The Magnetic Reconnection Code: Framework and Application

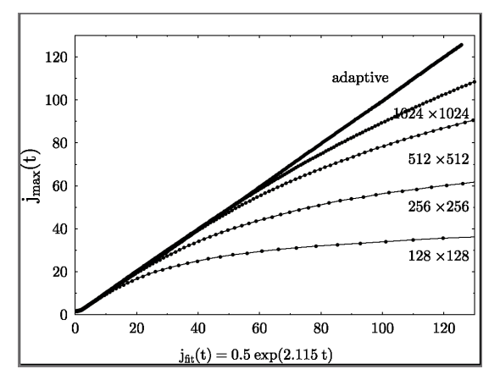
K. Germaschewski, A. Bhattacharjee (University of Iowa)
T. Linde, R. Rosner, A. Siegel (University of Chicago)
D. Keyes, F. Dobrian (Old Dominion University)

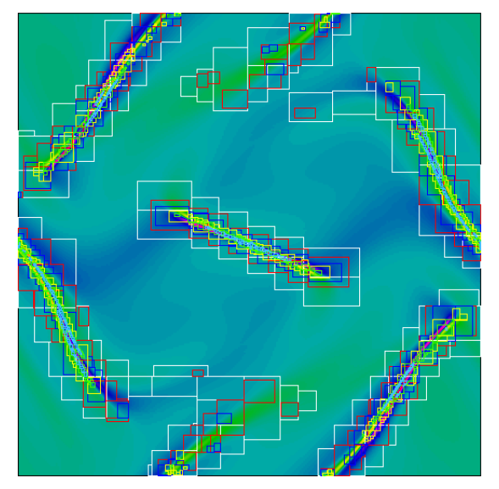
1 Adaptive Mesh Refinement

1.1 Example: 2D ideal MHD (previous work)

Efficiency of AMR

Level	# grids	# grid points
0	1	70225
1	83	146080
2	103	268666
3	153	545316
4	197	1042132
5	404	1926465
6	600	1967234



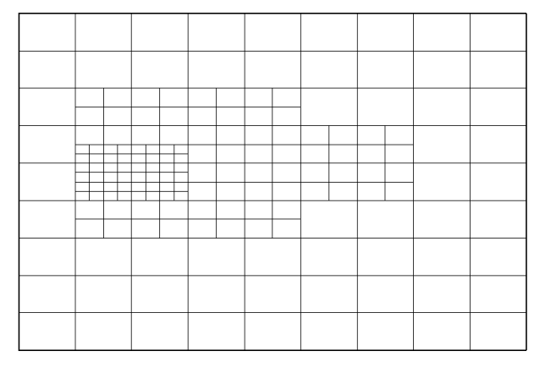


Grid points in adaptive simulation:	6976118
Grid points in non-adaptive simulation:	268730449
Ratio	0.02

1.2 Quad/Oct-Tree vs. arbitrary patches

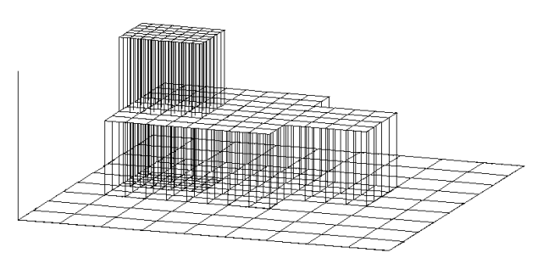
Two approaches:

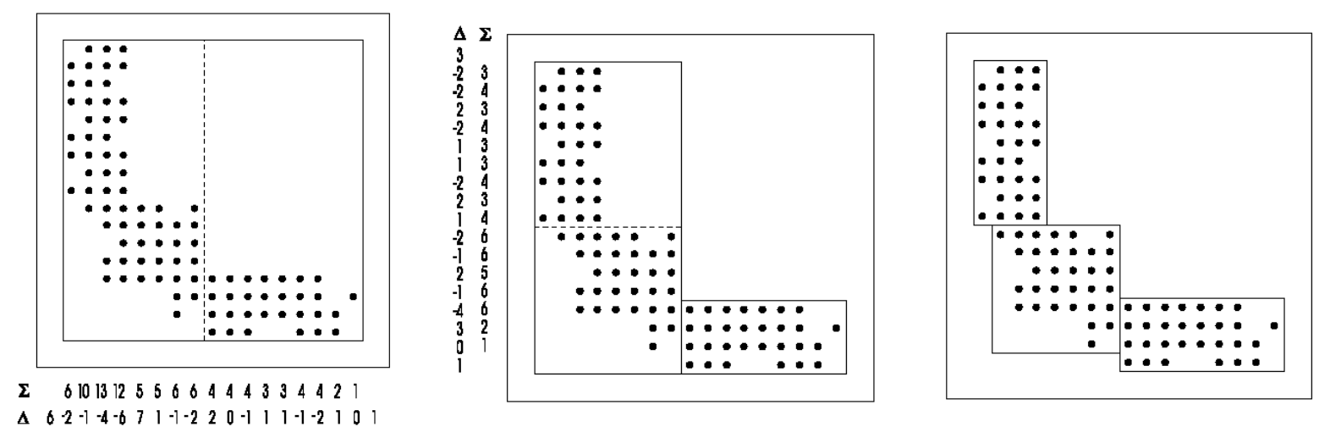
- Patches of arbitrary size
- Quad- / Oct-tree of refined grids



Advantages / Disadvantages

+ More effective covering
– Complicated data structures
– Difficult to generate optimal grids
– Harder load balancing on distributed memory archs

