Computational Astrophysics

Lecture 6: Software development and testing

Paul Ricker

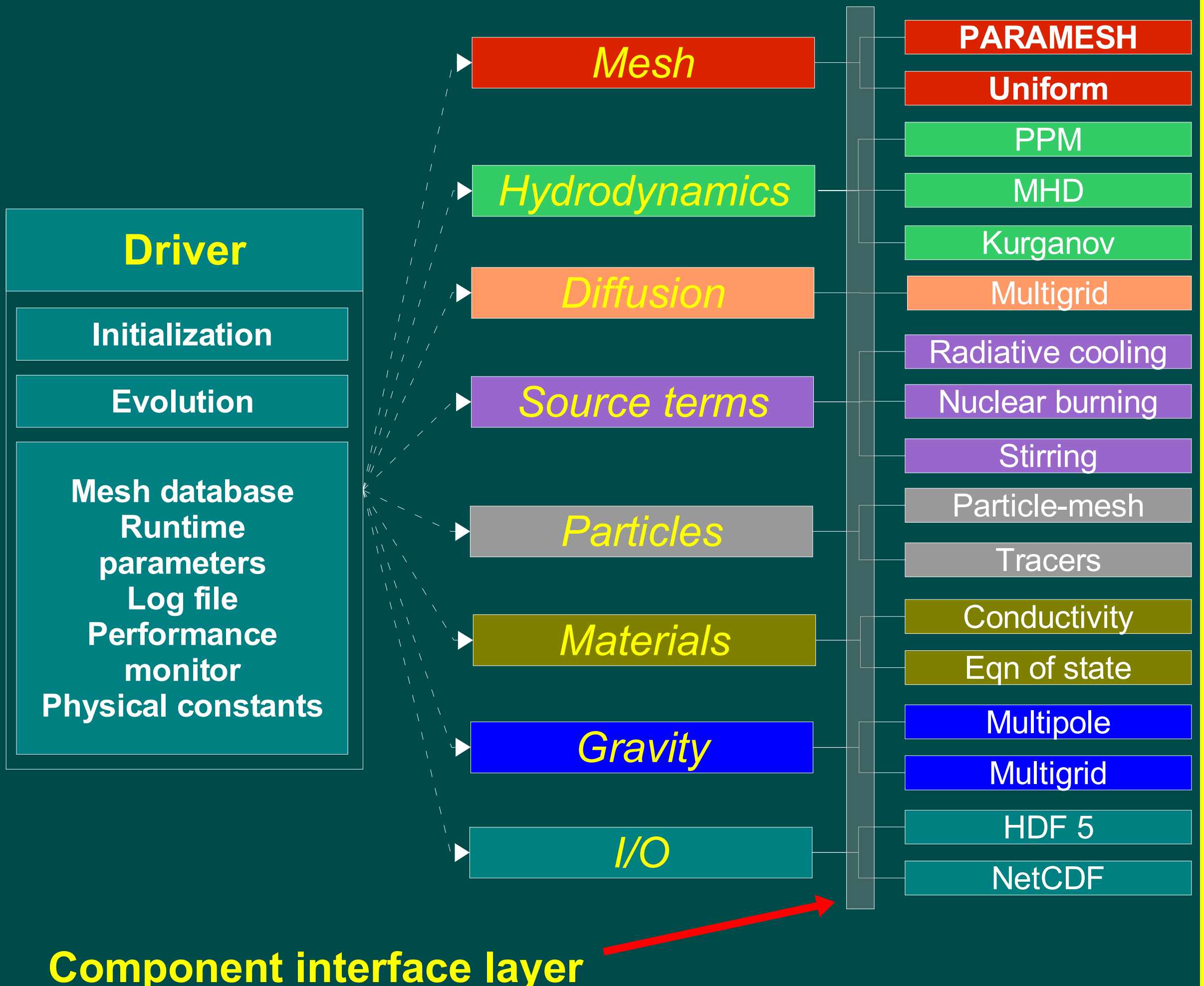
University of Illinois at Urbana-Champaign
National Center for Supercomputing Applications
Urbana, Illinois USA

Code development in astrophysics

- Traditional picture
  - Small codes (< 10,000 lines)
  - Limited physics scope
  - Simple data structures (e.g., uniform grids, serial)
  - Developed by individuals or small groups
  - Not shared with outside world
  - Informal testing process
  - Little emphasis on documentation or design
- Emerging picture
  - Large codes (> 100,000 lines)
  - Many different physical processes
  - Complex data structures (e.g., AMR grids, parallel)
  - Developed by teams of specialists
  - Often shared with and used by community
  - Demand for clear design, up-to-date documentation, and rigorous formal testing

FLASH (Fryxell et al. 2000)

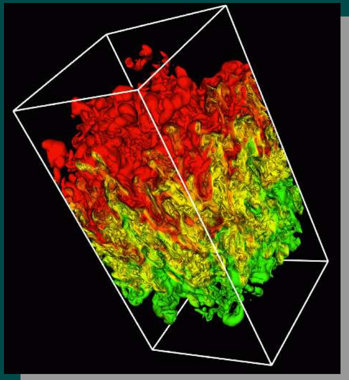
- Parallel AMR hydro code for astrophysical thermonuclear flash simulations developed under ASCI Alliances Program
- Framework designed to make creation and testing of physics modules “easy”
- Compile-time configuration handled by Python script, builds specified combination of modules

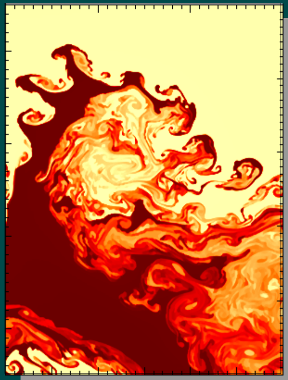


FLASH code framework
Ricker et al. (2000)
(astro-ph/0011502)

FLASH calculations

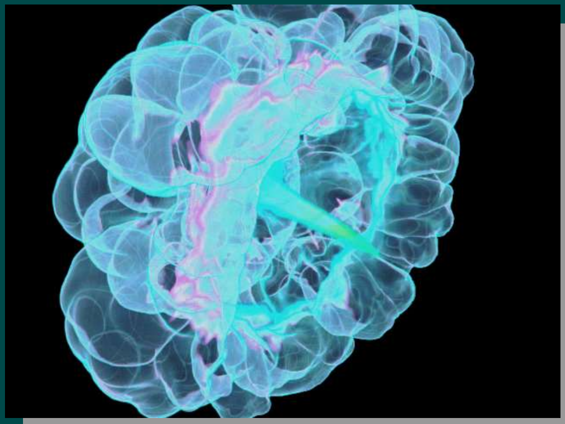
- Core astrophysics calculations
  - X-ray bursts
  - Novae and pre-nova mixing
  - Type Ia supernovae
- Microphysics calculations
  - Cellular detonations
  - Flame-vortex interactions
- Validation calculations
  - Rayleigh-Taylor instability
  - Richtmyer-Meshkov instability



