Distributed Implicit Discontinuous Galerkin MHD Solver

Lukas Korous, Jan Skala, Jan Kotek, Pavel Karban

University of West Bohemia - Faculty of Electrical Engineering, Czech Republic
Regional Innovation Centre for Electrical Engineering, Department of Theory of Electrical Engineering
Astronomical Institute of Czech Academy of Sciences, Ondřejov, Czech Republic
E-mail: korous@kte.zcu.cz, jan.skala@asu.cas.cz, jankotek@email.cz, karban@kte.zcu.cz

ABSTRACT

The work aims at solving the nonlinear ideal MHD equations using the Discontinuous Galerkin (DG) method. The nonlinear equations will be solved implicitly, and the problem of differentiating non-differentiable numerical fluxes (such as HLLD), will be taken care of by constructing the jacobian using numerical differentiation from the residual and using damped Newton's method. Overshoots and undershoots present in the vicinity of physical shock waves will be accounted for adopting a post-processing step implementing the vertex-based limiter.

This work is being implemented using the FE libraries deal.II and Trilinos in 3D and fully parallel/distributed manner, and once finished will be available at a public software repository.

There are several phenomena in the universe that we can look at as magnetohydrodynamic in nature - planets consisting of metals, interplanetary space, stars. As for the stars, these phenomena include spots, solar flares, solar winds, space weather. To study these phenomena, it is important that we are able to model them at a reasonable scale, in a reasonable detail, but most importantly - model them in a physically correct way. This means that on the path from our physical / mathematical model to numerical solutions, our algorithms should not spoil the solution by introducing non-physical oscillations, be in conflict with the model (having div $\mathbf{B} = 0$), add artificial diffusion, etc.

FORMULATION OF THE PROBLEM

Ideal MHD equations in the conservative form read:

$$\frac{\partial \boldsymbol{\Psi}}{\partial t} + \nabla \cdot \mathbf{F}(\boldsymbol{\Psi}) = \mathbf{S}, \qquad (1)$$

where $\boldsymbol{\Psi}$ is the *state vector*, $\mathbf{F}_i$, $i = 1, 2, 3$ are the *fluxes*, and $\mathbf{S}$ is the *source term*:

$$\boldsymbol{\Psi} = (\rho, \pi_1, \pi_2, \pi_3, U, B_1, B_2, B_3), \qquad (2)$$

$$\mathbf{F}_i = \begin{pmatrix} \pi_i \\ \frac{\pi_1 \pi_i}{\rho} - B_1 B_i + \frac{1}{2}\delta_{1i}(p + U_m) \\ \frac{\pi_2 \pi_i}{\rho} - B_1 B_i + \frac{1}{2}\delta_{2i}(p + U_m) \\ \frac{\pi_3 \pi_i}{\rho} - B_1 B_i + \frac{1}{2}\delta_{3i}(p + U_m) \\ \frac{\pi_i}{\rho}\left(\frac{\gamma}{\gamma-1}p + U_k\right) + \frac{2}{\rho}\varepsilon_{ijk}(\pi_k B_i - \pi_i B_k)B_k \\ \frac{\pi_i B_1 - \pi_1 B_i}{\rho} \\ \frac{\pi_i B_2 - \pi_2 B_i}{\rho} \\ \frac{\pi_i B_3 - \pi_3 B_i}{\rho} \end{pmatrix}, \qquad (3)$$

$$\mathbf{S} = (0, \rho g_1, \rho g_2, \rho g_3, \boldsymbol{\pi} \cdot \boldsymbol{g}, 0, 0, 0). \qquad (4)$$

Here, ρ is the plasma density, π_1, π_2, π_3 are the $x-, y-,$ and $z-$ momentum components respectively. U is the total energy, B_1, B_2, B_3 are the three components of magnetic field. Remaining quantities are p - plasma pressure, U_k - kinetic energy, U_m - magnetic energy, and $\boldsymbol{g}$ gravitational acceleration with components g_1, g_2, g_3.

