 & u + a \end{bmatrix} \mathbf{R}^{-1} \mathbf{q}$$

References I

- [Godlewski and Raviart, 1996] Godlewski, E. and Raviart, P.-A. (1996). *Numerical approximation of hyperbolic systems of conservation laws*. Springer Verlag, New York.
- [Kröner, 1997] Kröner, D. (1997). *Numerical schemes for conservation laws*. John Wiley & Sons and B. G. Teubner, New York, Leipzig.
- [LeVeque, 1992] LeVeque, R. J. (1992). *Numerical methods for conservation laws*. Birkhäuser, Basel.
- [Majda, 1984] Majda, A. (1984). *Compressible fluid flow and systems of conservation laws in several space variables*. Applied Mathematical Sciences Vol. 53. Springer-Verlag, New York.
- [Roe, 1981] Roe, P. L. (1981). Approximate Riemann solvers, parameter vectors and difference schemes. *J. Comput. Phys.*, 43:357–372.
- [Toro, 1999] Toro, E. F. (1999). *Riemann solvers and numerical methods for fluid dynamics*. Springer-Verlag, Berlin, Heidelberg, 2nd edition.
- [Toro et al., 1994] Toro, E. F., Spruce, M., and Speares, W. (1994). Restoration of the contact surface in the HLL-Riemann solver. *Shock Waves*, 4:25–34.

References II

- [Wada and Liou, 1997] Wada, Y. and Liou, M.-S. (1997). An accurate and robust flux splitting scheme for shock and contact discontinuities. *SIAM J. Sci. Comp.*, 18(3):633–657.

Outline

Lecture 2

Structured adaptive mesh refinement

Course *Block-structured Adaptive Finite Volume Methods in C++*

Meshes and adaptation

Adaptivity on unstructured and structured meshes
Available SAMR software

The serial Berger-Colella SAMR method

Data structures and numerical update
Conservative flux correction
Level transfer operators
The basic recursive algorithm
Block generation and flagging of cells

Parallel SAMR method

Domain decomposition
A parallel SAMR algorithm

Ralf Deiterding
University of Southampton
Engineering and the Environment
Highfield Campus, Southampton SO17 1BJ, UK

E-mail: r.deiterding@soton.ac.uk

Outline

Meshes and adaptation

- Adaptivity on unstructured and structured meshes
- Available SAMR software

The serial Berger-Colella SAMR method

- Data structures and numerical update
- Conservative flux correction
- Level transfer operators
- The basic recursive algorithm
- Block generation and flagging of cells

Parallel SAMR method

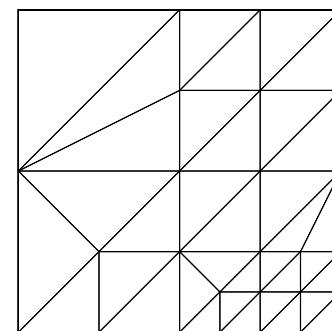
- Domain decomposition
- A parallel SAMR algorithm

Structured adaptive mesh refinement

Structured adaptive mesh refinement

Adaptivity on unstructured and structured meshes

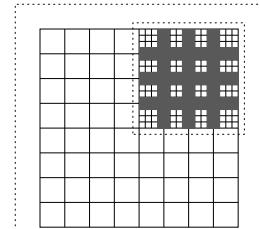
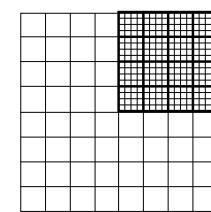
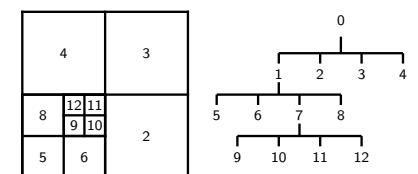
- Coarse cells replaced by finer ones
- Global time-step
- Cell-based data structures
- Neighborhoods have to stored
- Geometric flexible
- No hanging nodes
- Easy to implement
- Higher order difficult to achieve
- Cell aspect ratio must be considered
- Fragmented data
- Cache-reuse / vectorizaton nearly impossible
- Complex load-balancing
- Complex synchronization



- Base grid
- Solver
- Error indicators
- Grid manipulation
- Interpolation (restriction and prolongation)
- Load-balancing

Structured mesh refinement techniques

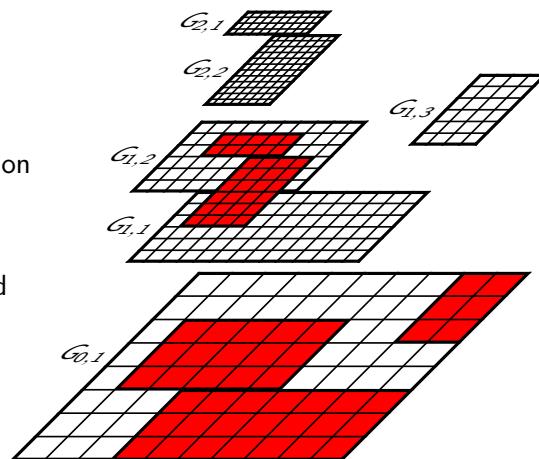
- Block-based data of equal size
- Block stored in a quad-tree
- Time-step refinement
- Global index coordinate system
- Neighborhoods need not be stored
- Numerical scheme only for single regular block necessary
- Easy to implement
- Simple load-balancing
- Parent/Child relations according to tree
- +/- Cache-reuse / vectorization only in data block



Wasted boundary space in a quad-tree

Block-structured adaptive mesh refinement (SAMR)

- ▶ Refined block overlay coarser ones
 - ▶ Time-step refinement
 - ▶ Block (aka patch) based data structures
 - ▶ Global index coordinate system
 - + Numerical scheme only for single patch necessary
 - + Efficient cache-reuse / vectorization possible
 - + Simple load-balancing
 - + Minimal synchronization overhead
 - Cells without mark are refined
 - Hanging nodes unavoidable
 - Cluster-algorithm necessary
 - Difficult to implement



Systems that support general SAMR

- ▶ SAMRAI - Structured Adaptive Mesh Refinement Application Infrastructure
 - ▶ Very mature SAMR system [Hornung et al., 2006]
 - ▶ Explicit algorithms directly supported, implicit methods through interface to Hypre package
 - ▶ Mapped geometry and some embedded boundary support
 - ▶ <https://computation-rnd.llnl.gov/SAMRAI/software.php>
 - ▶ BoxLib, AmrLib, MGLib, HGProj
 - ▶ Berkley-Lab-AMR collection of C++ classes by J. Bell et al., 50,000 LOC [Rendleman et al., 2000]
 - ▶ Both multigrid and explicit algorithms supported
 - ▶ <https://ccse.lbl.gov/Downloads/index.html>
 - ▶ Chombo
 - ▶ Redesign and extension of BoxLib by P. Colella et al.
 - ▶ Both multigrid and explicit algorithms demonstrated
 - ▶ Some embedded boundary support
 - ▶ <https://commons.lbl.gov/display/chombo>

Simplified structured designs

Distributed memory parallelization fully supported if not otherwise states.

- ▶ PARAMESH (Parallel Adaptive Mesh Refinement)
 - ▶ Library based on uniform refinement blocks [MacNeice et al., 2000]
 - ▶ Both multigrid and explicit algorithms considered
 - ▶ <http://sourceforge.net/projects/paramesh>
 - ▶ Flash code (AMR code for astrophysical thermonuclear flashes)
 - ▶ Built on PARAMESH
 - ▶ Solves the magneto-hydrodynamic equations with self-gravitation
 - ▶ <http://www.flash.uchicago.edu/site/flashcode>
 - ▶ Uintah (AMR code for simulation of accidental fires and explosions)
 - ▶ Only explicit algorithms considered
 - ▶ FSI coupling Material Point Method and ICE Method (Implicit, Continuous fluid, Eulerian)
 - ▶ <http://www.uintah.utah.edu>
 - ▶ DAGH/Grace [Parashar and Browne, 1997]
 - ▶ Just C++ data structures but no methods
 - ▶ All grids are aligned to bases mesh coarsened by factor 2
 - ▶ <http://userweb.cs.utexas.edu/users/dagh>

Outline

Meshes and adaptation

Adaptivity on unstructured and structured meshes
Available SAMR software

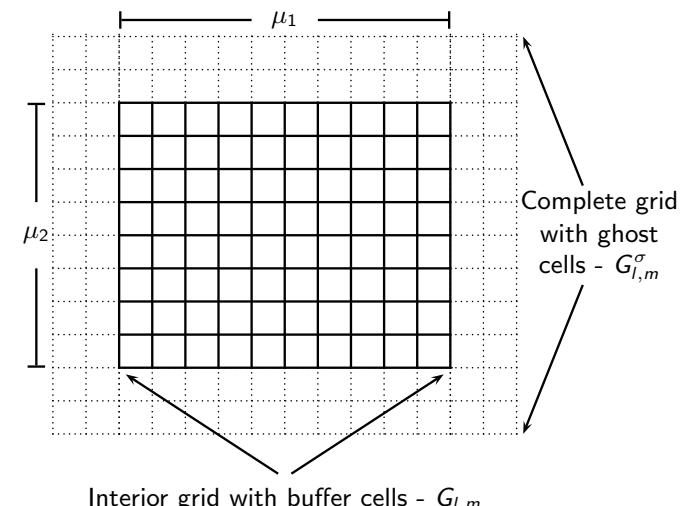
The serial Berger-Colella SAMR method

Data structures and numerical update
Conservative flux correction
Level transfer operators
The basic recursive algorithm
Block generation and flagging of cells

Parallel SAMR method

Domain decomposition
A parallel SAMR algorithm

The m th refinement grid on level l



Structured adaptive mesh refinement

Mesches and adaptation
oooooooo
Data structures and numerical update

Serial SAMR method

oooooooooooooooooooo

Parallel SAMR method

oooooooooooo

References

ooo

Structured adaptive mesh refinement

Mesches and adaptation
oooooooo
Data structures and numerical update

Serial SAMR method

oooooooooooooooooooo

Parallel SAMR method

oooooooooooo

References

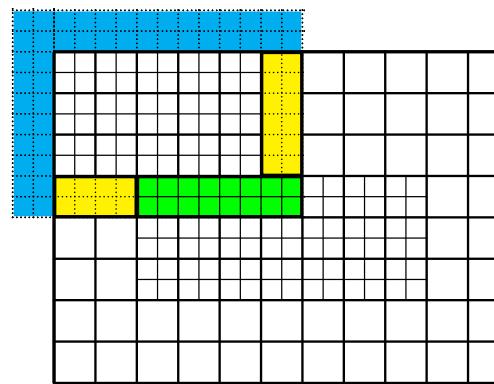
ooo

Refinement data

- ▶ Resolution: $\Delta t_l := \frac{\Delta t_{l-1}}{r_l}$ and $\Delta x_{n,l} := \frac{\Delta x_{n,l-1}}{r_l}$
- ▶ Refinement factor: $r_l \in \mathbb{N}, r_l \geq 2$ for $l > 0$ and $r_0 = 1$
- ▶ Integer coordinate system for internal organization [Bell et al., 1994]:

$$\Delta x_{n,l} \cong \prod_{\kappa=l+1}^{l_{\max}} r_\kappa$$
- ▶ Computational Domain: $G_0 = \bigcup_{m=1}^{M_0} G_{0,m}$
- ▶ Domain of level l : $G_l := \bigcup_{m=1}^{M_l} G_{l,m}$ with $G_{l,m} \cap G_{l,n} = \emptyset$ for $m \neq n$
- ▶ Refinements are properly nested: $G_l^1 \subset G_{l-1}$
- ▶ Assume a FD scheme with stencil radius s . Necessary data:
 - ▶ Vector of state: $\mathbf{Q}^l := \bigcup_m \mathbf{Q}(G_{l,m}^s)$
 - ▶ Numerical fluxes: $\mathbf{F}^{n,l} := \bigcup_m \mathbf{F}^n(\tilde{G}_{l,m})$
 - ▶ Flux corrections: $\delta\mathbf{F}^{n,l} := \bigcup_m \delta\mathbf{F}^n(\partial G_{l,m})$

Setting of ghost cells



- ▶ Synchronization with G_l - $\tilde{S}_{l,m}^s = \tilde{G}_{l,m}^s \cap G_l$
- ▶ Physical boundary conditions - $\tilde{P}_{l,m}^s = \tilde{G}_{l,m}^s \setminus G_0$
- ▶ Interpolation from G_{l-1} - $\tilde{I}_{l,m}^s = \tilde{G}_{l,m}^s \setminus (\tilde{S}_{l,m}^s \cup \tilde{P}_{l,m}^s)$

Numerical update

Time-explicit conservative finite volume scheme

$$\mathcal{H}^{(\Delta t)} : \mathbf{Q}_{jk}(t + \Delta t) = \mathbf{Q}_{jk}(t) - \frac{\Delta t}{\Delta x_1} (\mathbf{F}_{j+\frac{1}{2},k}^1 - \mathbf{F}_{j-\frac{1}{2},k}^1) - \frac{\Delta t}{\Delta x_2} (\mathbf{F}_{j,k+\frac{1}{2}}^2 - \mathbf{F}_{j,k-\frac{1}{2}}^2)$$

UpdateLevel(l)

```
For all  $m = 1$  To  $M_l$  Do
   $\mathbf{Q}(G_{l,m}^s, t) \xrightarrow{\mathcal{H}^{(\Delta t_l)}} \mathbf{Q}(G_{l,m}, t + \Delta t_l), \mathbf{F}^n(\bar{G}_{l,m}, t)$ 
  If level  $l > 0$ 
    Add  $\mathbf{F}^n(\partial G_{l,m}, t)$  to  $\delta\mathbf{F}^{n,l}$ 
  If level  $l+1$  exists
    Init  $\delta\mathbf{F}^{n,l+1}$  with  $\mathbf{F}^n(\bar{G}_{l,m} \cap \partial G_{l+1}, t)$ 
```

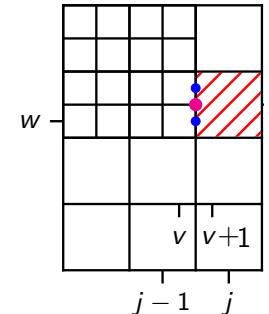
Conservative flux correction

Example: Cell j, k

$$\check{\mathbf{Q}}_{jk}^l(t + \Delta t_l) = \mathbf{Q}_{jk}^l(t) - \frac{\Delta t_l}{\Delta x_{1,l}} \left(\mathbf{F}_{j+\frac{1}{2},k}^{1,l} - \frac{1}{r_{l+1}^2} \sum_{\kappa=0}^{r_{l+1}-1} \sum_{\iota=0}^{r_{l+1}-1} \mathbf{F}_{v+\frac{1}{2},w+\iota}^{1,l+1}(t + \kappa \Delta t_{l+1}) \right) - \frac{\Delta t_l}{\Delta x_{2,l}} \left(\mathbf{F}_{j,k+\frac{1}{2}}^{2,l} - \mathbf{F}_{j,k-\frac{1}{2}}^{2,l} \right)$$

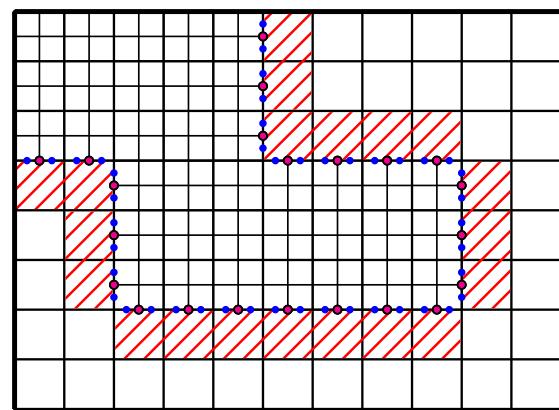
Correction pass:

1. $\delta\mathbf{F}_{j-\frac{1}{2},k}^{1,l+1} := -\mathbf{F}_{j-\frac{1}{2},k}^{1,l}$
2. $\delta\mathbf{F}_{j-\frac{1}{2},k}^{1,l+1} := \delta\mathbf{F}_{j-\frac{1}{2},k}^{1,l+1} + \frac{1}{r_{l+1}^2} \sum_{\iota=0}^{r_{l+1}-1} \mathbf{F}_{v+\frac{1}{2},w+\iota}^{1,l+1}(t + \kappa \Delta t_{l+1})$
3. $\check{\mathbf{Q}}_{jk}^l(t + \Delta t_l) := \mathbf{Q}_{jk}^l(t + \Delta t_l) + \frac{\Delta t_l}{\Delta x_{1,l}} \delta\mathbf{F}_{j-\frac{1}{2},k}^{1,l+1}$



Conservative flux correction II

- ▶ Level l cells needing correction $(G_{l+1}^{r_{l+1}} \setminus G_{l+1}) \cap G_l$
- ▶ Corrections $\delta\mathbf{F}^{n,l+1}$ stored on level $l+1$ along ∂G_{l+1} (lower-dimensional data coarsened by r_{l+1})
- ▶ Init $\delta\mathbf{F}^{n,l+1}$ with level l fluxes $\mathbf{F}^{n,l}(\bar{G}_l \cap \partial G_{l+1})$
- ▶ Add level $l+1$ fluxes $\mathbf{F}^{n,l+1}(\partial G_{l+1})$ to $\delta\mathbf{F}^{n,l+1}$



Cells to correct $\bullet \mathbf{F}^{n,l}$ $\bullet \mathbf{F}^{n,l+1}$ $\circ \delta\mathbf{F}^{n,l+1}$

Level transfer operators

Conservative averaging (restriction):

Replace cells on level l covered by level $l+1$, i.e. $G_l \cap G_{l+1}$, by

$$\hat{\mathbf{Q}}_{jk}^l := \frac{1}{(r_{l+1})^2} \sum_{\kappa=0}^{r_{l+1}-1} \sum_{\iota=0}^{r_{l+1}-1} \mathbf{Q}_{v+\kappa, w+\iota}^{l+1}$$

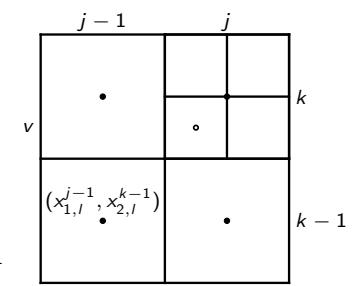
Bilinear interpolation (prolongation):

$$\begin{aligned} \mathbf{Q}_{vw}^{l+1} := & (1-f_1)(1-f_2) \mathbf{Q}_{j-1,k-1}^l + f_1(1-f_2) \mathbf{Q}_{j,k-1}^l + \\ & (1-f_1)f_2 \mathbf{Q}_{j-1,k}^l + f_1f_2 \mathbf{Q}_{jk}^l \end{aligned}$$

with factors $f_1 := \frac{x_{1,l+1}^v - x_{1,l}^{j-1}}{\Delta x_{1,l}}$, $f_2 := \frac{x_{2,l+1}^w - x_{2,l}^{k-1}}{\Delta x_{2,l}}$ derived from the spatial coordinates of the cell centers $(x_{1,l}^{j-1}, x_{2,l}^{k-1})$ and $(x_{1,l+1}^v, x_{2,l+1}^w)$.

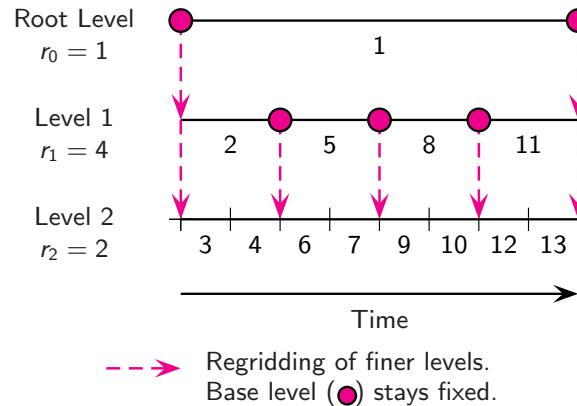
For boundary conditions on \tilde{I}_l^s : linear time interpolation

$$\tilde{\mathbf{Q}}^{l+1}(t + \kappa \Delta t_{l+1}) := \left(1 - \frac{\kappa}{r_{l+1}}\right) \check{\mathbf{Q}}^{l+1}(t) + \frac{\kappa}{r_{l+1}} \check{\mathbf{Q}}^{l+1}(t + \Delta t_l) \quad \text{for } \kappa = 0, \dots, r_{l+1}$$



Recursive integration order

- Space-time interpolation of coarse data to set $I_l^s, l > 0$
- Regridding:
 - Creation of new grids, copy existing cells on level $l > 0$
 - Spatial interpolation to initialize new cells on level $l > 0$



The basic recursive algorithm

```

AdvanceLevel( $l$ )
Repeat  $r_l$  times
  Set ghost cells of  $\mathbf{Q}'(t)$ 
  If time to regrid?
    Regrid()
  UpdateLevel()
  If level  $l + 1$  exists?
    Set ghost cells of  $\mathbf{Q}'(t + \Delta t_l)$ 
    AdvanceLevel( $l + 1$ )
    Average  $\mathbf{Q}'^{l+1}(t + \Delta t_l)$  onto  $\mathbf{Q}'(t + \Delta t_l)$ 
    Correct  $\mathbf{Q}'(t + \Delta t_l)$  with  $\delta\mathbf{F}^{l+1}$ 
     $t := t + \Delta t_l$ 
  ▶ Recursion
  ▶ Restriction and flux correction
  ▶ Re-organization of hierarchical data
Start - Start integration on level 0
 $l = 0, r_0 = 1$ 
AdvanceLevel( $l$ )

```

[Berger and Colella, 1988][Berger and Oliger, 1984]

Regridding algorithm

Regrid(l) - Regrid all levels $\iota > l$

```

For  $\iota = l_f$  Downto  $l$  Do
  Flag  $N^\iota$  according to  $\mathbf{Q}^\iota(t)$ 
  If level  $\iota + 1$  exists?
    Flag  $N^\iota$  below  $\breve{G}^{\iota+2}$ 
    Flag buffer zone on  $N^\iota$ 
    Generate  $\breve{G}^{\iota+1}$  from  $N^\iota$ 
   $\breve{G}_l := G_l$ 
  For  $\iota = l$  To  $l_f$  Do
     $C\breve{G}_\iota := G_0 \setminus \breve{G}_\iota$ 
     $\breve{G}_{\iota+1} := \breve{G}_{\iota+1} \setminus C\breve{G}_\iota$ 
Recompose( $l$ )

```

- Refinement flags:
 $N^\iota := \bigcup_m N(\partial G_{l,m})$
- Activate flags below higher levels
- Flag buffer cells of $b > \kappa_r$ cells, κ_r steps between calls of Regrid(l)
- Special cluster algorithm
- Use complement operation to ensure proper nesting condition

Recomposition of data

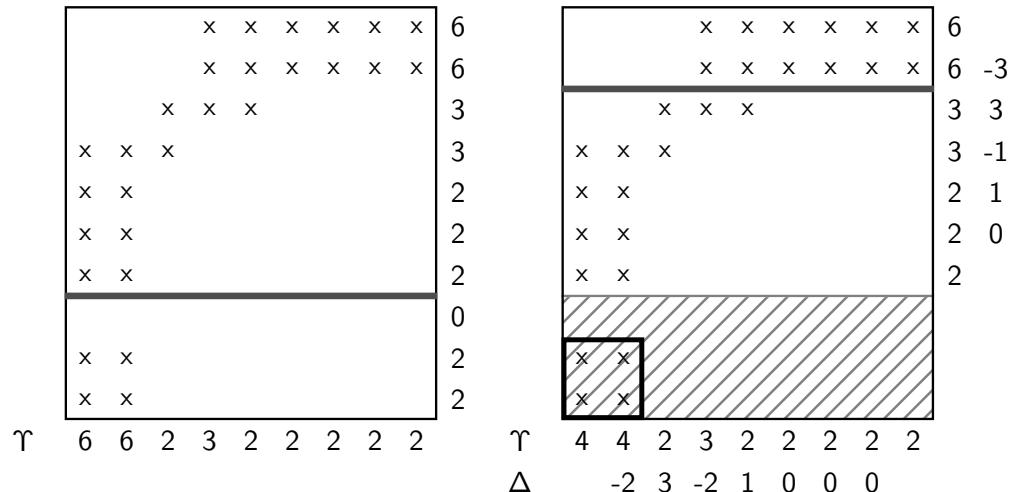
Recompose(l) - Reorganize all levels $\iota > l$

```

For  $\iota = l + 1$  To  $l_f + 1$  Do
  Interpolate  $\mathbf{Q}^{\iota-1}(t)$  onto  $\breve{\mathbf{Q}}^\iota(t)$ 
  Copy  $\mathbf{Q}^\iota(t)$  onto  $\breve{\mathbf{Q}}^\iota(t)$ 
  Set ghost cells of  $\breve{\mathbf{Q}}^\iota(t)$ 
   $\mathbf{Q}^\iota(t) := \breve{\mathbf{Q}}^\iota(t), G_\iota := \breve{G}_\iota$ 
  ▶ Creates max. 1 level above  $l_f$ , but can remove multiple level if  $\breve{G}_\iota$  empty (no coarsening!)
  ▶ Use spatial interpolation on entire data  $\breve{\mathbf{Q}}^\iota(t)$ 
  ▶ Overwrite where old data exists
  ▶ Synchronization and physical boundary conditions

```

Clustering by signatures



Υ Flagged cells per row/column

Δ Second derivative of Υ , $\Delta = \Upsilon_{\nu+1} - 2\Upsilon_\nu + \Upsilon_{\nu-1}$

Technique from image detection: [Bell et al., 1994], see also [Berger and Rigoutsos, 1991], [Berger, 1986]

Refinement criteria

Scaled gradient of scalar quantity w

$$|w(\mathbf{Q}_{j+1,k}) - w(\mathbf{Q}_{jk})| > \epsilon_w, |w(\mathbf{Q}_{j,k+1}) - w(\mathbf{Q}_{jk})| > \epsilon_w, |w(\mathbf{Q}_{j+1,k+1}) - w(\mathbf{Q}_{jk})| > \epsilon_w$$

Heuristic error estimation [Berger, 1982]:

Local truncation error of scheme of order o

$$\mathbf{q}(\mathbf{x}, t + \Delta t) - \mathcal{H}^{(\Delta t)}(\mathbf{q}(\cdot, t)) = \mathbf{C} \Delta t^{o+1} + O(\Delta t^{o+2})$$

For \mathbf{q} smooth after 2 steps Δt

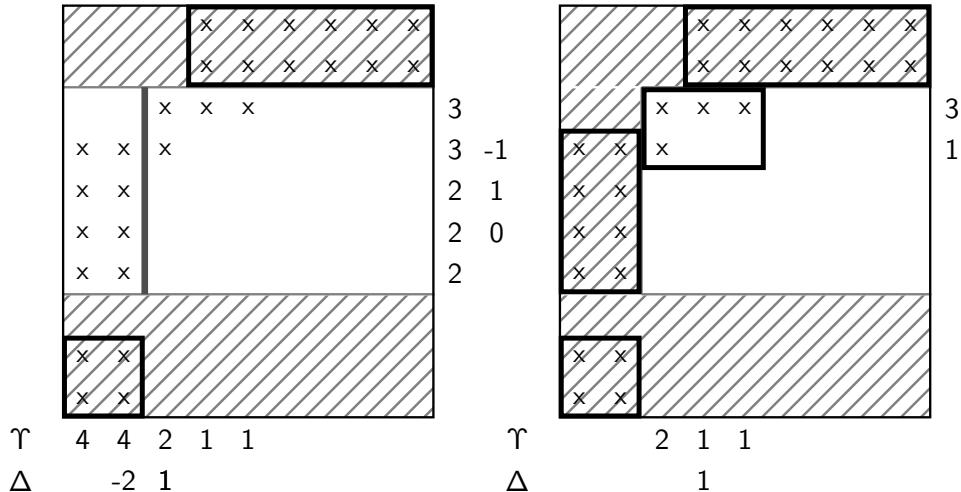
$$\mathbf{q}(\mathbf{x}, t + \Delta t) - \mathcal{H}_2^{(\Delta t)}(\mathbf{q}(\cdot, t - \Delta t)) = 2\mathbf{C} \Delta t^{o+1} + O(\Delta t^{o+2})$$

and after 1 step with $2\Delta t$

$$\mathbf{q}(\mathbf{x}, t + \Delta t) - \mathcal{H}^{(2\Delta t)}(\mathbf{q}(\cdot, t - \Delta t)) = 2^{o+1}\mathbf{C} \Delta t^{o+1} + O(\Delta t^{o+2})$$

Gives

$$\mathcal{H}_2^{(\Delta t)}(\mathbf{q}(\cdot, t - \Delta t)) - \mathcal{H}^{(2\Delta t)}(\mathbf{q}(\cdot, t - \Delta t)) = (2^{o+1} - 2)\mathbf{C} \Delta t^{o+1} + O(\Delta t^{o+2})$$

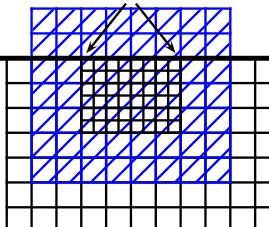


Recursive generation of $\tilde{G}_{l,m}$

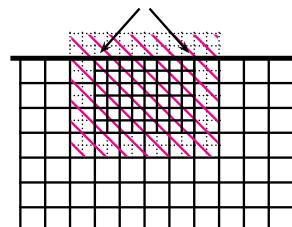
1. 0 in Υ
2. Largest difference in Δ
3. Stop if ratio between flagged and unflagged cell $> \eta_{tol}$

Heuristic error estimation for FV methods

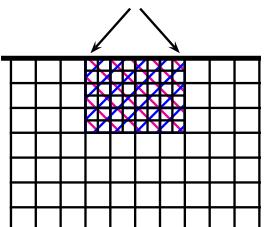
2. Create temporary Grid coarsened by factor 2 Initialize with fine-grid values of preceding time step



1. Error estimation on interior cells



3. Compare temporary solutions



$$\begin{aligned} \mathcal{H}^{\Delta t_l} \mathbf{Q}^l(t_l - \Delta t_l) &= \mathcal{H}^{\Delta t_l} (\mathcal{H}^{\Delta t_l} \mathbf{Q}^l(t_l - \Delta t_l)) \\ &= \mathcal{H}_2^{\Delta t_l} \mathbf{Q}^l(t_l - \Delta t_l) \\ &= \mathcal{H}^{2\Delta t_l} \bar{\mathbf{Q}}^l(t_l - \Delta t_l) \end{aligned}$$

Usage of heuristic error estimation

Current solution integrated tentatively 1 step with Δt_l and coarsened

$$\bar{Q}(t_l + \Delta t_l) := \text{Restrict} \left(\mathcal{H}_2^{\Delta t_l} \mathbf{Q}'(t_l - \Delta t_l) \right)$$

Previous solution coarsened and integrated 1 step with $2\Delta t_l$

$$Q(t_l + \Delta t_l) := \mathcal{H}^{2\Delta t_l} \text{Restrict} (\mathbf{Q}'(t_l - \Delta t_l))$$

Local error estimation of scalar quantity w

$$\tau_{jk}^w := \frac{|w(\bar{Q}_{jk}(t + \Delta t)) - w(Q_{jk}(t + \Delta t))|}{2^{o+1} - 2}$$

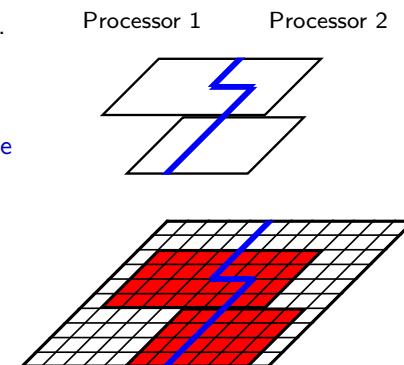
In practice [Deiterding, 2003] use

$$\frac{\tau_{jk}^w}{\max(|w(Q_{jk}(t + \Delta t))|, S_w)} > \eta_w^r$$

Parallelization strategies

Decomposition of the hierarchical data

- ▶ Distribution of each grid
- ▶ Separate distribution of each level, cf. [Rendleman et al., 2000]
- ▶ Rigorous domain decomposition
 - ▶ Data of all levels resides on same node
 - ▶ Grid hierarchy defines unique "floor-plan"
 - ▶ Redistribution of data blocks during reorganization of hierarchical data
 - ▶ Synchronization when setting ghost cells



Outline

Meshes and adaptation

Adaptivity on unstructured and structured meshes
Available SAMR software

The serial Berger-Colella SAMR method

Data structures and numerical update
Conservative flux correction
Level transfer operators
The basic recursive algorithm
Block generation and flagging of cells

Parallel SAMR method

Domain decomposition
A parallel SAMR algorithm

Rigorous domain decomposition formalized

Parallel machine with P identical nodes. P non-overlapping portions G_0^p , $p = 1, \dots, P$ as

$$G_0 = \bigcup_{p=1}^P G_0^p \quad \text{with} \quad G_0^p \cap G_0^q = \emptyset \quad \text{for } p \neq q$$

Higher level domains G_I follow decomposition of root level

$$G_I^p := G_I \cap G_0^p$$

With $\mathcal{N}_I(\cdot)$ denoting number of cells, we estimate the workload as

$$\mathcal{W}(\Omega) = \sum_{I=0}^{I_{\max}} \left[\mathcal{N}_I(G_I \cap \Omega) \prod_{\kappa=0}^I r_\kappa \right]$$

Equal work distribution necessitates

$$\mathcal{L}^p := \frac{P \cdot \mathcal{W}(G_0^p)}{\mathcal{W}(G_0)} \approx 1 \quad \text{for all } p = 1, \dots, P$$

[Deiterding, 2005]

Ghost cell setting

Local synchronization

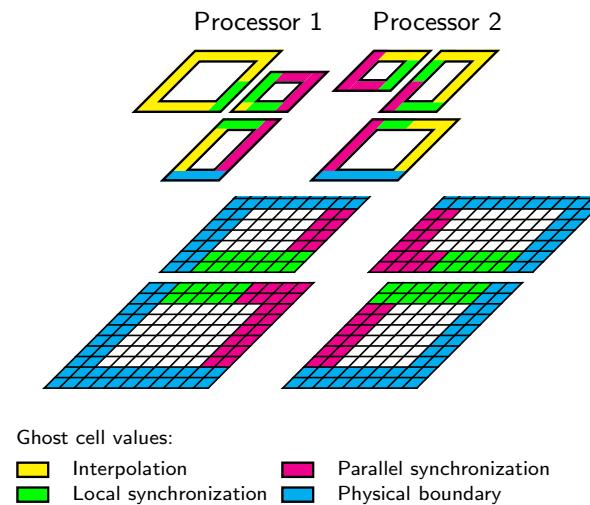
$$\tilde{S}_{l,m}^{s,p} = \tilde{G}_{l,m}^{s,p} \cap G_l^p$$

Parallel synchronization

$$\tilde{S}_{l,m}^{s,q} = \tilde{G}_{l,m}^{s,p} \cap G_l^q, q \neq p$$

Interpolation and physical boundary conditions remain strictly local

- ▶ Scheme $\mathcal{H}(\Delta t)$ evaluated locally
- ▶ Restriction and prolongation local



The recursive algorithm in parallel

```

AdvanceLevel(l)

Repeat r_l times
  Set ghost cells of Q^l(t)
  If time to regrid?
    Regrid(l)
    UpdateLevel(l)
    If level l + 1 exists?
      Set ghost cells of Q^l(t + Δt_l)
      AdvanceLevel(l + 1)
      Average Q^{l+1}(t + Δt_l) onto Q^l(t + Δt_l)
      Correct Q^l(t + Δt_l) with δF^{l+1}
      t := t + Δt_l

  UpdateLevel(l)

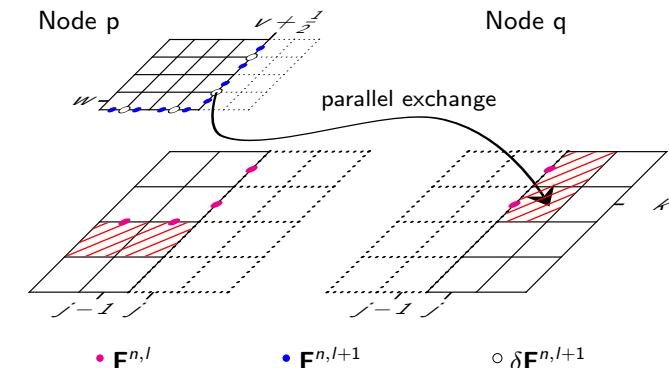
  For all m = 1 To M_l Do
    Q(G_{l,m}^s, t)  $\xrightarrow{\mathcal{H}(\Delta t_l)}$  Q(G_{l,m}, t + Δt_l), F^n(̄G_{l,m}, t)
    If level l > 0
      Add F^n(∂G_{l,m}, t) to δF^{n,l}
    If level l + 1 exists
      Init δF^{n,l+1} with F^n(̄G_{l,m} ∩ ∂G_{l+1}, t)

```

- ▶ Numerical update strictly local
- ▶ Inter-level transfer local
- ▶ Parallel synchronization
- ▶ Application of δF^{l+1} on ∂G_l^q

Parallel flux correction

1. Strictly local: Init $\delta F^{n,l+1}$ with $F^n(\bar{G}_{l,m} \cap \partial G_{l+1}, t)$
2. Strictly local: Add $F^n(\partial G_{l,m}, t)$ to $\delta F^{n,l}$
3. Parallel communication: Correct $Q^l(t + \Delta t_l)$ with δF^{l+1}



Regridding algorithm in parallel

Regrid(l) – Regrid all levels $\nu > l$

```

For ν = l_f Downto l Do
  Flag N^ν according to Q^l(t)
  If level ν + 1 exists?
    Flag N^ν below ̄G^{ν+2}
    Flag buffer zone on N^ν
    Generate ̄G^{ν+1} from N^ν
    ̄G_l := ̄G_l
    For ν = l To l_f Do
      C̄G_ν := ̄G_ν \ ̄G_l
      ̄G_{ν+1} := ̄G_{ν+1} \ C̄G_l^{ν+1}

```

Recompose(l)

- ▶ Need a ghost cell overlap of b cells to ensure correct setting of refinement flags in parallel
- ▶ Two options exist (we choose the latter):
 - ▶ Global clustering algorithm
 - ▶ Local clustering algorithm and concatenation of new lists \check{G}^{l+1}

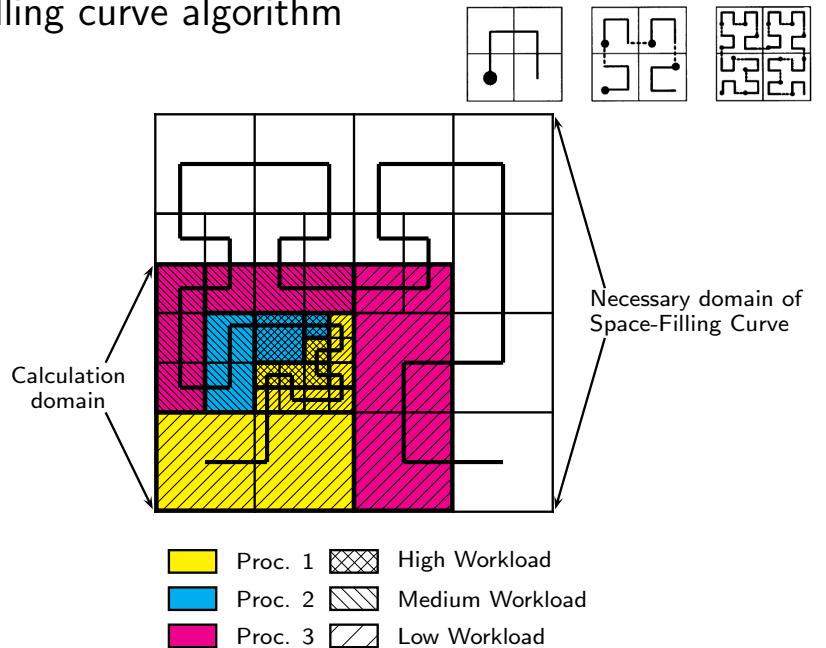
Recomposition algorithm in parallel

Recompose(/) - Reorganize all levels

```
Generate  $G_0^P$  from  $\{G_0, \dots, G_l, \check{G}_{l+1}, \dots, \check{G}_{l_f+1}\}$ 
For  $\iota = l + 1$  To  $l_f + 1$  Do
  If  $\iota > l$ 
     $\check{G}_\iota^P := \check{G}_\iota \cap G_0^P$ 
    Interpolate  $\mathbf{Q}^{\iota-1}(t)$  onto  $\check{\mathbf{Q}}^\iota(t)$ 
  else
     $\check{G}_\iota^P := G_\iota \cap G_0^P$ 
    If  $\iota > 0$ 
      Copy  $\delta\mathbf{F}^{n,\iota}$  onto  $\check{\delta\mathbf{F}}^{n,\iota}$ 
       $\delta\mathbf{F}^{n,\iota} := \check{\delta\mathbf{F}}^{n,\iota}$ 
    If  $\iota \geq l$  then  $\kappa_\iota = 0$  else  $\kappa_\iota = 1$ 
    For  $\kappa = 0$  To  $\kappa_\iota$  Do
      Copy  $\mathbf{Q}^\iota(t)$  onto  $\check{\mathbf{Q}}^\iota(t)$ 
      Set ghost cells of  $\check{\mathbf{Q}}^\iota(t)$ 
       $\check{\mathbf{Q}}^\iota(t) := \check{\mathbf{Q}}^\iota(t)$ 
     $G_\iota := \check{G}_\iota$ 
```

- ▶ Global redistribution can also be required when regridding higher levels and G_0, \dots, G_l do not change (drawback of domain decomposition)
- ▶ When $\iota > l$ do nothing special
- ▶ For $\iota \leq l$, redistribute additionally
 - ▶ Flux corrections $\delta\mathbf{F}^{n,\iota}$
 - ▶ Already updated time level $\mathbf{Q}^\iota(t + \kappa\Delta t_\iota)$

Space-filling curve algorithm



References I

- [Bell et al., 1994] Bell, J., Berger, M., Saltzman, J., and Welcome, M. (1994). Three-dimensional adaptive mesh refinement for hyperbolic conservation laws. *SIAM J. Sci. Comp.*, 15(1):127–138.
- [Berger, 1982] Berger, M. (1982). *Adaptive mesh refinement for hyperbolic differential equations*. PhD thesis, Stanford University. Report No. STAN-CS-82-924.
- [Berger, 1986] Berger, M. (1986). Data structures for adaptive grid generation. *SIAM J. Sci. Stat. Comput.*, 7(3):904–916.
- [Berger and Colella, 1988] Berger, M. and Colella, P. (1988). Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82:64–84.
- [Berger and LeVeque, 1998] Berger, M. and LeVeque, R. (1998). Adaptive mesh refinement using wave-propagation algorithms for hyperbolic systems. *SIAM J. Numer. Anal.*, 35(6):2298–2316.
- [Berger and Oliger, 1984] Berger, M. and Oliger, J. (1984). Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53:484–512.

