Magnetic acceleration and instabilities of astrophysical jets
Moll, R.

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Appendix A

Numerical Methods

A.1 Principles and Terminology in Computational Fluid Dynamics

Below, I give a rough overview on how the (M)HD equations are solved numerically on a grid without going into much detail. For a clear introduction to the subject matter of grid-based numerical HD, the reader is referred to Trac & Pen (2003). The generalization to MHD is explained in the same manner in Pen et al. (2003). Details about the computer code used in the present work can be found in Obergaulinger (2008).

The first step is to discretize space by dividing it into a finite number of small, disjoint volumes called grid cells or zones. It suffices to consider only one dimension whose coordinate shall be $x$, noting that with more dimensions, the algorithm outlined below is successively applied on the different dimensions using the same initial data. The cell which goes from $x_{i-\frac{1}{2}}$ to $x_{i+\frac{1}{2}}$ shall be denoted with $Z_i$.

Except for the induction equation (2.3.1d), which is treated separately, the MHD equations (2.3.1) can be written in the conservation form

$$\partial_t u + \nabla \cdot F = S$$  \hspace{1cm} (A.1)

where the conserved variable $u$ is one of $(\rho,\rho v, e)$, the flux $F$ is accordingly a vector or a rank 2 tensor and $S$ is an optional source term, e.g. gravity. For each cell $Z_i$, the set of conserved variables represents its physical state, either as volume average or as the value at the cell center $x_i$. This state may change due to fluxes across the cell’s boundary, i.e. the cell interfaces at $x_{i\pm\frac{1}{2}}$, or the source term. The fluxes are functions of the conserved variables. Hence, we can compute the fluxes at the cell interfaces if we reconstruct the conserved variables there by making use of an appropriate interpolating function $\phi_i(x)$. The grid cells used in this process are called reconstruction stencil. Several methods exist.

The piecewise-constant method (PCM) uses $\phi_i(x) = u_i$, that is, it simply uses the value at the cell center as an approximation to the value at the cell interface. Obviously, this method is not very accurate, but it is the numerically most stable one. The piecewise-linear method (PLM) uses linear interpolation between the value $u_i$ of
the considered cell and the values \( u_{i\pm1} \) of the neighboring cells. This yields two functions, \( \phi_i^- \) at the left interface \( x_{i-\frac{1}{2}} \) and \( \phi_i^+ \) at the right interface \( x_{i+\frac{1}{2}} \), which may be different in general. It suggests itself to use \( \phi_i^- \) for calculating the flux through the left interface and \( \phi_i^+ \) for calculating the flux through the right interface. However, as this is problematic at discontinuities according to experience, one is forced to find suitable combinations of the two. This task is dealt with by total variation diminishing (TVD) slope limiters. Higher-order reconstruction schemes employ polynomials of 2nd order or more for the interpolation. Having found \( \phi_i \) from such methods, one can take either \( \phi_i(x_{i\pm\frac{1}{2}}) \) directly (point-value formulation) or build the volume average of \( \phi_i \) over the staggered cells \( Z_{i\pm\frac{1}{2}} \) (volumetric formulation).

After reconstructing the conserved variables to the cell interfaces, we can compute the fluxes. In general, the flux at the interface \( x_{i+\frac{1}{2}} \) resulting from the reconstruction centered at \( Z_i \) will not be the same as the one resulting from the reconstruction centered at \( Z_{i+1} \), and one speaks of left and right flow states. Whenever a well-defined upwind direction can be identified, the obvious choice is to take the upwind flux, i.e. the one from the left cell if the velocity is directed to the right. However, this is not the case in general and one is left with a Riemann problem. For example, an Alfvén wave might propagate to the right while a sound wave travels to the left. Depending on the system (e.g. HD or MHD), an exact solution to this problem is fairly complicated, numerically expensive or even unknown. Therefore, approximate Riemann solvers are used to determine the “best” flux from the left and right states. If necessary, the accuracy of approximate solvers can be improved e.g. by the multi stage (MUSTA) method, which uses “virtual” time steps to find better solutions.