DG FORMULATION

DG formulation of the resulting space-discretized problem reads

$$\int_{\Omega_t} \frac{\partial \boldsymbol{\Psi}_h}{\partial t} \mathbf{v}_h \quad - \quad \sum_{K_i \in T_h} \int_{K_i} \mathbf{F}(\boldsymbol{\Psi}_h)(\nabla \cdot \mathbf{v}_h) \qquad (5)$$
$$+ \quad \sum_{\Gamma_{ij} \in \Gamma_I} \int_{\Gamma_{ij}} \mathbf{H}(\boldsymbol{\Psi}_h|_{ij}, \boldsymbol{\Psi}_h|_{ji}, \boldsymbol{n}_{ij}) \mathbf{v}_h$$
$$+ \quad \sum_{\Gamma_i \in \Gamma_B} \int_{\Gamma_i} \mathbf{H}\left(\boldsymbol{\Psi}_h|_i, \overline{\boldsymbol{\Psi}_h|_i}, \boldsymbol{n}_i\right) \mathbf{v}_h$$
$$= \quad \int_{\Omega_t} \mathbf{S} \mathbf{v}_h,$$

where $\mathbf{v}_h$ is a test function, Γ_I is a set of all internal interfaces in the mesh, and $\Gamma_{ij} \in \Gamma_I$ an interface between two elements - K_i and K_j. Similarly Γ_B is a set of all boundary interfaces in the mesh, and $\Gamma_i \in \Gamma_B$ an interface on the boundary that neighbors the element K_i. Meaning of $\overline{\boldsymbol{\Psi}_h|_i}$ depends on the boundary conditions. $\mathbf{H}(\boldsymbol{\Psi}_h|_{ij}, \boldsymbol{\Psi}_h|_{ji}, \boldsymbol{n}_{ij})$ is the *numerical flux* between states $\boldsymbol{\Psi}_h|_{ij}$ and $\boldsymbol{\Psi}_h|_{ji}$ in the direction of $\boldsymbol{n}_{ij}$.

SOLVING THE NONLINEAR PROBLEM

We would like to use the Newton's method to solve the nonlinear problem arising from discretizing (5) in time using Crank-Nicolson scheme. Damped Newton's method (with damping factor α) performs iterations

$$\mathbf{J}(\mathbf{x}_{k+1}^n)(\Delta \mathbf{x}_{k+1}^n) = -\mathbf{R}(\mathbf{x}_{k+1}^n) \qquad (6)$$
$$\mathbf{x}_{k+1}^{n+1} = \mathbf{x}_{k+1}^n + \alpha \Delta \mathbf{x}_{k+1}^n,$$

for $n = 0, ...,$ where $\mathbf{R}(\mathbf{x}_{k+1}^n)$ is the *residual*, $\mathbf{J}(\mathbf{x}_{k+1}^n) = \frac{d\mathbf{R}(\mathbf{x}_{k+1}^n)}{d\mathbf{x}_{k+1}^n}$ the *jacobian*, $\mathbf{x}_{k+1}^0 = \mathbf{x}_k$ and $\mathbf{x}_k$ is the DG solution vector from the k-th time step. For the residual, we have

$$\mathbf{R}(\mathbf{x})_i = \int_{\Omega_t} \frac{\boldsymbol{\Psi}_h}{\Delta t} + \sum_{K_i \in T_h} \int_{K_i} \mathbf{F}(\boldsymbol{\Psi}_h)(\nabla \cdot \mathbf{v}_{hi}) \qquad (7)$$
$$- \sum_{\Gamma_{ij} \in \Gamma_I} \int_{\Gamma_{ij}} \mathbf{H}(\boldsymbol{\Psi}_h|_{ij}, \boldsymbol{\Psi}_h|_{ji}, \boldsymbol{n}_{ij}) \mathbf{v}_{hi}$$
$$- \sum_{\Gamma_i \in \Gamma_B} \int_{\Gamma_i} \mathbf{H}\left(\boldsymbol{\Psi}_h|_i, \overline{\boldsymbol{\Psi}_h|_i}, \boldsymbol{n}_i\right) \mathbf{v}_{hi} + \int_{\Omega_t} \mathbf{S} \mathbf{v}_{hi},$$

where $\mathbf{v}_{hi}$ is the i-th test function, and $\boldsymbol{\Psi}_h$ is the (global) function corresponding to $\mathbf{x}$.