References II

- [Berger and Rigoutsos, 1991] Berger, M. and Rigoutsos, I. (1991). An algorithm for point clustering and grid generation. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(5):1278–1286.
- [Brown et al., 1997] Brown, D. L., Henshaw, W. D., and Quinlan, D. J. (1997). Overture: An object oriented framework for solving partial differential equations. In *Proc. ISCOPE 1997, appeared in Scientific Computing in Object-Oriented Parallel Environments*, number 1343 in Springer Lecture Notes in Computer Science.
- [Deiterding, 2003] Deiterding, R. (2003). *Parallel adaptive simulation of multi-dimensional detonation structures*. PhD thesis, Brandenburgische Technische Universität Cottbus.
- [Deiterding, 2005] Deiterding, R. (2005). Construction and application of an AMR algorithm for distributed memory computers. In Plewa, T., Linde, T., and Weirs, V. G., editors, *Adaptive Mesh Refinement - Theory and Applications*, volume 41 of *Lecture Notes in Computational Science and Engineering*, pages 361–372. Springer.
- [Gittings et al., 2008] Gittings, M., Weaver, R., Clover, M., Betlach, T., Byrne, N., Coker, R., Dendy, E., Hueckstaedt, R., New, K., Oakes, R., Rantal, D., and Stefan, R. (2008). The RAGE radiation-hydrodynamics code. *Comput. Sci. Disc.*, 1. doi:10.1088/1749-4699/1/1/015005.

References III

[Hornung et al., 2006] Hornung, R. D., Wissink, A. M., and Kohn, S. H. (2006). Managing complex data and geometry in parallel structured AMR applications. *Engineering with Computers*, 22:181–195.

[MacNeice et al., 2000] MacNeice, P., Olson, K. M., Mobarry, C., deFainchtein, R., and Packer, C. (2000). PARAMESH: A parallel adaptive mesh refinement community toolkit. *Computer Physics Communications*, 126:330–354.

[Parashar and Browne, 1997] Parashar, M. and Browne, J. C. (1997). System engineering for high performance computing software: The HDDA/DAGH infrastructure for implementation of parallel structured adaptive mesh refinement. In *Structured Adaptive Mesh Refinement Grid Methods*, IMA Volumes in Mathematics and its Applications. Springer.

[Rendleman et al., 2000] Rendleman, C. A., Beckner, V. E., Lijewski, M., Crutchfield, W., and Bell, J. B. (2000). Parallelization of structured, hierarchical adaptive mesh refinement algorithms. *Computing and Visualization in Science*, 3:147–157.

Lecture 3

Hyperbolic AMROC solvers

Course *Block-structured Adaptive Finite Volume Methods in C++*

Ralf Deiterding

University of Southampton

Engineering and the Environment

Highfield Campus, Southampton SO17 1BJ, UK

E-mail: r.deiterding@soton.ac.uk

Structured adaptive mesh refinement High-resolution methods oooooo	AMROC oooooooo	Clawpack solver oooooooooooo	WENO solver ooooooo	MHD solver oooooo	References ooo	39	Hyperbolic AMROC solvers High-resolution methods oooooo	AMROC oooooooooooo	Clawpack solver oooooooooooo	WENO solver ooooooo	MHD solver oooooo	References ooo	1
--	-------------------	---------------------------------	------------------------	----------------------	-------------------	----	---	-----------------------	---------------------------------	------------------------	----------------------	-------------------	---

Outline

High-resolution methods

- MUSCL and wave propagation
- Further methods

AMROC

- Overview and basic software design
- Classes

Clawpack solver

- AMR examples
- Software construction

WENO solver

- Large-eddy simulation
- Software construction

MHD solver

- Ideal magneto-hydrodynamics simulation
- Software design

Outline

High-resolution methods

- MUSCL and wave propagation
- Further methods

AMROC

- Overview and basic software design
- Classes

Clawpack solver

- AMR examples
- Software construction

WENO solver

- Large-eddy simulation
- Software construction

MHD solver

- Ideal magneto-hydrodynamics simulation
- Software design

Further high-resolution methods

Some further high-resolution methods (good overview in [Laney, 1998])

- ▶ FCT: 2nd order [Oran and Boris, 2001]
 - ▶ ENO/WENO: 3rd order [Shu, 97]
 - ▶ PPM: 3rd order [Colella and Woodward, 1984]

3rd order methods must make use of strong-stability preserving Runge-Kutta methods [Gottlieb et al., 2001] for time integration that use a multi-step update

$$\tilde{\mathbf{Q}}_j^v = \alpha_v \mathbf{Q}_j^n + \beta_v \tilde{\mathbf{Q}}_j^{v-1} + \gamma_v \frac{\Delta t}{\Lambda_v} \left(\mathbf{F}_{j+\frac{1}{2}}(\tilde{\mathbf{Q}}^{v-1}) - \mathbf{F}_{j-\frac{1}{2}}(\tilde{\mathbf{Q}}^{v-1}) \right)$$

with $\tilde{\mathbf{Q}}^0 := \mathbf{Q}^n$, $\alpha_1 = 1$, $\beta_1 = 0$; and $\mathbf{Q}^{n+1} := \tilde{\mathbf{Q}}^\gamma$ after final stage

Typical storage-efficient SSPRK(3,3)

$$\tilde{\mathbf{Q}}^1 = \mathbf{Q}^n + \Delta t \mathcal{F}(\mathbf{Q}^n), \quad \tilde{\mathbf{Q}}^2 = \frac{3}{4}\mathbf{Q}^n + \frac{1}{4}\tilde{\mathbf{Q}}^1 + \frac{1}{4}\Delta t \mathcal{F}(\tilde{\mathbf{Q}}^1),$$

$$\mathbf{Q}^{n+1} = \frac{1}{3}\mathbf{Q}^n + \frac{2}{3}\tilde{\mathbf{Q}}^2 + \frac{2}{3}\Delta t \mathcal{F}(\tilde{\mathbf{Q}}^2)$$

- ▶ “Adaptive Mesh Refinement in Object-oriented C++”
 - ▶ $\sim 46,000$ LOC for C++ SAMR kernel, $\sim 140,000$ total C++, C, Fortran-77
 - ▶ uses parallel hierarchical data structures that have evolved from DAGH
 - ▶ Implements explicit SAMR with different finite volume solvers
 - ▶ Embedded boundary method, FSI coupling
 - ▶ The Virtual Test Facility: AMROC V2.0 plus solid mechanics solvers
 - ▶ $\sim 430,000$ lines of code total in C++, C, Fortran-77, Fortran-90
 - ▶ autoconf / automake environment with support for typical parallel high-performance system
 - ▶ <http://www.vtf.website> [Deiterding et al., 2006][Deiterding et al., 2006]

Outline

High-resolution methods

MUSCL and wave propagation Further methods

AMROC

Overview and basic software design Classes

Clawpack solver

AMR examples

WENO solver

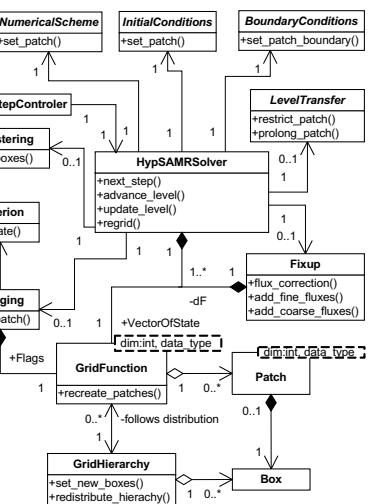
Large-eddy simulation Software construction

MHD solver

Ideal magneto-hydrodynamics simulation Software design

UML design of AMROC

- ▶ Classical framework approach with generic main program in C++
 - ▶ Customization / modification in Problem.h include file by derivation from base classes and redefining virtual interface functions
 - ▶ Predefined, scheme-specific classes provided for standard simulations
 - ▶ Clawpack, WENO: Standard simulations require only linking to F77 functions for initial and boundary conditions, source terms. No C++ knowledge required
 - ▶ Expert usage (algorithm modification, advanced output, etc.) in C++



AMR level - II

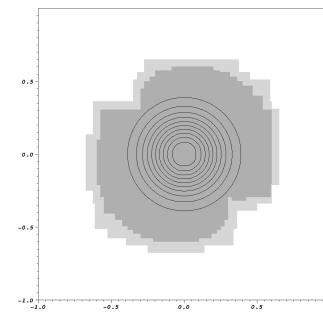
- ▶ **AMRFixup<VectorType, FixupType, dim>** implements the conservative flux correction, holds lower dimensional GridFunctions for correction terms
[code/amroc/doc/html/amr/classAMRFixup.html](#)
- ▶ **AMRFlagging<VectorType, FixupType, FlagType, dim>** calls a list of refinement criteria and stores results in scalar GridFunction for flags. All criteria are in amroc/amr/Criteria
[code/amroc/doc/html/amr/classAMRFlagging.html](#)
- ▶ **LevelTransfer<VectorType, dim>** provides patch-wise interpolation and restriction routines that are passed as parameters to GridFunction
[code/amroc/doc/html/amr/classLevelTransfer.html](#)
- ▶ **AMRTimestep** implements time step control for a Solver
[code/amroc/doc/html/amr/classAMRTimestep.html](#)
- ▶ **AMRInterpolation<VectorType, dim>** is an interpolation at arbitrary point location, typically used for post-processing
[code/amroc/doc/html/amr/classAMRInterpolation.html](#)

SAMR accuracy verification

Gaussian density shape

$$\rho(x_1, x_2) = 1 + e^{-\left(\frac{\sqrt{x_1^2+x_2^2}}{R}\right)^2}$$

is advected with constant velocities $u_1 = u_2 \equiv 1$, $p_0 \equiv 1$, $R = 1/4$



- ▶ Domain $[-1, 1] \times [-1, 1]$, periodic boundary conditions, $t_{end} = 2$
- ▶ Two levels of adaptation with $r_{1,2} = 2$, finest level corresponds to $N \times N$ uniform grid

Use *locally* conservative interpolation

$$\tilde{\mathbf{Q}}_{v,w}^l := \mathbf{Q}_{ij}^l + f_1(\mathbf{Q}_{i+1,j}^l - \mathbf{Q}_{i-1,j}^l) + f_2(\mathbf{Q}_{i,j+1}^l - \mathbf{Q}_{i,j-1}^l)$$

with factor $f_1 = \frac{x_{1,l+1}^v - x_{1,l}^v}{2\Delta x_{1,l}}$, $f_2 = \frac{x_{2,l+1}^w - x_{2,l}^w}{2\Delta x_{2,l}}$ to also test flux correction

This prolongation operator is not monotonicity preserving! Only applicable to smooth problems.

[code/amroc/doc/html/apps/clawpack_2applications_2euler_22d_2GaussianPulseAdvection_2src_2Problem_8h_source.html](#)

Outline

High-resolution methods

MUSCL and wave propagation
Further methods

AMROC

Overview and basic software design
Classes

Clawpack solver

AMR examples
Software construction

WENO solver

Large-eddy simulation
Software construction

MHD solver

Ideal magneto-hydrodynamics simulation
Software design

SAMR accuracy verification: results

VanLeer flux vector splitting with dimensional splitting, Minmod limiter

N	Unigrid		SAMR - fixup			SAMR - no fixup		
	Error	Order	Error	Order	$\Delta\rho$	Error	Order	$\Delta\rho$
20	0.10946400							
40	0.04239430	1.369						
80	0.01408160	1.590	0.01594820		0	0.01595980		2e-5
160	0.00492945	1.514	0.00526693	1.598	0	0.00530538	1.589	2e-5
320	0.00146132	1.754	0.00156516	1.751	0	0.00163837	1.695	-1e-5
640	0.00041809	1.805	0.00051513	1.603	0	0.00060021	1.449	-6e-5

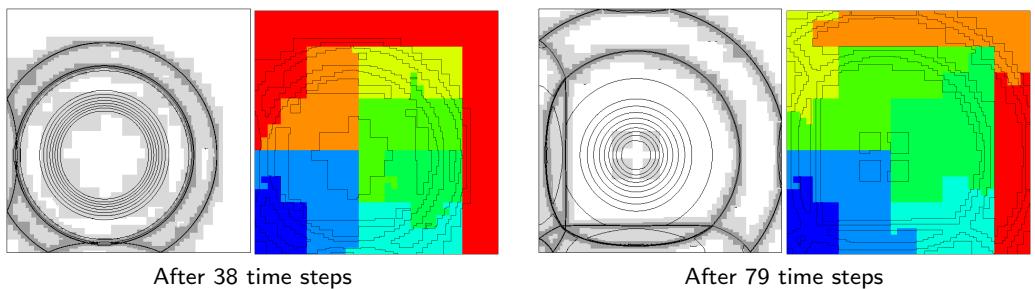
Fully two-dimensional Wave Propagation Method, Minmod limiter

N	Unigrid		SAMR - fixup			SAMR - no fixup		
	Error	Order	Error	Order	$\Delta\rho$	Error	Order	$\Delta\rho$
20	0.10620000							
40	0.04079600	1.380						
80	0.01348250	1.598	0.01536580		0	0.01538820		2e-5
160	0.00472301	1.513	0.00505406	1.604	0	0.00510499	1.592	5e-5
320	0.00139611	1.758	0.00147218	1.779	0	0.00152387	1.744	7e-5
640	0.00039904	1.807	0.00044500	1.726	0	0.00046587	1.710	6e-5

Benchmark run: blast wave in 2D

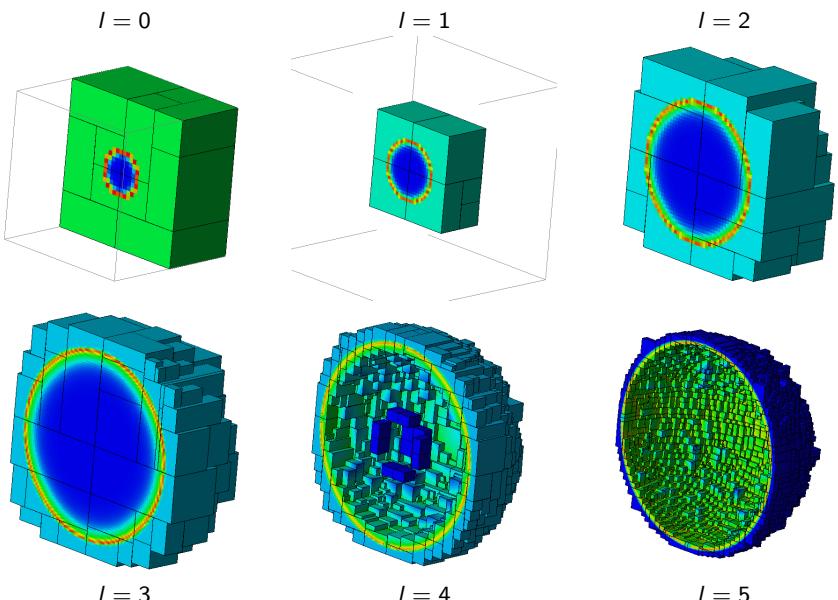
- ▶ 2D-Wave-Propagation Method with Roe's approximate solver
- ▶ Base grid 150×150
- ▶ 2 levels: factor 2, 4

Task [%]	$P=1$	$P=2$	$P=4$	$P=8$	$P=16$
Update by $\mathcal{H}^{(1)}$	86.6	83.4	76.7	64.1	51.9
Flux correction	1.2	1.6	3.0	7.9	10.7
Boundary setting	3.5	5.7	10.1	15.6	18.3
Recomposition	5.5	6.1	7.4	9.9	14.0
Misc.	4.9	3.2	2.8	2.5	5.1
Time [min]	151.9	79.2	43.4	23.3	13.9
Efficiency [%]	100.0	95.9	87.5	81.5	68.3



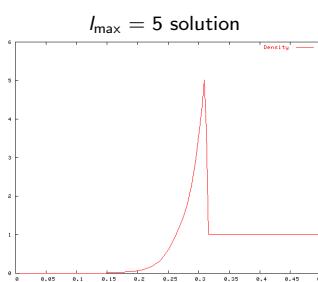
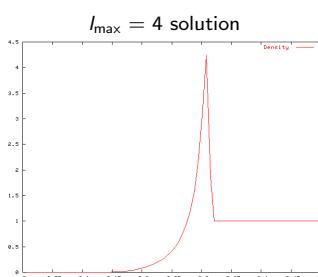
code/amroc/doc/html/apps/clawpack_2applications_2euler_22d_2Box_2src_2Problem_8h_source.html

Benchmark run 2: visualization of refinement



Benchmark run 2: point-explosion in 3D

- ▶ Benchmark from the Chicago workshop on AMR methods, September 2003
- ▶ Sedov explosion - energy deposition in sphere of radius 4 finest cells
- ▶ 3D-Wave-Prop. Method with hybrid Roe-HLL scheme
- ▶ Base grid 32^3
- ▶ Refinement factor $r_l = 2$
- ▶ Effective resolutions: 128^3 , 256^3 , 512^3 , 1024^3
- ▶ Grid generation efficiency $\eta_{tol} = 85\%$
- ▶ Proper nesting enforced
- ▶ Buffer of 1 cell



Benchmark run 2: performance results

Number of grids and cells

I	$I_{\max} = 2$		$I_{\max} = 3$		$I_{\max} = 4$		$I_{\max} = 5$	
	Grids	Cells	Grids	Cells	Grids	Cells	Grids	Cells
0	28	32,768	28	32,768	33	32,768	34	32,768
1	8	32,768	14	32,768	20	32,768	20	32,768
2	63	115,408	49	116,920	43	125,680	50	125,144
3			324	398,112	420	555,744	193	572,768
4					1405	1,487,312	1,498	1,795,048
5						5,266	5,266	5,871,128
Σ		180,944		580,568		2,234,272		8,429,624

Breakdown of CPU time on 8 nodes SGI Altix 3000 (Linux-based shared memory system)

Task [%]	$I_{\max} = 2$		$I_{\max} = 3$		$I_{\max} = 4$		$I_{\max} = 5$	
	Integration	Fixup	Integration	Fixup	Integration	Fixup	Integration	Fixup
Integration	73.7	2.6	77.2	46	72.9	2.6	37.8	45
Fixup								
Boundary	10.1	79	6.3	78	5.1	56	6.9	78
Recomposition	7.4		8.0		15.1		50.4	
Clustering	0.5		0.6		0.7		1.0	
Output/Misc	5.7		4.0		3.6		1.7	
Time [min]	0.5		5.1		73.0		2100.0	
Uniform [min]	5.4		160		~5,000		~180,000	
Factor of AMR savings	11		31		69		86	
Time steps	15		27		52		115	

code/amroc/doc/html/apps/clawpack_2applications_2euler_23d_2Sedov_2src_2Problem_8h_source.html

Components

Directory `amroc/clawpack/src` contains generic Fortran functions:

- ▶ **?d/integrator_extended**: Contains an extended version of Clawpack 3.0 by R. LeVeque. The MUSCL approach was added, 3d fully implemented, interfaces have been adjusted for AMROC. These codes are equation independent.
[code/amroc/doc/html/clp/files.html](#)
- ▶ **?d/equations**: Contains equation-specific Riemann solvers, flux functions as F77 routines.
- ▶ **?d/interpolation**: Contains patch-wise interpolation and restriction operators in F77.

Directory `amroc/clawpack` contains the generic C++ classes to interface the F77 library from ?d/integrator_extended with AMROC:

- ▶ **ClpIntegrator<VectorType, AuxVectorType, dim >**: Interfaces the F77 library from ?d/integrator_extended to `Integrator<VectorType, dim>`. Key function to fill is `CalculateGrid()`.

[code/amroc/doc/html/clp/classClpIntegrator_3_01VectorType_00_01AuxVectorType_00_012_01_4.html](#)

Components - II

- ▶ **ClpFixup<VectorType, FixupType, AuxVectorType, dim >**: The conservative flux correction is more complex in the waves of the flux difference splitting schemes. This specialization of `AMRFixup<VectorType, FixupType, dim>` considers this.

[code/amroc/doc/html/clp/classClpFixup.html](#)

- ▶ A generic main program **amroc/clawpack/mains/amr_main.C** instantiates `Integrator<VectorType, dim>`, `InitialCondition<VectorType, dim >`, `BoundaryConditions<VectorType, dim >`

[code/amroc/doc/html/clp/amr_main_8C.html](#)

- ▶ **Problem.h**: Allows simulation-specific alteration in class-library style by derivation from predefined classes specified in **ClpStdProblem.h**

[code/amroc/doc/html/clp/ClpStdProblem_8h.html](#)

- ▶ **ClpProblem.h**: General include before equation-specific C++ definition file is read that defines `VectorType` and provides Fortran function names required by `amroc/amr/F77Interfaces` classes

[code/amroc/doc/html/clp/ClpProblem_8h_source.html](#) [code/amroc/doc/html/clp/euler2_8h.html](#)

Functions to link in Makefile.am

Interface objects from `amroc/amr/F77Interfaces` are used to mimic the interface of standard Clawpack, which constructs specific simulations by linking F77 functions. Required functions are:

- ▶ **init.f**: Initial conditions.
- ▶ **physbd.f**: Boundary conditions.
- ▶ **combl.f**: Initialize application specific common blocks.
- ▶ **`$($EQUATION)/rp/rpn.f` and `rpt.f`**: Equation-specific Riemann solvers in normal and transverse direction.
- ▶ **`$($EQUATION)/rp/flx.f`, `$($EQUATION)/rp/rec.f`**: Flux and reconstruction for MUSCL slope limiting (if used), otherwise dummy-routines/flx.f and dummy-routines/rec.f may be used.
- ▶ **`$($EQUATION)/rp/chk.f`**: Physical consistency check for debugging.
- ▶ **`src.f`**: Source term for a splitting method., otherwise dummy-routines/src.f can be linked.
- ▶ **`setaux.f`**: Set data in an additional patch-wise auxiliary array, otherwise dummy-routines/saux.f can be linked.

Outline

High-resolution methods

MUSCL and wave propagation
Further methods

AMROC

Overview and basic software design
Classes

Clawpack solver

AMR examples
Software construction

WENO solver

Large-eddy simulation
Software construction

MHD solver

Ideal magneto-hydrodynamics simulation
Software design

Favre-averaged Navier-Stokes equations

$$\begin{aligned} \frac{\partial \bar{\rho}}{\partial t} + \frac{\partial}{\partial x_n} (\bar{\rho} \tilde{u}_n) &= 0 \\ \frac{\partial}{\partial t} (\bar{\rho} \tilde{u}_k) + \frac{\partial}{\partial x_n} (\bar{\rho} \tilde{u}_k \tilde{u}_n + \delta_{kn} \bar{\rho} - \tilde{\tau}_{kn} + \sigma_{kn}) &= 0 \\ \frac{\partial \bar{\rho} \bar{E}}{\partial t} + \frac{\partial}{\partial x_n} (\tilde{u}_n (\bar{\rho} \bar{E} + \bar{p}) + \tilde{q}_n - \tilde{\tau}_{nj} \tilde{u}_j + \sigma_n^e) &= 0 \\ \frac{\partial}{\partial t} (\bar{\rho} \tilde{Y}_i) + \frac{\partial}{\partial x_n} (\bar{\rho} \tilde{Y}_i \tilde{u}_n + \tilde{J}_n^i + \sigma_n^i) &= 0 \end{aligned}$$

with stress tensor

$$\tilde{\tau}_{kn} = \tilde{\mu} \left(\frac{\partial \tilde{u}_n}{\partial x_k} + \frac{\partial \tilde{u}_k}{\partial x_n} \right) - \frac{2}{3} \tilde{\mu} \frac{\partial \tilde{u}_j}{\partial x_i} \delta_{in},$$

heat conduction

$$\tilde{q}_n = -\tilde{\lambda} \frac{\partial \tilde{T}}{\partial},$$

and inter-species diffusion

$$\tilde{J}_n^i = -\bar{\rho} \tilde{D}_i \frac{\partial \tilde{Y}_i}{\partial x_n}$$

Favre-filtering

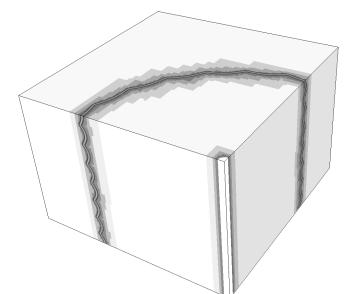
$$\tilde{\phi} = \frac{\bar{\rho}\phi}{\bar{\rho}} \quad \text{with} \quad \bar{\phi}(\mathbf{x}, t; \Delta_c) = \int_{\Omega} G(\mathbf{x} - \mathbf{x}'; \Delta_c) \phi(\mathbf{x}', t) d\mathbf{x}'$$

Numerical solution approach

- Subgrid terms σ_{kn} , σ_n^e , σ_n^i are computed by Pullin's stretched-vortex model
 - Cutoff Δ_c is set to local SAMR resolution Δx_l
 - It remains to solve the Navier-Stokes equations in the hyperbolic regime
 - 3rd order WENO method (hybridized with a tuned centered difference stencil) for convection
 - 2nd order conservative centered differences for diffusion

Example: Cylindrical Richtmyer-Meshkov instability

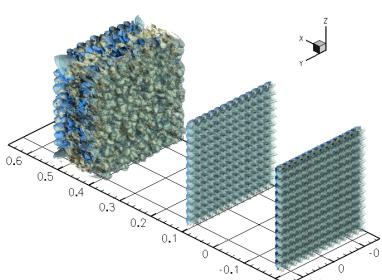
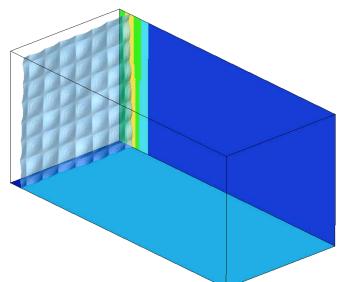
- ▶ Sinusoidal interface between two gases hit by shock wave
 - ▶ Objective is correctly predict turbulent mixing
 - ▶ Embedded boundary method used to regularize apex
 - ▶ AMR base grid $95 \times 95 \times 64$ cells, $r_{1,2,3} = 2$
 - ▶ $\sim 70,000$ h CPU on 32 AMD 2.5GHZ-quad-core nodes



Planar Richtmyer-Meshkov instability

- ▶ Perturbed Air-SF₆ interface shocked and re-shocked by Mach 1.5 shock
 - ▶ Containment of turbulence in refined zones
 - ▶ 96 CPUs IBM SP2-Power3
 - ▶ WENO-TCD scheme with LES model
 - ▶ AMR base grid $172 \times 56 \times 56$, $r_{1,2} = 2$, 10 M cells in average instead of 3 M (uniform)

Task	2ms (%)	5ms (%)	10ms (%)
Integration	45.3	65.9	52.0
Boundary setting	44.3	28.6	41.9
Flux correction	7.2	3.4	4.1
Interpolation	0.9	0.4	0.3
Reorganization	1.6	1.2	1.2
Misc.	0.6	0.5	0.5
Max. imbalance	1.25	1.23	1.30



Components

Directory `amroc/weno/src` contains the Fortran-90 solver library:

- ▶ **generic**: Implements the hybrid WENO-TCD method for Euler and Navier-Stokes equations, characteristic boundary conditions. Code uses F90 modules.
[code/amroc/doc/html/weno/files.html](#)
- ▶ **equations**: Contains routines that specify between LES and laminar flow, the criterion for scheme hybridization, source terms handled by splitting.

Directory `amroc/weno` contains the generic C++ class to interface the F90 library with AMROC:

- ▶ **WENOIntegrator<VectorType, dim >**: Interfaces the F90 solver to `Integrator<VectorType, dim>`. `CalculateGrid()` is called separately for each stage of the Runge-Kutta time integrator.
[code/amroc/doc/html/weno/classWENOIntegrator.html](#)
- ▶ **WENOFixup<VectorType, FixupType, dim >**: A specialized conservative flux correction that accumulates the correction terms throughout the stages of the Runge-Kutta time integrator.
[code/amroc/doc/html/weno/classWENOFixup.html](#)

Outline

High-resolution methods

MUSCL and wave propagation
Further methods

AMROC

Overview and basic software design
Classes

Clawpack solver

AMR examples
Software construction

WENO solver

Large-eddy simulation
Software construction

MHD solver

Ideal magneto-hydrodynamics simulation
Software design

Components - II

- ▶ **WENOInterpolation<VectorType, InterpolationType, OutputType, dim >**: Is a quite elaborate data collection class based on `AMRInterpolation<VectorType, dim>` geared toward statistics processing typical for turbulent simulations. Has run-time function parser.
[code/amroc/doc/html/weno/classWENOStatistics.html](#)

The interface otherwise follows the Clawpack integration closely:

- ▶ Generic main program **amroc/clawpack/mains/amr_main.C** is re-used.
- ▶ **Problem.h**: simulation-specific alteration of the C++ predefined classes specified in **WENOStdProblem.h**
[code/amroc/doc/html/weno/WENOStdProblem_8h.html](#)
- ▶ **WENOProblem.h**: Include required C++ class definitions definitions, all Fortran function names defined in **WENOStdFunctions.h**
[code/amroc/doc/html/weno/WENOProblem_8h_source.html](#) [code/amroc/doc/html/weno/WENOStdFunctions_8h.html](#)
- ▶ Interface objects from **amroc/amr/F77Interfaces** re-used and functions linked in Makefile.am as with Clawpack integrator

Governing equations

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0 \\ \frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot \left[\rho \mathbf{u}^t \mathbf{u} + \left(p + \frac{\mathbf{B} \cdot \mathbf{B}}{2} \right) \mathbf{I} - \mathbf{B}^t \mathbf{B} \right] &= \mathbf{0} \\ \frac{\partial \rho E}{\partial t} + \nabla \cdot \left[\left(\rho E + p + \frac{\mathbf{B} \cdot \mathbf{B}}{2} \right) \mathbf{u} - (\mathbf{u} \cdot \mathbf{B}) \mathbf{B} \right] &= 0 \\ \frac{\partial \mathbf{B}}{\partial t} + \nabla \cdot (\mathbf{u}^t \mathbf{B} - \mathbf{B}^t \mathbf{u}) &= \mathbf{0} \end{aligned}$$

with equation of state

$$p = (\gamma - 1) \left(\rho E - \rho \frac{\mathbf{u}^2}{2} - \frac{\mathbf{B}^2}{2} \right)$$

The ideal MDH model is still hyperbolic, yet by re-writing the induction equation, one finds that the magnetic field has to satisfy at all times t the elliptic constraint

$$\nabla \cdot \mathbf{B} = 0.$$

Generalized Lagrangian multipliers for divergence control

Hyperbolic-parabolic correction of 2d ideal MHD model [Dedner et al., 2002]:

$$\begin{aligned}
& \frac{\partial \rho}{\partial t} + \frac{\partial \rho u_x}{\partial x} + \frac{\partial \rho u_y}{\partial y} = 0 \\
& \frac{\partial(\rho u_x)}{\partial t} + \frac{\partial}{\partial x} \left[\rho u_x^2 + p \left(p + \frac{\mathbf{B} \cdot \mathbf{B}}{2} \right) - B_x^2 \right] + \frac{\partial}{\partial y} (\rho u_x u_y - B_x B_y) = 0 \\
& \frac{\partial(\rho u_y)}{\partial t} + \frac{\partial}{\partial x} (\rho u_x u_y - B_x B_y) + \frac{\partial}{\partial y} \left[\rho u_y^2 + p \left(p + \frac{\mathbf{B} \cdot \mathbf{B}}{2} \right) - B_y^2 \right] = 0 \\
& \frac{\partial(\rho u_z)}{\partial t} + \frac{\partial}{\partial x} (\rho u_z u_x - B_z B_x) + \frac{\partial}{\partial y} (\rho u_z u_y - B_z B_y) = 0 \\
& \frac{\partial \rho E}{\partial t} + \frac{\partial}{\partial x} \left[\left(\rho E + p + \frac{\mathbf{B} \cdot \mathbf{B}}{2} \right) \mathbf{u}_x - (\mathbf{u} \cdot \mathbf{B}) B_x \right] + \frac{\partial}{\partial y} \left[\left(\rho E + p + \frac{\mathbf{B} \cdot \mathbf{B}}{2} \right) \mathbf{u}_y - (\mathbf{u} \cdot \mathbf{B}) B_y \right] = 0 \\
& \frac{\partial B_x}{\partial t} + \frac{\partial \psi}{\partial x} + \frac{\partial}{\partial y} (u_y B_x - B_y u_x) = 0 \\
& \frac{\partial B_y}{\partial t} + \frac{\partial}{\partial x} (u_x B_y - B_x u_y) + \frac{\partial \psi}{\partial y} = 0 \\
& \frac{\partial B_z}{\partial t} + \frac{\partial}{\partial x} (u_x B_z - B_z u_x) + \frac{\partial}{\partial y} (u_y B_z - B_y u_z) = 0 \\
& \frac{\partial \psi}{\partial t} + c_h^2 \left(\frac{\partial B_x}{\partial x} + \frac{\partial B_y}{\partial y} \right) = -\frac{c_h^2}{c_n^2} \psi
\end{aligned}$$

Hyperbolic AMROC solver
High-resolution methods
OOOOO
Software design

Directory `amroc/mhd` contains the integrator that is in C++ throughout:

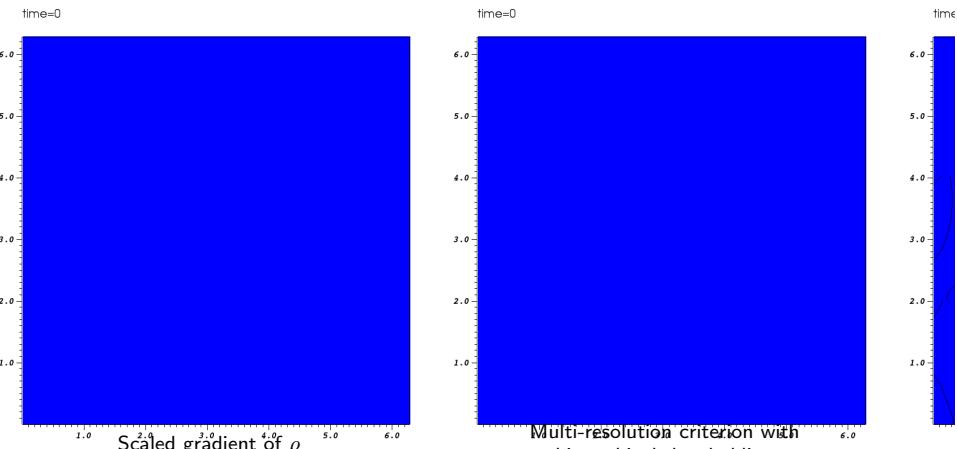
- ▶ **EGLM2D<DataType >:** GLM method in 2d plus standard initial and boundary conditions. Internal functions for flux evaluation, MUSCL reconstruction, etc. are member functions.
`code/amroc/doc/html/mhd/classEGLM2D.html`
 - ▶ Is derived from **SchemeBase<vector_type, dim >**, which is designed for the C++ interface classes in amroc/amr/Interfaces.
`code/amroc/doc/html/amr/classSchemeBase.html`
 - ▶ **amroc/amr/Interfaces:** Provides **SchemeIntegrator<SchemeType, dim >**, **SchemeInitialCondition<SchemeType, dim >**, **SchemeBoundaryCondition<SchemeType, dim >** and further interfaces that use classes derived from **SchemeBase<vector_type, dim >** as template parameter. This provides a single-class location for new schemes in C++.
`code/amroc/doc/html/amr/classSchemeIntegrator.html` `code/amroc/doc/html/amr/classSchemeInitialCondition.html`
 - ▶ **Problem.h:** Specific simulation is defined in Problem.h only. Predefined classes specified in **MHDSStdProblem.h** and **MHDProblem.h** similar but simpler as before.