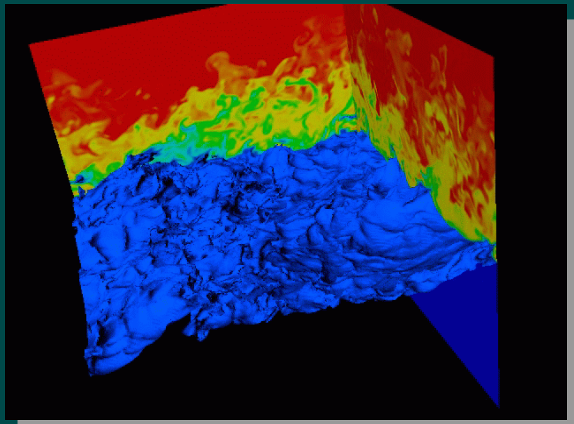


Rayleigh-Taylor instability

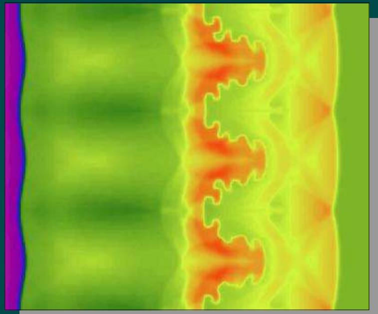
Richtmyer-Meshkov



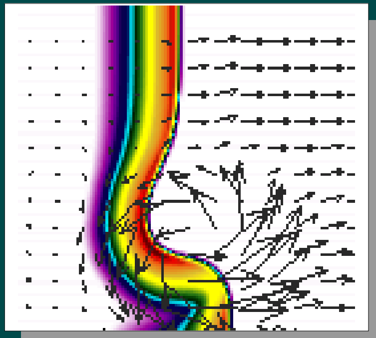
Type Ia supernova deflagrations



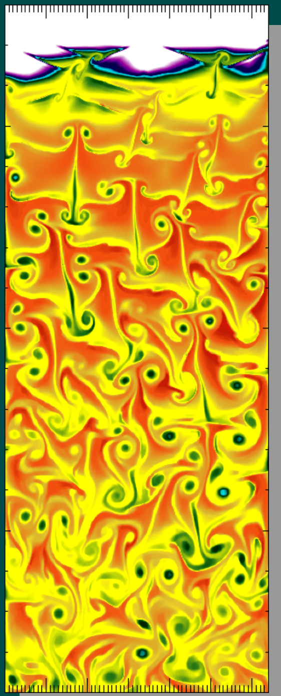
Gravity wave breaking on white dwarfs



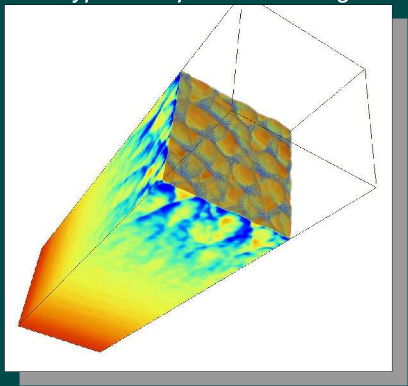
Laser-driven shock instabilities

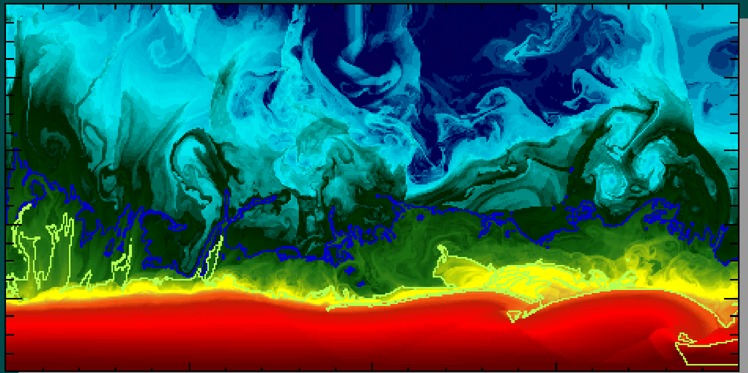


Flame-vortex interactions



Cellular detonations





Helium burning on neutron stars

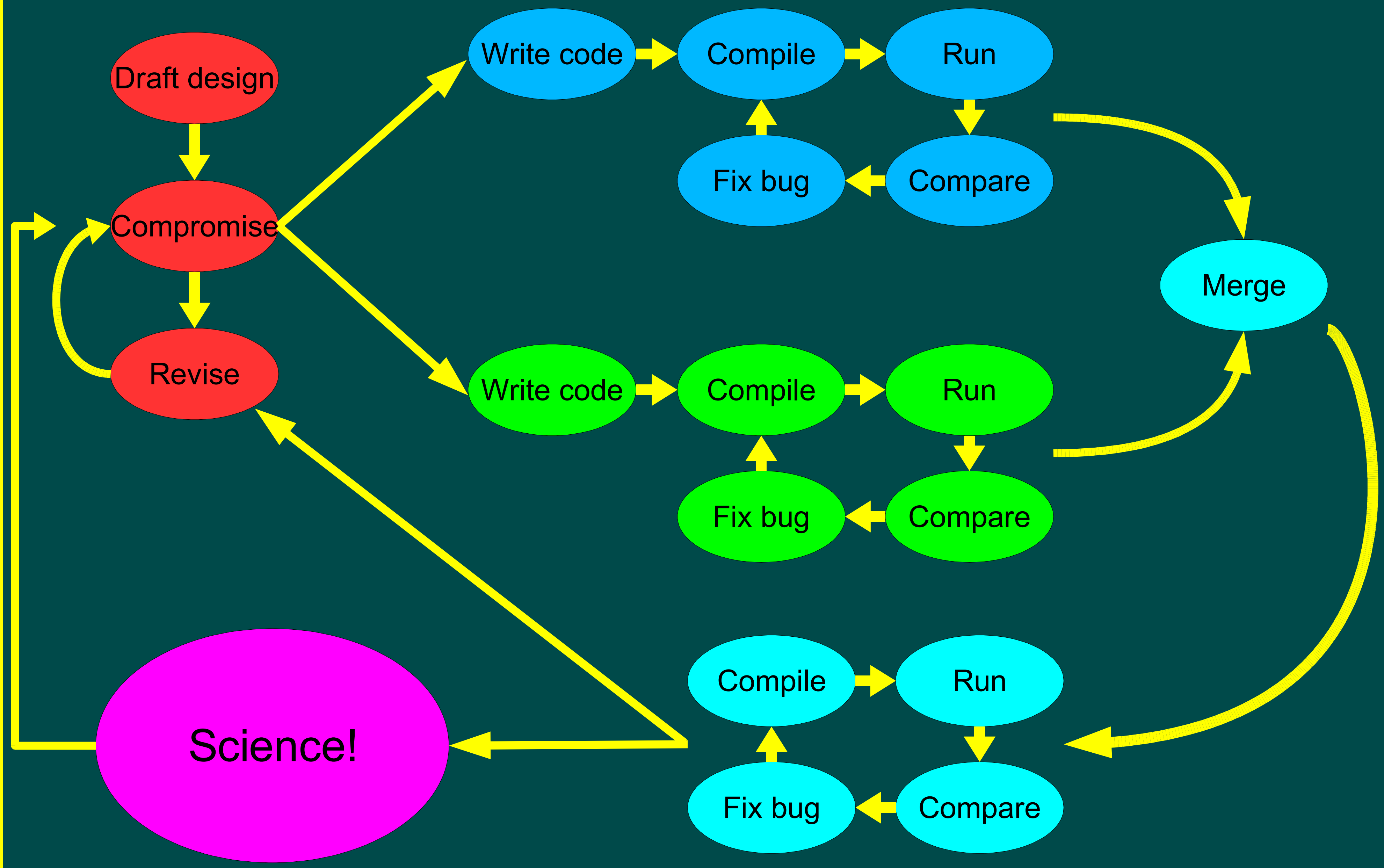
FLASH status

- Currently released: version 2.4
- Next release: version 2.5... soon!
- ~ 530,000 lines
  - Physics: Fortran 90 (65%), C (30%)
  - Configuration: Python (5%)
  - Support: Python, Perl, Java, and IDL
- Message-Passing Interface
- Parallel I/O: Hierarchical Data Format ver. 5
- Nightly test suite
- Free download (register): http://flash.uchicago.edu

FLASH physics – a sample

- Hydrodynamics
  - Shock-capturing algorithm (PPM; Colella & Woodward 1984)
  - Consistent multi-fluid advection (Plewa & Müller 1999)
  - Nonideal equation of state (Colella & Glaz 1985)
  - Nonequilibrium ionization
  - Magnetohydrodynamics (Powell et al. 1999) with div B cleaning
- Collisionless particles
  - Particle-mesh (e.g., Hockney & Eastwood 1988)
- Gravity
  - External fields
  - Multipole Poisson solver
  - Multigrid Poisson solver (Martin & Cartwright 1996)
  - Isolated boundaries (James 1977)
- Cosmological expansion and comoving coordinates
- Source terms
  - Explicit thermal conduction and viscosity (Spitzer 1962)
  - Optically thin radiative cooling (Peres et al. 1982 or Sutherland & Dopita 1993)

Code development cycle



Essential code development tools

- Compilers
  - Fortran 90 – G95, Intel, Portland Group, Lahey, NAG, Absoft
  - C/C++ – GCC, Intel, Portland Group
- Scripting languages
  - Python, Perl, Java
- Parallel communication and I/O
  - MPI, PVM, OpenMP
  - HDF5, NetCDF
- Code management tools
  - Building code – GNU Make, Ant
  - Version control – CVS, Aegis
  - Integrated development environments – Eclipse
- Debugging
  - Debuggers – GDB, IDB, DDD, TotalView
  - Bug tracking – BugZilla
  - Automated testing
- Performance measurement
  - SGI Perfex, PAPI
- Documentation
  - RoboDoc, Doxygen