The jacobian $\mathbf{J}(\mathbf{x}_{k+1}^n) = \frac{d\mathbf{R}(\mathbf{x}_{k+1}^n)}{d\mathbf{x}_{k+1}^n}$ is calculated numerically (numerical differentiation implementation from Trilinos package Sacado is used).
The iterations in (6) are performed until $||\mathbf{R}(\mathbf{x}_{k+1}^n)||$ is lower than a prescribed threshold.

HANDLING NONZERO DIVERGENCE

An infamous problem in numerically solving the MHD equations, which is present regardless of the presence of the continuity of the sought solution, is satisfying *Gauss's law*, i.e. the relationship

$$\text{div } \mathbf{B} = 0, \qquad (8)$$

to tackle this, an exactly divergence free FE space is a mathematically clean and fully reliable method how to satisfy the relation 8, as opposed to methods of *divergence cleaning*, or *Constrained-Transport (CT)*. In order to create a generic solver, it is a very favorable method.

ATTRIBUTES OF DG SOLUTION

There are several topics to be dealt with when solving MHD equations numerically using the DG method. First one is the selection of numerical flux $\mathbf{H}(\boldsymbol{\Psi}_h|_{ij}, \boldsymbol{\Psi}_h|_{ji}, \boldsymbol{n}_{ij})$ that should be as little diffusive as possible, and should satisfy the relations $\mathbf{H}(\boldsymbol{\Psi}_h, \boldsymbol{\Psi}_h, \boldsymbol{n}) = \mathbf{F}(\boldsymbol{\Psi}_h) \cdot \boldsymbol{n}$ and $\mathbf{H}(\boldsymbol{\Psi}_1, \boldsymbol{\Psi}_2, \boldsymbol{n}) = \mathbf{H}(\boldsymbol{\Psi}_2, \boldsymbol{\Psi}_1, -\boldsymbol{n})$. For the ideal MHD we chose the HLLD flux.

Another aspect of the solution, which occurs both in continuous and discontinuous FE simulations of the MHD system are spurious (nonphysical) oscillations appearing in the solution near discontinuities or sharp fronts. Here, in DG, the problem is much smaller than in the continuous case, as the instabilities occur only in regions adjacent to the sharp fronts.

Our goal is to build a software package that should give a reasonable and physically correct solution to all sorts of problems, a suitable method to handle discontinuities must not change the physics (as is the case in e.g. artificial diffusion) and must not require fine-tuning parameters for it to work. Therefore we chose a parameter-less method of post-processing nature (which does not change the equations being solved). Such a method is the *Vertex-based limiter* developed by D. Kuzmin, and successfully applied to DG for advection-diffusion problems in existing work.