Orszag-Tang vortex

- Adaptive solution on 50×50 grid with 4 additional levels refined by $r_l = 2$
 - Initial condition

$$\rho(x, y, 0) = \gamma^2, \quad u_x(x, y, 0) = -\sin(y), \quad u_y(x, y, 0) = \sin(x), \quad u_z(x, y, 0) = 0$$

$$\phi(x, y, 0) = \gamma, \quad B_x(x, y, 0) = -\sin(y), \quad B_y(x, y, 0) = 2\sin(x), \quad B_z(x, y, 0) = 0$$



http://code.amsrc.org/docs/html/applications_2cgml_23d_29rszgTangVortex_2src_2Bproblem_2h_source.html

Hyperbolic AMROC solvers	AMROC	Clawpack solver	WENO solver	MHD solver	Ref.
High-resolution methods	○○○○○	○○○○○○○○	○○○○○	○○○●○	○○
Software design					

Classes

Further hyperbolic solvers

- ▶ Is derived from **SchemeBase<vector_type, dim >**, which is designed for the C++ interface classes in amroc/amr/Interfaces.
[code/amroc/doc/html/amr/classSchemeBase.html](#)
- ▶ **amroc/amr/Interfaces:** Provides **SchemeIntegrator<SchemeType, dim >**, **SchemeInitialCondition<SchemeType, dim >**, **SchemeBoundaryCondition<SchemeType, dim >** and further interfaces that use classes derived from **SchemeBase<vector_type, dim >** as template parameter. This provides a single-class location for new schemes in C++
[code/amroc/doc/html/amr/classSchemeBase.html](#)
- ▶ **amroc/rim:** Riemann invariant manifold method. 2d implementation in F77 with straightforward integration into AMROC.
[code/amroc/doc/html/rim/files.html](#)
- ▶ **amroc/balans:** 2nd order accurate central difference scheme. 2d implementation in F77 and excellent template for Fortran scheme incorporation into AMROC.
[code/amroc/doc/html/balans/files.html](#)

References I

- [Colella and Woodward, 1984] Colella, P. and Woodward, P. (1984). The piecewise parabolic method (PPM) for gas-dynamical simulations. *J. Comput. Phys.*, 54:174–201.
- [Dedner et al., 2002] Dedner, A., Kemm, F., Kröner, D., Munz, C.-D., Schnitzer, T., and Wesenberg, M. (2002). Hyperbolic divergence cleaning for the MHD equations. *J. Comput. Phys.*, 175:645–673.
- [Deiterding et al., 2007] Deiterding, R., Cirak, F., Mauch, S. P., and Meiron, D. I. (2007). A virtual test facility for simulating detonation- and shock-induced deformation and fracture of thin flexible shells. *Int. J. Multiscale Computational Engineering*, 5(1):47–63.
- [Deiterding et al., 2006] Deiterding, R., Radovitzky, R., Mauch, S. P., Noels, L., Cummings, J. C., and Meiron, D. I. (2006). A virtual test facility for the efficient simulation of solid materials under high energy shock-wave loading. *Engineering with Computers*, 22(3-4):325–347.
- [Godlewski and Raviart, 1996] Godlewski, E. and Raviart, P.-A. (1996). *Numerical approximation of hyperbolic systems of conservation laws*. Springer Verlag, New York.

References III

- [LeVeque, 1997] LeVeque, R. J. (1997). Wave propagation algorithms for multidimensional hyperbolic systems. *J. Comput. Phys.*, 131(2):327–353.
- [Oran and Boris, 2001] Oran, E. S. and Boris, J. P. (2001). *Numerical simulation of reactive flow*. Cambridge Univ. Press, Cambridge, 2nd edition.
- [Pantano et al., 2007] Pantano, C., Deiterding, R., Hill, D. J., and Pullin, D. I. (2007). A low-numerical dissipation patch-based adaptive mesh refinement method for large-eddy simulation of compressible flows. *J. Comput. Phys.*, 221(1):63–87.
- [Shu, 97] Shu, C.-W. (97). Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. Technical Report CR-97-206253, NASA.
- [van Leer, 1979] van Leer, B. (1979). Towards the ultimate conservative difference scheme V. A second order sequel to Godunov's method. *J. Comput. Phys.*, 32:101–136.

References II

- [Gottlieb et al., 2001] Gottlieb, S., Shu, C.-W., and Tadmor, E. (2001). Strong stability-preserving high-order time discretization methods. *SIAM Review*, 43(1):89–112.
- [Harten, 1983] Harten, A. (1983). High resolution schemes for hyperbolic conservation laws. *J. Comput. Phys.*, 49:357–393.
- [Harten et al., 1976] Harten, A., Hyman, J. M., and Lax, P. D. (1976). On finite-difference approximations and entropy conditions for shocks. *Comm. Pure Appl. Math.*, 29:297–322.
- [Hirsch, 1988] Hirsch, C. (1988). *Numerical computation of internal and external flows*. John Wiley & Sons, Chichester.
- [Laney, 1998] Laney, C. B. (1998). *Computational gasdynamics*. Cambridge University Press, Cambridge.
- [Langseth and LeVeque, 2000] Langseth, J. and LeVeque, R. (2000). A wave propagation method for three dimensional conservation laws. *J. Comput. Phys.*, 165:126–166.

Lecture 4

Numerical methods for combustion research

Course *Block-structured Adaptive Finite Volume Methods in C++*

Ralf Deiterding

University of Southampton
Engineering and the Environment
Highfield Campus, Southampton SO17 1BJ, UK

E-mail: r.deiterding@soton.ac.uk

Outline

Complex geometry

- Boundary aligned meshes
- Cartesian techniques
- Implicit geometry representation
- Accuracy / verification
- Implementation

Combustion

- Equations
- Upwind schemes for combustion
- Tests with one-step chemistry
- Shock-induced combustion with real chemistry

Outline

Complex geometry

- Boundary aligned meshes
- Cartesian techniques
- Implicit geometry representation
- Accuracy / verification
- Implementation

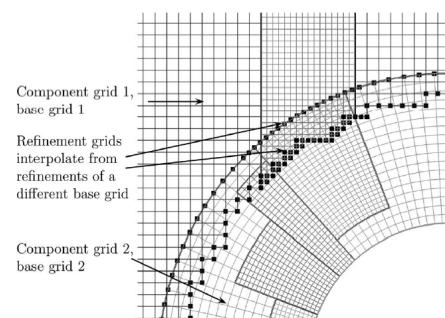
Combustion

- Equations
- Upwind schemes for combustion
- Tests with one-step chemistry
- Shock-induced combustion with real chemistry

SAMR on boundary aligned meshes

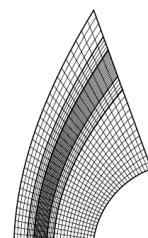
Analytic or stored geometric mapping of the coordinates
(graphic from [Yamaleev and Carpenter, 2002])

- Topology remains unchanged and thereby entire SAMR algorithm
- Patch solver and interpolation need to consider geometry transformation
- Handles boundary layers well



Overlapping adaptive meshes
[Henshaw and Schwendeman, 2003],
[Meakin, 1995]

- Idea is to use a non-Cartesian structured grids only near boundary
- Very suitable for moving objects with boundary layers
- Interpolation between meshes is usually non-conservative



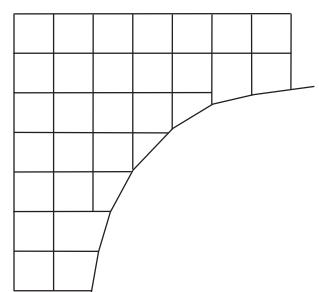
Cut-cell techniques

Accurate embedded boundary method

$$\begin{aligned} V_j^{n+1} \mathbf{Q}_j^{n+1} = & V_j^n \mathbf{Q}_j^n - \Delta t \left(A_{j+1/2}^{n+1/2} \mathbf{F}(\mathbf{Q}, j) \right. \\ & \left. - A_{j-1/2}^{n+1/2} \mathbf{F}(\mathbf{Q}, j-1) \right) \end{aligned}$$

Methods that represent the boundary sharply:

- Cut-cell approach constructs appropriate finite volumes
- Conservative by construction. Correct boundary flux
- Key question: How to avoid small-cell time step restriction in explicit methods?
 - Cell merging: [Quirk, 1994a]
- Usually explicit geometry representation used [Aftosmis, 1997], but can also be implicit, cf. [Nourgaliev et al., 2003], [Murman et al., 2003]



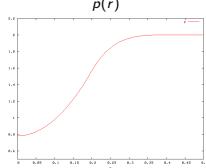
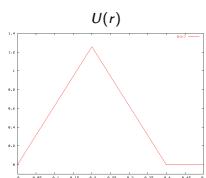
Accuracy test: stationary vortex

Construct non-trivial *radially symmetric* and *stationary* solution by balancing hydrodynamic pressure and centripetal force per volume element, i.e.

$$\frac{d}{dr} p(r) = \rho(r) \frac{U(r)^2}{r}$$

For $\rho_0 \equiv 1$ and the velocity field

$$U(r) = \alpha \cdot \begin{cases} 2r/R & \text{if } 0 < r < R/2, \\ 2(1 - r/R) & \text{if } R/2 \leq r \leq R, \\ 0 & \text{if } r > R, \end{cases}$$



one gets with boundary condition $p(R) = p_0 \equiv 2$ the pressure distribution

$$p(r) = p_0 + 2\rho_0\alpha^2 \cdot \begin{cases} r^2/R^2 + 1 - 2\log 2 & \text{if } 0 < r < R/2, \\ r^2/R^2 + 3 - 4r/R + 2\log(r/R) & \text{if } R/2 \leq r \leq R, \\ 0 & \text{if } r > R. \end{cases}$$

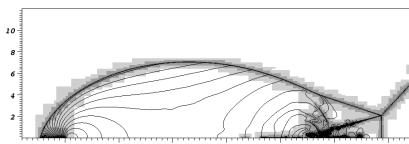
Entire solution for Euler equations reads

$$\rho(x_1, x_2, t) = \rho_0, u_1(x_1, x_2, t) = -U(r) \sin \phi, u_2(x_1, x_2, t) = U(r) \cos \phi, p(x_1, x_2, t) = p(r)$$

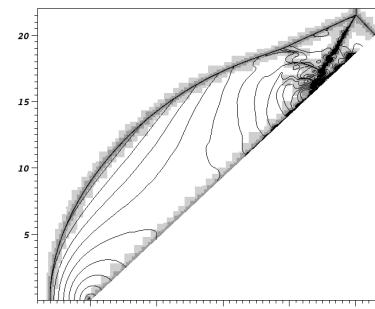
for all $t \geq 0$ with $r = \sqrt{(x_1 - x_{1,c})^2 + (x_2 - x_{2,c})^2}$ and $\phi = \arctan \frac{x_2 - x_{2,c}}{x_1 - x_{1,c}}$

Verification: shock reflection

- ▶ Reflection of a Mach 2.38 shock in nitrogen at 43° wedge
- ▶ 2nd order MUSCL scheme with Roe solver, 2nd order multidimensional wave propagation method



Cartesian base grid 360×160 cells on domain of $36 \text{ mm} \times 16 \text{ mm}$ with up to 3 refinement levels with $r_f = 2, 4, 4$ and $\Delta x_{1,2} = 3.125 \mu\text{m}$, 38 h CPU



GFM base grid 390×330 cells on domain of $26 \text{ mm} \times 22 \text{ mm}$ with up to 3 refinement levels with $r_f = 2, 4, 4$ and $\Delta x_{e,1,2} = 2.849 \mu\text{m}$, 200 h CPU

Stationary vortex: results

Compute one full rotation, Roe solver, embedded slip wall boundary conditions $x_{1,c} = 0.5, x_{2,c} = 0.5, R = 0.4, t_{end} = 1, \Delta h = \Delta x_1 = \Delta x_2 = 1/N, \alpha = R\pi$

No embedded boundary

N	Wave Propagation		Godunov Splitting	
	Error	Order	Error	Order
20	0.0111235		0.0182218	
40	0.0037996	1.55	0.0090662	1.01
80	0.0013388	1.50	0.0046392	0.97
160	0.0005005	1.42	0.0023142	1.00

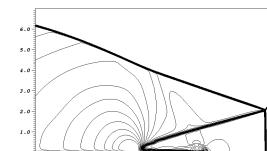
Marginal shear flow along embedded boundary, $\alpha = R\pi, R_G = R, U_W = 0$

N	Wave Propagation		Godunov Splitting	
	Error	Order	Error	Mass loss
20	0.0120056		0.0079236	0.0144203
40	0.0035074	1.78	0.0011898	0.0073070
80	0.0014193	1.31	0.0001588	0.0038401
160	0.0005032	1.50	5.046e-05	0.0018988

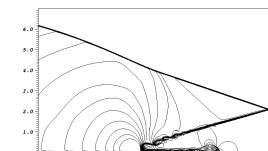
Major shear flow along embedded boundary, $\alpha = R\pi, R_G = R/2, U_W = 0$

N	Wave Propagation		Godunov Splitting	
	Error	Order	Error	Mass loss
20	0.0423925		0.0423925	0.0271446
40	0.0358735	0.24	0.0358735	0.0242260
80	0.0212340	0.76	0.0212340	0.0128638
160	0.0121089	0.81	0.0121089	0.0070906

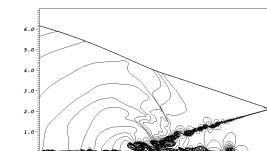
Shock reflection: SAMR solution for Euler equations



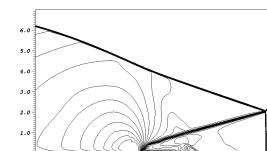
$\Delta x = 25 \text{ mm}$



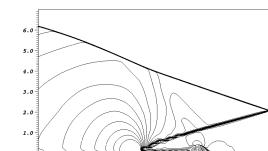
$\Delta x = 12.5 \text{ mm}$



$\Delta x = 3.125 \text{ mm}$



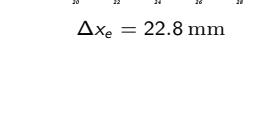
$\Delta x_e = 22.8 \text{ mm}$



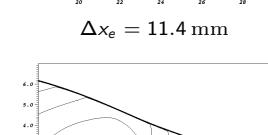
$\Delta x_e = 11.4 \text{ mm}$



$\Delta x_e = 2.849 \text{ mm}$



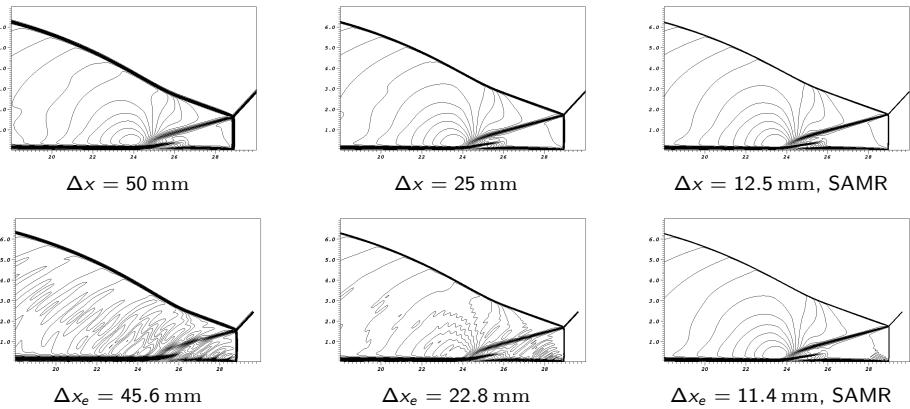
2nd order MUSCL scheme with Van Leer FVS, dimensional splitting



$\Delta x = 3.125 \text{ mm}$

Shock reflection: solution for Navier-Stokes equations

- ▶ No-slip boundary conditions enforced
- ▶ Conservative 2nd order centered differences to approximate stress tensor and heat flow



Outline

Complex geometry

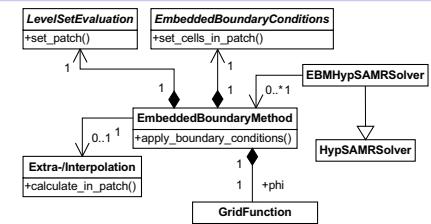
Boundary aligned meshes
Cartesian techniques
Implicit geometry representation
Accuracy / verification
Implementation

Combustion

Equations
Upwind schemes for combustion
Tests with one-step chemistry
Shock-induced combustion with real chemistry

Embedded boundary method

- ▶ Multiple independent EmbeddedBoundaryMethod objects possible
- ▶ Specialization of GFM boundary conditions
- ▶ The generic embedded boundary method is implemented in **GhostFluidMethod<VectorType, dim >** and has a **GFMLevelSet<DataType, dim >** and **GFMBoundary<VectorType, dim >** object.
- ▶ code/amroc/doc/html/amr/classGhostFluidMethod.html code/amroc/doc/html/amr/classGFMLevelSet.html
code/amroc/doc/html/amr/classGFMBoundary_3_01VectorType_00_012_01_4.html
- ▶ Multiple **GhostFluidMethod<VectorType, dim >** can be registered with **AMRGFMSolver<VectorType, FixupType, FlagType, dim >** and are called as part of the boundary condition setting routine.
- ▶ code/amroc/doc/html/amr/classAMRGFMSolver.html
- ▶ Interface classes to **GFMLevelSet<DataType, dim >** and **GFMBoundary<VectorType, dim >** are available amroc/amr/F77Interfaces and in amroc/amr/Interfaces
- ▶ code/amroc/doc/html/amr/classF77GFMBoundary.html code/amroc/doc/html/amr/classSchemeGFMBoundary.html make the approach available to all current solvers
- ▶ For instance code/amroc/doc/html/clp/ClpStdGFMProblem_8h.html or code/amroc/doc/html/weno/WENOStdGFMProblem_8h.html in Problem.h uses AMRGFMSolver<>



Governing equations for premixed combustion

Euler equations with reaction terms

$$\begin{aligned} \frac{\partial \rho_i}{\partial t} + \frac{\partial}{\partial x_n} (\rho_i u_n) &= \dot{\omega}_i, \quad i = 1, \dots, K \\ \frac{\partial}{\partial t} (\rho u_k) + \frac{\partial}{\partial x_n} (\rho u_k u_n + \delta_{kn} p) &= 0, \quad k = 1, \dots, d \\ \frac{\partial}{\partial t} (\rho E) + \frac{\partial}{\partial x_n} (u_n (\rho E + p)) &= 0 \end{aligned}$$

Ideal gas law and Dalton's law for gas-mixtures

$$p(\rho_1, \dots, \rho_K, T) = \sum_{i=1}^K p_i = \sum_{i=1}^K \rho_i \frac{\mathcal{R}}{W_i} T = \rho \frac{\mathcal{R}}{W} T \quad \text{with} \quad \sum_{i=1}^K \rho_i = \rho, Y_i = \frac{\rho_i}{\rho}$$

Caloric equation

$$h(Y_1, \dots, Y_K, T) = \sum_{i=1}^K Y_i h_i(T) \quad \text{with} \quad h_i(T) = h_i^0 + \int_0^T c_{pi}(s) ds$$

Computation of $T = T(\rho_1, \dots, \rho_K, e)$ from implicit equation

$$\sum_{i=1}^K \rho_i h_i(T) - \mathcal{R} T \sum_{i=1}^K \frac{\rho_i}{W_i} - \rho e = 0$$

for thermally perfect gases with $\gamma_i(T) = c_{pi}(T)/c_{vi}(T)$

Roe solver

Roe averages $\hat{\rho}$, \hat{u} , \hat{H} , \hat{W}_i , \hat{T} , \hat{h}_i , \hat{e}_i , \hat{Y}_i

Define $\hat{c}_{pi} = \frac{1}{T_r - T_l} \int_{T_l}^{T_r} c_{pi}(\tau) d\tau$, $\hat{c}_{vi} = \frac{1}{T_r - T_l} \int_{T_l}^{T_r} c_{vi}(\tau) d\tau$

$$\Delta f := f(q_R) - f(q_L) = \sum_{m=1}^M a_m \lambda_m(\hat{q}) r_m(\hat{q}) \quad \text{with} \quad \Delta q := q_R - q_L = \sum_{m=1}^M a_m r_m(\hat{q}).$$

With matrix of right eigenvectors

$$R_1(q) = \begin{bmatrix} Y_1 & 1 & 0 & \dots & 0 & 0 & 0 & Y_1 \\ 0 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \vdots & & & & \vdots & \vdots & \vdots & \vdots \\ Y_K & 0 & \dots & 0 & 1 & 0 & 0 & Y_K \\ u_1 - a & u_1 & \dots & u_1 & 0 & 0 & u_1 + a & \\ u_2 & u_2 & \dots & u_2 & 1 & 0 & u_2 & \\ u_3 & u_3 & \dots & u_3 & 0 & 1 & u_3 & \\ H - u_1 a & u^2 - \frac{\phi_1}{\bar{\gamma}} & \dots & u^2 - \frac{\phi_K}{\bar{\gamma}} & u_2 & u_3 & H + u_1 a & \end{bmatrix}$$

and evaluating $R^{-1}(\hat{q}) \Delta q$ one gets the characteristic wave strengths eventually as

$$a_1, a_{K+d+1} = \frac{\Delta p \mp \hat{\rho} \hat{a} \Delta u_1}{2 \hat{a}^2}, \quad a_{1+i} = \Delta \rho_i - \hat{Y}_i \frac{\Delta p}{\hat{a}^2}, \quad a_{K+n} = \hat{\rho} \Delta u_n.$$

[code/amroc/doc/html/clp/rp1eurhok.f_source.html](#)

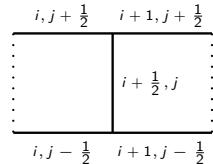
Roe solver - entropy and carbuncle fix

Entropy corrections [Harten, 1983]
[Harten and Hyman, 1983]

$$1. |\tilde{s}_t| = \begin{cases} |s_t| & \text{if } |s_t| \geq 2\eta \\ \frac{|s_t|^2}{4\eta} + \eta & \text{otherwise} \end{cases}$$

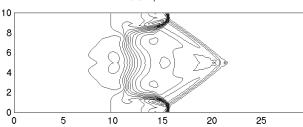
- $\eta = \frac{1}{2} \max_t \{|s_t(q_R) - s_t(q_L)|\}$
2. Replace $|s_t|$ by $|\tilde{s}_t|$ only if $s_t(q_L) < 0 < s_t(q_R)$

2D modification of entropy correction (here named EF 3)
[Sanders et al., 1998]:

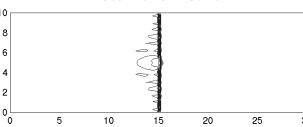


$$\tilde{\eta}_{i+1/2,j} = \max \{ \eta_{i+1/2,j}, \eta_{i,j-1/2}, \eta_{i,j+1/2}, \eta_{i+1,j-1/2}, \eta_{i+1,j+1/2} \}$$

Roe + EF 1.



Exact Riemann solver



Carbuncle phenomenon

- [Quirk, 1994b]
- Test from [Deiterding, 2003]

[code/amroc/doc/html/apps/clawpack_2applications_2euler_znd_22d_2Carbuncle_2src_2Problem_8h_source.html](#)

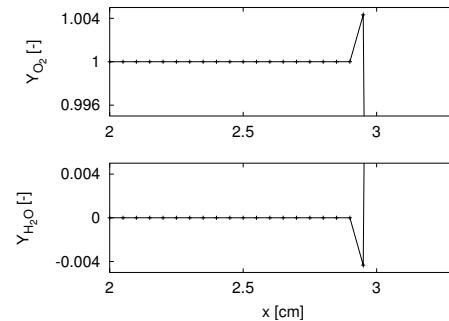
Roe solver - fixes

Mass fraction positivity: Calculate numerical fluxes of partial densities by [Larrouy, 1991]

$$F_i^*(q_L, q_R) = F_\rho(q_L, q_R) \cdot \begin{cases} Y_{i,L}, & F_\rho(q_L, q_R) > 0, \\ Y_{i,R}, & F_\rho(q_L, q_R) < 0. \end{cases}$$

to ensure positivity of Y_i .

Example: Mass fraction for a typical Riemann problem after 1 time step with the Roe solver



Further: Switch from Roe to HLL scheme near vacuum state to avoid unphysical values.

Riemann solver for combustion

(S1) Calculate standard Roe-averages $\hat{\rho}$, \hat{u}_n , \hat{H} , \hat{Y}_i , \hat{T} .

(S2) Compute $\hat{\gamma} := \hat{c}_\rho / \hat{c}_v$ with $\hat{c}_{\{p,v\}i} = \frac{1}{T_R - T_L} \int_{T_L}^{T_R} c_{\{p,v\}i}(\tau) d\tau$.

(S3) Calculate $\hat{\phi}_i := (\hat{\gamma} - 1) \left(\frac{\hat{u}_i^2}{2} - \hat{h}_i \right) + \hat{\gamma} R_i \hat{T}$ with standard Roe-averages \hat{e}_i or \hat{h}_i .

(S4) Calculate $\hat{a} := \left(\sum_{i=1}^K \hat{Y}_i \hat{\phi}_i - (\hat{\gamma} - 1) \hat{u}^2 + (\hat{\gamma} - 1) \hat{H} \right)^{1/2}$.

(S5) Use $\Delta q = q_R - q_L$ and Δp to compute the wave strengths a_m .

(S6) Calculate $\mathcal{W}_1 = a_1 \hat{r}_1$, $\mathcal{W}_2 = \sum_{\iota=2}^{K+d} a_\iota \hat{r}_\iota$, $\mathcal{W}_3 = a_{K+d+1} \hat{r}_{K+d+1}$.

(S7) Evaluate $s_1 = \hat{u}_1 - \hat{a}$, $s_2 = \hat{u}_1$, $s_3 = \hat{u}_1 + \hat{a}$.

(S8) Evaluate $\rho_{L/R}^*$, $u_{L/R}^*$, $e_{L/R}^*$, $a_{L/R}^*$ from $q_L^* = q_L + \mathcal{W}_1$ and $q_R^* = q_R - \mathcal{W}_3$.

(S9) If $\rho_{L/R}^* \leq 0$ or $e_{L/R}^* \leq 0$ use $F_{HLL}(q_L, q_R)$ and go to (S12).

(S10) Entropy correction: Evaluate $|\tilde{s}_t|$.

$$F_{Roe}(q_L, q_R) = \frac{1}{2} (f(q_L) + f(q_R) - \sum_{\iota=1}^3 |\tilde{s}_\iota| \mathcal{W}_\iota)$$

(S11) Positivity correction: Replace F_i by

$$F_i^* = F_\rho \cdot \begin{cases} Y_i^l, & F_\rho \geq 0, \\ Y_i^r, & F_\rho < 0. \end{cases}$$

(S12) Evaluate maximal signal speed by $S = \max(|s_1|, |s_3|)$.

[code/amroc/doc/html/clp/rp1eurhokefix.f_source.html](#)

[code/amroc/doc/html/clp/rp1eurhokefixg.f_source.html](#)

ZND Detonation Model with Simplified Chemistry

Assume a stationary 1D detonation with irreversible reaction



with energy release $q_0 = -\Delta h^0$ and $k^f(T) = K \exp(-E_A/\mathcal{R}T)$.

Simplified Kinetics $\dot{W}_A \dot{\omega}_A = -K \rho_A \exp(-E_A/\mathcal{R}T)$, $\dot{W}_B \dot{\omega}_B = -W_A \dot{\omega}_A$

With $\gamma_A = \gamma_B$ the equation of state is $p(\rho_A, \rho_B, e) = (\gamma - 1)(\rho e - \rho_A q_0)$

Integration of the stationary 1D Euler equations

$$\partial_{x'}(\rho u') = 0, \quad \partial_{x'}(\rho u'^2 + p) = 0, \quad \partial_{x'}[u'(\rho E + p)] = 0$$

$$\partial_{x'} \lambda = \frac{K \rho (1 - \lambda) \exp(-E_A^* \rho / p)}{\rho u'}$$

gives for λ , the mass fraction of the product B ,

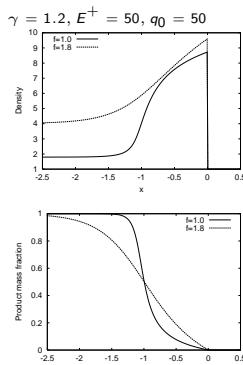
$$\frac{d\lambda}{dx'} = K(1 - \lambda) \exp\left(\frac{-E_A^* \rho(\lambda)/p(\lambda)}{u'(\lambda) + D}\right) = f(\lambda),$$

i.e. $\int_0^{x'} f(\lambda(\xi)) d\xi = \lambda(x')$.

D_{CJ} is the minimal detonation velocity. The overdrive-parameter $f = (D/D_{CJ})^2 \geq 1$ determines stability.

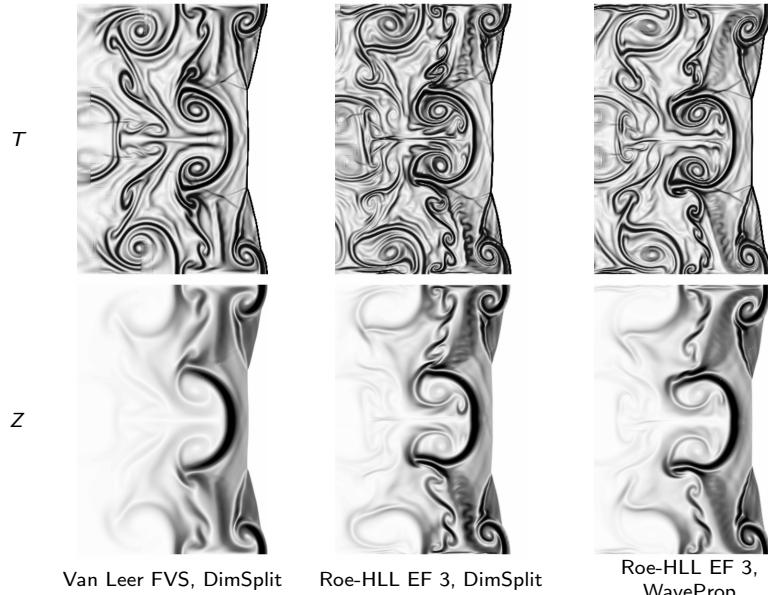
Normalization:

$$L_{1/2} = \int_0^{\frac{1}{2}} \frac{d\lambda}{f(\lambda)}$$



Comparison of FV Schemes: MUSCL, Van Albada-limiter

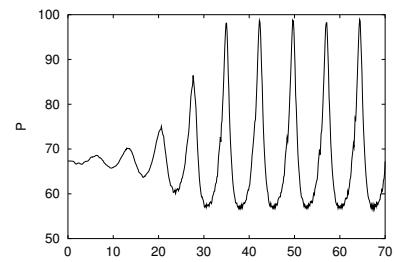
$$\gamma = 1.2, E = 50, q_0 = 50, f = 3.0, 40 \text{ Pts}/L_{1/2}$$



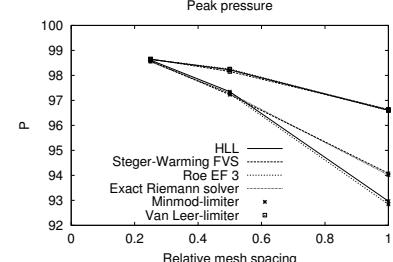
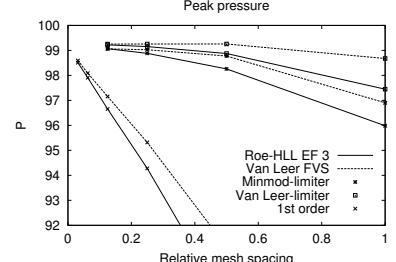
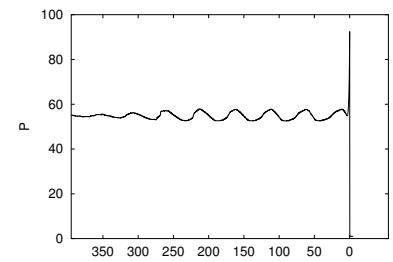
Unstable ZND Detonation

$$\gamma = 1.2, E^+ = 50, q_0 = 50, f = 1.6, CFL = 0.9$$

quasi-stationary



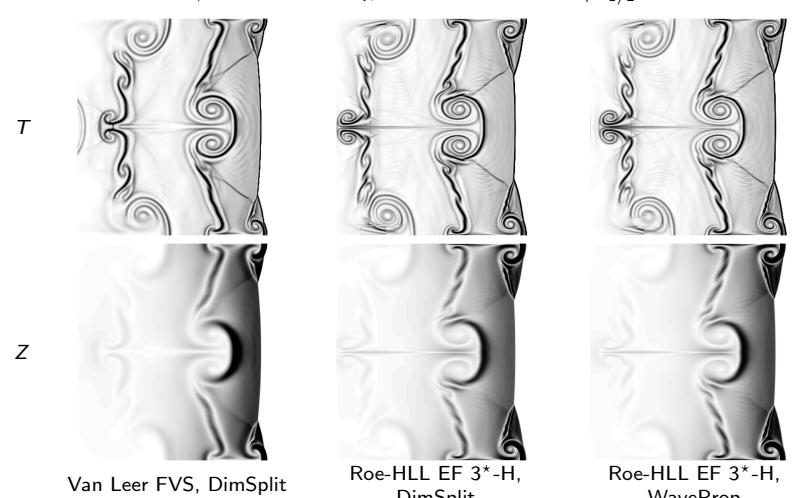
moving



code/amroc/doc/html/apps/clawpack_2applications_2euler_znd_21d_2StatDet_2src_2Problem_8h_source.html

Comparison of FV Schemes - II

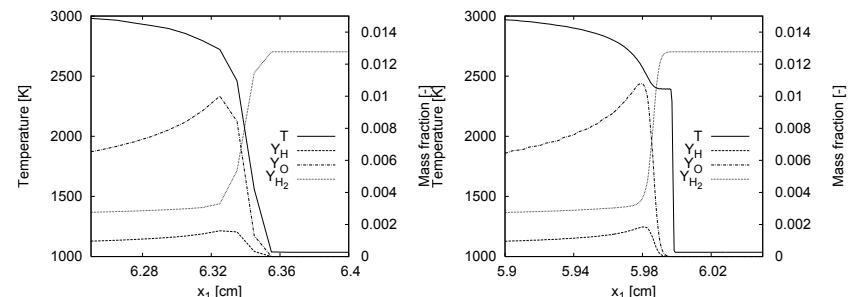
$$\gamma = 1.2, E = 10, q_0 = 50, f = 1.2, 40 \text{ Pts}/L_{1/2}$$



code/amroc/doc/html/apps/clawpack_2applications_2euler_znd_22d_2StatDet_2src_2Problem_8h_source.html

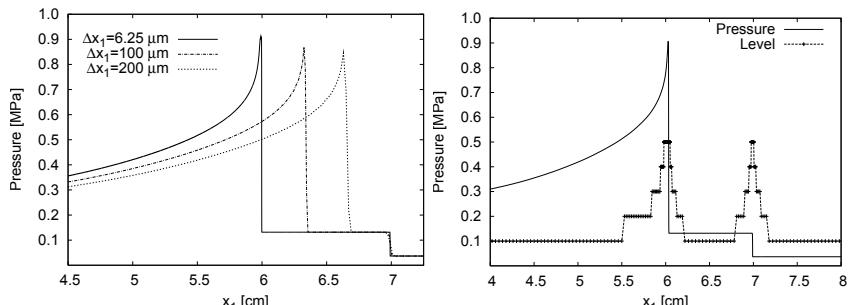
Detonations - motivation for SAMR

- Extremely high spatial resolution in reaction zone necessary.
- Minimal spatial resolution: $7 - 8 \text{ Pts}/l_{ig} \rightarrow \Delta x_1 \approx 0.2 - 0.175 \text{ mm}$
- Uniform grids for typical geometries: $> 10^7 \text{ Pts}$ in 2D, $> 10^9 \text{ Pts}$ in 3D \rightarrow Self-adaptive finite volume method (AMR)



Detonation ignition in a shock tube

- Shock-induced detonation ignition of $\text{H}_2 : \text{O}_2 : \text{Ar}$ mixture at molar ratios 2:1:7 in closed 1d shock tube
- Insufficient resolution leads to inaccurate results
- Reflected shock is captured correctly by FV scheme, detonation is resolution dependent
- Fine mesh necessary in the induction zone at the head of the detonation



Left: Comparison of pressure distribution $t = 170 \mu\text{s}$ after shock reflection. Right: Domains of refinement levels

Non-equilibrium mechanism for hydrogen-oxygen combustion

		A [cm, mol, s]	β	E_{act} [cal mol $^{-1}$]
1.	$\text{H} + \text{O}_2$	\rightarrow $\text{O} + \text{OH}$	1.86×10^{14}	0.00 16790.
2.	$\text{O} + \text{OH}$	\rightarrow $\text{H} + \text{O}_2$	1.48×10^{13}	0.00 680.
3.	$\text{H}_2 + \text{O}$	\rightarrow $\text{H} + \text{OH}$	1.82×10^{10}	1.00 8900.
4.	$\text{H} + \text{OH}$	\rightarrow $\text{H}_2 + \text{O}$	8.32×10^{99}	1.00 6950.
5.	$\text{H}_2\text{O} + \text{O}$	\rightarrow $\text{OH} + \text{HO}$	3.39×10^{13}	0.00 18350.
6.	$\text{OH} + \text{OH}$	\rightarrow $\text{H}_2\text{O} + \text{O}$	3.16×10^{12}	0.00 1100.
7.	$\text{H}_2\text{O} + \text{H}$	\rightarrow $\text{H}_2 + \text{OH}$	9.55×10^{13}	0.00 20300.
8.	$\text{H}_2 + \text{OH}$	\rightarrow $\text{H}_2\text{O} + \text{H}$	2.19×10^{13}	0.00 5150.
9.	$\text{H}_2\text{O}_2 + \text{OH}$	\rightarrow $\text{H}_2\text{O} + \text{HO}_2$	1.00×10^{13}	0.00 1800.
10.	$\text{H}_2\text{O} + \text{HO}_2$	\rightarrow $\text{H}_2\text{O}_2 + \text{OH}$	2.82×10^{13}	0.00 32790.
...
30.	$\text{OH} + \text{M}$	\rightarrow $\text{O} + \text{H} + \text{M}$	7.94×10^{19}	-1.00 103720.
31.	$\text{O}_2 + \text{M}$	\rightarrow $\text{O} + \text{O} + \text{M}$	5.13×10^{15}	0.00 115000.
32.	$\text{O} + \text{O} + \text{M}$	\rightarrow $\text{O}_2 + \text{M}$	4.68×10^{15}	-0.28 0.
33.	$\text{H}_2 + \text{M}$	\rightarrow $\text{H} + \text{H} + \text{M}$	2.19×10^{14}	0.00 96000.
34.	$\text{H} + \text{H} + \text{M}$	\rightarrow $\text{H}_2 + \text{M}$	3.02×10^{15}	0.00 0.