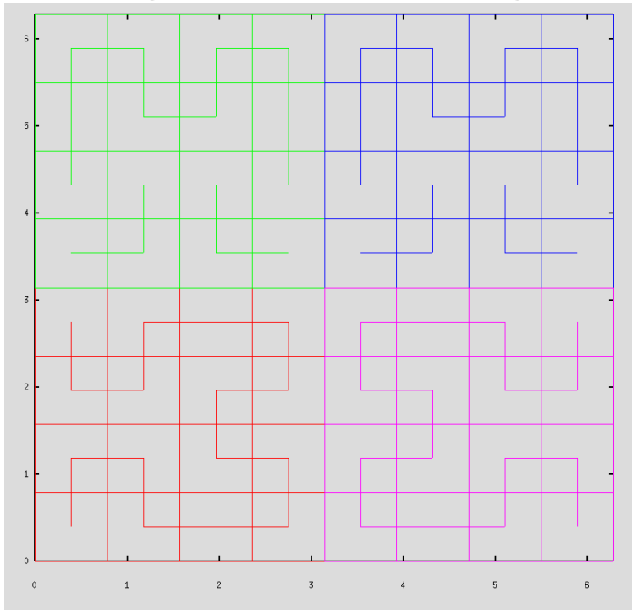



Δ
Σ
Σ 6 10 13 12 5 5 6 6 4 4 4 3 3 4 4 2 1
Δ 6 -2 -1 -4 -6 7 1 -1 -2 2 0 -1 1 1 -1 -2 1 0 1

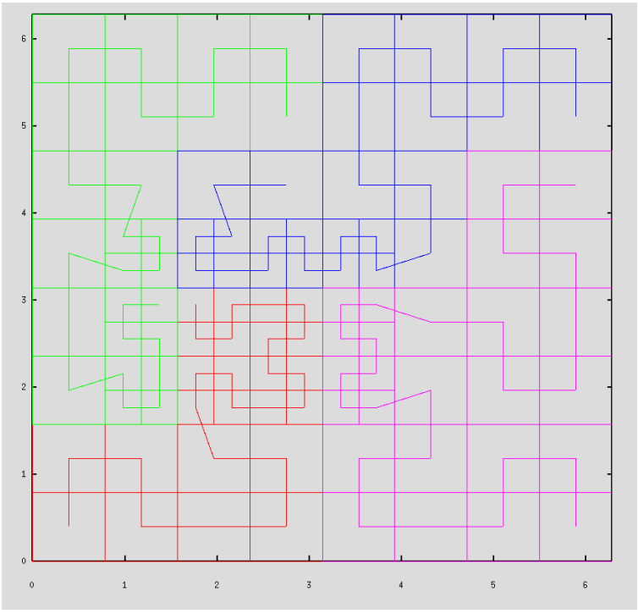
Methods from image processing used for finding optimal set of rectangular grids covering the underresolved points

1.3 Tree structured refinement and load balancing

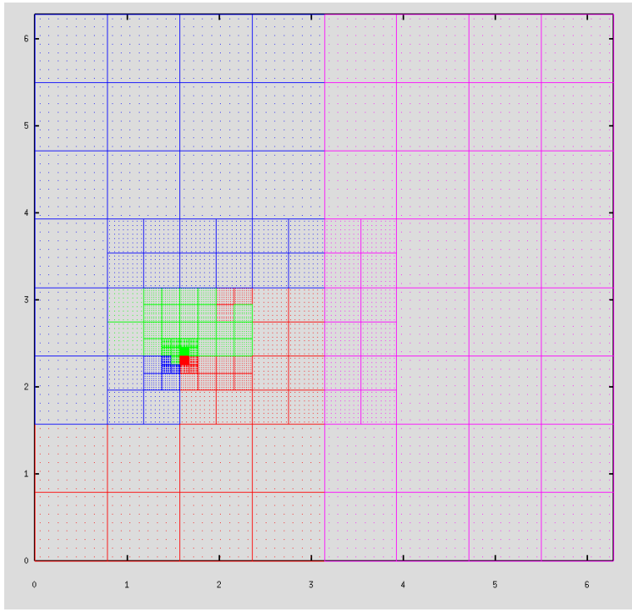
Shown is a domain $2\pi \times 2\pi$,
base level subdivided into
8×8 grids with 8×8 grid points each.



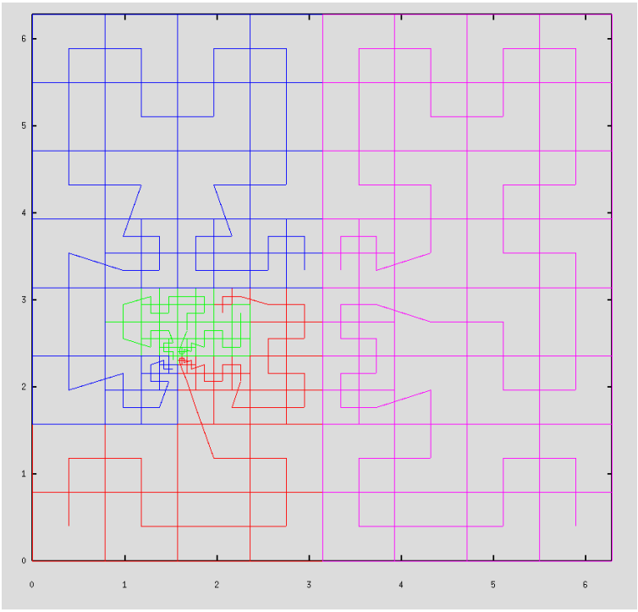
base level, load balanced to 4 processors, using the Hilbert-Peano space filling curve



one level of refinement, load balanced to 4 processors, using the space filling curve



