A&A manuscript no.
(will be inserted by hand later)

Your thesaurus codes are:
?? (02.13.2)

ASTRONOMY
AND
ASTROPHYSICS
11.10.1996

NUMERICAL SIMULATIONS OF MAGNETOHYDRODYNAMIC FLOWS

G. Tóth

Astronomical Institute, Utrecht University
Postbus 80000, 3508 TA Utrecht, The Netherlands
toth@fys.ruu.nl, http://www.fys.ruu.nl/~toth

Talk at Visegrád on May 23, written form Oct 7, 1996

Abstract. This review deals with the numerical simulation of magnetohydrodynamic flows. A general overview of the different numerical techniques compares their advantages, disadvantages and points to the appropriate application areas. Some specific schemes, and their implementation in the Versatile Advection Code are discussed in more detail.

Key words: numerical – magnetohydrodynamics

1. Introduction

There can be several reasons to do hydrodynamic and magnetohydrodynamic (MHD) computer simulations.

In astronomy, in contrast with physics, the observer usually cannot influence the object of the investigation, in other words, one cannot do experiments, only observations. Analytical calculations and numerical simulations provide the only substitute for experiments. Due to the complexity of the physical phenomena analytical calculations are limited to the simplest cases and many approximations have to be made. Numerical simulations have their own limitations too, but they can be used together with observations and analytical calculations in a complementary way which can lead to a deeper understanding of the examined phenomena.

Even in cases when experiments are possible to do, computer simulations may turn out to be more efficient and less expensive. This is the situation in aerospace engineering, where wind tunnel experiments are replaced by simulations for economical reasons. In fact, most of the modern numerical hydrodynamic schemes were invented and developed by researchers working in the aerospace industry. Magnetohydrodynamics has fewer engineering applications than hydrodynamics, although the thermonuclear fusion research also requires computer simulations of magnetised plasma.

Finally, computer scientists and the computer industry are always searching for applications that can prove the usefulness of their software and hardware. The memory and CPU required by three-dimensional MHD calculations are still a great challenge even for the latest parallel super computers. Massively parallel computers, on the other hand, require new numerical algorithms to be developed.

2. Equations

The magnetised plasma can be described by the following primitive variables: the mass density ρ, velocity $\mathbf{v}$, pressure p, and magnetic field $\mathbf{B}$, which are all functions of time and three (if no simplifying symmetry assumption is made) spatial coordinates. For ideal MHD the resistivity η is zero, while in resistive MHD $\eta > 0$ is not negligible at least in some parts of the flow. The units for the magnetic field are chosen such that the magnetic permeability $\mu = 1$.

The MHD equations can be expressed in various mathematical forms, however, for numerical models the conservative form is often preferred: the equations explicitly represent the conservation of mass, momentum, total energy, and induction of magnetic field. The conservative variables are ρ, the momentum density $\rho\mathbf{v}$, the total energy density e, and $\mathbf{B}$. This is especially important if *weak solutions* containing discontinuities are of interest.

In terms of these variables the partial differential equations of resistive MHD are

$$\frac{\partial\rho}{\partial t} + \nabla\cdot(\mathbf{v}\rho) = 0 \quad (1)$$

$$\frac{\partial\rho\mathbf{v}}{\partial t} + \nabla\cdot(\mathbf{v}\rho\mathbf{v} - \mathbf{B}\mathbf{B}) + \nabla p_{tot} = -(\nabla\cdot\mathbf{B})\mathbf{B} \quad (2)$$

$$\frac{\partial e}{\partial t} + \nabla\cdot(\mathbf{v}e + \mathbf{v}p_{tot} - \mathbf{B}\mathbf{B}\cdot\mathbf{v} - \mathbf{B}\times\eta\mathbf{J}) = -(\nabla\cdot\mathbf{B})\mathbf{B}\cdot\mathbf{v} \quad (3)$$

$$\frac{\partial\mathbf{B}}{\partial t} + \nabla\cdot(\mathbf{v}\mathbf{B} - \mathbf{B}\mathbf{v}) + \nabla\times(\eta\mathbf{J}) = -(\nabla\cdot\mathbf{B})\mathbf{v} \quad (4)$$

where the total pressure, total energy density, and current density are

$$p_{tot} = p + \mathbf{B}^2/2 \tag{5}$$
$$e = p/(\gamma - 1) + \rho v^2/2 + \mathbf{B}^2/2 \tag{6}$$
$$\mathbf{J} = \nabla \times \mathbf{B} \tag{7}$$

for an ideal gas with adiabatic index γ.

