IPP-Report

Max-Planck-Institut für Plasmaphysik

Nicolay J. Hammer, Roman Hatzky, HLST Core Team

Combining Runge-Kutta discontinous Galerkin methods with Various limiting methods

IPP 5/124
Oktober, 2010

Combining Runge-Kutta discontinuous Galerkin methods with various limiting methods

Nicolay J. Hammer *
Roman Hatzky †

HLST Core Team
Max-Planck-Institut für Plasmaphysik

March 25, 2010

*tel.: +49 (0)89 3299 3529 , email: nicolay.hammer@ipp.mpg.de
†tel.: +49 (0)89 3299 1707 , email: roman.hatzky@ipp.mpg.de

Contents

1 Introduction

The aims of this report are to summarise the basics of the discontinuous Galerkin finite element method (DG-FEM) and to provide building blocks, e.g. time integration schemes or limiting methods, to deal with associated challenges.

The basic idea of the Galerkin finite element method is to discretise the differential equation by so-called finite elements which got their name from their finite support.

In contrast to the continuous Galerkin finite element method (CG-FEM) the finite elements of the DG-FEM method are discontinuous and not continuous at their boundaries. As a consequence numerical fluxes evolve over the finite element boundaries which lead to additional terms in the numerical discretisation of the scheme. Therefore, DG-FEM methods have a hybrid character which places them in between finite volume schemes on the one hand and finite element schemes on the other hand.

The advantages are that DG-FEM methods can be applied to complicated geometries and boundary conditions while they are closely linked to the control volume approach of finite volume methods.

2 The linear Advection Equation in 1D

In this report we will discuss the spatial discretisation of a partial differential equation (PDE) using a DG-FEM. As an elementary example of a PDE we will use the linear Advection Equation in one spatial dimension.

The advection equation in one spatial dimension reads, when neglecting source terms, as

$$\frac{\partial}{\partial t} f(x,t) + a \frac{\partial}{\partial x} f(x,t) = 0 \qquad . \tag{1}$$

Here t denotes the time, x the spatial coordinate and a the advection velocity.

3 Using Legendre polynomials as base functions

A suitable choice for the base functions used for our discontinuous Galerkin finite element method (DG-FEM) are the well known Legendre polynomials. They have a lot of convenient characteristics which we will describe briefly in this section.

First of all, they can be calculated in an easy and stable way by using the zeroth and the first Legendre polynomial

$$P_0(x) = 1 \quad \text{and} \quad P_1(x) = x \tag{2}$$

and the following recursion formula

$$P_n(x) = \frac{2n-1}{n} P_{n-1}(x) - \frac{n-1}{n} P_{n-2}(x) \qquad , \tag{3}$$

where $(n = 2, 3, ...)$ and $x \in [-1, +1]$.

If we take a look at the Hilbert space $V := L^2([-1, +1], \mathbb{R})$ we see that the Legendre polynomials are forming a complete orthogonal system

$$\int_{-1}^{+1} P_n(x) P_m(x) \, \mathrm{d}x = \frac{2}{2n-1} \delta_{nm} \tag{4}$$

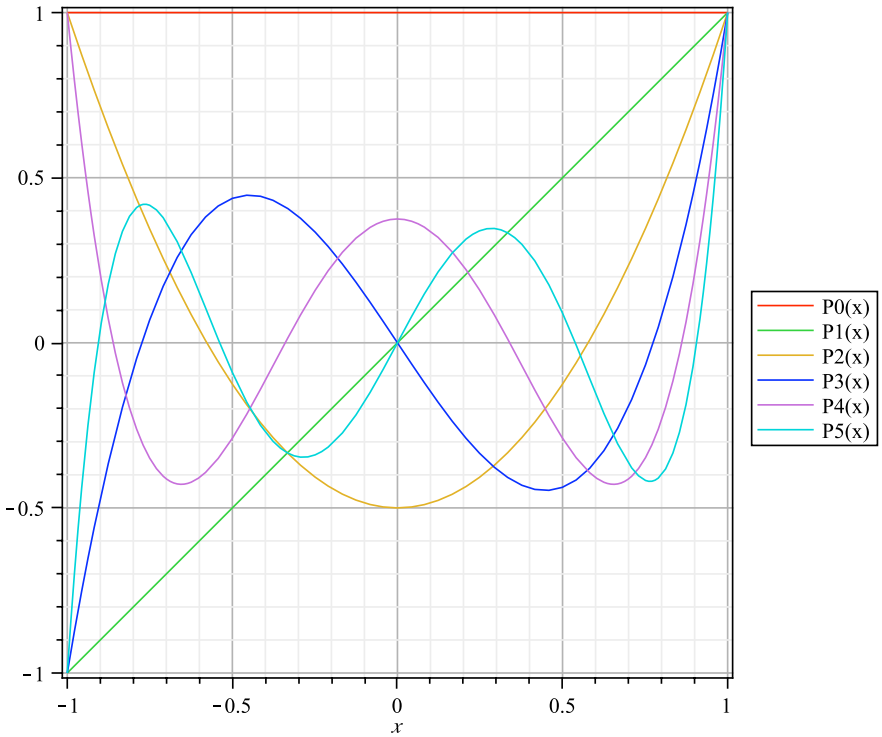

Figure 1: Plot showing the Legendre polynomials $P_n(x)$ for $(n = 0, ..., 5)$.

where the completeness tells us that any sufficiently smooth function $f \in V$ can be expanded in a series using Legendre polynomials

$$f(x) = \sum_{m=0}^{\infty} u_m P_m(x) \qquad . \tag{5}$$

Then the expansion coefficients can be derived in the following way:

$$u_n = \frac{2n-1}{2} \int_{-1}^{+1} f(x) P_n(x) \, \mathrm{d}x \qquad . \tag{6}$$

At last we define the Vandermonde matrix

$$\mathcal{V}_{nm} = P_m(x_n) \tag{7}$$

which will be useful for calculating Legendre expansion coefficients for discrete problems.

Note that, discrete Legendre polynomials are represented on inter-nodal points, i.e. grid points inside each finite element. The higher the order of the used polynomials is, the more of these inter-nodal points have to be used. This results in maximal CFL numbers for explicit schemes which are significantly smaller than 1.0 (see e.g. Figure 1).

4 Discretising space using a discontinuous Galerkin method with Legendre polynomials

4.1 Discretising the spatial dimension using the "strong" form given by Warburton and Hesthaven

The way to derive the "strong" form of Eq. (1) summarised here is described in Ref. [1], p. 20 ff. Note that quantities denoted by i are residing on the finite element i which is located on the coordinate interval $[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}]$ (compare Figure 2).

We start with the residual

$$\xi_i(x,t) = \frac{\partial f_i}{\partial t} + a\frac{\partial f_i}{\partial x} \tag{8}$$

which shall be orthogonal to every test function $\nu(x) \in V^p(x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}})$:

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \nu(x)\,\xi_i(x,t)\,\mathrm{d}x = \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \nu\left(\frac{\partial f_i}{\partial t} + a\frac{\partial f_i}{\partial x}\right)\,\mathrm{d}x = 0 \qquad . \tag{9}$$

Performing an integration by parts on the second term gives us

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \left(\nu(x)\frac{\partial f_i(x)}{\partial t} - a\frac{\partial \nu(x)}{\partial x} f_i(x)\right)\,\mathrm{d}x = -\Big[\nu(x) a f_i(x)\Big]_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \qquad . \tag{10}$$

If we have a closer look at the interfaces we see that the solution there is defined by the left and the right hand side state. From a physical point of view we can define the right hand side by introducing a numerical flux $\{af\}_i^*(x)$. Then Eq. (10) reads

$$\begin{aligned} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \left(\nu(x)\frac{\partial f_i(x)}{\partial t} \right. &- \left. a\frac{\partial \nu(x)}{\partial x} f_i(x)\right)\,\mathrm{d}x \\ &= -\Big[\nu(x)\{af\}_i^*(x)\Big]_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \qquad . \end{aligned} \tag{11}$$

Performing the integration by parts on the second term, once more, we end up with

$$\begin{aligned} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \left(\nu(x)\frac{\partial f_i(x)}{\partial t} \right. &+ \left. a\frac{\partial f_i(x)}{\partial x}\nu(x)\right)\,\mathrm{d}x \\ &= \Big[\nu(x)\Big(a f_i(x) - \{af\}_i^*(x)\Big)\Big]_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \end{aligned} \tag{12}$$

which is the "strong" form of Eq. (1) after Ref. [1], p. 22. Applying the integration limits to the right hand side term we get

$$\begin{aligned} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \left(\nu(x)\frac{\partial f_i(x)}{\partial t} \right. &+ \left. a\frac{\partial f_i(x)}{\partial x}\nu(x)\right)\,\mathrm{d}x \\ &= \nu(x_{i+\frac{1}{2}})\Big(a f_i(x_{i+\frac{1}{2}}) - \{af\}_i^*(x_{i+\frac{1}{2}})\Big) \\ &- \nu(x_{i-\frac{1}{2}})\Big(a f_i(x_{i-\frac{1}{2}}) - \{af\}_i^*(x_{i-\frac{1}{2}})\Big) \qquad . \end{aligned} \tag{13}$$

Finally, we have to specify the numerical flux $\{af\}_i^*(x)$. In our case we will use upwind fluxes, i.e. information is propagating only in down stream direction, which are defined as

$$\{af\}_i^*(x_{i+\frac{1}{2}}) = \begin{cases} af_i(x_{i+\frac{1}{2}}) & a \geq 0 \\ af_{i+1}(x_{i+\frac{1}{2}}) & a < 0 \end{cases} \quad . \tag{14}$$

This is the flux at the right element interface, the upwind flux at the left element interface can be derived by decrementing the index i by one.

