Zooming in on 3D magnetized plasmas with grid-adaptive simulations

R. Keppens[a]*, M. Nool[b] and J.P. Goedbloed[a]

[a]FOM-Institute for Plasma Physics 'Rijnhuizen',
P.O. Box 1207, 3430 BE Nieuwegein, The Netherlands
keppens@rijnh.nl – goedbloed@rijnh.nl

[b]CWI,
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
Margreet.Nool@cwi.nl

We present multidimensional hydro- and magnetohydrodynamic simulations, where the fine-scale dynamics is accurately and efficiently captured through a solution-adaptive meshing procedure. The Adaptive Mesh Refinement (AMR) strategy is implemented for any dimensionality and can be used generally for sets of conservation laws, combined with a suitable conservative high resolution discretization. Specifically, the 2D and 3D MHD scenarios discussed here make use of a level dependent spatial discretization: a fully upwind scheme on the finest grid level is combined with a Total Variation Diminishing Lax-Friedrich method on lower levels. The AMR process ensures that this combination acts as a low-cost hybrid scheme, accurately capturing all sharp flow features. Examples are given of planar evolutions and of 3D Rayleigh-Taylor unstable, magnetized plasma dynamics. Auto-parallelization on multi-processor SGI Origin architectures shows linear to superlinear speedup for non-adaptive 3D MHD calculations, but is as yet unsuccesful in parallelizing multi-level AMR calculations.

1. ALGORITHM AND IMPLEMENTATION ASPECTS

As a promising extension to the Versatile Advection Code initiated by Tóth [9] (VAC, see http://www.phys.uu.nl/~toth/), a fully automated Adaptive Mesh Refinement (AMR) [1] scheme is now incorporated and used for multidimensional hydro- and magnetohydrodynamic (MHD) studies. In essence, AMR generates or destructs – both controlled by the ensueing dynamics – hierarchically nested grid levels with subsequently finer mesh spacings. This AMR algorithm and, particularly, all complications associated with ensuring the global conservation property needed for shock-capturing calculations, have been addressed in the work by Berger and Colella [2]. In that pioneering paper, the authors stress that *"... it seems possible to implement a general code where the number of*

*Work done within the association agreement of Euratom and the 'Stichting voor Fundamenteel Onderzoek der Materie' (FOM) with financial support from the 'Nederlandse Organisatie voor Wetenschappelijk Onderzoek' (NWO) and Euratom. It is part of a project on 'Parallel Computational Magneto-Fluid Dynamics', an NWO Priority Program on Massively Parallel Computing. Use of computing facilities is funded by 'Nationale Computer Faciliteiten' (NCF).

dimensions is input". We make use of such a general code – AMRVAC – in simulations presented here, where both the dimensionality and the system of conservation laws is selected in a pre-processing stage.

Noteworthy is the fact that we allow for a level-dependent choice of the discretization scheme employed. We advocate the use of an approximate Riemann solver based method at the highest allowed level(s), in combination with the robust but more diffusive Total Variation Diminishing Lax-Friedrich (TVDLF, see e.g. [10]) method on the coarser levels. This is computationally efficient, and acts as a hybrid scheme since all sharp features are fully captured with the more accurate scheme.

For the MHD studies, we use the eight-wave formulation from Powell, see e.g. [7], to maintain the $\nabla \cdot \mathbf{B} = 0$ constraint to truncation error. The approach modifies the approximate Riemann solver to 'propagate' $\nabla \cdot \mathbf{B}$ errors with the plasma velocity and adds non-conservative source terms to the MHD equations proportional to $\nabla \cdot \mathbf{B}$. In the AMR simulations, adding these corrective source terms in a split fashion inactivates them at regridding operations.

The dimension-independence of the different numerical schemes, equation modules, and of the entire AMR algorithm is realized by the use of the Loop Annotation SYntax or LASY [8]. The code is configured to dimensionality using a single `Perl`-script, prior to compilation. We preferentially run AMRVAC on workstations or shared memory platforms, since the employed data structure still uses statically allocated linear arrays. Improvements to port the code to parallel architectures, as was done succesfully for VAC [6,5] earlier, are under consideration. First attempts to rely on auto-parallelization options, as available on multi-processor SGI Origin architectures, are reported for time-dependent 3D MHD simulations, for both non-adaptive VAC and grid-adaptive AMRVAC simulations.