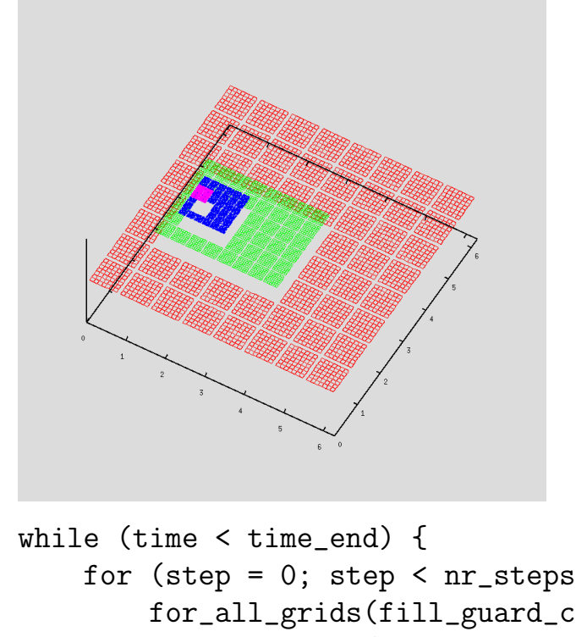
Four levels of refinement



Four levels of refinement, corresponding Hilbert-Peano curve

1.4 Time substepping

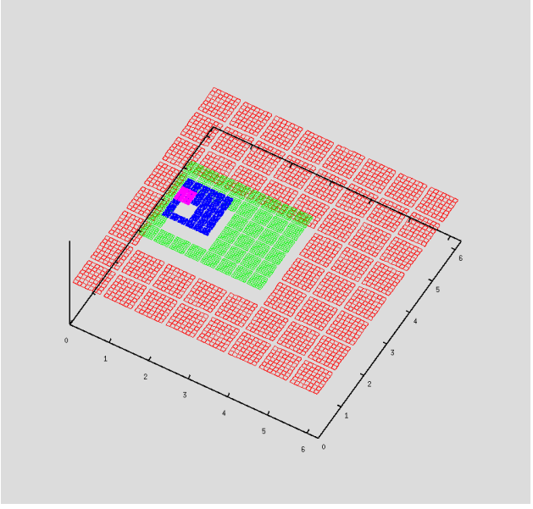
Adaptivity in space only, same timestep on all levels



Hyperbolic problem

$$\partial_t(\rho\mathbf{u}) + \mathbf{u}\cdot\nabla(\rho\mathbf{u}) = -\nabla p$$

```
while (time < time_end) {
    for (step = 0; step < nr_steps; step++)
        for_all_grids(fill_guard_cells);
        for_all_grids(do_substep, step);
    }
    time += dt;
}
```

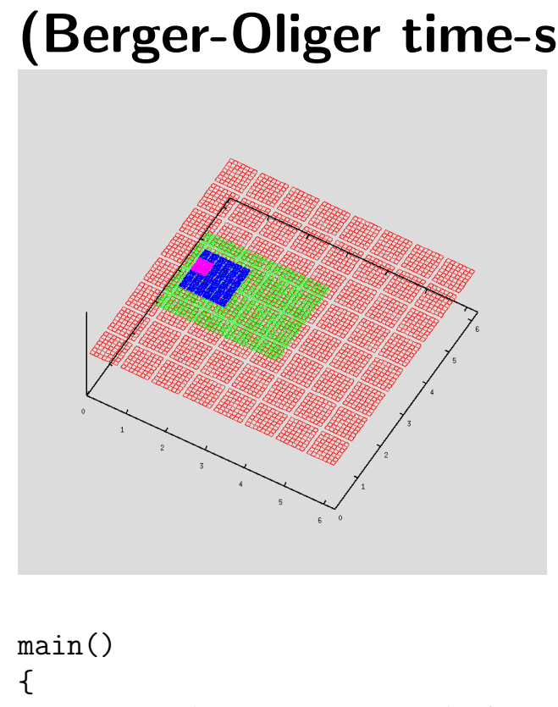


Elliptic problem

$$\partial_t\omega + \mathbf{v}\cdot\nabla\omega = \nu\nabla^2\omega$$
$$\omega = -\nabla^2\phi \quad \mathbf{v} = \hat{\mathbf{z}} \times \nabla\phi$$

```
while (time < time_end) {
    for (step = 0; step < nr_steps; step++)
        elliptic_solve();
        for_all_grids(fill_guard_cells);
        for_all_grids(do_substep, step);
    }
    time += dt;
}
```

Adaptivity in space and time (Berger-Oliger time-stepping)



Hyperbolic problem

$$\partial_t(\rho\mathbf{u}) + \mathbf{u}\cdot\nabla(\rho\mathbf{u}) = -\nabla p$$

```
main()
{
    while (time < time_end) {
        singlestep_on_level(0);
        time += dt;
}

void singlestep_on_level(int level)
{
    level_singlestep();
    time += dt;
    while (next_level->time < time) {
        singlestep_on_level(level + 1);
    }
    level_update_from(level + 1);
}
```

Advantages:

- More efficient, small steps on coarse levels are not necessary
- Possible to supply a level-independent CFL number

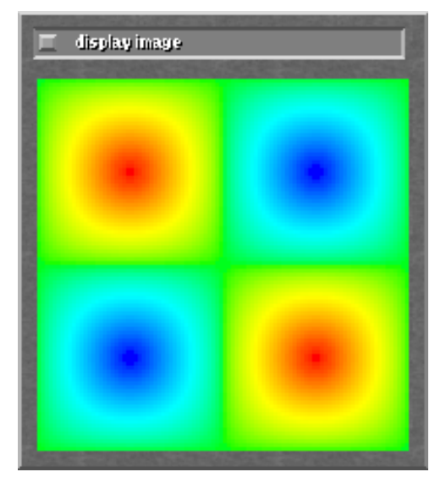
2 Elliptic solvers

2.1 Additive Schwarz Iteration

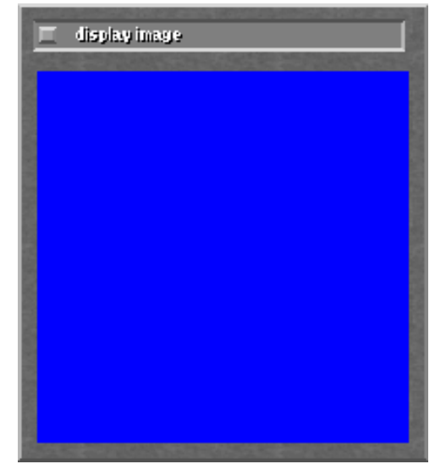
Example: $\omega = -\nabla^2\phi$, $\omega = 2\sin(x)\cos(x)$

