An Ideal Compressible Magnetohydrodynamic Solver with Parallel Block-structured Adaptive Mesh Refinement

Muller Moreira Lopes[a], Ralf Deiterding[b,*], Anna Karina Fontes Gomes[a], Odim Mendes[a], Margarete O. Domingues[a]

[a]*National Institute of Space Sciences (INPE), São José dos Campos, S. Paulo, 12.227-010, Brazil*
[b]*Aerodynamics and Flight Mechanics Research Group, University of Southampton, SO17 1BJ, United Kingdom*

Abstract

We present an adaptive parallel solver for the numerical simulation of ideal magnetohydrodynamics in two and three space dimensions. The discretisation uses a finite volume scheme based on a Cartesian mesh and an explicit compact Runge–Kutta scheme for time integration. Numerically, a generalized Lagrangian multiplier approach with a mixed hyperbolic-parabolic correction is used to guarantee a control on the incompressibility of the magnetic field. We implement the solver in the AMROC (Adaptive Mesh Refinement in Object-oriented C++) framework that uses a structured adaptive mesh refinement (SAMR) method discretisation-independent and is fully parallelised for distributed memory systems. Moreover, AMROC is a modular framework providing manageability, extensibility and efficiency. In this paper, we give an overview of the ideal magnetohydrodynamics solver developed in this framework and its capabilities. We also include an example of this solver's verification with other codes and its numerical and computational performance.

Keywords: AMROC, magnetohydrodynamics, finite-volume, mesh refinement

1. Introduction

The numerical simulation of plasma plays an important role in astrophysics and space physics due to its scale variabilities [1]. For studying macroscopic space plasma phenomena, the ideal magnetohydrodynamics (MHD) theory is a handy tool that treats the plasma as a perfect electrically conducting fluid under the influence of a magnetic field [2]. Since the last decades, a variety of numerical algorithms for multidimensional MHD based on finite volume method have been developed, for instances the ones discussed in [3].

*Corresponding Author

Email addresses: muller.lopes@inpe.br (Muller Moreira Lopes), r.deiterding@soton.ac.uk (Ralf Deiterding), anna.gomes@inpe.br (Anna Karina Fontes Gomes), odim.mendes@inpe.br (Odim Mendes), margarete.domingues@inpe.br (Margarete O. Domingues)

These algorithm have two significant extensions compared to the primary hydrodynamical case: The first is an extension of the Riemann solver used to compute the fluxes of each conserved quantity, and the second is the method that assures a control of the divergence-free constraint, *i.e.*, $\nabla \cdot \mathbf{B} = 0$. In general, realistic MHD simulations in the context of space weather forecast are performed under prohibitive computational costs, which remains a core challenge to be overcome. Therefore, in practice for multiscale space MHD applications very efficient frameworks are essential. As proposed here, a new MHD solver has been developed upon our serial Carmen–MHD code, cf. [4, 5] and references therein. The current solver is an ideal compressible MHD model with finite volume adaptive mesh in two and three space dimensions developed in the parallel AMROC framework, incorporating recent advances in the area of SAMR.

We organise the content as follows: In Section 2, we describe the governing equations, the finite volume method, and briefly the main ideas of the block-based adaptive mesh refinement (AMR) approach as implemented in AMROC. In Section 3, we develop the numerical experiments. Then, we draw the conclusions in Section 4.

2. Numerical methods

2.1. Governing equations

In the solution of hyperbolic conservation laws given by partial differential equations such as $\partial_t \mathbf{q} + \nabla \cdot \mathbf{f}(\mathbf{q}) = 0$, different length scales are ubiquitous. It is well established that for nonlinear flux functions $\mathbf{f}(\mathbf{q})$ even continuous initial data can develop into discontinuities over time [6]. In particular, we consider the compressible inviscid ideal magnetohydrodynamic equations, in Cartesian coordinates [2], that describe the physics of an ideal conducting fluid under the influence of a magnetic field. Basically this system is composed of the continuity, energy, momentum equations, and induction equations as

