Implementing a discontinuous Galerkin method for the compressible, inviscid Euler equations in the DUNE framework

Juan Pablo Gallego[1,*], **Johannes Löbbert**[1,**], **Steffen Müthing**[2,***], **Peter Bastian**[2,†], **Christian Klingenberg**[1,‡], and **Yinhua Xia**[1,3,§]

[1] Institut für Mathematik, Universität Würzburg
[2] Interdisziplinäres Institut für wissenschaftliches Rechnen, Universität Heidelberg
[3] Dept. of Mathematics, University of Science and Technology of China, Hefei

A discontinuous Galerkin scheme was implemented in the DUNE framework to solve the compressible, inviscid Euler equations in a multi-dimensional Cartesian grid. It uses a HLLC Riemann solver for the numerical fluxes in the interfaces, a total variation bounded limiter to handle discontinuities, a positivity preserving limiter for near vacuum conditions, and adaptive mesh refinement (AMR).

Copyright line will be provided by the publisher

1 Discontinuous Galerkin method scheme

The spatial discretization of the Euler equations of inviscid compressible flow is achieved by using a cartesian grid in one, two or three space dimensions, and by choosing a set of piecewise polynomials to be used as test and trial functions. Then the weak formulation of the system of conservation law is given by:

$$\int_{\tau_k} \mathbf{u}_t v d\mathbf{x} - \int_{\tau_k} \mathbf{f}(\mathbf{u}) \cdot \nabla_{\mathbf{x}} v d\mathbf{x} + \int_{\partial\tau_k} \mathbf{f}(\mathbf{u}) \cdot \vec{\mathbf{n}} v d\Gamma = 0 \tag{1}$$

Since the solution is not continuous on the interfaces between the cells, the surface integral in the scheme is calculated by solving a Riemann problem to "communicate" between the cells. For more details see [1], [2], [4], [5], [6].

2 Limiters

In order to control spurious oscillation in the numerical solution, due to possible discontinuities, a Total Variation Bounded (TVB) limiter has been implemented based on the minmod function. When the linear component of the approximation is greater than the limit criteria $M \cdot h^2$ (where M is a parameter to be chosen) the solution in the cell is reduced to a first order polynomial. The slope $u^m_{x_i}$ of this polynomial in the $i-$direction is calculated comparing $\hat{u}_{x_i}$ (which is the difference between the mean values of the conserved variable along the interfaces $x_{i-1/2}$ and $x_{i+1/2}$) and the forward and backward differences (defined as $\Delta_+ u = \bar{u}_{x_{i+1}} - \bar{u}_{x_i}$ and $\Delta_- u = \bar{u}_{x_i} - \bar{u}_{x_{i-1}}$), where $\bar{u}_{x_i}$ is the mean value in the cell I_i (see Fig. 1).

$$u^m_{x_i} = \text{minmod}\left(\hat{u}_{x_i}, \Delta_+ u, \Delta_- u\right) \tag{2}$$

Then the limited solution will be computed by multiplying the scalar functions ϕ_i to the linear components in each direction, and adding them together with the average value $\bar{u}_{x_i}$ of the conserved variable in the specific cell.

$$u_h = \bar{u}_{x_i} + \sum u^m_{x_i} \cdot \phi_i \quad . \tag{3}$$

Additionally a positivity preserving limiter has been implemented in order to take account of near-vacuum conditions. Its implementation, which is well known for scalar equations, needed to be extended to our system of equations. Special care has to been taken for the values of both density and pressure, thus the mean values of these quantities ($\bar{\rho}$ and $\bar{P}$) are controlled after each stage of the time integration is computed. If $\bar{\rho}$ or $\bar{P}$ are below a certain $\epsilon-$value in any cell, the whole time step must be restarted and the CFL condition is reduced by a factor of two. If it is not the case, then the minimum value of the density ρ_{min} is verified. In case ρ_{min} is below ϵ, it must be limited (see Fig. 2, and [3] for more information) and the energy is recomputed with the data of momentum and density. Then the minimum value of the pressure P_{min} is verified. If it is below ϵ, it is fixed to ϵ and then the energy is recalculated out of the values of density, pressure and momentum. The positivity preserving limiting procedure does not affect the momentum.