From now on, we will refer to the case of a positive advection velocity ($a \geq 0$), hence the first right hand side term in Eq. (13) cancels and we are left with

$$\begin{aligned} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \left(\nu(x) \frac{\partial f_i(x)}{\partial t} \right. &+ \left. a \frac{\partial f_i(x)}{\partial x} \nu(x) \right) \, \mathrm{d}x \\ &= \nu(x_{i-\frac{1}{2}}) a \left(f_{i-1}(x_{i-\frac{1}{2}}) - f_i(x_{i-\frac{1}{2}}) \right) \quad . \end{aligned} \tag{15}$$

Now, using Eq. (6), we have to find $u_n \in V^p(x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}})$ such that $\forall \nu \in V^p(x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}})$ the following equation is fulfilled

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \nu \left(\frac{\partial u_i}{\partial t} + a \frac{\partial u_i}{\partial x} \right) \, \mathrm{d}x = \nu(x_{i-\frac{1}{2}}) a \left(u_{i-1}(x_{i-\frac{1}{2}}) - u_i(x_{i-\frac{1}{2}}) \right) \tag{16}$$

which is Eq. (15) expressed in Legendre coefficients. Here p denotes the maximum polynomial order used for this scheme. Using the coordinate transformation

$$\tilde{x} = \frac{2x - (x_{i+\frac{1}{2}} + x_{i-\frac{1}{2}})}{(x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}})} \quad , \quad \frac{\mathrm{d}\tilde{x}}{\mathrm{d}x} = \frac{2}{(x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}})} \tag{17}$$

we can write Eq. (16) as

$$\begin{aligned} & \sum_{m=0}^{p} \left\{ \left(\int_{-1}^{+1} P_n(\tilde{x}) P_m(\tilde{x}) \frac{\mathrm{d}x}{\mathrm{d}\tilde{x}} \, \mathrm{d}\tilde{x} \right) \frac{\mathrm{d}\vec{u}_{i,m}}{\mathrm{d}t} \right\} \\ + \quad & a \sum_{m=0}^{p} \left\{ \left(\int_{-1}^{+1} P_n(\tilde{x}) \frac{\mathrm{d}}{\mathrm{d}x} P_m(\tilde{x}) \frac{\mathrm{d}x}{\mathrm{d}\tilde{x}} \, \mathrm{d}\tilde{x} \right) \vec{u}_{i,m} \right\} \\ = \quad & a P_n(-1) \sum_{m=0}^{p} P_m(1) u_{i-1,m} - a P_n(-1) \sum_{m=0}^{p} P_m(-1) u_{i,m} \quad . \end{aligned} \tag{18}$$

We introduce the definition of the Mass matrix $\mathcal{M}_{nm}$ and the Stiffness matrix $\mathcal{D}_{nm}$

$$\mathcal{M}_{nm} := \frac{\mathrm{d}x}{\mathrm{d}\tilde{x}} \int_{-1}^{+1} P_n(\tilde{x}) P_m(\tilde{x}) \, \mathrm{d}\tilde{x} \tag{19}$$

$$\tilde{\mathcal{D}}_{nm} := \frac{\mathrm{d}x}{\mathrm{d}\tilde{x}} \int_{-1}^{+1} P_n(\tilde{x}) \frac{\mathrm{d}}{\mathrm{d}x} P_m(\tilde{x}) \, \mathrm{d}\tilde{x} \quad , \tag{20}$$

as well as the definitions of the flux matrices

$$\tilde{\mathcal{F}}_{nm}^{+} := P_n(-1) P_m(1) \tag{21}$$

$$\tilde{\mathcal{F}}_{nm}^{-} := P_n(-1) P_m(-1) \quad . \tag{22}$$

Then we can write the flux terms as

$$a\,\tilde{\mathcal{F}}^{+}\vec{u}_{i-1} = aP_n(-1)\sum_{m=0}^{p} P_m(1)u_{i-1,m} \tag{23}$$

$$a\,\tilde{\mathcal{F}}^{-}\vec{u}_{i} = aP_n(-1)\sum_{m=0}^{p} P_m(-1)u_{i,m} \tag{24}$$

Now we have to compute the Mass matrix $\mathcal{M}_{nm}$, the Stiffness matrix $\tilde{\mathcal{D}}_{nm}$ and the matrices for the flux terms $\tilde{\mathcal{F}}^{+}_{nm}$ and $\tilde{\mathcal{F}}^{-}_{nm}$. We will see that this can be done quite easily, using a few adjuvant mathematical relations.

First of all, to compute the Mass matrix $\mathcal{M}_{nm}$ [Eq. (19)] we use Eq. (4). We simply get

$$\begin{aligned}\mathcal{M}_{nm} &= \frac{(x_{i+1}-x_i)}{2}\frac{2}{2n-1}\delta_{nm} \\ &= \frac{\Delta_+ x_i}{2}\frac{2}{2n-1}\mathcal{I}_{nm}\end{aligned} \tag{25}$$

where $\Delta_+ x_i$ denotes the difference operator $\Delta_+ x_i := x_{i+1} - x_i$.

To see how to compute the Stiffness matrix $\tilde{\mathcal{D}}_{nm}$ we have to go a bit more into details. If we expand Eq. (5) up to order p ($m = 0, .., p$) and derive it, we get

$$\frac{\mathrm{d}f_i(x)}{\mathrm{d}x} = \sum_{m=0}^{p} u^{(1)}_{i,m}P_m(x) \qquad , \tag{26}$$

where $u^{(1)}_m$ is given by

$$u^{(1)}_{i,m} = (2m+1)\sum_{n=0}^{p}[(n+m) \bmod 2]\,u_{i,n} \qquad . \tag{27}$$

This can be written in matrix notation

$$\vec{u}^{(1)}_i = \hat{\tilde{\mathcal{D}}}\vec{u}_i \qquad , \tag{28}$$

by defining the coefficient differentiation matrix (CDM)

$$\hat{\tilde{\mathcal{D}}}_{nm} = \begin{cases} (2m+1) & \text{if } m > n \text{ and } n+m \text{ odd} \\ 0 & \text{otherwise} \end{cases} \qquad . \tag{29}$$

Therewith, we can calculate the Stiffness matrix in a rather simple way. Using Eqs. (19) and (20) and a change of the differentiation variable, we get the following expression for the Stiffness matrix

$$\begin{aligned}\tilde{\mathcal{D}}_{nm} &= \int_{-1}^{+1} P_n(\tilde{x})\frac{\mathrm{d}}{\mathrm{d}\tilde{x}}P_m(\tilde{x})\,\mathrm{d}\tilde{x} \\ &= \sum_{l=0}^{p}\mathcal{M}_{nl}\hat{\tilde{\mathcal{D}}}_{lm} \\ &= \frac{2}{2n+1}\hat{\tilde{\mathcal{D}}}_{lm} \qquad .\end{aligned} \tag{30}$$

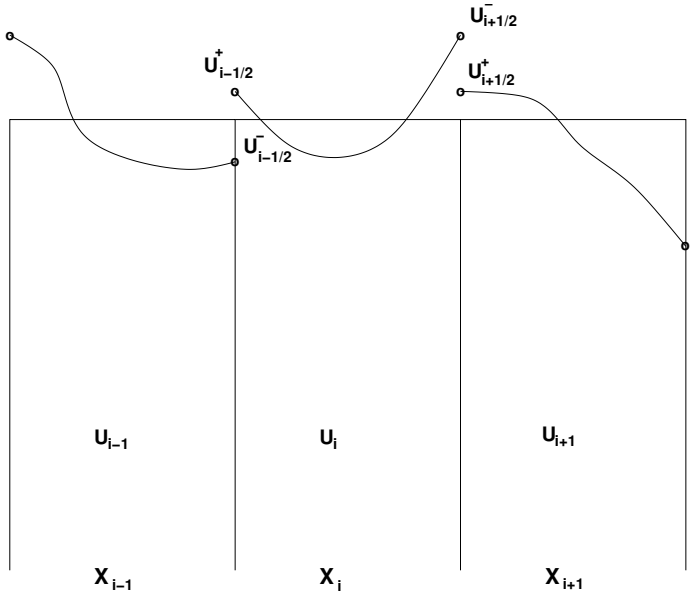

Figure 2: Sketch showing the left and right hand side of both of the left and right element interface, respectively.