NUMERICAL EXAMPLE

We present results from one benchmark - MHD Blast - designed at *www.astro.princeton.edu/ js-tone/Athena/tests/blast/blast.html*.
The figures below are performed with a standard set of basis functions for DG - i.e. not with the exactly divergence free basis we want to use - that is a work-in-progress currently and it shall only stabilize and qualitatively improve the results.
The following figures show the density, magnitude of momentum, magnitude of magnetic field, and the pressure in the region $-0.5 \leq x \leq 0.5; -0.5 \leq y \leq 0.5$.
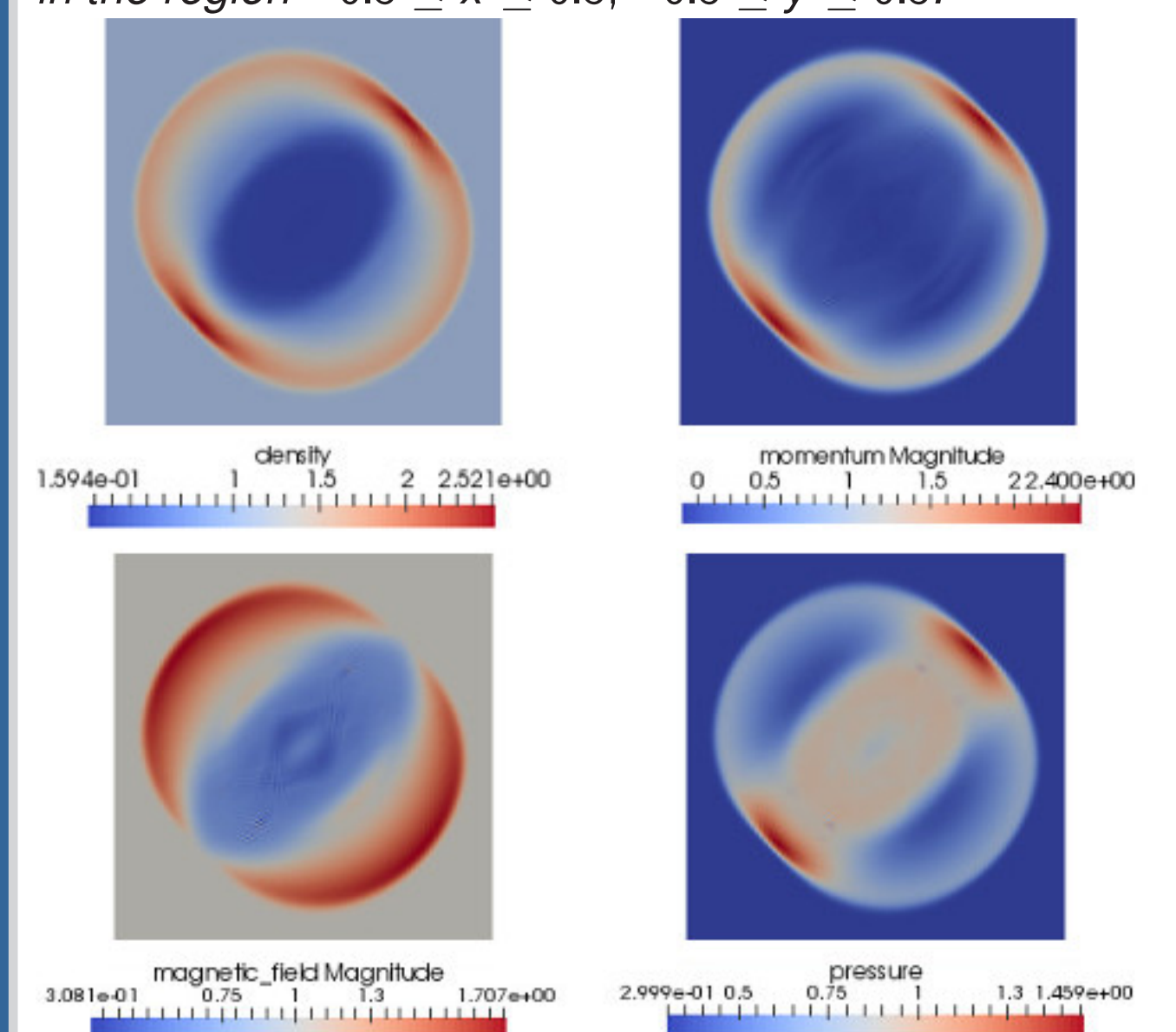

Figure 1: MHD Blast benchmark results

CONCLUSION

The work is in progress, but already it is observable that the DG approach yields qualitatively good results, and due to the implementation of nonlinear solver, and parallel abilities of the used library deal.ii, the code is very efficient. Currently implemented feature is the mentioned vertex based limiter.

ACKNOWLEDGMENT