Compressible MHD numerical simulation techniques

Pekka Janhunen, FMI/Space
Graduate Summer School Lecture,
Mariehamn, 16 August 2010

Contents

- MHD (without simplifying assumption of incompressibility)

- Grid types
- Time discretisation
- Elliptic solvers

- Some future prospects ...

Fluid simulation: Euler equations

Primitive form:

$$\begin{aligned}\frac{\partial \rho}{\partial t} &= -\nabla \cdot (\rho \mathbf{v}) \\ \rho \frac{d\mathbf{v}}{dt} &= -\nabla P + \nu \rho \nabla^2 \mathbf{v} \\ \frac{d}{dt}\left(P \rho^{-\gamma}\right) &= 0\end{aligned}$$

Conservative form:

$$\begin{aligned}\frac{\partial \rho}{\partial t} &= -\nabla \cdot \mathbf{p} \\ \frac{\partial \mathbf{p}}{\partial t} &= -\nabla \cdot \left(\frac{\mathbf{p}\mathbf{p}}{\rho} + P\mathrm{I}\right) \\ \frac{\partial U}{\partial t} &= -\nabla \cdot \left[(U+P)\frac{\mathbf{p}}{\rho}\right]\end{aligned}$$

Conservative form guarantees correct shock speed and preserves conserved quantities to roundoff error when discretised using finite-volume method (FVM)

$$P = (\gamma - 1)\left(U - \frac{\mathbf{p}^2}{2\rho}\right)$$

MHD equations: primitive form

$$
\begin{aligned}
\frac{\partial \rho}{\partial t} &= -\nabla \cdot (\rho \mathbf{v}) \\
\rho \frac{d\mathbf{v}}{dt} &= \mathbf{j} \times \mathbf{B} - \nabla P \\
\frac{dP}{dt} &= -\gamma P \nabla \cdot \mathbf{v} \\
\frac{\partial \mathbf{B}}{\partial t} &= \nabla \times (\mathbf{v} \times \mathbf{B})
\end{aligned}
$$

where $\mathbf{j} \equiv (\nabla \times \mathbf{B})/\mu_0$ and $d/dt \equiv \partial/\partial t + \mathbf{v} \cdot \nabla$.

Primitive variable 8-tuple: (ρ,**v**,P,**B**)

γ=adiabatic index, usually γ=5/3; generally γ=(f+2)/f where f=number of degrees of freedom

The primitive form is the simplest to write down and remember

MHD equations: conservative form

$$
\begin{aligned}
\frac{\partial \rho}{\partial t} &= -\nabla \cdot \mathbf{p} \\
\frac{\partial \mathbf{p}}{\partial t} &= -\nabla \cdot \left[\frac{\mathbf{p}\mathbf{p}}{\rho} + \left(P + \frac{\mathbf{B}^2}{2\mu_0} \right) \mathrm{I} - \frac{1}{\mu_0} \mathbf{B}\mathbf{B} \right] \\
\frac{\partial U}{\partial t} &= -\nabla \cdot \left[\left(U + P - \frac{\mathbf{B}^2}{2\mu_0} \right) \frac{\mathbf{p}}{\rho} + \frac{1}{\mu_0 \rho} \mathbf{B} \times (\mathbf{p} \times \mathbf{B}) \right] \\
\frac{\partial \mathbf{B}}{\partial t} &= \nabla \times \left(\frac{\mathbf{p}}{\rho} \times \mathbf{B} \right) \qquad (1)
\end{aligned}
$$

- Variable 8-tuple is now $(\rho, \mathbf{p}=\rho\mathbf{v}, U, \mathbf{B})$
  where $U = P/(\gamma-1)+p^2/(2\rho)+B^2/(2\mu_0)$

MHD: Semiconservative form

$$
\begin{aligned}
\frac{\partial \rho}{\partial t} &= -\nabla \cdot \mathbf{p} \\
\frac{\partial \mathbf{p}}{\partial t} &= -\nabla \cdot \left(\frac{\mathbf{pp}}{\rho} + P\mathrm{I} \right) + \mathbf{j} \times \mathbf{B} \\
\frac{\partial U}{\partial t} &= -\nabla \cdot \left[(U + P) \frac{\mathbf{p}}{\rho} \right] + \mathbf{j} \cdot \mathbf{E} \\
\frac{\partial \mathbf{B}}{\partial t} &= \nabla \times \left(\frac{\mathbf{p}}{\rho} \times \mathbf{B} \right)
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{p} &\equiv \rho \mathbf{v} \\
\mathbf{j} &\equiv (1/\mu_0) \nabla \times \mathbf{B} \\
\mathbf{E} &\equiv \mathbf{B} \times \mathbf{v} \\
U &\equiv \frac{P}{\gamma - 1} + \frac{1}{2} \rho \mathbf{v}^2
\end{aligned}
$$

Non-conservative electromagnetic force and energy flux added toe conservative fluid (Euler) equation

Does *not* guarantee correct shock speed or exact energy and momentum conservation

Non-idealities in MHD

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot (\rho \mathbf{v}) + D_\rho \nabla^2 \rho$$

$$\rho \frac{d\mathbf{v}}{dt} = \mathbf{j} \times \mathbf{B} - \nabla P + \nu \nabla^2 (\rho \mathbf{v})$$

$$\frac{dP}{dt} = -\gamma P \nabla \cdot \mathbf{v} + D_P (\gamma - 1) \nabla^2 U$$

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{v} \times \mathbf{B}) - \nabla \times \left(\frac{\mathbf{j}}{en} \times \mathbf{B} \right) + \frac{\eta}{\mu_0} \nabla^2 \mathbf{B}$$

Hall-term

- note that $\mathbf{j} = en(\mathbf{v}_i - \mathbf{v}_e)$; in MHD, $v = \mathbf{v}_i$, thus **B** is frozen into electron flow when Hall-term included
- if diffusion parameters are not constant, they must appear inside gradient

FVM Godunov-type methods

- FVM = Finite-volume method:
  Store ***volume averages*** of quantities over each computational cell (instead of pointwise values)
- Equations of the form ***du/dt = -div(something)***
  are naturally discretised with FVM using cell interface fluxes. This yields automatically to a ***conservative discretisation***.
- Godunov-type method:
  - Propagate staircase FVM representation exactly
  - Compute new cell averages from propagated state
  - Godunov-type method can be written down with interfaces fluxes only (no cell averaging needed)

Riemann solvers

- “Riemann solver” = Solution of initial-value problem with stepwise initial data
- In 1-D Euler equations, even exact Riemann solver is known
- In 1-D MHD, practically usable exact solver does not exist
- In 2-D and 3-D, no exact solvers, but alternating direction discretisation works rather well in practice
- An exception: div(B)=0 constraint is problematic in MHD in 2-D and 3-D since it is essentially multidimensional

MHD wavemodes

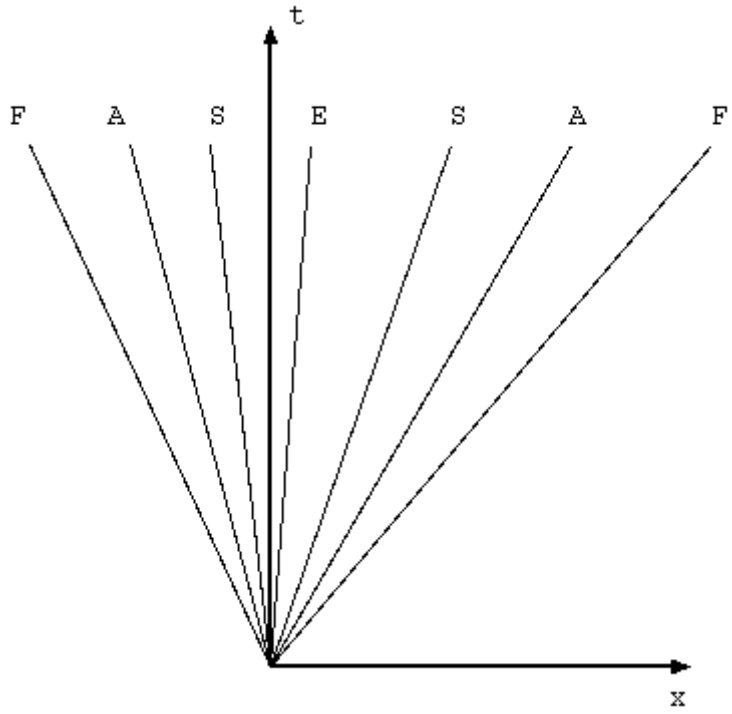
$$\lambda_e = v_k \qquad \text{Entropy wave}$$