At last we may use a property of Legendre polynomials, namely

$$P_n(\pm 1) = (\pm 1)^n \qquad . \tag{31}$$

Therewith, the matrices in Eqs. (21) and (22) read as

$$\tilde{\mathcal{F}}^+_{nm} = (-1)^n \tag{32}$$
$$\tilde{\mathcal{F}}^-_{nm} = (-1)^{n+m} \tag{33}$$

Finally, using Eqs. (23) and (24), we can write Eq. (16) in the compact form

$$\boxed{\mathcal{M}\frac{\mathrm{d}\vec{u}_i}{\mathrm{d}t} = -a\left[\tilde{\mathcal{D}}\vec{u}_i - \left(\tilde{\mathcal{F}}^+\vec{u}_{i-1} - \tilde{\mathcal{F}}^-\vec{u}_i\right)\right]} \qquad . \tag{34}$$

4.2 Discretising the spatial dimension using the weak form

We start with Eq. (10) which is the weak form of Eq. (1). Inserting the integration limits on the right hand side we can see that $[\nu(x_{i-\frac{1}{2}})af_i(x_{i-\frac{1}{2}})]$ and $[\nu(x_{i+\frac{1}{2}})af_i(x_{i+\frac{1}{2}})]$ denote the flux terms through the left and right boundary of the finite element, respectively. The flux depends on both the left and right hand side state of the element interface (Figure 2). To allow for this we introduce the flux functions

$$\hat{h}_{i-\frac{1}{2}}(f^-, f^+) := \hat{h}(f^-(x_{i-\frac{1}{2}}), f^+(x_{i-\frac{1}{2}})) \tag{35}$$
$$\hat{h}_{i+\frac{1}{2}}(f^-, f^+) := \hat{h}(f^-(x_{i+\frac{1}{2}}), f^+(x_{i+\frac{1}{2}})) \qquad . \tag{36}$$

Therewith, Eq. (10) reads

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \left(\nu(x) \frac{\partial f_i(x)}{\partial t} - a \frac{\partial \nu(x)}{\partial x} f_i(x) \right) \mathrm{d}x = - \left(\nu(x_{i+\frac{1}{2}}) \hat{h}_{i+\frac{1}{2}}(f^-, f^+) - \nu(x_{i-\frac{1}{2}}) \hat{h}_{i-\frac{1}{2}}(f^-, f^+) \right) . \tag{37}$$

In the most simplest case, the upwind case, i.e. information is propagating only in down stream direction, the flux function reads

$$\hat{h}_{i-\frac{1}{2}}(f^-, f^+) = \begin{cases} a\, f_{i-1}^-(x_{i-\frac{1}{2}}) & a \geq 0 \\ a\, f_i^+ \ (x_{i-\frac{1}{2}}) & a < 0 \end{cases} \tag{38}$$

$$\hat{h}_{i+\frac{1}{2}}(f^-, f^+) = \begin{cases} a\, f_i^- \ (x_{i+\frac{1}{2}}) & a \geq 0 \\ a\, f_{i+1}^+(x_{i+\frac{1}{2}}) & a < 0 \end{cases} . \tag{39}$$

However, to simplify matters, from now on we will refer to the upwind case corresponding to $a \geq 0$.

Now, using again the coordinate transformation [Eq. (17)], the expansion of function $f_i(x)$ in Legendre polynomials [Eq. (5)], and choosing the test function to be an element of the polynomial Hilbert space $\nu(x) \in V^p(x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}})$, we get

$$\begin{aligned} & \sum_{m=0}^{p} \left\{ \left(\int_{-1}^{+1} P_n(\tilde{x}) P_m(\tilde{x}) \frac{\mathrm{d}x}{\mathrm{d}\tilde{x}} \, \mathrm{d}\tilde{x} \right) \frac{\mathrm{d}\vec{u}_{i,m}}{\mathrm{d}t} \right\} \\ - \ & a \sum_{m=0}^{p} \left\{ \left(\int_{-1}^{+1} \frac{\mathrm{d}}{\mathrm{d}x} P_n(\tilde{x}) P_m(\tilde{x}) \frac{\mathrm{d}x}{\mathrm{d}\tilde{x}} \, \mathrm{d}\tilde{x} \right) \vec{u}_{i,m} \right\} \\ = \ & - \left(a P_n(1) \sum_{m=0}^{p} P_m(1) u_{i,m} - a P_n(-1) \sum_{m=0}^{p} P_m(1) u_{i-1,m} \right) . \end{aligned} \tag{40}$$

We introduce the definition of the Mass matrix $\mathcal{M}_{nm}$ and the Stiffness matrix $\mathcal{D}_{nm}$:

$$\mathcal{M}_{nm} = \frac{\mathrm{d}x}{\mathrm{d}\tilde{x}} \int_{-1}^{+1} P_n(\tilde{x}) P_m(\tilde{x}) \, \mathrm{d}\tilde{x} \tag{41}$$

$$\mathcal{D}_{nm} = \frac{\mathrm{d}x}{\mathrm{d}\tilde{x}} \int_{-1}^{+1} \frac{\mathrm{d}}{\mathrm{d}x} P_n(\tilde{x}) P_m(\tilde{x}) \, \mathrm{d}\tilde{x} \tag{42}$$

To calculate the Mass and the Stiffness matrix as well as the flux matrices we may use the similarities to the scheme from Ref. [1] (see Subsection 4.1). One can easily see, that both schemes have the same Mass matrix and we can use the result of Eq. (25).

In case of the Stiffness matrix we use the results of Eqs. (29) and (30). However, from Eq. (42) it can be seen that the partial derivative was shifted to the test function's base polynomial and thus, the CDM is the transposed matrix of the CDM in Warburton's case. It has the following form

$$\hat{\mathcal{D}}_{nm} = \begin{cases} (2m+1) & \text{if } m < n \text{ and } n+m \text{ odd} \\ 0 & \text{otherwise} \end{cases} . \tag{43}$$

Using this CDM we get the following Stiffness matrix

$$\mathcal{D}_{nm} = \frac{2}{2m+1}\hat{\mathcal{D}}_{nm} \quad . \tag{44}$$

Last but not least, we can define the flux matrices similarly to the Eqs. (32) and (33)

$$\mathcal{F}^{+}_{nm} = (-1)^n \tag{45}$$
$$\mathcal{F}^{-}_{nm} = 1 \quad . \tag{46}$$

Finally, using Eqs. (40) – (46) we can write the scheme in short notation [compare Eq. (34)]

$$\boxed{\mathcal{M}\frac{\mathrm{d}\vec{u}_i}{\mathrm{d}t} = a\left[\mathcal{D}\vec{u}_i + \left(\mathcal{F}^{+}\vec{u}_{i-1} - \mathcal{F}^{-}\vec{u}_i\right)\right]} \quad . \tag{47}$$

5 Explicit 3rd order TVD-Runge-Kutta scheme

Explicit Runge-Kutta (ERK) time integration schemes preserve the total variation diminishing (TVD) character of a spatial discretisation method, if they are convex with respect to their intermediate steps and are at least of order $p+1$ with respect to the order of the spatial discretisation method.

The TVD-Runge-Kutta scheme of third order in Ref. [7] as given in Ref. [8], looks the following way

$$\vec{u}^{\,(1)} = \vec{u}^{\,n} + \Delta t\, L(\vec{u}^{\,n}) \tag{48}$$
$$\vec{u}^{\,(2)} = \frac{3}{4}\vec{u}^{\,n} + \frac{1}{4}\vec{u}^{\,(1)} + \frac{1}{4}\Delta t\, L(\vec{u}^{\,(1)}) \tag{49}$$
$$\vec{u}^{\,n+1} = \frac{1}{3}\vec{u}^{\,n} + \frac{2}{3}\vec{u}^{\,(2)} + \frac{2}{3}\Delta t\, L(\vec{u}^{\,(2)}) \quad . \tag{50}$$

In the case of a simple Eulerian forward time discretisation for the ODE given in Eq. (47), we get

$$\vec{u}_i^{\,(1)} = \vec{u}_i^{\,n} + \Delta t\mathcal{M}^{-1}a\left[\mathcal{D}\vec{u}_i^{\,n} + \left(\mathcal{F}^{+}\vec{u}_{i-1}^{\,n} - \mathcal{F}^{-}\vec{u}_i^{\,n}\right)\right] \tag{51}$$
$$\vec{u}_i^{\,(2)} = \frac{3}{4}\vec{u}_i^{\,n} + \frac{1}{4}\vec{u}_i^{\,(1)} + \frac{1}{4}\Delta t\mathcal{M}^{-1}a\left[\mathcal{D}\vec{u}_i^{\,(1)} + \left(\mathcal{F}^{+}\vec{u}_{i-1}^{\,(1)} - \mathcal{F}^{-}\vec{u}_i^{\,(1)}\right)\right] \tag{52}$$
$$\vec{u}_i^{\,n+1} = \frac{1}{3}\vec{u}_i^{\,n} + \frac{2}{3}\vec{u}_i^{\,(2)} + \frac{2}{3}\Delta t\mathcal{M}^{-1}a\left[\mathcal{D}\vec{u}_i^{\,(2)} + \left(\mathcal{F}^{+}\vec{u}_{i-1}^{\,(2)} - \mathcal{F}^{-}\vec{u}_i^{\,(2)}\right)\right] \tag{53}$$

6 Solving the equation system using an implicit time discretisation

We start with Eq. (47) which is an ODE in time. Using an implicit (Eulerian backward) time discretisation and generalising the result for arbitrary upwind directions, we get

$$\mathcal{M}\left(\frac{\vec{u}_i^{\,n+1} - \vec{u}_i^{\,n}}{\Delta t}\right) = a\mathcal{D}\vec{u}_i^{\,n+1} + \frac{a+|a|}{2}\left(\mathcal{F}^{r+}\vec{u}_{i-1}^{\,n+1} - \mathcal{F}^{r-}\vec{u}_i^{\,n+1}\right) \tag{54}$$
$$+ \frac{a-|a|}{2}\left(\mathcal{F}^{l+}\vec{u}_i^{\,n+1} - \mathcal{F}^{l-}\vec{u}_{i+1}^{\,n+1}\right) \quad . \tag{55}$$