Third body efficiencies: $f(\text{O}_2) = 0.40$, $f(\text{H}_2\text{O}) = 6.50$

[Westbrook, 1982]

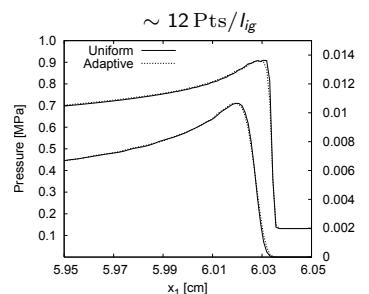
Detonation ignition in 1d - adaptive vs. uniform

Uniformly refined vs. dynamic adaptive simulations (Intel Xeon 3.4 GHz CPU)

$\Delta x_1 [\mu\text{m}]$	Uniform			Adaptive			
	Cells	$t_m [\mu\text{s}]$	Time [s]	l_{max}	r_l	$t_m [\mu\text{s}]$	Time [s]
400	300	166.1	31				
200	600	172.6	90	2	2	172.6	99
100	1200	175.5	277	3	2,2	175.8	167
50	2400	176.9	858	4	2,2,2	177.3	287
25	4800	177.8	2713	4	2,2,4	177.9	393
12.5	9600	178.3	9472	5	2,2,2,4	178.3	696
6.25	19200	178.6	35712	5	2,2,4,4	178.6	1370

Refinement criteria:

Y_i	$S_{Y_i} \cdot 10^{-4}$	$\eta_{Y_i}^r \cdot 10^{-3}$
O_2	10.0	2.0
H_2O	7.8	8.0
H	0.16	5.0
O	1.0	5.0
OH	1.8	5.0
H_2	1.3	2.0

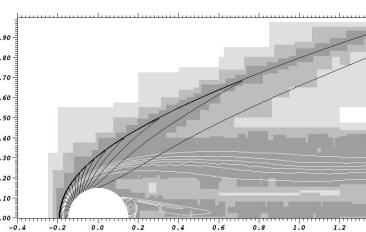
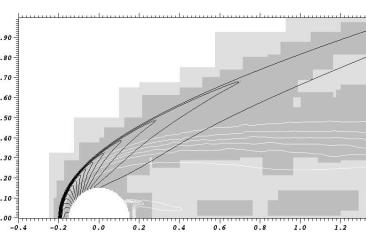


$$\epsilon_\rho = 0.07 \text{ kg m}^{-3}, \epsilon_p = 50 \text{ kPa}$$

code/amroc/doc/html/apps/clawpack_2applications_2euler_chem_2id_2ReflecDet_2src_2Problem_8h_source.html

Shock-induced combustion around a sphere

- Spherical projectile of radius 1.5 mm travels with constant velocity $v_I = 2170.6 \text{ m/s}$ through $\text{H}_2 : \text{O}_2 : \text{Ar}$ mixture (molar ratios 2:1:7) at 6.67 kPa and $T = 298 \text{ K}$
- Cylindrical symmetric simulation on AMR base mesh of 70×40 cells
- Comparison of 3-level computation with refinement factors 2,2 ($\sim 5 \text{ Pts}/l_{ig}$) and a 4-level computation with refinement factors 2,2,4 ($\sim 19 \text{ Pts}/l_{ig}$) at $t = 350 \mu\text{s}$
- Higher resolved computation captures combustion zone visibly better and at slightly different position (see below)



Iso-contours of p (black) and Y_{H_2} (white) on refinement domains for 3-level (left) and 4-level computation (right)

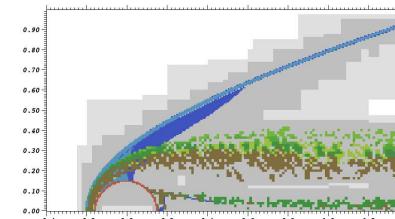
code/amroc/doc/html/apps/clawpack_2applications_2euler__chem_22d_2Sphere_2src_2Problem_8h_source.html

References I

- [Aftosmis, 1997] Aftosmis, M. J. (1997). Solution adaptive Cartesian grid methods for aerodynamic flows with complex geometries. Technical Report Lecture Series 1997-2, von Karman Institute for Fluid Dynamics.
- [Berger and Helzel, 2002] Berger, M. J. and Helzel, C. (2002). Grid aligned h-box methods for conservation laws in complex geometries. In *Proc. 3rd Int. Symp. Finite Volumes for Complex Applications*, Porquerolles.
- [Deiterding, 2003] Deiterding, R. (2003). *Parallel adaptive simulation of multi-dimensional detonation structures*. PhD thesis, Brandenburgische Technische Universität Cottbus.
- [Deiterding et al., 2006] Deiterding, R., Radovitzky, R., Mauch, S. P., Noels, L., Cummings, J. C., and Meiron, D. I. (2006). A virtual test facility for the efficient simulation of solid materials under high energy shock-wave loading. *Engineering with Computers*, 22(3-4):325–347.
- [Grossmann and Cinella, 1990] Grossmann, B. and Cinella, P. (1990). Flux-split algorithms for flows with non-equilibrium chemistry and vibrational relaxation. *J. Comput. Phys.*, 88:131–168.

Combustion around a sphere - adaptation

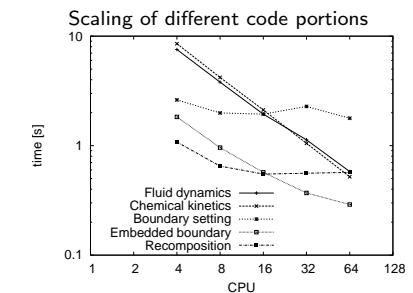
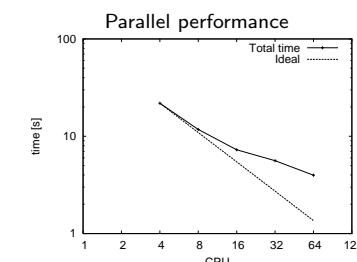
Refinement indicators on $l = 2$ at $t = 350 \mu\text{s}$.
Blue: ϵ_p , light blue: ϵ_p , green shades: $\eta_{Y_i}^r$, red: embedded boundary



Refinement criteria:

Y_i	$S_{Y_i} \cdot 10^{-4}$	$\eta_{Y_i}^r \cdot 10^{-4}$
O_2	10.0	4.0
H_2O	5.8	3.0
H	0.2	10.0
O	1.4	10.0
OH	2.3	10.0
H_2	1.3	4.0

$\epsilon_p = 0.02 \text{ kg m}^{-3}$, $\epsilon_p = 16 \text{ kPa}$



References II

- [Harten, 1983] Harten, A. (1983). High resolution schemes for hyperbolic conservation laws. *J. Comput. Phys.*, 49:357–393.
- [Harten and Hyman, 1983] Harten, A. and Hyman, J. M. (1983). Self-adjusting grid methods for one-dimensional hyperbolic conservation laws. *J. Comput. Phys.*, 50:235–269.
- [Henshaw and Schwendeman, 2003] Henshaw, W. D. and Schwendeman, D. W. (2003). An adaptive numerical scheme for high-speed reactive flow on overlapping grids. *J. Comput. Phys.*, 191:420–447.
- [Larrouy, 1991] Larrouy, B. (1991). How to preserve the mass fractions positivity when computing compressible multi-component flows. *J. Comput. Phys.*, 95:59–84.
- [Larrouy and Fezoui, 1989] Larrouy, B. and Fezoui, L. (1989). On the equations of multi-component perfect or real gas inviscid flow. In Carasso et al., editor, *Proc. of Second Int. Conf. on Nonlinear Hyperbolic Equations - Theory, Numerical Methods, and Applications, Aachen 1988*, Lecture Notes in Mathematics 1402, pages 69–98. Springer-Verlag Berlin.

References III

- [Liu and Vinokur, 1989] Liu, Y. and Vinokur, M. (1989). Nonequilibrium flow computations. I. An analysis of numerical formulations of conservation laws. *J. Comput. Phys.*, 83:373–397.
- [Mauch, 2003] Mauch, S. P. (2003). *Efficient Algorithms for Solving Static Hamilton-Jacobi Equations*. PhD thesis, California Institute of Technology.
- [Meakin, 1995] Meakin, R. L. (1995). An efficient means of adaptive refinement within systems of overset grids. In *12th AIAA Computational Fluid Dynamics Conference, San Diego*, AIAA-95-1722-CP.
- [Mittal and Iaccarino, 2005] Mittal, R. and Iaccarino, G. (2005). Immersed boundary methods. *Annu. Rev. Fluid Mech.*, 37:239–261.
- [Murman et al., 2003] Murman, S. M., Aftosmis, M. J., and Berger, M. J. (2003). Implicit approaches for moving boundaries in a 3-d Cartesian method. In *41st AIAA Aerospace Science Meeting*, AIAA 2003-1119.
- [Nourgaliev et al., 2003] Nourgaliev, R. R., Dinh, T. N., and Theofanous, T. G. (2003). On capturing of interfaces in multimaterial compressible flows using a level-set-based Cartesian grid method. Technical Report 05/03-1, Center for Risk Studies and Safety, UC Santa Barbara.

References IV

- [Pember et al., 1999] Pember, R. B., Bell, J. B., Colella, P., Crutchfield, W. Y., and Welcome, M. L. (1999). An adaptive Cartesian grid method for unsteady compressible flows in irregular regions. *J. Comput. Phys.*, 120:287–304.
- [Quirk, 1994a] Quirk, J. J. (1994a). An alternative to unstructured grids for computing gas dynamics flows around arbitrarily complex two-dimensional bodies. *Computers Fluids*, 23:125–142.
- [Quirk, 1994b] Quirk, J. J. (1994b). A contribution to the great Riemann solver debate. *Int. J. Numer. Meth. Fluids*, 18:555–574.
- [Roma et al., 1999] Roma, A. M., Perskin, C. S., and Berger, M. J. (1999). An adaptive version of the immersed boundary method. *J. Comput. Phys.*, 153:509–534.
- [Sanders et al., 1998] Sanders, R., Morano, E., and Druguet, M.-C. (1998). Multidimensional dissipation for upwind schemes: Stability and applications to gas dynamics. *J. Comput. Phys.*, 145:511–537.
- [Sethian, 1999] Sethian, J. A. (1999). *Level set methods and fast marching methods*. Cambridge University Press, Cambridge, New York.

References V

- [Shuen et al., 1990] Shuen, J.-S., Liou, M.-S., and van Leer, B. (1990). Inviscid flux-splitting algorithms for real gases with non-equilibrium chemistry. *J. Comput. Phys.*, 90:371–395.
- [Tseng and Ferziger, 2003] Tseng, Y.-H. and Ferziger, J. H. (2003). A ghost-cell immersed boundary method for flow in complex geometry. *J. Comput. Phys.*, 192:593–623.
- [Westbrook, 1982] Westbrook, C. K. (1982). Chemical kinetics of hydrocarbon oxidation in gaseous detonations. *Combust. Flame*, 46:191–210.
- [Yamaleev and Carpenter, 2002] Yamaleev, N. K. and Carpenter, M. H. (2002). On accuracy of adaptive grid methods for captured shocks. *J. Comput. Phys.*, 181:280–316.

Lecture 5

Detonation simulation

Course *Block-structured Adaptive Finite Volume Methods in C++*

Ralf Deiterding

University of Southampton
Engineering and the Environment
Highfield Campus, Southampton SO17 1BJ, UK

E-mail: r.deiterding@soton.ac.uk

Outline

Detonation simulation

Detonation structures

Combustion with viscous terms

Combustion induced by projectiles

Finite volume scheme

Higher order schemes

Hybrid methods

Planar ZND Structure

Steady situation under Galilean transformation:

$$\frac{\partial}{\partial x'}(\rho u') = 0$$

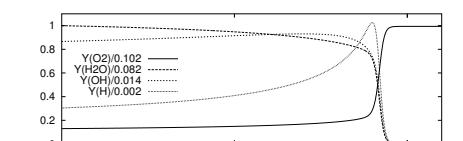
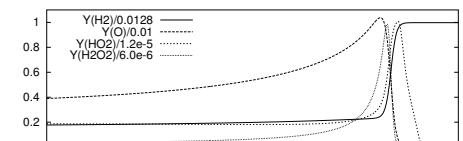
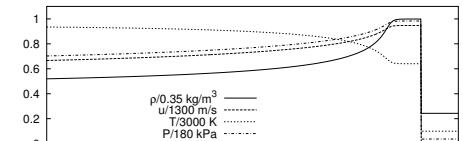
$$\frac{\partial}{\partial x'}(\rho u'^2 + p) = 0$$

$$\frac{\partial}{\partial x'}(u' \rho H) = 0$$

$$\frac{\partial Y_i}{\partial x'} = \frac{W_i \dot{\omega}_i \left(\rho \frac{Y_1}{W_1}, \dots, \rho \frac{Y_K}{W_K}, T \right)}{\rho u'}$$

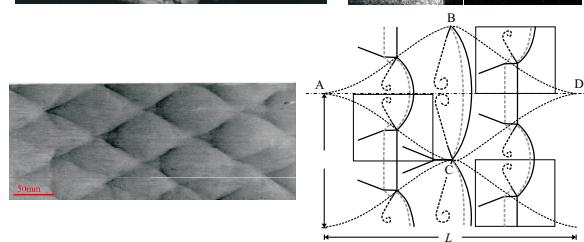
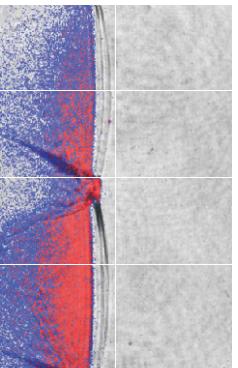
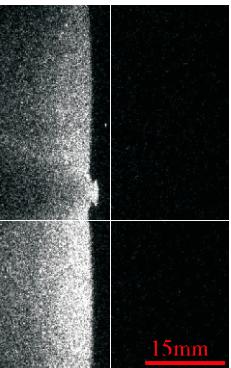
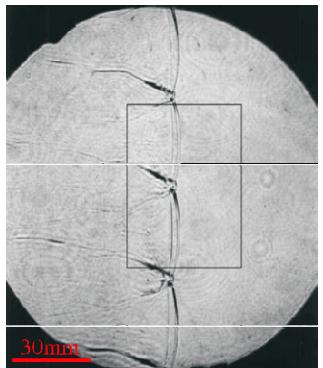
CJ-detonation of $\text{H}_2 : \text{O}_2 : \text{Ar}$ with molar ratios 2 : 1 : 7 at $T_0 = 298 \text{ K}$ and $p_0 = 6.67 \text{ kPa}$, $d_{\text{CJ}} \approx 1627 \text{ m/s}$.

$t_{ig} \approx 3.55 \mu\text{s}$, $u'_{vn} \approx 395.5 \text{ m/s}$, $l_{ig} \approx 0.14 \text{ cm}$.



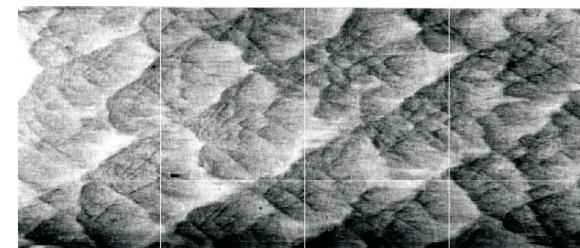
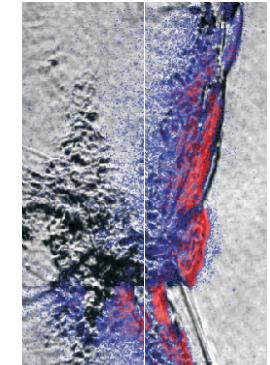
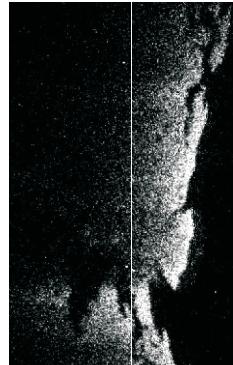
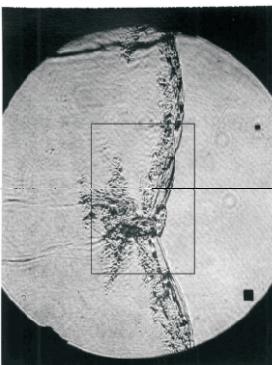
Cf. code/amroc/doc/html/apps/clawpack_2applications_2euler_chem_21d_2ModelDetonation_2src_2Problem_8h_source.html

Detonation cell structure in 2D - Regular instability



E: Reflected shock. F: Slip line. G: Diffusive extension of slip line.

Transverse detonation structure - irregular instability

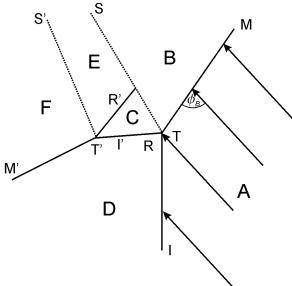


Shock polar analysis of triple points in detonations

- Neglect reaction, but consider $c_{pi}(T)$
- Data extracted point-wise from simulation
- Primary triple point T travels exactly at tip of Mach stem \rightarrow use oblique shock relations between A and B

$$\begin{aligned}\rho_A u_A \sin(\phi_B) &= \rho_B u_B \sin(\phi_B - \theta_B), \\ \rho_A + \rho_A u_A^2 \sin^2(\phi_B) &= \rho_B + \rho_B u_B^2 \sin^2(\phi_B - \theta_B)\end{aligned}$$

to evaluate inflow velocity as $u_A = \frac{1}{\sin \phi_B} \sqrt{\frac{\rho_B(p_B - p_A)}{\rho_A(p_B - p_A)}}$

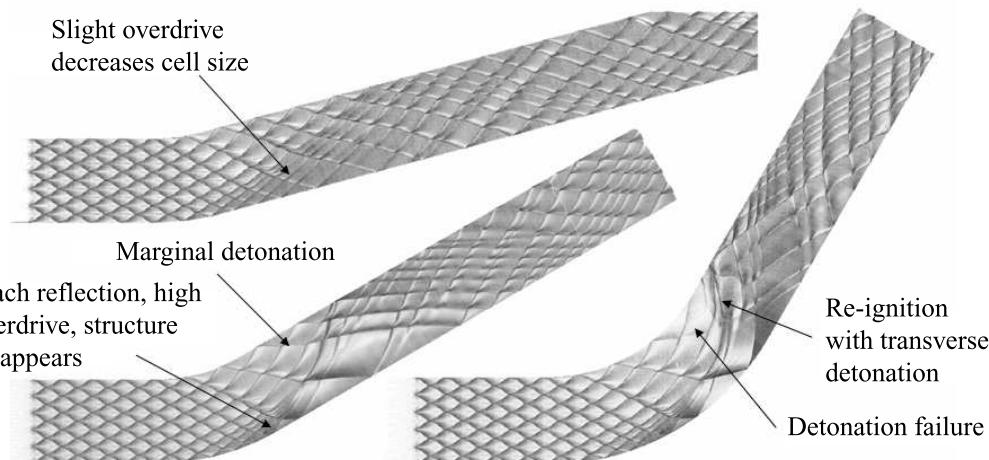


- Measure inflow angle ϕ_B between Mach stem and triple point trajectory
- Velocity a of T' relative to T cannot be derived that easily: Oblique shock relations across C and D hold true both in frame of reference for T and T'

$$\begin{aligned}\rho_C (u_{C,n} - a_n) &= \rho_D (u_{D,n} - a_n) \\ \rho_C + \rho_C (u_{C,n} - a_n)^2 &= \rho_D + \rho_D (u_{D,n} - a_n)^2 \quad \rightarrow a_n = 0, a_t \text{ arbitrary} \\ u_{C,t} - a_t &= u_{D,t} - a_t \\ h_C + \frac{1}{2} (u_{C,n} - a_n)^2 &= h_D + \frac{1}{2} (u_{D,n} - a_n)^2\end{aligned}$$

Estimate $a_t = \frac{L_R}{t_{\text{init}}}$

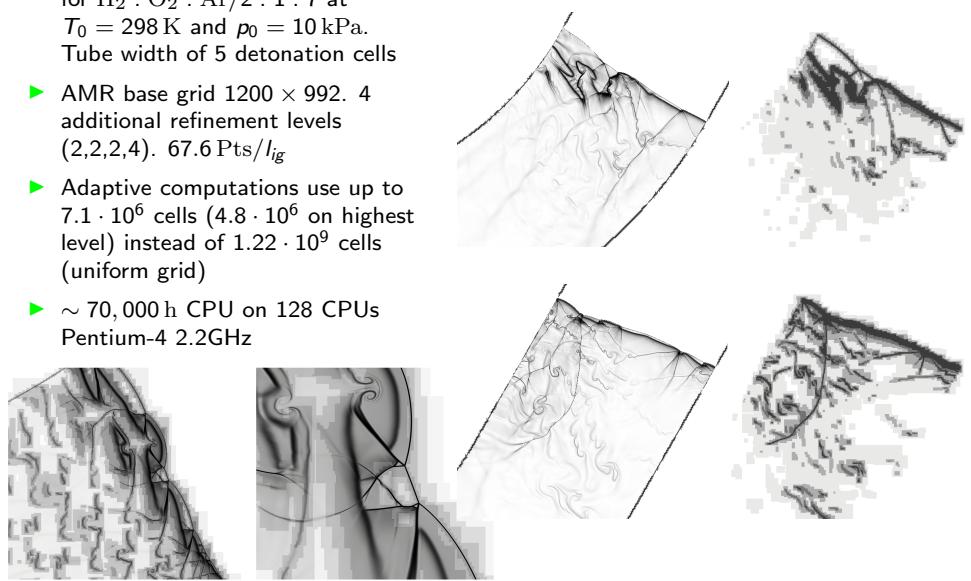
Triple point tracks



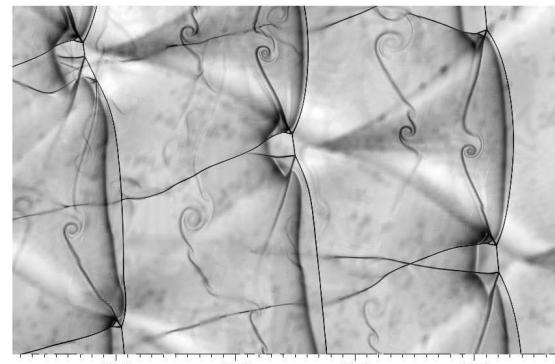
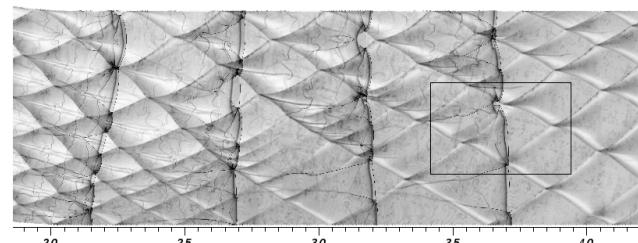
$\varphi = 15^\circ$ (left, top), $\varphi = 30^\circ$ (left, bottom), and $\varphi = 60^\circ$ (right)

Detonation propagation through pipe bends

- 2D Simulation of CJ detonation for $H_2 : O_2 : Ar/2 : 1 : 7$ at $T_0 = 298 K$ and $p_0 = 10 kPa$. Tube width of 5 detonation cells
- AMR base grid 1200×992 , 4 additional refinement levels (2,2,2,4). $67.6 \text{ Pts}/l_{ig}$
- Adaptive computations use up to $7.1 \cdot 10^6$ cells ($4.8 \cdot 10^6$ on highest level) instead of $1.22 \cdot 10^9$ cells (uniform grid)
- $\sim 70,000$ h CPU on 128 CPUs Pentium-4 2.2GHz

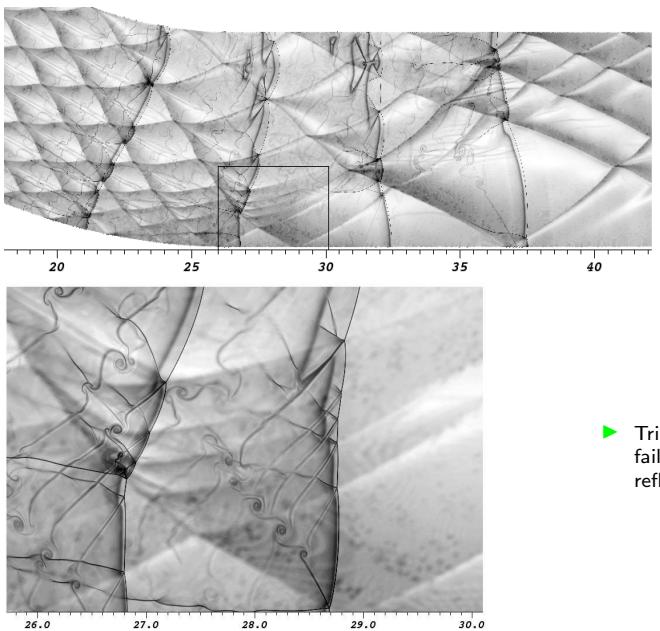


Triple point structures – $\varphi = 15^\circ$



- Triple point re-initiation after bend with change from transitional to Double Mach reflection

Triple point structures – $\varphi = 30^\circ$



- ▶ Triple point quenching and failure as single Mach reflection

Transition criteria

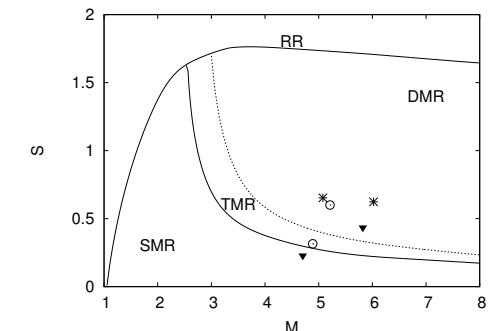
Solve system of oblique shock relations numerically and determine transition boundaries [Ben-Dor, 2007].

- ▶ Regular reflection (RR): $M_B^T < 1$
- ▶ Single Mach reflection (SMR): $M_C^T < 1$ and $M_B^T > 1$
- ▶ Transitional Mach reflection: $M_C^{T'} < 1$ and $M_C^T > 1$
- ▶ Double Mach reflection: $M_C^{T'} > 1$ and $M_C^T > 1$
- ▶ Here: Nonreactive $H_2 : O_2 : Ar$ mixture at initially 298 K and 10 kPa

For detonations:

$$S := \frac{p_C - p_D}{p_D}$$

[Deiterding, 2011]

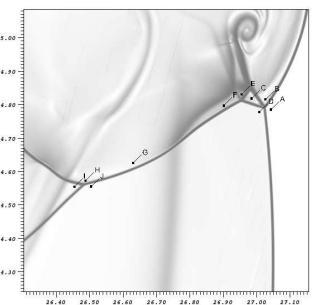


TMR/DMR transition for $a_t = 100$ m/s

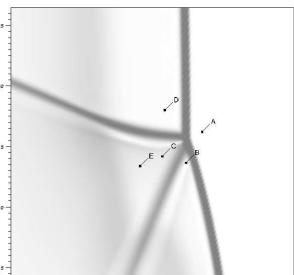
- ▶ Non-reactive shock wave reflection theory seems applicable to predict local triple point structure and stability
- ▶ Triple point type is determined solely by S and M . Useful to determine type in underresolved situations.

Triple point structures, $\varphi = 15^\circ$

Strong DMR structure in diffraction region behind bend,
 $S = 1.062$

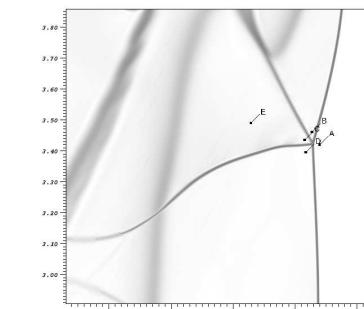


TMR structure in compression region shortly behind bend, $S = 0.338$



Triple point structures

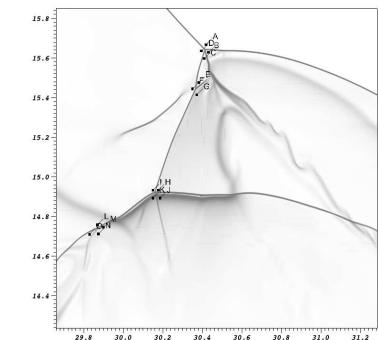
TMR structure in marginal region near limit of detonability, $\varphi = 30^\circ$, $S = 0.338$



	p/p_A	r/r_A	T [K]	v [m/s]	M
A	1.00	1.00	298	1835	5.249
B	33.77	4.33	2326	447	0.469
C	33.12	5.80	1701	1111	1.355
D	16.06	3.67	1304	1363	1.889
E	66.90	9.10	2191	758	0.818
F	57.94	7.64	2259	668	0.710
G	35.28	3.41	3235	699	
H	38.98	3.41	3589	593	
I	23.66	2.37	3149	969	
J	13.58	1.67	2570	1347	

	p/p_A	r/r_A	T [K]	v [m/s]	M
A	1.00	1.00	298	1908	5.459
B	34.14	4.21	2418	647	0.666
C	35.49	4.95	2135	929	1.015
D	26.53	4.09	1934	1085	1.243
E	34.91	4.88	2134	938	1.025

Re-ignition with strong DMR and transverse detonation,
 $\varphi = 45^\circ$, $S = 1.377$

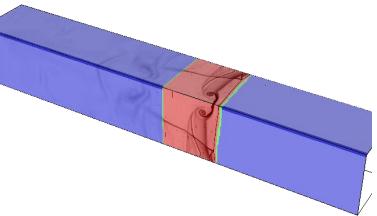


	p/p_A	r/r_A	T [K]	v [m/s]	M
A	1.00	1.00	298	1812	5.186
B	32.58	4.27	2272	456	0.483
C	33.23	6.21	1594	1156	1.454
D	13.98	3.58	1162	1446	2.119
E	31.54	6.30	1492	1208	1.569
F	16.13	4.14	1161	1393	2.042
G	41.63	7.45	1665	1034	1.274
H	30.57	6.31	1443	1180	1.557
I	14.11	3.85	1092	1431	2.161
J	77.31	9.08	2610	756	
K	78.85	8.59	2812	521	

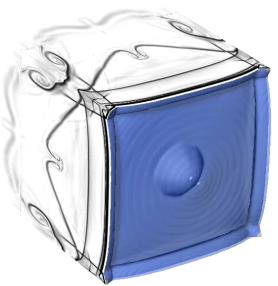
Detonation cell structure in 3D

- ▶ Simulation of only one quadrant
- ▶ 44.8 Pts/ l_{ig} for H₂ : O₂ : Ar CJ detonation
- ▶ SAMR base grid 400x24x24, 2 additional refinement levels (2, 4)
- ▶ Simulation uses ~ 18 M cells instead of ~ 118 M (unigrid)
- ▶ ~ 51, 000 h CPU on 128 CPU Compaq Alpha. \mathcal{H} : 37.6%, S : 25.1%

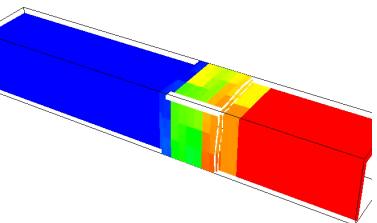
Schlieren on refinement levels



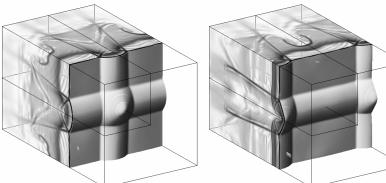
Schlieren and isosurface of Y_{OH}



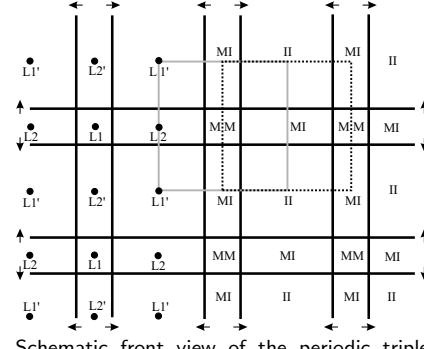
Distribution to 128 processors



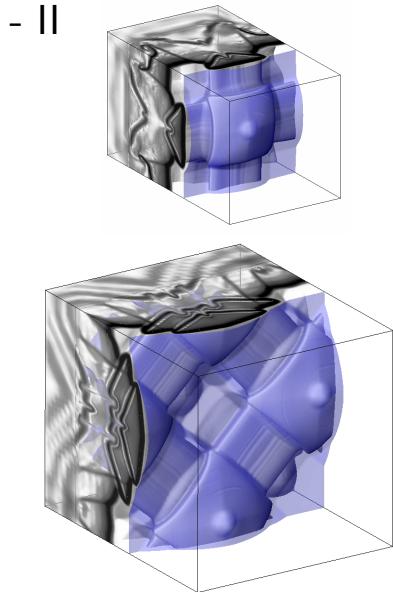
Detonation cell structure in 3D - II



Schlieren plots of density, mirrored for visualization

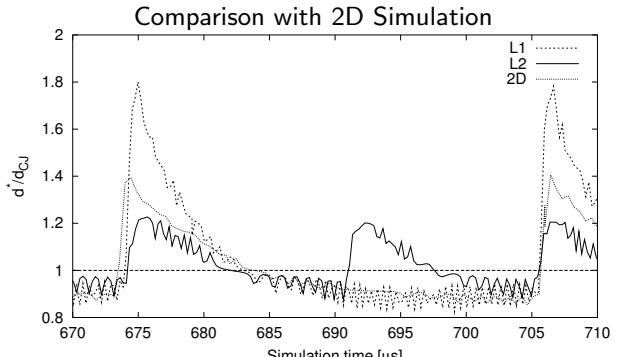
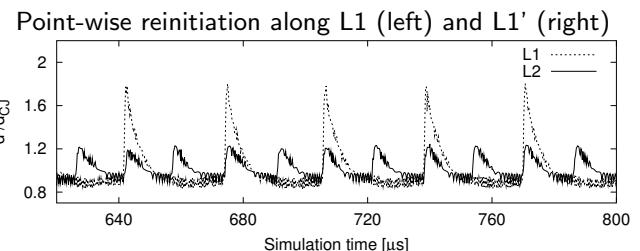