* e-mail juan.gallego@mathematik.uni-wuerzburg.de, phone +49 (0)931 318 3415 fax +49 (0)931 318 3494
** e-mail otto@loebbi.net
*** e-mail steffen.muething@iwr.uni-heidelberg.de, phone +49 6221 54 8888, fax +49 6221 54 8884
† e-mail peter.bastian@iwr.uni-heidelberg.de, phone +49 6221 54 8261, fax +49 6221 54 8884
‡ e-mail klingenberg@mathematik.uni-wuerzburg.de, phone +49 (0)931 318 5045 fax +49 (0)931 318 3494
§ e-mail yhxia@ustc.edu.cn

Copyright line will be provided by the publisher

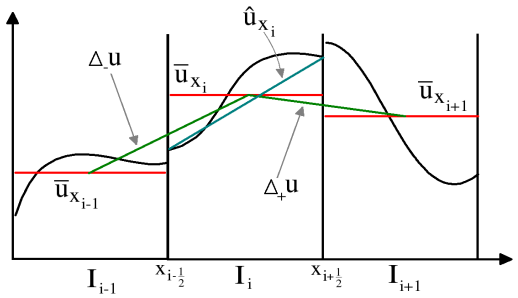

Fig. 1: Slope comparison for the TVB limiter in each dimension

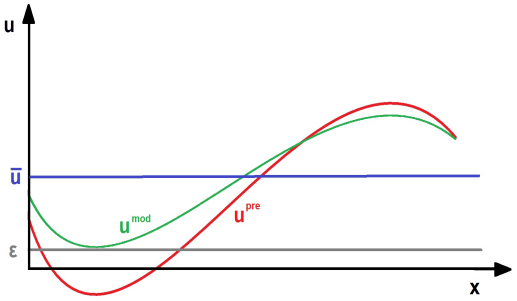

Fig. 2: Positivity preserving limiter, the solution is shrank around the mean value so the minimum is fixed to ϵ

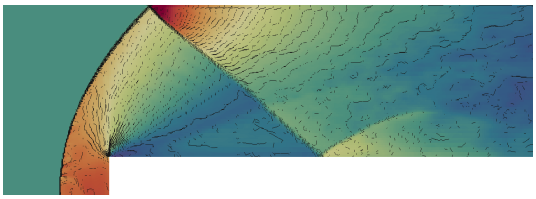

Fig. 3: Forward facing step problem with contour lines of density after 1.5 seconds for Mach 3.

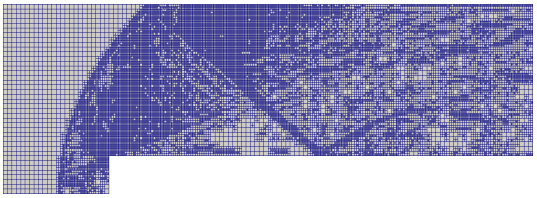

Fig. 4: Forward facing step problem mesh refinement from a maximum $\Delta x = 1/40$ to a minimum $\Delta x = 1/120$.

3 DUNE framework

The Distributed and Unified Numerics Environment (DUNE) is a C++ toolbox for solving PDE with grid based methods. It is divided in modules depending on their functionality, such as a grid managing (dune-grid), linear algebra operations (dune-istl) and function space discretization (dune-localfunctions). Our code is based on the the external module dune-pdelab. It provides a general scheme in which the basic functionality of DUNE is combined in order to both, generate implementations in a faster way, and provide additional helpful features, like for example the automatic looping on the different entities of the grid, parallelization, load balancing. The grid was implemented using ALUGrid external module, which allows the use of adaptive mesh refinement.

4 Implementation features

The code uses a nodal Lagrangian basis (see [6]), where the expansion coefficients are exactly the point-wise values of the functions in the nodes of the element. These values are used to calculate the flux functions of the conservation laws in each node. The numerical fluxes are computed through a HLLC Riemann solver. A multi-stage strong stability preserving scheme (Shu's third order SSP scheme, see [3]) was implemented for the time integration. The TVB and the positivity preserving limiters are used as post-processing procedures which modify the solution after each stage of the time integration. The codes includes adaptive mesh refinement triggered by the limiting criteria. In Fig. 3 we show a simulation of supersonic flow past a forward facing step, see [7], using our code. Fig. 4 shows the performance of the AMR implemented.

References

[1] B. Cockburn and C. W. Shu, J. Comput. Phys. **84**, 90 - 113 (1989).
[2] B. Cockburn and C. W. Shu, J. Comput. Phys. **141**, 199 - 224 (1998).
[3] X. Zhang, and C. W. Shu, J. Comput. Phys. **229**, 8918 - 8934 (2010).
[4] R. J. LeVeque, Numerical methods for conservation laws (Birkhäuser Verlag, Berlin, 1999).
[5] P. D. Lax, Hyperbolic systems of conservation laws and the mathematical theory of shock waves (SIAM, Philadelphia, 1973).
[6] J. S. Hesthaven and T. Warburton, Nodal discontinuous Galerkin methods, algorithms, analysis and applications (Springer, New York, 2008).
[7] P. Woodward and P. Colella, J. Comput. Phys. **54**, 115 - 173 (1984).

Copyright line will be provided by the publisher