$$\mathbf{q} = \begin{pmatrix} \rho \\ E \\ \rho\mathbf{u} \\ \mathbf{B} \end{pmatrix}, \quad \mathbf{f}(\mathbf{q}) = \begin{pmatrix} \rho\mathbf{u} \\ \left(E + p + \dfrac{\mathbf{B}\cdot\mathbf{B}}{2}\right)\mathbf{u} - \mathbf{B}(\mathbf{u}\cdot\mathbf{B}) \\ \rho\mathbf{u}\mathbf{u} + \left(p + \dfrac{\mathbf{B}\cdot\mathbf{B}}{2}\right)\mathbf{I} - \mathbf{B}\mathbf{B} \\ \mathbf{u}\mathbf{B} - \mathbf{B}\mathbf{u} \end{pmatrix}. \tag{1}$$

In the latter, ρ is the density, $\mathbf{u}$ the fluid velocity vector, $\mathbf{B}$ the magnetic field vector, and E the total energy density defined as $E = \dfrac{p}{\gamma - 1} + \dfrac{\rho u^2}{2} + \dfrac{B^2}{2}$, where p is the gas pressure considering the state equation assuming an ideal gas plasma. All variables depend on the spatial location $\mathbf{x}$ and time t and these equations are represented in non-dimensional form such that the magnetic permeability μ is normalised to one.

Due to Gauss' law of magnetism, this system has also a physical constraint in the magnetic field, given by $\nabla \cdot \mathbf{B} = 0$. In numerical simulations the divergence constraint is not always satisfied as first realised by Brackbill and Barnes [7]. In order to minimize this effect, several types of corrections have been used, as described in [3] and

references therein. Here, we have chosen a divergence transport approach known as parabolic-hyperbolic divergence cleaning proposed by Dedner *et al.* in [8] in combination with the correction proposed by Mignone and Tzeferacos in [9]. This correction consists in adding a differential operator $D = \frac{1}{c_h^2}\frac{\partial}{\partial t} + \frac{1}{c_p^2}$ to the divergence constraint of the **B** field, resulting in a new system composed basically of the continuity, energy, momentum equations, and the additional equations

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \cdot (\mathbf{uB} - \mathbf{Bu} + \psi \mathbf{I}) = 0, \quad \frac{\partial \psi}{\partial t} + c_h^2 \nabla \cdot \mathbf{B} = -\frac{c_h^2}{c_p^2}\psi \tag{2}$$

where ψ is a scalar-valued function, $\mathbf{I}$ the identity tensor, and c_p and c_h are the parabolic and hyperbolic constants, respectively. The constant c_h is defined as

$$c_h = \max\left[\nu\frac{\Delta h}{\Delta t}, \max\left(|u_i| \pm c_f\right)\right], \tag{3}$$

where $\Delta h = \min(\Delta x, \Delta y, \Delta z)$, $\Delta x, \Delta y, \Delta z$ are the mesh sizes in each direction, ν the Courant number, u_i is the velocity of the i-th component, and c_f is the fast magnetoacoustic wave of the MHD model. The c_p value is defined in terms of the parameter $\alpha_p = \Delta h \frac{c_h}{c_p^2}$, where $\alpha_p \in [0, 1]$, as described in [9]. We also have γ denoting the specific heat ratio. This system is well known as the Generalized Lagrange Multiplier (GLM) approach, and it reduces to the usual MHD system when $\psi = 0$.

In practical MHD cases, discontinuous shock and contact waves can develop, which requires the use of shock-capturing methods. In particular, the finite volumes method have been constructed to handle particularly this behaviour in a robust and oscillation free way [10]. Since in inviscid problems discontinuities and contact waves are usually very localised, a local increase of mesh resolution is beneficial to represent these jumps as accurately as possible.

In the computations presented here, we use a finite volume scheme based on a Cartesian mesh with HLLD numerical flux introduced by [11] with monotonised central (MC) symmetric limiter discussed in [12] and an explicit second order Runge–Kutta scheme for time integration. We apply the divergence control parameter $\alpha_p = 0.4$ and $\psi \equiv 0$ in the initial conditions.