Having determined the interface fluxes, the solution can be advanced in time e.g. using a Runge–Kutta method. The timestep must small enough to satisfy the Courant–Friedrichs–Lewy (CFL) condition, which states that \( \Delta t \) must be smaller than the time needed for the fastest wave to cross the cell, at any cell in the whole grid.

As mentioned above, the induction equation requires special treatment. In contrast to Eq. (A.1), it has the form \( \partial_t \mathbf{B} + \nabla \times \mathbf{F} = 0 \) where \( \mathbf{F} \) is \( c \) times the electric field. It is important that the magnetic remains solenoidal at all times. This can be achieved via the technique of constrained transport, which relies on the magnetic field being defined at the center of the cell faces: \( B_x \) at \( (x_{i\frac{1}{2}}, y_j, z_k) \), \( B_y \) at \( (x_i, y_{j\frac{1}{2}}, z_k) \) and \( B_z \) at \( (x_i, y_j, z_{k\frac{1}{2}}) \) for the cell \( Z_{i,j,k} \). The electric field is reconstructed at the (center of the) cell’s edges, the various components being distributed such that each is parallel to its edge\(^1\). \( \partial_t \mathbf{B} \) can then calculated. How the initial magnetic field can be installed in a staggered grid in a way that guarantees \( \nabla \cdot \mathbf{B} = 0 \) is described in the next section.

\(^1\)Note that there are several possibilities. One may first reconstruct \( v \) to the cell face, compute the electric field \( \mathbf{E} \) and then reconstruct \( \mathbf{E} \) at the cell edges, or one may first reconstruct \( v \) and \( \mathbf{B} \) at the cell edges and compute \( \mathbf{E} \) there.
A.2 Imposing the Initial Magnetic Field

Setting up the initial magnetic field for a simulation is not trivial. Every grid cell must be solenoidal in the magnetic field as best as possible, ideally up to machine precision. A practical way to do this is by means of the corresponding vector potential (Evans & Hawley 1988). First of all, \( A \) is prescribed at the center of the grid cells’ edges. The components are distributed such that they point along the respective edge: \( A_x \) is defined at \( \left( i - \frac{1}{2}, j - \frac{1}{2}, k - \frac{1}{2} \right) \), \( A_y \) is defined at \( \left( i - \frac{1}{2}, j - \frac{1}{2}, k \right) \) and \( A_z \) is defined at \( \left( i - \frac{1}{2}, j + \frac{1}{2}, k - \frac{1}{2} \right) \) for all \( i, j \) and \( k \). The magnetic field components, which are defined at the center of the cell faces with normal orientation, can now be computed from the vector potential components of the surrounding edges. In Cartesian coordinates, we have simply

\[
B^i_{x} = \nabla \times A = \frac{\Delta y}{\Delta z} \left( A^i_{z} - A^i_{y} \right)
\]

and so on, with \( \Delta \) denoting the line elements (edge lengths). In general, one has to resort to the definition of the curl as circulation per unit area. For example, the \( x \) component of \( B \) follows from

\[
S^{i-\frac{1}{2}j,k} B^{i-\frac{1}{2}j,k} = \oint_{S} d\ell \cdot A
\]

with \( S \) and \( l \) denoting the face area and edge length, respectively. Following this procedure, the set magnetic field is automatically solenoidal. Special care has to be taken at coordinate singularities.

A.3 Boundaries Conditions

Experience has shown that the success of simulations stands or falls with the implementation of the boundary conditions, hence some remarks are in order. The standard boundary conditions are realized via ghost cells as follows. As an example, the \( x \)-left boundary shall be considered, with \( i = \frac{1}{2} \) being the index of the boundary interface.