Decomposition into 3 × 3 grids, 1 point overlap

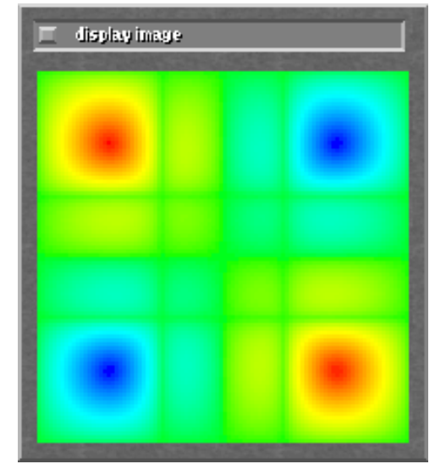
ω



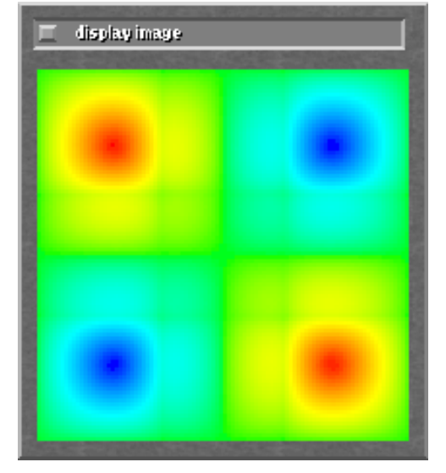
ϕ, 0. iteration



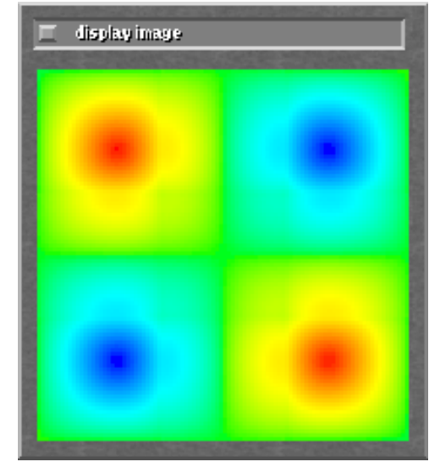
ϕ, 1. iteration



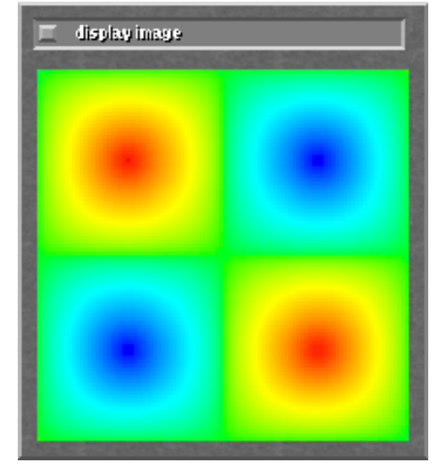
ϕ, 5. iteration



ϕ, 10. iteration

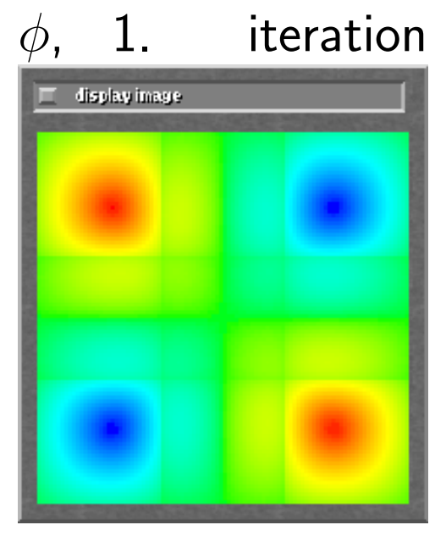


ϕ, 50. iteration

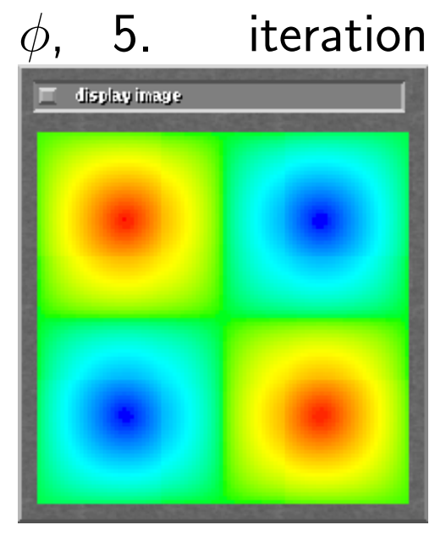


Decomposition into 3 × 3 grids, 4 points overlap

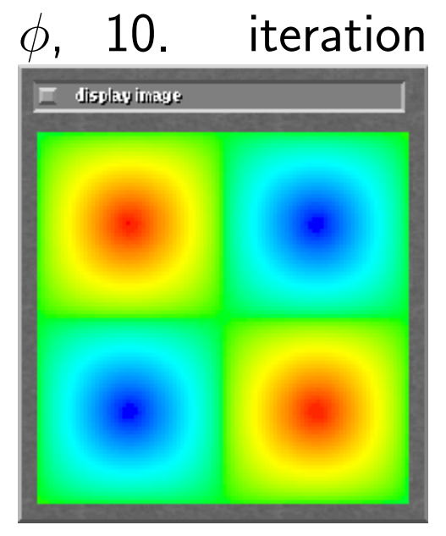
ϕ, 1. iteration



ϕ, 5. iteration



ϕ, 10. iteration



```
void level::schwarz_iteration()
{
    fill_external_boundary();
    range = calc_range();
    do {
        for_each_grid(poisson_solve);
        error = exchange_internal_boundary();
    } while (error/range > threshold);
}
```