Schematic front view of the periodic triple
Detonation simulation
Combustion induced by projectiles



Schlieren plots of Y_{OH}
code/amroc/doc/html/apps/clawpack_2applications_2euler__chem_23d_2StrehlowH202_2StatDet_2src_2Problem_8h_source.html

Temporal Development of Detonation Velocity



Axisymmetric Navier-Stokes equations with chemical reaction

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial(\mathbf{f} - \mathbf{f}_v)}{\partial x} + \frac{\partial(\mathbf{g} - \mathbf{g}_v)}{\partial y} = \frac{\alpha}{y}(\mathbf{c} - \mathbf{g} + \mathbf{g}_v) + \mathbf{s}$$

$$\mathbf{q} = \begin{bmatrix} \rho_i \\ \rho u \\ \rho v \\ \rho E \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} \rho_i u \\ \rho u^2 + p \\ \rho u v \\ u(\rho E + p) \end{bmatrix}, \quad \mathbf{g} = \begin{bmatrix} \rho_i v \\ \rho u v \\ \rho v^2 + p \\ v(\rho E + p) \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ p - \tau_{\theta\theta} \end{bmatrix}, \quad \mathbf{s} = \begin{bmatrix} \dot{\omega}_i \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{f}_v = \begin{bmatrix} \rho D_i \frac{\partial Y_i}{\partial x} \\ \tau_{xx} \\ \tau_{xy} \\ k \frac{\partial T}{\partial x} + \rho \sum h_j D_j \frac{\partial Y_j}{\partial x} + u \tau_{xx} + v \tau_{xy} \end{bmatrix} \quad \begin{aligned} \tau_{xx} &= -\frac{2}{3} \mu (\nabla \cdot \mathbf{v}) + 2\mu \frac{\partial u}{\partial x} \\ \tau_{yy} &= -\frac{2}{3} \mu (\nabla \cdot \mathbf{v}) + 2\mu \frac{\partial v}{\partial y} \\ \tau_{\theta\theta} &= -\frac{2}{3} \mu (\nabla \cdot \mathbf{v}) + 2\mu \frac{v}{y} \end{aligned}$$

$$\mathbf{g}_v = \begin{bmatrix} \rho D_i \frac{\partial Y_i}{\partial y} \\ \tau_{xy} \\ \tau_{yy} \\ k \frac{\partial T}{\partial y} + \rho \sum h_j D_j \frac{\partial Y_j}{\partial y} + u \tau_{xy} + v \tau_{yy} \end{bmatrix} \quad \begin{aligned} \tau_{xy} &= \mu \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) \\ \nabla \cdot \mathbf{v} &= \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \alpha \frac{v}{y} \right) \end{aligned}$$

Chemistry and transport properties

Arrhenius-kinetics:

$$\dot{\omega}_i = \sum_{j=1}^M (\nu_{ji}^r - \nu_{ji}^f) \left[k_j^f \prod_{n=1}^K \left(\frac{\rho_n}{W_n} \right)^{\nu_{jn}^f} - k_j^r \prod_{n=1}^K \left(\frac{\rho_n}{W_n} \right)^{\nu_{jn}^r} \right] \quad i = 1, \dots, K$$

- ▶ Parsing of mechanisms and evaluation of $\dot{\omega}_i$ with Chemkin-II
- ▶ $c_{pi}(T)$ and $h_i(T)$ tabulated, linear interpolation between values

Mixture viscosity $\mu = \mu(T, Y_i)$ with Wilke formula

$$\mu = \sum_{i=1}^K \frac{Y_i \mu_i}{W_i \sum_{m=1}^K Y_m \Phi_{im} / W_m} \text{ with } \Phi_{im} = \frac{1}{\sqrt{8}} \left(1 + \frac{W_i}{W_m} \right)^{-\frac{1}{2}} \left(1 + \left(\frac{\mu_i}{\mu_m} \right)^{\frac{1}{2}} \left(\frac{W_m}{W_i} \right)^{\frac{1}{4}} \right)^2$$

Mixture thermal conductivity $k = k(T, Y_i)$ following Mathur

$$k = \frac{1}{2} \left(W \sum_{i=1}^K \frac{Y_i k_i}{W_i} + \frac{1}{W \sum_{i=1}^K Y_i / (W_i k_i)} \right)$$

Mixture diffusion coefficients $D_i = D_i(T, p, Y_i)$ from binary diffusion $D_{mi}(T, p)$ as

$$D_i = \frac{1 - Y_i}{W \sum_{m \neq i} Y_m / (W_m D_{mi})}$$

- ▶ Evaluation with Chemkin-II Transport library

Finite volume discretization

Time discretization $t_n = n\Delta t$, discrete volumes $I_{jk} =$

$$[x_j - \frac{1}{2}\Delta x, x_j + \frac{1}{2}\Delta x] \times [y_k - \frac{1}{2}\Delta y, y_k + \frac{1}{2}\Delta y] \times \dots = [x_{j-1/2}, x_{j+1/2}] \times [y_{k-1/2}, y_{k+1/2}]$$

Approximation $\mathbf{Q}_{jk}(t) \approx \frac{1}{|I_{jk}|} \int_{I_{jk}} \mathbf{q}(\mathbf{x}, t) dx$ and numerical fluxes

$$\mathbf{F}(\mathbf{Q}_{jk}(t), \mathbf{Q}_{j+1,k}(t)) \approx \mathbf{f}(\mathbf{q}(x_{j+1/2}, y_k, t)),$$

$$\mathbf{F}_v(\mathbf{Q}_{jk}(t), \mathbf{Q}_{j+1,k}(t)) \approx \mathbf{f}_v(\mathbf{q}(x_{j+1/2}, y_k, t), \nabla \mathbf{q}(x_{j+1/2}, y_k, t))$$

yield (for simplicity)

$$\mathbf{Q}_{jk}^{n+1} = \mathbf{Q}_{jk}^n - \frac{\Delta t}{\Delta x} \left[\mathbf{F}(\mathbf{Q}_{jk}^n, \mathbf{Q}_{j+1,k}^n) - \mathbf{F}(\mathbf{Q}_{j-1,k}^n, \mathbf{Q}_{jk}^n) \right] + \frac{\Delta t}{\Delta x} \left[\mathbf{F}_v(\mathbf{Q}_{jk}^n, \mathbf{Q}_{j+1,k}^n) - \mathbf{F}_v(\mathbf{Q}_{j-1,k}^n, \mathbf{Q}_{jk}^n) \right]$$

- ▶ Riemann solver to approximate $\mathbf{F}(\mathbf{Q}_{jk}^n, \mathbf{Q}_{j+1,k}^n)$
- ▶ 1st-order finite differences for $\mathbf{F}_v(\mathbf{Q}_{jk}^n, \mathbf{Q}_{j+1,k}^n)$ yield 2nd-order accurate central differences in (*)

Stability condition used:

$$\max_{i,j,k} \left\{ \frac{\Delta t}{\Delta x} (|u_{jk}| + c_{jk}) + \frac{8}{3} \frac{\mu_{jk} \Delta t}{\rho_{jk} \Delta x^2}, \frac{\Delta t}{\Delta x} (|u_{jk}| + c_{jk}) + \frac{2k_j \Delta t}{c_{v,jk} \rho_j \Delta x^2}, \frac{\Delta t}{\Delta x} (|u_{jk}| + c_{jk}) + D_{i,jk} \frac{\Delta t}{\Delta x^2} \right\} \leq 1$$

Splitting method

$$\partial_t \mathbf{q} + \partial_x (\mathbf{f} - \mathbf{f}_v) + \partial_y (\mathbf{g} - \mathbf{g}_v) = \frac{\alpha}{y} (\mathbf{c} - \mathbf{g} + \mathbf{g}_v) + \mathbf{s}$$

Dimensional splitting for PDE

$$\mathcal{X}^{(\Delta t)} : \partial_t \mathbf{q} + \partial_x (\mathbf{f}(\mathbf{q}) - \mathbf{f}_v(\mathbf{q})) = 0, \quad \text{IC: } \mathbf{Q}(t_m) \xrightarrow{\Delta t} \tilde{\mathbf{Q}}^{1/2}$$

$$\mathcal{Y}^{(\Delta t)} : \partial_t \mathbf{q} + \partial_y (\mathbf{g}(\mathbf{q}) - \mathbf{g}_v(\mathbf{q})) = 0, \quad \text{IC: } \tilde{\mathbf{Q}}^{1/2} \xrightarrow{\Delta t} \tilde{\mathbf{Q}}$$

Treat right-hand side as source term

$$\mathcal{C}^{(\Delta t)} : \partial_t \mathbf{q} = \frac{\alpha}{y} (\mathbf{c}(\mathbf{q}) - \mathbf{g}(\mathbf{q}) + \mathbf{g}_v(\mathbf{q})), \quad \text{IC: } \tilde{\mathbf{Q}} \xrightarrow{\Delta t} \bar{\mathbf{Q}}$$

Chemical source term

$$\mathcal{S}^{(\Delta t)} : \partial_t \mathbf{q} = \mathbf{s}(\mathbf{q}), \quad \text{IC: } \bar{\mathbf{Q}} \xrightarrow{\Delta t} \mathbf{Q}(t_m + \Delta t)$$

Formally 1st-order algorithm

$$\mathbf{Q}(t_m + \Delta t) = \mathcal{S}^{(\Delta t)} \mathcal{C}^{(\Delta t)} \mathcal{Y}^{(\Delta t)} \mathcal{X}^{(\Delta t)}(\mathbf{Q}(t_m))$$

but all sub-operators 2nd-order accurate or higher.

Finite volume discretization – cont.

Symmetry source term $\mathcal{C}^{(\Delta t)}$: Use

$$\mathbf{Q}_{jk}^{n+1} = \mathbf{Q}_{jk}^n + \Delta t \left(\frac{\alpha}{y} (\mathbf{c}(\mathbf{Q}_{jk}^n) - \mathbf{g}(\mathbf{Q}_{jk}^n)) + \frac{1}{2} (\mathbf{G}_v(\mathbf{Q}_{jk}^n, \mathbf{Q}_{j,k+1}^n) + \mathbf{G}_v(\mathbf{Q}_{j,k-1}^n, \mathbf{Q}_{jk}^n)) \right)$$

within explicit 2nd-order accurate Runge-Kutta method

- ▶ Gives 2nd-order central difference approximation of \mathbf{G}_v
- ▶ Transport properties μ, k, D_i are stored in vector of state \mathbf{Q} and kept constant throughout entire time step

Chemical source term $\mathcal{S}^{(\cdot)}$:

- ▶ 4th-order accurate semi-implicit ODE-solver subcycles within each cell
- ▶ ρ, e, u, v remain unchanged!

$$\partial_t \rho_i = W_i \dot{\omega}_i(\rho_1, \dots, \rho_K, T) \quad i = 1, \dots, K$$

Lehr's ballistic range experiments

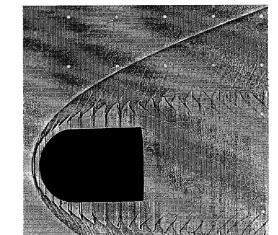
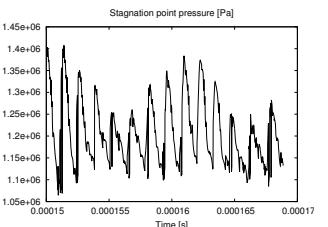
- ▶ Spherical-nosed projectile of radius 1.5 mm travels with constant velocity through stoichiometric H₂ : O₂ : N₂ mixture (molar ratios 2:1:3.76) at 42.663 kPa and T = 293 K [Lehr, 1972]
- ▶ Mechanism by [Jachimowski, 1988]: 19 equilibrium reactions, 9 species. Chapman Jouguet velocity ~ 1957 m/s.
- ▶ Axisymmetric Navier-Stokes and Eulers simulations on AMR base mesh of 400 × 200 cells, physical domain size 6 cm × 3 cm
- ▶ 4-level computations with refinement factors 2,2,4 to final time t = 170 μ s. Refinement downstream removed.
- ▶ Main configurations
 - ▶ Velocity $v_f = 1931$ m/s ($M = 4.79$), ~ 40 Pts/ l_{ig}
 - ▶ Velocity $v_f = 1806$ m/s ($M = 4.48$), ~ 60 Pts/ l_{ig}
- ▶ Various previous studies with not entirely consistent results. E.g. [Yungster and Radhakrishnan, 1996], [Axdahl et al., 2011]
- ▶ Stagnation point location and pressure tracked in every time step
- ▶ All computations were on 32 cores requiring ~ 1500 h CPU each

code/amroc/doc/html/apps/clawpack_2applications_2euler__chem_22d_2SphereLehr_2src_2Problem_8h_source.html

code/amroc/doc/html/apps/clawpack_2applications_2euler__chem_22d_2SphereLehrNav_2src_2Problem_8h_source.html

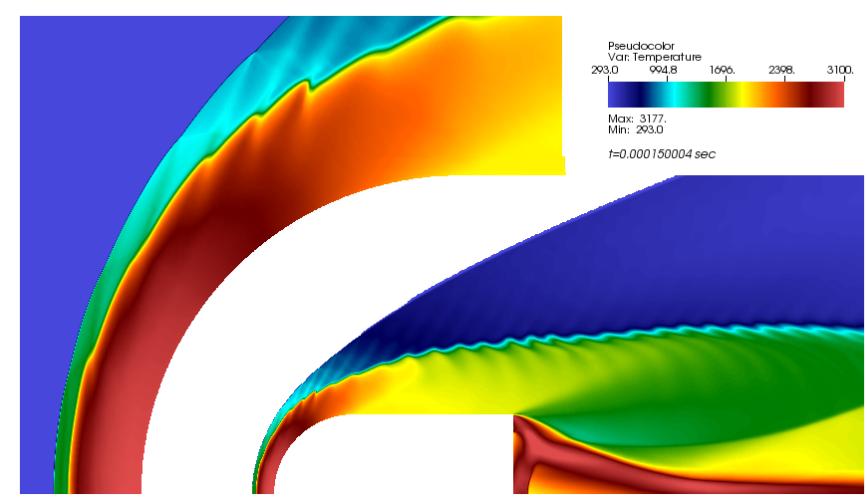
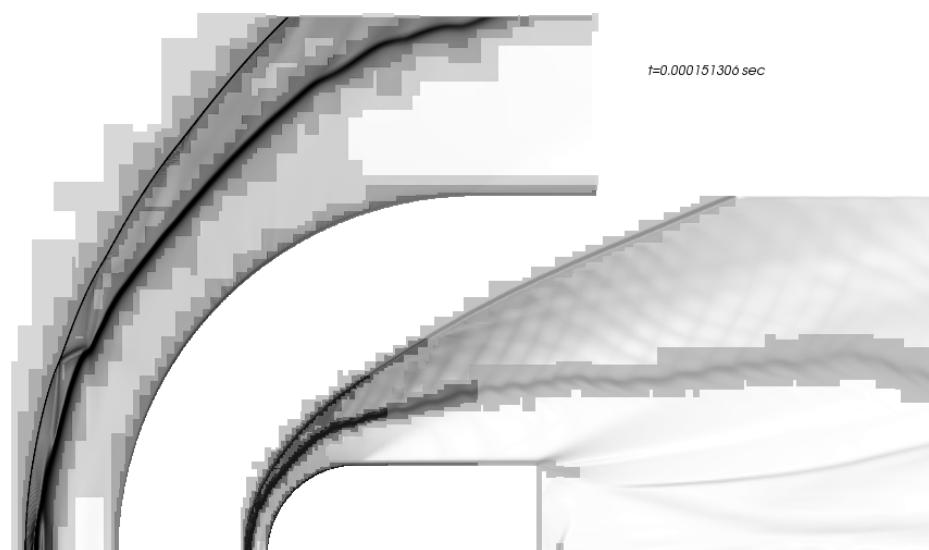
Viscous case – M = 4.79

- ▶ 5619 iterations with CFL=0.9 to t = 170 μ s
- ▶ Oscillation frequency in last 20 μ s: ~ 722 kHz (viscous), ~ 737 kHz (inviscid)
- ▶ Experimental value: ~ 720 kHz



Schlieren plot of density

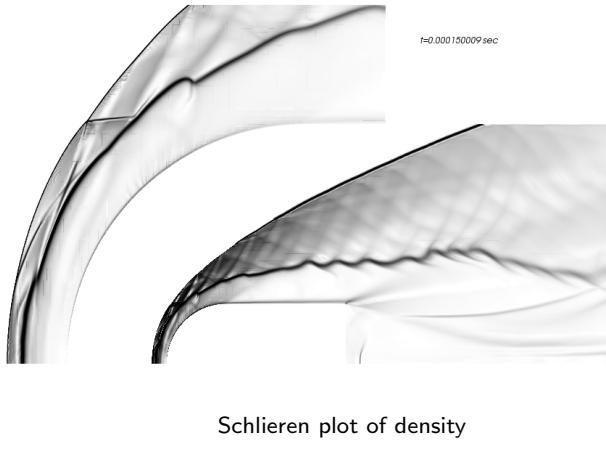
Comparison of temperature field



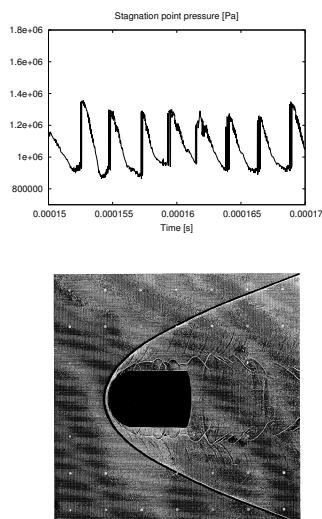
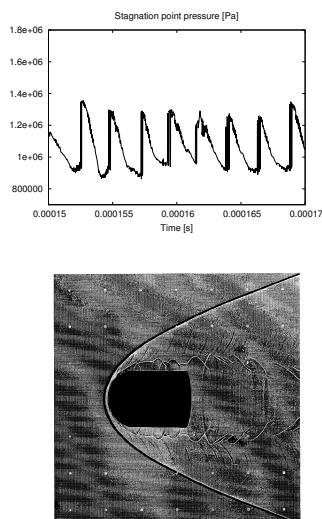
Inviscid

Viscous case – $M = 4.48$

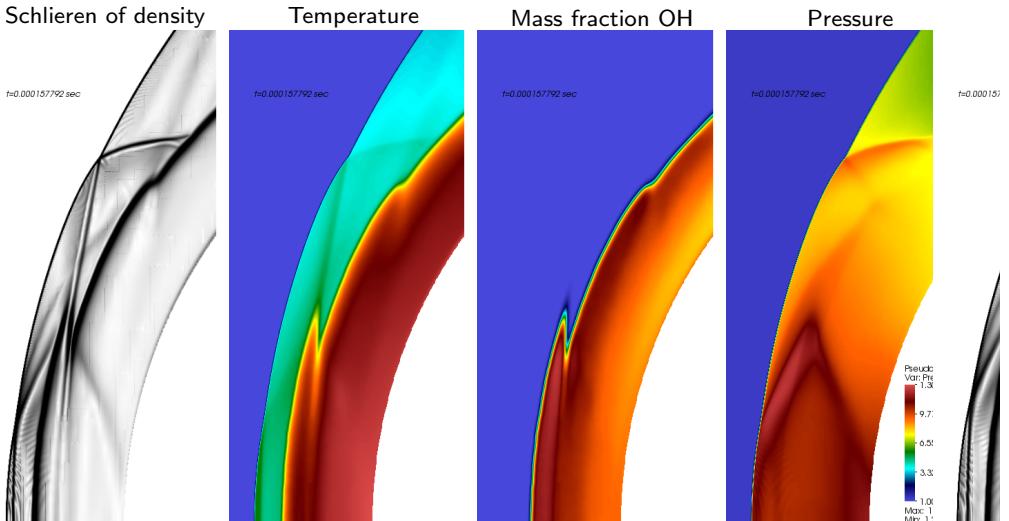
- ▶ 5432 iterations with CFL=0.9 to $t = 170 \mu\text{s}$
- ▶ Oscillation frequency in last $20 \mu\text{s}$: $\sim 417 \text{ kHz}$
- ▶ Experimental value: $\sim 425 \text{ kHz}$



Schlieren plot of density



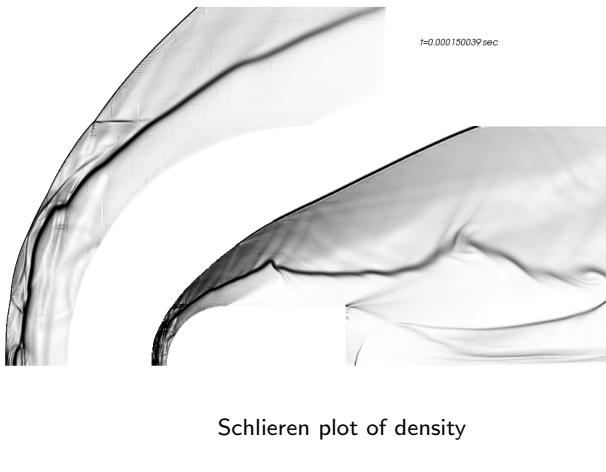
Oscillation mechanism



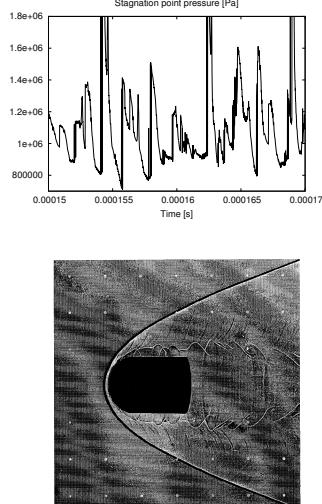
- ▶ Oscillation created by accelerated reaction due to slip line from previous triple point

Inviscid case – $M = 4.48$

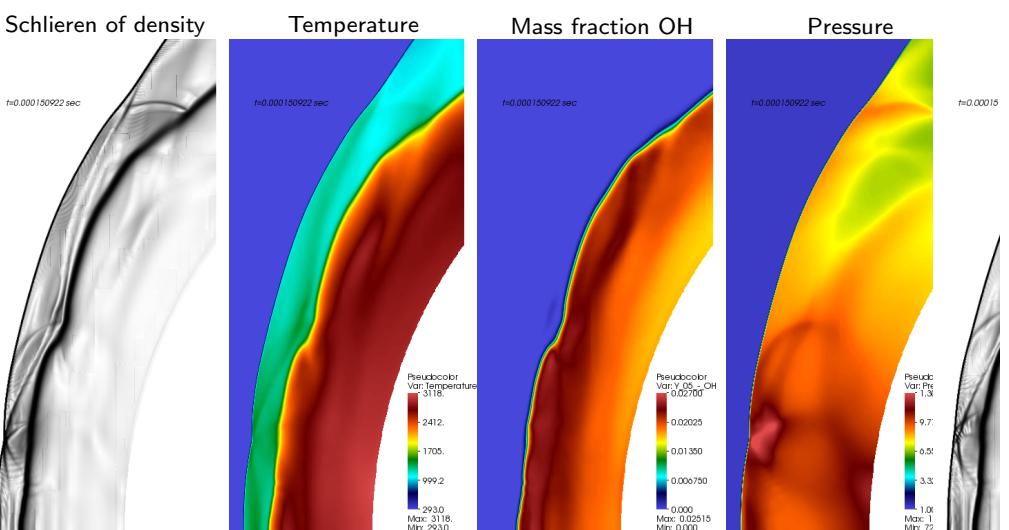
- ▶ 4048 iterations with CFL=0.9 to $t = 170 \mu\text{s}$
- ▶ Oscillation frequency in last $20 \mu\text{s}$: $\sim 395 \text{ kHz}$
- ▶ Experimental value: $\sim 425 \text{ kHz}$



Schlieren plot of density



Perturbed oscillation mechanism



- ▶ Small perturbations can quickly create numerous triple points

Hybrid method

Convective numerical flux is defined as

$$\mathbf{F}_{inv}^n = \begin{cases} \mathbf{F}_{inv-WENO}^n, & \text{in } \mathcal{C} \\ \mathbf{F}_{inv-CD}^n, & \text{in } \bar{\mathcal{C}}, \end{cases}$$

- For LES: 3rd order WENO method, 2nd order TCD [Hill and Pullin, 2004]
- For DNS: Symmetric 6th order WENO, 6th-order CD scheme [Ziegler et al., 2011]

Use WENO scheme to only capture shock waves but resolve interface between species.

Shock detection based on using two criteria together:

1. Lax-Liu entropy condition $|u_R \pm a_R| < |u_* \pm a_*| < |u_L \pm a_L|$ tested with a threshold to eliminate weak acoustic waves. Used intermediate states at cell interfaces:

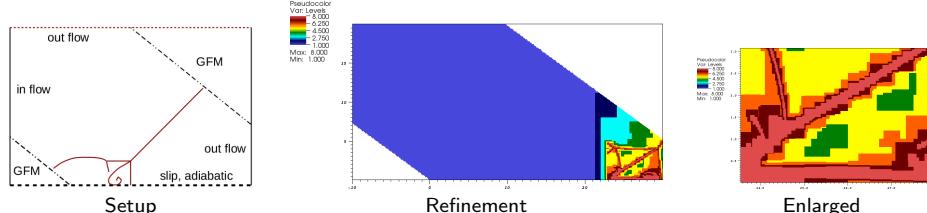
$$u_* = \frac{\sqrt{\rho_L u_L} + \sqrt{\rho_R u_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \quad a_* = \sqrt{(\gamma_* - 1)(h_* - \frac{1}{2} u_*^2)}, \dots$$

2. Limiter-inspired discontinuity test based on mapped normalized pressure gradient θ_j

$$\phi(\theta_j) = \frac{2\theta_j}{(1+\theta_j)^2} \quad \text{with} \quad \theta_j = \frac{|p_{j+1} - p_j|}{|p_{j+1} + p_j|}, \quad \phi(\theta_j) > \alpha_{Map}$$

DNS of shear layer in detonation triple point

- Calorically perfect two-species model with $\gamma = 1.29499$ and $h_0 = 54,000 \text{ J/mol}$ and one-step Arrhenius reaction with parameters $E_a = 30,000 \text{ J/mol}$, $A = 6 \cdot 10^5 \text{ s}^{-1}$, $W = 0.029 \text{ kg/mol}$ → 1d ZND theory predicts $d_{CJ} = 1587.8 \text{ m/s}$
- For dynamic viscosity, heat conductivity, and mass diffusion simple Sutherland models are used
- Distance $L(t) = d_{CJ} \sin(\theta)t$ is used to define a Reynolds number as $\text{Re} = \frac{\rho_0 \theta_0 L(t)}{\mu_0}$
- Viscous shear layer thickness, thermal heat conduction layer thickness, and mass diffusion layer thickness grow as $\delta_{visc} \approx \sqrt{\frac{\mu}{\rho}} t$, $\delta_{cond} \approx \sqrt{\frac{k_{ref}}{\rho c_v}} t$, $\delta_{mass,i} \approx \sqrt{\frac{D_i}{\rho}} t$
- Only shock thickness not resolved → "pseudo-DNS"
- Computations with WENO/CD/RK3 use SAMR base mesh 320×160 and up to 8 levels refined by factor 2, domain: $40 \text{ mm} \times 20 \text{ mm}$
- Computations with MUSCL scheme use base mesh 590×352 and up to 7 levels refined by factor 2, domain: $40 \text{ mm} \times 22 \text{ mm}$



SAMR flux correction for Runge-Kutta method

Recall Runge-Kutta temporal update

$$\tilde{\mathbf{Q}}_j^v = \alpha_v \mathbf{Q}_j^m + \beta_v \tilde{\mathbf{Q}}_j^{v-1} + \gamma_v \frac{\Delta t}{\Delta x_n} \Delta \mathbf{F}^n(\tilde{\mathbf{Q}}^{v-1})$$

rewrite scheme as

$$\mathbf{Q}^{m+1} = \mathbf{Q}^m - \sum_{v=1}^{\tau} \varphi_v \frac{\Delta t}{\Delta x_n} \Delta \mathbf{F}^n(\tilde{\mathbf{Q}}^{v-1}) \quad \text{with} \quad \varphi_v = \gamma_v \prod_{\nu=v+1}^{\tau} \beta_{\nu}$$

Flux correction to be used

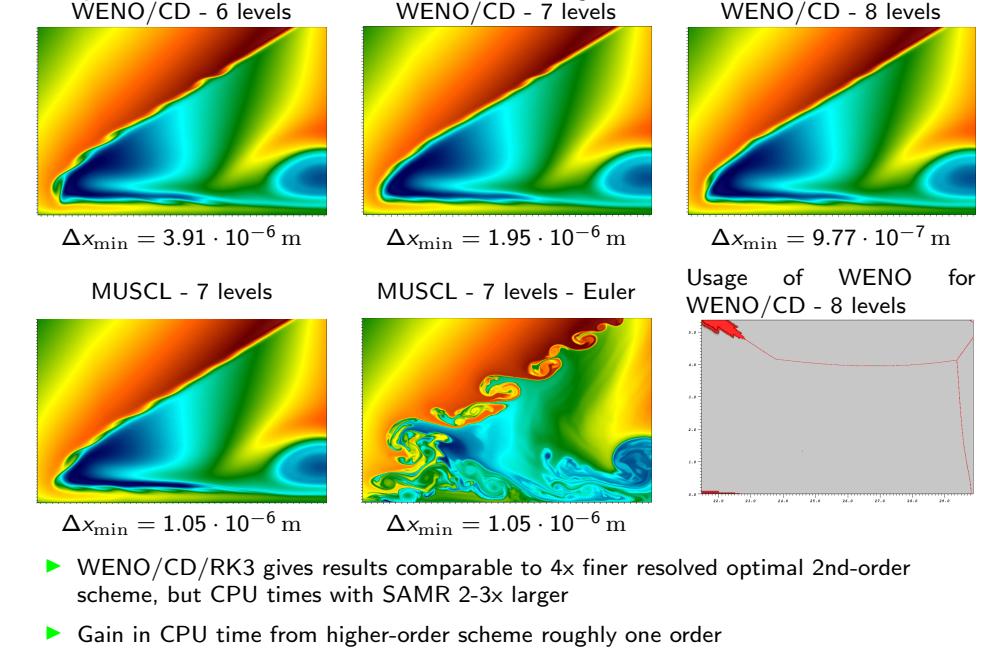
1. $\delta \mathbf{F}_{i-\frac{1}{2},j}^{1,l+1} := -\varphi_1 \mathbf{F}_{i-\frac{1}{2},j}^{1,l}(\tilde{\mathbf{Q}}^0), \quad \delta \mathbf{F}_{i-\frac{1}{2},j}^{1,l+1} := \delta \mathbf{F}_{i-\frac{1}{2},j}^{1,l+1} - \sum_{v=2}^{\tau} \varphi_v \mathbf{F}_{i-\frac{1}{2},j}^{1,l}(\tilde{\mathbf{Q}}^{v-1})$
2. $\delta \mathbf{F}_{i-\frac{1}{2},j}^{1,l+1} := \delta \mathbf{F}_{i-\frac{1}{2},j}^{1,l+1} + \frac{1}{r_{l+1}} \sum_{\nu=0}^{r_{l+1}-1} \sum_{v=1}^{\tau} \varphi_v \mathbf{F}_{v+\frac{1}{2},w+l}^{1,l+1} (\tilde{\mathbf{Q}}^{v-1}(t + \kappa \Delta t_{l+1}))$

Storage-efficient SSPRK(3,3):

v	α_v	β_v	γ_v	φ_v
1	1	0	1	$\frac{1}{6}$
2	$\frac{3}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{6}$
3	$\frac{1}{2}$	$\frac{3}{2}$	$\frac{2}{3}$	$\frac{2}{3}$

[Pantano et al., 2007]

Computational results for shear layer



- WENO/CD/RK3 gives results comparable to 4x finer resolved optimal 2nd-order scheme, but CPU times with SAMR 2-3x larger
- Gain in CPU time from higher-order scheme roughly one order

References I

- [Axdahl et al., 2011] Axdahl, E., Kumar, A., and Wilhite, A. (2011). Study of unsteady, sphere-driven, shock-induced combustion for application to hypervelocity airbreathing propulsio. In *Proc. 47th AIAA/ASME/SAE/SAE Joint Propulsion Conference & Exhibit*.
- [Ben-Dor, 2007] Ben-Dor, G. (2007). *Shock wave reflection phenomena*. Springer-Verlag, Berlin, Heidelberg, 2nd edition.
- [Deiterding, 2011] Deiterding, R. (2011). High-resolution numerical simulation and analysis of Mach reflection structures in detonation waves in low-pressure $\text{h}_2 : \text{o}_2 : \text{ar}$ mixtures: a summary of results obtained with adaptive mesh refinement framework AMROC. *J. Combustion*, 2011:738969.
- [Hill and Pullin, 2004] Hill, D. J. and Pullin, D. I. (2004). Hybrid tuned center difference - WENO method for large eddy simulations in the presence of strong shocks. *J. Comput. Phys.*, 194(2):435–450.
- [Jachimowski, 1988] Jachimowski, C. J. (1988). An analytical study of the hydrogen-air reaction mechanism with application to scramjet combustion. Technical Report TP-2791, NASA.

References II

- [Lehr, 1972] Lehr, H. F. (1972). Experiments on shock-induced combustion. *Astronautica Acta*, 17:589–597.
- [Pantano et al., 2007] Pantano, C., Deiterding, R., Hill, D. J., and Pullin, D. I. (2007). A low-numerical dissipation patch-based adaptive mesh refinement method for large-eddy simulation of compressible flows. *J. Comput. Phys.*, 221(1):63–87.
- [Yungster and Radhakrishnan, 1996] Yungster, S. and Radhakrishnan, K. (1996). A fully implicit time accurate method for hypersonic combustion: application to shock-induced combustion instability. *Shock Waves*, 5:293–303.
- [Ziegler et al., 2011] Ziegler, J. L., Deiterding, R., Shepherd, J. E., and Pullin, D. I. (2011). An adaptive high-order hybrid scheme for compressive, viscous flows with detailed chemistry. *J. Comput. Phys.*, 230(20):7598–7630.

Outline

Lecture 6

Fluid-structure interaction simulation

Course *Block-structured Adaptive Finite Volume Methods in C++*

Fluid-structure interaction

- Coupling to a solid mechanics solver
- Implementation
- Rigid body motion
- Thin elastic and deforming thin structures
- Deformation from water hammer
- Real-world example

Massively parallel SAMR

Performance data from AMROC

Ralf Deiterding
University of Southampton
Engineering and the Environment
Highfield Campus, Southampton SO17 1BJ, UK

E-mail: r.deiterding@soton.ac.uk

Outline

Fluid-structure interaction

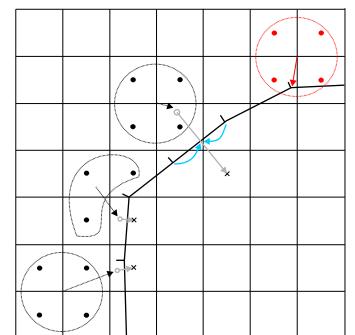
- Coupling to a solid mechanics solver
- Implementation
- Rigid body motion
- Thin elastic and deforming thin structures
- Deformation from water hammer
- Real-world example

Massively parallel SAMR

Performance data from AMROC

Construction of coupling data

- ▶ Moving boundary/interface is treated as a moving contact discontinuity and represented by level set [Fedkiw, 2002][Arienti et al., 2003]
- ▶ Efficient construction of level set from triangulated surface data with closest-point-transform (CPT) algorithm [Mauch, 2003]
- ▶ One-sided construction of mirrored ghost cell and new nodal point values
- ▶ Gathering of solid force and momentum information and solution of equations of motion on central node
- ▶ Stable explicit coupling possible if geometry and velocities are prescribed for compressible fluid [Specht, 2000]



Usage of SAMR

- ▶ Eulerian SAMR + non-adaptive Lagrangian FEM scheme
- ▶ Exploit SAMR time step refinement for effective coupling to solid solver
 - ▶ Lagrangian simulation is called only at level $l_c \leq l_{\max}$
 - ▶ SAMR refines solid boundary at least at level l_c
 - ▶ Additional levels can be used resolve geometric ambiguities
- ▶ Nevertheless: Inserting sub-steps accommodates for time step reduction from the solid solver within an SAMR cycle
- ▶ Communication strategy:
 - ▶ Updated boundary info from solid solver must be received before regridding operation
 - ▶ Boundary data is sent to solid when highest level available
- ▶ Inter-solver communication (point-to-point or globally) managed on the fly special coupling module

SAMR algorithm for FSI coupling

`AdvanceLevel(l)`

Repeat r_l times

Set ghost cells of $\mathbf{Q}'(t)$

CPT(φ' , C' , \mathcal{I} , δ_l)

If time to regrid?