Open (“Outflow”) The objective here is to allow material and waves to cross the boundary smoothly, without spurious forces at the boundary interface. This is done by setting \( U^{0,ij,k} = U^{L,ij,k} \), \( U^{-1,ij,k} = U^{R,ij,k} \) and so forth for the hydrodynamic quantities \( U = (\rho, p, v) \) and for the gravitational potential \( \Phi \).
Figure A.1: Results of the test described in Sect. A.3 with $B_y = 4$ ($\beta = 1/16$, $v_{A,y} \approx 4.4$) and $v_x = 0.3$ at the $y$-left boundary, of which the first ghost cell is included in the plot. In the left-hand case all quantities were held at their initial values in the boundary ghost cells, whereas in the right-hand case open conditions were used for $B$. The horizontal dotted lines show the imposed $v_x$ and the expected $B_x$ (multiplied with 10). The vertical dotted line is at $y = v_{A,y} t$.

The transverse magnetic field components, $B_y$ and $B_z$ in the given case, are treated likewise: $B_{y,j-1/2,k} = B_{y,j-1/2,k}^0$, $B_{z,j,k-1/2} = B_{z,j,k-1/2}^1$ etc. The normal magnetic field component is computed from the constraint that $\nabla \cdot B$ should vanish also in the ghost cells.

Reflective These are realized just as open boundaries, except that the signs of the normal velocity and the transverse magnetic field are reversed.

Periodic The values from the cells next to the opposing boundary are copied to the ghost cells: $U^{0,j,k} = U^{N_x,j,k}$, $U^{-1,j,k} = U^{N_x-1,j,k}$ etc.

It has proven itself beneficial to fix as few quantities as possible to avoid spurious oscillations in the solution. This can be demonstrated in 2D Cartesian simulations that are in some sense analogous to the conditions at the lower boundary of the jet simulations. The setup consists of a homogeneous magnetic field and a thereto orthogonal velocity field at the boundary. The initial conditions are: $B = B_y \hat{e}_y$, $\rho = \rho_0 = \text{const}$, $p = p_0 = \text{const}$ and $v_x = v_y = 0$ inside the computational domain. The boundary conditions are periodic in $x$ and open at the $y$-right. At the $y$-left, $v_x$ is fixed at a nonzero value and the other quantities are either held at their initial values or copied from the computational domain like in open boundaries. The differing results from the two approaches can be seen in Fig. A.1 (as usual, in normalized units). The expected value for the orthogonal field is $B_x = -v_x B_y / v_{A,y}$ as the perturbation propagates with Alfvén velocity. Obviously, the results are much better.
if open conditions are used for $B$. It also helps if $v_x$ is prescribed in the cells directly above the boundary. In all the tests with subsonic velocities, $\rho$, $p$ and $v_y$ do not show a significant departure from their initial values. $B_y$ remained constant in all tests.

It may still be necessary to fix some values, either in the ghost cells or next to the boundary in the domain, for the sake of numerical stability. The problems seem to correlate with the “harshness” of the conditions in many cases. The jet simulations presented above get along without such fixes as long as the imposed rotation velocity is moderate. Boundary effects can also be minimized by putting the boundaries as far away as possible from the region of interest, for instance with grid cells that grow exponentially in size. Unfortunately, this cannot be applied to the jet inlet boundary and it is of limited use in spherical grids.

In an attempt to mimic the jet front, one can also prescribe the post-shock state in the ghost cells. If the propagation velocity of a disturbance exceeds the fast magnetosonic speed, a fast shock may form. Fast shocks have the special property of refracting the magnetic field away from the shock normal. When a fast shock propagates in the direction of the magnetic field, it has the effect of kinking the magnetic field lines. That is, in a setup similar to the test simulation described above, it creates a normal field component $B_x$. Such a shock is called switch-on shock and exists only when the Alfvén speed exceeds the sound velocity in the unshocked plasma or, equivalently, if the fast magnetosonic velocity is the Alfvén speed. The numerical solver handles fast shocks well in Cartesian 2D simulations, see Fig. A.2, with jump conditions from Priest (1982).