2. AUTO-PARALLELIZATION RESULTS

The performance and parallel scaling achieved with the Versatile Advection Code [6] on a fully 3D MHD calculation of a Kelvin-Helmholtz unstable magnetized jet, with resolution $50 \times 100 \times 100$ is briefly summarized as follows

- using Fortran 90 on a single processor of a Cray C90, this calculation reached a sustained performance of 388.4 Mflops (40 % peak), executing one time step in 8.4 seconds. With autotasking, a speedup of 3.7 was reached on 4 processors.
- on distributed memory platforms (Cray T3E, IBM SP), the use of High Performance Fortran yielded nearly linear speedup.

In the course of 2001, an SGI Origin 3800 with 1024 processors, **TERAS**, is being installed at the Dutch supercomputing centre SARA in Amsterdam. Ultimately, 512 processors will be accessible in a virtually shared manner using OpenMP. The RISC processors are 500 Mhz, IP35 MIPS R14000, with a peak performance of 1 Gflops. At a minimal effort, automatic generation of parallel code is possible using the compiler option `f90 -apo`, which is subsequently executed on N_p processors by setting the environment variable `OMP_NUM_THREADS`.

Our first attempt to use this auto-parallelization option on the same 3D MHD calculation of a magnetized jet evolution is shown in Fig. 1: the speedup is superlinear up to

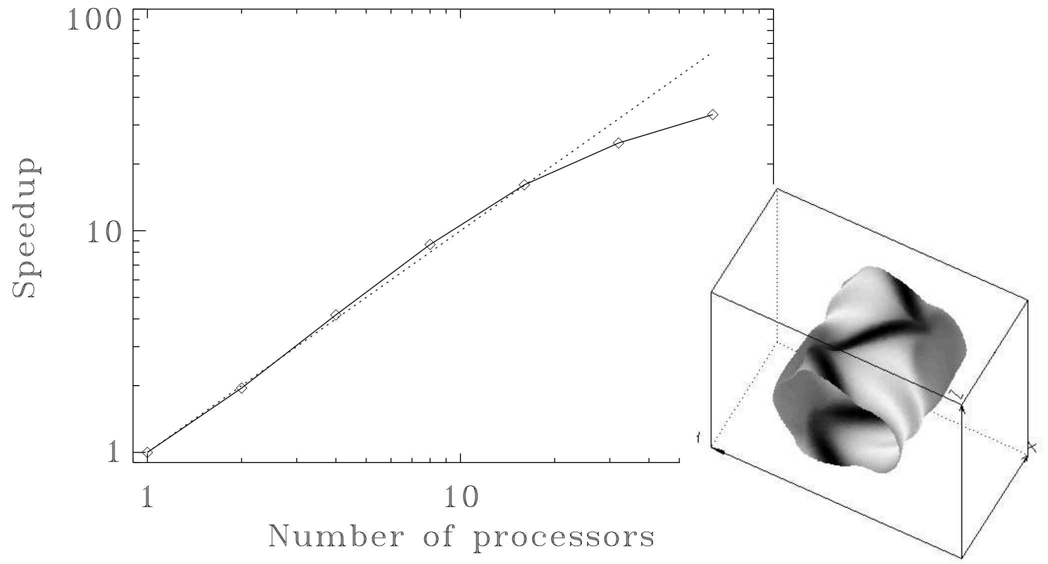

Figure 1. Scaling of VAC up to 64 processors on a SGI Origin 3800 using auto-parallelization (`OpenMP`). A snapshot of the 3D magnetized jet simulation used in the timing experiments is shown as an inset: the jet surface colored by thermal pressure is deformed due to the Kelvin-Helmholtz instability.

16 processors, and then levels off to a value of 33.4 for 64 processors. We note that the memory requirements for this testcase are only 327 Mb. While the speedup results are extremely encouraging, the single processor execution time per time step is 39.8 seconds, a factor of 4.7 longer than on the vector Cray C90 machine (which had the same peak performance per processor). The performance analysis tool `perfex` reports a rather low 53 Mflops on a single CPU of the SGI Origin 3800, which presumably underestimates the actual number of floating-point operations per second[2]. However, judging from the observed differences (Cray versus SGI) in execution time per time step, this particular simulation is currently only achieving up to 10 % of the peak performance on TERAS.