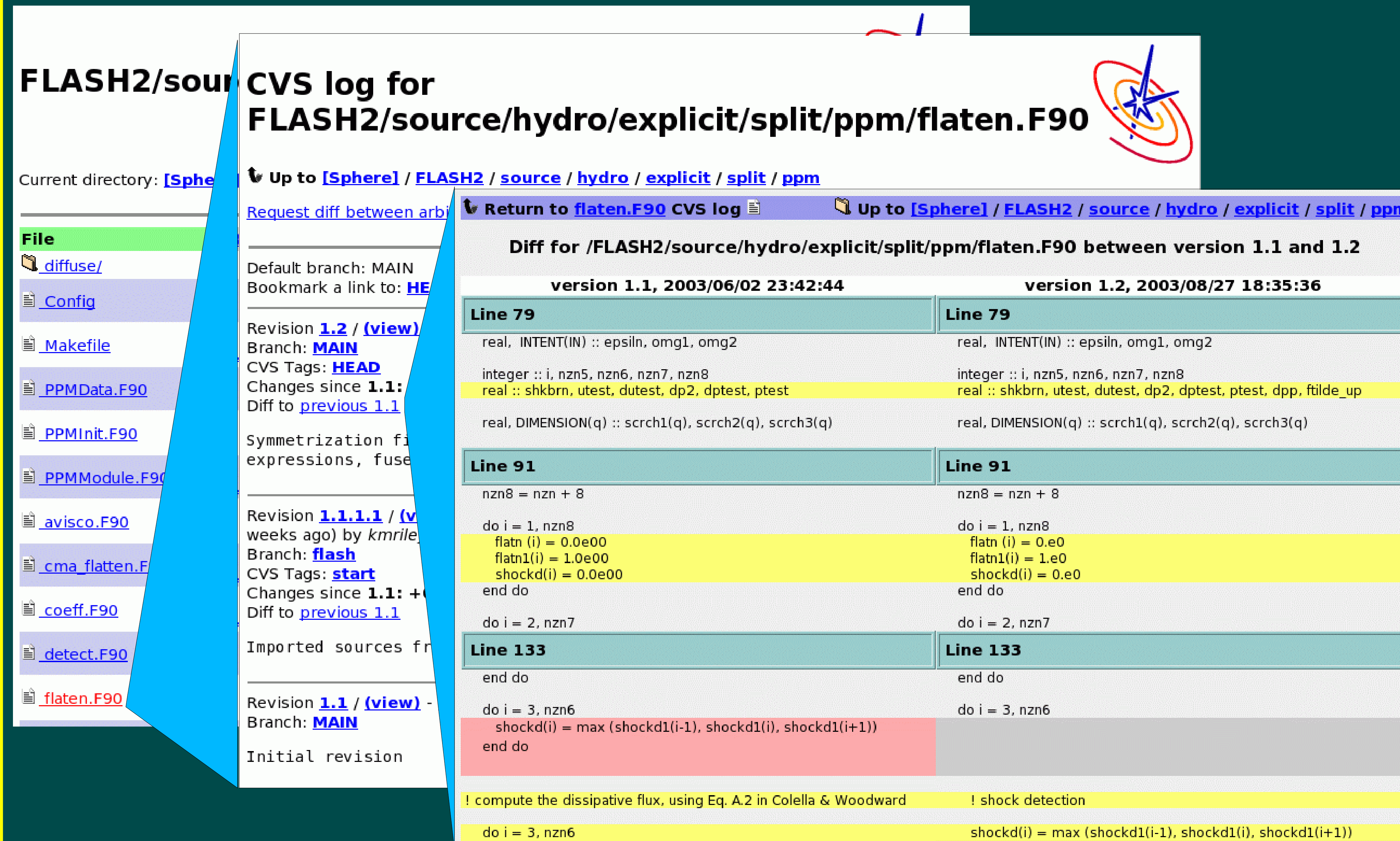
Make

Recompile only those files that have changed

```
# Makefile for 2D N-body demo program

F90 = ifort

.SUFFIXES : .f90
.f90.o :
        $(F90) -O2 -r8 -i4 -c $*.f90

nbody2d   : nbody2d.o init.o poisson2d.o pm2d.o cic.o \
            leapfrog.o extpot.o
        $(F90) -O2 -o nbody2d nbody2d.o init.o poisson2d.o \
               pm2d.o cic.o leapfrog.o extpot.o

nbody2d.o : init.f90 pm2d.f90 leapfrog.f90
pm2d.o    : poisson2d.f90 cic.f90 extpot.f90
```

Version control

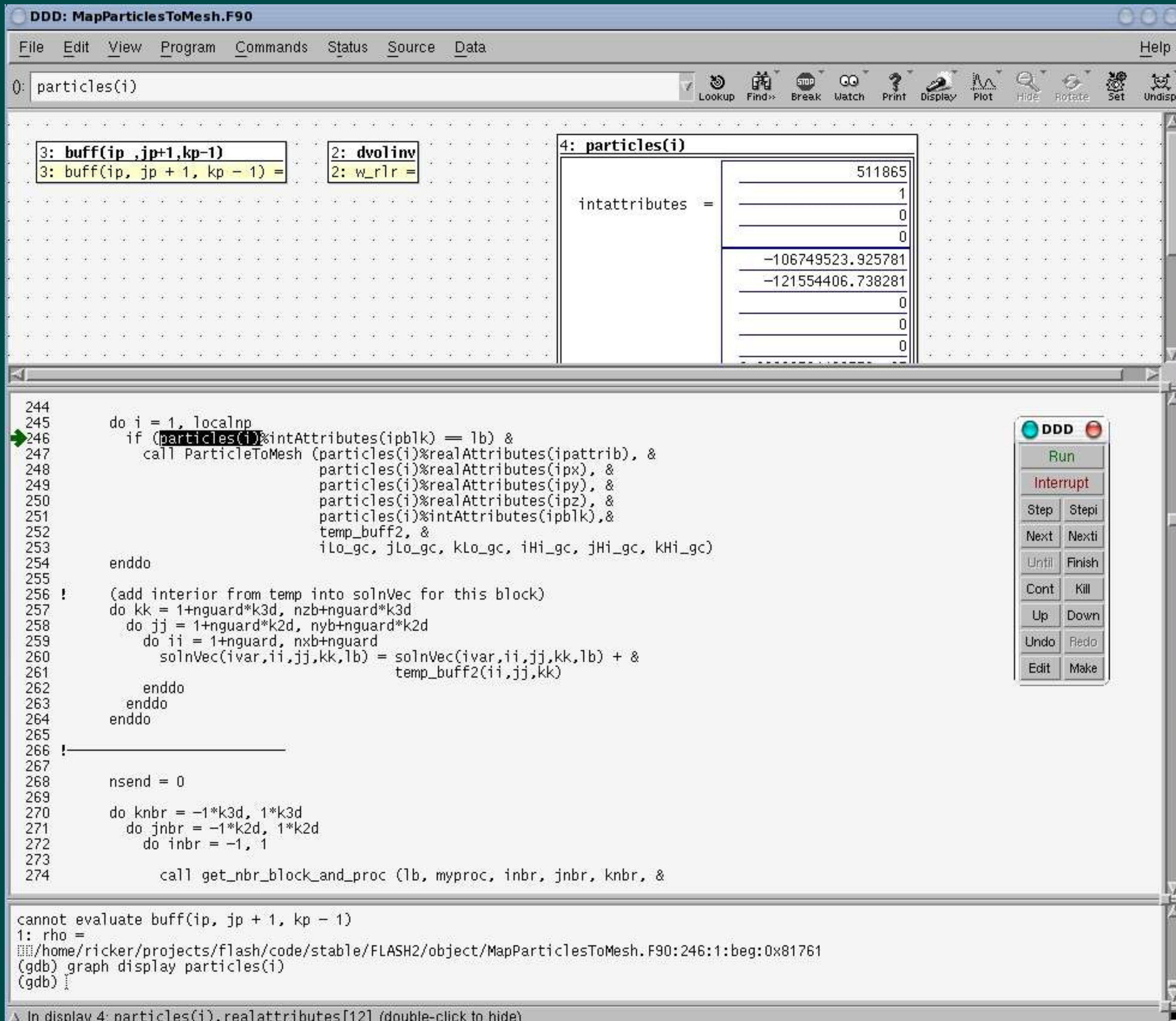
Concurrent Versioning System: http://www.cvshome.org