The *initial condition* should satisfy

$$\nabla \cdot \mathbf{B} = 0 \tag{8}$$

The exact solution of the MHD equations (1-4) keeps $\nabla \cdot \mathbf{B} = 0$ indefinitely, thus the terms on the right hand sides of equations (2-4) should always remain zero. In multidimensional numerical calculations, however, the discretisation errors may produce a finite divergence of the magnetic field unless the scheme is specifically designed to keep the discretised form of $\nabla \cdot \mathbf{B}$ zero. It was found by Powell (1994) that writing the MHD equations in the above form together with a modified Riemann solver is stable unlike the usual form in which these "corrective source terms" are dropped. Tóth and Odstrčil (1996) found that Powell's source terms are beneficial for the flux-corrected transport (FCT) and total variation diminishing Lax-Friedrich (TVDLF) schemes as well. Note, that these terms are not in a conservative form, but they are usually small. Other methods to eliminate errors in $\nabla \cdot \mathbf{B}$ will be discussed in section 5.

In general a system of conservation laws with some extra source terms can be written as

$$\partial_t \boldsymbol{U} + \partial_i \boldsymbol{F}_i(U) = \boldsymbol{S}(U) \tag{9}$$

where $\boldsymbol{U}$, $\boldsymbol{F}_i$, and $\boldsymbol{S}$ contain the conservative variables, the fluxes, and the source terms respectively, and ∂_i represents the spatial derivative in direction i, and a summation is implied over $i = 1, 2, 3$.

3. Discretisation in Space

In this section a very concise overview of the different approaches towards spatial discretisation is given. The interested reader may look at standard text books (e.g. Fletcher 1991) describing the standard discretisation techniques and read some more recent articles (e.g. Lele 1992, Yee 1996, and Lowrie et al 1995) on the relatively new compact schemes.

3.1. Finite Differences

The simplest discrete representation of spatial derivatives is by finite differences. The variables are given as pointwise values U_i at locations x_i. Difference formulas of a given order of accuracy can be derived from Taylor expansion around the grid points. The advantage of finite differences is its simplicity, it can be implemented easily and efficiently. The main disadvantage is that there is no guarantee for conservation of quantities on non-uniform grids.

3.2. Finite Volume

In the finite volume discretisation space is divided into grid cells, and the cell-averages $V_i^{-1} \int_{V_i} U$ are known for each cell. The differential equations are discretised in their integral form, i.e. the fluxes through the cell interfaces are added to and subtracted from the cell-averages. This method automatically leads to a conservative discretisation on arbitrary structured or unstructured grids. The disadvantage of finite volumes is in the difficulty to go higher than second order spatial accuracy and the relatively complicated representation of second derivatives in space.

3.3. Finite Elements

In the finite elements method the nodal values U_i are given at the x_i points (nodes), while at other locations U is represented by localised interpolating polynomials, i.e. the solution consists of finite elements. The discretisation of the partial differential equations happens by the weighted residuals method. The advantage of the finite element discretisation lies in its systematic approach to high-order accuracy on arbitrary grids by use of high order interpolation polynomials. On the other hand it is rather complicated, and even with explicit time integration schemes it requires the inversion of huge linear systems. Discontinuities are also a problem for standard finite element methods.

3.4. Pseudo-Spectral Methods

A further step from the local description towards a global description is representing U by a linear combination of some orthogonal functions, e.g. by its Fourier components. Again the method of weighted residuals can be applied to obtain the discretised equations. In practice non-linear terms in the equations have to be evaluated in physical space since the corresponding convolutions in Fourier space are very expensive. This combination of spectral and physical space is called the pseudo-spectral method. The transformation between these two discrete sets of variables can be efficiently done by fast Fourier transforms (FFT). Pseudo-spectral methods can be extremely accurate with only a few modes representing U, this is referred to as *exponential convergence*, since the discretisation error decreases exponentially with the increasing number of modes. Disadvantages of this method include the problems with satisfying arbitrary boundary conditions, and accurate representation of discontinuous solutions.