We subsequently used AMRVAC as a Domain Decompositioner (DD) to analyse the obtainable speedup through auto-parallelization on the SGI Origin 3800. To use AMRVAC as a DD, we simply set the maximum number of levels to one, and specify the largest block size at (pre)compile time. The AMR implementation is then only used to decompose the full computational domain in blocks at time $t = 0$, and to ensure the correct filling of internal ghost cell boundaries created by the DD. We then run the same 3D MHD simulation as used for VAC, except that the total resolution is set to $52 \times 104 \times 104$. Running the N-block simulation on N processors yields the speedup curve shown in Fig. 2. Again a superlinear behavior is found up to 8 processors. The speedup reaches 13.8 on 16 processors, but then degrades when using 32 processors. Again the obtained single processor performance is estimated to be at most 10 % peak. While this can certainly be improved, the automated parallelization results are very promising, at least up to several tens of processors.

[2] A two floating-point multiply-add operation counts as one instruction for `perfex`.

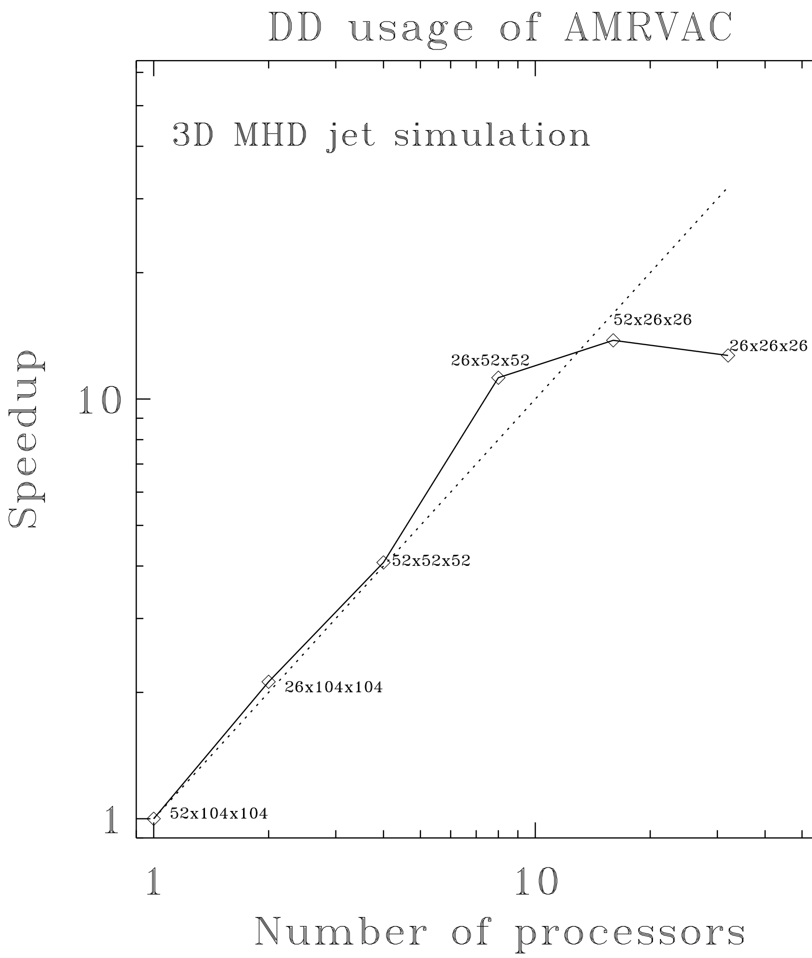

Figure 2. Scaling of AMRVAC up to 32 processors on the SGI Origin 3800 using auto-parallelization. The same 3D MHD jet simulation as for Fig. 1 is domain decomposed with AMRVAC. The block size is indicated: the N block case is run on N processors.