Sorting all terms depending on time step $n+1$ to the left hand side and all terms depending on time step n to the right hand side gives *

$$\vec{u}_i^{n+1} - \Delta t \mathcal{M}^{-1} \Big[a\mathcal{D}\vec{u}_i^{n+1} + \frac{a+|a|}{2}\left(\mathcal{F}^{r+}\vec{u}_{i-1}^{n+1} - \mathcal{F}^{r-}\vec{u}_i^{n+1}\right) \tag{56}$$

$$+ \frac{a-|a|}{2}\left(\mathcal{F}^{l+}\vec{u}_i^{n+1} - \mathcal{F}^{l-}\vec{u}_{i+1}^{n+1}\right) \Big] = \vec{u}_i^n \tag{57}$$

Eq. (56) can be written in matrix form as follows

$$(\mathcal{I} - \mathcal{Q})\,\vec{u}^{n+1} = \vec{u}^n \tag{58}$$

where the matrix $\mathcal{Q}$ of dimension $I \times J$ is defined as

$$\mathcal{Q} = \begin{pmatrix} \mathcal{B} & \mathcal{C} & 0 & & \cdots & & \mathcal{A} \\ \mathcal{A} & \mathcal{B} & \mathcal{C} & 0 & & \cdots & 0 \\ 0 & \mathcal{A} & \mathcal{B} & \mathcal{C} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \mathcal{A} & \mathcal{B} & \mathcal{C} & 0 \\ 0 & & \cdots & 0 & \mathcal{A} & \mathcal{B} & \mathcal{C} \\ \mathcal{C} & 0 & & \cdots & 0 & \mathcal{A} & \mathcal{B} \end{pmatrix} \tag{59}$$

with the following block matrix elements

$$\mathcal{A}_{nm} = \frac{a+|a|}{2}\Delta t \sum_{l=0}^{p} \mathcal{M}_{nl}^{-1}\mathcal{F}_{lm}^{r+} \tag{60}$$

$$\mathcal{B}_{nm} = \Delta t \sum_{l=0}^{p} \mathcal{M}_{nl}^{-1}\left(a\mathcal{D}_{lm} + \frac{a-|a|}{2}\mathcal{F}_{lm}^{l+} - \frac{a+|a|}{2}\mathcal{F}_{lm}^{r-}\right) \tag{61}$$

$$\mathcal{C}_{nm} = -\frac{a-|a|}{2}\Delta t \sum_{l=0}^{p} \mathcal{M}_{nl}^{-1}\mathcal{F}_{lm}^{l-} \quad . \tag{62}$$

Here $I, J = (p+1)\cdot N_x$ where p and N_x denote the order of polynomial approximation and the number of grid points in spatial x direction, respectively. We also use the definition of the mass matrix $\mathcal{M}_{nm}$ [Eq. (41)], the stiffness matrix $\mathcal{D}_{nm}$ [Eq. (42)]. The flux matrices $\mathcal{F}_{nm}^{r+}$ and $\mathcal{F}_{nm}^{r-}$ are defined as

$$\mathcal{F}_{nm}^{r+} = (-1)^n \tag{63}$$

$$\mathcal{F}_{nm}^{r-} = 1 \tag{64}$$

$$\mathcal{F}_{nm}^{l+} = (-1)^{n+m} \tag{65}$$

$$\mathcal{F}_{nm}^{l-} = (-1)^m \quad . \tag{66}$$

Note that, the first and the last row of $\mathcal{Q}$ indicates periodic boundary conditions.

Moreover, we may then generalise Eq. (59) towards a generalised Crank–Nicholson (CR) time discretisation

$$[\mathcal{I} - \theta\mathcal{Q}]\,\vec{u}^{n+1} = [\mathcal{I} + (1-\theta)\mathcal{Q}]\,\vec{u}^n \quad . \tag{67}$$

*Compare the explicit time discretisation [Eqs. (51) – (53)].

The generalised CR is a centred time differencing scheme, hence it is second order accurate. Note that Eq. (67) includes three "special" cases. First in case of $\theta = 0$ a fully explicit scheme. Secondly, a fully implicit scheme for $\theta = 1$. And finally, if $\theta = 1/2$ then Eq. (67) resembles the "classical" CR time discretisation scheme.

Finally, Eqs. (58) and (67) can be solved by inverting the left hand side matrix using a sparse matrix solver.

7 Implicit Runge-Kutta schemes

7.1 Multi-stage Gauss-Legendre type scheme of order $2s$

In this section a multi-stage implicit Runge-Kutta (IRK) scheme of Gauss-Legendre type will be given (see e.g. Refs. [9],[10]). Generally, implicit Runge-Kutta schemes of s stages are $2s$ order accurate, therefore the four stage IRK given below is 8th order accurate.

In general notation the s-stage implicit Runge-Kutta scheme reads

$$\vec{u}^{(1)} = f\left(t^n + c_1\Delta t, \vec{u}^n + \sum_{k=1}^{s} a_{1k}\Delta t\vec{u}^{(k)}\right) \tag{68}$$

$$\vec{u}^{(2)} = f\left(t^n + c_2\Delta t, \vec{u}^n + \sum_{k=1}^{s} a_{2k}\Delta t\vec{u}^{(k)}\right) \tag{69}$$

$$\vdots$$

$$\vec{u}^{(s)} = f\left(t^n + c_s\Delta t, \vec{u}^n + \sum_{k=1}^{s} a_{sk}\Delta t\vec{u}^{(k)}\right) \tag{70}$$

$$\vec{u}^{n+1} = f\left(t^n + \Delta t, \vec{u}^n + \sum_{k=1}^{s} a_{(s+1)k}\Delta t\vec{u}^{(k)}\right) \tag{71}$$

Eqs. (68) – (71) can be written as

$$\vec{u}^{(1)} = \vec{u}^n + \sum_{k=1}^{s} a_{1k}\Delta t L(\vec{u}^{(k)}) \tag{72}$$

$$\vec{u}^{(2)} = \vec{u}^n + \sum_{k=1}^{s} a_{2k}\Delta t L(\vec{u}^{(k)}) \tag{73}$$

$$\vdots$$

$$\vec{u}^{(s)} = \vec{u}^n + \sum_{k=1}^{s} a_{sk}\Delta t L(\vec{u}^{(k)}) \tag{74}$$

$$\vec{u}^{n+1} = \vec{u}^n + \sum_{k=1}^{s} a_{(s+1)k}\Delta t L(\vec{u}^{(k)}) \tag{75}$$

where $L(\vec{u})$ denotes the usual differential operator [compare Eqs. (48) – (50)].

The coefficients of the Runge-Kutta scheme can be given in the Butcher's

array, as follows

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \cdots & a_{1s} \\ c_2 & a_{21} & a_{22} & \cdots & a_{2s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & a_{s2} & \cdots & a_{ss} \\ \hline & a_{t1} & a_{t2} & \cdots & a_{ts} \end{array} \quad . \tag{76}$$

Note that here t denotes $t = s + 1$.

Generally, for a nonlinear differential operator L, the coupled Runge-Kutta equation system given in Eqs. (72) – (75) is solved by making use of iterative methods. We tried a fixed point iteration (e.g. see `http://en.wikipedia.org/wiki/Fixed_point_iteration`). However, in the linear case discussed in this report, the Runge-Kutta equation system can be solved directly by matrix inversion as well (Subsection 7.1).

All coefficients listed below can be found in Refs. [9], [10]. First of all, we will give the Butcher's array of the four stage Gauss-Legendre IRK of eighth order (i.e. $s = 4$)

$$\begin{array}{c|cccc} \frac{1}{2} - \omega_2 & \omega_1 & \omega_1' - \omega_3 + \omega_4' & \omega_1' - \omega_3 - \omega_4' & \omega_1 - \omega_5 \\ \frac{1}{2} - \omega_2' & \omega_1 - \omega_3' + \omega_4 & \omega_1' & \omega_1' - \omega_5' & \omega_1 - \omega_3' - \omega_4 \\ \frac{1}{2} + \omega_2' & \omega_1 + \omega_3' + \omega_4 & \omega_1' + \omega_5' & \omega_1' & \omega_1 + \omega_3' - \omega_4 \\ \frac{1}{2} + \omega_2 & \omega_1 + \omega_5 & \omega_1' + \omega_3 + \omega_4' & \omega_1' + \omega_3 - \omega_4' & \omega_1 \\ \hline & 2\,\omega_1 & 2\,\omega_1' & 2\,\omega_1' & 2\,\omega_1 \end{array} \tag{77}$$

where the omegas are defined as follows

$$\begin{aligned} \omega_1 &= \frac{18 - \sqrt{30}}{144} & , \quad \omega_1' &= \frac{18 + \sqrt{30}}{144} \\ \omega_2 &= \frac{1}{2}\left(\frac{15 + 2\sqrt{30}}{35}\right)^{\frac{1}{2}} & , \quad \omega_2' &= \frac{1}{2}\left(\frac{15 - 2\sqrt{30}}{35}\right)^{\frac{1}{2}} \\ \omega_3 &= \omega_2\left(\frac{4 + \sqrt{30}}{24}\right) & , \quad \omega_3' &= \omega_2'\left(\frac{4 - \sqrt{30}}{24}\right) \\ \omega_4 &= \omega_2\left(\frac{8 + 5\sqrt{30}}{168}\right) & , \quad \omega_4' &= \omega_2'\left(\frac{8 - 5\sqrt{30}}{168}\right) \\ \omega_5 &= \omega_2 - 2\,\omega_3 & , \quad \omega_5' &= \omega_2' - 2\,\omega_3' \end{aligned} \tag{78}$$

The Butcher's array of the three stage Gauss-Legendre IRK of sixth order (i.e. $s = 3$) is determined by

$$\begin{array}{c|ccc} \frac{1}{2} - \frac{\sqrt{15}}{10} & \frac{5}{36} & \frac{2}{9} - \frac{\sqrt{15}}{15} & \frac{5}{36} - \frac{\sqrt{15}}{30} \\ \frac{1}{2} & \frac{5}{36} + \frac{\sqrt{15}}{24} & \frac{2}{9} & \frac{5}{36} - \frac{\sqrt{15}}{24} \\ \frac{1}{2} + \frac{\sqrt{15}}{10} & \frac{5}{36} + \frac{\sqrt{15}}{30} & \frac{2}{9} + \frac{\sqrt{15}}{15} & \frac{5}{36} \\ \hline & \frac{5}{18} & \frac{4}{9} & \frac{5}{18} \end{array} \quad , \tag{79}$$

whereas those of the two stage, fourth order (i.e. $s = 2$) and one stage, second order (i.e. $s = 1$) IRK are determined by

$$\begin{array}{c|cc} \frac{1}{2}-\frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4}-\frac{\sqrt{3}}{6} \\ \frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{4}+\frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \quad , \qquad (80) \qquad \begin{array}{c|c} \frac{1}{2} & \frac{1}{2} \\ \hline & 1 \end{array} \quad , \qquad (81)$$

respectively.