3.5. Compact Schemes

In the computational fluid dynamics research there is a new trend of developing compact schemes with high order accuracy, but localised description of data. There are several approaches to obtain such a scheme: Padé discretisation of derivatives (implicit finite differences); using the multi-dimensional information around a grid point rather than extending the stencil in one spatial direction (multi-D finite difference); or storing higher order moments of U in the grid cells (generalisation of finite volume, or discontinuous finite elements). Although these schemes are very promising they are still in an experimental stage at the moment.

4. Discretisation in Time

4.1. Explicit

In explicit schemes the fluxes and the sources are calculated at the n-th time level and their contribution is added to the current values of U. The general conservation law (10) can be discretised in time as

$$\boldsymbol{U}^{n+1} = \boldsymbol{U}^n + \Delta t[-\partial_i \boldsymbol{F}_i(\boldsymbol{U}^n) + \boldsymbol{S}(\boldsymbol{U}^n)] \qquad (10)$$

This is a first order approximation in Δt, but it is easy to form higher order Runge-Kutta or predictor-corrector type schemes.

Explicit time integration is simple and fast, and it is easily parallelised for parallel computers. The main disadvantage is that the time step Δt is limited by numerical stability requirements. For the MHD equations the stability condition requires that the time step is shorter than the crossing time of the grid cells by the fast magnetosonic wave, $\Delta t < \Delta x / c_{fast}$. For resistive MHD an additional constraint is the diffusion time, $\Delta t < (\Delta x)^2/\eta$. In cases when fast waves are really present and the time evolution should be accurately simulated, the stability limitation coincides with the accuracy limitation, since one cannot calculate the interaction of fast waves without representing them at the discrete time levels in each cell they are passing through.

In some time accurate calculations, however, the fast waves that set the limit on the time step are not actually present at all, thus the accuracy would not require short time steps, and the explicit scheme may be very inefficient. This is also the case in steady state calculations, where the transients leading to the final state are of no interest.

4.2. Implicit

Stability of the time integration can be achieved by implicit time integration

$$\boldsymbol{U}^{n+1} = \boldsymbol{U}^n + \Delta t[-\partial_i \boldsymbol{F}_i(\boldsymbol{U}^{n+1}) + \boldsymbol{S}(\boldsymbol{U}^{n+1})] \qquad (11)$$

Note that the fluxes and the sources are evaluated at the next time level. This simple scheme is only first order accurate in time, but that can be improved easily for time accurate calculations.

While the stability of the implicit time discretisation is a great improvement over explicit schemes, one has to solve the implicit equations for the unknown $\boldsymbol{U}^{n+1}$. Usually a Newton-Raphson method is used which requires inversion of the $[I - \Delta t \partial(\partial_i \boldsymbol{F}_i)/\partial \boldsymbol{U}]$ matrix. In 1D there are efficient direct solvers, while in multidimensional cases iterative and multigrid methods need to be used. This can be costly both in terms of CPU and storage requirements.

4.3. Semi-implicit

Semi-implicit methods try to combine the stability of implicit methods with the efficiency of explicit methods. The idea is to treat only that part of the equations implicitly, which causes the stability problems. For example, in resistive MHD when the diffusion time $(\Delta x)^2/\eta$ becomes the limiting time for Δt, one can treat the resistive terms implicitly, i.e. calculating them at the next time level, while all the other terms explicitly. There is a lot gained relative to the fully implicit method: the size of the matrix to be inverted decreased since only $\mathbf{B}$ and e are implicit, the matrix elements are much simpler, and the structure of the matrix is symmetric and diagonally dominated, which is much easier to solve with iterative methods than the advection dominated fully implicit problem. The only problem with semi-implicit methods is that they are equation and problem dependent.