3. PLANAR EVOLUTIONS

To demonstrate the AMRVAC code potentials for grid-adaptive calculations, we start with two examples of planar (2D) evolutions. By selecting the Euler system, we can simulate the interaction of a Mach 2.5 shock with a low density 'bubble'. The pre-shock region has a unit sound speed c_s, except in a circular low density region with $c_s = 5$. This problem is inspired by a similar shock-bubble interaction as found on the website for the AMRCLAW software package at `http://www.amath.washington.edu/~claw/` by LeVeque and coworkers. In the example calculation on that website, the up-down symmetry with respect to the normal from the bubble centre to the shock front was exploited. Here, we let the shock front make an angle of 60° with the x-axis, leaving the symmetry as a check on the calculation. On a 80×40 base grid on the domain $[0,2] \times [0,1]$, we allow for 3 levels. Subsequent refinement ratios are set to 2 and 4, so that the AMR simulation would yield an effective resolution of 640×320. Snapshots of the density (schlieren-plot) and grid structure at time $t = 0$ and at time $t = 0.3$ are shown in

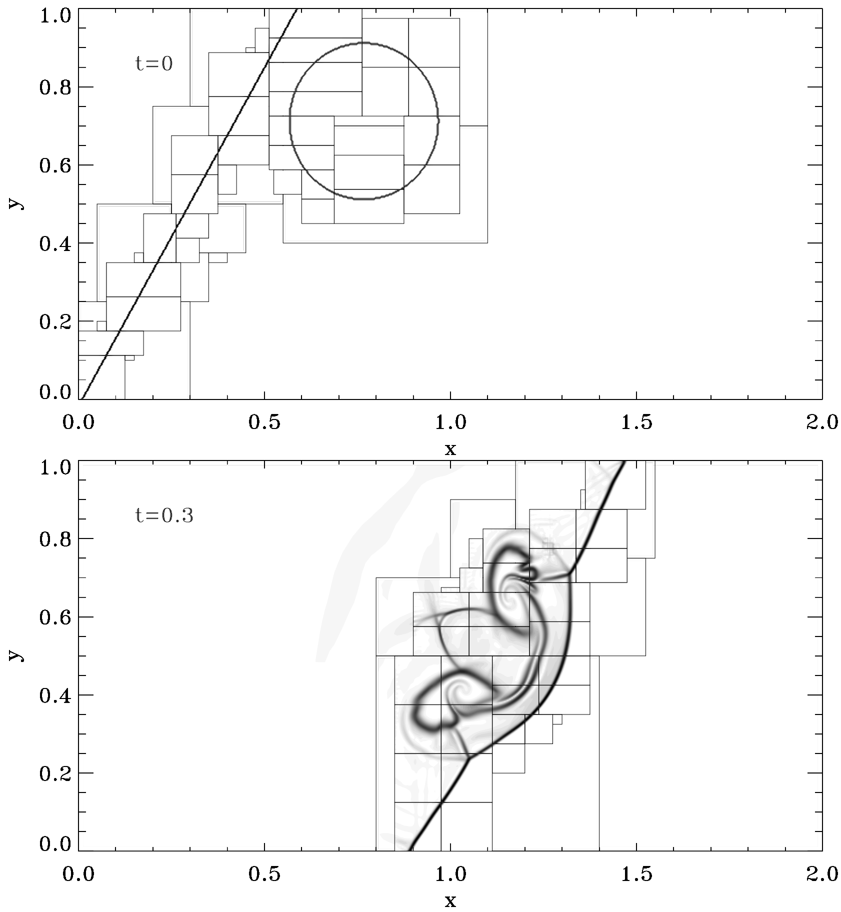

Figure 3. Two-dimensional shock-bubble interaction, allowing for three refinement levels in AMRVAC. The AMR process generates the grids at $t = 0$ at the discontinuities (the shock and the bubble boundary). The density is shown as a schlieren plot.

