Freie Universität Berlin

SEMI-ANALYTICAL SEMI-LAGRANGIAN DISCONTINUOUS GALERKIN ADVECTION SCHEME FOR THE COMPRESSIBLE LINEAR ADVECTION EQUATION

Dissertation
zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
am Fachbereich Mathematik und Informatik
der Freien Universität Berlin
vorgelegt von

LINDA MICHALK

Berlin, 2018

Erstgutachter:
Prof. Dr. Rupert Klein
Institut für Mathematik
Fachbereich Mathematik und Informatik
Freie Universität Berlin
Arnimallee 6
14195 Berlin
Telefon: +49 30 838 75414
Fax: +49 30 838 75412
e-mail: rupert.klein@math.fu-berlin.de

Zweitgutachter:
Prof. Dr. Claus-Dieter Munz
Institut für Aerodynamik und Gasdynamik
Universität Stuttgart
Pfaffenwaldring 21
70569 Stuttgart
Telefon: +49 711 685 63401
Fax: +49 711 685 53402
e-mail: munz@iag.uni-stuttgart.de

Tag der Disputation: 10. April 2018

DANKSAGUNG

Zuallererst möchte ich mich bei meinem Betreuer Herrn Rupert Klein bedanken. Er hat mich an dieses spannende und relevante Thema herangeführt. In vielen intensiven Gesprächen habe ich wertvolle fachliche Anregungen erfahren.
Stefan Vater hat vorgeschlagen, die Second Order Moments Methode der Klasse der DG Verfahren gegenüber zu stellen. Mit Carsten Schultz konnte ich technische Details zur Trajektorienberechnung erörtern. Dank gebührt auch meinem Kollegen Martin Papke, mit dem ich nicht nur während unserer Mensabesuche über fachliche Dinge diskutieren konnte. Stetige Motivation habe ich von Thomas von Larcher in und außerhalb der Kaffeepausen erhalten. Fürs gründliche Korrekturlesen möchte ich mich bei Stephan Gerber, Thomas von Larcher, Gottfried Hastermann, Marnie Christensen, Wilma Weps und Florian Thiel bedanken. Mit viel Geduld hat mir Patrik Marschalik bei allen LaTex-Fragen zur Seite gestanden.
Meine Eltern haben mich in jeglicher Hinsicht während der Promotion unterstützt. Vielen Dank dafür! Und auch meinem Niels möchte ich danken. Er hat mich in den letzten Jahren durch alle Höhen und Tiefen begleitet und war immer für mich da.

CONTENTS

1 INTRODUCTION

Weather prediction, climate change, the distribution of atmospheric pollutants, and the development of tornados or hurricanes have been the subject of research for quite a long time and are of great interest. All of these topics can be described with the help of fluid dynamical processes. Without the development of numerical methods to solve the arising equations it is nearly impossible to find solutions to these problems, as the equations are generally too complicated to be solved analytically. [16]

The Navier-Stokes equation describes the momentum balance of viscous fluids, e.g. it is applied to atmospheric or oceanic flows. When this equation is combined with equations for the conservation of mass and energy, it becomes possible to model all kinds of fluid dynamical processes. The equations are simplified depending on the application. For example, the Euler equations neglect friction and can be used to model phenomena such as sound waves, gravity waves, cyclones, or the general circulation of the atmosphere. For weather prediction, yet another set of equations with different assumptions is used, see e.g. [17].

While the mass of particles within a control volume is conserved in time, the density of the control volume can change. According to the equation for conservation of mass, if the density of the control volume is taken into account, the only change that can occur in time comes in the form of flow with a given velocity over the boundaries of the control volume. Conservation of (total) energy consists of the motion of particles carrying energy. In addition to transport, energy changes due to compression or expansion as well as due to molecular processes. This means that the power of the pressure is part of the balance of energy. Heat conduction and the diffusion of mass contribute to the conservation of energy because of molecular transport phenomena. To model gravitational force, the gradient of the geopotential is added to the equation in conservative form. The change of momentum caused by the flow of mass is part of the momentum equation. Pressure and shear forces have to be included as well. Gravity has an influence on momentum. Thus, the geopotential is embedded just as it is for energy. In the context of meteorological problems, the equations are often given in a rotating reference frame wherein the Earth is stationary. Therefore, fictitious forces such as the Coriols force must be introduced. They appear in the momentum equation and can be written in conservative form. Note that in the case of a compressible fluid, an equation of state as the ideal gas law is needed to close the system and determine all unknowns. Also, the conservation of energy can be expressed as an equation for the evolution of potential temperature.

In any equation of the governing equation system, advection plays a role. The transport of mass and the associated transport of energy and momentum attached to the mass particles are advective processes. Therefore, advective terms are a fundamental part of the equations describing fluid flow. Tracers as state variables are called active tracers, because they directly and actively affect the solution. The phenomenon of advection can also occur in form of a passive tracer. In this case, the tracer is advected along the velocity field without any or with only negligible feedback to the system.

Under certain conditions, active tracers can be approximated by passive tracers, which simplifies the calculations. For example, when considering the atmospheric temperature field at large scales, the active tracer potential temperature can be treated as a passive tracer. [63]

Chemical tracers as ozone can be considered passive or active tracers, see e.g. [1] and [38]. At a time scale of several weeks or less, ozone can be treated as a passive tracer in the context of general circulations in the lower stratosphere. In this case, the spatial distribution due to advective transport plays a significant role, see [1]. If ozone is considered an active tracer, then the feedback occurs in a different form than the feedback of air density, momentum or energy. It does not feed back directly to induce changes in temperature and wind distribution. Instead, due to chemical reactions it can influence the ozone layer, which in turn changes radiative forcing, which then influences temperature and wind distribution. These chemical processes happen on a longer time scale.

In the ocean, temperature and salinity are examples of active tracers. Passive tracers include carbon dioxide and chlorofluorocarbon [7]. Gerdes emphasizes the importance of advection of active and passive tracers in [18]: "The evolution of temperature, salinity and passive tracers in the ocean interior is dominated by advection, although mixing processes are crucial for various aspects of the circulation. An essential requirement for ocean circulation models, therefore, is that the advection process be properly represented."

Each year to celebrate Saint Patrick's Day, a part of the Chicago river is dyed green. A picture taken in 2009 can be seen in Figure 1.1. Though, there is no (scientific) reason for this action other than the celebrations, it illustrates a passive tracer in a river. Observing the green dye, one can see how it is advected downstream.

Figure 1.1: Green dye (passive tracer) in the Chicago River on St. Patrick's Day. Credits: [65]

Processes in nature lead to models with equations of nonlinear structure including nonlinear advection. Different options exist to treat nonlinearities numerically. One possible approach is to linearize the system of governing equations, see [16]. Another ansatz using the EULAG model is applied in e.g. [44]. The authors describe a way to find the numerical solution to the equations of the anelastic model. This is obtained from the compressible Euler equations under certain assumptions about the density and the gravity term, see [27]. These equations can be expressed as advection equations with source terms as right-hand side. The velocity and the state variables are determined half a time step ahead by means of interpolation and extrapolation. These values are used as input for the linear advection equation. The source term is treated separately. In this thesis, we will address the case of linear advection with variable velocity as an important component of fluid dynamical systems.