Figure A.2: Switch-on shock with maximal deflection in the shock frame (left) and in the laboratory frame (right). In the shock frame, the shock rests at $y = 5$. The vertical line shows the theoretical position of the shock front in the laboratory frame.
Appendix B

Coordinate Transformations

In this work, several coordinate systems are used alternately: a Cartesian system \((x, y, z)\) as the basis, a traditional spherical coordinate system \((r, \theta, \phi)\) in which the \(z\)-axis is the polar axis, an alternative spherical coordinate system \((r, \theta, \varphi)\) in which the \(y\)-axis is the polar axis and the cylindrical coordinate systems \((R_z, \varphi, z)\) and \((R_y, \varphi, y)\) where \(R\) denotes the perpendicular distance to the respective axis. The \((r, \theta, \varphi)\) system is connected to the \((r, \theta, \varphi)\) system by the substitution \((x, y, z) \rightarrow (z, x, y)\).

The conversion formulae are

\[
\begin{align*}
x &= r \sin \theta \cos \phi = R_z \cos \phi = R_y \sin \varphi \\
y &= r \sin \theta \sin \phi = R_z \sin \phi = r \cos \theta \\
z &= r \cos \theta = z = r \sin \theta \cos \varphi = R_y \cos \varphi
\end{align*}
\]

To convert vector fields from the \((r, \theta, \varphi)\) to the \((r, \vartheta, \phi)\) system, one needs to know the versors \(\hat{e}_\vartheta\) and \(\hat{e}_\phi\) (\(\hat{e}_r\) is unchanged) in the \((r, \theta, \varphi)\) system, i.e. expressed as a linear combination of \(\hat{e}_r, \hat{e}_\theta\) and \(\hat{e}_\varphi\). The spherical versors \(\hat{e}_r, \hat{e}_\theta\) and \(\hat{e}_\varphi\), expressed in Cartesian coordinates (i.e. as a linear combination of \(\hat{e}_x, \hat{e}_y\) and \(\hat{e}_z\)), can be readily calculated from Eqs. (B.1) or looked up in textbooks such as Nolting (2002). From them, one can determine the Cartesian versors expressed in spherical coordinates by adding up the projections along the three coordinate directions:

\[
\begin{align*}
\hat{e}_x &= (\hat{e}_x \cdot \hat{e}_r) \hat{e}_r + (\hat{e}_x \cdot \hat{e}_\theta) \hat{e}_\theta + (\hat{e}_x \cdot \hat{e}_\varphi) \hat{e}_\varphi \\
&= \sin \theta \cos \phi \hat{e}_r + \cos \theta \cos \phi \hat{e}_\theta - \sin \phi \hat{e}_\varphi \\
\hat{e}_y &= \sin \theta \sin \phi \hat{e}_r + \cos \theta \sin \phi \hat{e}_\theta + \cos \phi \hat{e}_\varphi \\
\hat{e}_z &= \cos \theta \hat{e}_r - \sin \theta \hat{e}_\varphi \\
\end{align*}
\]

Having these, one can write the azimuthal versor

\[
\hat{e}_\varphi = \cos \varphi \hat{e}_x - \sin \varphi \hat{e}_z
\]

as a combination of \(\hat{e}_\theta\) and \(\hat{e}_\varphi\) (the \(\hat{e}_r\) component vanishes). \(\hat{e}_\theta\) can be found in a similar manner or, more easily, via the relation \(\hat{e}_\theta = \hat{e}_\varphi \times \hat{e}_r\). Finally,

\[
\hat{e}_{R_y} = \sin \varphi \hat{e}_x + \cos \varphi \hat{e}_z.
\]
Appendix C
Visualization

