A General Code for Modeling MHD flows on Parallel Computers: Versatile Advection Code

Gábor Tóth

Sterrenkundig Instituut, Utrecht University
Postbus 80000, 3508 TA Utrecht, The Netherlands
E-mail: toth@fys.ruu.nl

(Received)

A general tool for solving MHD and hydrodynamical problems typical of astrophysical applications is designed and implemented. The code allows the user to solve a hyperbolic system of partial differential equations with a variety of modern high resolution numerical schemes on 1, 2 or 3D grids with slab or axial symmetry. The equations may contain fluxes and source terms. The initial and boundary conditions can be chosen from a number of implemented types, but the code is structured in a manner that allows the user to write his/her own subroutines for a new type of initial or boundary condition, or – as a matter of fact – for a new set of equations. Alternative numerical schemes can be added with relatively small effort. The choice of algorithms allows the user to find the optimal one for the application at hand and/or to compare results of different schemes thus gaining insight into the numerical effects. An emphasis is put on independence of computer platforms, which is very important in parallel computational environments, where standardization has not taken place yet. Efficiency of the code is also a high priority, the options available for the user should not compromise the performance.

1. INTRODUCTION

Numerical simulation is an important tool to investigate astrophysical hydrodynamical and magneto-hydrodynamical problems. Though the physical assumptions vary a lot among the applications, in most cases a conservative system of hyperbolic partial differential equations

$$\partial_t \vec{U} + \partial_i \vec{F}_i = \vec{S}$$

has to be solved, where $\vec{U}$, $\vec{F}_i$ and $\vec{S}$ are the conservative variables, the fluxes and the source terms, respectively, and i represents the 1, 2 or 3 spatial coordinates. In hydrodynamics, for example, $\vec{U}$ consists of mass, momentum and total energy densities, in MHD the magnetic field components are the additional variables. The source terms may represent external gravitational forces, radiative cooling or heating terms etc. If the resistivity and/or viscosity is small but not negligible, their effect may be represented by source terms depending on $\vec{U}$ and its gradients. In 1 or 2D one may assume slab or axial symmetry in the ignored direction(s). In axial symmetry source terms appear due to pressure, Coriolis and centrifugal forces etc.

Using the conservative form of the equations for numerical discretisation have two major advantages. First, the global conservation of mass, momentum, energy and magnetic induction is ensured, second, appropriate numerical schemes can find weak solutions with discontinuities. In astrophysical problems the flows are often supersonic or super Alfvénic, thus the proper representation of shock waves and other discontinuities is necessary. Shock capturing hydrodynamical schemes have been developed and used for industrial and research purposes for many years, but their generalization to the MHD equations have been started relatively recently.

In the Versatile Advection Code the modern high resolution schemes are applied to a general system of hyperbolic equations, which is specified in a separate module, thus it can solve the hydrodynamic or the ideal MHD equations in a similar fashion. Since the methods generalize to multidimensional systems of equations, the code is written in a dimension independent notation which is translated by a preprocessor to a Fortran 90 or Fortran 77 source code. The preprocessor can configure VAC to solve 1, 2 or 3D problems. A single general code can be developed, maintained and modified more efficiently than the typical single purpose numerical codes. In the design of this software a preference is given to general methods over dimension-, equation- or problem-dependent algorithms, but the performance in terms of speed or accuracy should not be compromised.

2. STRUCTURE OF SOURCE CODE AND COMPILATION

The source code is divided into smaller units, or modules. Some modules have several versions, for example the VACPHYS module which contains the equation specific information: fluxes, sources, the eigenvalues and eigenvectors for the approximate Riemann solver, etc. Each system of equations is specified by its respective VACPHYS.EQUATION module. The boundary conditions and the initial conditions are often problem specific, these special subroutines are collected in the VACUSR.PROBLEM modules. The source code is precompiled and compiled into two executable programs, `vacini` and `vac`. The corresponding main programs and major subroutines are found in the VACINI and VAC modules. As the name suggests, `vacini` generates the initial conditions, while `vac` evolves the solution in time. Both programs use the input and output subroutines of the VACIO module. The predefined boundary conditions, and other grid related subroutines are in VACGRID, finally the numerical algorithms are grouped into the modules VACTVD and VACFCT, which contain the Flux Corrected Transport (FCT) and Total Variation Diminishing (TVD) schemes, respectively.