LINEAR ADVECTION – CONSTANT VELOCITY

Physically, the linear advection equation with a constant coefficient might not be of great interest. To find an example for a phenomenon that can be modeled with this equation, many simplifying assumptions have to be made. For example, the motion

of a passive tracer in a fluid in a pipe with constant background velocity, without the effects of friction at the walls and other external forces such as gravity can be described using the linear advection equation with a constant velocity coefficient. Another example are packages considered as tracers on a conveyor belt operated with constant velocity. Or cars moving on a one-lane road at the same velocity and therefore with the same distances between them.

The linear advection equation with constant velocity u and y being the advected quantity for $y, u : \mathbb{R} \times [0, T] \to \mathbb{R}$ is given by

$$\begin{aligned} y(x,t)_t + uy(x,t)_x &= 0, \\ y(x,0) &= y_0(x). \end{aligned} \tag{1.1}$$

Periodic boundary conditions are assumed. The solution is to be computed on the time interval $[0, T]$. Because the velocity does not depend on space, (1.1) can equally be written in conservation form. In the case of constant velocity u, we know the analytical solution, which yields

$$y(x,t) = y_0(x - ut). \tag{1.2}$$

Nevertheless, from the point of view of the numerics solving (1.1) is not as simple as it might seem. As Leonard states in [34], "Modeling of highly advective transport is embarrassingly difficult, even in the superficially simple case of one-dimensional constant-velocity flow." A practical advantage of dealing with the numerics of this equation is knowing the analytical solution, because the numerical solution can be verified at any point in time and space.

LINEAR ADVECTION – VARIABLE VELOCITY

Even the simplified case of advection with a constant coefficient reveals itself as a numerically interesting and difficult case. However, advection processes in nature are far more complex, as velocity fields are obviously neither constant in time nor in space. Including a variable velocity coefficient makes it possible to capture more of these processes.

Still, finding realistic examples that are entirely modeled by this equation is nearly as difficult as for the equation with a constant velocity coefficient. The example of the conveyor belt can be extended to several conveyor belts in a row, operated at different velocities. Varying velocities cause the packages on the belts to be moved closer together if the packages are advected from a faster moving belt to a slower moving belt, or farther apart in the reverse case. The example of the cars can be extended to traffic moving in zones with different speed limits. In order to obey (1.3), cars have to reach the allowed maximum speed. Furthermore, there must be enough space between the cars, so that when they enter a section with lower speed limit, they do not crash. A more realistic model - which then becomes a nonlinear model - would incorporate the fact that a high density of cars leads to slower traffic. This is not covered by the linear advection equation. Nevertheless, a complicated nonlinear advection process in nature can still be approximated by a linear model.

The linear advection equation with variable velocity coefficient is given by

$$\begin{aligned} y(x,t)_t + u(x,t)y(x,t)_x &= 0, \qquad && \text{for } \mathbb{R} \times [0, T], \\ y(x,0) &= y_0(x) && \forall\, x \in \mathbb{R}. \end{aligned} \tag{1.3}$$

A problem that emerges from this form of the linear advection equation (1.3) is that it is not written in conservation form in contrast to (1.1). Thus, the traced quantity y is not necessarily conserved if the evolution of y is computed employing this nonconservative form. In order for a numerical method to inherit the conservation property for y, the equation itself must be in conservation form. To achieve this

goal we need to introduce the variable ρ, which can be interpreted as a density distribution for the surrounding fluid as air or water, and is therefore given in kilograms of fluid per cubic meter. Then, y becomes the mixing ratio of the tracer divided by mass (kilogram of tracer per kilogram of fluid) and further ρy is the tracer density (kilogram of tracer per cubic meter)

$$\rho_t + (u\rho)_x = 0, \tag{1.4}$$
$$(\rho y)_t + (u\rho y)_x = 0 \tag{1.5}$$

This set of equations with according initial values implies (1.3). Within this set of equations, two equations have to be solved, but now they are in conservation form. The mass of the fluid and the mass of the tracer are conserved in time. From that we can develop a numerical method with the desired property. Both equations are treated equally, with the same numerical method. After the numerical solutions of ρ and ρy respectively are computed, we finally obtain the information on the traced quantity y by dividing ρy by ρ.

SUMMARY OF NUMERICAL ADVECTION METHODS

Many different numerical methods are available for solving fluid dynamical problems. They can be roughly divided into different classes: finite difference methods, finite volume methods, discontinuous Galerkin methods, finite element methods and spectral methods. The latter two are not discussed in this thesis, while the other methods are briefly introduced in the following. Beforehand, we want to mention that the choice of the class of numerical methods depends on the specific problem and on the focus of the solution. Properties like conservation of a quantity, high order accuracy, positive definiteness, computational efficiency, the possibility of large time steps, etc. can influence the choice of numerical scheme. All of the numerical methods listed below have been tested with the linear advection equation as it is of simple structure and, in the case of a constant velocity coefficient, the solution is known, which makes it a practical test problem. While some of the methods are only intended to solve the advection equation, others are generalized to treat other equations as well. This summary is by no means exhaustive; only some aspects and methods are introduced that are relevant to the method developed in this thesis.

The class of finite difference methods was applied as early as in 1922 when Lewis Richardson used these methods to solve fluid dynamical equations in an attempt at numerical weather prediction, described in [49]. The fundamental work of Courant, Friedrichs and Lewy in 1928 [37] further investigates finite difference methods that generally enable a discretization of partial differential equations (PDE). The finite differences, that is the differences of pointwise given values, approximate the derivatives in the equation. In [37], the authors make a finding of great importance: the correlation of the grid cell sizes and the characteristics of the problem must fulfill certain inequalities in order to obtain convergence of the finite difference approximation. Accordingly, this condition is known as the Courant-Friedrichs-Lewy (CFL) condition. The ratio of the length of a time step in a numerical method as well as the velocity and the size of a grid cell must be limited by the CFL number (or Courant number). This can be a restrictive condition on the time step size to fulfill the stability requirement. Finite difference schemes are not further addressed in this thesis: however, the CFL condition plays a crucial role in other numerical methods as well.

One class of numerical schemes used to compute the solution to conservation laws are finite volume methods. The PDE is written in integral form in space and in time. Thus, cell averages, obtained by integration and cellwise averaging, are used to describe the numerical solution instead of pointwise values. Finite volume methods make use of the divergence theorem by converting the integral of the divergence term in surface integrals. These surface integrals integrated in time

describe the fluxes that flow from one grid cell to the adjacent neighbors. The outgoing flux from one cell is identical to the ingoing flux of the neighboring cell. Therefore, finite volume methods are conservative, as the total integral over the whole domain is preserved. The key challenge for finite volume methods is to find good approximations of the fluxes. A thorough discussion of finite volume methods can be found e.g. in [36].