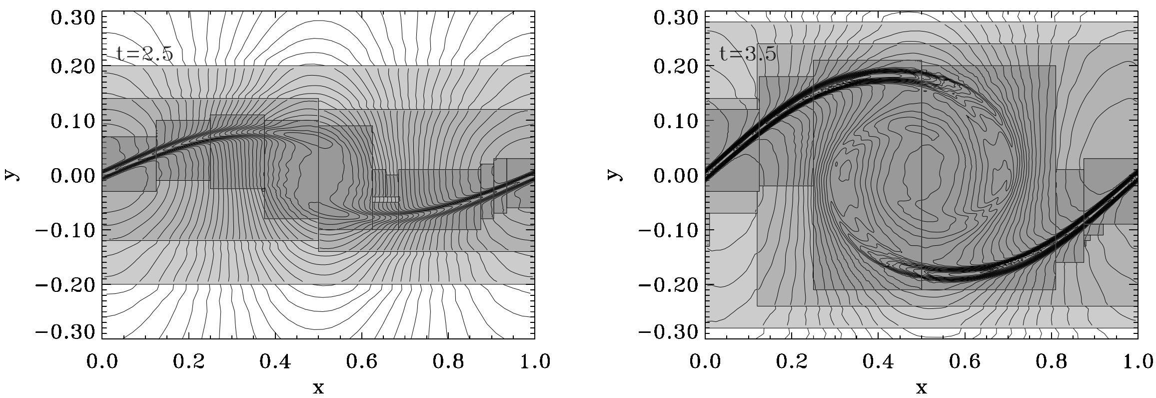

Figure 4. Density evolution in an unstable magnetized shear flow. We show the central region of the computational domain $y \in [-1, 1]$ only. The 4 grid levels are indicated.

Figure 3. We used the TVDLF scheme on levels 1 and 2 and an upwind scheme on the finest level.

A demonstrative MHD problem is a 2D simulation of a magnetized shear flow, in a Kelvin-Helmholtz unstable, subsonic (Mach 0.5) parameter regime. The detailed problem description is borrowed from Keppens *et al.* [4], where high resolution simulations on static grids with VAC are reported. A small perturbation of the system triggers a vortical flow. The magnetic field becomes amplified in a spiral pattern, in turn leading to narrow lanes of low density. Using 4 grid levels on a $[0,1] \times [-1,1]$ domain and a base resolution of 50×100, we recover the high resolution results. Figure 4 shows the density and grid structure at 2.5 and 3.5 sound crossing times. The dynamic regridding process nicely traces the fine-scale density structures. Again, the approximate Riemann-solver is only operative at the highest level.

In summary, the grid-adaptive strategy now allows to efficiently simulate multidimensional HD and MHD scenarios without sacrificing in accuracy. In a recent paper [3], we quantify the obtainable efficiency in computing time for a variety of 1D, 2D, and 3D HD and MHD problems. Indeed, for 2D HD and MHD problems like those given in this section, AMR execution times can be a factor of 10 to 20 times shorter than the corresponding high resolution static grid calculation.

Auto-parallelization experiments on SGI platforms with multi-level grid-adaptive 2D hydrodynamic shock problems were not succesful (no speedup). This is influenced by the involved spatio-temporal intra-level interpolations needed to fill ghost cells for individual grid patches, but is primarily due to the fact that our AMR implementation creates optimally fitted, but very different sized higher level grids. This would suggest algorithmic changes to (1) enforce the same time step on all levels; and (2) change the AMR algorithm to create fixed grid block sizes.

4. 3D SIMULATIONS

We simulated a Rayleigh-Taylor unstable 3D configuration where a heavy plasma rests on top of a lighter plasma in an external gravitational field. With gravity in the $-y$-direction of the unit cube, we run two cases: (1) a pure hydrodynamic scenario where the interface separating the two uniform density regions at time $t = 0$ is given by $y_{\rm int} = 0.8 + 0.05 \sin 6\pi x \sin 4\pi z$; and (2) a full MHD problem with a uniform horizontal magnetic field $\mathbf{B} = 0.1\hat{e}_x$ and $y_{\rm int} = 0.8 + 0.05 \sin 2\pi x \sin 2\pi z$. The density contrast is 10. We start with a very coarse base grid of size $20 \times 40 \times 20$, and allow for three refinement levels, achieving a $80 \times 160 \times 80$ resolution locally.

Snapshots of the density structure for the hydro simulation at time $t = 1.2$ in Fig. 5 show that the initial perturbation leads to the formation of 12 ‘fingers’ (spikes) of falling high density plasma. The resulting flow field around the pillars induces roll-up and a clear pairwise interaction along the x-direction causes a distinct asymmetry in the mixing process. Fig. 6 depicts the density structure at $t = 1$ for the MHD case in two vertical cutting planes. In the plane parallel to the initial uniform magnetic field, the formation of fine scale structure is effectively suppressed by the stabilizing magnetic tension. In the plane orthogonal to the field, shorter wavelength features are clearly visible.