7.2 Multi-stage Radau type scheme of order $(2s-1)$

In this section a multi-stage implicit Runge-Kutta scheme of Radau type will be given (see e.g. Ref. [10]). Implicit Runge-Kutta schemes of this type with s stages are $(2s-1)$ order accurate, therefore the three stage IRK given here is 5th order accurate.

The Radau type scheme is identical to the Gauss-Legendre type scheme [Eq. (75) – Eq. (75)], however, it uses differently chosen coefficients. Therefore, in this sub-section we will give only the Butcher's arrays. All coefficients listed below can be found in Ref. [10].

The Butcher's array of the three stage IRK of fifth order (i.e. $s=3$) is determined by

$$\begin{array}{c|ccc} \frac{4-\sqrt{6}}{10} & \frac{88-7\sqrt{6}}{360} & \frac{296-169\sqrt{6}}{1800} & \frac{-2+3\sqrt{6}}{225} \\ \frac{4+\sqrt{6}}{10} & \frac{296+169\sqrt{6}}{1800} & \frac{88+7\sqrt{6}}{360} & \frac{-2-3\sqrt{6}}{225} \\ 1 & \frac{16-\sqrt{6}}{36} & \frac{16+\sqrt{6}}{36} & \frac{1}{9} \\ \hline & \frac{16-\sqrt{6}}{36} & \frac{16+\sqrt{6}}{36} & \frac{1}{9} \end{array} \quad , \qquad (82)$$

and the two stage, third order (i.e. $s=2$) is given by

$$\begin{array}{c|cc} \frac{1}{3} & \frac{5}{12} & -\frac{1}{12} \\ 1 & \frac{3}{4} & \frac{1}{4} \\ \hline & \frac{3}{4} & \frac{1}{4} \end{array} \quad . \qquad (83)$$

7.3 Multi-stage SDIRK scheme of 4th order

Another family of multi-stage implicit Runge-Kutta schemes which is known to be strongly stable and monotonic, are the singly-diagonally-implicit Runge-Kutta (SDIRK) schemes (see e.g. Ref. [11]).

The SDIRK scheme is identical to the Gauss-Legendre type scheme [Eq. (75) – Eq. (75)], however, it uses differently chosen coefficients which are identical to zero above the diagonal of its Butcher's array. Below, we will give only the Butcher's arrays which can be found in Ref. [11]. Note that, because the coefficients of this method cannot be expressed analytically, the time coefficients in the first column are not given.

The Butcher's array of a four stage SDIRK of fourth order ($s=4$) is determined by

0.097961082941				
0.262318069183	0.097961082941			
0.230169419019	0.294466719347	0.097961082941		
0.210562684389	0.269382888280	0.307008634881	0.097961082941	
0.222119403264	0.282060762166	0.236881213175	0.258938621395	

. (84)

7.4 Solving the IRK equation system by matrix inversion

In case of a linear differential operator L, Eqs. (72) – (75) can be solved directly by inverting the matrix of the equation system.

Applying the general implicit Runge-Kutta methods [Eqs. (72) – (75)] to the implicit time discretisation of linear advection equation given in Eq. (58) leads to the equation system

$$[\mathcal{I} - \mathcal{R}]\,\vec{u}^{\,\mathrm{LHS}} = \vec{u}^{\,\mathrm{RHS}} \qquad . \tag{85}$$

which can be expressed using the block matrix

$$\mathcal{R} = \begin{pmatrix} a_{11}\mathcal{Q} & a_{12}\mathcal{Q} & \cdots & a_{1s}\mathcal{Q} & 0 \\ a_{21}\mathcal{Q} & a_{22}\mathcal{Q} & \cdots & a_{2s}\mathcal{Q} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{s1}\mathcal{Q} & a_{s2}\mathcal{Q} & \cdots & a_{ss}\mathcal{Q} & 0 \\ a_{t1}\mathcal{Q} & a_{t2}\mathcal{Q} & \cdots & a_{ts}\mathcal{Q} & 0 \end{pmatrix} \qquad . \tag{86}$$

Here, the matrix blocks of $\mathcal{R}$ are defined as product of the Runge-Kutta coefficients [Eq. (76)] and the matrix of the implicit time discretisation scheme [Eq. (59)].

The right hand side state vector is assembled by $s+1$ times the coefficient vector $\vec{u}^{\,n}$ at time step n

$$\vec{u}^{\,\mathrm{RHS}} = (\,\vec{u}^{\,n}, \vec{u}^{\,n}, \cdots, \vec{u}^{\,n}, \vec{u}^{\,n}\,)^{\mathrm{T}} \tag{87}$$

and the left hand side state vector by the coefficient vectors of all intermediate Runge-Kutta integration steps $1, \cdots, s$ and the coefficient vector $\vec{u}^{\,n+1}$ at new time step $n+1$

$$\vec{u}^{\,\mathrm{LHS}} = \left(\,\vec{u}^{(1)}, \vec{u}^{(2)}, \cdots, \vec{u}^{(s)}, \vec{u}^{\,n+1}\,\right)^{\mathrm{T}} \qquad . \tag{88}$$

Eq. (85) can be solved by inverting the left hand side matrix $[\mathcal{I} - \mathcal{R}]$. Then the coefficient vector $\vec{u}^{\,n+1}$ at new time step $n+1$ can be directly derived from the left hand side state vector

$$\vec{u}^{\,\mathrm{LHS}} = [\mathcal{I} - \mathcal{R}]^{-1}\,\vec{u}^{\,\mathrm{RHS}} \qquad . \tag{89}$$

7.5 Efficiency of implicit Runge Kutta schemes

The implicit Runge-Kutta schemes which are listed in Subsects. 7.1 – 7.3 have undergone an extensive testing in combination with specific options of the DG-FEM-IMPLICIT solver.

The various IRK schemes turned out, to be not unconditionally stable when the solution was obtained using an iterative method. In contrast to that, some of the IRK schemes, namely the fully implicit Gauss-Legendre and Radau IRK schemes, are stable with very large time steps when the solution was obtained directly. Note that, fully implicit denotes, that every equation depends on every other equation. Moreover, with increasing number of stages of the IRK scheme and increasing time step, the number of iteration steps for the fixed point iteration is increasing drastically. Furthermore, even though solving the IRK equation system directly is more efficient than using an iterative method, these schemes are still numerically inefficient, compared to an explicit time discretisation scheme.

Nevertheless, all of them showed growing spurious oscillations with increasing time step. It turned out, that this phenomena can be controlled to a certain extent, using the moment limiter methods listed above. However, with increasing time step the artefacts grow stronger and eventually the limiting method fails to suppress the wiggles.

These facts set an effective upper limit to the time step when using higher order implicit methods, depending on the order and type of IRK in combination with the order of the DG-FEM scheme used. Generally, the higher the order of the IRK scheme compared to the order of the DG-FEM scheme, the larger the size of the time step that can be reached without developing spurious artefacts when using a moment limiter. This can be seen in Figs. 3 and 4. Furthermore, the higher the order p of the DG-FEM scheme, the better the artefacts seem to be controlled by the moment limiter.

The observation described above raises the question about the numerical efficiency. Even though, some of the higher order implicit schemes tested, were able to produce equivalent accurate results compared to higher order explicit schemes, they were numerically much more expensive (up to a factor of ten), independent whether the IRK equation system was solved directly or using an iterative method.

From the performance point of view the SDIRK schemes yield the most promising results. Because this method is singly implicit only a small number of iterations, i.e. typically around two, have to be performed to solve the Runge-Kutta equation system. Here singly implicit denotes, that only neighbouring equations are depending on each other. In this case solving the IRK equation system using an iterative method is much faster than solving directly. However, the maximum time step reachable is not very large, since the implemented SDIRK scheme (Subsection 7.3) is only fourth order accurate.

Contrary to that, the IRK schemes of Gauss-Legendre type deliver very good accuracy, since the order of these IRK schemes is twice the number of stages employed. Therefore much larger time steps can be achieved. However, due to the fully implicit IRK equation system, the equation systemhas to be solved directly which makes these schemes numerically very costly.

It remains to be explored whether other IRK schemes perform better. There exist a whole variety of IRK schemes which could, from similar considerations as above, be suitable, e.g. explicit singly-diagonally implicit Runge-Kutta (ESDIRK) schemes or other members of the diagonally implicit Runge-Kutta (DIRK) family, and so on.