Godunov's method developed in 1959 [21] is among the finite volume methods. In the first step of the method, piecewise constant functions are defined for each grid cell. Then, the solution to Riemann problems is computed forward in time at the cell boundaries. Finally, integration and averaging leads to the piecewise constant functions after the time step. As only the cell average needs to be computed at the end and the equation to be solved is written in integral form, the solution to the Riemann problem is simplified. Godunov's method is first-order accurate, because it uses piecewise constant functions to represent the solution. To increase the accuracy, higher order methods were designed.

A second-order accurate sequel to Godunov's method is the Monotone Upstream Scheme for Conservation Laws (MUSCL), introduced in [32] by van Leer in 1979. This method follows the same structure as Godunov's method with the difference that instead of the constant functions in each cell, linear functions are reconstructed after each time step. In 1984, Colella and Woodward generalized the functions to parabolas, leading to the piecewise parabolic method (PPM) [13], which uses the same framework and is of third-order accuracy. The higher-order extensions to Godunov's method have in common that spurious oscillations can arise in the numerical solution near discontinuities, whereas in smooth regions the high-order accurate methods approximate the analytical solution more accurately.

This problem led to another field of interest in the community: total variation diminishing (TVD) methods and in connection to that flux and slope limiting. The methods MUSCL and PPM can avoid oscillations in the numerical solution when slope limiting is applied. In the reconstruction step, the linear or parabolic function respectively has to be modified in such a way that existing extremal points are not incremented and no additional extremal points are created. As a consequence, the slope of the reconstructed function is set to zero at extrema. Furthermore, the slope is limited such that all other points of the function lie within the range of neighboring function values.

A different approach involves limiting the flux. This scheme was introduced by Boris and Book [6] as the flux corrected transport (FCT) method and further developed by Zalesak [66]. The idea of FCT methods is to compute both a low-order numerical flux as in Godunov's method and a high-order flux, and use a combination of both fluxes: the low-order flux near discontinuities and the high-order flux at smooth regions. The numerical solution employing the low-order flux is first computed and then, in a second step, corrected using a weighted difference of the low- and high-order flux, known as the anti-diffusive flux. The weight is determined in such a way as to avoid new extrema and the amplification of existing ones.

A different but related approach to the FCT method is the multidimensional positive definite advection transport algorithm (MPDATA) developed by Smolarkiewicz [56] in 1983. This method is iterative and corrects the diffusive error in the numerical flux repeatedly in order to increase accuracy. The FCT method computes low- and high-order fluxes in order to determine the flux correction. In contrast, with MPDATA it is sufficient to compute low-order fluxes with corrected antidiffusive velocities in each iteration. A more detailed description of MPDATA is given in Section 2.6.

Yet another family of methods for solving fluid dynamical problems are Discontinuous Galerkin (DG) methods. They were originally introduced by Reed and Hill [47] in 1973 to solve the neutron transport equation. LeSaint and Raviart [35] conducted a first analysis and a proof of convergence in 1974. DG methods are a

combination of finite volume methods and finite element methods. They are written in conservation form as finite volume methods using numerical fluxes across cell interfaces and solved using a weak formulation as employed in finite element methods. Much of the theory developed in each of the other classes is applied in DG methods, such as slope limiters used for finite volume methods or theorems for convergence established for finite element methods. Further developments and a thorough description of DG methods can be found e.g. in [10]. An introduction to DG methods is given in Section 2.4, because the new numerical method presented in this thesis is related to the family of DG methods.

The concept of semi-Lagrangian methods is described in Section 2.5 and therefore only sketched here. In an Eulerian framework, as used in all the other methods described above, the evolution of the solution of fluid flow is studied from a fixed point in space. In contrast, in a Lagrangian framework, the evolution is observed by following a particle of the fluid in motion. In a fully Lagrangian method, a set of fluid particles is traced in time, with the disadvantage that these particles will end up irregularly spaced. The semi-Langrangian approach combines the Eulerian and the Lagrangian frameworks. Exactly these particles that arrive at the Eulerian grid points at the end of a time step are traced in a Lagrangian manner. In this way the grid points remain regularly distributed and large time steps can be made, which is the advantage of Lagrangian methods.

The ability to avoid restrictions on the time step because of the CFL condition is of particular importance for grids with strongly varying cell sizes. Different scales in horizontal and vertical direction often occur in atmospheric phenomena. Grid aspect ratios of $\mathcal{O}(100)$ are typical when comparing horizontal and vertical scales, see e.g. [41]. The smallest grid cell determines the time step size. In more than one space dimension, when operator splitting [62] is used to compute the solution, small cells in one space dimension can limit the time step sizes of the other space dimensions. Methods that are not affected by the CFL condition for stability reasons are favorable in such situations.

We would like to highlight one particular numerical method - the second order moments (SOM) method, and not only because "[...] the use of the SOM method significantly improves tracer distributions and transports compared to FCT and QUICKer [quadratic upstream interpolation for convective kinematics], thus leading to a better representation of ocean currents, notably boundary currents and frontal systems" [26]. The scheme was developed independently by Prather [43] in 1986 and by van Leer [31] in 1977 and can be assigned to the class of finite volume methods and DG methods. The tracer is represented by a higher-order polynomial for each grid cell. Polynomials are advected and after one time step projected again onto the space of high-order polynomials in a way that is comparable to the procedure used for DG methods. Yet, the classical finite volume conservation form is used as starting point for the computation.

In this thesis, we develop a new numerical method that is based on the SOM method. Therefore, Section 2.3 is devoted to a detailed description of this numerical scheme. The method is designed to incorporate the best assets of the above-mentioned numerical methods. First, it uses the conservation property of FV methods. Second, the ability must be given to limit the slope of the functions representing the advected quantity for positive definiteness and avoid spurious oscillations. Third, the high accuracy of DG methods is incorporated, and further it achieves the accuracy of the SOM method because of its exact integration in time. And last, the CFL stability criterion is resolved using a similar ansatz, which traces characteristics back in time, as in semi-Lagrangian methods.

Different methods have been developed that originate from the same idea to combine a semi-Lagrangian ansatz with DG methods. There is a class of methods that pursue that goal called Lagrange-Galerkin methods, described e.g. in [19]. A numerical scheme that belongs to this class as well, though named a semi-Lagrangian discontinuous Galerkin scheme, is described in [22]. The numerical solution is de-

rived from the Lagrangian form of the integral over the product of the advected quantity and test functions. This test function has to fulfill a certain property, i.e. solve the adjoint problem of the linear advection equation conservation form. This special form makes it possible to determine the solution: however, the integrals are determined approximately.
Another related method, also known as a semi-Lagrangian discontinuous Galerkin scheme, is introduced in [48]. It uses the weak discontinuous Galerkin formulation of the advection equation as a basis. The numerical flux, the trajectories described by an ordinary differential equation, and the integrals that arise from that equation are solved approximately.
The numerical method constructed in this thesis stands in contrast to the other methods as it is suitable for divergent velocity fields. It explicitly treats the transport of density and tracer density. More importantly, the ordinary differential equation defining the trajectories and all integrals that originate from the solution ansatz are solved analytically. This gives an insight into the possible accuracy of the numerical solution that can be achieved. However, as a consequence of the projection of the analytical solution onto the space of polynomials after each time step, the method results in a semi-analytical scheme. In this thesis, we develop and examine a semi-analytical, semi-Lagrangian discontinuous Galerkin (SASLDG) advection scheme for the compressible linear advection equation.