2.2. Block-structured AMR and AMROC framework

The structured adaptive mesh refinement method for finite volume methods introduced by Berger and Collela [13] follows a patch-oriented refinement approach, where non-overlapping rectangular sub-meshes $G_{\ell,m}$ define by $G_\ell := \bigcup_{m=1}^{M_\ell} G_{\ell,m}$ the domain of an entire level with index $\ell = 0, \cdots, L$.

As the construction of refinement proceeds recursively, a hierarchy of sub-meshes successively contained within the next coarser level domain is created. This method has become an important mesh adaptation approach for hyperbolic conservation laws.

The characteristic of the SAMR algorithm is that refinement patches overlay coarser mesh data structures, instead of being embedded, thereby avoiding data fragmentation. Values of cells covered by finer sub-meshes are subsequently overwritten by averaged

fine mesh values, which, in general, would lead to a loss of conservation on the coarser mesh. A remedy to this problem is to replace the coarse mesh numerical fluxes at refinement boundaries with the sum of fine mesh fluxes along the corresponding coarse cell boundary, cf. [13, 14].

The recursive nature of the algorithm allows only the addition of one new level in each refinement operation. The patch-based approach does not require special coarsening operations; sub-meshes are simply removed from the hierarchy. The coarsest possible resolution is thereby restricted to the level zero mesh. It is assumed that all mesh widths on level ℓ are r_ℓ-times finer than on the level $\ell - 1$, which ensures that a time-explicit finite volume scheme remains stable under a CFL-type condition on all levels of the hierarchy. The numerical update is applied on the level ℓ by calling a single-mesh routine implementing the uniform scheme in a loop over all the sub-meshes. The regularity of the input data allows a straightforward implementation of the scheme and further permits optimisation to take advantage of high-level caches, and pipelining, for instance. New refinement meshes are initialised by interpolating the vector of conservative quantities $\mathbf{Q}$ from the next coarser level. However, data in cells already refined is copied directly from the previous refinement patches. *Ghost cells* around each patch are used to decouple the sub-meshes computationally. Ghost cells outside of the root domain G_0 are used to implement physical boundary conditions. Ghost cells in G_ℓ have a unique interior cell analogue and are set by copying the data value from the patch where the interior cell is contained (synchronisation). For $\ell > 0$, internal boundaries can also be used. If recursive time step refinement is employed, ghost cells at the internal refinement boundaries on the level ℓ are set by time-space interpolation from the two previously calculated time steps of level $\ell - 1$. Otherwise, spatial interpolation from the level $\ell - 1$ is sufficient.

AMROC implements the SAMR method discretisation-independent in one to three space dimensions and is fully parallelised for distributed memory systems [15, 14]. With its parallel distribution strategy the overlapping ghost cell regions of neighbouring patch blocks are synchronised over processor borders as boundary conditions are applied. The communication between processors is achieved through the MPI-library and a space filling curve algorithm is used for load-balanced data distribution as the refinement mesh is evolving during the course of a simulation. The parallel distribution is accomplished as a rigorous mesh decomposition in entities of the base level mesh, but note that the workload of all higher refinement levels including time step refinement is considered [16].

Adaptation along discontinuities can be achieved by evaluating gradients multiplied by the step size in all directions (*scaled gradient*). Cell(j, k) is flagged for refinement if at least one of the three relations

$$|w(\mathbf{Q}_{j+1,k}) - w(\mathbf{Q}_{j,k})| > \epsilon^w, \; |w(\mathbf{Q}_{j,k+1}) - w(\mathbf{Q}_{j,k})| > \epsilon^w, \\ |w(\mathbf{Q}_{j+1,k+1}) - w(\mathbf{Q}_{j,k})| > \epsilon^w$$

is satisfied for an arbitrary scalar quantity w, which is derived from the numerical vector of state $\mathbf{Q}^\ell(t)$ on level ℓ. The constant ϵ^w denotes the prescribed refinement threshold.