Implementation in C / C++

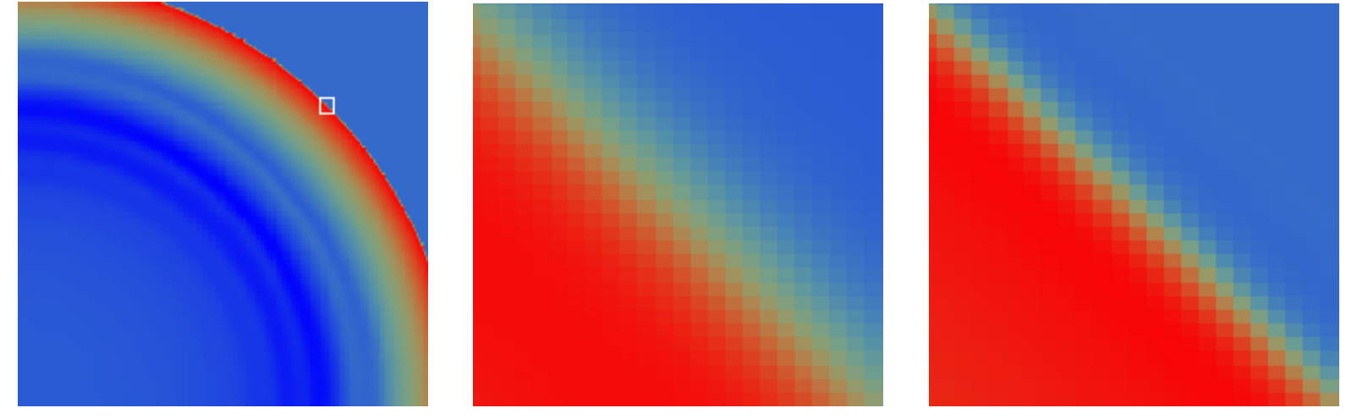
- Problem specific driver using generic library functions, rather than having a generic driver which uses callbacks: cleaner, less surprises, more flexible.
- Using a modern, object oriented language facilitates this approach.

3 Central weighted ENO

Nessyahu and Tadmor (1990), Kurganov and Levy (2000)

3.1 Application: Sedov-type explosion

Nessyaho-Tadmor vs. 3rd order CWENO (J. Dreher)



Why central schemes?

- no (approximate) Riemann solver necessary
- straightforward to generalize to multidimensional systems
- high order
- properties like WENO, monotone, TVD depend on appropriate reconstruction

3.2 Conservation laws

$$\frac{\partial}{\partial t}u(x,t) + \frac{\partial}{\partial_x}f(u(x,t)) = 0$$

Extensions to Lax-Friedrichs scheme:

$$u_j^{n+1} = \frac{u_{j+1}^n + u_{j-1}^n}{2} - \frac{\Delta t}{2\Delta x}\left(f(u_{j+1}^n) - f(u_{j-1}^n)\right)$$

$$\Longleftrightarrow \frac{u_j^{n+1} - u_j^n}{\Delta t} + \frac{1}{2\Delta x}\left(f(u_{j+1}^n) - f(u_{j-1}^n)\right) = \frac{(\Delta x)^2}{2\Delta t}\,\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}$$

(low order, dissipation depends on timestep)

Use cell averages for discretization:

$$\bar{u}_j^n \equiv \frac{1}{\Delta x}\int_{x_{j-1/2}}^{x_{j+1/2}} u(x,t^n)dx$$

$$\Longrightarrow \quad \bar{u}_j^{n+1} = \bar{u}_j^n - \frac{1}{\Delta x}\int_{t^n}^{t^{n+1}}\left[f(u(x_{j+1/2},\tau)) - f(u(x_{j-1/2},\tau))\right]\,d\tau$$

Piecewiese polynomial reconstruction:

$$u(x,t^n) \approx \sum_j P_j(x)\chi_{[x_{j-1/2},x_{j+1/2}]}$$

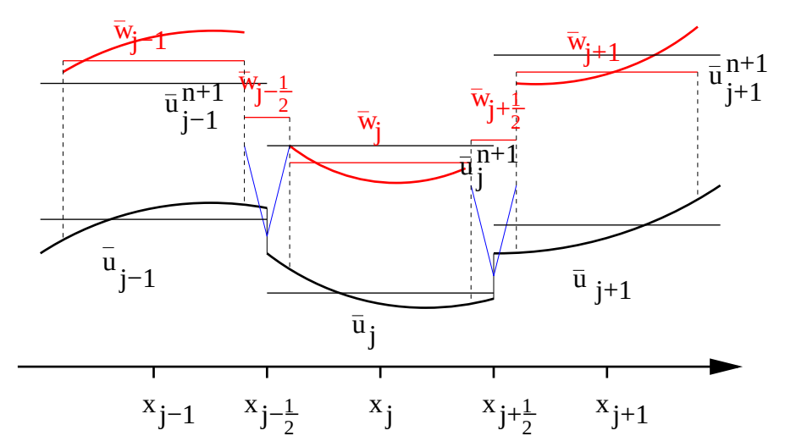
Using a constant reconstruction, we recover the (staggered) Lax-Friedrichs scheme, using a linear approximation gives the second order Nessyahu-Tadmor (NT) scheme. Limiting is necessary to prevent oscillations.

3.3 Third order CWENO (central weighted ENO)

Build reconstruction

$$P_j(x) = w_L P_L(x) + w_R P_R(x) + w_c P_C(x)$$

where the weights w favor $P_C(x)$ when the field is smooth and switch to the one-sided linear reconstructions in the presence of large gradients.