OUTLINE

Chapter 2 introduces numerical methods for the linear advection equation that form the basis for the SASLDG method, i.e. the SOM method by Prather and the equal method by van Leer. In addition, the classes of DG methods and semi-Lagrangian methods are described. The method MPDATA is presented, because a combination of MPDATA and the SASLDG method via operator splitting is applied to a certain kind of problem. Additionally, the von Neumann analysis is described as a tool for analyzing the accuracy of linear numerical methods, which will be applied in a modified way to the nonlinear MPDATA.

The SASLDG method is developed and described in Chapter 3. After a preview that highlights the concept of the method, the construction of the method starts with the trajectories that result from the semi-Lagrangian ansatz, followed by a description of the exact solution. The projection step is elaborated in the ensuing section. The overall algorithm is described to provide an idea of the structure of the actual implementation. The limits of the integration, which are part of the projection step, are described in the subsequent section. This chapter concludes with the numerical results of the SASLDG method in one space dimension.

A possibility to extend this approach to two space dimensions is discussed in Chapter 4. We introduce operator splitting, which enables a straightforward extension. Subsequently, we present the results of numerical tests of the SASLDG method in two dimensions. In the following section, we explain a possible treatment for numerical solutions on grids with high aspect ratio. We suggest using a combination of MPDATA and a modified version of the SASLDG method. Extensive numerical tests conclude this chapter.

An analysis of theoretical properties such as consistency, stability and convergence is carried out in Chapter 5. In addition, to the L_1- and L_2-stability analysis, a von Neumann stability analysis is applied to gain further insight into the SASLDG method. Last, we confirm the theoretical findings with numerically determined convergence rates.

In Chapter 6, we discuss the results of the preceding chapters, focusing in particular on the outcome of the numerical tests in one and two space dimensions. Moreover, we give an outlook on open questions and further research on this topic.

2 BASIC METHODS AND BACKGROUND

In this chapter we introduce the numerical methods that build the foundation for the SASLDG method and are related to it. Further, we discuss the technique of Fourier analysis that is useful to examine stability and convergence properties of numerical schemes. We begin with a description of the discretization of the computational domain and introduce the notation used in this thesis.

Section 2.1 presents the Fourier analysis. In the case of a constant velocity field the numerical methods developed by van Leer (Section 2.2) and Prather (Section 2.3) are identical to the SASLDG method. Thus, their ansatz and procedure is very similar to the SASLDG method. Discontinuous Galerkin methods, introduced in Section 2.4, provide a different possibility to solve the linear advection equation. It will turn out that in a special case a Discontinuous Galerkin method can be equal to van Leer's and Prather's method. Therefore, they offer a different point of view to the SASLDG method. The semi-Lagrangian schemes introduced in Section 2.5 enable a possibility to extend the SASLDG method with respect to the size of the time step.

In Section 2.6 we describe the numerical method MPDATA. This scheme is suited to be coupled with the SASLDG method to deal with a special class of problems, where the aspect ratio of a grid is high with resulting flat grid cells. This problem is discussed in Chapter 4.

The spatial domain Ω is subdivided in N grid cells $\Omega_i = [x_{i-1/2}, x_{i+1/2}]$ for $i = 1, \dots, N$ with grid cell length $\Delta x_i = x_{i+1/2} - x_{i-1/2}$. The grid cells are not necessarily of equidistant length. The cell center is denoted by x_i. If not mentioned otherwise the spatial domain equals the unit interval $[0, 1]$. The time interval $[0, T]$ is discretized by n time steps of length $t^{k+1} - t^k = \Delta t$, with $n\Delta t = T$.

The notion of the Courant-Friedrichs-Lewy (CFL) number or Courant number appears at many points in this thesis. It is defined as

$$\sigma = \frac{u\Delta t}{\Delta x} \tag{2.1}$$

for constant velocity u and equidistant grid cells. The given relation (2.1) of velocity, time step and grid cell size is a value used often as an abbreviation or in connection with time step size restrictions for stability reasons for many numerical methods. If the velocity is variable in space, the CFL number can be defined globally by taking the global maximum velocity or locally for each grid cell by taking the local maximum velocity. Then, the local CFL number reads

$$\sigma_i = \max_{x \in \Omega_i} u(x) \frac{\Delta t}{\Delta x_i}. \tag{2.2}$$

2.1 FOURIER ANALYSIS

Fourier analysis is a tool that can be used to find analytical solutions of partial differential equations as the linear advection equation (applied in Section 2.1.1) or to examine properties of a numerical method in the framework of a von Neumann stability analysis (see Section 2.1.2).

2.1.1 Fourier Transform

We begin with the definition of the Fourier transform of a L^2-function.

Definition 2.1.1 (Fourier Transform) *Let $x \in \mathbb{R}$ and $y(x) : \mathbb{R} \to \mathbb{R}$, with $y \in L^2(\mathbb{R})$. Then*

$$y(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{y}(x) e^{ikx} \, \mathrm{d}k \tag{2.3}$$

where

$$\hat{y}(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} y(x) e^{-ikx} \, \mathrm{d}x \tag{2.4}$$

is the Fourier transform of y.

With the help of Definition 2.1.1 we can find the analytical solution to the linear advection equation (1.1).

We represent the solution using the Fourier transform

$$y(x,t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{y}(k,t) e^{ikx} \, \mathrm{d}k \,. \tag{2.5}$$

Now we use the definiton of the linear advection equation (1.1) and find that the following holds

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{y}_t(k,t) e^{ikx} \, \mathrm{d}k = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} -uik\hat{y}(k,t) e^{ikx} \, \mathrm{d}k \tag{2.6}$$

for all x and t. For the initial state, $t = 0$, we can derive the equality

$$\hat{y}(k,0) = \hat{y}_0(k) \tag{2.7}$$

of the Fourier transform. Further we obtain the ordinary differential equation (ODE) for each k,

$$\hat{y}_t(k,t) = -uik\hat{y}(k,t), \tag{2.8}$$

with the initial values given in (2.7). The solution of this ODE can be computed

$$\hat{y}(k,t) = \hat{y}_0(k) e^{-uikt} \tag{2.9}$$

for each k. The solution to the linear advection equation is then given by

$$y(x,t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{y}_0(k) e^{-uikt} e^{ikx} \, \mathrm{d}k \tag{2.10}$$

$$= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{y}_0(k) e^{ik(x-ut)} \, \mathrm{d}k \tag{2.11}$$

$$= y_0(x-ut). \tag{2.12}$$

Note that the modulus of the Fourier coefficients of the linear advection equation does not change in time, as

$$|\hat{y}(k,t)| = |\hat{y}_0(k)||e^{-uikt}| \tag{2.13}$$

$$= |\hat{y}_0(k)|. \tag{2.14}$$

The typical representation of a plane wave is of the form

$$y = \hat{y} e^{i(kx-\omega t)}, \tag{2.15}$$

where k is the wave number and ω the frequency. Comparing this with (2.11) we find that the frequency of the waves of the Fourier transform of the linear advection equation is given by

$$\omega(k) = uk. \tag{2.16}$$

The dependence of ω on k is called the dispersion relation. In the case of the linear advection equation ω and k are proportional.