$$\lambda_A = v_k \pm v_A |\cos\alpha| \qquad \text{Alfven wave}$$

$$\lambda_f = v_k \pm v_f \qquad \text{Fast wave}$$

$$\lambda_s = v_k \pm v_s \qquad \text{Slow wave}$$

$$\lambda_0 = 0 \qquad \text{Null wave}$$



where v_A is the scalar Alfvén speed,

$$v_A = \frac{|\mathbf{B}|}{\sqrt{\mu_0 \rho}},$$

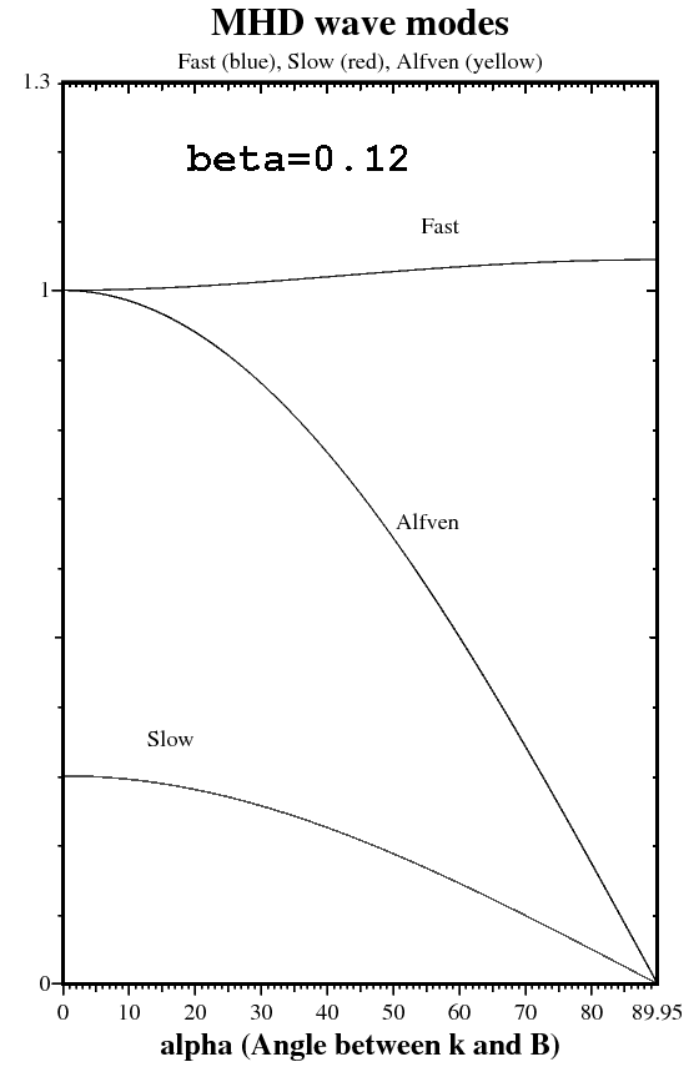
v_f is the so-called fast wave speed,

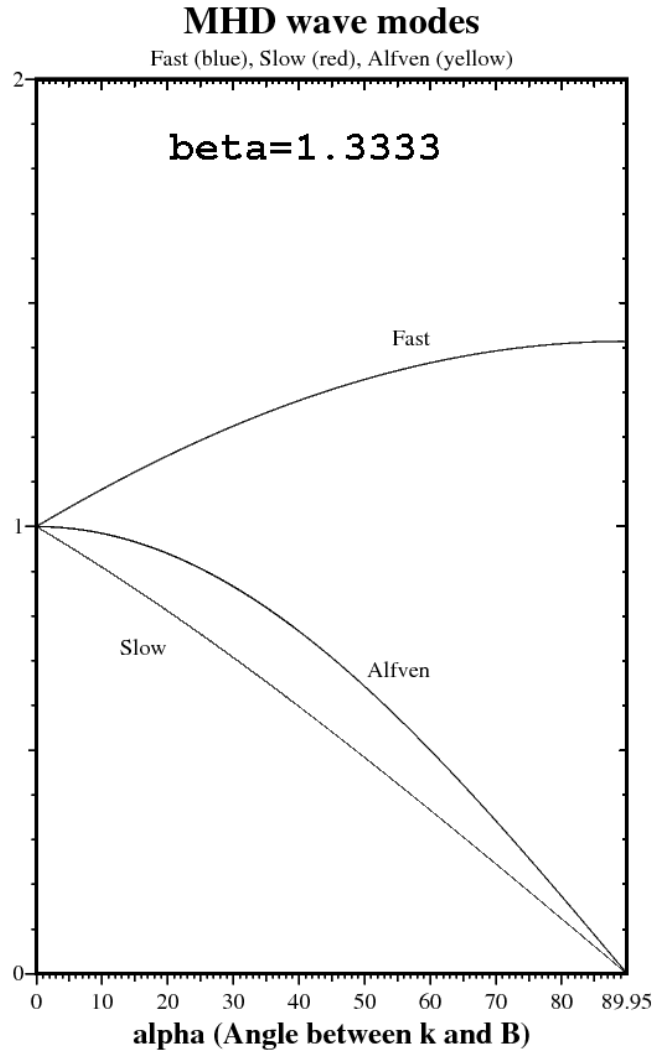
$$v_f^2 = \frac{1}{2}\left[c_s^2 + v_A^2 + \sqrt{\left(c_s^2 + v_A^2\right)^2 - 4c_s^2 v_A^2 \cos^2\alpha}\right]$$

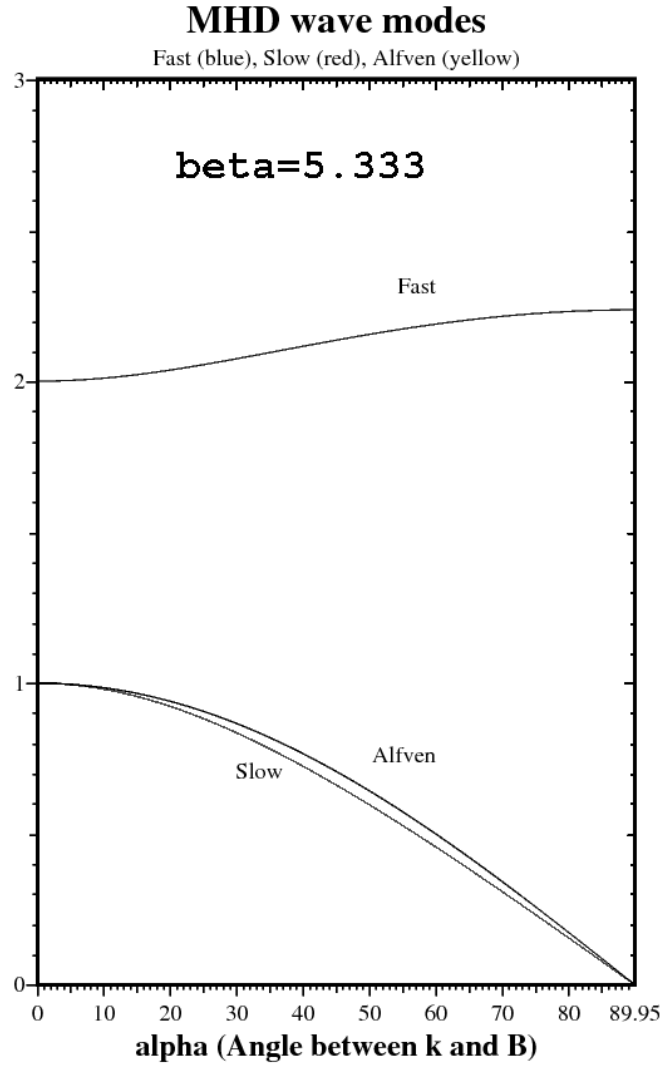
and v_s is the so-called slow wave speed,

$$v_s^2 = \frac{1}{2}\left[c_s^2 + v_A^2 - \sqrt{\left(c_s^2 + v_A^2\right)^2 - 4c_s^2 v_A^2 \cos^2\alpha}\right].$$

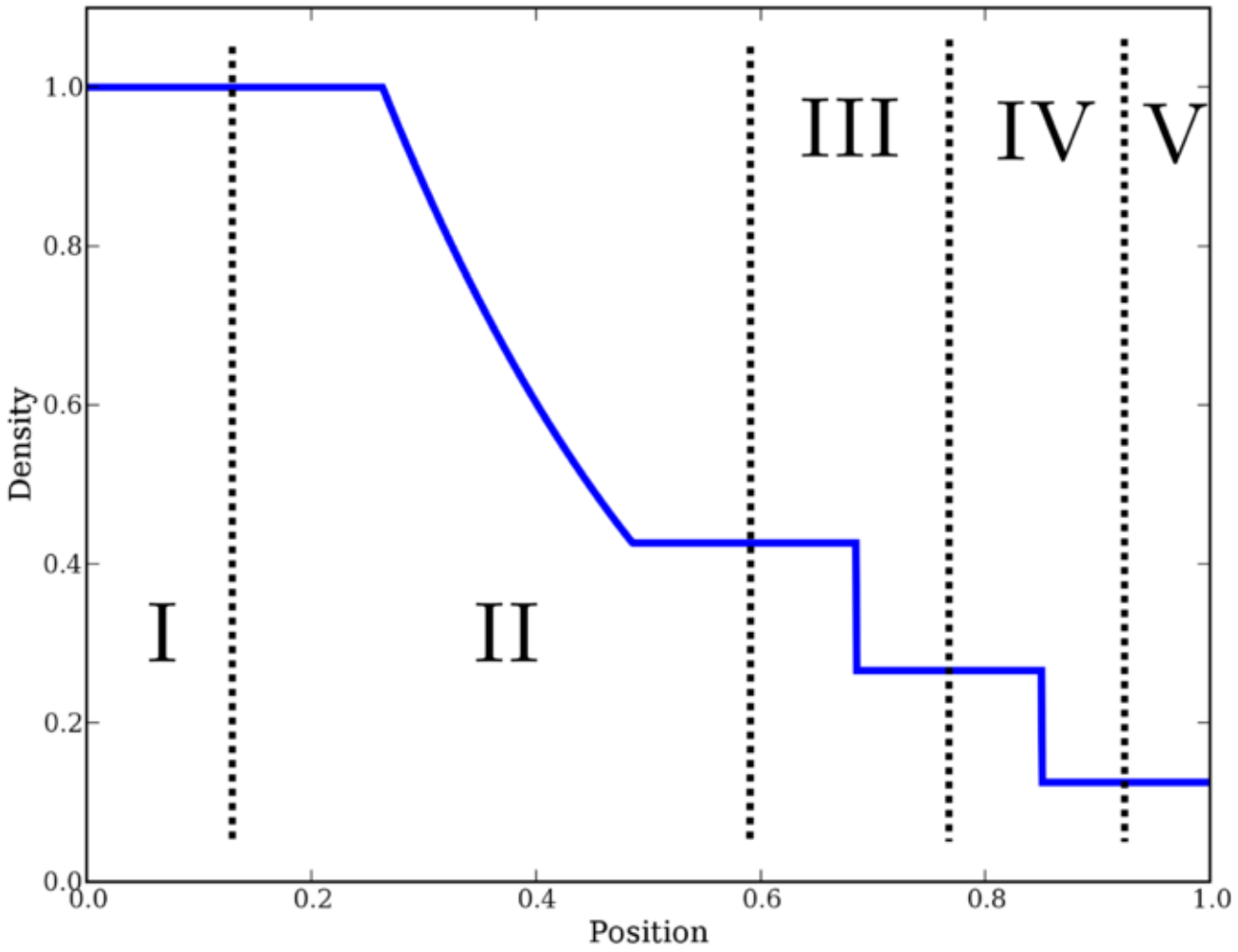
MHD wavemodes 2







Hydrodynamic Sod shock tube



1-D approximate Riemann solvers

- Harten-Lax-vanLeer (HLL) solver:
  Uses one intermediate state which propagates away from the discontinuity and which is defined as the spatial average of the *exact* Riemann problem solution
- Easy to construct for any equation system, only interface fluxes needed
- If continuum problem preserves positivity, so does the HLL-discretisation (!)
- Drawback: Has high diffusion
- Recent developments: Generalise HLL to have more intermediate states