Since the same source code is used for 1, 2 or 3D simulations, a special purpose preprocessor VACPP translates the dimension-independent notation into standard Fortran 90. Fortran 90 will probably become the standard computer language for scientific software. It expresses parallelism clearly, thus most parallel computers alreadt have (sometimes limited) Fortran 90 compilers that can compile the code without the need of explicit message passing. Fortran 90 is fully compatible with Fortran 77 which allows combining the code with existing Fortran routines on workstations or vector super computers. At the moment, however, Fortran 90 compilers are not yet available for many non-parallel computers, thus a freely distributed software, ADAPTOR by Brandes (1994) may be used to translate the Fortran 90 source to Fortran 77. For parallel computers without an efficient Fortran 90 compiler ADAPTOR can also produce the necessary message passing. This feature will be tested for VAC in the near future. On the other hand VAC was successfully transported to the Connection Machine, CM5. A Perl script translates the Fortran 90 source code to CMFortran, the data-parallel language of the Connection Machine, which is a limited version of Fortran 90. Simple Perl scripts and a makefile take care of all the precompilation procedures. Perl is already installed on most UNIX systems, but in any case it is a freely distributed software by L. Wall (1986).

3. AVAILABLE OPTIONS

The Versatile Advection Code is under development, but it has already reached a stage when it can be used for scientific research. The modular structure of the code allows for a choice of the equations to be solved. At the moment five systems of equations are available: the simple transport equation for test purposes, the Euler equations of compressible hydrodynamics with adiabatic or full energy equation, and the ideal MHD equations with isothermal or full energy equation. For 2 or 2+1/2 dimensions the resistive MHD equations are also implemented.

The grid is presently restricted to be Cartesian but it may be non-uniform. The number of spatial dimensions $1 \leq D \leq 3$ and the number of components $D \leq C \leq 3$ for the vector variables (e.g. the momentum and the magnetic field) may have 6 different combinations. Slab, cylindrical, spherical or nozzle symmetry may be assumed in 1D for the other two spatial directions, and slab or cylindrical symmetry in 2D. The equation and the dimensionality is set by the preprocessor, the type of symmetry is determined in the parameter file read by the executable program `vac`. At the moment one may configure VAC in some 30 different ways, while typical hydro codes solve only a fixed system of equations, e.g. the 2D axial symmetric ideal MHD equations. To use such a code for 1D hydrodynamic calculations one would have to set the not needed variables, i.e. the magnetic field components, to zero, and to use an elongated 2D grid with a few cells across. This creates a huge overhead for both CPU and memory usage. And, of course, such a code can never do a simulation in 3D, or in 2D with slab symmetry.

At present three algorithms are available for solving the differential equations. The Flux Corrected Transport scheme (FCT), a Total Variation Diminishing (TVD) scheme with a Roe-type Riemann solver (Roe 1981), and a Lax-Friedrich type TVD scheme (TVDLF) with no Riemann solver. There are two versions of the FCT scheme, ETBFCT by Boris and Book (1973) and a more recent variation YDFCT by Odstrčil (1993). Both TVD methods (Yee 1989) can use different limiters. All these schemes are explicit, second order in space and time, and they are able to simulate shocks and other discontinuities as well as smooth flows. Multidimensional equations are solved by Strang type dimensional splitting. Source terms may be included in a time split fashion or added at the same time as the fluxes.