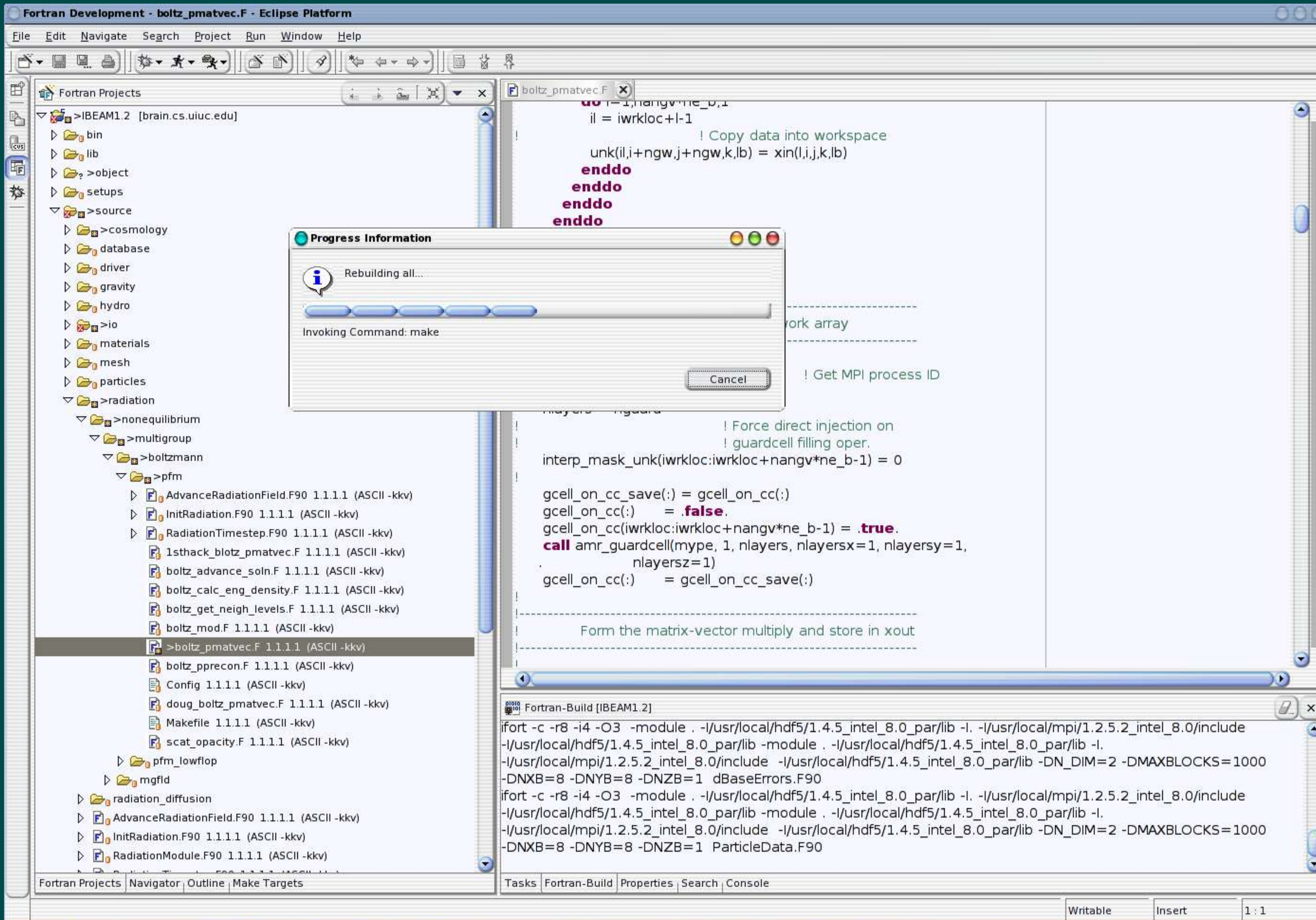
Debugging

Data Display Debugger

http://www.gnu.org/software/ddd/



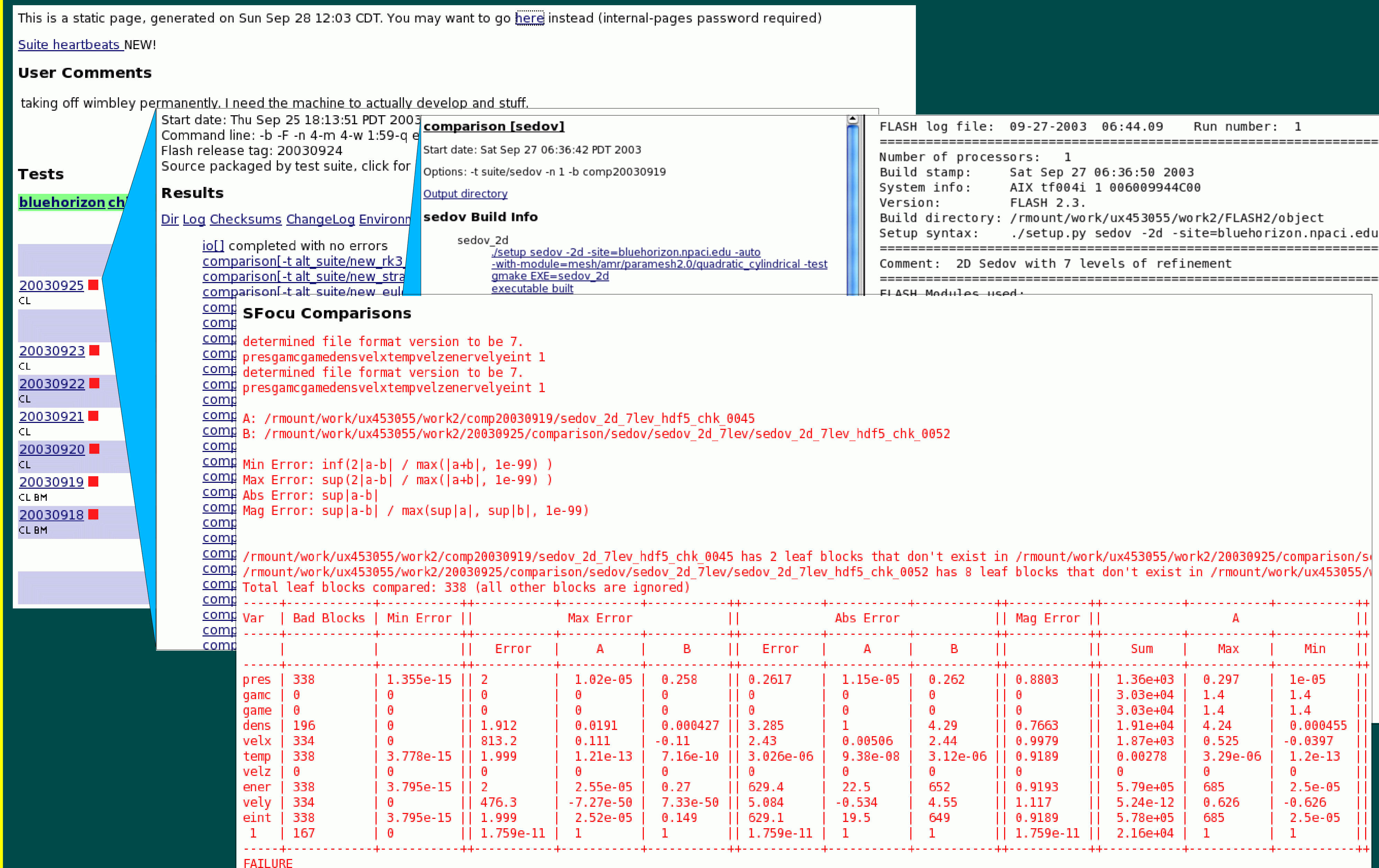
Eclipse



http://www.eclipse.org/

http://brain.cs.uiuc.edu/photran/photran.html

FLASH test suite



This is a static page, generated on Sun Sep 28 12:03 CDT. You may want to go here instead (internal-pages password required)

Suite heartbeats NEW!

User Comments

taking off wimbley permanently. I need the machine to actually develop and stuff.

Tests

bluehorizon ch

20030925 CL
20030923 CL
20030922 CL
20030921 CL
20030920 CL
20030919 CL BM
20030918 CL BM

Start date: Thu Sep 25 18:13:51 PDT 2003
Command line: -b -F -n 4-m 4-w 1:59-q e
Flash release tag: 20030924
Source packaged by test suite, click for

Results

Dir Log Checksums ChangeLog Environ

io[] completed with no errors
comparison[-t alt_suite/new_rk3
comparison[-t alt_suite/new_stra
comparison[-t alt_suite/new_eul

comparison [sedov]

Start date: Sat Sep 27 06:36:42 PDT 2003

Options: -t suite/sedov -n 1 -b comp20030919

Output directory

sedov Build Info

sedov_2d
./setup sedov -2d -site=bluehorizon.npaci.edu -auto -with-module=mesh/amr/paramesh2.0/quadratic_cylindrical -test
gmake EXE=sedov_2d
executable built

```
FLASH log file:  09-27-2003  06:44.09    Run number:  1
==========================================================
Number of processors:   1
Build stamp:     Sat Sep 27 06:36:50 2003
System info:     AIX tf004i 1 006009944C00
Version:         FLASH 2.3.
Build directory: /rmount/work/ux453055/work2/FLASH2/object
Setup syntax:    ./setup.py sedov -2d -site=bluehorizon.npaci.edu -
==========================================================
Comment:  2D Sedov with 7 levels of refinement
==========================================================
FLASH Modules used:
```

SFocu Comparisons

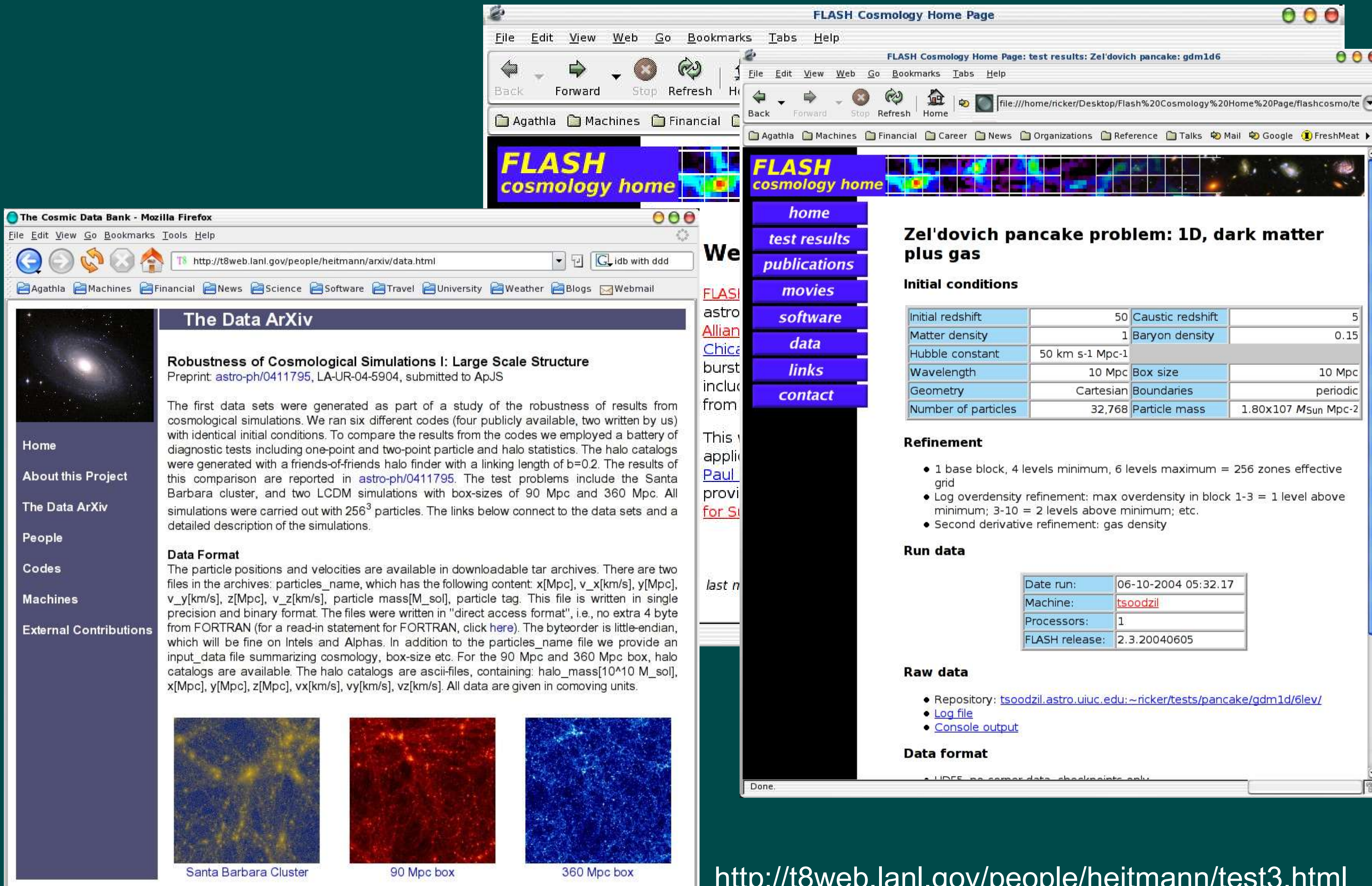
```
determined file format version to be 7.
presgamcgamedensvelxtempvelzenervelyeint 1
determined file format version to be 7.
presgamcgamedensvelxtempvelzenervelyeint 1

A: /rmount/work/ux453055/work2/comp20030919/sedov_2d_7lev_hdf5_chk_0045
B: /rmount/work/ux453055/work2/20030925/comparison/sedov/sedov_2d_7lev/sedov_2d_7lev_hdf5_chk_0052

Min Error: inf(2|a-b| / max(|a+b|, 1e-99) )
Max Error: sup(2|a-b| / max(|a+b|, 1e-99) )
Abs Error: sup|a-b|
Mag Error: sup|a-b| / max(sup|a|, sup|b|, 1e-99)

/rmount/work/ux453055/work2/comp20030919/sedov_2d_7lev_hdf5_chk_0045 has 2 leaf blocks that don't exist in /rmount/work/ux453055/work2/20030925/comparison/s
/rmount/work/ux453055/work2/20030925/comparison/sedov/sedov_2d_7lev/sedov_2d_7lev_hdf5_chk_0052 has 8 leaf blocks that don't exist in /rmount/work/ux453055/
Total leaf blocks compared: 338 (all other blocks are ignored)
```