Central to the block-structured mesh refinement approach is the utilisation of a dedicated *cluster* algorithm to create blocks from individual cells tagged for refinement.

u_x	$5\left[\tanh\left(20y+10\right)-\left(\tanh\left(20y-10\right)+1\right)\right]$
u_y	$\frac{1}{4}\sin\left(2\pi x\right)\left(e^{-100(y+\frac{1}{2})^2}-e^{-100(y-\frac{1}{2})^2}\right)$

Table 1: Kelvin-Helmholtz instability: Initial condition for u_x and u_y.

We use a recursive algorithm proposed by Bell et al. [17] for this purpose. The algorithm starts with the steepest zero crossing and uses recursively weaker ones, until the ratio between flagged and all cells in every new mesh is above the prescribed value $0 < \eta \leq 1$. In practice, we use $\eta = 0.7$ on the present computations. A buffer zone of one cell is added around tagged cells to avoid degradation of results from interpolation. More details about these strategies can be found in [14].

3. Results

3.1. Kelvin-Helmholtz instability

In the context of space sciences, the Kelvin-Helmholtz (KH) instability appears in many phenomena such as the solar corona, the ionosphere and astrophysical objects. This instability is a phenomenon which occurs in single continuous fluids with a velocity shear layer or at the interface of two fluids with different velocities [18]. In this test, in order to study KH symmetry, two velocity shear layers in opposite directions are inserted at the lines $y = 0.5$ and $y = -0.5$. Along with these shear layers, a perturbation in the u_y component is inserted in order to create a vortex. The perturbations in u_y around $y = 0.5$ and $y = -0.5$ are in opposite direction, hence the vortices will be counter-rotating. The initial configuration is described in [8] by the values $\rho = 1$, $p = 50$, $B_x = 1$, $u_z = B_y = B_z = 0$. The settings for u_x and u_y are given in Table 1.

The computational domain is $[0, 1]\times[-1, 1]$ with periodic boundary conditions and the computations reach time $t_{\text{end}} = 0.5$. The simulation parameters are $\nu = 0.4$, $\gamma = 1.4$. As refinement criteria for the SAMR algorithm, we used the scaled gradient threshold $\epsilon^{\rho} = 0.01$, as applied to the density field only. Note that in these tests the finest level is always at a resolution equivalent to a uniform mesh with 2048^2 cells but the number of levels used is varied between one and three, using a refinement factor 2 on all levels.

The numerical solution at t_{end} obtained for the variable pressure in the case of three refinement levels running on eight processors is given in Fig. 1 (left). This picture uses ten contour curves, from minimum value to maximum pressure value. Super-imposed are the refinement levels of the mesh, where red is the most refined and blue is the coarsest level. We observe that the refinement zones agree with the isolines and the symmetry of the instability is almost preserved. Figure 1 (right) shows the distribution of the cells updated by each processor related to this solution.

The $\mathbb{L}^1$ errors for the physical variables ρ, p, and the maximum of the errors among all the components of the magnetic field B_i and velocity field u_i are presented in Table 2. The errors double between the refinement levels for all variables, which is expected as the computations with higher refinement use consecutively coarser cells on the base mesh. The errors in the density are the smallest, whilst pressure has the largest ones, as expected.

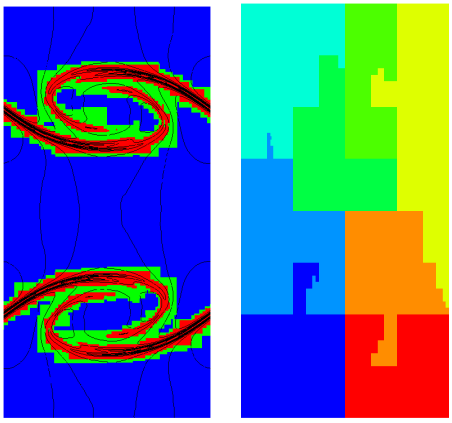

Figure 1: Kelvin-Helmholtz instability: Contour plot of pressure with background colour according to refinement level (left). Mesh distribution in each processor at t_{end} (right).