5. How to keep $\nabla \cdot \mathbf{B} = 0$

The MHD equations have a special property which does not occur in the hydrodynamic equations, namely the requirement that the magnetic field should remain divergence free. This is automatically satisfied in 1D simulations, but the standard discretisation methods do not guarantee that in multi-dimensional simulations. There are several ways to deal with this problem.

The simplest approach is to increase the spatial resolution until the divergence of $\mathbf{B}$ become small and the solution starts to converge. This may not work if the scheme is unstable due to the errors in $\nabla \cdot \mathbf{B}$, which is the case for some Riemann type solvers. The non-conservative source terms on the right hand sides of equations (2-4) with a modified Riemann solver (Powell 1994) eliminate the instability, but not the error. For some problems the use of Powell's source terms seems to be sufficient, the errors in $\nabla \cdot \mathbf{B}$ remain small, and the conservation of quantities is satisfactory. In other problems, especially for long time integrations, the error may grow to unacceptable values.

Another way to keep $\nabla \cdot \mathbf{B}$ exactly zero is to rewrite the MHD equations by using the *vector potential* $\mathbf{A}$ instead of $\mathbf{B} = \nabla \times \mathbf{A}$. The disadvantage of this approach is that the order of spatial derivatives increases by one, which reduces the order of accuracy by one; the equations became complicated in 3D, and the boundary conditions on the vector potential may not be physically intuitive.

For finite difference schemes the *constrained transport* method by Evans and Hawley (1988) offers a simple and efficient solution by using staggered grids. The magnetic field components are represented on the cell interfaces, while density, momentum and energy in the cell centers. Unfortunately this arrangement excludes finite volume discretisation, and Riemann solver based shock-capturing schemes.

The most general fix to the problem is the *projection scheme* originally proposed by Brackbill and Barnes (1980), which is a correction to the magnetic field after the time step is completed by some arbitrary numerical scheme. The name comes from the idea that the $\mathbf{B}$ field is projected to a divergence free $\mathbf{B}'$ field which is as close as possible to $\mathbf{B}$. To achieve this, one needs to solve a Poisson equation for the potential ϕ

$$\nabla^2\phi = \nabla\cdot\mathbf{B} \tag{12}$$

and then to correct the magnetic field to $\mathbf{B}' = \mathbf{B} - \nabla\phi$. It is easy to show that the correction does not affect the vector potential, the current density or the integrated magnetic flux. In fact one can prove that $\int dx(\mathbf{B}' - \mathbf{B})^2$ is minimal with the constraint $\nabla\cdot\mathbf{B}' = 0$. In summary the projection scheme is the smallest possible correction that removes the unphysical part of the numerical solution without affecting the physically meaningful quantities. The price to pay for these nice properties is the Poisson problem, but that can be solved efficiently with either direct or iterative solvers. It is also important to apply the boundary conditions to $\mathbf{B}'$, and to choose good boundary conditions for the Poisson equation.

6. Implementation and Testing

In the previous sections a great number of options are listed for both space and time discretisation. It is not practical to implement all these schemes and just see which one works best for the problem we want to solve. Instead, one needs to go through a number of steps to make a choice.

First, the class of problems to be solved should be considered. Even if at the beginning it seems that we are interested in doing a single simulation only, unavoidably we will want to solve other related problems later, especially with all the efforts put into the software development. The selected class of problems should be written in a general mathematical form, e.g. like equation (9). Using general notation in the software is preferred, since the different variables can be handled in a uniform manner, and changes to the equation can be done easily.

In the selection of the discretisation technique several things should be taken into account. The geometry of the computational domain determines if a Cartesian grid is sufficient, or the more general structured, or even unstructured grids are necessary. If we expect discontinuities in the solution only shock capturing methods should be used, while smooth solutions can be best approximated with high order methods. The time scales determine if explicit time integration schemes are efficient or not. The hardware to be used depends on the size of the problem and of course on the availability of super computers. If parallel machines are needed, the numerical techniques need to parallelise well.