C.1 Volume Rendering

Volume rendering is one of the most practical ways to visualize a three-dimensional scalar field. The basic idea is to assign light emission and absorption properties, depending on the field(s) to be visualized, to every voxel in the volume. The contributions of all cells along a particular line-of-sight are then added up to give a 2D image. Volume rendering is a standard feature of most visualization software and the implementation is technically complex in general (arbitrary viewing angle, perspective view). However, a volume renderer for lines-of-sight parallel to one of the coordinate axes is easy to implement and the results are in some cases better than with standard software, see e.g. Fig. 3.2. The starting point for creating a volume renderer is the equation of radiative transfer,

\[
\frac{dI}{ds} = -\kappa I + \epsilon, \quad (C.1)
\]

whereby the light intensity along the line-of-sight is reduced by absorption, parameterized by the field \( \kappa \), and augmented by emission, parametrized by the field \( \epsilon \). If the total volume is divided by discrete cells in which \( \kappa \) and \( \epsilon \) are constant, then the solution is

\[
I(s_{i+\frac{1}{2}}) = I(s_{i-\frac{1}{2}})e^{-\kappa_i\Delta s_i} + \frac{\epsilon_i}{\kappa_i}\left(1 - e^{-\kappa_i\Delta s_i}\right), \quad (C.2)
\]

where \( \Delta s_i \) is the intersection length of the line-of-sight with the \( i \)-th cell. A Taylor expansion for small \( \Delta s_i \) yields

\[
I(s_{i+\frac{1}{2}}) = I(s_{i-\frac{1}{2}}) \cdot (1 - \kappa_i\Delta s_i) + \epsilon_i\Delta s_i. \quad (C.3)
\]

For practical purposes, it is important [and essential if Eq. (C.3) is used] that \( \kappa \Delta s \) is not larger than one in any cell, i.e. only the cell with the largest value should be optically dense. Therefore, the field which controls the absorption should be normalized to the range \([0, 1]\). Moreover, exponentiating the field controls the level of detail shown in the rendered image. It is possible and often useful to associate different fields with absorption and emission.
C.2 Numerical Computation of the Stream Function

Field lines (streamlines) in a 2D vector field can be found by determining the level sets of a scalar field $\Psi$, the stream function, whose derivatives give the vector field. This works for all solenoidal fields, such as the magnetic field or an incompressible fluid flow. If the 2D field comprises of the projection of a 3D field onto a 2D plane, then the field must be symmetric in plane’s normal direction. An application to this method is the plot in Fig. 3.9.

In Cartesian coordinates $(x, y, z)$, the field $u = u(x, y)\hat{e}_x + v(x, y)\hat{e}_y + w(x, y)\hat{e}_z$ is connected to the stream function $\Psi = \Psi(x, y)\hat{e}_z$ by $u = \nabla \times \Psi + w\hat{e}_z$. The level sets $d\Psi = 0$ of such a stream function coincide with the field lines of the parallel field in the $z = \text{const}$ plane:

$$0 = d\psi = \frac{\partial \psi}{\partial x} dx + \frac{\partial \psi}{\partial y} dy = -v dx + u dy \iff \frac{dx}{dy} = \frac{u}{v}. \quad (C.1)$$

Numerically, we seek a solution to $u = \partial_y \Psi$, $v = -\partial_x \Psi$. It is straightforward to show that the following two functions are independent solutions, provided that $\nabla \cdot u = 0$:

$$\Psi_1 = \int_{x_0}^{x} dx' \psi(x', y) - \int_{y_0}^{y} dy' \psi(x, y'), \quad (C.2a)$$

$$\Psi_2 = -\int_{x_0}^{x} dx' \psi(x', y) + \int_{y_0}^{y} dy' \psi(x_0, y') \quad (C.2b)$$

where $x_0$ and $y_0$ are constant coordinates, conveniently chosen to be at the border of the grid. The integrals can be found with a cumulative version of the composite rectangle or of the Simpson rule. Eventually, the two solutions may be combined to a single one: $\Psi = (\Psi_1 + \Psi_2)/2$.