3.4 Transition from full-discrete to semi-discrete scheme

Consider the limit $\Delta t \longrightarrow 0$ to derive the semi-discrete scheme

$$\frac{d}{dt}\bar{u}_j(t) = \lim_{\Delta t\longrightarrow 0}\frac{\bar{u}_j^{n+1} - \bar{u}_j^n}{\Delta t}.$$

which is obtained as

$$\begin{aligned}\frac{d\bar{u}_j}{dt} \;=\; & -\frac{1}{2\Delta x}\left[f(u^+_{j+1/2}(t)) + f(u^-_{j+1/2}(t)) - f(u^+_{j-1/2}(t)) + f(u^-_{j-1/2}(t))\right] \\ & + \frac{a_{j+1/2}(t)}{2\Delta x}\left[u^+_{j+1/2}(t) - u^-_{j+1/2}(t)\right] + \frac{a_{j-1/2}(t)}{2\Delta x}\left[u^+_{j-1/2}(t) - u^-_{j-1/2}(t)\right]\end{aligned}$$

4 Divergence cleaning

Dedner et al (2002)

Initial condition:

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

Evolution of magnetic field: (ideal MHD)

$$\partial_t\mathbf{B} + \nabla\times(\mathbf{B}\times\mathbf{u}) = 0$$

Analytically, $\nabla\cdot(\nabla\times\cdot) \equiv 0$, but usually not in discretized numerical form.

Solutions:

- constrained transport methods
- Hodge projection
- truncation-error method

4.1 Hyperbolic divergence cleaning

Replace equation for magnetic field with:

$$\begin{aligned}\partial_t\mathbf{B} + \nabla\times(\mathbf{uB} - \mathbf{Bu}) + \nabla\psi &= 0 \qquad (1)\\ \mathcal{D}(\psi) + \nabla\cdot\mathbf{B} &= 0 \qquad (2)\\ \Longrightarrow \qquad \partial_t(\nabla\cdot\mathbf{B}) + \nabla^2\psi &= 0 \qquad (3)\end{aligned}$$

where $\mathcal{D}$ is a linear differential operator.

Choose $\mathcal{D}(\psi) \equiv 0$ (elliptic correction):

ψ is a Lagrange multiplier. For numerical solution, use two-step approach: First solve original system, obtaining $\mathbf{B}^{n*}$. Discretizing Eq.(3) in time:

$$-\nabla^2\psi^{n*} = \frac{1}{\Delta t}\left(\nabla\cdot\mathbf{B}^{n*} - \nabla\cdot\mathbf{B}^n\right) = \frac{1}{\Delta t}\nabla\cdot\mathbf{B}^{n*} \qquad (4)$$

which is solved for ψ and used to complete solving Eq. (1):

$$\mathbf{B}^{n+1} = \mathbf{B}^{n*} - \Delta t\nabla\psi^{n*} \qquad (5)$$

$$\begin{aligned}\partial_t\mathbf{B} + \nabla\times(\mathbf{uB} - \mathbf{Bu}) + \nabla\psi &= 0 \qquad (6)\\ \mathcal{D}(\psi) + \nabla\cdot\mathbf{B} &= 0 \qquad (7)\end{aligned}$$

Choose $\mathcal{D}(\psi) = \frac{1}{c_p^2}\psi$ (parabolic correction):

From Eqs. (6), (7) we obtain the heat equation

$$\partial_t\psi - c_p^2\nabla^2\psi = 0. \qquad (8)$$

Substituting $\mathcal{D}(\psi)$ into Eq. (7) gives ψ which we can plug into Eq. (6):

$$\partial_t\mathbf{B} + \nabla\times(\mathbf{uB} - \mathbf{Bu}) = c_p^2\nabla(\nabla\cdot\mathbf{B}) \qquad (9)$$

Choose $\mathcal{D}(\psi) = \frac{1}{c_h^2}\partial_t\psi$ (hyperbolic correction):

From Eqs. (6), (7) we obtain the wave equation

$$\partial_{tt}\psi - c_h^2\nabla^2\psi = 0. \qquad (10)$$

Local divergence errors are propagated to the boundary with the finite speed $c_h > 0$.

Choose $\mathcal{D}(\psi) = \frac{1}{c_h^2}\partial_t\psi + \frac{1}{c_p^2}\psi$ (hyperbolic/parabolic correction):

We obtain the telegraph equation

$$\partial_{tt}\psi + \frac{c_h^2}{c_p^2}\partial_t\psi - c_h^2\nabla^2\psi = 0. \qquad (11)$$

Local divergence errors are dissipated and propagated away.
The divergence constraint, equation (2) becomes

$$\partial_t\psi + c_h^2\nabla\cdot\mathbf{B} = -\frac{c_h^2}{c_p^2}\psi. \qquad (12)$$

5 2D Hall-MHD reconnection: the sawtooth instability

(Grasso/Pegoraro/Porcelli/Califano 1999)

5.1 Model equations

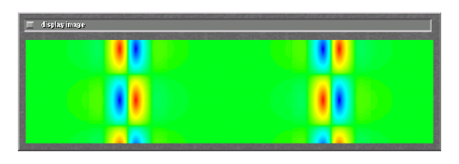
$$\begin{aligned}\partial_t F + [\phi, F] &= \rho_s^2[U,\psi]\\ \partial_t U + [\phi, U] &= [J,\psi]\\ F &= \psi + d_e^2 J\\ J &= -\nabla^2\psi \qquad\qquad \mathbf{B} = B_0\hat{\mathbf{z}} + \nabla\psi\times\hat{\mathbf{z}}\\ U &= \nabla^2\phi \qquad\qquad \mathbf{v} = \hat{\mathbf{z}}\times\nabla\phi\end{aligned}$$

with $[A,B] = \hat{\mathbf{z}}\cdot\nabla A\times\nabla B$.