The popular FCT schemes solve each equation separately. Artificial diffusion is added only where it is needed for numerical stability. The FCT schemes are good in resolving contact discontinuities, they can handle shocks well, but close to the discontinuities some numerical noise might be generated. The TVD schemes on the other hand are very efficient in maintaining monotone profiles by appropriately limiting the slopes of the variables. The TVD scheme with the approximate Riemann solver can resolve discontinuities as well as or better than FCT without the spurious oscillations, but the Riemann solver is specific to the equation, and it is not as robust as the TVD Lax-Friedrich scheme, which does not need a Riemann solver at all. The TVDLF scheme is somewhat more diffusive than the other two. For a new problem the user may try all three methods, and based on the results may decide which one to use for further investigations. Switching from one method to another is as simple as changing a few characters in the parameter file.

The boundary conditions are implemented by layers of ghost cells around the mesh. The values in the ghost cells are updated in every time step, thus the discretised equations can be applied to the cells inside the physical mesh without making special cases for the edges. This way the boundary conditions are fully specified in the subroutines updating the ghost cells, and the boundary conditions become independent of the choice of the numerical schemes. On each side of the grid the boundary layer is divided into one or more boundary regions. For each boundary region and each variable the type of the boundary is defined in the parameter file. At present there are six predefined boundary types: `periodic`, `symmetric`, `antisymmetric`, `fixed` and `continuous`. The continuous boundary condition is an approximate representation of an open boundary by putting the gradients normal to the boundary to zero in the ghost cells. The periodic boundary regions on the opposite sides of the grid may be shifted sideways relative to each other which creates a shear in the periodicity. Time dependent or more complicated boundary conditions can be realized via the `special` boundary type which results in a call of a user defined subroutine in the VACUSR module.

4. TESTS

The code has been extensively tested (Tóth and Odstrčil 1995) on standard numerical tests. These include the MHD shock tube problem by Brio and Wu (1988), the interaction of two strong hydrodynamic blast waves and the 2D double Mach reflection problems by Woodward and Collela (1984), and the simulation of the Orszag-Tang MHD vortex system in 2D (see Figure 1 and compare it with Zachary et al. 1994). To demonstrate the ability of VAC to solve 3D problems the 1D MHD shock tube problem was rotated into the diagonal direction of a 3D cube. This test showed that the dimensionally split algorithms are quite accurate even when the discontinuities are not aligned with the axes of the grid. Though at the moment the divergence of the magnetic field is not forced to be zero numerically, this did not seem to cause serious errors in the multidimensional MHD simulations.

VAC was compared with three other codes employing similar FCT or TVD algorithms by solving identical problems on a DEC Alpha work station. In spite of its general design, VAC was faster than or as fast as the other less general codes. On the DEC Alpha workstation the VACPP precompiled source code can be compiled directly by the FORTRAN 90 compiler, while on a DEC 5000 workstation and a Cray C90 vector super computer VAC has been successfully run by compiling the ADAPTOR generated Fortran 77 code. On the Connection Machine 5 the CMFortran code generated by a Perl script was used. Tests on other parallel machines with message passing (like the Cray T3D) are planned for the near future.