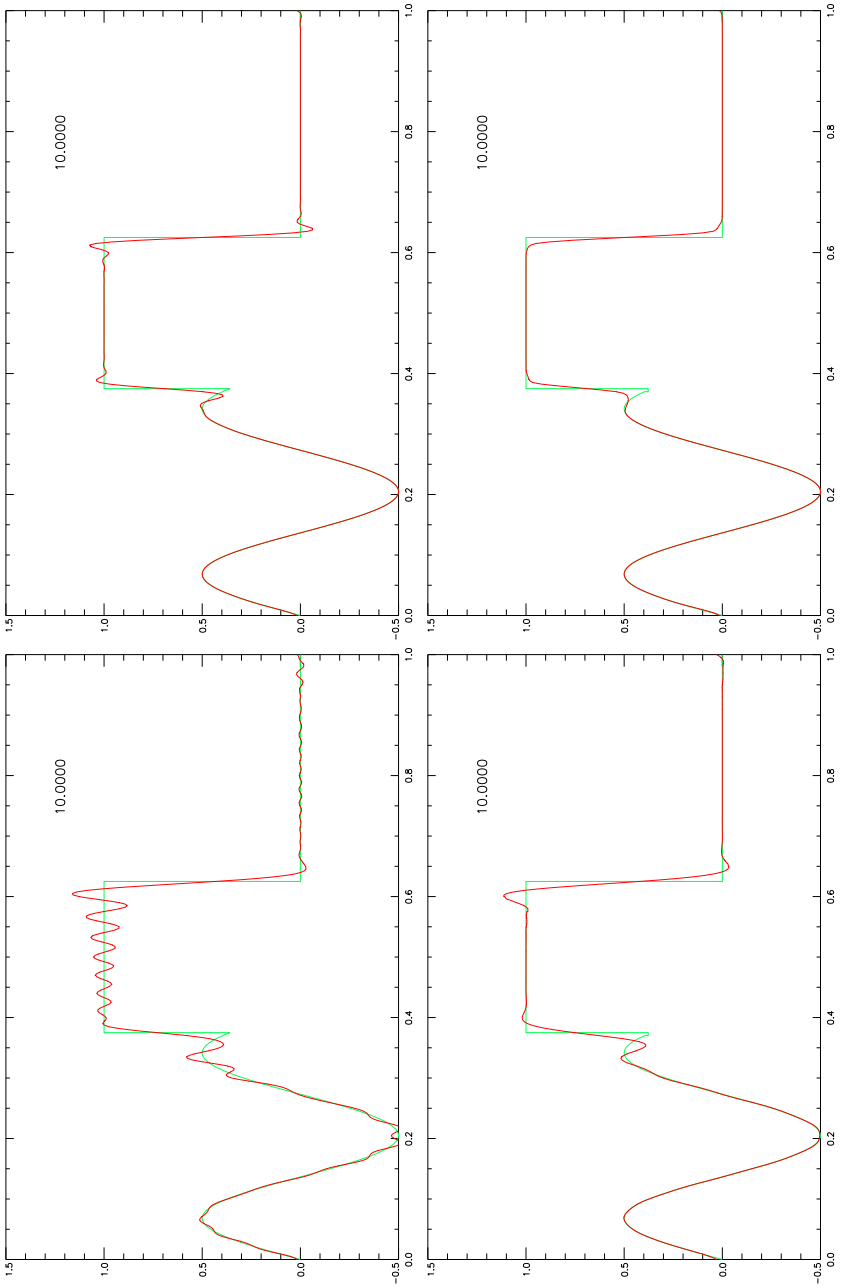

Figure 3: Solution of the advection equation using an 6th order implicit Runge-Kutta scheme of Gauss-Legendre type (Subsection 7.1) in combination with a DG-FEM scheme of order $p = 2$. Upper row correponds to CFL number 1.25 and lower row to 2.5. At the same time the right column shows solutions with limiting, whereas the left column without limiting.

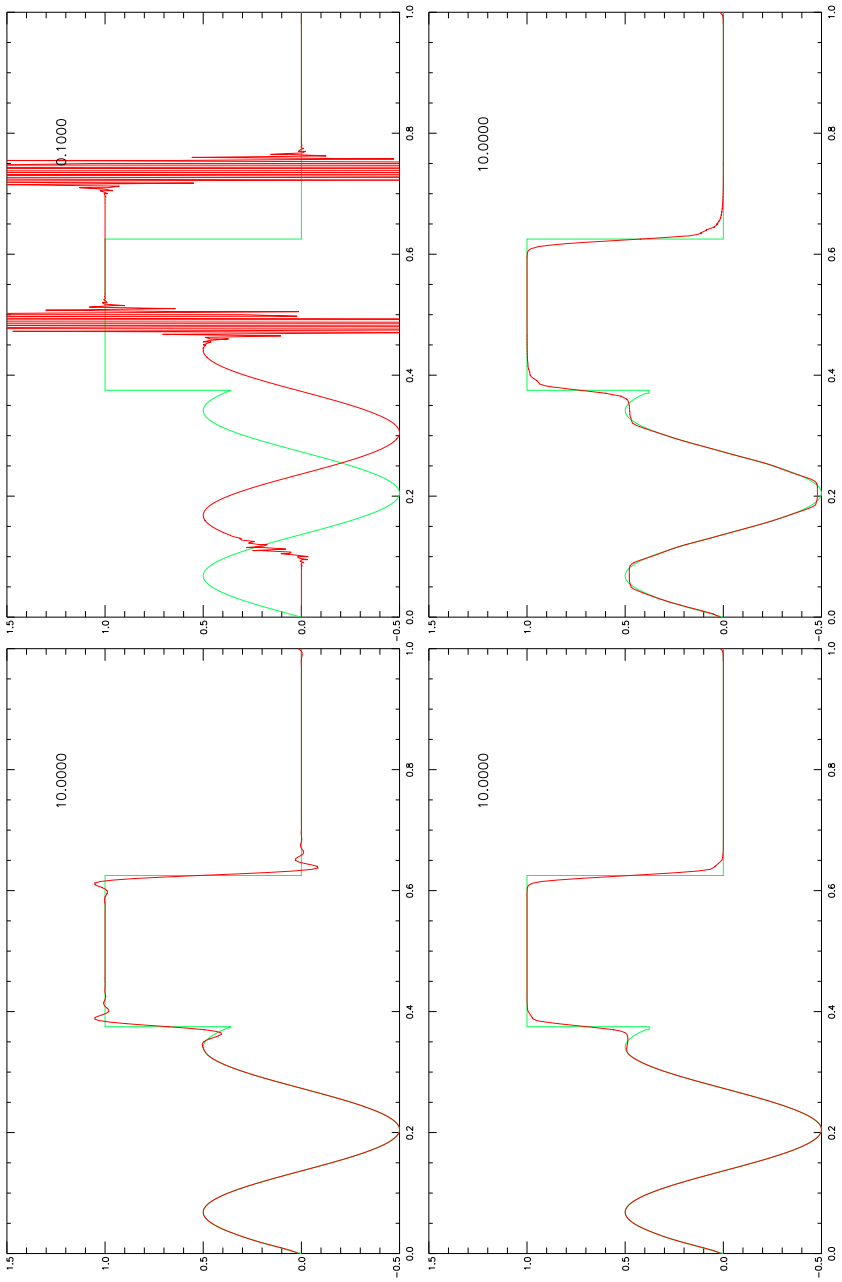

Figure 4: Solution of the advection equation using a 4 stage 4th order singly-diagonally implicit Runge-Kutta (SDIRK) scheme (Subsection 7.3) in combination with a DG-FEM scheme of order $p = 2$. Upper row correponds to CFL number 1.25 and lower row to 0.5. At the same time the right column shows solutions with limiting, the left column without limiting. Note that, without limiter the scheme is unstable for large CFL numbers (upper left), hence, the calculations was stopped at a time of 0.1

8 Limiting methods

First we define the so called minmod function used several times in the limiting methods discussed in this section

$$\text{minmod}(a_1, a_2, ..., a_n) = \begin{cases} s \cdot \min_{1 \le i \le n} |a_i| & \text{if } s = \text{sign}(a_1) = \cdots \\ & \cdots = \text{sign}(a_n) \\ 0 & \text{otherwise} \end{cases} . \quad (90)$$

It picks the argument with the smallest absolute value from a set of arguments with the same sign. In case of different argument signs, it assumes an extremum and returns the value zero.

8.1 A TVDM minmod slope limiter

This is a slope limiter method based on the minmod function [Eq. (90)] acting on the flux terms Ref. [2], p. 94. It limits the higher order part of the flux terms by using the difference quotients of the zeroth order terms. Then the modified flux function reads

$$\hat{h}^{\text{lim}}(u) = \hat{h}(u_i^0 + \hat{u}_{i,\text{mod}}^1) \quad (91)$$

$$\hat{u}_{i,\text{mod}}^1 = \text{minmod}\left(u_i^1, u_{i+1}^0 - u_i^0, u_i^0 - u_{i-1}^0\right) . \quad (92)$$

Note that this limiter is preserving the total variation diminishing in the mean (TVDM) character of a scheme, however, it generates some severe artefacts in the numerical solution (Figure 5 and 6, lower right panel). Local extrema within smooth parts of the solution are suffering from clipping and distortion. Consequently, this may not be a preferable limiting method.

8.2 A simple moment limiter

The slope limiter given here is a simple case of a so called moment limiter given in Ref. [3]. This limiter uses the difference quotients to limit the slope of the next higher order. Therefore, this moment limiter realises a convex limiter.