Regrid(l)

UpdateLevel(l)

If level $l+1$ exists?

Set ghost cells of $\mathbf{Q}'(t + \Delta t_l)$

AdvanceLevel($l+1$)

Average $\mathbf{Q}'^{l+1}(t + \Delta t_l)$ onto $\mathbf{Q}'(t + \Delta t_l)$

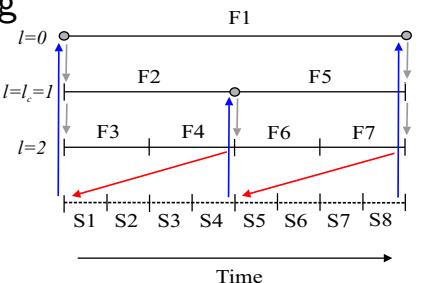
If $l = l_c$?

SendInterfaceData($p^F(t + \Delta t_l)|_{\mathcal{I}}$)

If $(t + \Delta t_l) < (t_0 + \Delta t_0)$?

ReceiveInterfaceData(\mathcal{I} , $\mathbf{u}^S|_{\mathcal{I}}$)

$t := t + \Delta t_l$



- ▶ Call CPT algorithm before Regrid(l)
- ▶ Include also call to CPT(\cdot) into Recompose(l) to ensure consistent level set data on levels that have changed
- ▶ Communicate boundary data on coupling level l_c

Fluid and solid update / exchange of time steps

FluidStep()

$$\Delta\tau_F := \min_{l=0, \dots, l_{\max}} (R_l \cdot \text{StableFluidTimeStep}(l), \Delta\tau_S)$$

$$\Delta t_l := \Delta\tau_F / R_l \text{ for } l = 0, \dots, L$$

ReceiveInterfaceData(\mathcal{I} , $\mathbf{u}^S|_{\mathcal{I}}$)

AdvanceLevel(0)

SolidStep()

$$\Delta\tau_S := \min(K \cdot R_{lc} \cdot \text{StableSolidTimeStep}(), \Delta\tau_F)$$

Repeat R_{lc} times

$$t_{\text{end}} := t + \Delta\tau_S / R_{lc}, \Delta t := \Delta\tau_S / (KR_{lc})$$

While $t < t_{\text{end}}$

SendInterfaceData($\mathcal{I}(t)$, $\bar{\mathbf{u}}^S|_{\mathcal{I}}(t)$)

ReceiveInterfaceData($p^F|_{\mathcal{I}}$)

UpdateSolid($p^F|_{\mathcal{I}}$, Δt)

$$t := t + \Delta t$$

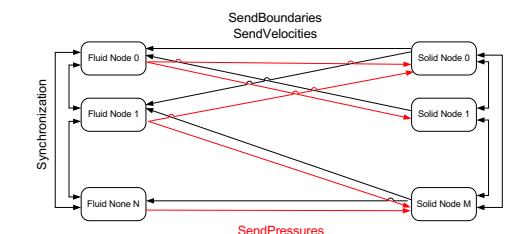
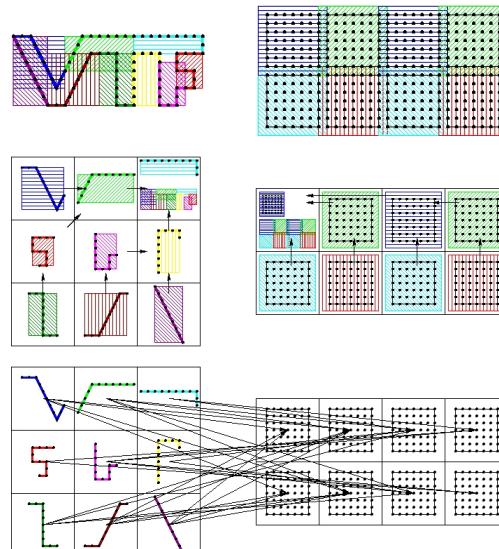
$$\Delta t := \min(\text{StableSolidTimeStep}(), t_{\text{end}} - t)$$

$$\text{with } R_l = \prod_{i=0}^l r_i$$

- ▶ Time step stays constant for R_{lc} steps, which corresponds to one fluid step at level 0

Eulerian/Lagrangian communication module

1. Put bounding boxes around each solid processors piece of the boundary and around each fluid processors grid
2. Gather, exchange and broadcast of bounding box information
3. Optimal point-to-point communication pattern, non-blocking

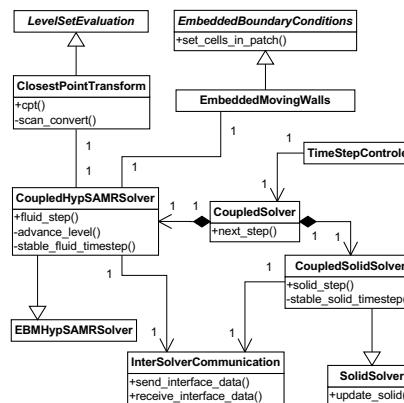


Parallelization strategy for coupled simulations

Coupling of an Eulerian FV fluid Solver and a Lagrangian FEM Solver:

- ▶ Distribute both meshes separately and copy necessary nodal values and geometry data to fluid nodes
- ▶ Setting of ghost cell values becomes strictly local operation
- ▶ Construct new nodal values strictly local on fluid nodes and transfer them back to solid nodes
- ▶ Only surface data is transferred
- ▶ Asynchronous communication ensures scalability
- ▶ Generic encapsulated implementation guarantees reusability

FSI coupling



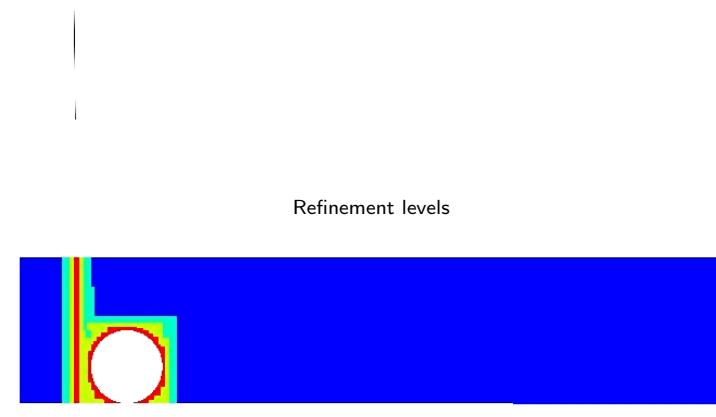
- ▶ Coupling algorithm implemented in further derived HypSAMRSolver class
- ▶ Level set evaluation always with CPT algorithm
- ▶ Parallel communication through efficient non-blocking communication module ELC
- ▶ Time step selection for both solvers through CoupledSolver class

- ▶ AMRELCGFSolver<VectorType, FixupType, FlagType, dim> is the derived AMRSolver<> class. [code/amroc/doc/html/amr/classAMRELCGFSolver.html](#)
- ▶ Uses the Eulerian interface of the Lagrangian communication routines [code/stlib/doc/html/elc/elc_page.html](#)
- ▶ and the closest point transform algorithm [code/stlib/doc/html/cpt/cpt_page.html](#) through the CPTLevelSet<DataType, dim> [code/amroc/doc/html/amr/classCPTLevelSet.html](#)

Lift-up of a spherical body

Cylindrical body hit by Mach 3 shockwave, 2D test case by [Falcovitz et al., 1997]

Schlieren plot of density



code/amroc/doc/html/apps/clawpack_2applications_2euler_22d_2SphereLiftOff_2src_2Problem_8h_source.html

Verification and validation

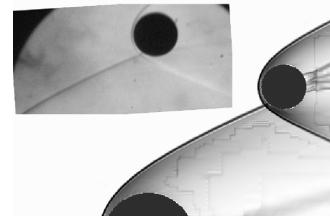
Static force measurements, $M = 10$:
[Laurence et al., 2007]

- Refinement study: $40 \times 40 \times 32$ base grid, up to without AMR up to $\sim 209.7 \cdot 10^6$ cells, largest run $\sim 35,000$ h CPU

I_{\max}	C_D	ΔC_D	C_L	ΔC_L
1	1.264		-0.176	
2	1.442	0.178	-0.019	0.157
3	1.423	-0.019	0.052	0.071
4	1.408	-0.015	0.087	0.035

- Comparison with experimental results: 3 additional levels, ~ 2000 h CPU

	Experimental	Computational
C_D	1.11 ± 0.08	1.01
C_L	0.29 ± 0.05	0.28



Dynamic motion, $M = 4$:

- Base grid $150 \times 125 \times 90$, two additional levels with $r_{1,2} = 2$
- 24,704 time steps, 36,808 h CPU on 256 cores IBM BG/P



[Laurence and Deiterding, 2011]

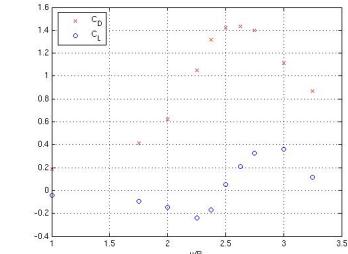
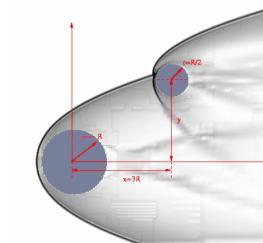
Proximal bodies in hypersonic flow

Flow modeled by Euler equations for a single polytropic gas with $p = (\gamma - 1) \rho e$

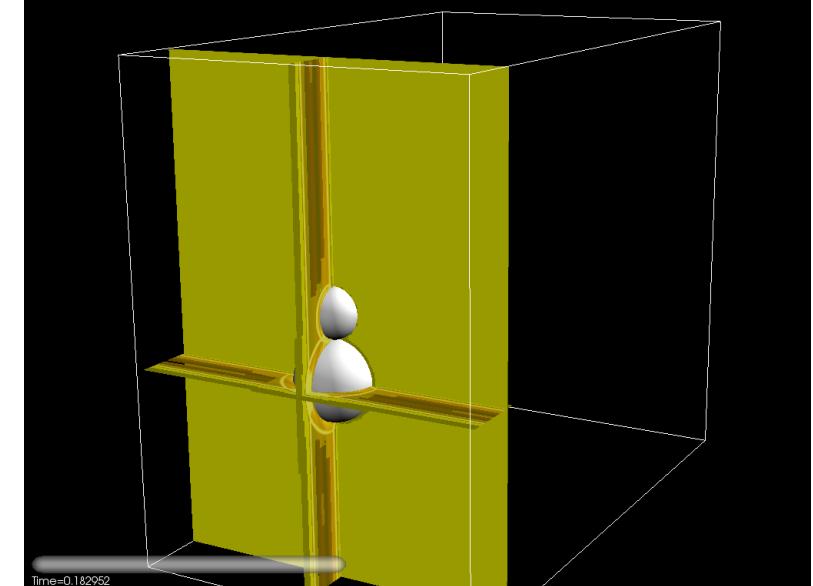
$$\partial_t \rho + \partial_{x_n}(\rho u_n) = 0, \quad \partial_t(\rho u_k) + \partial_{x_n}(\rho u_k u_n + \delta_{kn} p) = 0, \quad \partial_t(\rho E) + \partial_{x_n}(u_n(\rho E + p)) = 0$$

Numerical approximation with

- Finite volume flux-vector splitting scheme with MUSCL reconstruction, dimensional splitting
- Spherical bodies, force computation with overlaid latitude-longitude mesh to obtain drag and lift coefficients $C_{D,L} = \frac{2F_{D,L}}{\rho v^2 \pi r^2}$
- inflow $M = 10$, C_D and C_L on secondary sphere, lateral position varied, no motion



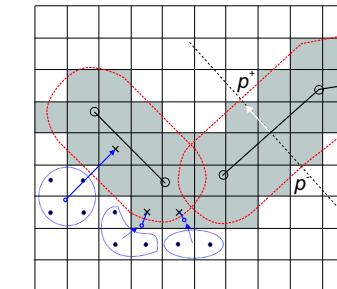
Schlieren graphics on refinement regions



code/amroc/doc/html/apps/clawpack_2applications_2euler_23d_2Spheres_2src_2Problem_8h_source.html

Treatment of thin structures

- Thin boundary structures or lower-dimensional shells require "thickening" to apply embedded boundary method
- Unsigned distance level set function φ
- Treat cells with $0 < \varphi < d$ as ghost fluid cells
- Leaving φ unmodified ensures correctness of $\nabla\varphi$
- Use face normal in shell element to evaluate in $\Delta p = p^+ - p^-$
- Utilize finite difference solver using the beam equation



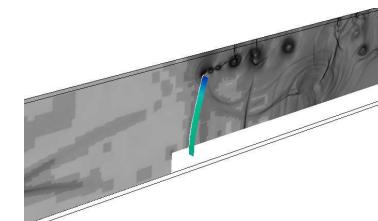
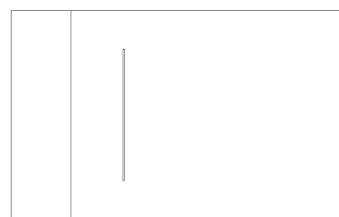
$$\rho_s h \frac{\partial^2 w}{\partial t^2} + EI \frac{\partial^4 w}{\partial x^4} = p^F$$

to verify FSI algorithms

Shock-driven elastic panel motion

Test case suggested by [Giordano et al., 2005]

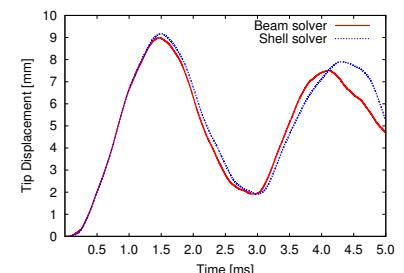
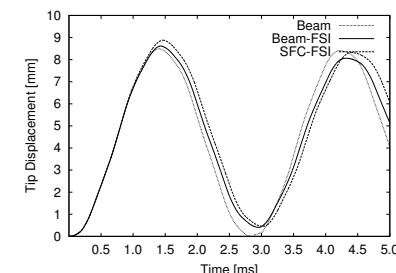
- Forward facing step geometry, fixed walls everywhere except at inflow
- SAMR base mesh $320 \times 64 (\times 2)$, $r_{1,2} = 2$
- Intel 3.4GHz Xeon dual processors, GB Ethernet interconnect
- Beam-FSI:** 12.25 h CPU on 3 fluid CPU + 1 solid CPU
[code/doc/html/capps/beam-amroc_2VibratingBeam_2src_2FluidProblem_8h_source.html](#),
[code/doc/html/capps/beam-amroc_2VibratingBeam_2src_2SolidProblem_8h_source.html](#)
- FEM-FSI:** 322 h CPU on 14 fluid CPU + 2 solid CPU
[code/doc/html/capps/sfc-amroc_2VibratingPanel_2src_2FluidProblem_8h_source.html](#),
[code/doc/html/capps/VibratingPanel_2src_2ShellManagerSpecific_8h_source.html](#)



$t = 1.56$ ms after impact

FSI verification by elastic vibration

- Thin steel plate (thickness $h = 1$ mm, length 50 mm), clamped at lower end
- $\rho_s = 7600 \text{ kg/m}^3$, $E = 220 \text{ GPa}$, $I = h^3/12$, $\nu = 0.3$
- Modeled with beam solver (101 points) and thin-shell FEM solver (325 triangles) by F. Cirak
- Left: Coupling verification with constant instantaneous loading by $\Delta p = 100 \text{ kPa}$
- Right: FSI verification with Mach 1.21 shockwave in air ($\gamma = 1.4$)



Detonation-driven plastic deformation

Chapman-Jouguet detonation in a tube filled with a stoichiometric ethylene and oxygen ($\text{C}_2\text{H}_4 + 3 \text{O}_2$, 295 K) mixture. Euler equations with single exothermic reaction $A \rightarrow B$

$$\partial_t \rho + \partial_{x_n}(\rho u_n) = 0, \quad \partial_t(\rho u_k) + \partial_{x_n}(\rho u_k u_n + \delta_{kn} p) = 0, \quad k = 1, \dots, d$$

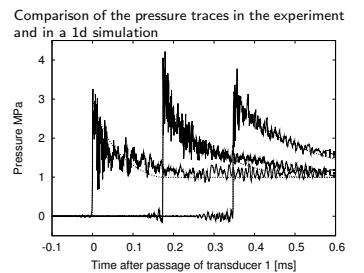
$$\partial_t(\rho E) + \partial_{x_n}(u_n(\rho E + p)) = 0, \quad \partial_t(Y\rho) + \partial_{x_n}(Y\rho u_n) = \psi$$

with

$$p = (\gamma - 1)(\rho E - \frac{1}{2}\rho u_n u_n - \rho Y q_0) \quad \text{and} \quad \psi = -k Y \rho \exp\left(\frac{-E_A \rho}{p}\right)$$

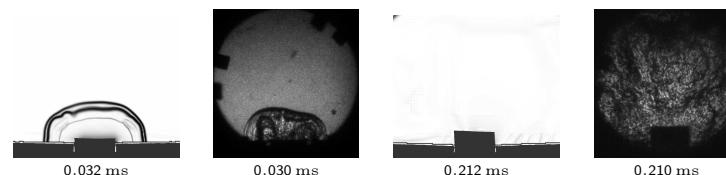
modeled with heuristic detonation model by [Mader, 1979]

$$\begin{aligned} V &:= \rho^{-1}, \quad V_0 := \rho_0^{-1}, \quad V_{\text{CJ}} := \rho_{\text{CJ}} \\ Y' &:= 1 - (V - V_0)/(V_{\text{CJ}} - V_0) \\ \text{If } 0 < Y' < 1 \text{ and } Y' > 10^{-8} \text{ then} \\ \text{If } Y < Y' \text{ and } Y' < 0.9 \text{ then } Y' &:= 0 \\ \text{If } Y' < 0.99 \text{ then } p' &:= (1 - Y')p_{\text{CJ}} \\ \text{else } p' &:= p \\ \rho_A &:= Y' \rho \\ E &:= p' / (\rho(\gamma - 1)) + Y' q_0 + \frac{1}{2} u_n u_n \end{aligned}$$

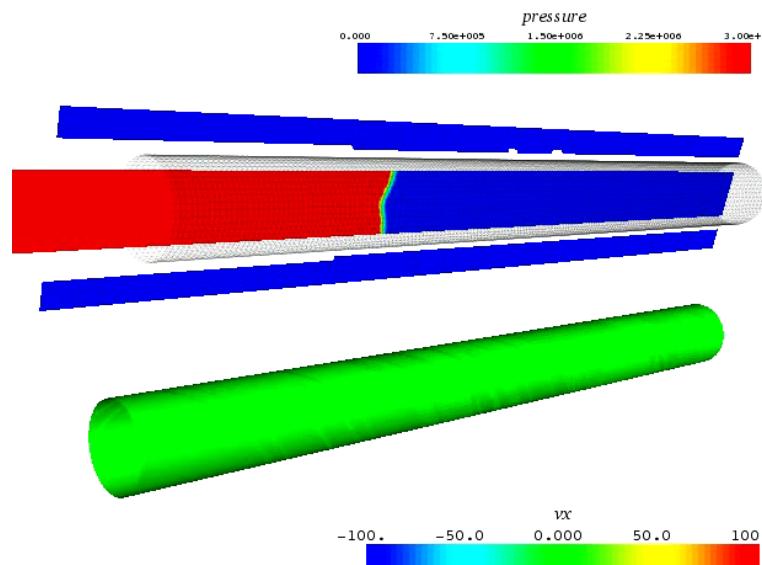


Tube with flaps

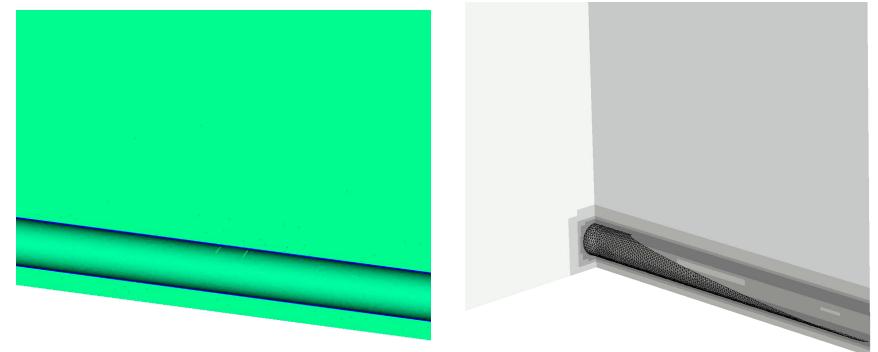
- ▶ Fluid: VanLeer FVS
 - ▶ Detonation model with $\gamma = 1.24$, $p_{CJ} = 3.3 \text{ MPa}$, $D_{CJ} = 2376 \text{ m/s}$
 - ▶ AMR base level: $104 \times 80 \times 242$, $r_{1,2} = 2$, $r_3 = 4$
 - ▶ $\sim 4 \cdot 10^7$ cells instead of $7.9 \cdot 10^9$ cells (uniform)
 - ▶ Tube and detonation fully refined
 - ▶ Thickening of 2D mesh: 0.81 mm on both sides (real 0.445 mm)
- ▶ Solid: thin-shell solver by F. Cirak
 - ▶ Aluminum, J2 plasticity with hardening, rate sensitivity, and thermal softening
 - ▶ Mesh: 8577 nodes, 17056 elements
- ▶ 16+2 nodes 2.2 GHz AMD Opteron quad processor, PCI-X 4x Infiniband network, $\sim 4320 \text{ h CPU}$ to $t_{end} = 450 \mu\text{s}$



Coupled fracture simulation



Tube with flaps: results



Fluid density and displacement in y-direction in solid

Schlieren plot of fluid density on refinement levels

[Cirak et al., 2007]

[code/doc/html/capps/sfc-amroc_2TubeCJBurnFlaps_2src_2FluidProblem_8h_source.html](#),
[code/doc/html/capps/TubeCJBurnFlaps_2src_2ShellManagerSpecific_8h_source.html](#)

Underwater explosion modeling

Volume fraction based two-component model with $\sum_{i=1}^m \alpha^i = 1$, that defines mixture quantities as

$$\rho = \sum_{i=1}^m \alpha^i \rho^i, \quad \rho u_n = \sum_{i=1}^m \alpha^i \rho^i u_n^i, \quad \rho e = \sum_{i=1}^m \alpha^i \rho^i e^i$$

Assuming total pressure $p = (\gamma - 1) \rho e - \gamma p_\infty$ and speed of sound $c = (\gamma(p + p_\infty)/\rho)^{1/2}$ yields

$$\frac{p}{\gamma - 1} = \sum_{i=1}^m \frac{\alpha^i p^i}{\gamma^i - 1}, \quad \frac{\gamma p_\infty}{\gamma - 1} = \sum_{i=1}^m \frac{\alpha^i \gamma^i p_\infty^i}{\gamma^i - 1}$$

and the overall set of equations [Shyue, 1998]

$$\partial_t \rho + \partial_{x_n} (\rho u_n) = 0, \quad \partial_t (\rho u_k) + \partial_{x_n} (\rho u_k u_n + \delta_{kn} p) = 0, \quad \partial_t (\rho E) + \partial_{x_n} (\rho u_n (E + p)) = 0$$

$$\frac{\partial}{\partial t} \left(\frac{1}{\gamma - 1} \right) + u_n \frac{\partial}{\partial x_n} \left(\frac{1}{\gamma - 1} \right) = 0, \quad \frac{\partial}{\partial t} \left(\frac{\gamma p_\infty}{\gamma - 1} \right) + u_n \frac{\partial}{\partial x_n} \left(\frac{\gamma p_\infty}{\gamma - 1} \right) = 0$$

Oscillation free at contacts: [Abgrall and Karni, 2001][Shyue, 2006]

Approximate Riemann solver

Use HLLC approach because of robustness and positivity preservation

$$\mathbf{q}^{HLLC}(x_1, t) = \begin{cases} \mathbf{q}_L^*, & x_1 < s_L t, \\ \mathbf{q}_L^*, & s_L t \leq x_1 < s^* t, \\ \mathbf{q}_R^*, & s^* t \leq x_1 \leq s_R t, \\ \mathbf{q}_R^*, & x_1 > s_R t, \end{cases}$$

Wave speed estimates [Davis, 1988] $s_L = \min\{u_{1,L} - c_L, u_{1,R} - c_R\}$,

$s_R = \max\{u_{1,L} + c_L, u_{1,R} + c_R\}$

Unknown state [Toro et al., 1994]

$$s^* = \frac{p_R - p_L + s_L u_{1,L} (s_L - u_{1,L}) - \rho_R u_{1,R} (s_R - u_{1,R})}{\rho_L (s_L - u_{1,L}) - \rho_R (s_R - u_{1,R})}$$

$$\mathbf{q}_\tau^* = \left[\eta, \eta s^*, \eta u_2, \eta \left[\frac{(\rho E)_\tau}{\rho_\tau} + (s^* - u_{1,\tau}) \left(s_\tau + \frac{p_\tau}{\rho_\tau (s_\tau - u_{1,\tau})} \right) \right], \frac{1}{\gamma_\tau - 1}, \frac{\gamma_\tau p_\infty, \tau}{\gamma_\tau - 1} \right]^T$$

$$\eta = \rho_\tau \frac{s_\tau - u_{1,\tau}}{s_\tau - s^*}, \quad \tau = \{L, R\}$$

Evaluate waves as $\mathcal{W}_1 = \mathbf{q}_L^* - \mathbf{q}_L$, $\mathcal{W}_2 = \mathbf{q}_R^* - \mathbf{q}_R$, $\mathcal{W}_3 = \mathbf{q}_R - \mathbf{q}_R^*$ and $\lambda_1 = s_L$,

$\lambda_2 = s^*$, $\lambda_3 = s_R$ to compute the fluctuations $\mathcal{A}^- \Delta = \sum_{\lambda_\nu < 0} \lambda_\nu \mathcal{W}_\nu$,

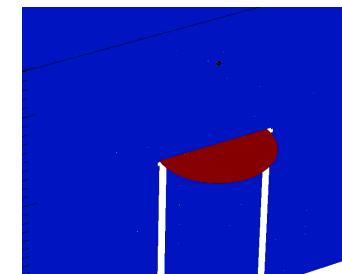
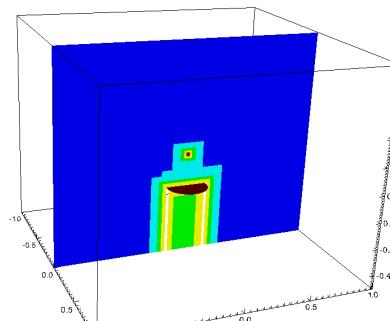
$\mathcal{A}^+ \Delta = \sum_{\lambda_\nu \geq 0} \lambda_\nu \mathcal{W}_\nu$ for $\nu = \{1, 2, 3\}$

Overall scheme: Wave Propagation method [Shyue, 2006]

Underwater explosion simulation

- AMR base grid $50 \times 40 \times 50$, $r_{1,2,3} = 2$, $r_4 = 4$, $l_c = 3$, highest level restricted to initial explosion center, 3rd and 4th level to plate vicinity
- Triangular mesh with 8148 elements
- Computations of 1296 coupled time steps to $t_{end} = 1$ ms
- 10+2 nodes 3.4 GHz Intel Xeon dual processor, ~ 130 h CPU

Maximal deflection [mm]		
	Exp.	Sim.
20 g, $d = 25$ cm	28.83	25.88
30 g, $d = 30$ cm	30.09	27.31



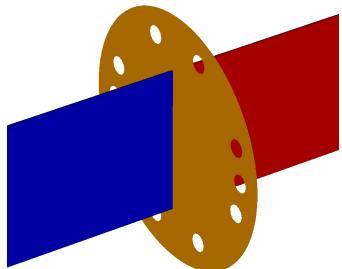
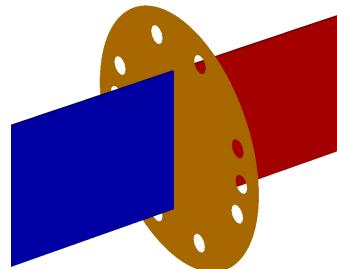
Underwater explosion FSI simulations

- Air: $\gamma^A = 1.4$, $p_\infty^A = 0$, $\rho^A = 1.29 \text{ kg/m}^3$
- Water: $\gamma^W = 7.415$, $p_\infty^W = 296.2 \text{ MPa}$, $\rho^W = 1027 \text{ kg/m}^3$
- Cavitation modeling with pressure cut-off model at $p = -1 \text{ MPa}$
- 3D simulation of deformation of air backed aluminum plate with $r = 85 \text{ mm}$, $h = 3 \text{ mm}$ from underwater explosion
 - Water basin [Ashani and Ghamsari, 2008] $2 \text{ m} \times 1.6 \text{ m} \times 2 \text{ m}$
 - Explosion modeled as energy increase ($m_{C4} \cdot 6.06 \text{ MJ/kg}$) in sphere with $r=5\text{mm}$
 - $\rho_s = 2719 \text{ kg/m}^3$, $E = 69 \text{ GPa}$, $\nu = 0.33$, J2 plasticity model, yield stress $\sigma_y = 217.6 \text{ MPa}$
- 3D simulation of copper plate $r = 32 \text{ mm}$, $h = 0.25 \text{ mm}$ rupturing due to water hammer
 - Water-filled shocktube 1.3 m with driver piston [Deshpande et al., 2006]
 - Piston simulated with separate level set, see [Deiterding et al., 2009] for pressure wave
 - $\rho_s = 8920 \text{ kg/m}^3$, $E = 130 \text{ GPa}$, $\nu = 0.31$, J2 plasticity model, $\sigma_y = 38.5 \text{ MPa}$, cohesive interface model, max. tensile stress $\sigma_c = 525 \text{ MPa}$

Plate in underwater shocktube

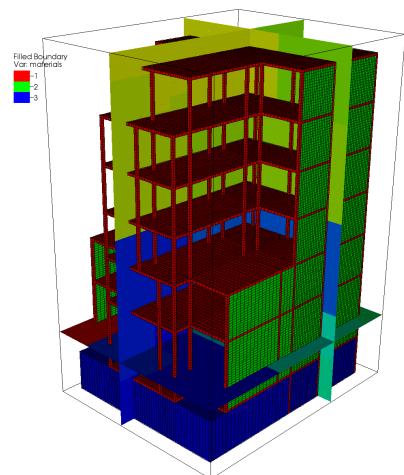
- AMR base mesh $374 \times 20 \times 20$, $r_{1,2} = 2$, $l_c = 2$, solid mesh: 8896 triangles
- ~ 1250 coupled time steps to $t_{end} = 1 \text{ ms}$
- 6+6 nodes 3.4 GHz Intel Xeon dual processor, ~ 800 h CPU

code/doc/html/capps/sfc-amroc_2WaterBlastFracture_2src_2FluidProblem_8h_source.html,
code/doc/html/capps/WaterBlastFracture_2src_2ShellManagerSpecific_8h_source.html



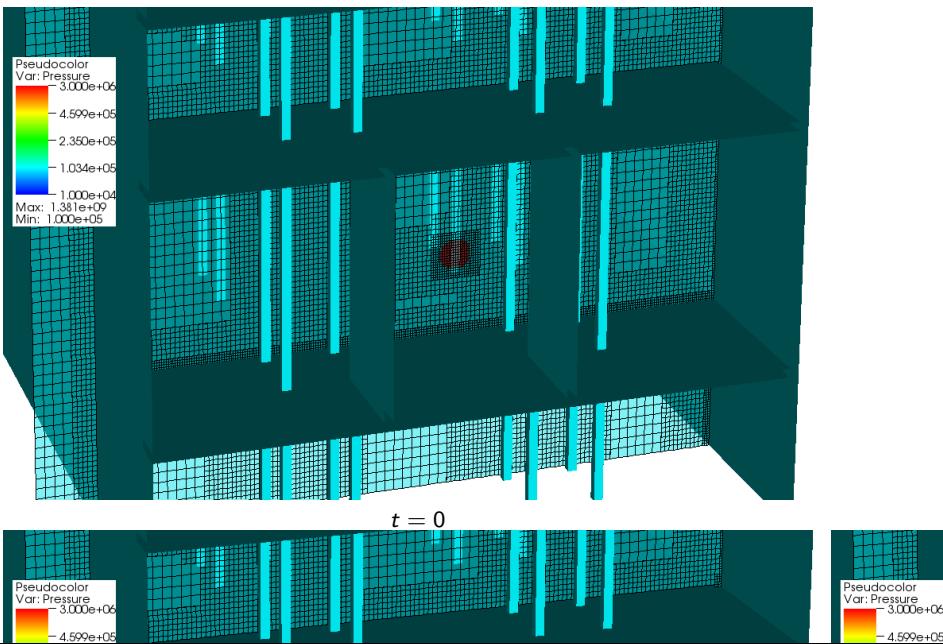
Blast explosion in a multistory building

- 20 m × 40 m × 25 m seven-story building similar to [Luccioni et al., 2004]
- Spherical energy deposition ≡ 400 kg TNT, $r = 0.5$ m in lobby of building
- SAMR: 80 × 120 × 90 base level, three additional levels $r_{1,2} = 2$, $l_{fsi} = 1$, $k = 1$
- Simulation with ground: 1,070 coupled time steps, 830 h CPU (~ 25.9 h wall time) on 31+1 cores
- ~ 8,000,000 cells instead of 55,296,000 (uniform)
- 69,709 hexahedral elements and with material parameters. [Deiterding and Wood, 2013]



	ρ_s [kg/m ³]	σ_0 [MPa]	E_T [GPa]	β	K [GPa]	G [GPa]	$\bar{\epsilon}^p$	p_f [MPa]
Columns	2010	50	11.2	1.0	21.72	4.67	0.02	-30
Walls	2010	25	11.2	1.0	6.22	4.67	0.01	-15

Blast explosion in a multistory building – II



Outline

Fluid-structure interaction

- Coupling to a solid mechanics solver
- Implementation
- Rigid body motion
- Thin elastic and deforming thin structures
- Deformation from water hammer
- Real-world example

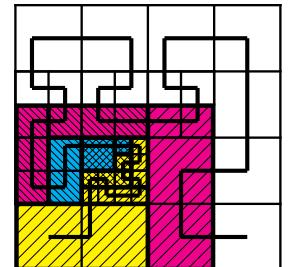
Massively parallel SAMR

- Performance data from AMROC

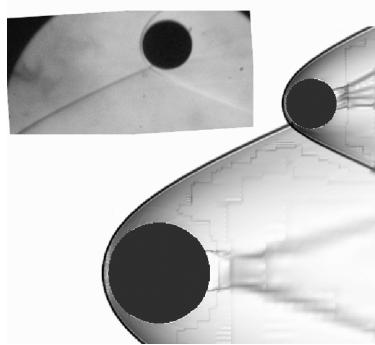
Parallelized construction of space-filling curve

Computation of space filling curve

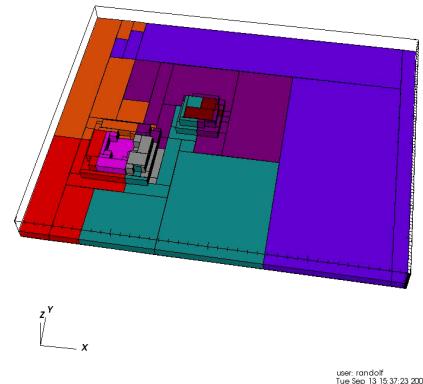
- Partition-Init
 - Compute aggregated workload for new grid hierarchy and project result onto level 0
 - Construct recursively SFC-units until work in each unit is homogeneous, GuCFactor defines minimal coarseness relative to level-0 grid
- Partition-Calc
 - Compute entire workload and new work for each processor
 - Go sequentially through SFC-ordered list of partitioning units and assign units to processors, refine partition if necessary and possible
- Ensure scalability of Partition-Init by creating SFC-units strictly local
- Currently still use of MPI_allgather() to create globally identical input for Partition-Calc (can be a bottleneck for weak scalability)



Partitioning example



DB: trace8_vtk



- ▶ Cylinders of spheres in supersonic flow
- ▶ Predict force on secondary body
- ▶ Right: 200x160 base mesh, 3 Levels, factors 2,2,2, 8 CPUs

[Laurence et al., 2007]

Cost of SAMR and ghost-fluid method

- ▶ Flux correction is negligible
- ▶ Clustering is negligible (already local approach). For the complexities of a scalable global clustering algorithm see [Gunney et al., 2007]
- ▶ Costs for GFM constant around $\sim 36\%$
- ▶ Main costs: Regrid(1) operation and ghost cell synchronization

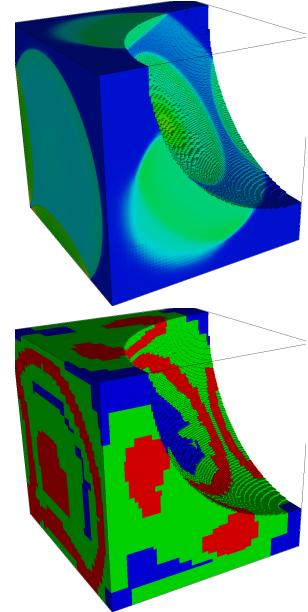
CPUs	16	32	64
Time per step	32.44s	18.63s	11.87s
Uniform	59.65s	29.70s	15.15s
Integration	73.46%	64.69%	50.44%
Flux Correction	1.30%	1.49%	2.03%
Boundary Setting	13.72%	16.60%	20.44%
Regridding	10.43%	15.68%	24.25%
Clustering	0.34%	0.32%	0.26%
Output	0.29%	0.53%	0.92%
Misc.	0.46%	0.44%	0.47%
CPUs	16	32	64
Time per step	43.97s	25.24s	16.21s
Uniform	69.09s	35.94s	18.24s
Integration	59.09%	49.93%	40.20%
Flux Correction	0.82%	0.80%	1.14%
Boundary Setting	19.22%	25.58%	28.98%
Regridding	7.21%	9.15%	13.46%
Clustering	0.25%	0.23%	0.21%
GFM Find Cells	2.04%	1.73%	1.38%
GFM Interpolation	6.01%	10.39%	7.92%
GFM Overhead	0.54%	0.47%	0.37%
GFM Calculate	0.70%	0.60%	0.48%
Output	0.23%	0.52%	0.74%
Misc.	0.68%	0.62%	0.58%