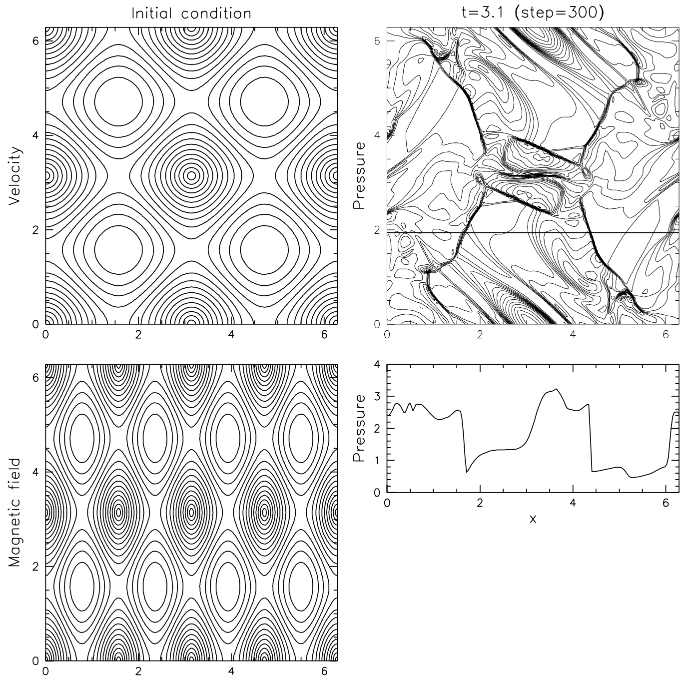

FIGURE 1. Initial condition (left) and a snapshot of the evolution (right) of the Orszag-Tang MHD vortex system. The boundaries are periodic, the calculations were done by the TVD scheme with the approximate Riemann solver on a 192×192 grid.

5. FUTURE PLANS

VAC will be extended and developed further. A projection scheme (see Brackbill and Barnes 1980) which can eliminate the divergence of the magnetic field without changing the vector potential or the currents is going to be added. The projection scheme can be used in combination with any of the numerical algorithms. A spatially third order MUSCL TVD scheme using the same approximate Riemann solvers as the current TVD scheme can be implemented with no difficulty. The fully multidimensional FCT scheme with Zalesak's (1979) limiter is also among the plans requiring little extra coding. The restriction to Cartesian grids may be eliminated by the use of generalized coordinates. This is relatively straightforward for the TVDLF and FCT methods, but more complicated for the Riemann solvers. Non-linear Riemann solvers, semi-implicit and implicit methods are also considered for future developments.

ACKNOWLEDGMENTS

The Versatile Advection Code is being developed at the Astronomical Institute at Utrecht. The project is done in a collaboration with the FOM Institute for Plasma Physics, the Mathematics department at Utrecht and the CWI at Amsterdam. The work has been funded by the Dutch Science Fundation (NWO) in a program of Massive Parallel Processing Applied to Magnetohydrodynamics. The author also receives partial support from the Hungarian Science Foundation, grant F 017313. The author acknowledges collaboration on testing the code and on the implementation of the FCT algorithms with D. Odstrčil, and thanks P. Meijer for his contribution in porting the code to the Connection Machine and the Cray C90. A shorter version of this paper will appear in the proceedings of the IAU Colloqium 153, held in Makuhari, Japan, May 22-26, 1995.

REFERENCES

J. P. Boris and D. L. Book, Flux-corrected transport. I. SHASTA, A fluid transport algorithm that works, *J. Comput. Phys.* **11**, 38 (1973).

J. U. Brackbill and D. C. Barnes, The effect of nonzero $\nabla \cdot B$ on the numerical solution of the magnetohydrodynamic equations, *J. Comput. Phys.* **35**, 426 (1980).

T. Brandes, ADAPTOR, Automatic DAta Parallelism translatOR, available via anonymous ftp from e.g. `ftp.gmd.de:gmd/adaptor`, (1994).

M. Brio and C. C. Wu, An upwind differencing scheme for the equations of ideal magnetohydrodynamics, *J. Comput. Phys.* **75**, 400 (1988).

D. Odstrčil, Improved FCT algorithm for shock hydrodynamics, *J. Comput. Phys.* **108**, 218 (1993).

P. L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, *J. Comput. Phys.* **43**, 357 (1981).

G. Tóth and D. Odstrčil, Comparison of some FCT and TVD schemes for hydrodynamic and MHD problems, submitted to *J. Comput. Phys.*, (1995).

L. Wall, Perl, A Practical Extraction and Report Language, available via anonymous ftp from e.g. `ftp.uu.net:languages/perl`, (1986).

P. R. Woodward and P. Colella, The numerical simulation of two-dimensional flows with strong shocks, *J. Comput. Phys.* **54**, 115 (1984).

H. C. Yee, A class of high-resolution explicit and implicit shock-capturing methods, *NASA TM-101088* (1989).

A. L. Zachary, A. Malagoli, and P. Colella, A higher-order Godunov method for multidimensional ideal magnetohydrodynamics, *SIAM J. Sci. Comput.* **15**, 263 (1994).

S. T. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, *J. Comput. Phys.* **31**, 335 (1979).