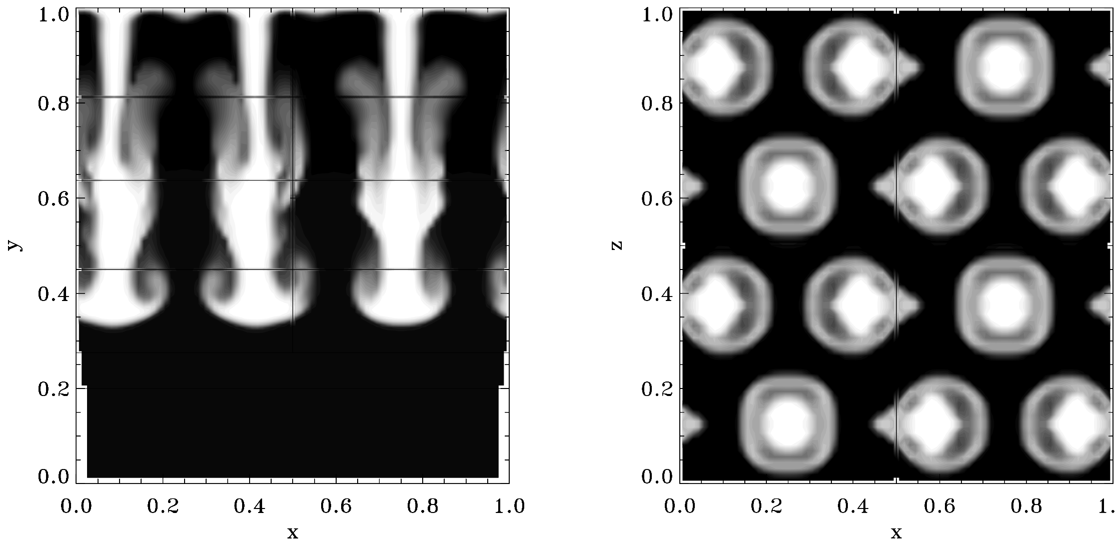

Figure 5. Density structure at $t = 1.2$ in the planes $z = 0.4$ and $y = 0.4$ for a 3D HD simulation of a Rayleigh-Taylor instability. Under the influence of gravity, the heavy plasma mixes into the lighter one underneath. We use AMR with 3 grid levels.

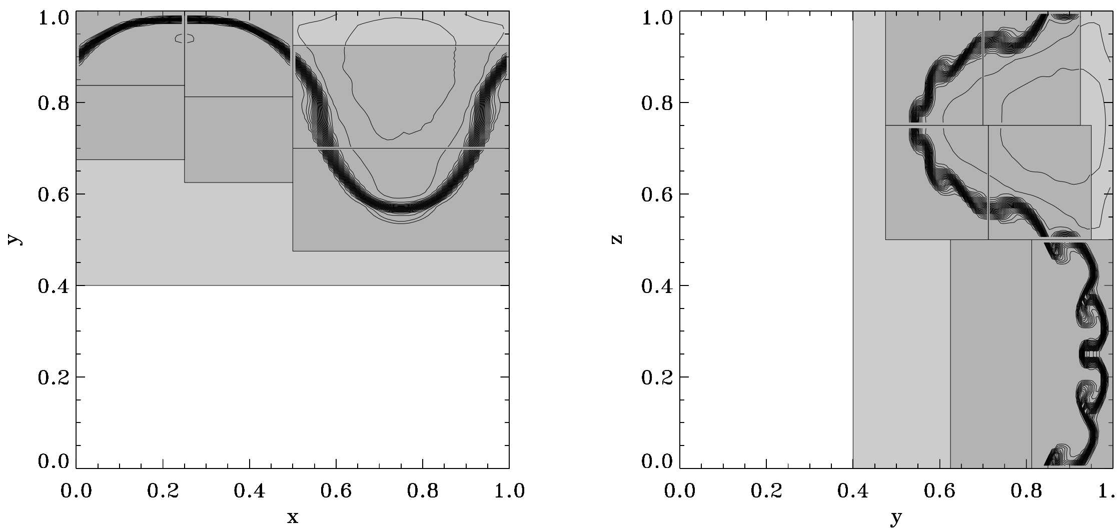

Figure 6. Density structure at $t = 1$ in the planes $z = 0.2$ and $x = 0.3$ for a 3D MHD simulation of a Rayleigh-Taylor instability (gravity along $-y$).