Levels	Variables			
	ρ	p	$\max(B_i)$	$\max(u_i)$
2	0.01	0.65	0.12	0.13
3	0.02	1.49	0.25	0.29

Table 2: Kelvin-Helmholtz instability: $\mathbb{L}^1$ errors considering uniform mesh of 2048^2 cells using two processors.

To verify the scalability of the parallel algorithm, we consider simulations with one to three refinement levels, where the most refined mesh is 2048^2 cells for every number of levels, using 2^n processors, with $n = 0$ to 4. The computational time in minutes of those experiments are given in Table 3. As expected, considering the uniform mesh there is a time decay of roughly $\frac{1}{2}$ between 2^n and 2^{n+1} processors. Similar results can be observed for varying refinement levels.

3.2. Orszag & Tang vortex

This test problem has been introduced by Orszag and Tang [19]. Since then, it has been extensively used in verification tests of ideal MHD simulations, for instance, in [20, 21, 22]. Due to its physical characteristics it is also a well-known model for 2D turbulence, as described in [23] using Fourier spectral methods. In detail, it verifies the transitions in MHD structures, and consequently how robust the code is at handling the formation of MHD shocks, shock-shock interactions, and moreover, it helps in the identification of how significant magnetic monopoles affect the numerical solutions.

The initial conditions are for both cases $\rho = \gamma^2$, $p = \gamma$, $B_x = -\sin(y)$, $B_y = \sin(2x)$, and $B_z = 0$. The values for the velocity components are given in Table 4. Note that in the 3D case a perturbation parameter $\varepsilon_p = 0.2$ is included in addition into all components, following the example presented in [24].

		Processors				
		1	**2**	**4**	**8**	**16**
Levels	**1**	5790	3214	1597	825	573
	2	1187	627	300	165	113
	3	411	235	129	82	66

Table 3: Kelvin-Helmholtz instability: Elapsed time, in minutes, of the computations using one to three refinement levels as a function of the number of processors. Simulations were performed using two nodes in a cluster with processors Intel Xeon 2.20 GHz with 20 cores each, dividing the processes equally among the two nodes.

	2D	**3D**
u_x	$-\sin(y)$	$-\left[1+\varepsilon_p \sin(z)\right]\sin(y)$
u_y	$\sin(x)$	$\left[1+\varepsilon_p \sin(z)\right]\sin(x)$
u_z	0	$\varepsilon_p \sin(z)$

Table 4: Orszag-Tang vortex: Initial condition for velocity field.

The computational domain is $[0, 2\pi]$ in every direction with periodic boundary conditions until the final time $t_{\text{end}} = \pi$. These simulations are done using $\gamma = \frac{5}{3}$, and $\nu = 0.4$, and 0.3, with scaled gradient threshold applied to the density field of $\epsilon^{\rho} = 0.1$ and 0.5 for 2D, and 3D cases, respectively.

To visualize the local convergence of the Orszag-Tang vortex, we present a cut of the pressure field at $y = 0.64\pi$ and compare it to the results obtained in the literature (Fig. 2). All experiments were run with a 200^2 mesh, except the Miyoshi-Kusano one, which used a mesh size of 192^2. The FLASH solutions are computed using the HLLD flux and the eight-wave divergence cleaning scheme. The Londrilllo-Del Zanna result uses the third order LF–CENO scheme and staggered collocation for the magnetic field [22]. The Miyoshi-Kusano results used the HLLD flux and the mixed GLM divergence cleaning [11] as we do. Our results are obviously in good agreement with respect to the main structures of the solution.