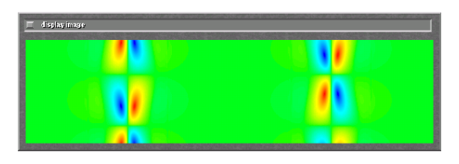
Equilibrium

$$\begin{aligned}\phi_{eq} &= U_{eq} = 0\\ \psi_{eq} &= J_{eq} = \cos(x) \quad , \quad F_{eq} = (1 + d_e^2)\cos(x)\end{aligned}$$

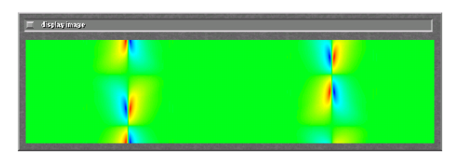
5.2 Case $\rho_s = 0$



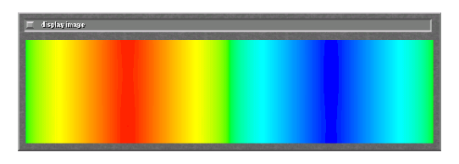
vorticity U
$t = 300$



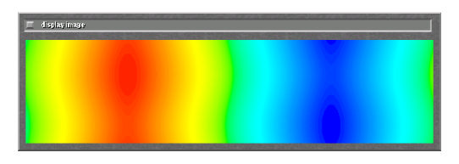
vorticity U
$t = 430$



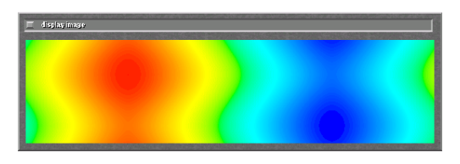
vorticity U
$t = 455$



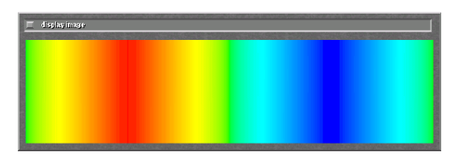
magnetic stream-function ψ
$t = 300$



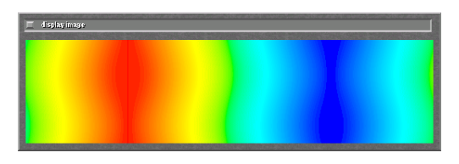
magnetic stream-function ψ
$t = 430$



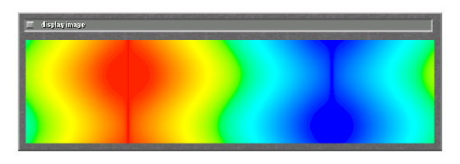
magnetic stream-function ψ
$t = 455$



F
$t = 300$

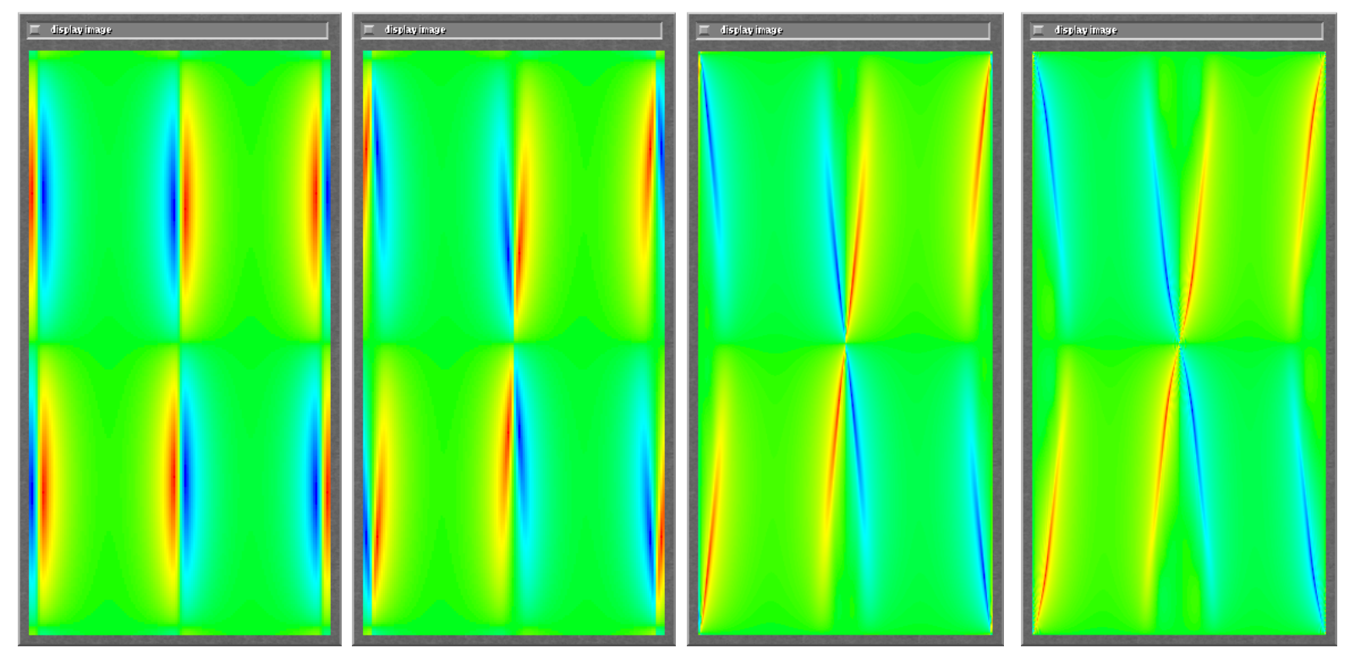


F
$t = 430$

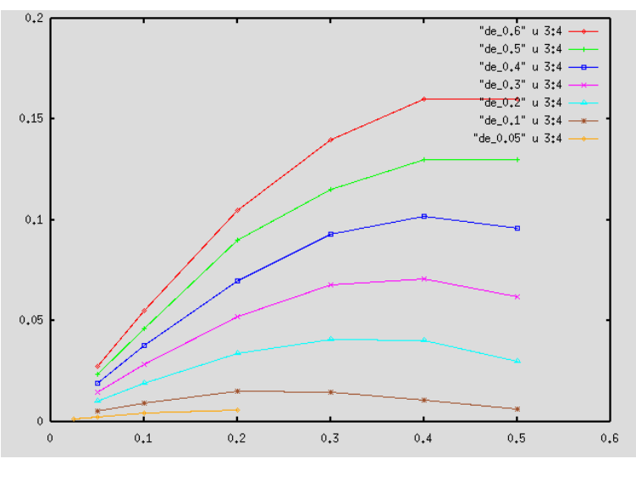


F
$t = 455$

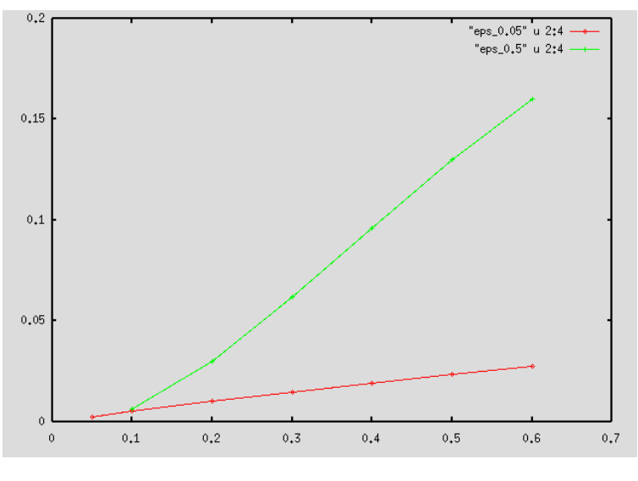
5.3 Case $\rho_s \neq 0$



Growth rate vs. d_e / aspect ratio

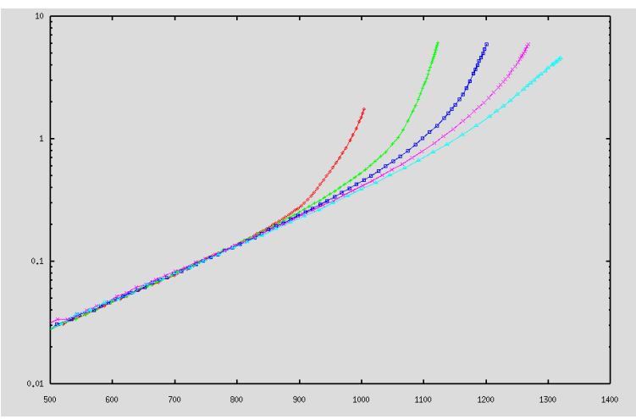


Growth rate vs. aspect ratio, different values of d_e.

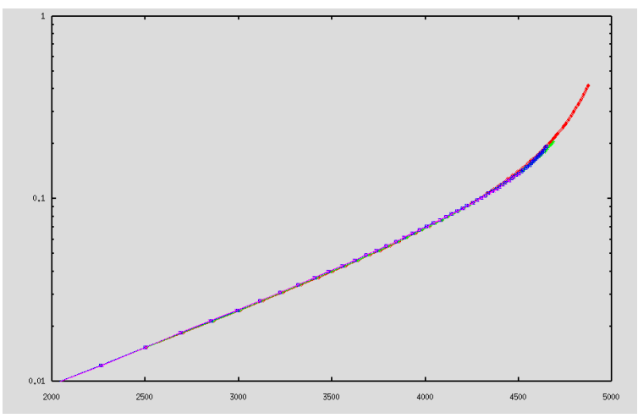


Growth rate vs. d_e, aspect ratio 0.5 and 0.05.

Time evolution of the island width against time
time axis normalized by linear growth rate



Case $\rho_s = 0.1 = \text{const}$
$d_e = 0.05,\ 0.1,\ 0.15,\ 0.2,\ 0.3$
(Red, green, blue, . . .)



Case $d_e = 0.025 = \text{const}$
$\rho_s = 0,\ 0.0125,\ 0.025,\ 0.05$
(Red, green, blue, . . .)

6 Implicit solvers

The example of reconnection in two-dimensional incompressible Hall-MHD is used to evaluate the trade-offs between explicit and implicit time stepping.