In spherical coordinates $(r, \theta, \phi)$, the field $u = u(r, \theta)\hat{e}_r + v(r, \theta)\hat{e}_\theta + w(r, \theta)\hat{e}_\phi$ is connected to the stream function $\Psi(r, \theta)$ by $\nabla \Psi \times \hat{e}_\phi / R$ where $R = r \sin \theta$. Analogous to the Cartesian case, the level sets $d\Psi = 0$ coincide with the field lines of the parallel field in the $\phi = \text{const}$ plane. We need to solve $u = -\partial_\theta \Psi / (r R)$, $v = \partial_r \Psi / R$. The following two functions are independent solutions, provided that $\nabla \cdot u = 0$:

$$\Psi_1 = -\int_{\theta_0}^{\theta} d\theta' r^2 \sin \theta' u(r, \theta') + \int_{r_0}^{r} dr' \sin \theta_0 v(r', \theta_0), \quad (C.3a)$$

$$\Psi_2 = \int_{r_0}^{r} dr' r' \sin \theta v(r', \theta) - \int_{\theta_0}^{\theta} d\theta' r_0^2 \sin \theta' u(r_0, \theta'). \quad (C.3b)$$
The visualization method presented in this section proved itself to be a viable tool for visualizing vector fields in 2D slices. The main advantage of it over more simple methods like drawing arrows at equidistant positions is that it gives a good impression of the field lines’ course without the need to calculate them explicitly. The method is applied in Kirby et al. (1999) and in Laidlaw et al. (2005). However, the procedure is only outlined there. The following is a possible implementation. It was used to draw Fig. 4.7.

Consider the vector field \( u(x,y) \hat{e}_x + v(x,y) \hat{e}_y \). The local (at the centroid \( S \)) field strength and direction shall be encoded in the size and orientation of a filled isosceles triangle (wedge) like the one drawn in the adjacent figure. Taking \( A = \alpha \sqrt{u^2 + v^2} \) for the area, where \( \alpha \) is an appropriate scaling parameter, we get the triangle’s base \( c \) from \( A = \beta c^2 / 2 \), where the parameter \( \beta := \tilde{h} / c \) determines the sharpness of the wedge. The height of the triangle is \( h = \beta c = \tilde{h} + \bar{h} = h/3 + 2h/3 \) and the tilt angle is \( \varphi = \arctan(v/u) \). From this, we can calculate the points \( Q, C, P \) and \( O \):

\[
\begin{align*}
  x_Q &= x_S + \bar{h} \cos \varphi, \\
  y_Q &= y_S + \bar{h} \sin \varphi, \\
  x_C &= x_S - \bar{h} \cos \varphi, \\
  y_C &= y_S - \bar{h} \sin \varphi, \\
  x_{P,O} &= x_C \pm \frac{c}{2} \sin \varphi, \\
  y_{P,O} &= y_C \mp \frac{c}{2} \cos \varphi.
\end{align*}
\]

The isosceles trapezoid spanned by the points \( M, N, L \) and \( R \) marks the region where there must be no overlap with another wedge. It is determined by the parameter \( \eta \), from which \( b = c + 2\eta c \) and \( a = c \bar{h} / h + 2\eta c \). The positions of the points are

\[
\begin{align*}
  x_{M,N} &= x_S \pm \frac{a}{2} \sin \varphi, \\
  y_{M,N} &= y_S \pm \frac{a}{2} \cos \varphi, \\
  x_{R,L} &= x_C \pm \frac{b}{2} \sin \varphi, \\
  y_{R,L} &= y_C \mp \frac{b}{2} \cos \varphi.
\end{align*}
\]

The procedure is now as follows:

1. Randomly select a point \((x, y)\).

2. Calculate the coordinates \((O, P, M, N, \ldots)\) of a possible new wedge.
3. Determine whether the trapezoid \((L, R, N, M)\) of the new wedge would overlap with existing ones. If it doesn’t, draw it and go back to step 1. If it does, discard it and continue with step 4.

4. Determine whether a preassigned maximum number of consecutive fails has been reached. If not, go back to step 1. If yes, stop.

Viable (starting) values for the parameters are \(\alpha = 100/\text{median field value}\) pixels, \(\beta = 4\), \(\eta = 0.7\) and 250 for the maximum number of consecutive fails.