1-D approx. Riemann solvers, cont'd

- ***Roe's approximate Riemann solver:***
  Linearise equations around the interface
  and solve the linearised Riemann problem
  exactly
  - Need to specify the averaging scheme used (one possibility: "√ρ-averaging = Roe-averaging")
  - Needs to be able to solve the linearised eigensystem (eigenvalues, left and right eigenvectors analytically); this is possible but rather cumbersome for MHD
  - Can yield negative pressures if the interface jump is large

MHD monopole removal methods

- So-called ***elliptic cleaning:***
  write **B'** = **B** + grad(ΔΦ) and require div(**B'**)=0, which gives Poisson equation for scalar field ΔΦ, with div(**B**) as source term.
  May produce negative pressures since P depends on **B**. One way to fix is to break energy conservation locally in these (hopefully rare) cases.
- Yee-mesh method: store **B** as interface surface averages (only normal component stored on each surface). Breaks conservation, method no longer fully FVM.

MHD monopole removal methods 2

- “Convect” monopoles to the boundaries where they are absorbed
  - Powell's method: apparently works, but is not consistent with natural extension of Maxwell's equations with monopoles
  - Janhunen's method tries to fix this
- New, clever methods which try to combine conservation with div(B)=0 (ongoing research)

Origin of the monopole problem

- If problem is solved alternating in X,Y,Z, in each 1-D subproblem the 1-D div(**B**) is just dB_x/dx etc. which can be zero only if **B**-field is trivial. Thus, the 1-D subproblems always "see" nonzero div(**B**), even if the true 3-D div(**B**) vanishes in some discrete sense.
- Continuum MHD equations have *no solution* if the initial state has nonzero div(**B**)
- Adding monopoles to MHD theory at the fundamental level is one possibility, and has given rise to some new numerical methods

Domain of validity of MHD

- Euler equations hold for scales much larger than collision mean free path (counterexample: re-entry vehicle physics)
- Likewise for MHD, but in addition, in the perpendicular direction it suffices scale to be much larger than ion Larmor radius
- Non-MHD scale processes, if nonlinear, may have global consequences that make MHD invalid *at all scales* in principle. (This holds also in fluid dynamics.)
- Mass, momentum and energy conservation described by MHD are exactly valid. To the extent these fix the solution, MHD is good.

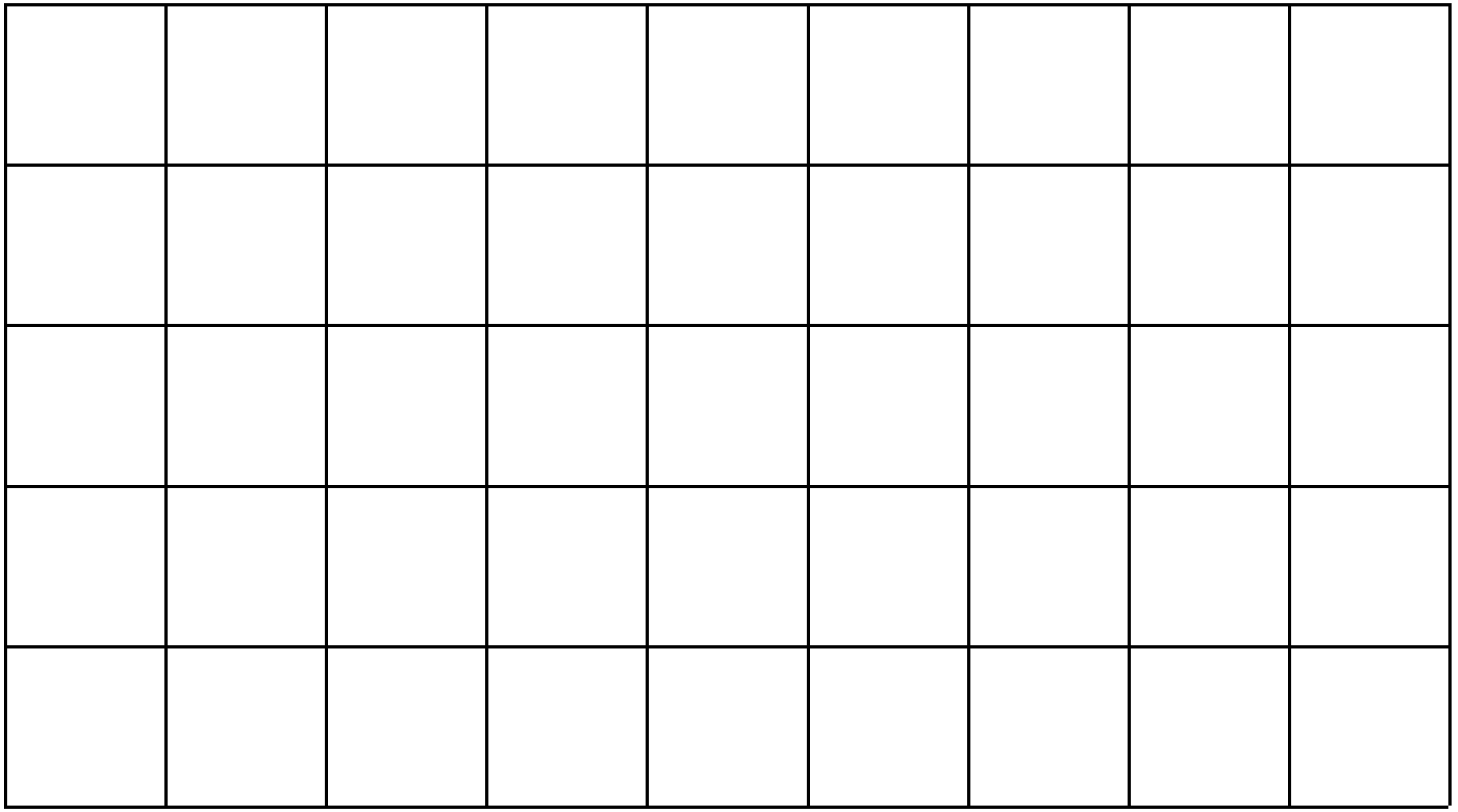
MHD implementation issues

- Serious MHD code typically must use some form of *grid adaptation*, which inevitably makes the implementation rather complex.
- To handle the complexity, careful planning and good language (***C++***) are needed. Heavy hand-optimisation of time-critical code, although desirable, is not always realistic because of the complexity.
- MHD is never very simple to program, not even on uniform grid. Even in fluid dynamics there has been room for commercial software industry (Fluent, Fidap, Elmer...), and FD is just a subset of MHD.