Being incompressible, the fast sound waves have already been filtered out of the problem, so that neither the explicit nor the implicit scheme need to handle them, for the explicit scheme this comes at the expense of solving elliptic problems at each time step. However, the explicit scheme is still limited by the Courant-Friedrichs-Levy stability criterion, necessitating small time time steps as spatial resolution increases. These time steps are smaller than necessary for the desired accuracy, since the reconnection phenomena take place at a slower time scale. On the other hand, the explicit time steps are much cheaper than implicit solves at not too large resolutions, making the explicit code the preferred approach. Since the implicit method is not constrained to time step limitations as solution increases and can be implemented to scale as $(O(n))$ for large problems using Newton-Krylov-Schwarz methods, we expect a break-even point to exist at which the implicit solver proves favorable to the explicit time stepping.
The set of equations that are solved by the implicit solver is

$$\begin{aligned}-\nabla^2\phi^{n+1} - U^{n+1} &= 0\\ (1 - d_e^2\nabla^2)\psi^{n+1} - F^{n+1} &= 0\\ \frac{U^{n+1} - U^n}{\Delta t} + \mathbf{v}\cdot\nabla U^{n+1} - \frac{1}{d_e^2}\mathbf{B}\cdot\nabla F^{n+1} - \nu\nabla^2 U^{n+1} &= 0\\ \frac{F^{n+1} - F^n}{\Delta t} + \mathbf{v}\cdot\nabla F^{n+1} - \rho_s^2\mathbf{B}\cdot\nabla U^{n+1} - \nu\nabla^2 F^{n+1} &= 0\end{aligned}$$

where $\mathbf{v} = \hat{\mathbf{z}}\times\nabla\phi^{n+1}$, $\mathbf{B} = \nabla\psi^{n+1}\times\hat{\mathbf{z}}$.
To compare these two fundamentally different algorithms, we are using the PETSc library, which is being optimized for the given problem in a collaboration with David Keyes / the TOPS group.