First performance assessment

- ▶ Test run on 2.2 GHz AMD Opteron quad-core cluster connected with Infiniband
- ▶ Cartesian test configuration
- ▶ Spherical blast wave, Euler equations, 3rd order WENO scheme, 3-step Runge-Kutta update
- ▶ AMR base grid 64^3 , $r_{1,2} = 2$, 89 time steps on coarsest level
- ▶ With embedded boundary method: 96 time steps on coarsest level
- ▶ Redistribute in parallel every 2nd base level step
- ▶ Uniform grid $256^3 = 16.8 \cdot 10^6$ cells

Level	Grids	Cells
0	115	262,144
1	373	1,589,808
2	2282	5,907,064

Grid and cells used on 16 CPUs



AMROC scalability tests

Basic test configuration

- ▶ Spherical blast wave, Euler equations, 3D wave propagation method
- ▶ AMR base grid 32^3 with $r_{1,2} = 2, 4$. 5 time steps on coarsest level
- ▶ Uniform grid $256^3 = 16.8 \cdot 10^6$ cells, 19 time steps
- ▶ Flux correction deactivated
- ▶ No volume I/O operations
- ▶ Tests run IBM BG/P (mode VN)

Weak scalability test

- ▶ Reproduction of configuration each 64 CPUs
- ▶ On 1024 CPUs: $128 \times 64 \times 64$ base grid, $> 33,500$ Grids, $\sim 61 \cdot 10^6$ cells, uniform $1024 \times 512 \times 512 = 268 \cdot 10^6$ cells

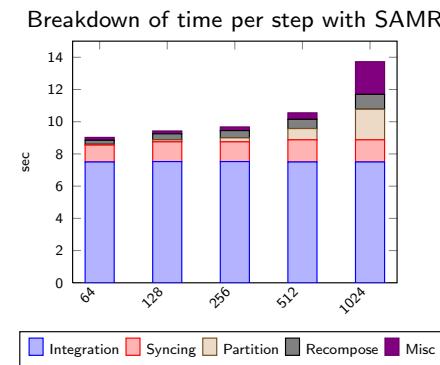
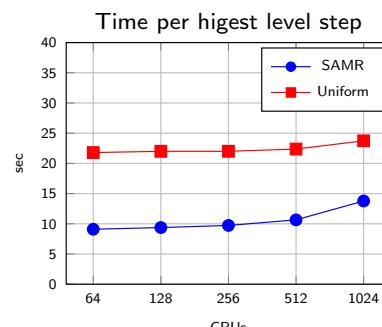
Level	Grids	Cells
0	606	32,768
1	575	135,312
2	910	3,639,040

Strong scalability test

- ▶ $64 \times 32 \times 32$ base grid, uniform $512 \times 256 \times 256 = 33.6 \cdot 10^6$ cells

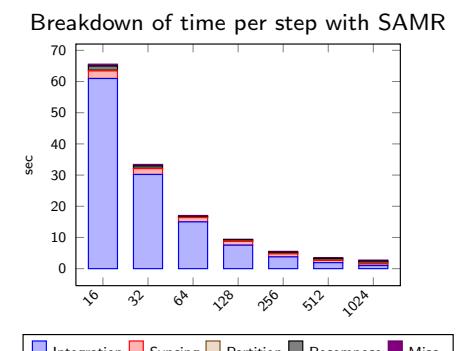
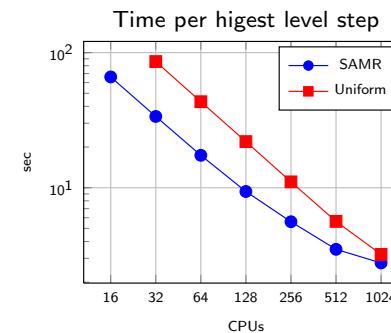
Level	Grids	Cells
0	1709	65,536
1	1735	271,048
2	2210	7,190,208

Weak scalability test



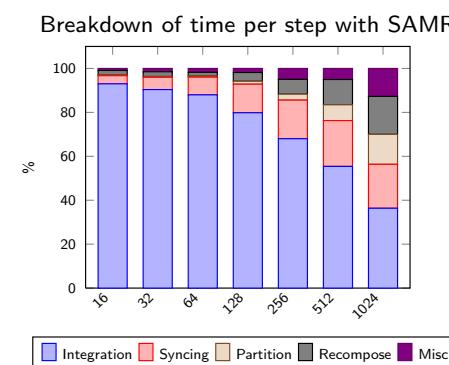
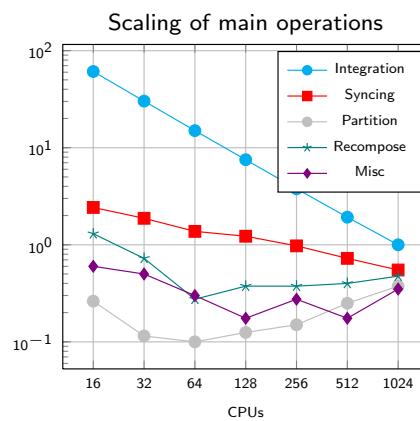
- ▶ Costs for Syncing basically constant
- ▶ Partitioning, Recompose, Misc (origin not clear) increase
- ▶ 1024 required usage of -DUAL option due to usage of global lists data structures in Partition-Calc and Recompose

Strong scalability test



- ▶ Uniform code has basically linear scalability (explicit method)
- ▶ SAMR visibly loses efficiency for > 512 CPU, or 15,000 finite volume cells per CPU

Strong scalability test - II



- ▶ Perfect scaling of Integration, reasonable scaling of Syncing
- ▶ Strong scalability of Partition needs to be addressed (eliminate global lists)

References I

- [Abgrall and Karni, 2001] Abgrall, R. and Karni, S. (2001). Computations of compressible multifluids. *J. Comput. Phys.*, 169:594–523.
- [Arienti et al., 2003] Arienti, M., Hung, P., Morano, E., and Shepherd, J. E. (2003). A level set approach to Eulerian-Lagrangian coupling. *J. Comput. Phys.*, 185:213–251.
- [Ashani and Ghamsari, 2008] Ashani, J. Z. and Ghamsari, A. K. (2008). Theoretical and experimental analysis of plastic response of isotropic circular plates subjected to underwater explosion loading. *Mat.-wiss. u. Werkstofftechn.*, 39(2):171–175.
- [Cirak et al., 2007] Cirak, F., Deiterding, R., and Mauch, S. P. (2007). Large-scale fluid-structure interaction simulation of viscoplastic and fracturing thin shells subjected to shocks and detonations. *Computers & Structures*, 85(11-14):1049–1065.
- [Davis, 1988] Davis, S. F. (1988). Simplified second-order Godunov-type methods. *SIAM J. Sci. Stat. Comp.*, 9:445–473.

References II

[Deiterding et al., 2009] Deiterding, R., Cirak, F., and Mauch, S. P. (2009). Efficient fluid-structure interaction simulation of viscoplastic and fracturing thin-shells subjected to underwater shock loading. In Hartmann, S., Meister, A., Schäfer, M., and Turek, S., editors, *Int. Workshop on Fluid-Structure Interaction. Theory, Numerics and Applications, Herrsching am Ammersee 2008*, pages 65–80. kassel university press GmbH.

[Deiterding and Wood, 2013] Deiterding, R. and Wood, S. L. (2013). Parallel adaptive fluid-structure interaction simulations of explosions impacting on building structures. *Computers & Fluids*, 88:719–729.

[Deshpande et al., 2006] Deshpande, V. S., Heaver, A., and Fleck, N. A. (2006). An underwater shock simulator. *Royal Society of London Proceedings Series A*, 462(2067):1021–1041.

[Falcovitz et al., 1997] Falcovitz, J., Alfandary, G., and Hanoch, G. (1997). A two-dimensional conservation laws scheme for compressible flows with moving boundaries. *J. Comput. Phys.*, 138:83–102.

References III

[Fedkiw, 2002] Fedkiw, R. P. (2002). Coupling an Eulerian fluid calculation to a Lagrangian solid calculation with the ghost fluid method. *J. Comput. Phys.*, 175:200–224.

[Giordano et al., 2005] Giordano, J., Jourdan, G., Burtschell, Y., Medale, M., Zeitoun, D. E., and Houas, L. (2005). Shock wave impacts on deforming panel, an application of fluid-structure interaction. *Shock Waves*, 14(1-2):103–110.

[Gunney et al., 2007] Gunney, B. T., Wissink, A. M., and Hysoma, D. A. (2007). Parallel clustering algorithms for structured AMR. *J. Parallel and Distributed Computing*, 66(11):1419–1430.

[Laurence and Deiterding, 2011] Laurence, S. J. and Deiterding, R. (2011). Shock-wave surfing. *J. Fluid Mech.*, 676:369–431.

[Laurence et al., 2007] Laurence, S. J., Deiterding, R., and Hornung, H. G. (2007). Proximal bodies in hypersonic flows. *J. Fluid Mech.*, 590:209–237.

[Luccioni et al., 2004] Luccioni, B. M., Ambrosini, R. D., and Danesi, R. F. (2004). Analysis of building collapse under blast loads. *Engineering & Structures*, 26:63–71.

References IV

[Mader, 1979] Mader, C. L. (1979). *Numerical modeling of detonations*. University of California Press, Berkeley and Los Angeles, California.

[Mauch, 2003] Mauch, S. P. (2003). *Efficient Algorithms for Solving Static Hamilton-Jacobi Equations*. PhD thesis, California Institute of Technology.

[Shyue, 1998] Shyue, K.-M. (1998). An efficient shock-capturing algorithm for compressible multicomponent problems. *J. Comput. Phys.*, 142:208–242.

[Shyue, 2006] Shyue, K.-M. (2006). A volume-fraction based algorithm for hybrid barotropic and non-barotropic two-fluid flow problems. *Shock Waves*, 15:407–423.

[Specht, 2000] Specht, U. (2000). *Numerische Simulation mechanischer Wellen an Fluid-Festkörper-Mediengrenzen*. Number 398 in VDI Reihe 7. VDU Verlag, Düsseldorf.

[Toro et al., 1994] Toro, E. F., Spruce, M., and Speares, W. (1994). Restoration of the contact surface in the HLL-Riemann solver. *Shock Waves*, 4:25–34.

Lecture 7 Lattice Boltzmann methods

Course *Block-structured Adaptive Finite Volume Methods in C++*

Ralf Deiterding

University of Southampton
Engineering and the Environment
Highfield Campus, Southampton SO17 1BJ, UK

E-mail: r.deiterding@soton.ac.uk

Outline

Adaptive lattice Boltzmann method

- Construction principles
- Adaptive mesh refinement for LBM
- Implementation
- Verification

Realistic aerodynamics computations

- Vehicle geometries

Fluid-structure coupling

- Rigid body dynamics
- Validation simulations

Wind turbine wake aerodynamics

- Mexico benchmark
- Simulation of wind turbine wakes
- Wake interaction prediction

Approximation of Boltzmann equation

Is based on solving the Boltzmann equation with a simplified collision operator

$$\partial_t f + \mathbf{u} \cdot \nabla f = \omega(f^{eq} - f)$$

- Kn = $I_f/L \ll 1$, where I_f is replaced with Δx
- Weak compressibility and small Mach number assumed
- Assume a simplified phase space

Equation is approximated with a splitting approach.

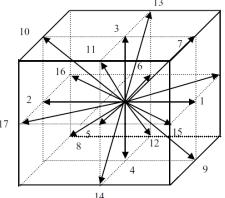
1.) Transport step solves $\partial_t f_\alpha + \mathbf{e}_\alpha \cdot \nabla f_\alpha = 0$

Operator: $\mathcal{T}: \tilde{f}_\alpha(\mathbf{x} + \mathbf{e}_\alpha \Delta t, t + \Delta t) = f_\alpha(\mathbf{x}, t)$

$$\rho(\mathbf{x}, t) = \sum_{\alpha=0}^{18} f_\alpha(\mathbf{x}, t), \quad \rho(\mathbf{x}, t) u_i(\mathbf{x}, t) = \sum_{\alpha=0}^{18} \mathbf{e}_{\alpha i} f_\alpha(\mathbf{x}, t)$$

Discrete velocities:

$$\mathbf{e}_\alpha = \begin{cases} 0, & \alpha = 0, \\ (\pm 1, 0, 0)c, (0, \pm 1, 0)c, (0, 0, \pm 1)c, & \alpha = 1, \dots, 6, \\ (\pm 1, \pm 1, 0)c, (\pm 1, 0, \pm 1)c, (0, \pm 1, \pm 1)c, & \alpha = 7, \dots, 18, \end{cases}$$



Outline

Adaptive lattice Boltzmann method

- Construction principles
- Adaptive mesh refinement for LBM
- Implementation
- Verification

Realistic aerodynamics computations

- Vehicle geometries

Fluid-structure coupling

- Rigid body dynamics
- Validation simulations

Wind turbine wake aerodynamics

- Mexico benchmark
- Simulation of wind turbine wakes
- Wake interaction prediction

Classes

Directory `amroc/lbm` contains the lattice Boltzmann integrator that is in C++ throughout and also is built on the classes in `amroc/amr/Interfaces`.

- ▶ Several SchemeType-classes are already provided: `LBMD1Q3<DataType>`, `LBMD2Q9<DataType>`, `LBMD3Q19<DataType>`, `LBMD2Q9Thermal<DataType>`, `LBMD3Q19Thermal<DataType>` included a large number of boundary conditions.

`code/amroc/doc/html/lbm/classLBMD1Q3.html` `code/amroc/doc/html/lbm/classLBMD2Q9.html`

`code/amroc/doc/html/lbm/classLBMD3Q19Thermal.html`

- ▶ Using function within `LBMD?D?`, the special coarse-fine correction is implemented in `LBMFixup<LBMTyp, FixupTyp, dim>`

`code/amroc/doc/html/lbm/classLBMFixup.html`

- ▶ `LBMIntegrator<LBMTyp, dim>`, `LBMGFMBoundary<LBMTyp, dim>`, etc. interface to the generic classes in `amroc/amr/Interfaces`

`code/amroc/doc/html/amr/classSchemeGFBoundary.html`

- ▶ **Problem.h:** Specific simulation is defined in `Problem.h` only. Predefined classes specified in `LBMStdProblem.h`, `LBMStdGFMProblem.h` and `LBMProblem.h`.

`code/amroc/doc/html/lbm/LBMProblem_8h_source.html` `code/amroc/doc/html/lbm/LBMStdProblem_8h.html`

`code/amroc/doc/html/lbm/LBMStdGFMProblem_8h.html`

Flow over cylinder in 2d - $Re = 300$, $u = 0.2415 \text{ m/s}$

Isolines on refinement and distribution to processors



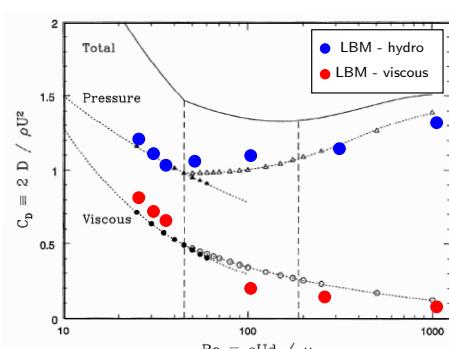
Mesh adaptation with LBM:

1. Level-wise evaluation of $\omega_L^I = \frac{c_s^2}{\nu + \Delta t/c_s^2/2}$
2. Exchange of distributions streaming across refinement interfaces

`code/amroc/doc/html/apps/lbm_2applications_2Navier-Stokes_22d_2CylinderDrag_2src_2Problem_8h_source.html`

Flow over 2D cylinder, $d = 2 \text{ cm}$

- ▶ Air with
 $\nu = 1.61 \cdot 10^{-5} \text{ m}^2/\text{s}$,
 $\rho = 1.205 \text{ kg/m}^3$
- ▶ Domain size
 $[-8d, 24d] \times [-8d, 8d]$
- ▶ Dynamic refinement based on velocity. Last level to refine structure further.
- ▶ Inflow from left.
Characteristic boundary conditions [Schlaffer, 2013] elsewhere.
- ▶ Base lattice 320×160 , 3 additional levels with factors $r_I = 2, 4, 4$.
- ▶ Resolution: ~ 320 points in diameter d
- ▶ Computation of C_D on 400 equidistant points along circle and averaged over time. Comparison above with [Henderson, 1995].



Outline

Adaptive lattice Boltzmann method

- Construction principles
- Adaptive mesh refinement for LBM
- Implementation
- Verification

Realistic aerodynamics computations

- Vehicle geometries

Fluid-structure coupling

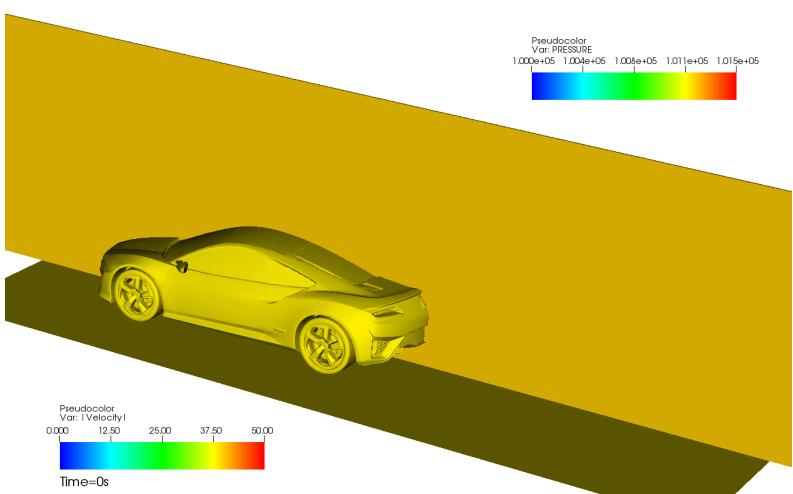
- Rigid body dynamics
- Validation simulations

Wind turbine wake aerodynamics

- Mexico benchmark
- Simulation of wind turbine wakes
- Wake interaction prediction

Wind tunnel simulation of a prototype car

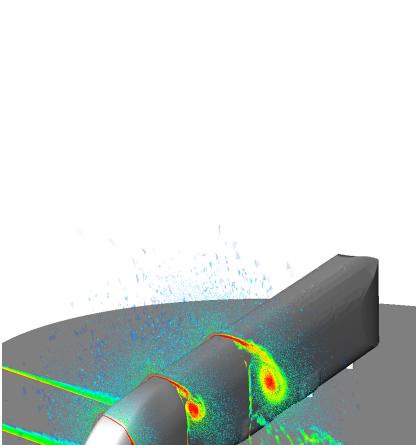
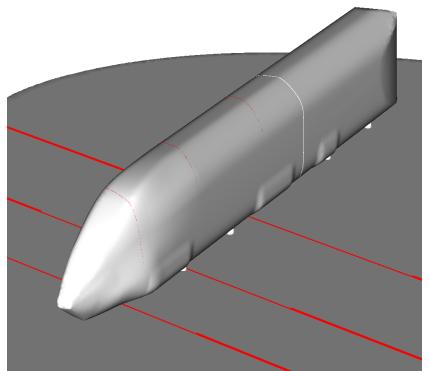
Fluid velocity and pressure on vehicle



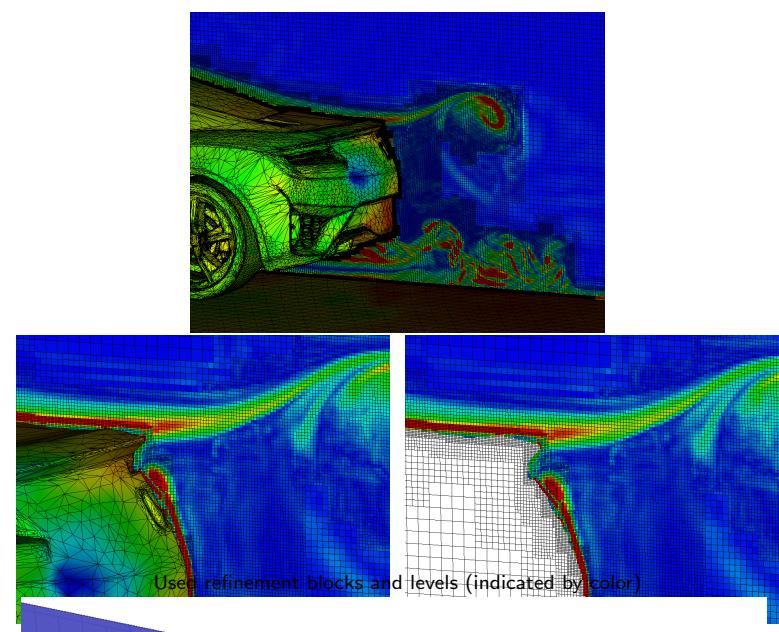
- Inflow 40 m/s. LES model active. Characteristic boundary conditions.
- To $t = 0.5$ s (~ 4 characteristic lengths) with 31,416 time steps on finest level in ~ 37 h on 200 cores (7389 h CPU). Channel: 15 m \times 5 m \times 3.3 m

Next Generation Train (NGT)

- 1:25 train model of 74,670 triangles
- Wind tunnel: air at room temperature with 33.48 m/s, $Re = 250,000$, yaw angle 30°
- Comparison between LBM (fluid air) and incompressible OpenFOAM solvers



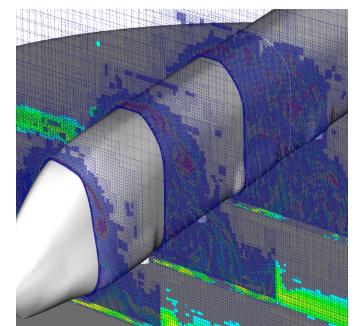
Mesh adaptation



NGT model

- LBM-AMR computation with 5 additional levels, factor 2 refinement (uniform: 120.4e9 cells)
- Dynamic AMR until $T_c = 34$, then static for $\sim 12 T_c$ to obtain average coefficients
- OpenFOAM simulations by M. Fragner (DLR)

Simulation	Mesh	CFX	CFY	CMX
Wind tunnel	—	-0.06	-5.28	-3.46
DDES	low	-0.40	-5.45	-3.61
Σ only	low	0.10	-0.04	-0.05
LES	high	-0.45	-6.07	-4.14
DDES	high	-0.43	-5.72	-3.77
LBM - p only	—	-0.30	-5.09	-3.46

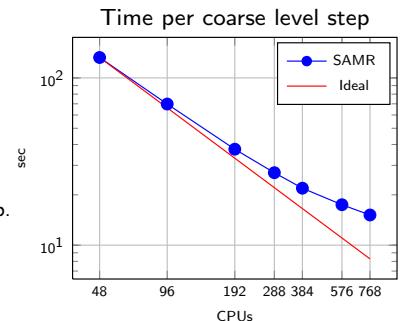


	LBM	DDES(I)	LES	DDES(h)
Cells	147M	34.1M	219M	219M
y^+	43	3.2	1.7	1.7
x^+, z^+	43	313	140	140
Δx wake [mm]	0.936	3.0	1.5	1.5
Runtime [T_c]	34	35.7	10.3	9.2
Processors	200	80	280	280
CPU [h]	34,680	49,732	194,483	164,472
$T_c/\Delta t$	1790	1325	1695	1695
CPU [h]/ $T_c/1M$ cells	5.61	39.75	86.4	81.36

Adaptive LBM code 16x faster than OpenFOAM with PISO algorithm on static mesh!

Strong scalability test (1:25 train)

- Computation is restarted from disk checkpoint at $t = 0.526408$ s from 96 core run.
- Time for initial re-partitioning removed from benchmark.
- 200 coarse level time steps computed.
- Regridding and re-partitioning every 2nd level-0 step.
- Computation starts with 51.8M cells ($I_3: 10.2M, I_2: 15.3M, I_1: 21.5M, I_0 = 4.8M$) vs. 19.66 billion (uniform).
- Portions for parallel communication quite considerable (4 ghost cells still used).



Time in % spent in main operations

Cores	48	96	192	288	384	576	768
Time per step	132.43s	69.79s	37.47s	27.12s	21.91s	17.45s	15.15s
Par. Efficiency	100.0	94.88	88.36	81.40	75.56	63.24	54.63
LBM Update	5.91	5.61	5.38	4.92	4.50	3.73	3.19
Regridding	15.44	12.02	11.38	10.92	10.02	8.94	8.24
Partitioning	4.16	2.43	1.16	1.02	1.04	1.16	1.34
Interpolation	3.76	3.53	3.33	3.05	2.83	2.37	2.06
Sync Boundaries	54.71	59.35	59.73	56.95	54.54	52.01	51.19
Sync Fixup	9.10	10.41	12.25	16.62	20.77	26.17	28.87
Level set	0.78	0.93	1.21	1.37	1.45	1.48	1.47
Interp./Extrap.	3.87	3.81	3.76	3.49	3.26	2.75	2.39
Misc	2.27	1.91	1.79	1.67	1.58	1.38	1.25

Two-segment hinged wing

Configuration by [Toomey and Eldredge, 2008].

Manufactured bodies in tank filled with water.

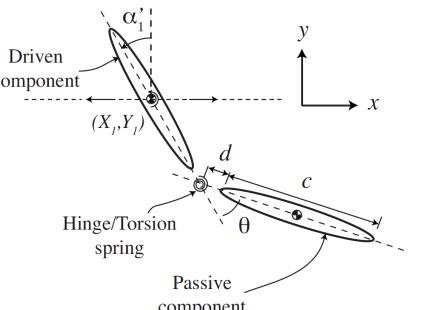
Prescribed translation and rotation

$$X_t(t) = \frac{A_0}{2} \frac{G_t(\text{ft})}{\max G_t} C(\text{ft}), \quad \alpha_1(t) = -\beta \frac{G_r(\text{ft})}{\max G_r}$$

with $G_r(t) = \tanh[\sigma_r \cos(2\pi t + \Phi)]$,

$$G_t(t) = \int_t \tanh[\sigma_t \cos(2\pi t')] dt'$$

- 7 cases constructed by varying σ_r, σ_t, Φ
- Rotational Reynolds number
 $Re_r = 2\pi\beta\sigma_r f c^2 / (\tanh(\sigma_r)\nu)$ varied between 2200 and 7200 in experiments
- [Toomey and Eldredge, 2008] reference simulations with a viscous particle method are for $Re_r = \{100, 500\}$



A_0 (cm)	7.1
c (cm)	5.1
d (cm)	0.25
ρ_b (kg/m ³)	5080
f (Hz)	0.15

Motion solver

Based on the Newton-Euler method solution of dynamics equation of kinetic chains [Tsai, 1999]

$$\left(\frac{\mathbf{F}}{\tau_p} \right) = \begin{pmatrix} \mathbf{m}_1 & -\mathbf{m}[\mathbf{c}]^\times \\ \mathbf{m}[\mathbf{c}]^\times \mathbf{I}_{cm} & -\mathbf{m}[\mathbf{c}]^\times [\mathbf{c}]^\times \end{pmatrix} \begin{pmatrix} \mathbf{a}_p \\ \boldsymbol{\alpha} \end{pmatrix} + \begin{pmatrix} \mathbf{m}[\omega]^\times [\omega]^\times \mathbf{c} \\ [\omega]^\times (\mathbf{I}_{cm} - \mathbf{m}[\mathbf{c}]^\times [\mathbf{c}]^\times) \omega \end{pmatrix}.$$

m = mass of the body, 1 = the 4×4 homogeneous identity matrix,

\mathbf{a}_p = acceleration of link frame with origin at p in the preceding link's frame,

\mathbf{I}_{cm} = moment of inertia about the center of mass,

ω = angular velocity of the body,

$\boldsymbol{\alpha}$ = angular acceleration of the body,

\mathbf{c} is the location of the body's center of mass,

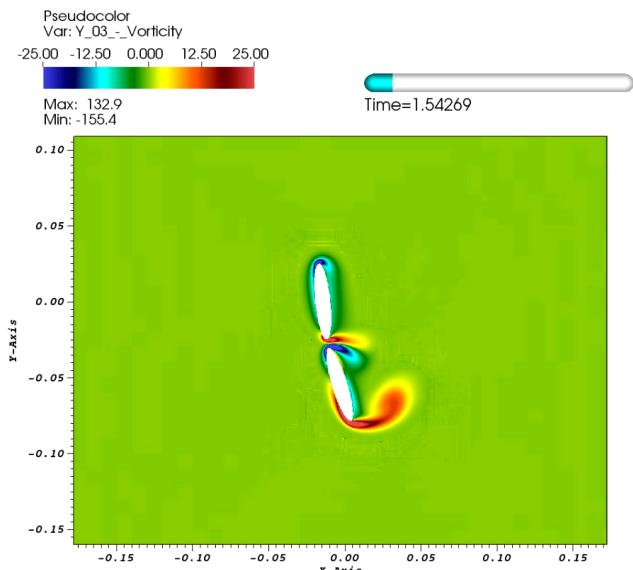
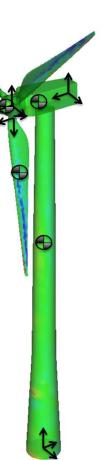
and $[\mathbf{c}]^\times, [\omega]^\times$ denote skew-symmetric cross product matrices.

Here, we additionally define the total force and torque acting on a body,

$$\mathbf{F} = (\mathbf{F}_{FSI} + \mathbf{F}_{prescribed}) \cdot \mathcal{C}_{xyz} \text{ and}$$

$$\tau = (\tau_{FSI} + \tau_{prescribed}) \cdot \mathcal{C}_{\alpha\beta\gamma} \text{ respectively.}$$

Where \mathcal{C}_{xyz} and $\mathcal{C}_{\alpha\beta\gamma}$ are the translational and rotational constraints, respectively.



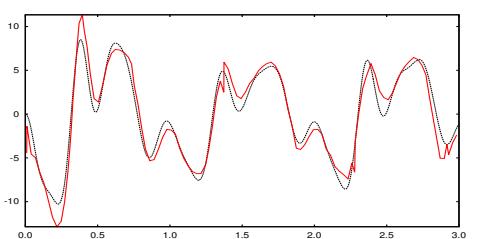
Right: computed vorticity field (enlarged)

Quantitative comparison

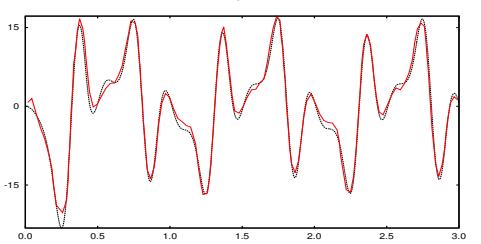
- Evaluate normalized force $F_{x,y} = 2F_{x,y}^*/(\rho_f c^3)$ and moment $M = 2M^*/(\rho_f f^2 c^4)$ over 3 periods
- [Wood and Deiterding, 2015] Used finest spatial resolution $\Delta x/c = 0.0122$
[Toomey and Eldredge, 2008]: $\Delta x/c = 0.013$ ($Re_r = 100$), $\Delta x/c = 0.0032$ ($Re_r = 500$)
- Temporal resolution ~ 113 and ~ 28 times finer

Hinge deflection angle over time

Case 1
 $\sigma_t = 0.628$
 $\sigma_r = 0.628$
 $\phi = 0$

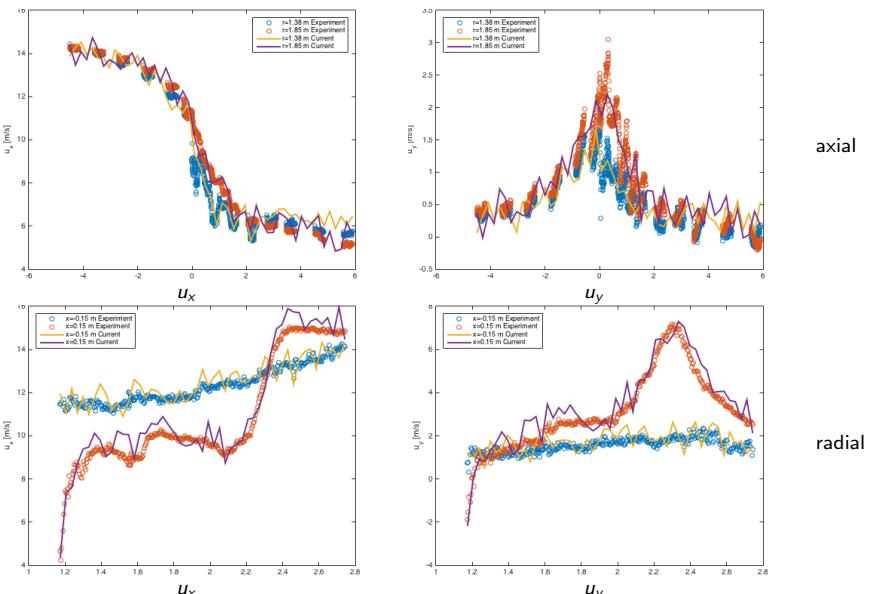


Case 2
 $\sigma_t = 1.885$
 $\sigma_r = 1.885$
 $\phi = 0^\circ$



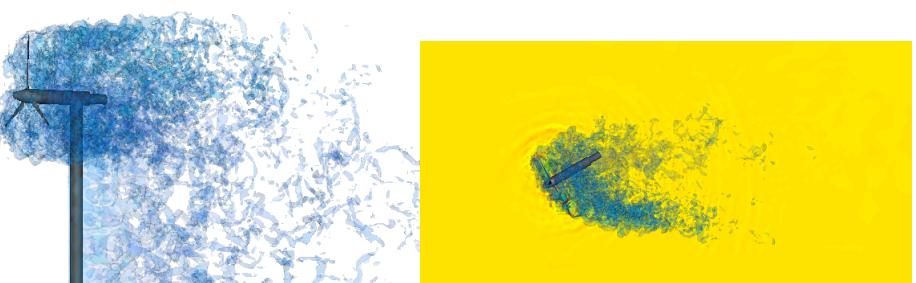
Experimental results (—);
 Current (---)

Comparison along transects – 0° inflow



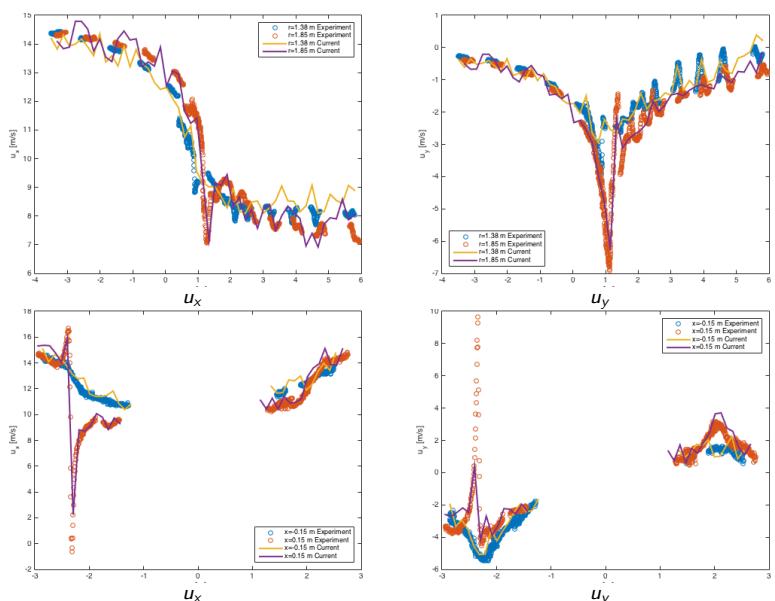
22

Mexico experimental turbine – 30° yaw



- Load collected as average during $t \in [5, 10]$ on blade 1 as it passes through $\theta = 0^\circ$ (pointing vertically upwards), 35 rotations
- Blade loads: F_x : Ref = 13.66 N, cur. = 14.8 N (8.3%)
 T_x : Ref = 7.72 Nm, cur. = 8.36 Nm (8.3%)
- Level 0: 768,000 cells
 Level 1: 1,524,826 cells
 Level 2: 6,832,602 cells
 Level 3: 3,019,205 cells

Comparison along transects – 30° yaw

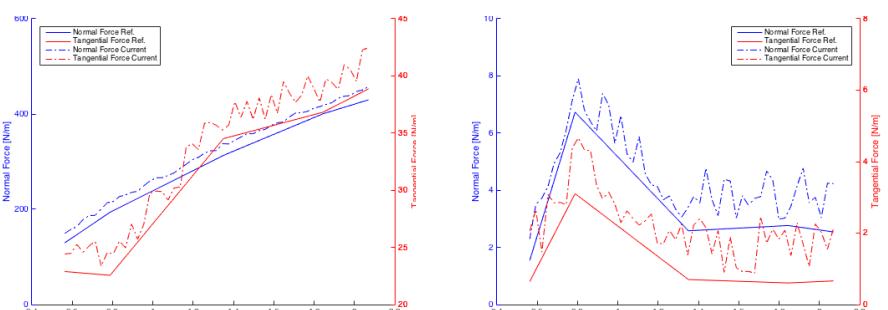


axial

radial

Normalized %-error along transects

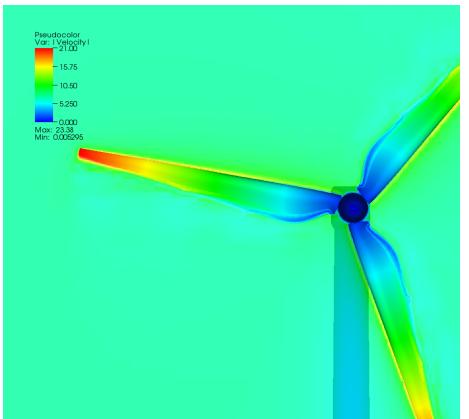
transect	0°		30°		
	yaw	in	out	in	out
Axial	u_x	6.416	7.663	5.742	6.410
	u_y	3.400	4.061	3.043	3.373
	u_z	3.073	3.678	2.752	3.068
Radial	up	6.556	7.325	7.093	6.655
	down	3.409	3.809	3.684	3.466
	up	3.242	3.659	3.511	3.294



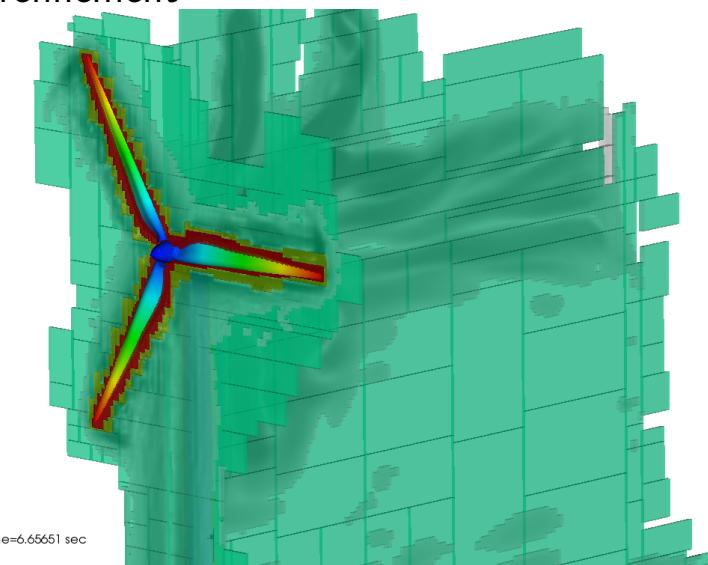
Comparison of normal and tangential forces on sections of blade 1 when $\theta_x = 0^\circ$ (pointing vertically upward) in aligned (left) and yaw 30°

Simulation of a single turbine

- Geometry from realistic Vestas V27 turbine. Rotor diameter 27 m, tower height ~ 35 m. Ground considered.
- Prescribed motion of rotor with 15 rpm. Inflow velocity 7 m/s.
- Simulation domain 200 m \times 100 m \times 100 m.
- Base mesh 400 \times 200 \times 200 cells with refinement factors 2,2,4. Resolution of rotor and tower $\Delta x = 3.125$ cm.
- 141,344 highest level iterations to $t_e = 30$ s computed.