Var	Bad Blocks	Min Error	Max Error			Abs Error			Mag Error	A		
			Error	A	B	Error	A	B		Sum	Max	Min
pres	338	1.355e-15	2	1.02e-05	0.258	0.2617	1.15e-05	0.262	0.8803	1.36e+03	0.297	1e-05
gamc	0	0	0	0	0	0	0	0	0	3.03e+04	1.4	1.4
game	0	0	0	0	0	0	0	0	0	3.03e+04	1.4	1.4
dens	196	0	1.912	0.0191	0.000427	3.285	1	4.29	0.7663	1.91e+04	4.24	0.000455
velx	334	0	813.2	0.111	-0.11	2.43	0.00506	2.44	0.9979	1.87e+03	0.525	-0.0397
temp	338	3.778e-15	1.999	1.21e-13	7.16e-10	3.026e-06	9.38e-08	3.12e-06	0.9189	0.00278	3.29e-06	1.2e-13
velz	0	0	0	0	0	0	0	0	0	0	0	0
ener	338	3.795e-15	2	2.55e-05	0.27	629.4	22.5	652	0.9193	5.79e+05	685	2.5e-05
vely	334	0	476.3	-7.27e-50	7.33e-50	5.084	-0.534	4.55	1.117	5.24e-12	0.626	-0.626
eint	338	3.795e-15	1.999	2.52e-05	0.149	629.1	19.5	649	0.9189	5.78e+05	685	2.5e-05
1	167	0	1.759e-11	1	1	1.759e-11	1	1	1.759e-11	2.16e+04	1	1

FAILURE

Posting detailed test results

http://www.astro.uiuc.edu/~pmricker/research/codes/flashcosmo/



http://t8web.lanl.gov/people/heitmann/test3.html

Types of tests

- Unit testing
  - Does each subroutine accept the expected range of input, and produce the expected range of output?
  - Are “contracts” fulfilled?
- Verification
  - “The process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution of the model.” (AIAA)
  - Are we solving the equations right?
- Validation
  - “The process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model.” (AIAA)
  - Are we solving the right equations?

Ideas to consider in verification tests

Physical limits and symmetries

- Mach number $\ll 1, \gg 1$
- Pressure gradients dominant or insignificant
- Diffusive terms on, off
- Long-range forces on, off, pre-specified
- Exploit symmetries of equations – translation, rotation, parity, Galilean and Lorentz transformations

Geometry

- Test problems with similar symmetries, especially if mesh symmetries are different
- Treatment of mesh boundaries – interaction of flow features with boundaries
- Treatment of coordinate singularities (e.g., cylindrical or spherical coordinates; general relativistic calculations)
- Degenerate geometries – do the centered case as well as the offset case

Ideas to consider in verification tests

Simplified models

- Simple source terms sometimes yield self-similar solutions (e.g., thermal bremsstrahlung instead of full cooling curve with atomic lines etc.)
- Perfect-gas equation of state instead of full equation of state
- Radiation field put in “by hand”

Numerical limits

- Sensitivity to convergence criteria for iterative methods
- Sensitivity to linear stability/accuracy criteria (e.g., CFL number)
- Sensitivity to artificial dissipation strength/type

Exercise of conditionals

- Nonlinear schemes have a lot of switches. Have all combinations been tried? (e.g., a centered rarefaction occurs but you've only tested shocks)

Verification problems useful in astrophysics

One-dimensional shock problems

- Sod (1978) problem
- Sedov (1959) problem
- Zalesak (2000) strong shock problem
- Shu-Osher (1998) problem
- Woodward-Colella (1984) interacting blast wave problem
- Brio-Wu (1988) MHD shock-tube problem

Two-dimensional shock problems

- Emery (1968) wind tunnel problem
- Double Mach reflection from a wedge (e.g., Woodward & Colella 1984)

Fluid instability problems

- Jeans (1902) instability
- Kelvin-Helmholtz (1800's) instability
- Gravity waves
- Orszag-Tang (1979) MHD vortex

Verification problems useful in astrophysics

Gravitational collapse problems

- Plane-parallel collapse (Zel'dovich (1970) pancake – see Anninos & Norman 1994)
- Spherical dust cloud collapse (Colgate & White 1966; Bertschinger 1985)
- Isothermal sphere collapse (Foster & Chevalier 1992)

Hydrodynamic stability problems (two books by Chandrasekhar)

- Maclaurin (1700's) spheroid
- Jacobi and Dedekind (1800's) ellipsoids
- Inhomogeneous polytropes

Radiation/radiative cooling hydrodynamics problems

- Stability of radiative shocks (e.g., Blondin, Chevalier papers)
- Self-similar cooling flow model (Chevalier *again*)

Particle tests

- Kepler (1600's) problem; any “dusty” gas problems

Measures of global error

Integral quantities – mass, momentum, energy, etc.

- Crude “catastrophic failure” test – esp. for explicitly conservative finite-volume schemes
- More informative if method is not explicitly conservative (e.g., source terms, finite-difference, internal energy method for hypersonic flows, etc.)

Function-integral error norms

- L1 norm

$$\text{L1 norm} \equiv \frac{1}{N}\sum_i \left| f_i^{\text{numeric}} - f_i^{\text{analytic}} \right|$$

- L2 norm

$$\text{L2 norm} \equiv \left[\frac{1}{N}\sum_i \left| f_i^{\text{numeric}} - f_i^{\text{analytic}} \right|^2 \right]^{1/2}$$

- Maximum norm

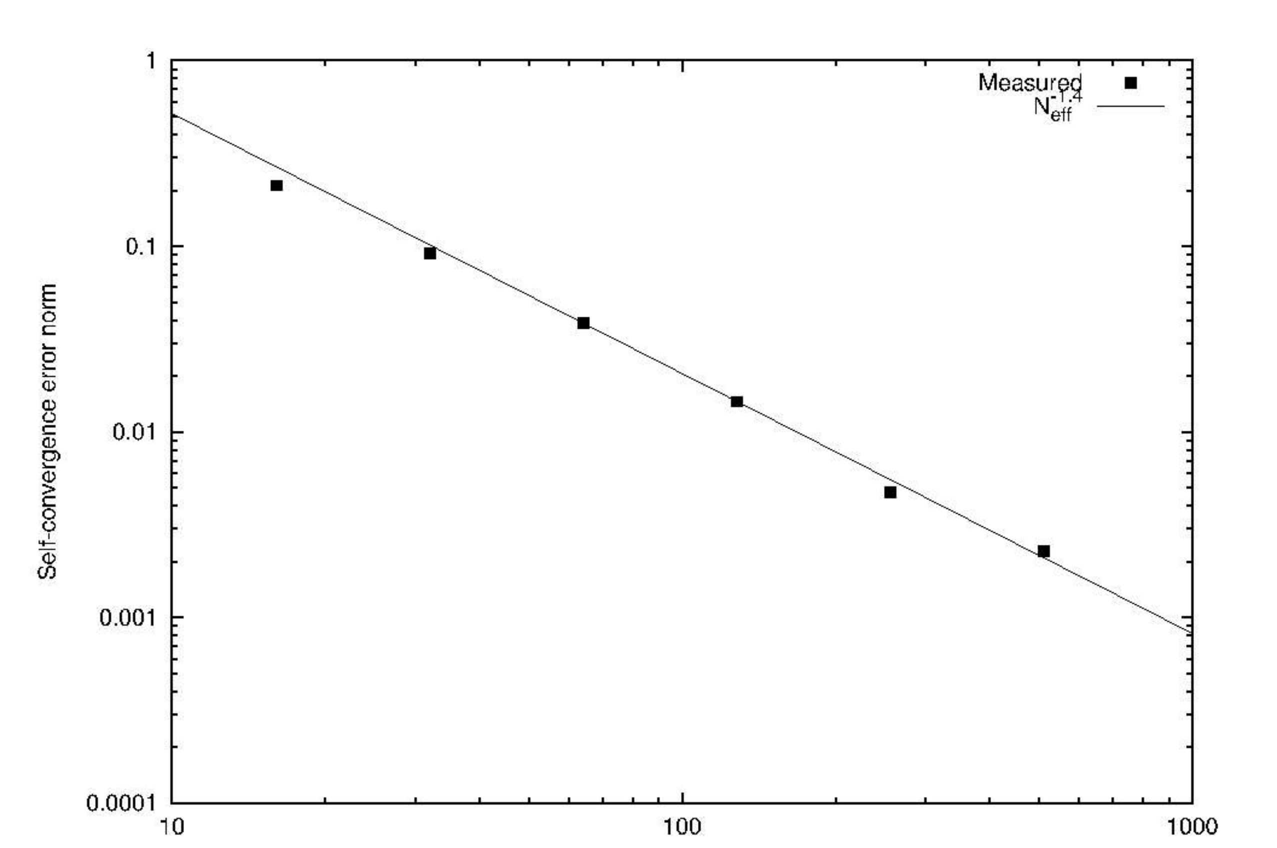
$$\text{max norm} \equiv \sup \left| f_i^{\text{numeric}} - f_i^{\text{analytic}} \right|$$

Measures of global error

Want to show global error convergence rate (generally poorer than local rate)