The phase velocity for each wave is defined as

$$v = \frac{\omega}{k}. \tag{2.17}$$

Each wave occuring in (2.11) has the same phase velocity

$$v = u, \tag{2.18}$$

for all k. As the phase speed in (2.18) is the same for all k, we know that the analytical solution to the linear advection equation does not disperse. If we had different values of v for waves of different wave number k, then the waves would disperse.

2.1.2 Von Neumann stability analysis

As we have seen in the previous section the Fourier transform enables us to find an analytical solution to a partial differential equation and, furthermore, reveals information on some properties of the solution to the equation. The discrete Fourier transform (DFT) is the discrete version of the Fourier transform. It enables us to determine properties of a numerical method. For linear numerical methods it is possible to conduct a von Neumann stability analysis based on the DFT. It shows stability properties of the method and gives information on the numerical diffusion and on the dispersion introduced artificially by the method.

Here, we follow the description by Hirsch [25] and decompose the numerical solution u_j^n at time step n and grid point j in finite Fourier series in space. This is done with a finite sum over all wave numbers that can be resolved on the grid. Here, we make use of MATLAB's DFT function *fft*.

Definition 2.1.2 (Discrete Fourier Transform) *Let u_j be a vector of length N, defining values on the unit interval $[0,1]$ on grid points $x_j = (j-1)\Delta x$ for $j = 1,\dots,N$ with equidistant spacing Δx. Then, the DFT is given by*

$$\hat{y}_k = \sum_{j=1}^{N} y_j e^{-2\pi i(k-1)(j-1)/N} \tag{2.19}$$

$$= \sum_{j=1}^{N} y_j e^{-2\pi i(k-1)x_j}, \quad \text{for } k = 1,\dots,N, \tag{2.20}$$

and the inverse DFT is given by

$$y_j = \frac{1}{N}\sum_{k=1}^{N} \hat{y}_k e^{2\pi i(k-1)(j-1)/N} \tag{2.21}$$

$$= \frac{1}{N}\sum_{k=1}^{N} \hat{y}_k e^{2\pi i(k-1)x_j}, \quad \text{for } j = 1,\dots,N. \tag{2.22}$$

The coefficient $\hat{y}_k$ is the amplitude associated with the kth wave number. To simplify notation later on, the DFT can also be expressed using the value α_k defined as

$$\alpha_k = 2\pi(k-1)\Delta x, \quad k = 1,\dots,N. \tag{2.23}$$

Then, we can write the kth Fourier coefficient as

$$\hat{y}_k = \sum_{j=1}^{N} y_j e^{-i\alpha_k(j-1)} \quad \text{for } k = 1, \ldots, N. \tag{2.24}$$

A linear numerical method is said to be stable according to von Neumann's method if for any arbitrary k the amplitude $\hat{y}_k$ does not grow indefinitely over time. This statement leads to the definition of the amplification factor

$$G_{\text{num}} = \frac{\hat{y}_k^{n+1}}{\hat{y}_k^n}, \tag{2.25}$$

where the index n denotes the number of time steps. The condition has to hold for any k. Hence the von Neumann stability condition states that the modulus of the amplification factor has to be bounded by one

$$|G_{\text{num}}| \leq 1, \quad \text{for all } k \in \{1, \ldots, N\}. \tag{2.26}$$

Note that an amplification factor of less than one results in a stable numerical method. For the linear advection equation, we know that the smaller the factor the more damping occurs in the numerical solution.

We know the exact amplification factor for the linear advection equation from (2.9). It is given by

$$G_{\text{ana}} = e^{i\omega\Delta t} = e^{iuk\Delta t}. \tag{2.27}$$

Note that $|G_{\text{ana}}| = 1$. We can rewrite 2.25 in polar form and obtain

$$G_{\text{num}}(k) = |G_{\text{num}}(k)|e^{i\Phi(k)}, \tag{2.28}$$

for some $\Phi(k)$. In that way, the diffusive and dispersive error can be examined individually.

Then, the absolute diffusion error e_{diff}, and the relative diffusion $\tilde{e}_{\text{diff}}$ error are given by

$$e_{\text{diff}}(k) = |G_{\text{num}}(k)| - 1 \tag{2.29}$$

and

$$\tilde{e}_{\text{diff}}(k) = \frac{|G_{\text{num}}(k)|}{|G_{\text{ana}}|}, \tag{2.30}$$

for each k. Furthermore, we obtain the absolute dispersion error

$$e_{\text{disp}}(k) = \Phi(k) - uk\Delta t, \tag{2.31}$$

and the relative dispersion error

$$\tilde{e}_{\text{disp}}(k) = \frac{\Phi(k)}{uk\Delta t}. \tag{2.32}$$

In the following example we show the procedure of the von Neumann stability analysis by means of the first order upwind (FOU) method.

Example 2.1.1 *We use the FOU method to solve the linear advection equation* (1.4) *in conservation form. The scheme reads*

$$\rho_j^{n+1} = \rho_j^n - \left(F(\rho_j^n, \rho_{j+1}, u_{j+1/2}^n) - F(\rho_{j-1}^n, \rho_j, u_{j-1/2}^n)\right) \tag{2.33}$$

where the numerical flux F is given by

$$F(\rho_j, \rho_{j+1}, u) = \frac{\Delta t}{2\Delta x}\left((u + |u|)\rho_j + (u - |u|)\rho_{j+1}\right). \tag{2.34}$$

Assuming a positive velocity u, the numerical method simplifies to

$$\rho_j^{n+1} = \rho_j^n - \sigma\left(\rho_j^n - \rho_{j-1}^n\right), \tag{2.35}$$

where σ is the CFL number. We replace each coefficient ρ_j^n with its Fourier transform and indicate the time level at the Fourier coefficients and multiply by N,

$$\sum_{k=1}^{N} \hat{\rho}_k^{n+1} e^{ij\alpha_k} = \sum_{k=1}^{N} \hat{\rho}_k^n e^{ij\alpha_k} - \sigma\left(\sum_{k=1}^{N} \hat{\rho}_k^n e^{ij\alpha_k} - \sum_{k=1}^{N} \hat{\rho}_k^{n+1} e^{i(j-1)\alpha_k}\right). \tag{2.36}$$

In the next step it suffices to consider only the coefficients for one arbitrary k. Division by $\exp(ij\alpha_k)$ leads to

$$\hat{\rho}_k^{n+1} = \hat{\rho}_k^n\left(1 - \sigma\left(1 - e^{-i\alpha_k}\right)\right). \tag{2.37}$$

We obtain the amplification factor for the FOU method by division of (2.37) by $\hat{\rho}_k^n$ according to (2.25),

$$G_{\mathrm{FOU}} = 1 - \sigma\left(1 - e^{-i\alpha_k}\right). \tag{2.38}$$

The modulus yields the amplitude

$$|G_{\mathrm{FOU}}(k)| = \left(1 - 4\sigma(1-\sigma)\sin^2\left(\frac{\alpha_k}{2}\right)\right)^{\frac{1}{2}}, \tag{2.39}$$

and the argument of the complex expression is the phase of the FOU method

$$\Phi_{\mathrm{FOU}}(k) = \arctan\left(\frac{\sigma\sin(\alpha_k)}{1 - \sigma + \sigma\cos(\alpha_k)}\right). \tag{2.40}$$

With the modulus of the amplification factor (2.39) we can determine the stability of the FOU method with the von Neumann stability condition (2.26). Looking at (2.39) we notice that the method is stable, if and only if $0 \leq 1 - 4\sigma(1-\sigma) \leq 1$, which holds if $0 \leq \sigma \leq 1$.