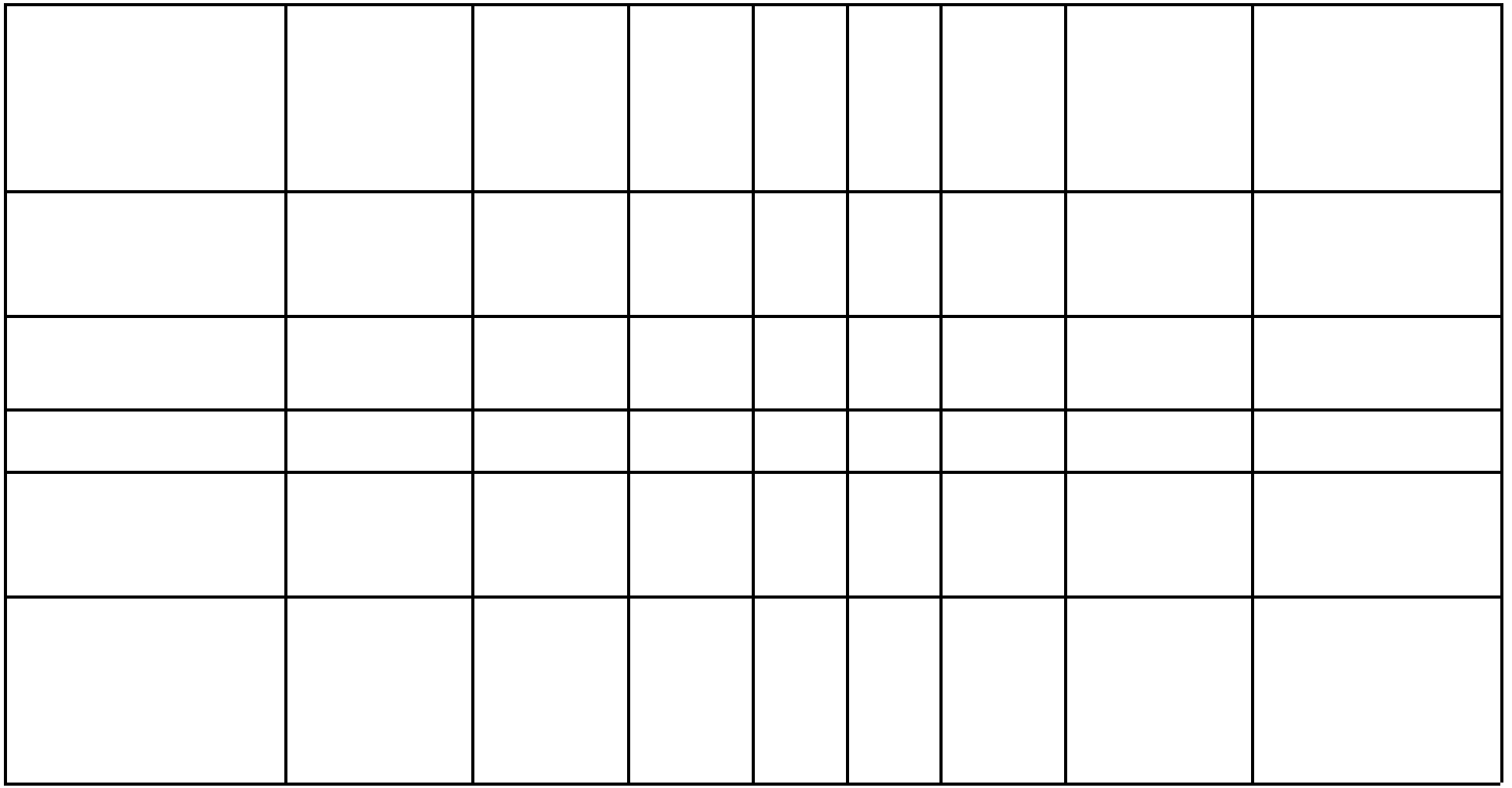
Adaptive mesh techniques

- Grid types
  - uniform grid
  - stretched or deformed uniform grid
  - cell-by-cell hierarchically refined cubic grid
  - block-by-block refined, locally uniform grid
  - fully general grid containing arbitrary-shaped cells
- Grid can be fixed in time (*adapted* grid) or change dynamically during the run (*adaptive* grid). In the latter case, the grid refinement and coarsening may be based on the solution alone (*fully automatic*) or include a user-specified component (*semiautomatic*).