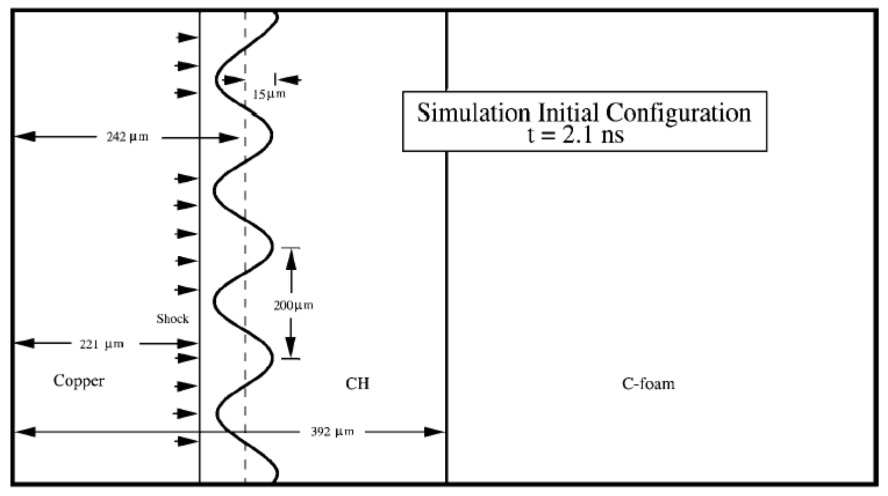
In asymptotic regime, error $\varepsilon \propto \Delta x^p$; can estimate p even without exact solution using three solutions on grids with refinement factor r:

$$p = \frac{\ln E(f_3, f_2) - \ln E(f_2, f_1)}{\ln r}$$

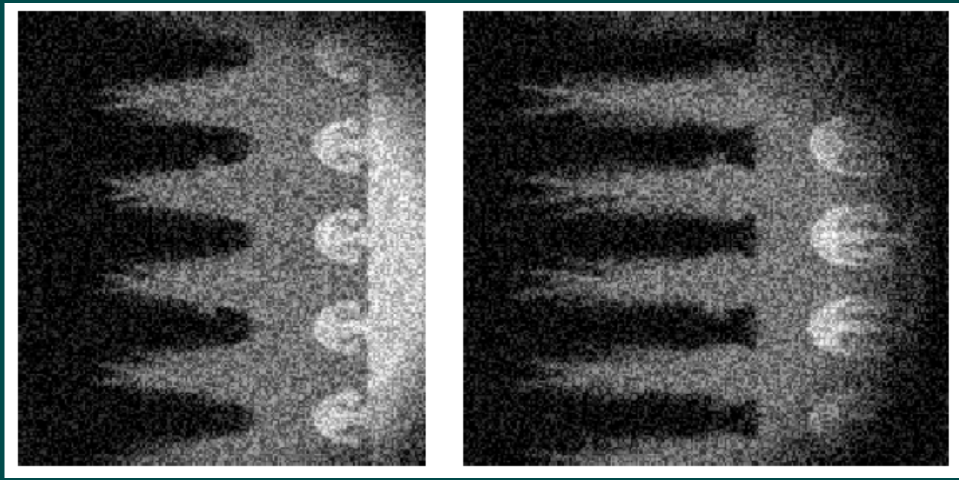


Validation example

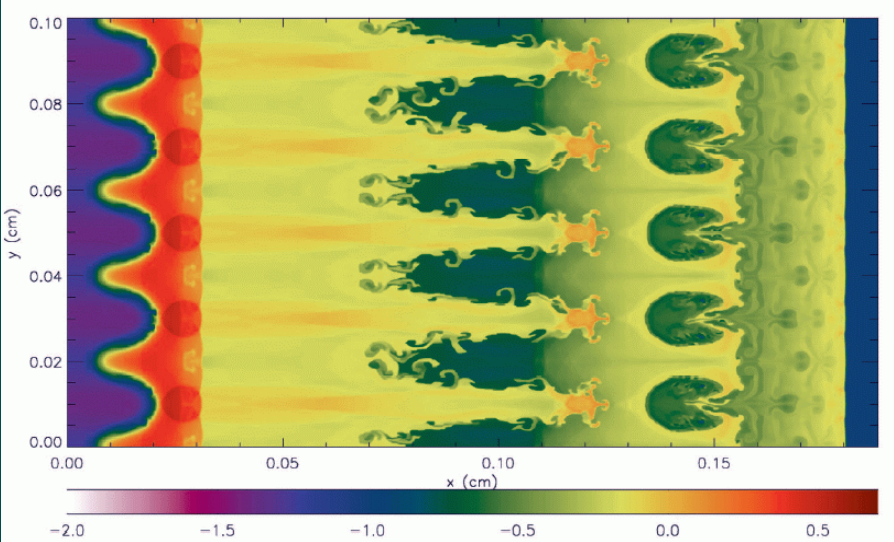
3-layer laser-driven shock experiments (Calder et al. 2002)



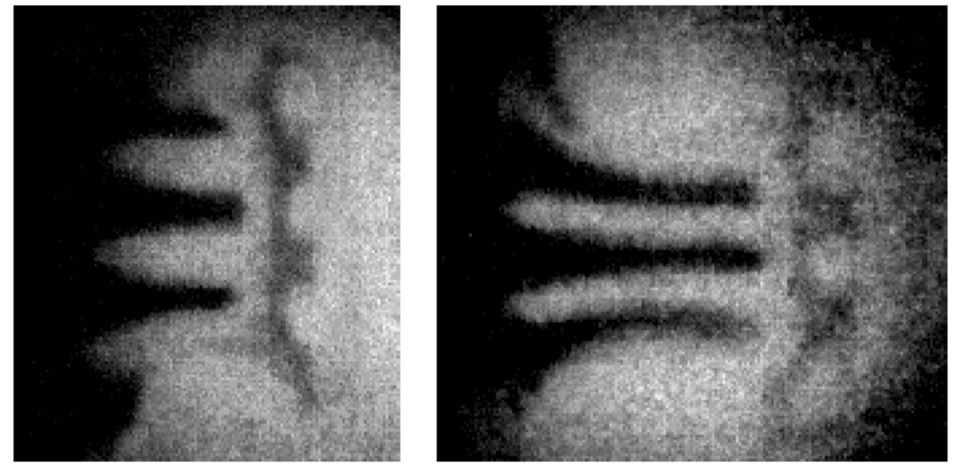
Simulation



Simulation



Experiment



$t = 39.9$ ns 66.0 ns

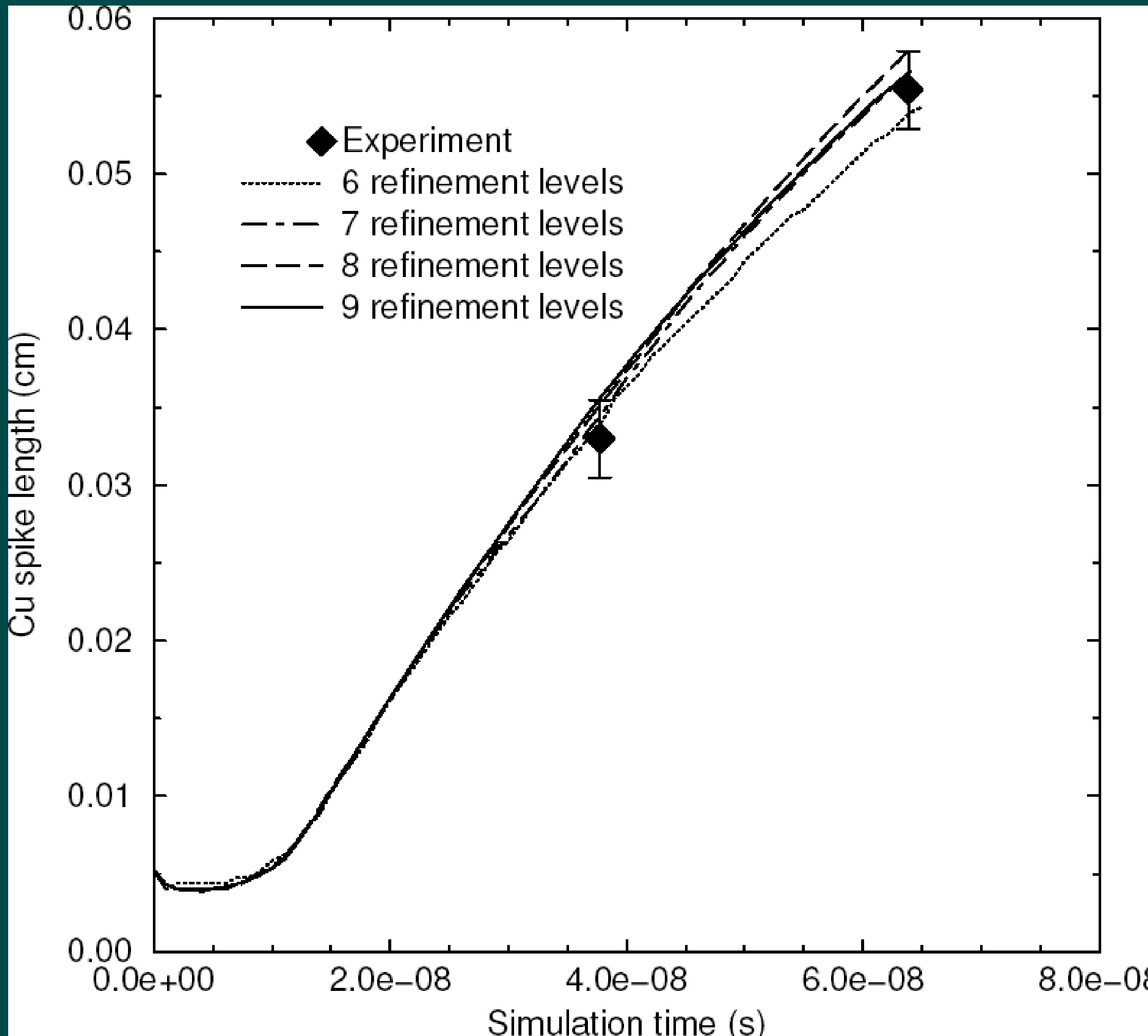
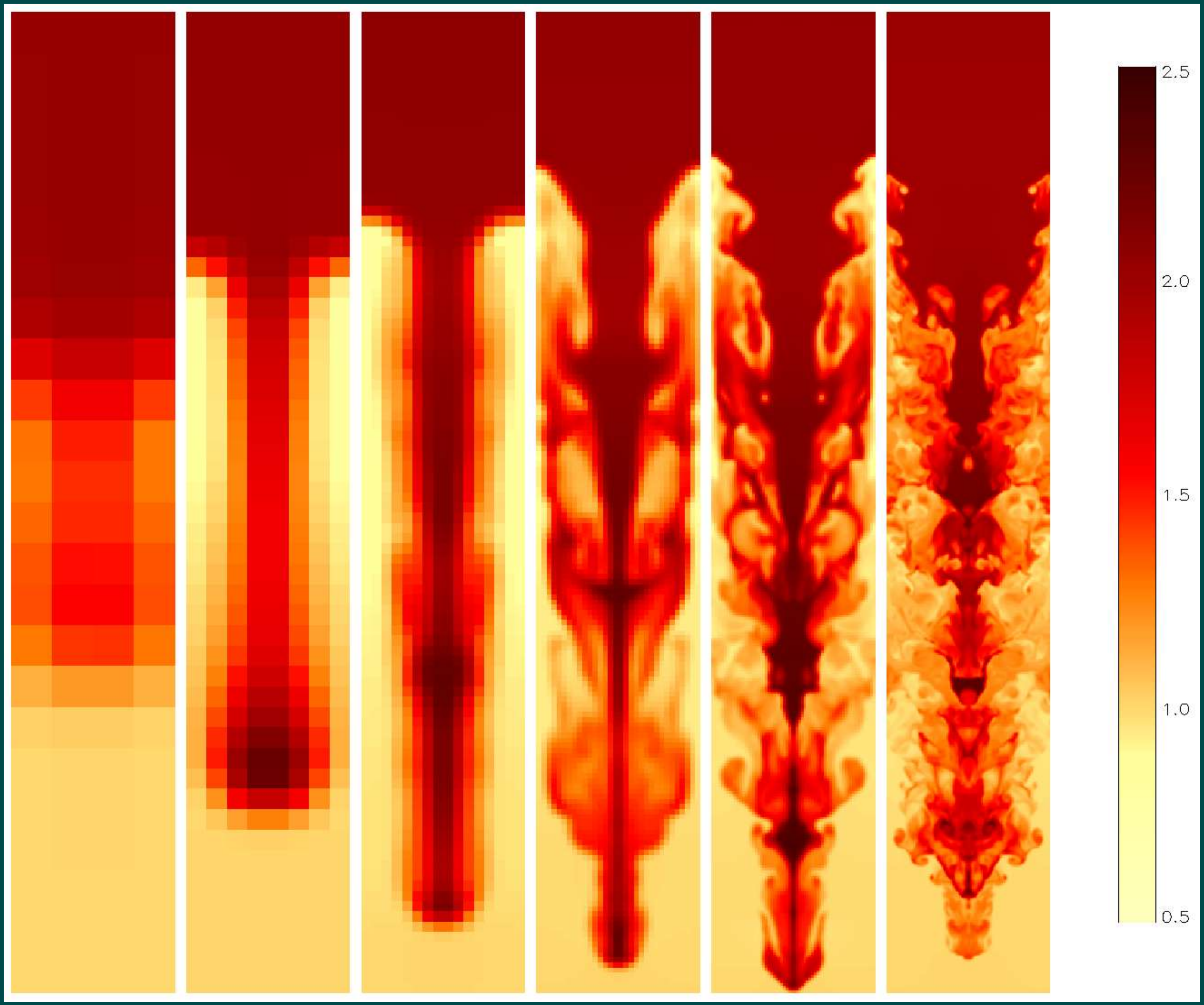