The limiter can be written in short form as

$$u_i^{n+1} = \text{minmod}\left(u_i^{n+1}, \frac{u_{i+1}^n - u_i^n}{2n+1}, \frac{u_i^n - u_{i-1}^n}{2n+1}\right) \quad (93)$$

using the minmod function defined in Eq. (90).

This limiter yields comparable results to the TVDM minmod slope limiter (Subsection 8.1). However, methods using this limiter are simply total variation bound (TVB). And moreover, this limiting method also suffers from clipping and distortion of local extrema (Figure 5 and 6, upper left panel).

8.3 A generalised moment limiter

This generalised moment limiter was developed in Ref. [4] and is based on the more simple moment limiter given in Ref. [3] which was described in Subsection 8.2.

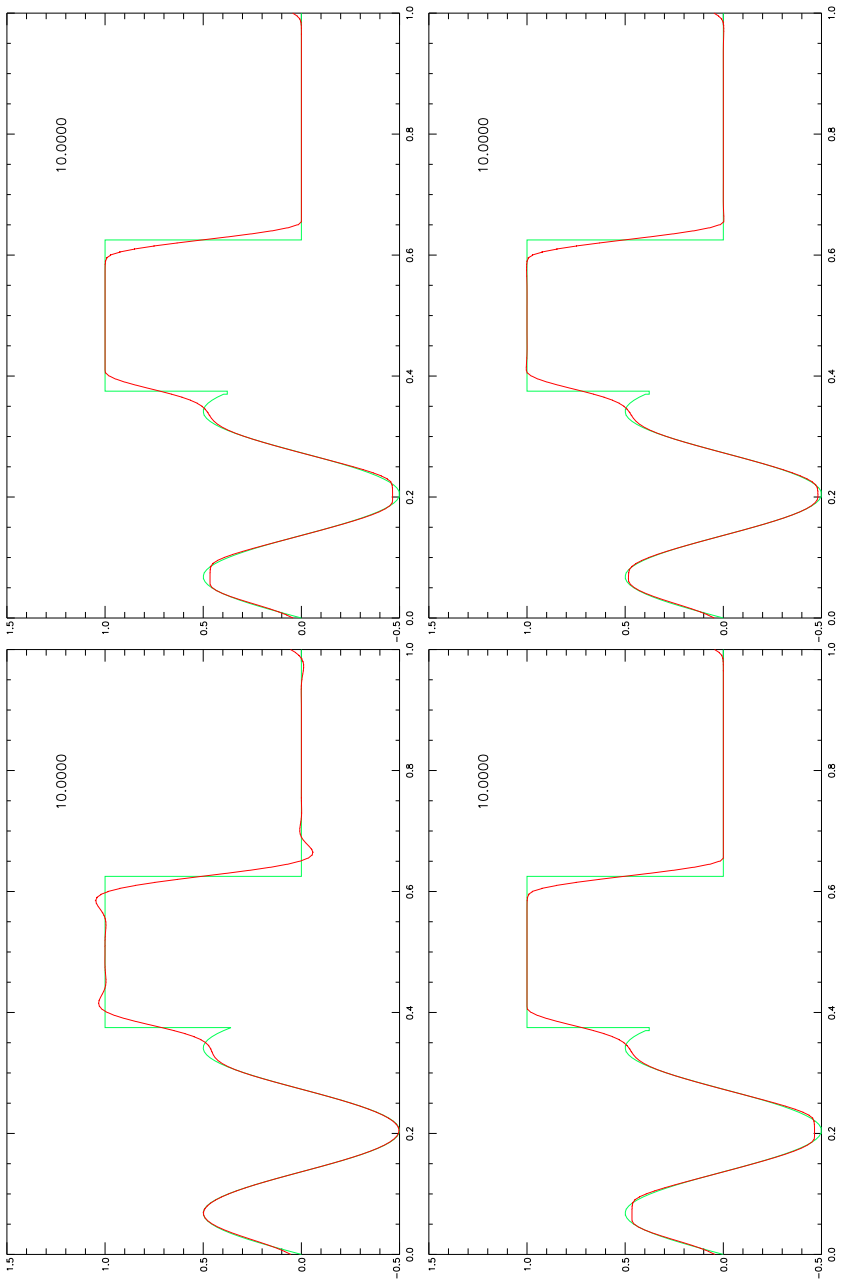

Figure 5: Solution of the advection equation using a square function plus a sine wake as initial condition. The x-axis indicates the time and the y-axis indicates the function Value. Red curves show the solution at $t = 10$, whereas green curves give the initial state. The calculations used 200 spatial grid points, polynomial approximation of order $p = 1$, CFL number of 0.2, a 3rd order TVD-ERK scheme (Section 5), and 1st order upwind fluxes. Lower left panel: without limiting. Lower right panel: using a TVDM minmod slope limiter (Subsection 8.1). Upper left panel: using a simple moment limiter (Subsection 8.2). Upper right panel: using a generalised moment limiter (Subsection 8.3).

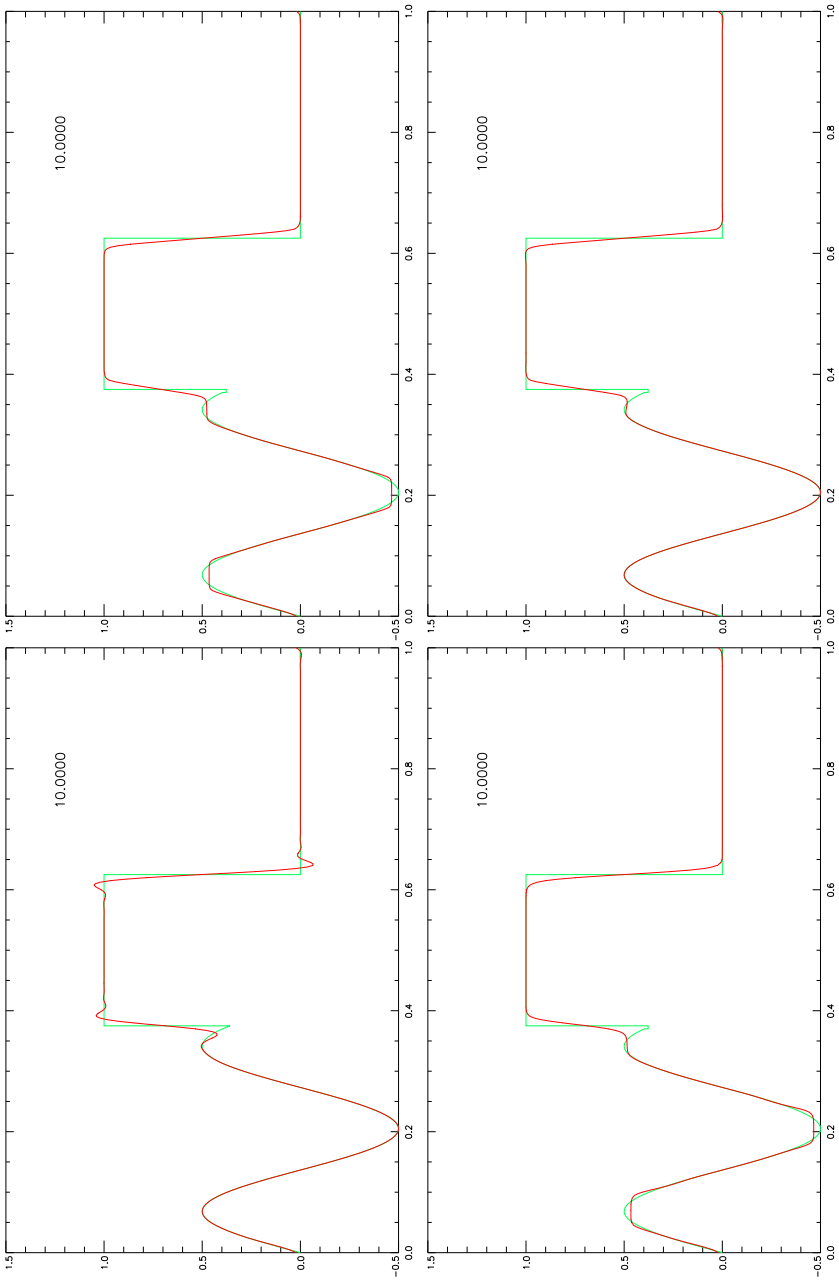

Figure 6: Same as Figure 5, however, using a polynomial approximation of order $p = 2$. Lower left panel: without limiting. Lower right panel: using a TVDM minmod slope limiter (Subsection 8.1). Upper left panel: using a simple moment limiter (Subsection 8.2). Upper right panel: using a generalised moment limiter (Subsection 8.3).

The limiter can be written in short form as

$$u_i^{n+1} = \begin{cases} u_i^{n+1} & \text{if } |u_i^{n+1}| \leq M_{n+1}h^2 \\ \text{minmod}\left(u_i^{n+1}, \frac{u_{i+1}^n - u_i^n}{2n+1}, \frac{u_i^n - u_{i-1}^n}{2n+1}\right) & \text{if } |u_i^{n+1}| \geq M_{n+1}h^2 \end{cases} \tag{94}$$

where $M_k \in [0, \infty]$ denotes an upper limit for the limiting and $h = (x_{i+1} - x_i)$ denotes the size of the current finite element. Note that for $M_k = \infty$ there is no limiting at all and for $M_k = 0$ the generalised moment limiter resembles the moment limiter given in Subsection 8.2.