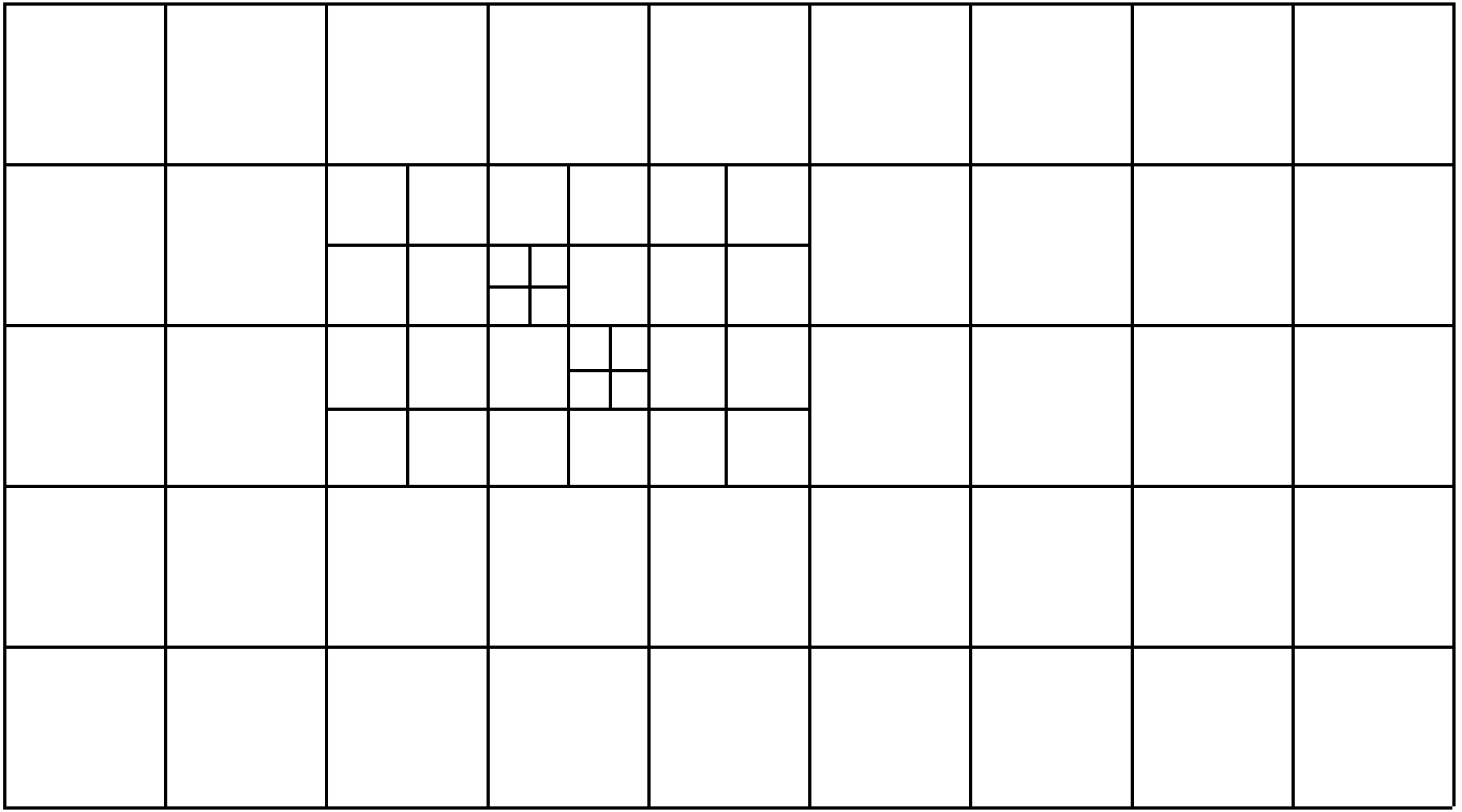
Uniform grid



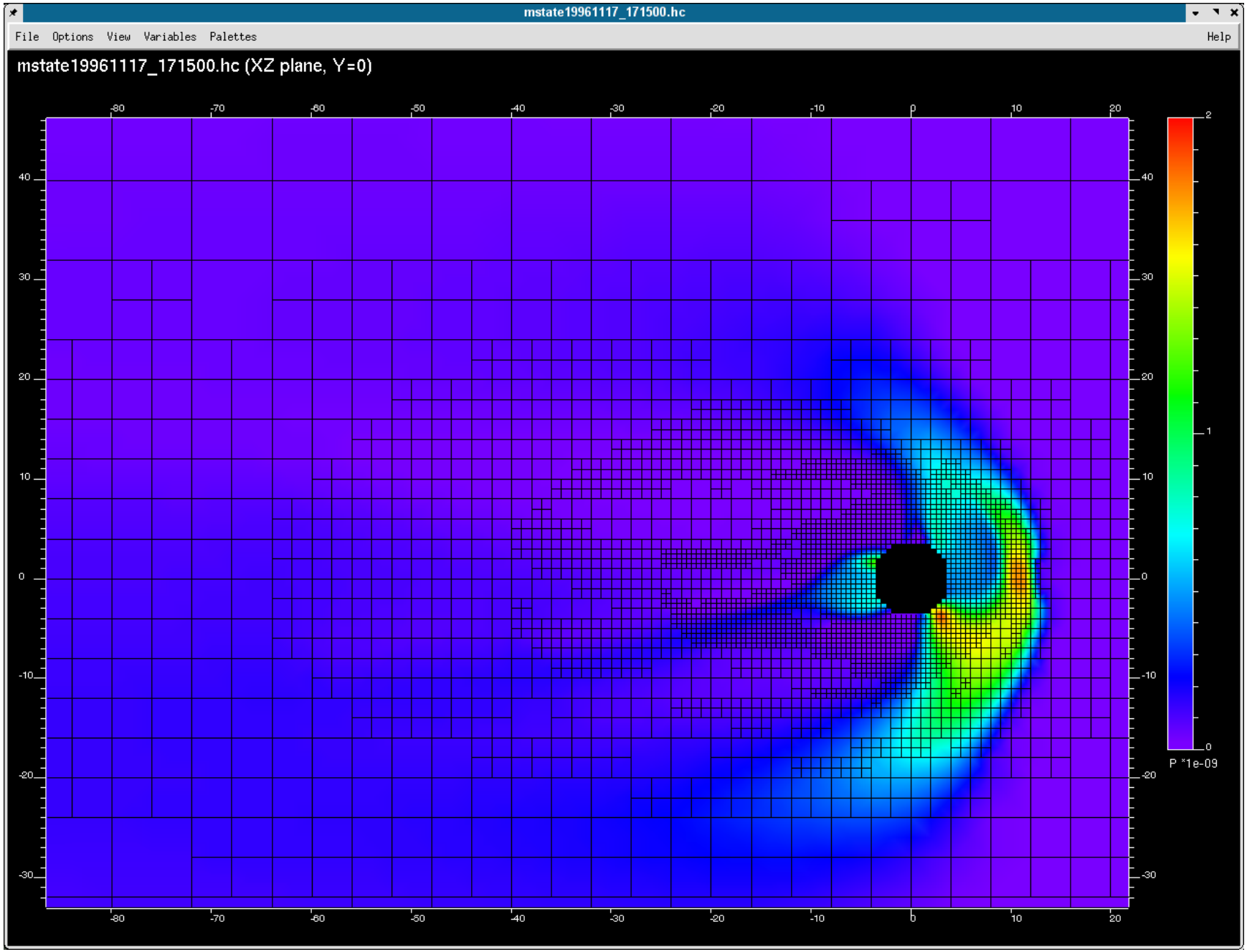
Stretched grid



Hierarchical Cartesian (HC) grid



Hierarchically refined cubic grid



Block-refined grids

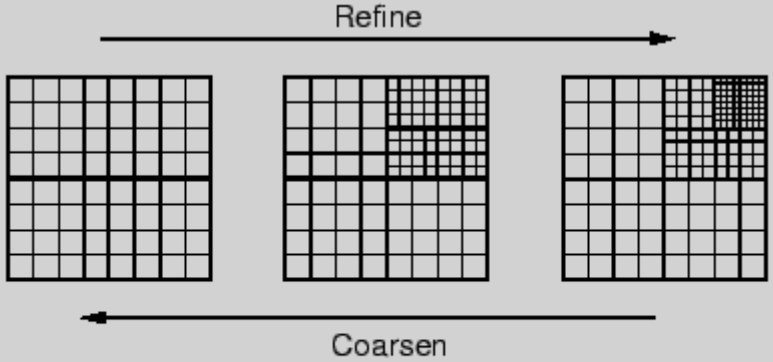


Figure 3.2: 2D example of the refining and coarsening of blocks that are 4x4 cells in size.

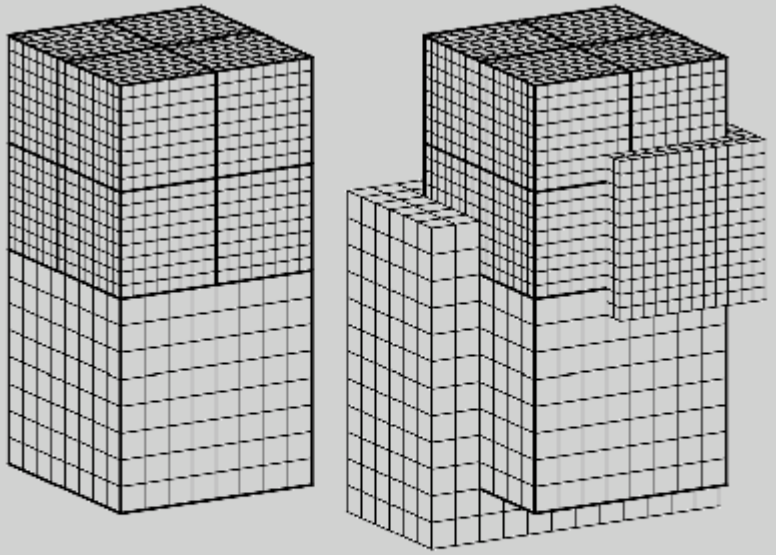
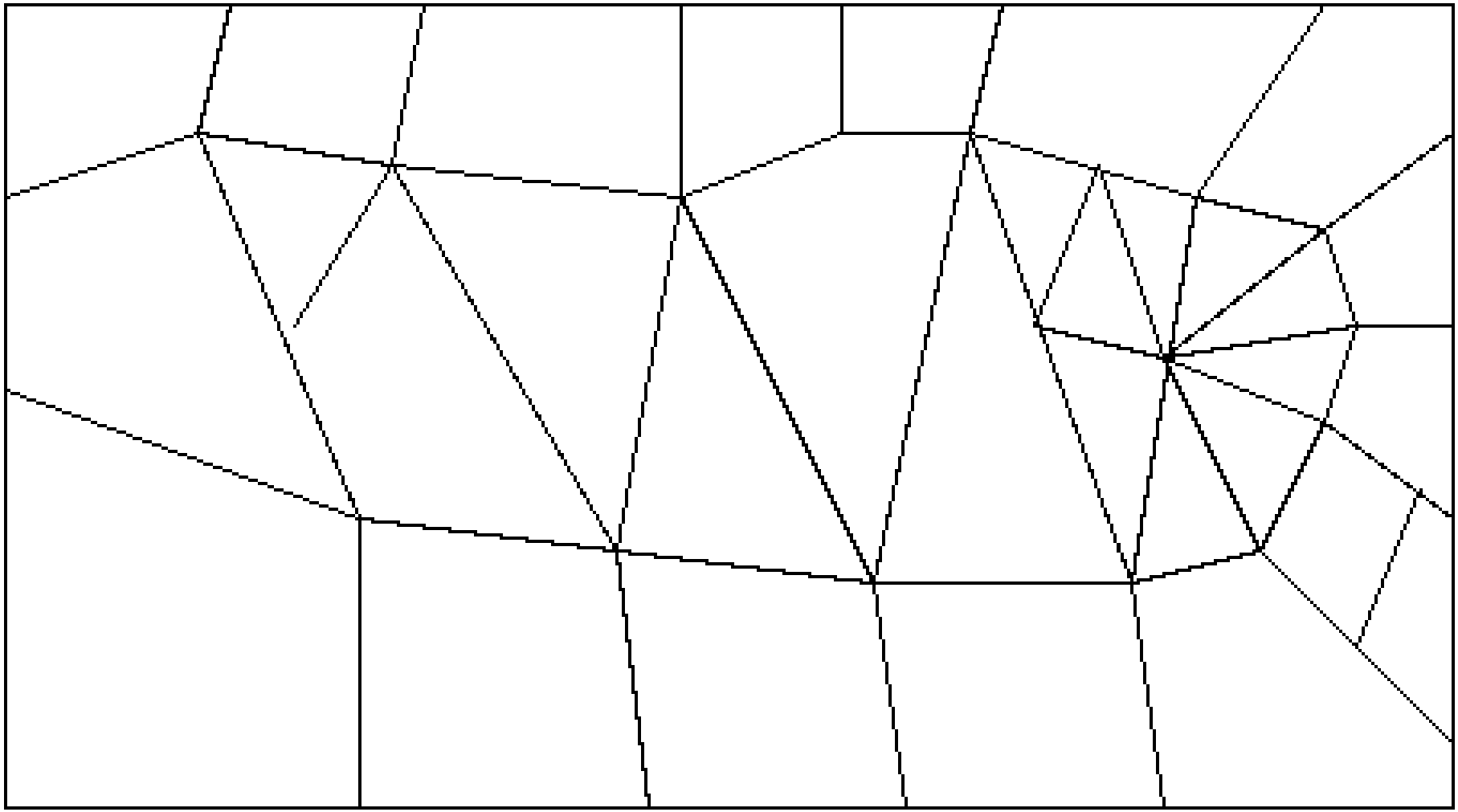


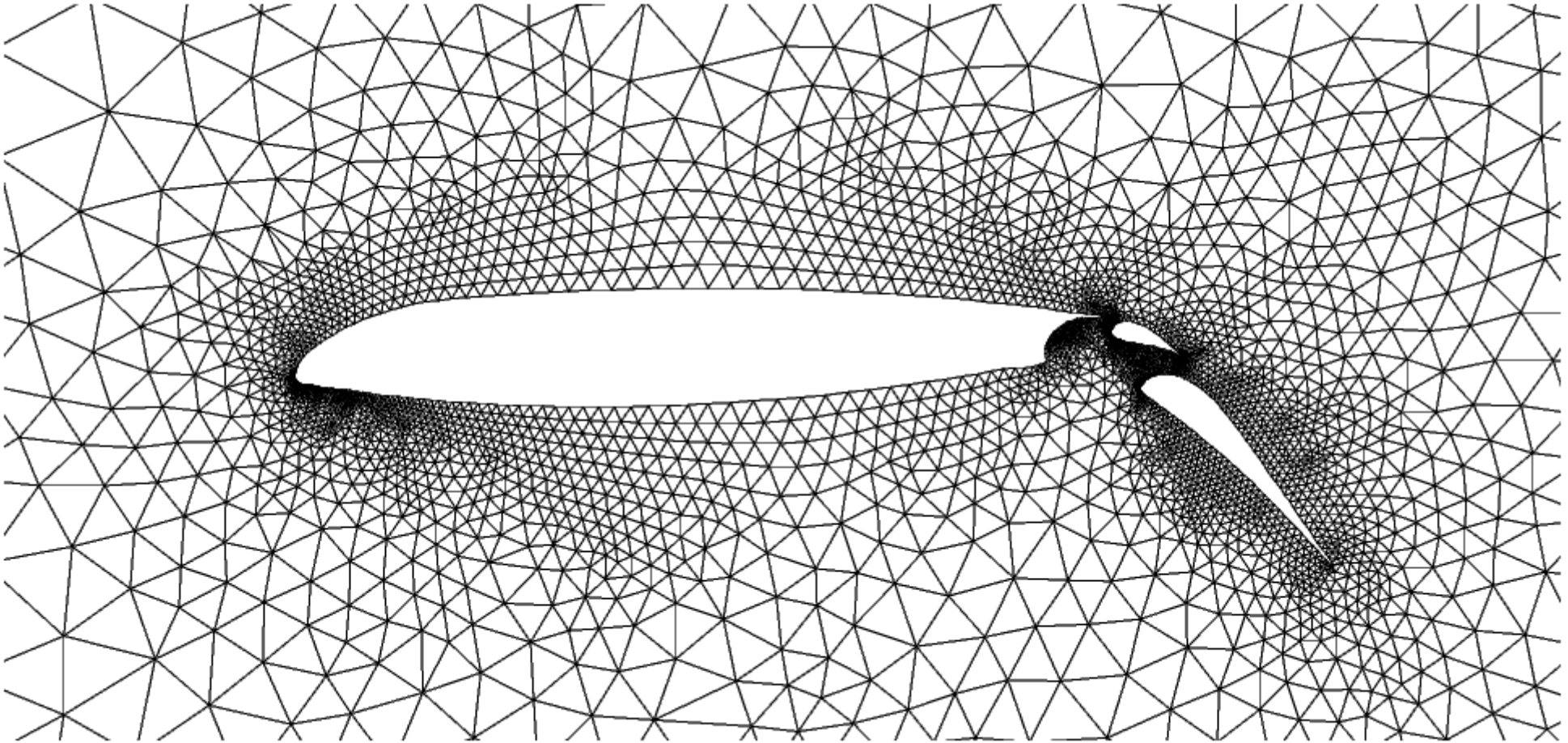
Figure 3.1: (left) Self-similar blocks used in parallel block-based AMR scheme. (right) Self-similar blocks illustrating the double layer of ghost cells for both coarse and fine blocks.