For the parallel tests, our simulations are performed with one to three refinement levels, where the most refined mesh is 2048^2 (2D) and 128^3 (3D). In Fig. 3, we present similar graphs as in Fig. 1 for this 2D case. As desired, the refinement agrees with the structures present in the solutions, and there is a uniform distribution among the processors considering the refinement regions. The errors, using a $\mathbb{L}^1$ norm in relation with the solution in a uniform mesh are presented in Table 5. In this case, density and pressure present similar errors and the largest values are for two and three refinement levels. Again, for three decomposition levels the error roughly doubles in relation to the second refinement level. In Fig. 4, the pressure solution and the mesh distribution among the processors are presented at t_{end}. The solution symmetries are almost preserved, as desired, and the processor distributions follow well-balanced patterns observed already in the 2D cases.

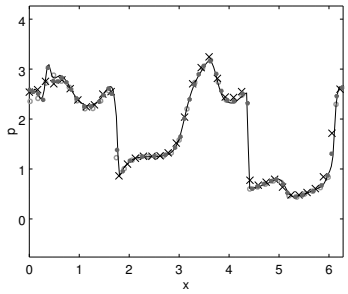

Figure 2: 1D cut comparison of pressure solutions at $y = 0.64\pi$ at t_{end} obtained from our simulations (line) with other simulations: Londrilllo-Del Zanna (cross), Miyoshi-Kusano (dot), and FLASH eight-wave (circle).

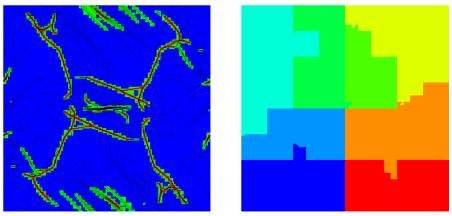

Figure 3: 2D Orszag-Tang vortex: Contour plot of pressure with background colour according to refinement level (left) and mesh distribution in each processor at the final iteration (right).

The computational time in minutes for 2^n processors with $n = 0$ to 4 of those experiments are given in Table 6. In the 2D experiments after four processors the scalability is close to two. In 3D, all experiments present scalability near two, except for 16 processors, where probably the communication costs dominate due to the small problem size.

3.3. Shock-Cloud iteration

To check the performance of the numerical scheme when dealing with super-fast flows, this problem presents a disruption of a high-density magnetic cloud by a strong shock wave, as described in [1]. The initial condition is constructed by considering two regions, the first one defines the advancing plasma – which causes the shock – and the other is a stationary state where the shock advances. These regions are limited by the

	Levels	Variables			
		ρ	p	$\max(B_i)$	$\max(u_i)$
2D	**2**	0.62	0.64	0.32	0.23
	3	1.24	1.39	0.68	0.48

Table 5: Orszag-Tang vortex: $\mathbb{L}^1$ errors for 2D cases.

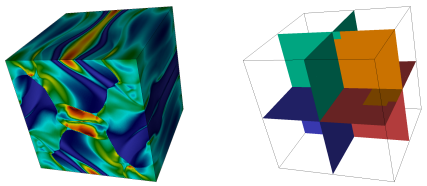

Figure 4: 3D Orszag-Tang vortex: Solution for pressure (left) and mesh distribution in each processor (right) at t_{end}.

	Level	Processors				
		1	**2**	**4**	**8**	**16**
2D	**1**	1174	632	306	160	98
	2	409	240	136	77	41
	3	351	176	89	49	29
3D	**1**	99	69	28	17	11
	2	68	41	21	12	8
	3	69	50	20	12	8

Table 6: Orszag-Tang vortex: Elapsed time of the computations using one to three refinement levels as a function of the number of processors. Simulations were performed using two nodes in a cluster with processors Intel Xeon 2.20 GHz with 20 cores each. Here, we divided the processes equally among the two nodes.

domain boundaries and a plane parallel to the yz plane at $x = 0.05$. Inside the second region, we define the cloud as a high density region in hydrostatic equilibrium with the surrounding plasma.