In our opinion this limiter yields the best results of all deployed methods and probably is the preferable method of choice. The limited solution does not suffer from clipping at all. Moreover, due to the flexible upper bound of limiting this limiter can preserve more fine structured details of the solution, especially when combined with higher order methods ($p > 1$). However, there is no mathematical proof of its TVB properties, although it originates from the simple moment limiter given in Subsection 8.2.

Note that a very similar limiter can be found in Ref. [5].

8.4 A flux limited centred (FLIC) flux

The flux limited centred (FLIC) flux is a general flux limiter approach given in Ref. [6], using the first order centred (FORCE) flux as a monotone low order flux and the second order Richtmyer (RM) flux as high order flux.

$$\begin{aligned} \hat{h}_{i+\frac{1}{2}}^{\text{FLIC}}(u^-, u^+) &= \hat{h}_{i+\frac{1}{2}}^{\text{FORCE}}(u^-, u^+) \\ &+ \phi_{i+\frac{1}{2}} \left[\hat{h}_{i+\frac{1}{2}}^{\text{RM}}(u^-, u^+) - \hat{h}_{i+\frac{1}{2}}^{\text{FORCE}}(u^-, u^+)\right] \end{aligned} \tag{95}$$

where the FORCE flux is given by

$$\hat{h}_{i+\frac{1}{2}}^{\text{FORCE}}(u^-, u^+) = \frac{1}{2}\left[\hat{h}_{i+\frac{1}{2}}^{\text{LF}}(u^-, u^+) + \hat{h}_{i+\frac{1}{2}}^{\text{RM}}(u^-, u^+)\right] \tag{96}$$

and the RM flux by

$$\begin{aligned} \hat{h}_{i+\frac{1}{2}}^{\text{RM}}(u^-, u^+) &= \hat{h}(u^*) \\ u^* &= \frac{1}{2}\left[(u^- + u^+) - \frac{\Delta t}{\Delta x}\left(\hat{h}(u^-) - \hat{h}(u^+)\right)\right] \quad . \end{aligned} \tag{97}$$

Note that LF in Eq. (96) denotes the common Lax-Friedrich numerical flux which is defined as

$$\hat{h}_{i+\frac{1}{2}}^{\text{LF}}(u^-, u^+) = \frac{1}{2}\left[\hat{h}(u^-) + \hat{h}(u^+) - C(u^- + u^+)\right] \tag{98}$$

with $C = |h^{'}(u)|$ which is the eigenvalue of the Jacobian in the one dimensional case.

The limiting function in Eq. (95) is defined as

$$\phi_{i+\frac{1}{2}} = \min\left[\phi(r_{i+\frac{1}{2}}^-), \phi(r_{i+\frac{1}{2}}^+)\right] \tag{99}$$

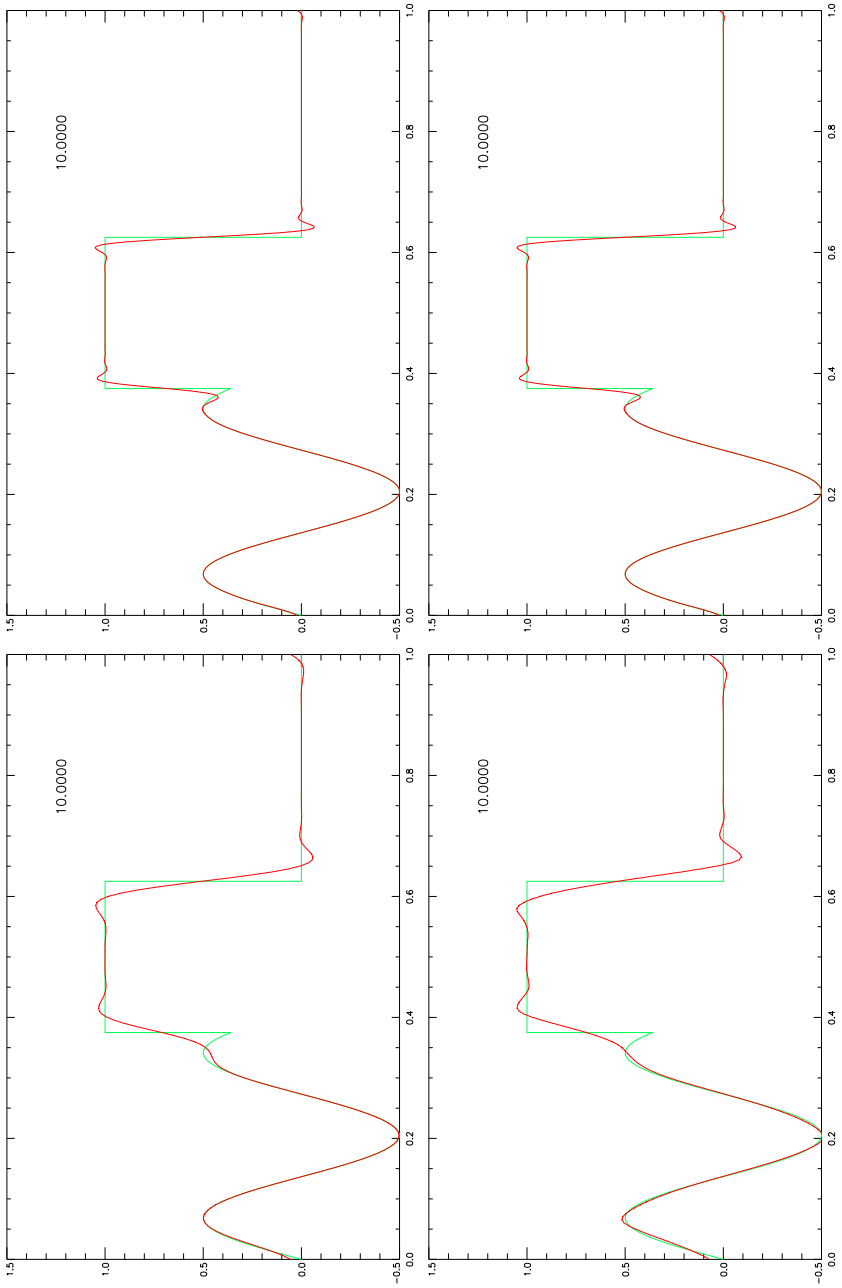

Figure 7: Same as Figure 5. Lower left panel: Polynomial approximation of order $p = 1$ using upwind fluxes. Lower right panel: Same as lower left panel, but using a generalised flux limiter approach and Sweby's flux limiter function. Upper left panel: Polynomial approximation of order $p = 2$ using upwind fluxes. Upper right panel: Same as upper left panel, but using the same flux limiter as in panel 2.

with the left and right hand side slope given by

$$r^-_{i+\frac{1}{2}} = \frac{u^0_i - u^0_{i-1}}{u^0_{i+1} - u^0_i} \quad , \qquad r^+_{i+\frac{1}{2}} = \frac{u^0_{i+2} - u^0_{i+1}}{u^0_{i+1} - u^0_i} \quad , \tag{100}$$

respectively.

We used Sweby's limiter function

$$\phi(r) = \begin{cases} 0 & \beta r \leq 0 \\ \beta r & 0 \leq \beta r \leq 1 \\ 1 & \beta r \geq 1 \end{cases} \quad . \tag{101}$$

This limiter function yields a TVDM method for $\beta \in [1, 2]$. Note that Ref. [6] used the minbee limiter, which is included in the Sweby limiter as a special case where $\beta = 1$. Furthermore, this limiting method can be combined with a whole bunch of well known limiter functions (e.g. see `http://en.wikipedia.org/wiki/Flux_limiters`)

The flux limiter method is not capable of eliminating spurious oscillations in the numerical solution of the advection equation when using higher order RKDG schemes ($p > 0$, compare Figure 7). The blended flux provided by this method has in either case a lower or equal viscosity than the upwind flux. Hence, wiggles will be less efficiently suppressed compared to the standard DG method using the upwind flux.

References

[1] J. S. Hesthaven and T. Warburton, *Nodal Discontinuous Galerkin Methods*, volume 54 of *Texts in Applied Mathematics*, Springer New York, 2008.

[2] B. Cockburn, Discontinuous Galerkin Methods for Convection-Dominated Problems, in *High-Order Methods for Computational Physiscs*, volume 9, pp. 69–224, Springer-Verlag Berlin Heidelberg New York, 1999.

[3] R. Biswas, K. D. Devine, and J. E. Flaherty, *Applied Numerical Mathematics* **14**, 255 (1994).

[4] H. M. Lui, Runge-Kutta Discontinuous Galerkin Method for the Boltzmann Equation, Master's thesis, Massachusetts Institute of Technology, 2006.

[5] L. Krivodonova, *Journal of Computational Physics* **226**, 879 (2007).

[6] J. Qiu, B. C. Khoo, and C.-W. Shu, *Journal of Computational Physics* **212**, 540 (2006).

[7] B. Cockburn and C.-W. Shu, *Mathematics of Computation* **52**, 411 (1989).

[8] S. Gottlieb and C. W. Shu, *Mathematics of Computation* **67**, 73 (1998).

[9] M. S. H. Khiyal and K. Rashid, *Journal of Applied Sciences 5* **3**, 411 (2005).

[10] G. Staff, Convergence and Stability of the Parareal algorithm: A numerical and theoretical investigation, Technical report, Norges Teknisk-Naturvitenskapelige Universitet, 2003.

[11] L. Ferracina and M. Spijker, *Applied Numerical Mathematics* **58**, 1675 (2008).