- Block-refined grids are locally uniform and thus fast to traverse; on the other hand there is some overhead because some cells are refined unnecessarily

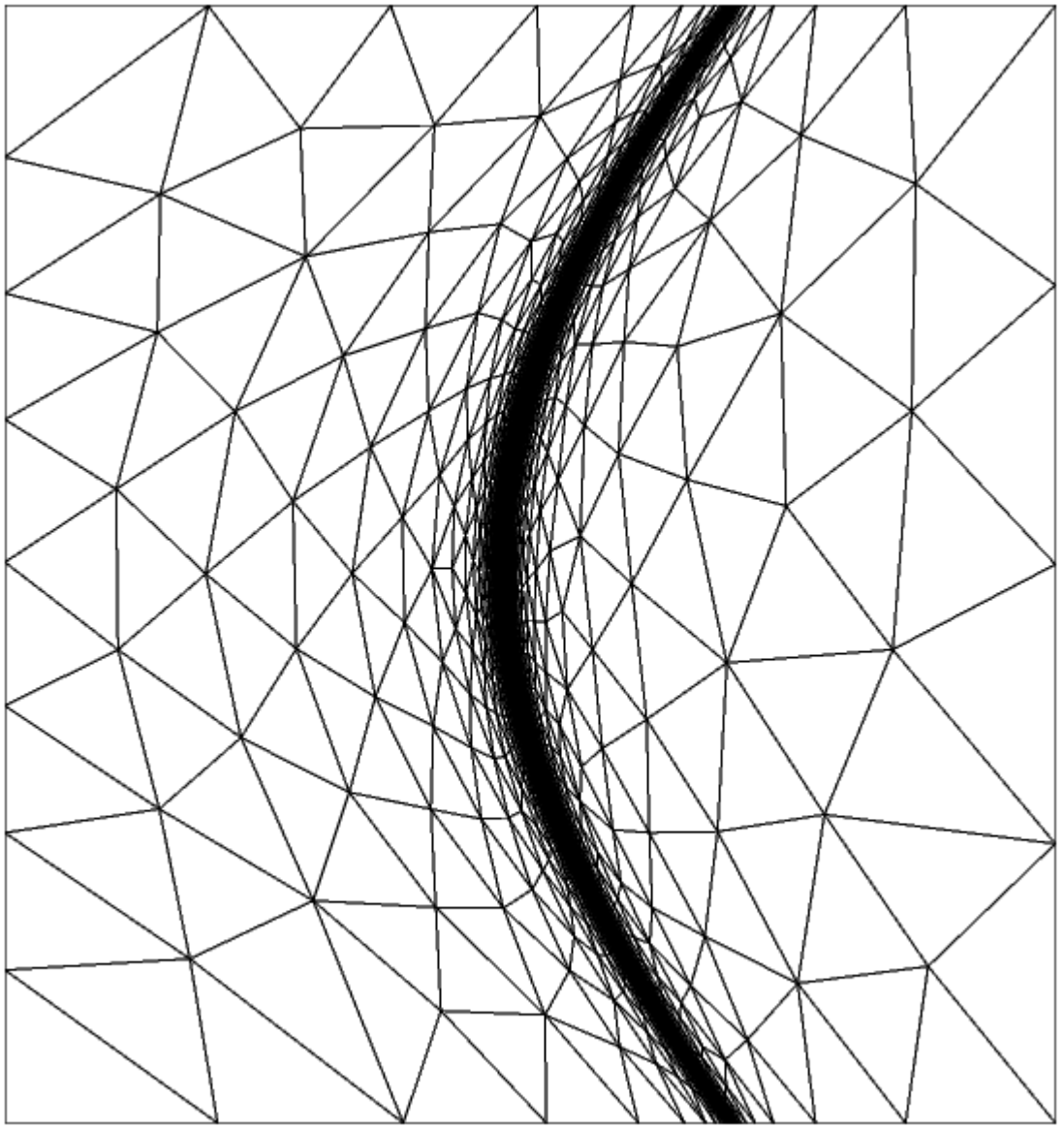
General grid



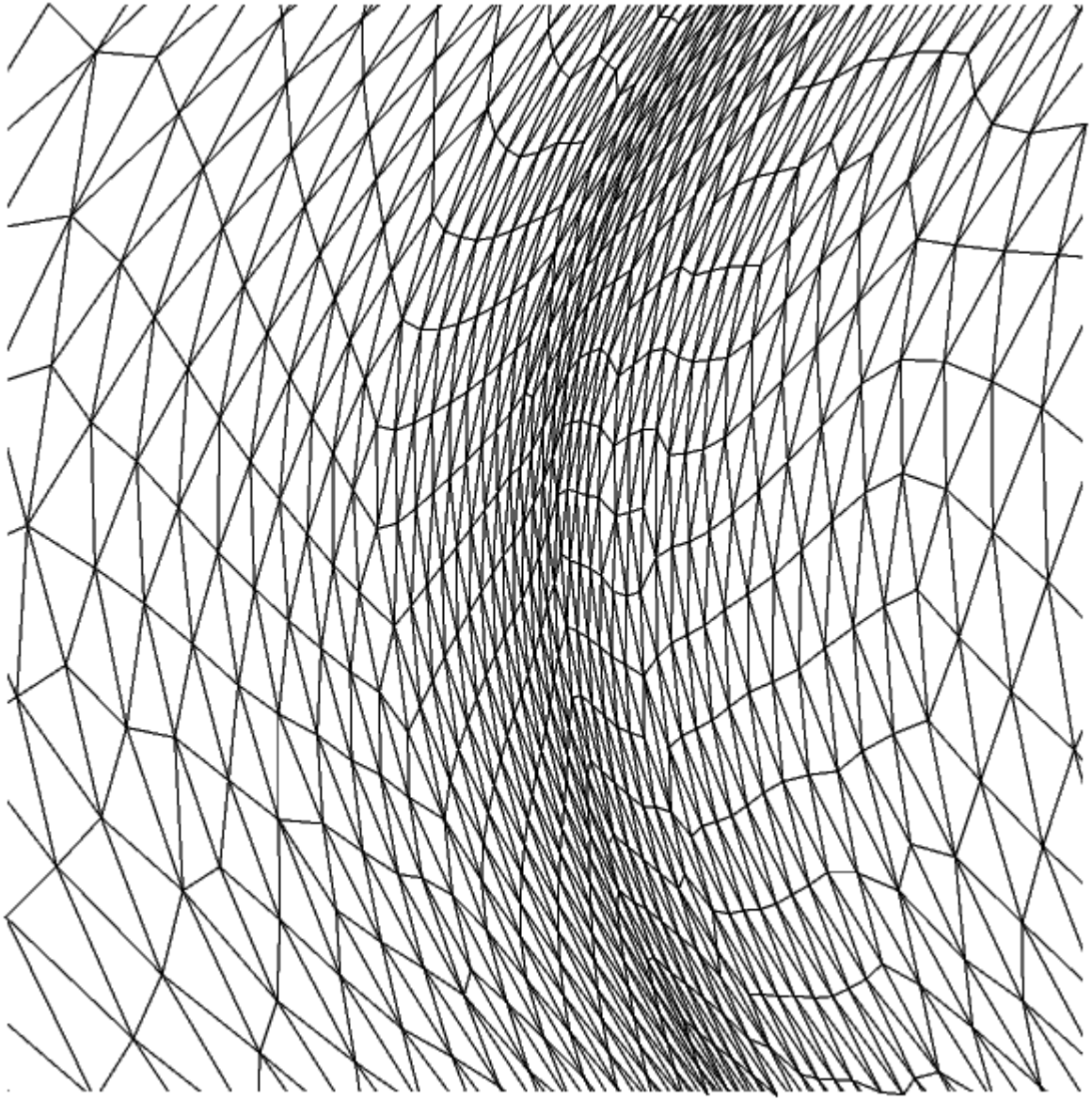
General grid

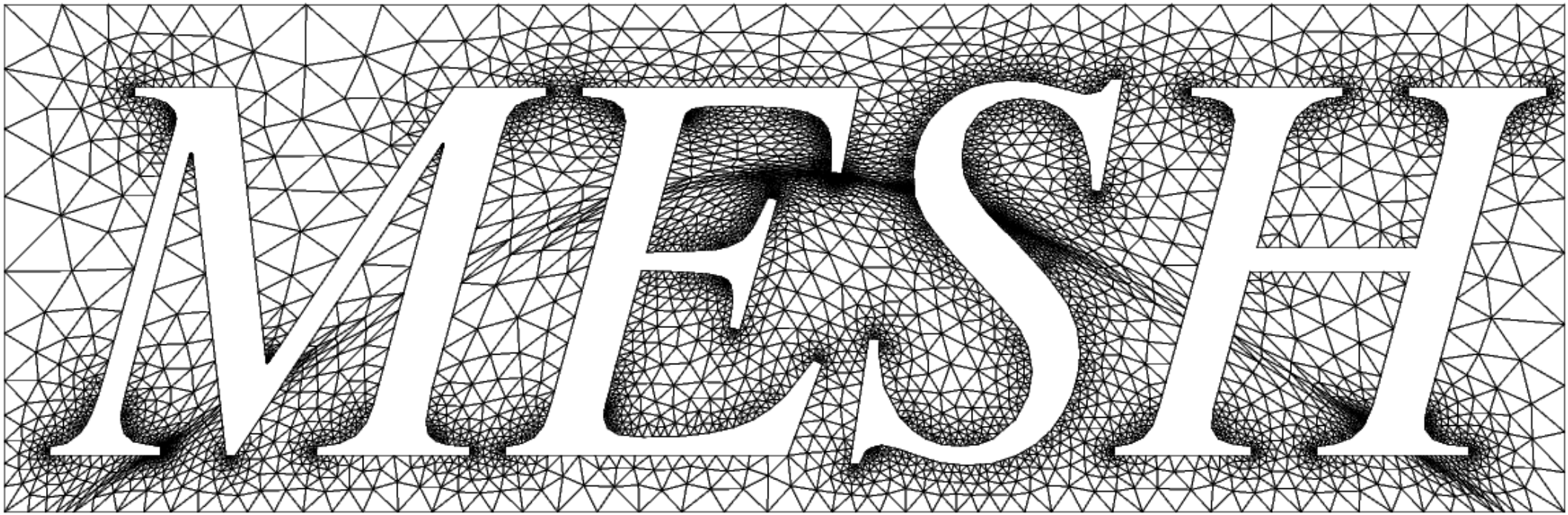


Anisotropic, general grid

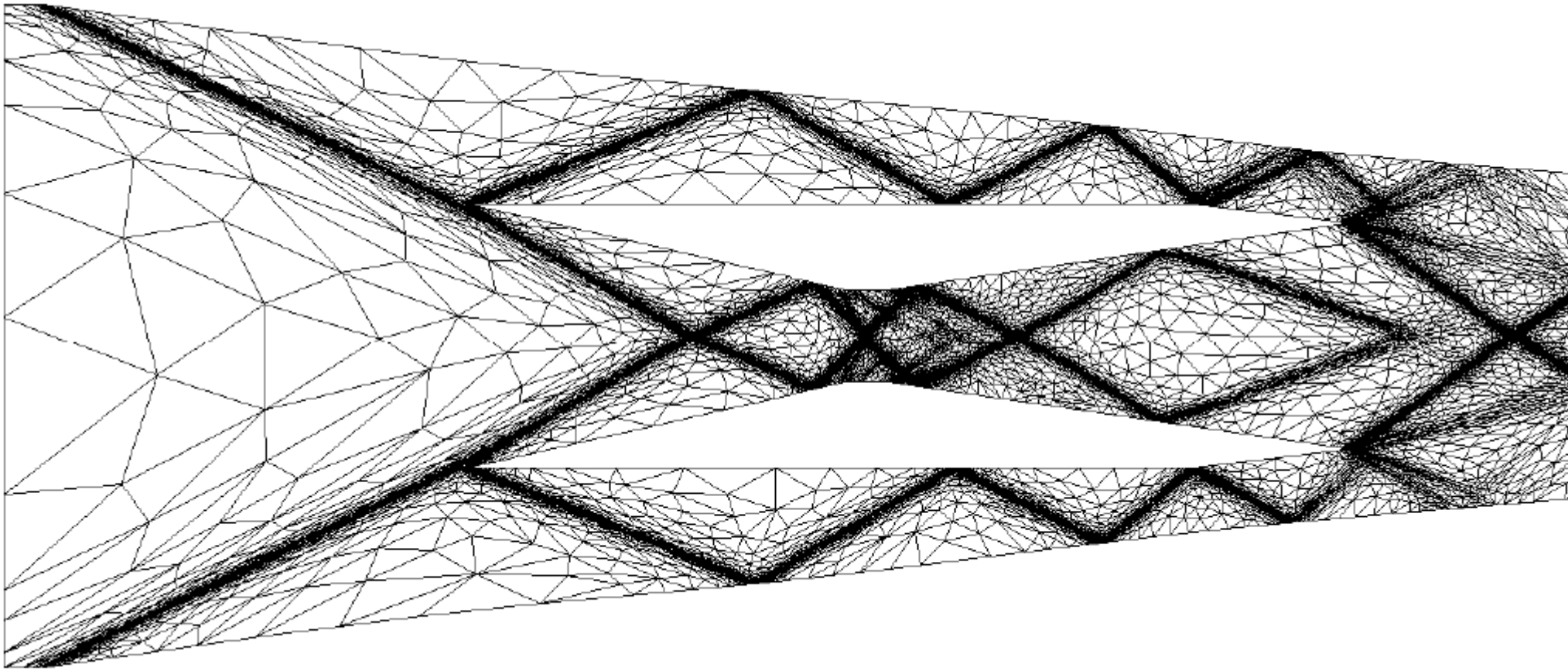


Detail of the previous

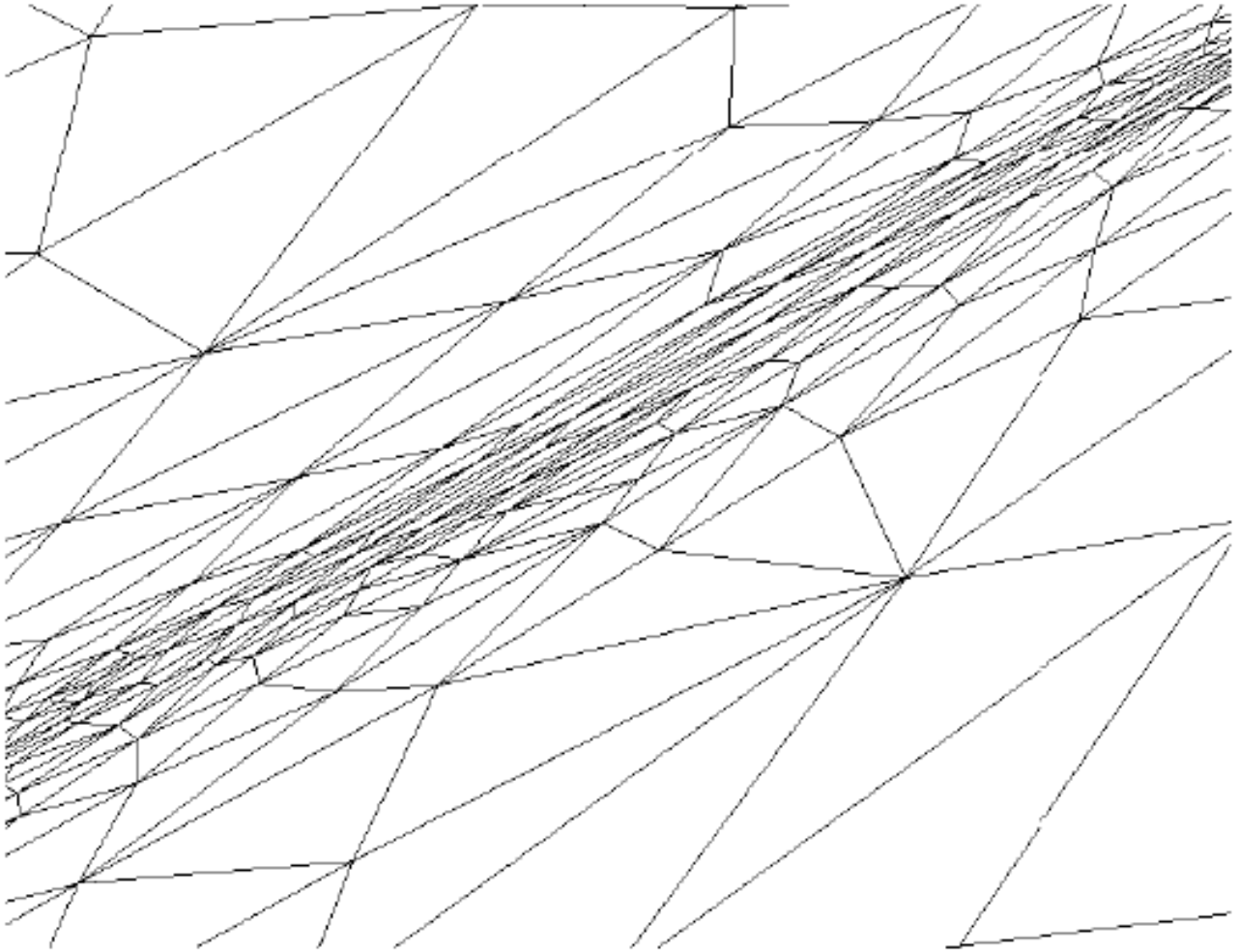



MESH

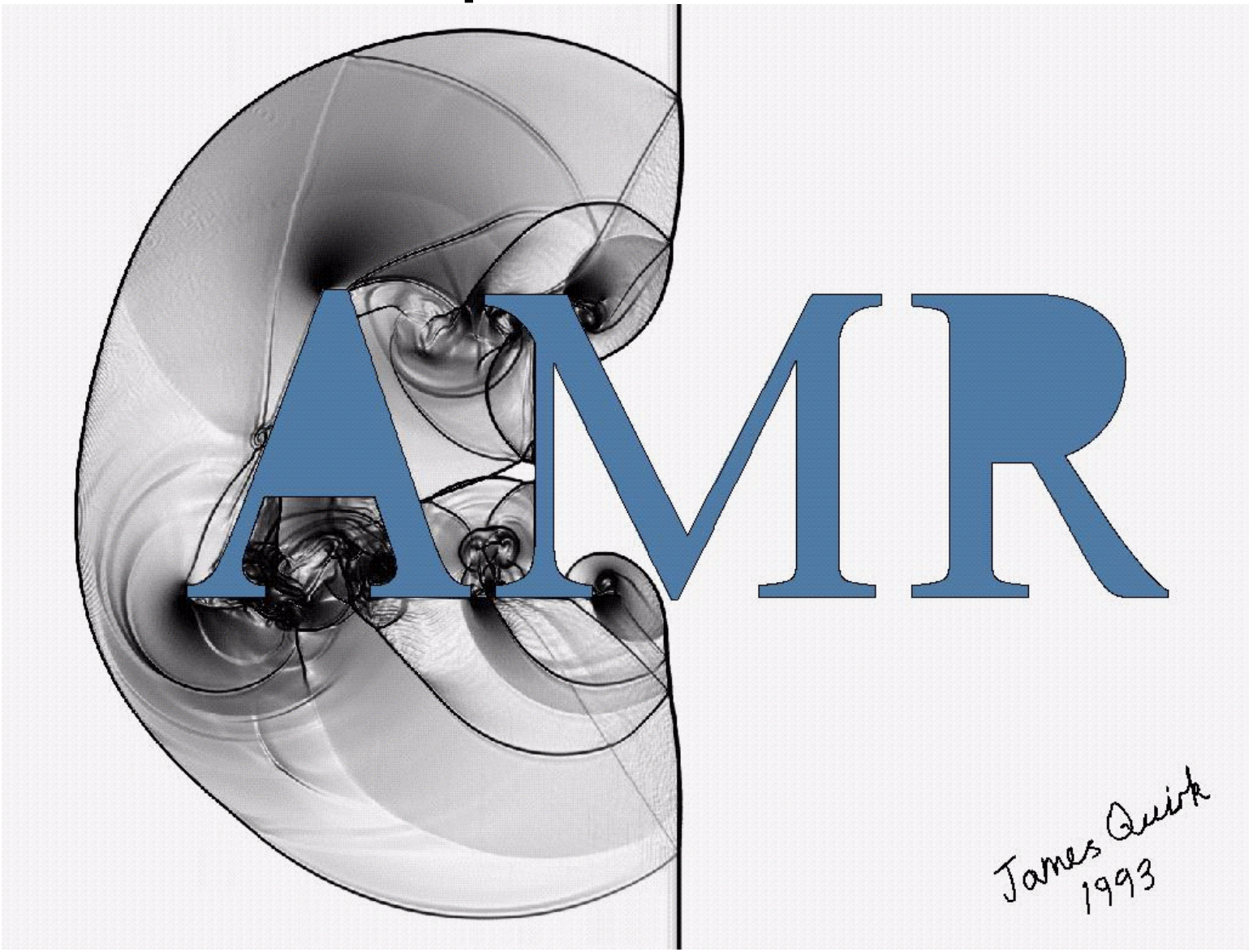
2-D supersonic flow channel



Local zoom of the previous



AMR = Adaptive Mesh Refinement



Comments on grid types

- Rectangle/cubic/hexahedral grids typically work better numerically than triangle/tetrad/simplex grids. The reasons are not too well known.
- Computer time and memory saving due to use of adaptive grid types can be huge, and increases with problem size.
- Having said that, it is also true that the approximation quality never improves by removing grid points.
- *In given CPU-time and memory*, however, use of adaptive grid is likely to be the *optimal solution* (*sometimes by quite a large margin*).

Time discretisation

- Time discretisation can be *explicit* or *implicit*.
- In *explicit* scheme, timestep at a cell must be shorter than fastest wave travel time across cell (the Courant condition, or CFL condition).
- The maximum usable timestep may vary widely across the grid, because both Alfven speed and cell size differ. To save computing time, one may use *temporal subcycling* to take short steps only where needed.
- *Implicit MHD* has been studied and to some extent used by the Michigan group.
  - Need to solve large nonlinear system by Newton iterations → convergence not foolproof

Parallelisation

- In large codes, parallelisation is usually needed nowadays.
- Most widely used parallelisation strategy is domain decomposition, i.e. each processor owns a specific domain and handles all computation in that domain. Domain boundaries may move or remain fixed, depending on the application.
- Domain-decomposed MHD on uniform or stretched grids has been in use for some years.
- How to combine domain decomposition with adaptive gridding has been an active research topic for the last 5-10 years

Elliptic equations

- Types of elliptic equations:
  - Poisson
  - Separable coefficients (Poisson and Helmholz in spherical coordinates, e.g.)
  - General elliptic equations (coefficients depend on all coordinates in non-factorable way)
- Where they arise from:
  - Electrostatic particle or Vlasov simulation (Poisson)
  - Implicit particle simulation (General elliptic)
  - Current continuity equation in ionosphere (General elliptic)
  - div(B) removal in MHD (Poisson)
  - Electromagnetic particle sim. (Poisson + Helmholz)

“Rapid” elliptic solvers