For these 3D Rayleigh-Taylor simulations, the AMR efficiency – defined as the ratio of the time needed to run the corresponding high resolution static grid case to the execution time for the AMR case – remains limited: we only gain a factor of 2.8 for the MHD case. This is because the level one grid resolution is so coarse that level two grids are covering 30 % of the entire computational domain at $t = 0$ (and 16.25 % for level $l = 3$ grids) and this increases as time progresses to reach 60 % (43 % for $l = 3$) at time $t = 1$. Taking that into account, the obtained speedup is indeed optimal, which is also confirmed by the observation that only 1.35 % of the entire CPU time is devoted to AMR-specific calculations (regridding, updating including flux fixings). Much better efficiencies are reachable as soon as the base level grid is not underresolving the physical process.

This latter statement is demonstrated clearly for a 3D advection problem in the table below. We simulated the advection of a sphere of radius 0.2 across the diagonal of a unit cube. Within the sphere, the density was set to 2, while it is 0.5 exterior to the sphere. The advection velocity $\mathbf{v} = (1, 1, 1)$ brings the sphere at $t = 1$ to its original centered position using triple periodic boundary conditions. We compare execution time and memory requirements for a 320^3 static grid simulation on the Origin 3800, with an AMR run exploiting 3 levels, with a 40^3 base grid, and consecutive refinement ratios 2 and 4. Since the coverage of the highest grid level now remains roughly at 10 % throughout the simulation, a significant reduction by a factor of 19.9 in computing time is realized. Note also that the time spent on the regridding process is fully negligible. As this tenfold efficiency is already reached for a 3 level calculation in a pure advection problem, much higher efficiencies will hold in practical 3D MHD simulations. The table also indicates the much reduced memory requirements for the AMR simulations. Our current implementation uses up to 3 solution vectors per level and an order of magnitude in storage is gained. Indeed, the high resolution case could only be run exploiting 4 TERAS processors, while the AMR run fitted in the 1Gb available on a single CPU.

resolution	levels	timing (sec)	eff.	AMR%	solution storage
$320^3 \times [0,1]$	1	172197	-	-	$\mathcal{O}(10^8)$ DP numbers
	3	8653	19.9	4.14	1.68×10^7 DP numbers

At the time of this writing, more efforts are needed on (i) visualizing 3D grid-adaptive magnetized plasma dynamics; and (ii) tuning the software to parallel platforms.

REFERENCES

1. M.J. Berger, *SIAM J. Sci. Stat. Comput.* **7**, 904 (1986).
2. M.J. Berger and P. Colella, *J. Comput. Phys.* **82**, 64 (1989).
3. R. Keppens *et al.*, *J. Comput. Phys.*, submitted (2001).
4. R. Keppens *et al.*, *J. Plasma Phys.* **61**, 1 (1999).
5. R. Keppens, in *Proc. of ParCFD 2000* (Trondheim, Norway), Eds. Ecer *et al.* (2001).
6. R. Keppens and G. Tóth, *Parallel Computing* **26**, 705 (2000).
7. K.G. Powell *et al.*, *J. Comput. Phys.* **154**, 284 (1999).
8. G. Tóth, *J. Comput. Phys.* **138**, 981 (1997).
9. G. Tóth, *Astrophys. Lett. & Comm.* **34**, 245 (1996).
10. G. Tóth and D. Odstrčil, *J. Comput. Phys.* **128**, 82 (1996).

Zooming in on 3D magnetized plasmas with grid-adaptive simulations

R. Keppens[1], M. Nool, & J.P. Goedbloed

Oral contribution to
Parallel CFD 2001, **Egmond aan Zee, The Netherlands. May 21-23, 2001.**

Appeared in:
Parallel Computational Fluid Dynamics – Practice and Theory,
P. Wilders et al. (eds.), 2002, Elsevier Science B.V., pp. 215-222

[1]keppens@rijnh.nl, FOM-Institute for Plasma Physics 'Rijnhuizen',
P.O. Box 1207, 3430 BE Nieuwegein, The Netherlands