Before one starts to develop a new code from scratch it is worthwhile to check whether an existing code can be used. It is possible that there already is a working software which needs only minor modifications to solve our problems. Of course, codes written for special problems by other researchers are usually difficult to get, to understand, and to modify. Another option is to use a general software that covers the problems we want to solve. There are a few such general MHD codes publicly available, like the ZEUS code by Stone and Norman (1992) at *http://zeus.ncsa.uiuc.edu:8080/lca_home_page.html*, and the Versatile Advection Code (VAC) by Tóth (1996) at *http://www.fys.ruu.nl/~toth*.

New or modified codes should be thoroughly tested on simple problems with known solutions before an attempt is made to do the simulations we are interested in. The results of the simulations should be checked against physics (e.g. conservation of quantities not built explicitly into the numerical scheme), against other numerical methods (if available), and most importantly, against grid refinement, i.e. *the solution should converge as the spatial and temporal resolution is increased.*

7. Versatile Advection Code

7.1. *Software*

The general purpose VAC software is being developed by the author since 1994. The philosophy behind VAC is using a single versatile software with options and switches for various problems, rather than developing a different method or version for each problem separately. The advantage of such a general approach is a reduction of overall time for software development, easier maintenance, compatibility of different parts, automatic extension of new features to all existing applications. The price of the general approach is some added complexity in the source code.

VAC is written in a dimension independent notation which is translated by a special-purpose preprocessor (Tóth 1996) to Fortran 90, and it can be further translated to Fortran 77. The preprocessor and the translator to Fortran 77 are written in the Perl language. The Perl interpreter is freely available from the net, and is already installed on most UNIX systems. The Fortran 90 code expresses data parallelism clearly, which is important for parallel computers. An automatic translation to high performance Fortran (HPF) is under development. VAC has been installed and is being used on a number of workstations (DEC, SGI, IBM, SUN), on the vector super computer Cray C90, and it has been tested for scaling properties on the parallel Connection Machine CM5.

The source code is divided into smaller units, called 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. This is the only part of the source code which the user is expected to modify, and it can be written in the dimension independent notation as well as Fortran 90 or Fortran 77.

The documentation, about 100 pages, is written in hypertext markup language (HTML), which can be viewed and/or printed from the standard web browsers, e.g. Netscape or Mosaic. There is a user interface written in Perl, which lets the user to configure, edit, compile, and run VAC, and visualise results through the same web browser. Of course, one can do all these steps from the UNIX shell as well. The manual pages and a demo version of the interface can be looked at the author's home page. The latest version of VAC was distributed to 16 users.

7.2. Equations

VAC aims to solve a set of conservation laws with source terms of the form equation (9).

At the moment five VACPHYS.EQUATIONS modules are implemented: the simple transport equation for test purposes, the Euler equations of compressible hydrodynamics with adiabatic or full energy equation, and the resistive MHD equations with isothermal or full energy equation (1-4). Source terms for external gravity, heat conduction, and viscosity can be included and modified as library subroutines. Other source terms can be defined in a user written subroutine in the VACUSR module.

All equation-modules are written in the dimension independent notation, thus 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.

7.3. Geometry

The equations can be solved on a 1, 2 or 3D structured grid with conservative finite volume discretisation. A structured grid can be thought of as a continuous mapping from a Cartesian mesh. Structured grids include Cartesian, polar and spherical grids as trivial examples, but they can be used in more complicated geometries as well. In 1D and 2D both slab and cylindrical symmetry may be assumed for the ignored dimension(s).

Boundary conditions are implemented by two 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. For each boundary and each variable the type of the boundary can be defined. At present there are seven predefined boundary types: *periodic, symmetric, antisymmetric, fixed, fixed-gradient* 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 fixed-gradient type is similar, except that the gradient can be different from zero. Time dependent or more complicated boundary conditions can be realised via the `special` boundary type which results in a call of a user defined subroutine.

7.4. Numerical Schemes

At present four algorithms are available for solving the differential equations. The Flux Corrected Transport scheme (FCT), two Total Variation Diminishing schemes with a Roe-type Riemann solver (TVD and TVD-MUSCL), and a Lax-Friedrich type TVD scheme (TVDLF) with no Riemann solver. See the article by Tóth and Odstrčil for a description of these schemes and the references provided there.