- Applicable to constant-coefficient equations or equations where coefficients factor like f(x) g(y)
- If parallelisation strategy allows it, FFT method is straightforward and nearly optimal speed (although not quite; for better, see Hockney and Eastwood book)
- One of the dimensions can also be done with tridiagonal solver, which can be parallelised better than FFT e.g. by pipelining
- Equally important than speed is the fact that the FFT/tridiag methods are usually quite stable and produce reliable answer

Iterative elliptic solvers

- If rapid solvers are not applicable, iterative solvers must be used
- There are many (Gauss, Gauss-Seidel, SOR, ADI, SIP, multigrid,...), but only one really works well: the Conjugate Gradient (or Bi-CG) algorithm
- Look up the pseudocode from Numerical Recipes and implement in your language
  - It's beautiful, relatively simple and very general!
- In many cases you don't even need a preconditioner
  - But restarting the algorithm every now and then is simple to program and may be beneficial also
- CG has applications also in data analysis (e.g. auroral tomography)

About roundoff error in general

- When grids become larger, roundoff error becomes more and more of a problem
- The problem is especially prominent with elliptic solvers
- For example, the SIPSOL solver works very well up to grid sizes about 50x50; for larger than that, it often does not converge at all
- It is easy to generate example problems (e.g., just by using random number coefficients) where any known elliptic solver fails to converge
- Robustness is much more severe problem than what you get by e.g. reading “Numerical Recipes” (which is otherwise very recommendable book!)

Future prospects of simulations

- Simulations will probably separate into two disciplines:
  - (1) Quick solution of problems using self-written simple programs on your PC or laptop
  - (2) Large, professional-quality parallelised software systems emerge for attacking challenging problems
- Since CPU speed increases:
  - Importance of ***numerical stability and robustness*** will increase
  - Importance of ***grid adaptivity*** will increase
  - Importance of ***Vlasov*** simulation will increase
  - Importance of ***multiphysical problems*** will increase

Some advice

- Computing speed grows exponentially with time. Thus every year, some problems turn from non-solvable to solvable. This has been the situation ever since the first computers were built.
- However, not all relevant problems are solved during the year they first become solvable!
- Thus, there are physically relevant problems that are easy to simulate nowadays, which would have been at “grand challenge” class 10-20 years ago, but which nobody yet simulated.
- Thus, while hard problems are challenging, do not ignore the easy ones!