The stability analysis introduced in this section will be used to examine a few of the numerical methods, that are presented in the following. Though, the von Neumann stability analysis is in general eligible for linear numerical methods only, we will apply a modified version to the nonlinear method MPDATA, see Section 2.6. Then, the SASLDG method is investigated using the Fourier analysis in Chapter 5.

2.2 TOWARDS THE ULTIMATE CONSERVATIVE DIFFERENCE SCHEME

In sequel IV of the paper series "Towards the Ultimate Conservative Difference Scheme. IV. a new approach to Numerical Convection", see [31], van Leer presents different upstream centered schemes to find the numerical solution of the linear advection equation (1.1). The most accurate method is described in two different versions: a second-order accurate method (labeled scheme III) and a third-order accurate method (labeled scheme VI). The latter one will be discussed here. This numerical method turns out to be equal to the SOM method introduced in Section 2.3.

The notion of van Leer's method is to use the known analytical solution of the linear advection equation (1.2) to proceed the spatial distribution in time and apply least-squares fitting to remain in a simple, i.e. the polynomial space. Thus, any arbitrary initial distribution, given as a function $W(t^0, x)$ at time t^0, is projected in the least-squares sense onto polynomials of order two in each grid cell as a first step. The coefficients of the polynomials are considered as independent state quantities

for each cell. This results in a function $w(t,x) \in L^1(\Omega)$ defined for the whole domain and is given for each grid cell $w|_{\Omega_i} \in P^2(\Omega_i), i = 1,\dots,N$ where $P^2(\Omega_i)$ denotes the space of polynomials of degree two. It is of the form

$$w(t^n,x) = w_{0,i}^n k_0^{(i)}(x) + w_{x,i}^n k_1^{(i)}(x) + \frac{1}{2} w_{xx,i}^n k_2^{(i)}(x), \tag{2.41}$$

for $x_{i-1/2} \le x \le x_{i+1/2}$ at time t^n with Legendre polynomials k_j for $j = 0,1,2$ chosen as basis functions. These are given for the ith grid cell by

$$k_0^{(i)}(x) = 1 \tag{2.42}$$

$$k_1^{(i)}(x) = \frac{x - x_i}{\Delta x} \tag{2.43}$$

$$k_2^{(i)}(x) = \left(\frac{x - x_i}{\Delta x}\right)^2 - \frac{1}{12}. \tag{2.44}$$

To ensure the conservation of the advected quantity the coefficient $w_{0,i}$, also called weight, is taken to be the exact average of each grid cell

$$w_{0,i}^n = \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} W(t^0,x)\,\mathrm{d}x\,. \tag{2.45}$$

The sum of $w_{0,i}^n$ over all cells, that is the integral of $W(t^0,x)$ over the whole computational domain Ω, is kept constant over all time steps. The second coefficient in (2.41) is an approximation to the first derivative

$$w_{x,i}^n = \frac{\int_{x_{i-1/2}}^{x_{i+1/2}} W(t^0,x)\left(\frac{x-x_i}{\Delta x}\right)\mathrm{d}x}{\int_{x_{i-1/2}}^{x_{i+1/2}} \left(\frac{x-x_i}{\Delta x}\right)^2 \mathrm{d}x} \tag{2.46}$$

and the third one is given by

$$w_{xx,i}^n = 2\frac{\int_{x_{i-1/2}}^{x_{i+1/2}} W(t^0,x)\left(\left(\frac{x-x_i}{\Delta x}\right)^2 - \frac{1}{12}\right)\mathrm{d}x}{\int_{x_{i-1/2}}^{x_{i+1/2}} \left(\left(\frac{x-x_i}{\Delta x}\right)^2 - \frac{1}{12}\right)^2 \mathrm{d}x}. \tag{2.47}$$

Note that $w_{xx,i}^n$ is an approximation to the second derivative of the given distribution at time n.

The coefficients w_0^n, w_x^n and w_{xx}^n are independent of each other and need to be stored for each time step. After replacing the initial distribution $W(t^0,x)$ with a polynomial $w(t^0,x)$ we can determine the exact solution of w after the first time step at the time t^1, that is

$$W(t^1,x) = w(t^0, x - \sigma\Delta x), \tag{2.48}$$

where σ is the Courant number. The new distribution $W(t^1,x)$ at time t^1 is used as a new initial distribution and we can repeat the first step of the algorithm, i.e. the projection step that yields the polynomial $w(t^n,x)$ for $n = 1$.

In the following, we show the projection formulas for the time step $n+1$. Again, all coefficients have to be computed separately. For w_0 we discuss all steps in detail, for w_x and $w_x x$ the final results are given. The function $W(t^{n+1},x)$ is used as initial

distribution that is to be projected onto P^2. It is replaced by the shifted polynomials according to (2.48).

$$w_{0,i}^{n+1} = \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} W(t^{n+1}, x) k_0^{(i)}(x)\, \mathrm{d}x \tag{2.49}$$

$$= \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} w(t^n, x - \sigma\Delta x) k_0^{(i)}(x)\, \mathrm{d}x\,. \tag{2.50}$$

The next step is substituting the argument of w by $\eta = x - \sigma\Delta x$. Then, we devide the integral into two pieces, such that one integral is contained in cell $i-1$ and the other one in the ith cell.

$$w_{0,i}^{n+1} = \frac{1}{\Delta x} \int_{x_{i-1/2}-\sigma\Delta x}^{x_{i+1/2}-\sigma\Delta x} w(t^n, \eta) k_0^{(i)}(\eta + \sigma\Delta x)\, \mathrm{d}\eta \tag{2.51}$$

$$\begin{aligned} &= \frac{1}{\Delta x}\Bigg(\int_{x_{i-1/2}-\sigma\Delta x}^{x_{i-1/2}} w(t^n, \eta) k_0^{(i)}(\eta + \sigma\Delta x)\, \mathrm{d}\eta \\ &\quad + \int_{x_{i-1/2}}^{x_{i+1/2}-\sigma\Delta x} w(t^n, \eta) k_0^{(i)}(\eta + \sigma\Delta x)\, \mathrm{d}\eta \Bigg). \end{aligned} \tag{2.52}$$