Adaptive refinement



Dynamic evolution of refinement blocks (indicated by color).

[code/doc/html/capps/motion-amroc_2WindTurbine_Terrain_2src_2fluidProblem_8h_source.html](#),
[code/doc/html/capps/motion-amroc_2WindTurbine_Terrain_2src_2solidProblem_8h_source.html](#),
[code/doc/html/capps/Terrain_2src_2Terrain_8h_source.html](#)

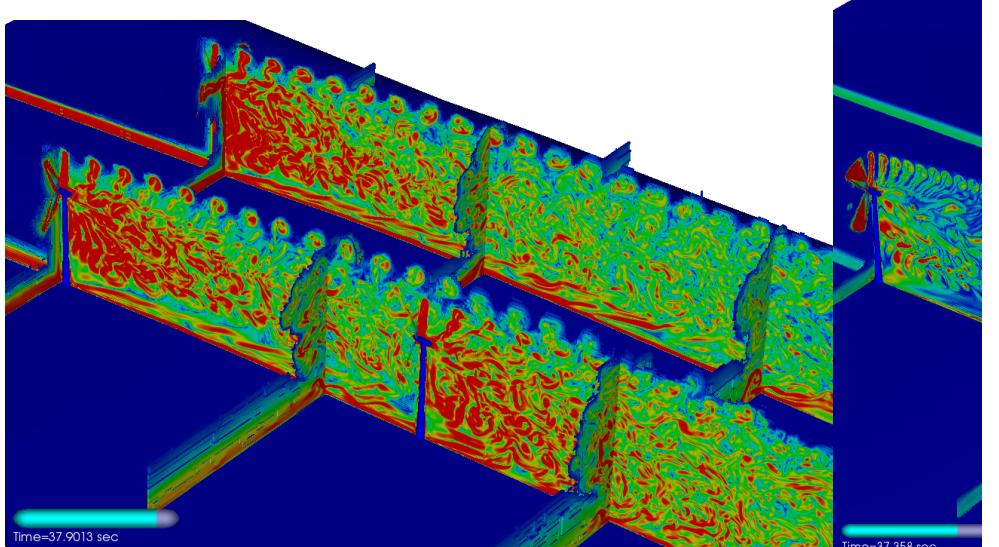
Simulation of the SWIFT array

- ▶ Three Vestas V27 turbines. 225 kW power generation at wind speeds 14 to 25 m/s (then cut-off)
- ▶ Prescribed motion of rotor with 33 and 43 rpm. Inflow velocity 8 and 25 m/s
- ▶ TSR: 5.84 and 2.43, $Re_r \approx 919,700$ and 1,208,000
- ▶ Simulation domain 448 m × 240 m × 100 m
- ▶ Base mesh 448 × 240 × 100 cells with refinement factors 2,2,4. Resolution of rotor and tower $\Delta x = 6.25$ cm
- ▶ 94,224 highest level iterations to $t_e = 40$ s computed, then statistics are gathered for 10 s [Deiterding and Wood, 2015]

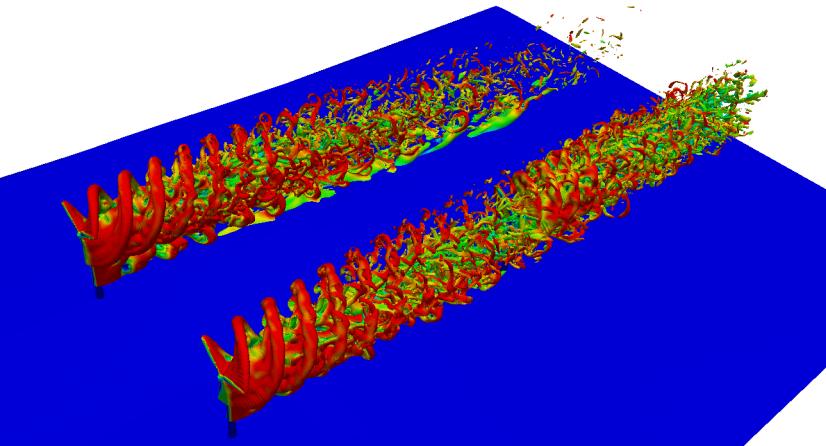


Vorticity generation - $u = 8$ m/s, 33 rpm

$u = 25$ m/s,
43 rpm

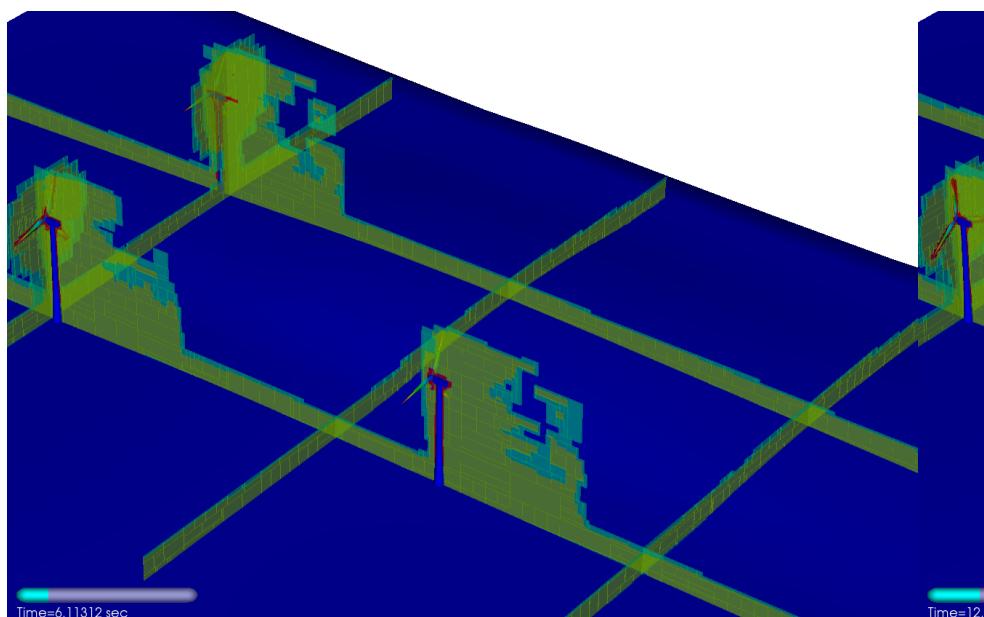


Wake propagation through array – 25 m/s, 43 rpm

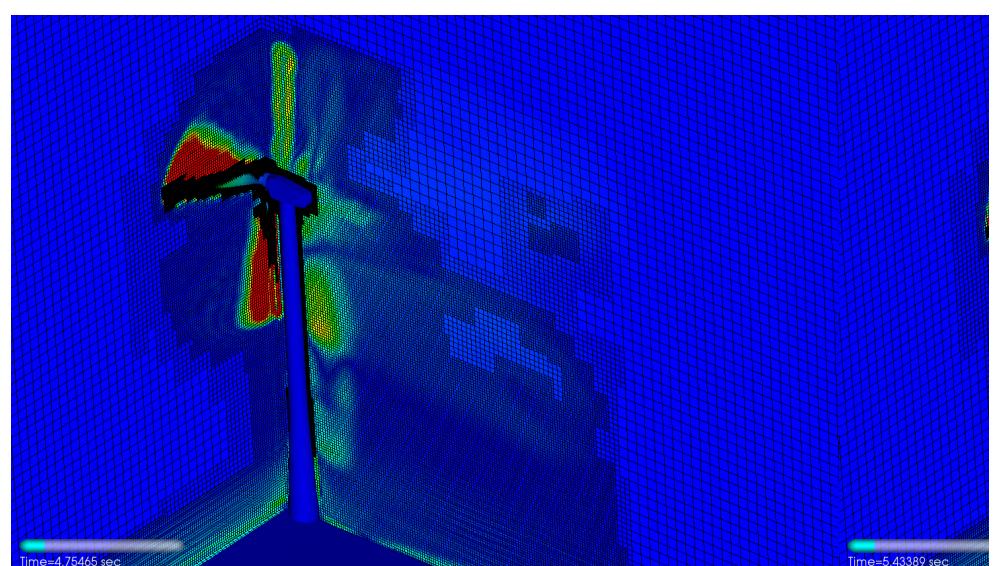
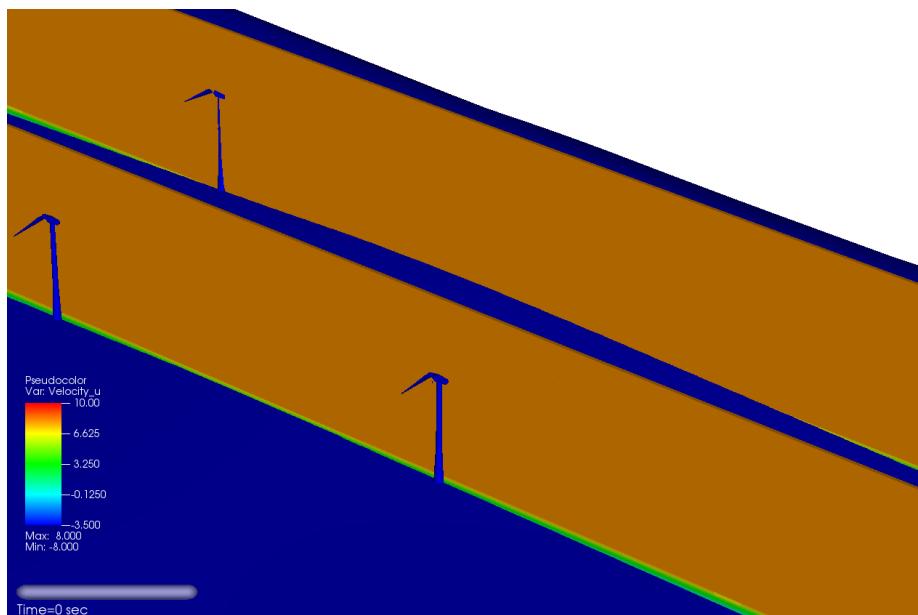


- ▶ On 288 cores Intel Xeon-Ivybridge 10 s in 38.5 h (11,090 h CPU)
- ▶ Only levels 0 and 1 used for iso-surface visualization
- ▶ At t_e approximately 140M cells used vs. 44 billion (factor 315)
- ▶ Only levels 0 and 1 used for iso-surface visualization

Level	Grids	Cells
0	3,234	10,752,000
1	11,921	21,020,256
2	66,974	102,918,568
3	896	5,116,992

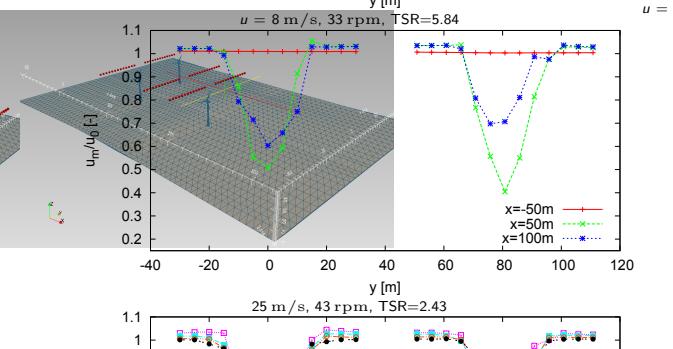
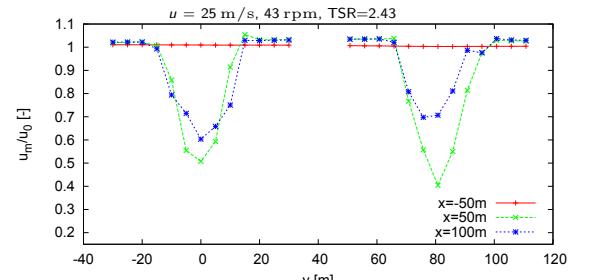
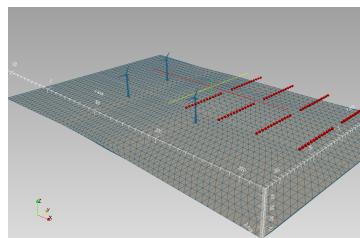
Refinement $u = 8 \text{ m/s}$, 33 rpm

Wake refinement behind a leading turbine

Axial velocity - $u = 8 \text{ m/s}$, 33 rpm

Mean point values

- ▶ Turbines located at $(0, 0, 0)$, $(135, 0, 0)$, $(-5.65, 80.80, 0)$
- ▶ Lines of 13 sensors with $\Delta y = 5 \text{ m}$, $z = 37 \text{ m}$ (approx. center of rotor)
- ▶ u and p measured over $[40 \text{ s}, 50 \text{ s}]$ (1472 level-0 time steps) and averaged



References I

[Chen et al., 2006] Chen, H., Filippova, O., Hoch, J., Molvig, K., Shock, R., Teixeira, C., and Zhang, R. (2006). Grid refinement in lattice Boltzmann methods based on volumetric formulation. *Physica A*, 362:158–167.

[Deiterding and Wood, 2015] Deiterding, R. and Wood, S. L. (2015). An adaptive lattice boltzmann method for predicting wake fields behind wind turbines. In Breitsamer, C. e. a., editor, *Proc. 19th DGLR-Fachsymposium der STAB, Munich, 2014*, Notes on Numerical Fluid Mechanics and Multidisciplinary Design. Springer. in press.

[Hähnel, 2004] Hähnel, D. (2004). *Molekulare Gasdynamik*. Springer.

[Henderson, 1995] Henderson, R. D. (1995). Details of the drag curve near the onset of vortex shedding. *Phys. Fluids*, 7:2102–2104.

[Hou et al., 1996] Hou, S., Sterling, J., Chen, S., and Doolen, G. D. (1996). A lattice Boltzmann subgrid model for high Reynolds number flows. In Lawniczak, A. T. and Kapral, R., editors, *Pattern formation and lattice gas automata*, volume 6, pages 151–166. Fields Inst Comm.

References II

[Schepers and Boorsma, 2012] Schepers, J. G. and Boorsma, K. (2012). Final report of ie a task 29: Mexnext (phase 1) – Analysis of Mexico wind tunnel measurements. Technical Report ECN-E-12-004, European research Centre of the Netherlands.

[Schlaffer, 2013] Schlaffer, M. B. (2013). *Non-reflecting boundary conditions for the lattice Boltzmann method*. PhD thesis, Technical University Munich.

[Toomey and Eldredge, 2008] Toomey, J. and Eldredge, J. D. (2008). Numerical and experimental study of the fluid dynamics of a flapping wing with low order flexibility. *Physics of Fluids*, 20(7):073603.

[Tsai, 1999] Tsai, L. (1999). *Robot Analysis: The Mechanics of Serial and Parallel Manipulators*. Wiley.

[Wood and Deiterding, 2015] Wood, S. L. and Deiterding, R. (2015). A lattice boltzmann method for horizontal axis wind turbine simulation. In *14th Int. Conf. on Wind Engineering*.

[Yu, 2004] Yu, H. (2004). *Lattice Boltzmann equation simulations of turbulence, mixing, and combustion*. PhD thesis, Texas A&M University.

Outline

Lecture 8

Structured AMR for elliptic problems

Course *Block-structured Adaptive Finite Volume Methods in C++*

Adaptive geometric multigrid methods

- Linear iterative methods for Poisson-type problems
- Multi-level algorithms
- Multigrid algorithms on SAMR data structures
- Example
- Comments on parabolic problems

Outline

Adaptive geometric multigrid methods

- Linear iterative methods for Poisson-type problems
- Multi-level algorithms
- Multigrid algorithms on SAMR data structures
- Example
- Comments on parabolic problems

Iterative methods

Jacobi iteration

$$Q_{jk}^{m+1} = \frac{1}{\sigma} \left[(Q_{j+1,k}^m + Q_{j-1,k}^m) \Delta x_2^2 + (Q_{j,k+1}^m + Q_{j,k-1}^m) \Delta x_1^2 - \Delta x_1^2 \Delta x_2^2 \psi_{jk} \right]$$

Lexicographical Gauss-Seidel iteration (use updated values when they become available)

$$Q_{jk}^{m+1} = \frac{1}{\sigma} \left[(Q_{j+1,k}^m + Q_{j-1,k}^{m+1}) \Delta x_2^2 + (Q_{j,k+1}^m + Q_{j,k-1}^{m+1}) \Delta x_1^2 - \Delta x_1^2 \Delta x_2^2 \psi_{jk} \right]$$

Efficient parallelization / patch-wise application not possible!

Checker-board or Red-Black Gauss Seidel iteration

1. $Q_{jk}^{m+1} = \frac{1}{\sigma} \left[(Q_{j+1,k}^m + Q_{j-1,k}^m) \Delta x_2^2 + (Q_{j,k+1}^m + Q_{j,k-1}^m) \Delta x_1^2 - \Delta x_1^2 \Delta x_2^2 \psi_{jk} \right]$
for $j+k \bmod 2 = 0$
2. $Q_{jk}^{m+1} = \frac{1}{\sigma} \left[(Q_{j+1,k}^{m+1} + Q_{j-1,k}^{m+1}) \Delta x_2^2 + (Q_{j,k+1}^{m+1} + Q_{j,k-1}^{m+1}) \Delta x_1^2 - \Delta x_1^2 \Delta x_2^2 \psi_{jk} \right]$
for $j+k \bmod 2 = 1$

Gauss-Seidel methods require $\sim 1/2$ of iterations than Jacobi method, however, iteration count still proportional to number of unknowns [Hackbusch, 1994]

Poisson equation

$$\begin{aligned}\Delta q(\mathbf{x}) &= \psi(\mathbf{x}), \quad \mathbf{x} \in \Omega \subset \mathbb{R}^d, \quad q \in C^2(\Omega), \quad \psi \in C^0(\Omega) \\ q &= \psi^\Gamma(\mathbf{x}), \quad \mathbf{x} \in \partial\Omega\end{aligned}$$

Discrete Poisson equation in 2D:

$$\frac{Q_{j+1,k} - 2Q_{jk} + Q_{j-1,k}}{\Delta x_1^2} + \frac{Q_{j,k+1} - 2Q_{jk} + Q_{j,k-1}}{\Delta x_2^2} = \psi_{jk}$$

Operator

$$\mathcal{A}(Q_{\Delta x_1, \Delta x_2}) = \begin{bmatrix} \frac{1}{\Delta x_2^2} & & \\ \frac{1}{\Delta x_1^2} & -\left(\frac{2}{\Delta x_1^2} + \frac{2}{\Delta x_2^2}\right) & \frac{1}{\Delta x_2^2} \\ & \frac{1}{\Delta x_2^2} & \end{bmatrix} Q(x_{1,j}, x_{2,k}) = \psi_{jk}$$

$$Q_{jk} = \frac{1}{\sigma} \left[(Q_{j+1,k} + Q_{j-1,k}) \Delta x_2^2 + (Q_{j,k+1} + Q_{j,k-1}) \Delta x_1^2 - \Delta x_1^2 \Delta x_2^2 \psi_{jk} \right]$$

$$\text{with } \sigma = \frac{2\Delta x_1^2 + 2\Delta x_2^2}{\Delta x_1^2 \Delta x_2^2}$$

Smoothing vs. solving

ν iterations with iterative linear solver

$$Q^{m+\nu} = \mathcal{S}(Q^m, \psi, \nu)$$

Defect after m iterations

$$d^m = \psi - \mathcal{A}(Q^m)$$

Defect after $m + \nu$ iterations

$$d^{m+\nu} = \psi - \mathcal{A}(Q^{m+\nu}) = \psi - \mathcal{A}(Q^m + v_\nu^m) = d^m - \mathcal{A}(v_\nu^m)$$

with correction

$$v_\nu^m = \mathcal{S}(\vec{0}, d^m, \nu)$$

Neglecting the sub-iterations in the smoother we write

$$Q^{n+1} = Q^n + v = Q^n + \mathcal{S}(d^n)$$

Observation: Oscillations are damped faster on coarser grid.

Coarse grid correction:

$$Q^{n+1} = Q^n + v = Q^n + \mathcal{PSR}(d^n)$$

where \mathcal{R} is suitable restriction operator and \mathcal{P} a suitable prolongation operator

Two-grid correction method

Relaxation on current grid:

$$\bar{Q} = \mathcal{S}(Q^n, \psi, \nu)$$

$$Q^{n+1} = \bar{Q} + \mathcal{P}\mathcal{S}(\vec{0}, \cdot, \mu)\mathcal{R}(\psi - \mathcal{A}(\bar{Q}))$$

Algorithm:

$$\begin{aligned} \bar{Q} &:= \mathcal{S}(Q^n, \psi, \nu) \\ d &:= \psi - \mathcal{A}(\bar{Q}) \\ d &:= \psi - \mathcal{A}(\bar{Q}) \\ d_c &:= \mathcal{R}(d) \\ v_c &:= \mathcal{S}(0, d_c, \mu) \\ v &:= \mathcal{P}(v_c) \\ Q^{n+1} &:= \bar{Q} + v \end{aligned}$$

$$\begin{aligned} d &:= \psi - \mathcal{A}(Q) \\ v &:= \mathcal{S}(0, d, \nu_1) \\ r &:= d - \mathcal{A}(v) \\ d_c &:= \mathcal{R}(r) \\ v_c &:= \mathcal{S}(0, d_c, \mu) \\ v &:= v + \mathcal{P}(v_c) \\ d &:= d - \mathcal{A}(v) \\ r &:= \mathcal{S}(0, d, \nu_2) \\ Q^{n+1} &:= Q + v \end{aligned}$$

[Hackbusch, 1985]

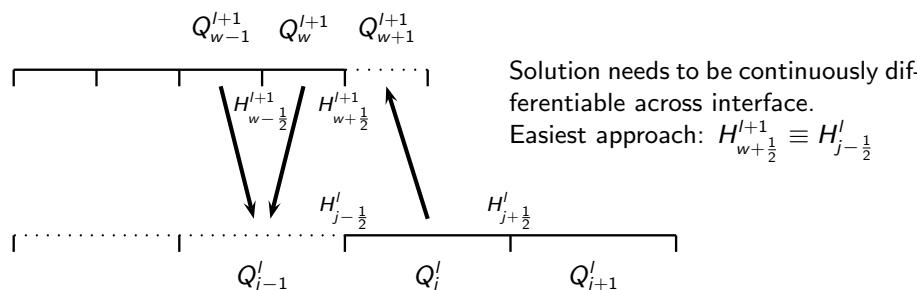
Stencil modification at coarse-fine boundaries in 1D

1D Example: Cell j , $\psi - \nabla \cdot \nabla q = 0$

$$d_j^I = \psi_j - \frac{1}{\Delta x_I} \left(\frac{1}{\Delta x_{I+1}} (Q'_{j+1} - Q'_j) - \frac{1}{\Delta x_I} (Q'_j - Q'_{j-1}) \right) = \psi_j - \frac{1}{\Delta x_I} (H'_{j+\frac{1}{2}} - H'_{j-\frac{1}{2}})$$

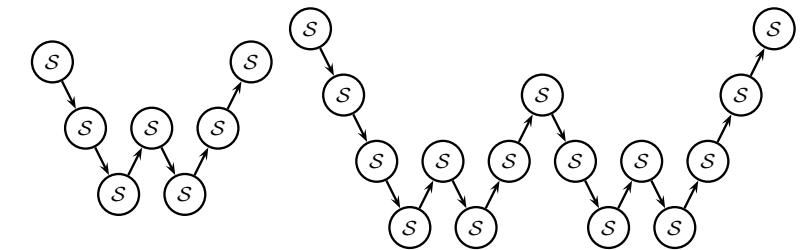
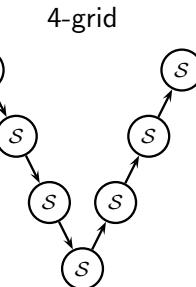
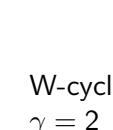
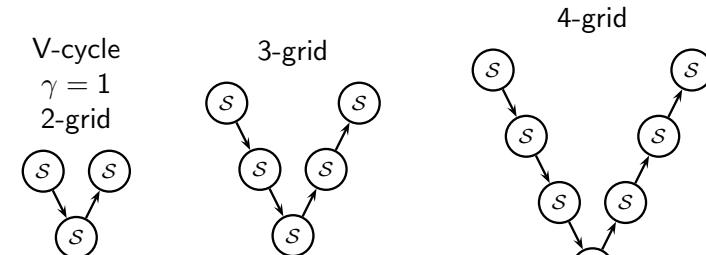
H is approximation to derivative of Q' .

Consider 2-level situation with $r_{I+1} = 2$:



No specific modification necessary for 1D vertex-based stencils, cf.
 [Bastian, 1996]

Multi-level methods and cycles



[Hackbusch, 1985] [Wesseling, 1992] ...

Stencil modification at coarse-fine boundaries in 1D II

Set $H'^{I+1}_{w+\frac{1}{2}} = H_I$. Inserting Q gives

$$\frac{Q'^{I+1}_{w+1} - Q'^{I+1}_w}{\Delta x_{I+1}} = \frac{Q'_j - Q'^{I+1}_w}{\frac{3}{2} \Delta x_{I+1}}$$

from which we readily derive

$$Q'^{I+1}_{w+1} = \frac{2}{3} Q'_j + \frac{1}{3} Q'^{I+1}_w$$

for the boundary cell on $I+1$. We use the flux correction procedure to enforce

$$H'^{I+1}_{w+\frac{1}{2}} \equiv H'_{j-\frac{1}{2}}$$

and thereby $H'_{j-\frac{1}{2}} \equiv H_I$.

Correction pass [Martin, 1998]

$$1. \delta H'^{I+1}_{j-\frac{1}{2}} := -H'_{j-\frac{1}{2}}$$

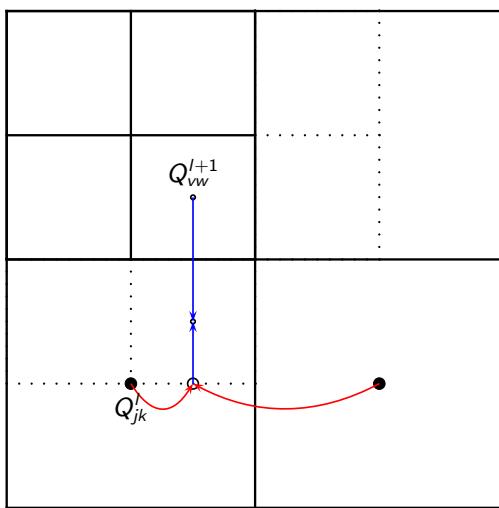
$$2. \delta H'^{I+1}_{j-\frac{1}{2}} := \delta H'^{I+1}_{j-\frac{1}{2}} + H'^{I+1}_{w+\frac{1}{2}} = -H'_{j-\frac{1}{2}} + (Q'_j - Q'^{I+1}_w)/\frac{3}{2} \Delta x_{I+1}$$

$$3. d'_j := d'_j + \frac{1}{\Delta x_I} \delta H'^{I+1}_{j-\frac{1}{2}}$$

yields

$$d'_j = \psi_j - \frac{1}{\Delta x_I} \left(\frac{1}{\Delta x_{I+1}} (Q'_{j+1} - Q'_j) - \frac{2}{3 \Delta x_{I+1}} (Q'_j - Q'^{I+1}_w) \right)$$

Stencil modification at coarse-fine boundaries: 2D



$$Q_{v,w-1}' = \frac{1}{3} Q_{vw} + \frac{2}{3} \left(\frac{3}{4} Q_{jk} + \frac{1}{4} Q_{j+1,k} \right)$$

In general:

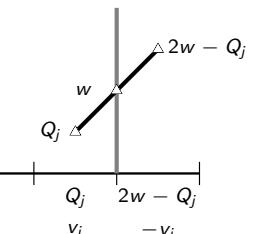
$$Q_{v,w-1}' = \left(1 - \frac{2}{r_{l+1} + 1} \right) Q_{vw} + \frac{2}{r_{l+1} + 1} \left((1-f) Q_{jk} + f Q_{j+1,k} \right)$$

with

$$f = \frac{x_{1,l+1}^v - x_{1,l}^j}{\Delta x_{1,l}}$$

Components of an SAMR multigrid method

- ▶ Stencil operators
 - ▶ Application of defect $d^l = \psi^l - \mathcal{A}(Q^l)$ on each grid $G_{l,m}$ of level l
 - ▶ Computation of correction $v^l = \mathcal{S}(0, d^l, \nu)$ on each grid of level l
- ▶ Boundary (ghost cell) operators
 - ▶ Synchronization of Q^l and v^l on \tilde{S}_l^1
 - ▶ Specification of Dirichlet boundary conditions for a finite volume discretization for $Q^l \equiv w$ and $v^l \equiv w$ on \tilde{P}_l^1
 - ▶ Specification of $v^l \equiv 0$ on $\tilde{\gamma}_l^1$
 - ▶ Specification of $Q_l = \frac{(r_l-1)Q^{l+1} + 2Q^l}{r_l + 1}$ on $\tilde{\gamma}_l^1$
- ▶ Coarse-fine boundary flux accumulation and application of δH^{l+1} on defect d^l
- ▶ Standard prolongation and restriction on grids between adjacent levels
- ▶ Adaptation criteria
 - ▶ E.g., standard restriction to project solution on 2x coarsened grid, then use local error estimation
- ▶ Looping instead of time steps and check of convergence



Additive geometric multigrid algorithm

`AdvanceLevelMG(l) - Correction Scheme`

```

Set ghost cells of  $Q^l$ 
Calculate defect  $d^l$  from  $Q^l, \psi^l$             $d^l := \psi^l - \mathcal{A}(Q^l)$ 
If ( $l < l_{max}$ )
  Calculate updated defect  $r^{l+1}$  from  $v^{l+1}, d^{l+1}$       $r^{l+1} := d^{l+1} - \mathcal{A}(v^{l+1})$ 
  Restrict  $d^{l+1}$  onto  $d^l$                                  $d^l := \mathcal{R}_l^{l+1}(r^{l+1})$ 
Do  $\nu_1$  smoothing steps to get correction  $v^l$            $v^l := \mathcal{S}(0, d^l, \nu_1)$ 
If ( $l > l_{min}$ )
  Do  $\gamma > 1$  times
    AdvanceLevelMG( $l - 1$ )
  Set ghost cells of  $v^{l-1}$ 
  Prolongate and add  $v^{l-1}$  to  $v^l$                    $v^l := v^l + \mathcal{P}_l^{l-1}(v^{l-1})$ 
  If ( $\nu_2 > 0$ )
    Set ghost cells of  $v^l$ 
    Update defect  $d^l$  according to  $v^l$                  $d^l := d^l - \mathcal{A}(v^l)$ 
    Do  $\nu_2$  post-smoothing steps to get  $r^l$            $r^l := \mathcal{S}(v^l, d^l, \nu_2)$ 
    Add additional correction  $r^l$  to  $v^l$              $v^l := v^l + r^l$ 
Add correction  $v^l$  to  $Q^l$                           $Q^l := Q^l + v^l$ 

```

Additive Geometric Multiplicative Multigrid Algorithm

```

Start - Start iteration on level  $l_{max}$ 
For  $l = l_{max}$  Downto  $l_{min} + 1$  Do
  Restrict  $Q^l$  onto  $Q^{l-1}$ 
  Regrid(0)
  AdvanceLevelMG( $l_{max}$ )

```

See also: [Trottenberg et al., 2001], [Canu and Ritzdorf, 1994]
Vertex-based: [Brandt, 1977], [Briggs et al., 2001]

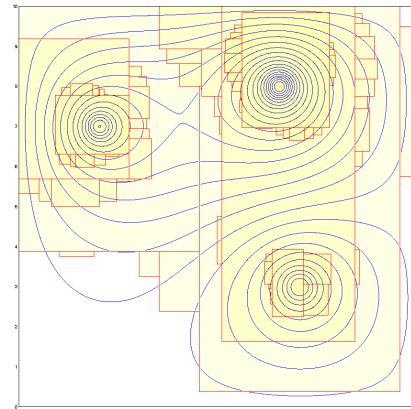
Example

On $\Omega = [0, 10] \times [0, 10]$ use hat function

$$\psi = \begin{cases} -A_n \cos\left(\frac{\pi r}{2R_n}\right), & r < R_n \\ 0 & \text{elsewhere} \end{cases}$$

with $r = \sqrt{(x_1 - X_n)^2 + (x_2 - Y_n)^2}$
to define three sources with

n	A_n	R_n	X_n	Y_n
1	0.3	0.3	6.5	8.0
2	0.2	0.3	2.0	7.0
3	-0.1	0.4	7.0	3.0



	128 × 128	1024 × 1024	1024 × 1024
l_{max}	3	0	0
l_{min}	-4	-7	-4
ν_1	5	5	5
ν_2	5	5	5
V-Cycles	15	16	341
Time [sec]	9.4	27.7	563

Stop at $\|d^l\|_{max} < 10^{-7}$ for $l \geq 0$, $\gamma = 1$, $r_l = 2$

References I

- [Bastian, 1996] Bastian, P. (1996). *Parallele adaptive Mehrgitterverfahren*. Teubner Skripten zur Numerik. B. G. Teubner, Stuttgart.
- [Brandt, 1977] Brandt, A. (1977). Multi-level adaptive solutions to boundary-value problems. *Mathematics of Computations*, 31(183):333–390.
- [Briggs et al., 2001] Briggs, W. L., Henson, V. E., and McCormick, S. F. (2001). *A Multigrid Tutorial*. Society for Industrial and Applied Mathematics, 2nd edition.
- [Canu and Ritzdorf, 1994] Canu, J. and Ritzdorf, H. (1994). Adaptive, block-structured multigrid on local memory machines. In Hackbusch, W. and Wittum, G., editors, *Adaptive Methods-Algorithms, Theory and Applications*, pages 84–98, Braunschweig/Wiesbaden. Proceedings of the Ninth GAMM-Seminar, Vieweg & Sohn.
- [Hackbusch, 1985] Hackbusch, W. (1985). *Multi-Grid Methods and Applications*. Springer Verlag, Berlin, Heidelberg.
- [Hackbusch, 1994] Hackbusch, W. (1994). *Iterative solution of large sparse systems of equations*. Springer Verlag, New York.

Some comments on parabolic problems

- ▶ Consequences of time step refinement
- ▶ Level-wise elliptic solves vs. global solve
- ▶ If time step refinement is used an elliptic flux correction is unavoidable.
- ▶ The correction is explained in Bell, J. (2004). Block-structured adaptive mesh refinement. Lecture 2. Available at <https://ccse.lbl.gov/people/jbb/shortcourse/lecture2.pdf>.

References II

- [Martin, 1998] Martin, D. F. (1998). *A cell-centered adaptive projection method for the incompressible Euler equations*. PhD thesis, University of California at Berkeley.
- [Trottenberg et al., 2001] Trottenberg, U., Oosterlee, C., and Schüller, A. (2001). *Multigrid*. Academic Press, San Antonio.
- [Wesseling, 1992] Wesseling, P. (1992). *An introduction to multigrid methods*. Wiley, Chichester.