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 either by Strang type dimensional splitting, or by adding the fluxes from all directions at the same time. Source terms may be included in a time split fashion or added at the same time as the fluxes.

The well-known 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 schemes 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 the TVD and TVD-MUSCL schemes are 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.

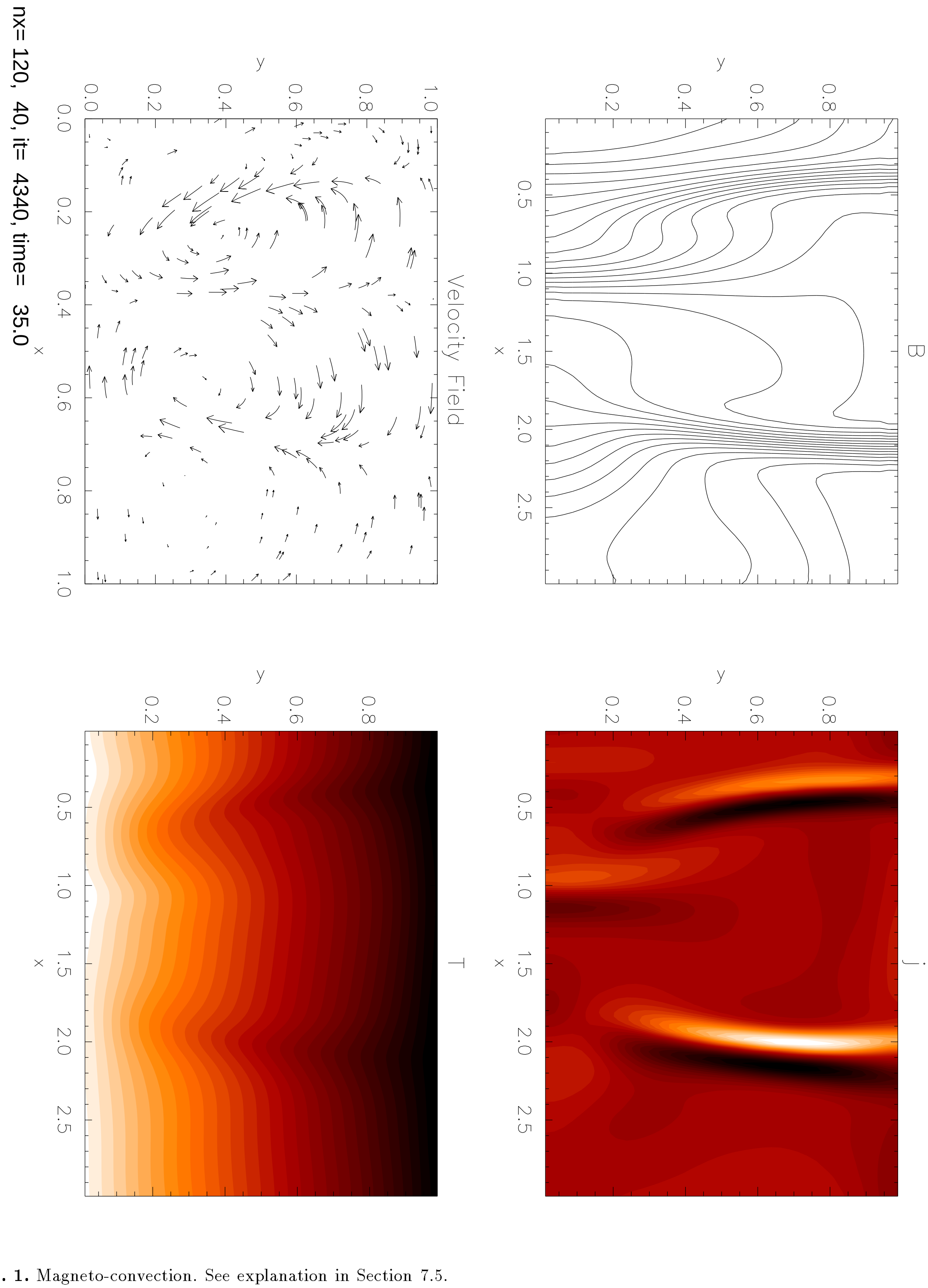

Fig. 1. Magneto-convection. See explanation in Section 7.5.