Last, the cellwise defined polynomials can be plugged into the equations

$$\begin{aligned} w_{0,i}^{n+1} = \frac{1}{\Delta x}\Bigg(&\int_{x_{i-1/2}-\sigma\Delta x}^{x_{i-1/2}} \Big(w_{0,i-1}^n\, k_0^{(i-1)}(\eta) + w_{x,i-1}^n\, k_1^{(i-1)}(\eta) \\ &+ \frac{1}{2} w_{xx,i-1}^n\, k_2^{(i-1)}(\eta) \Big) k_0^{(i)}(\eta + \sigma\Delta x)\, \mathrm{d}\eta \\ &+ \int_{x_{i-1/2}}^{x_{i+1/2}-\sigma\Delta x} \Big(w_{0,i}^n k_0^{(i)}(\eta) + w_{x,i}^n k_1^{(i)}(\eta) \\ &+ \frac{1}{2} w_{xx,i}^n k_2^{(i)}(\eta) \Big) k_0^{(i)}(\eta + \sigma\Delta x)\, \mathrm{d}\eta \Bigg), \end{aligned} \tag{2.53}$$

and be easily integrated

$$\begin{aligned} w_{0,i}^{n+1} = {} & \sigma\, w_{0,i-1}^n + (1-\sigma)\, w_{0,i}^n + \left(\frac{1}{2}\sigma^2 - \frac{1}{2}\sigma \right) w_{x,i}^n \\ & + \left(-\frac{1}{2}\sigma^2 + \frac{1}{2}\sigma \right) w_{x,i-1}^n + \left(+\frac{1}{6}\sigma^3 - \frac{1}{4}\sigma^2 + \frac{1}{12}\sigma \right) w_{xx,i-1}^n \\ & + \left(-\frac{1}{6}\sigma^3 + \frac{1}{4}\sigma^2 - \frac{1}{12}\sigma \right) w_{xx,i}^n. \end{aligned} \tag{2.54}$$

Similarly, $w_{x,i}^{n+1}$ is determined

$$w_{x,i}^{n+1} = \frac{12}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} W(t^1, x) k_1^{(i)}(x)\, \mathrm{d}x \tag{2.55}$$

$$\begin{aligned} &= \left(6\sigma^2 - 6\sigma\right) w_{0,i-1} + \left(-6\sigma^2 + 6\sigma\right) w_{0,i} \\ &\quad + \left(-2\sigma^3 + 6\sigma^2 - 3\sigma\right) w_{x,i-1} + \left(2\sigma^3 - 3\sigma + 1\right) w_{x,i} \\ &\quad + \left(\frac{1}{2}\sigma^4 - 2\sigma^3 + 2\sigma^2 - \frac{1}{2}\sigma\right) w_{xx,i-1} \\ &\quad + \left(-\frac{1}{2}\sigma^4 + \sigma^2 - \frac{1}{2}\sigma\right) w_{xx,i}, \end{aligned} \tag{2.56}$$

as well as the coefficient $w_{xx,i}^{n+1}$

$$w_{xx,i}^{n+1} = \frac{360}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} W(t^1, x)\, k_2^{(i)}(x)\, \mathrm{d}x \tag{2.57}$$

$$\begin{aligned} &= \left(120\sigma^3 - 180\sigma^2 + 60\sigma\right) w_{0,i-1} \\ &\quad + \left(-120\sigma^3 + 180\sigma^2 - 60\sigma\right) w_{0,i} \\ &\quad + \left(-30\sigma^4 + 120\sigma^3 - 120\sigma^2 + 30\sigma\right) w_{x,i-1} \\ &\quad + \left(30\sigma^4 - 60\sigma^2 + 30\sigma\right) w_{x,i} \\ &\quad + \left(6\sigma^5 - 30\sigma^4 + 50\sigma^3 - 30\sigma^2 + 5\sigma\right) w_{xx,i-1} \\ &\quad + \left(-6\sigma^5 + 10\sigma^3 - 5\sigma + 1\right) w_{xx,i}. \end{aligned} \tag{2.58}$$

The property to conserve the advected quantity over all time can be checked by looking at the integral over the whole domain at time t^{n+1}

$$\int_\Omega w(t^{n+1}, x)\, \mathrm{d}x = \sum_{i=1}^{N} w_{0,i}^{n+1} = \sum_{i=1}^{N} w_{0,i}^{n} = \cdots = \sum_{i=1}^{N} w_{0,i}^{0}. \tag{2.59}$$

The sum of all coefficients $w_{0,i}$ remains the same over time. The other terms do not influence the integral over Ω, because of the orthogonality of the basis functions.

The discussion of the accuracy of the scheme is skipped here. Van Leer only examines the accuracy of the second-order version scheme III. The third- order scheme VI is equal to the SASLDG method (which will be developed in Chapter 3) if the velocity of the advection equation is constant as assumed here. Therefore, the accuracy is thoroughly examined in Chapter 5.

To achieve monotonicity of the scheme van Leer proposes to limit the coefficient $w_{x,i}$, that approximates the slope of the numerical solution within a grid cell. He suggests to manipulate the slope in such a way, that the function values $w(t^n, x_{i-1/2})$ and $w(t^n, x_{i+1/2})$ at the left and right boundary of the ith cell remain between the mean values of the neighboring cells $w_{0,i-1}$ and $w_{0,i+1}$,

$$[w_{x,i}]^{\mathrm{MON}} = \begin{cases} \min\left(2|w_{0,i} - w_{0,i-1}|, |w_{x,i}|, 2|w_{0,i+1} - w_{0,i}|\right)\mathrm{sgn}(w_{x,i}) \\ \qquad \text{if } \mathrm{sgn}(w_{0,i} - w_{0,i-1}) = \mathrm{sgn}\, w_{x,i} = \mathrm{sgn}(w_{0,i+1} - w_{0,i}) \\ 0 \quad \text{otherwise.} \end{cases} \tag{2.60}$$

This slope limiting also ensures positive definiteness, if the initial values are positive. The monotonicity property is only developed for the second-order schemes. Van

Leer does not discuss the possibility to manipulate the coefficient $w_{xx,i}$ and thus influence the curvature of the numerical solution.

The extension to solve equations of compressible flow is discussed in the sequel [32] of the paper series. The resulting method is the Monotone Upstream Scheme for Conservation Laws (MUSCL), also described with a further development in [33]. The MUSCL scheme uses a different idea to propagate the numerical solution in time. In the conclusion of [32] van Leer states that a worthwile improvement of the method could be an incorporation of the least-squares fitting. Because of the deviation from the original idea the extension to variable velocity coefficient is not followed up here.

2.3 SECOND-ORDER MOMENTS METHOD

The second-order moments (SOM) method developed by Prather in [43] is identical to the scheme of van Leer as shown below. Thus, the idea to shift given polynomial distributions according to the exact solution, and then use a projection step to recover polynomials for each cell, is the same as van Leers. However, the way to reach the same result differs a bit as Prather does some explicit extra calculations that demonstrate the details of the scheme.