We consider the cloud region as a sphere with centre at $(0.25, 0.5, 0.5)$ and radius $r_0 = 0.15$. The advancing plasma initial condition is given by $\rho = 3.86859$, $p = 167.345$, $v_x = 11.2536$, $v_y = v_z = B_x = 0$, $B_y = 2.1826182$ and $B_z = -B_y$. The initial configuration of the stationary state is given by $p = 1$, $\mathbf{u} = 0$, $B_y = B_z = 0.56418958$. The density ρ is 10 inside the cloud and 1 otherwise. The computational domain is $[0, 1]^3$ with outlet boundaries until the time $t_{\text{end}} = 0.06$. These simulations use the parameters $\nu = 0.3$, $\gamma = \dfrac{5}{3}$, and the scaled gradient threshold $\epsilon^\rho = 8.0$.

As in the other two configurations, the simulations were performed with one to three refinement levels, where the most refined mesh is 128^3 cells, using 2^n processors with n from one to three. The pressure solution on the adaptive mesh at t_{end} is presented in Fig. 5 (left). We can observe that the symmetry is almost perfectly preserved and the adaptive mesh refines all relevant structures, particularly the bow shock. The mesh distribution is well balanced considering, again, the difference in computational costs required by the adaptive meshes. Similar to the KH instability, density has the lowest errors and pressure the largest values as usual for these cases. Similarly to the other configurations, the errors increase when more aggressive mesh adaptation, *i.e.*, a higher

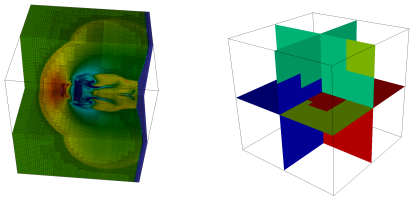

Figure 5: 3D Shock-Cloud interaction: pressure plotted over the adapted mesh (left), and mesh distribution in each processor (right) at t_{end}.

Levels	Variables			
	ρ	p	$\max(B_i)$	$\max(u_i)$
2	0.04	1.47	0.07	0.08
3	0.13	6.24	0.20	0.40

Table 7: 3D Shock-Cloud interaction: $\mathbb{L}^1$ errors.

number of levels, is employed.

The computational times in minutes of those experiments are given in Table 8. We observe that in these experiments the maximum scalability is near 1.8 for one level and one to two processor, and it reduces to a factor of 1.2 for level two and comparing the one and two processor cases.

4. Conclusions

In this paper we presented some widely used verification tests for compressible ideal MHD in our new MHD solver in the AMROC framework. For one test case with available results, our solutions are in good agreeement with predictions from similar codes. Moreover, the accuracy of the numerical solutions also exhibits the expected behaviour comparing the uniform mesh and the adaptive meshes. Scalability and accuracy of the new MHD solver were particularly investigated. In general, all test cases show good parallel performance, confirming the quality of the implementation.

		Processors			
		1	**2**	**4**	**8**
Levels	**1**	79.8	43.1	24.0	17.0
	2	8.2	6.5	3.4	2.5
	3	3.1	2.3	1.5	1.0

Table 8: 3D Shock-Cloud interaction: Elapsed time of the computations using one to three refinement levels as a function of the number of processors. Simulations were performed in a workstation with processors Intel Xeon 2.20 GHz with 12 cores each.

Acknowledgements

The authors thank the FAPESP SPRINT – University of Southampton (Grant:16/ 50016-9), FAPESP (Grant: 2015/ 25624-2), CNPq (Grants: 306038/2015-3, 140626/ 2014-0, 141741/2013-9), and FINEP (Grant: 0112052700) for financial support of this research. ML thankfully acknowledges financial support from CAPES for his doctorate sandwich stage at the University of Southampton. We also thank M. Banik and V. E. Menconi for their helpful computational assistance.

References

[1] G. Tóth, B. van der Holst, I. V. Sokolov, D. L. De Zeeuw, T. I. Gombosi, F. Fang, Adaptive numerical algorithms in space weather modeling, J. Comput. Phys. 231 (3) (2012) 870–903.

[2] J. Bittencourt, Fundamentals of Plasma Physics, Springer, 2004.

[3] P. F. Hopkins, A constrained-gradient method to control divergence errors in numerical MHD, Mon. Notices Royal Astron. Soc. 462 (11) (2016) 576–587.