Prospects in the application domain

- In plasma physics, simulation models have not yet reached the maturity of other “normal” physics:
  - In my opinion this is mainly due to the fact the plasma physics should really be done in 6-D phase space.
  - For an arbitrary planet, shape and size of magnetosphere is simulated correctly, but not such processes as rate of atmospheric erosion or coherent radio output power (for the latter two, predicting even the correct order of magnitude is hard)
  - Compare this to climate simulation (neutral fluid dynamics with radiation), where e.g. global temperature comes out correctly to ~1 % or better
  - Plasma ***is*** inherently difficult, both in theory, and to simulate.

Quantifying plasma difficulties

- In neutral fluid, we have the *sound wave* and the *entropy wave* (*contact discontinuity*). Sound is typically approximated away in geophysical flows.
- In MHD, the sound wave splits into *slow*, *Alfven* and *fast magnetosonic* wave. Propagation properties depend on **B**-field direction.
- In real (i.e. kinetic-based) plasma physics, we have almost too many wave modes to list and remember:
  - whistler waves (modified fast magnetosonic)
  - ion acoustic waves
  - ion Bernstein waves
  - lower and upper hybrid waves
  - etc., etc., (e.g., all electron modes missing from this list...)

Importance of plasma in space

- Gravity governs the universe, but plasma physics has its role to play everywhere. Many of the nontrivial questions in space are plasma-physical:
  - ***reconnection*** and its resulting particle acceleration
  - ***shock acceleration***
  - ***dynamo action*** (magnetic fields of stars, planets, magnetars..)
  - details of gamma ray bursts, gamma flares, supernovas
  - details of solar system formation
  - details of stellar evolution (loss of angular momentum?)
  - primordial magnetic fields (are there any?)
- Many of these have *astrobiological dimension,* e.g:
  - stability of planetary atmospheres on low-mass planets&stars
  - details of life-threatening supernovas and gamma ray bursts
  - strength of flares on lighter-than-Sun stars

Importance of plasma in space 2

- To attack some of these questions, we need *multiphysical advanced simulations*, *together with* good physical understanding of their results.
- For example, for stars: "*relativistic* or *non-relativistic MHD with gravity, radiation and particle physics & good equations of state, parallelised on automatically adapted grid*" would be high on the wish list
- Or, for planets: May I have one of your "*parallelised Vlasov solvers with anisotropically adaptive grid, with neutral collisions and charge-exchange processes and modelled small-scale wave-electron interactions*" (with home delivery, please)