Fig. 16.— Cu spike length vs. time. The curves are from simulations at 6, 7, 8, and 9 levels of refinement simulations (effective resolutions of 256×512, 512×1024, 1024×2048, 2048×4096), and the points with error bars are results from the experiment. The error bars represent $\pm 25\ \mu$m, and the width of the symbols represents the timing error.

Validation (code comparison) example

Calder et al. (2002) - single-mode Rayleigh-Taylor instability



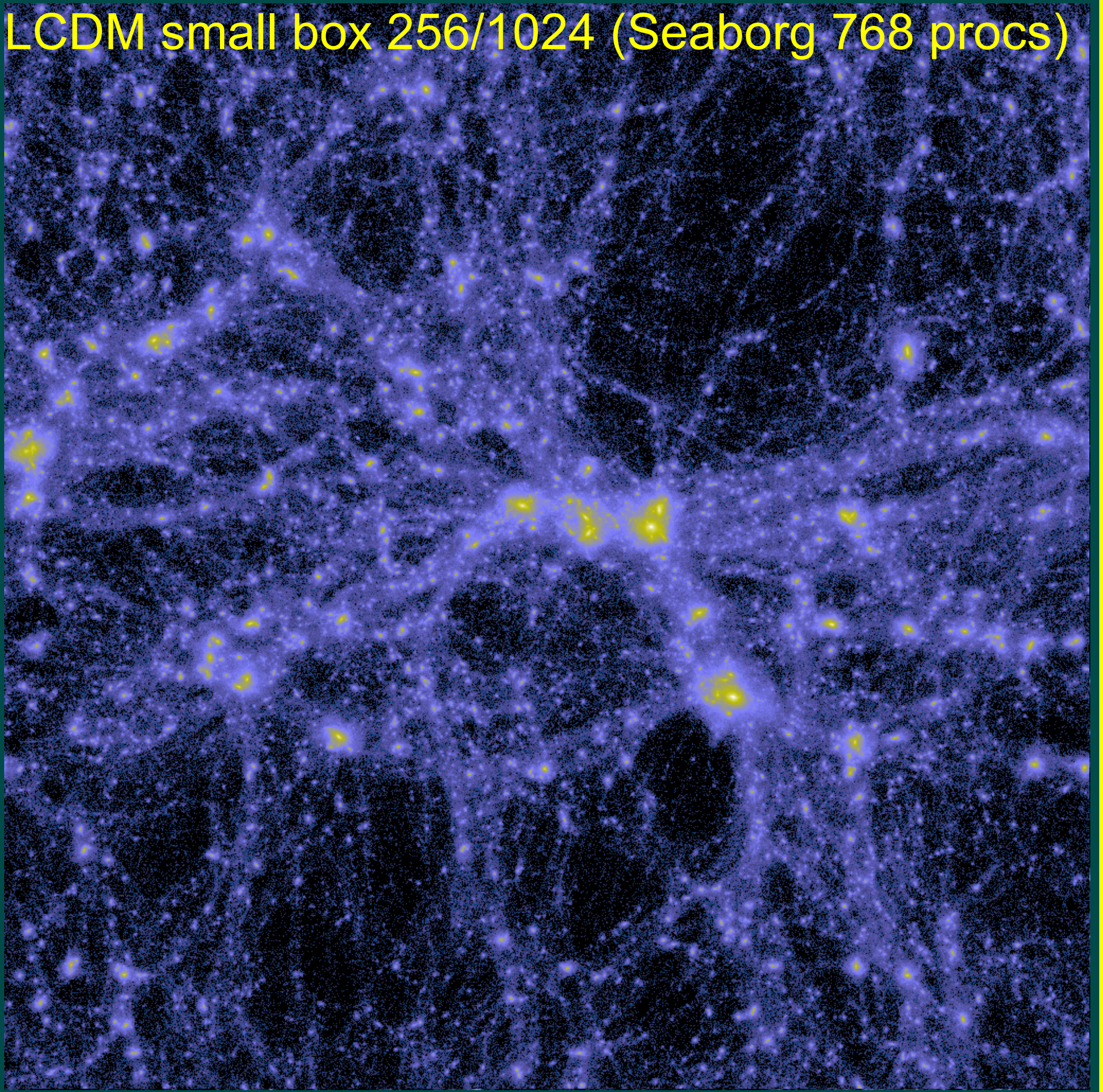
λ_{eff} = 4 8 16 32 64 128 zones

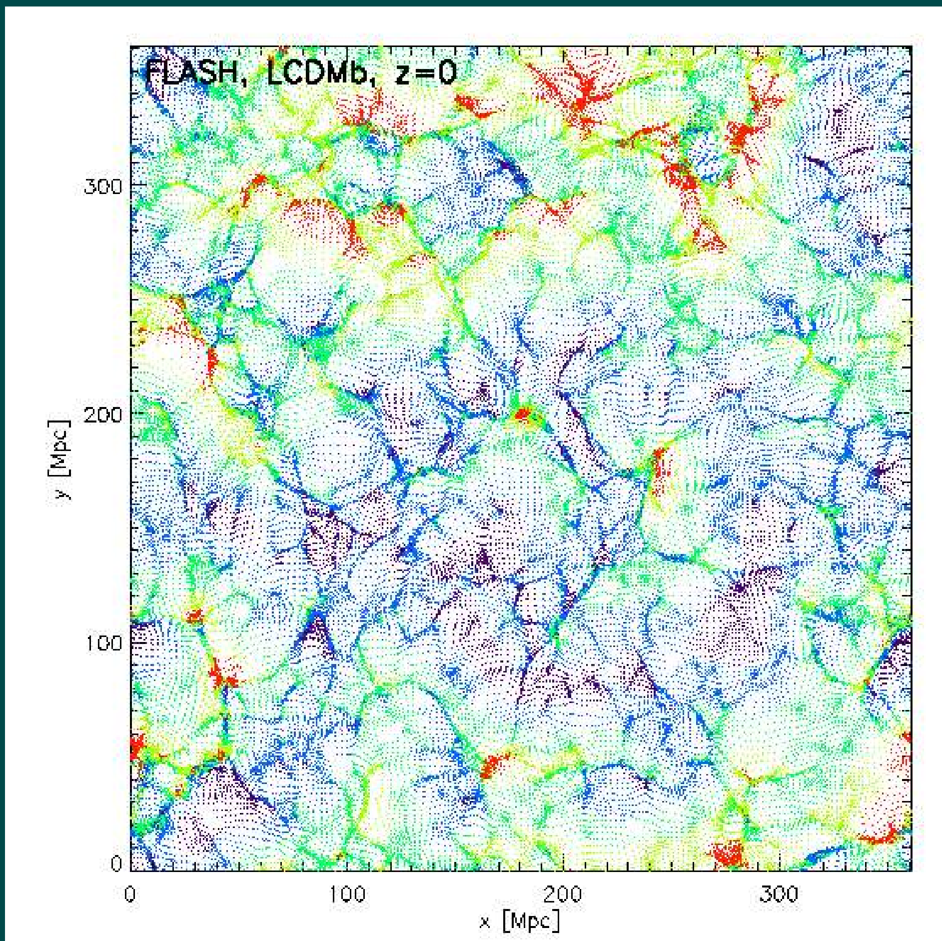
Code comparison example

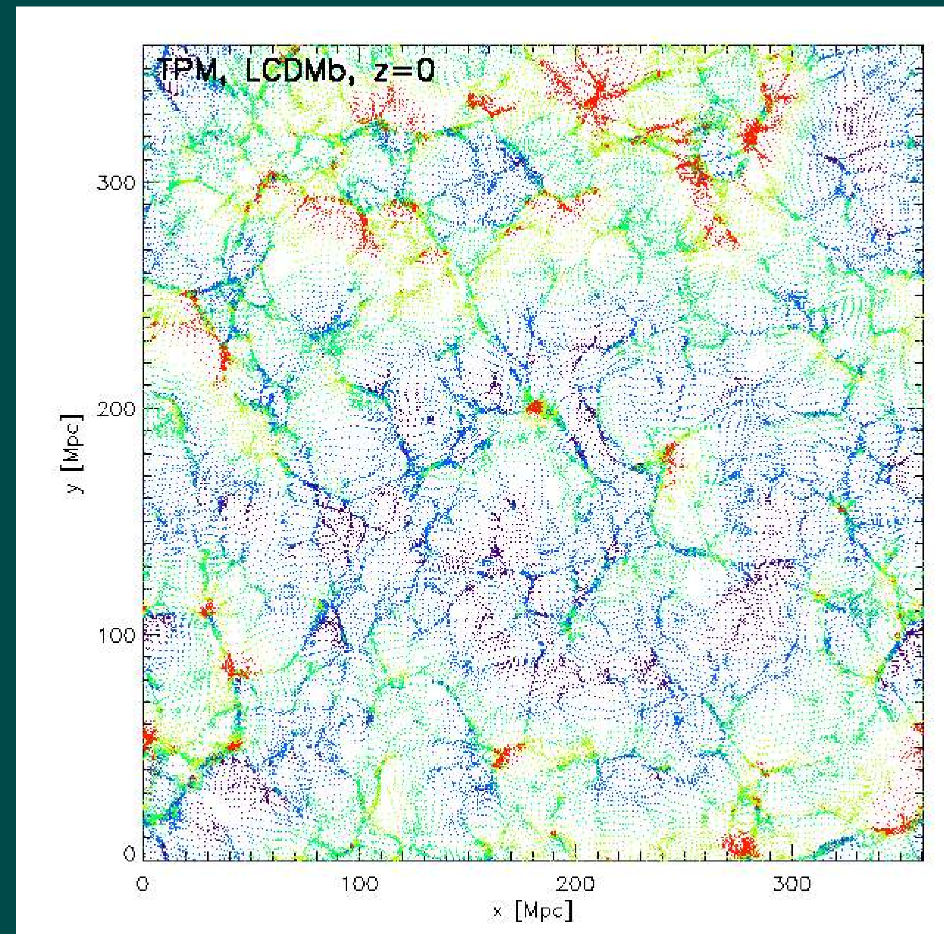