The projection scheme to eliminate divergence **B** is also implemented, as well as Powell's MHD Riemann solver and the corrective source terms.

7.5. Simulations

The code has been extensively tested (Tóth and Odstrčil 1996) on standard numerical tests. It is currently used in several research projects, with applications ranging from accretion disks, and atmospheres of magnetised stars to solar magnetic flux tubes, coronal holes, prominencies, and reconnection.

As a fairly complex and spectacular example, a simulation of magneto-convection is shown. This simulation has originally been done by Hurlburt and Toomre (1988), and it was implemented for VAC by R. Keppens. The resistive MHD equations (1-4) are solved with external gravity, thermal conduction, and viscosity source terms. The lower and upper boundaries are solid walls of different temperatures, the left and right boundaries are periodic. Initially the stratified gas is in hydrostatic balance, and there is a uniform vertical magnetic field. A small perturbation in the velocity is added. Due to the thermal convection the perturbation grows and after some transients convection cells are formed with circular motion. The magnetic field is expelled from the convection cells, and strong flux sheets are formed between the cells.

The figure shows the magnetic field, the current density, the velocity field and the temperature distribution in the transient stage. The figure is a single frame from the animation which can be produced by the IDL procedure *animate* provided with VAC.

7.6. Development

VAC is a constantly evolving software. Currently the main effort of is put into developing implicit and semi-implicit time integration schemes. Another target is to run the code on massively parallel computers other than the CM5. An automatic translation tool will be developed that can add the parallel HPF directives to the code automatically. First the explicit schemes should become parallelisable, later the implicit algorithms as well.

8. Conclusion

Numerical simulation of MHD flows is not a simple problem. There is no one "perfect" numerical scheme that could solve, or which would be optimal for, all possible problems. It is important to know the available methods, their strengths and weaknesses, before one starts to write a program. Due to the complexity of the numerical algorithms it may be worthwhile to use or modify existing software. This allows the researcher to concentrate on the simulations and not on the software development. Numerical codes and the results of the simulations should always be thoroughly tested and verified. On the other hand, well designed numerical simulations can provide a wealth of information on the physical processes which we could otherwise only observe.

Acknowledgements. The project of Massive Parallel Processing Applied to Magnetohydrodynamics is funded by the Dutch Science Foundation (NWO), and it is done in a collaboration with the FOM Institute for Plasma Physics, the Mathematics Department at Utrecht, and the Center for Mathematics and Computer Science (CWI) at Amsterdam. The author is also supported by the Hungarian Academy of Sciences (OTKA grant F 017313).

References

Brackbill J. U., Barnes D. C., 1980, J. Comput. Phys. 35, 426
DeVore R. C., 1991, J. Comput. Phys. 92, 142
Evans C. R., Hawley J.F., 1988, ApJ 332, 659
Fletcher C. A. J., 1991, Computational Techniques for Fluid Dynamics, Springer, Berlin
Hurlburt N. E., Toomre J., 1988, ApJ 327, 920
Lele S. K., 1992, J. Comput. Phys. 103, 16
Lowrie R. B., Roe P. L.S. K., van Leer B., 1995, AIAA paper 95-1658
Powell K. G., 1994, ICASE Report No 94-24, Langley, VA
Stone J. M., Norman M. L., 1992, ApJS 80, 753
Tóth G., 1996, Astrophys. Lett. & Comm. 34, 245
Tóth G., 1996, The LASY Preprocessor and its Application to General Multi-Dimensional Codes, submitted for publication to J. Comput. Phys.
Tóth G., Odstrčil D., 1996, Comparison of some Flux Corrected Transport and Total Variation Diminishing Numerical Schemes for Hydrodynamic and Magnetohydrodynamic Problems, accepted for publication by J. Comput. Phys.
Yee H. C., 1996, Explicit and Implicit Compact High-Resolution Shock-Capturing Methods I: Formulation, accepted for publication by J. Comput. Phys.

This article was processed by the author using Springer-Verlag LaTeX A&A style file *L-AA* version 3.