[4] A. K. F. Gomes, M. O. Domingues, K. Schneider, O. Mendes, R. Deiterding, An adaptive multiresolution method for ideal magnetohydrodynamics using divergence cleaning with parabolic-hyperbolic correction, Appl. Numer. Math. 95 (2015) 199–213.

[5] Domingues, M. O., Gomes, A. K. F., Gomes, S. M., Mendes, O., Di Pierro, B., Schneider, K., Extended generalized lagrangian multipliers for magnetohydrodynamics using adaptive multiresolution methods, ESAIM: Proc. 43 (2013) 95–107.

[6] A. Majda, Compressible fluid flow and systems of conservation laws in several space variables, Springer, 1984.

[7] J. U. Brackbill, D. C. Barnes, The effect of nonzero $\nabla \cdot \mathbf{B}$ on the numerical solution of the magnetohysdrodynamic equations, J. Comput. Phys. 35 (1980) 426–430.

[8] A. Dedner, F. Kemm, D. Kröner, C.-D. Munz, T. Schnitzer, M. Wesenberg, Hyperbolic divergence cleaning for the MHD equations, J. Comput. Phys. 175 (2) (2002) 645–673.

[9] A. Mignone, P. Tzeferacos, A second-order unsplit Godunov scheme for cell-centered MHD: The CTU-GLM scheme, J. Comput. Phys. 229 (6) (2010) 2117–2138.

[10] R. J. Leveque, Finite Volume Methods for Hyperbolic Systems, Cambridge University Press, 2002.

[11] T. Miyoshi, K. A. Kusano, A multi-state HLL approximate Riemann solver for ideal magnetohydrodynamics, J. Comput. Phys. 208 (2005) 315–344.

[12] E. F. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics, Springer, 1999.

[13] M. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. 82 (1989) 64–84.

[14] R. Deiterding, Block-structured adaptive mesh refinement - theory, implementation and application, ESAIM: Proc. 34 (2011) 97–150.

[15] R. Deiterding, Parallel adaptive simulation of multi-dimensional detonation structures, Ph.D. thesis, Brandenburgische Technische Universität Cottbus (Sep 2003).

[16] R. Deiterding, Construction and application of an AMR algorithm for distributed memory computers, in: T. Plewa, T. Linde, V. G. Weirs (Eds.), Adaptive Mesh Refinement - Theory and Applications, Springer, 2005, pp. 361–372.

[17] J. Bell, M. Berger, J. Saltzmann, M. Welcome, Three-dimensional adaptive mesh refinement for hyperbolic conservation laws, SIAM J. Sci. Comput. 15 (1994) 127–138.

[18] A. Frank, T. W. Jones, D. Ryu, J. B. Gaalaas, The magnetohydrodynamic Kelvin-Helmholtz instability: A two-dimensional numerical study, Astrophys. J. 460 (1996) 777–793.

[19] S. A. Orszag, C.-M. Tang, Small-scale structure of two-dimensional magnetohydrodynamic turbulence, J. Fluid Mech. 90 (1) (1979) 129–143.

[20] D. Ryu, F. Miniati, T. Jones, A. Frank, A divergence-free upwind code for multidimensional magnetohydrodynamic flows, Astrophys. J. 509 (1) (1998) 244–255.

[21] W. Dai, P. R. Woodward, A simple finite difference scheme for multidimensional magnetohydrodynamical equations, J. Comput. Phys. 142 (2) (1998) 331–369.

[22] P. Londrillo, L. Del Zanna, High-order upwind schemes for multidimensional magnetohydrodynamics, Astrophys. J. 530 (1) (2000) 508–524.

[23] J. M. Picone, R. B. Dahlburg, Evolution of the Orszag–Tang vortex system in a compressible medium. II. Supersonic flow, Phys. Fluids B 3 (1) (1991) 29–44.

[24] C. Helzel, J. A. Rossmanith, B. Taetz, An unstaggered constrained transport method for the 3D ideal magnetohydrodynamic equations, J. Comput. Phys. 230 (10) (2011) 3803–3829.