ΛCDM model

Heitmann et al. (2004)

- Two box sizes
  - $L = 64h^{-1}$ Mpc
  - $L = 256h^{-1}$ Mpc
- Comparisons
  - 256^3 particles / 1024^3 grid
  - 512^3 particles / 512^3 grid



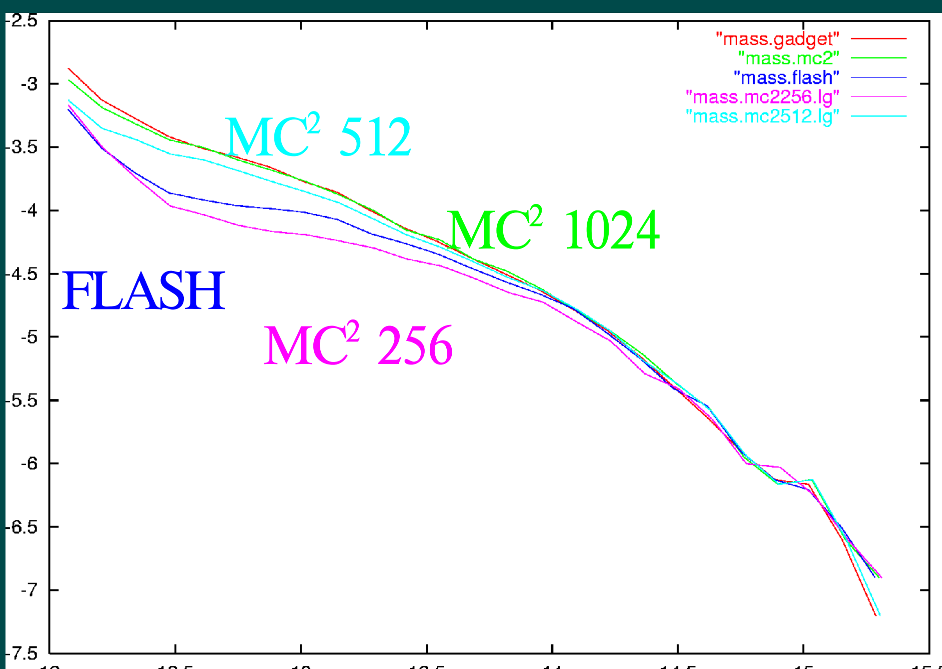


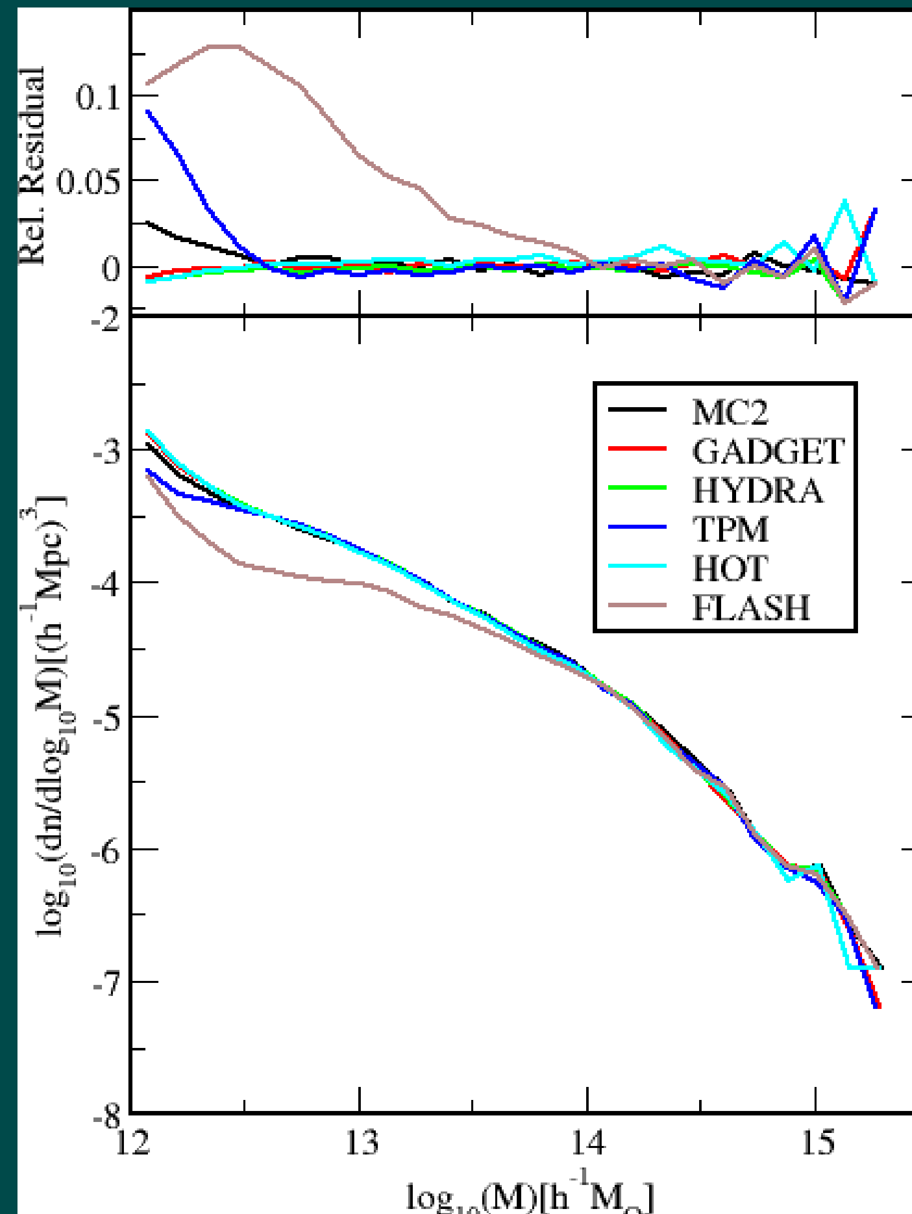


0-125 125-250 250-375 375-500 km/s

Halo mass function

- All codes agree at high masses
- At highest masses, too few halos despite large box
- At lower masses, AMR simulation agrees with lower-mass PM runs
- Below ~ 30 particles/halo, counts dominated by halo finder systematics
- Calculations must be validated as well as codes!





Thank you!