2.3.1 The algorithm

We assume to begin with a given piecewise polynomial distribution, with the maximum degree of two, otherwise a projection onto the polynomial space $P^2(\Omega_i)$ will lead to that. The polynomial distributions are given for each grid cell

$$f(x) = m_0 K_0(x) + m_x K_1(x) + m_{xx} K_2(x). \tag{2.61}$$

The basis for the representation of the polynomials is the following orthogonal basis for the interval $[0, \Delta x]$

$$K_0(x) = 1, \tag{2.62}$$
$$K_1(x) = x - \frac{\Delta x}{2}, \tag{2.63}$$
$$K_2(x) = x^2 - \Delta x x + \frac{1}{6}\Delta x^2. \tag{2.64}$$

Now, in contrast to the method of van Leer the piecewise polynomial distribution is not propagated directly and as a whole according to the analytical solution, but it is decomposed into pieces as a first step. For each grid cell it is determined which part of the distribution is advected to the neighboring grid cell and which part remains within the cell. Note, that the method is only intended for CFL numbers less than unity. If we assume u being positive, the right part $V^R = [\Delta x - u\Delta t, \Delta x]$ is the one that is shifted to the right adjacent cell and the left part $V^L = [0, \Delta x - u\Delta t]$ stays in the cell. Then, the polynomial is projected onto those two parts. We define the basis functions for V^L

$$K_0^L = 1, \tag{2.65}$$
$$K_1^L = x - \frac{\Delta x}{2}(1 - \sigma), \tag{2.66}$$
$$K_2^L = x^2 - \Delta x(1 - \sigma)x + \frac{\Delta x^2}{6}(1 - \sigma)^2, \tag{2.67}$$

and the basis functions for V^R given by

$$K_0^R = 1, \tag{2.68}$$

$$K_1^R = x - \frac{\Delta x}{2}(2-\sigma), \tag{2.69}$$

$$K_2^R = x^2 - \Delta x(2-\sigma)x + (1-\sigma)\Delta x^2 + \frac{\Delta x^2}{6}\sigma^2, \tag{2.70}$$

where σ is the CFL number. The decomposition is done via a L^2-projection $\mathbb{P}^L$ onto the subintervals V^L

$$\mathbb{P}^L f(x) := \sum_{i=0}^{2} \frac{\int_0^{\Delta x - u\Delta t} f(x)K_i^L(x)\,\mathrm{d}x}{\int_0^{\Delta x - u\Delta t} K_i^L(x)K_i^L(x)\,\mathrm{d}x} K_i^L(x), \tag{2.71}$$

$$= m_0^L K_0^L(x) + m_1^L K_1^L(x) + m_2^L K_2^L(x), \tag{2.72}$$

$$=: f^L(x), \tag{2.73}$$

and with the projection operator $\mathbb{P}^R$ repectively onto V^R,

$$\mathbb{P}^R f(x) := \sum_{i=0}^{2} \frac{\int_{\Delta x - u\Delta t}^{\Delta x} f(x)K_i^R(x)\,\mathrm{d}x}{\int_{\Delta x - u\Delta t}^{\Delta x} K_i^R(x)K_i^R(x)\,\mathrm{d}x} K_i^R(x), \tag{2.74}$$

$$= m_0^R K_0^R(x) + m_1^R K_1^R(x) + m_2^R K_2^R(x), \tag{2.75}$$

$$=: f^R(x). \tag{2.76}$$

Hence, we obtain new coefficients $m_j^{L,i}$ and $m_j^{R,i}$ for $j \in \{0,1,2\}$ to represent the two new polynomials $f^{L,i}(x)$ and $f^{R,i}(x)$ for each grid cell. For the left subinterval V^L we have

$$m_0^{L,i} = m_0 - m_1\frac{1}{2}(u\Delta t) + m_2\frac{1}{6}u\Delta t(2u\Delta t - \Delta x), \tag{2.77}$$

$$m_1^{L,i} = m_1 - m_2(u\Delta t), \tag{2.78}$$

$$m_2^{L,i} = m_2, \tag{2.79}$$

and further the coefficients for $f^{R,i}(x)$ are given by

$$m_0^{R,i} = m_0 + m_1\frac{1}{2}(\Delta x - u\Delta t) + m_2\frac{1}{6}(\Delta x - u\Delta t)(\Delta x - 2u\Delta t), \tag{2.80}$$

$$m_1^{R,i} = m_1 + m_2(\Delta x - u\Delta t), \tag{2.81}$$

$$m_2^{R,i} = m_2. \tag{2.82}$$

Note, that the graph of the function f does not change during the projection step. Only the representation has changed.

In the next step the coefficients are moved either to the neighboring cell or just within the cell. Specifically, in this method that means that all $m_j^{R,i}$ of cell i become $m_j^{L,i+1}$ of the adjacent cell $i+1$ ($m_j^{R,i} \to m_j^{L,i+1}$), and all $m_j^{L,i}$ become $m_j^{R,i}$ within the ith cell ($m_j^{L,i} \to m_j^{R,i}$). The basis function for V^L and V^R defined in (2.77) - (2.77) still hold, but their length has to be adapted. This is achieved by replacing σ by $1-\sigma$.

The step of shifting the coefficients corresponds to the propagation of the distribution of the polynomials in van Leers method in (2.48). The piecewise polynomial distributions on the subintervals are moved using the knowledge of the exact solution. Generally, the function based on the shifted coefficients in each grid cell is not

continuous. Another projection step is needed to retain a polynomial for each cell. The basis functions (2.62) - (2.64) are used to project the piecewise polynomials onto the interval $[0, \Delta x]$

$$\mathbb{P}f = \sum_{i=0}^{2} \frac{\int_0^{\Delta x} f(x) K_i(x)\,\mathrm{d}x}{\int_0^{\Delta x} K_i(x) K_i(x)\,\mathrm{d}x} K_i(x) \tag{2.83}$$

$$= \sum_{i=0}^{2} \frac{\int_0^{u\Delta t} \sum m_j^L K_j(x)^L K_i(x)\,\mathrm{d}x + \int_{u\Delta t}^{\Delta x} \sum m_j^R K_j^R(x) K_i(x)\,\mathrm{d}x}{\int_0^{\Delta x} K_i(x) K_i(x)\,\mathrm{d}x} K_i(x) \tag{2.84}$$

$$= \sum_{k=0}^{2} m_i K_i(x). \tag{2.85}$$

Figure 2.1 illustrates the main steps and the idea of the SOM method. By means of exemplarily three grid cells it shows one advection step with the substeps. A Courant number of 0.25 is chosen for this example. The initial piecewise polynomial distribution is shown in Figure 2.1(a). The function is is continuous within a grid cell. Figure 2.1(b) illustrates the decomposition of the polynomial into two parts $f^L(x)$ (colored in red) and $f^R(x)$ (colored in blue) corresponding to (2.73) and (2.76), respectively. The actual advection of the distribution is pictured in Figure 2.1(c). The polynomials f^L, formerly located at the left side of each grid cell, are moved to the right, and the polynomials f^R are advected into the right adjacent grid cell. The result of the final projection step (2.85) is shown in Figure 2.1(d). Each grid cell consists of one continuous polynomial.

For the representation of the coefficients for the polynomials Prather chooses to weigh the coefficients in such a way, that they have the same physical unit, that is the mass of tracer. That leads to

$$f(x) = m_0 + m_1 \left(x - \frac{\Delta x}{2}\right) + m_2 \left(x^2 - \Delta x\, x + \frac{\Delta x^2}{6}\right) \tag{2.86}$$

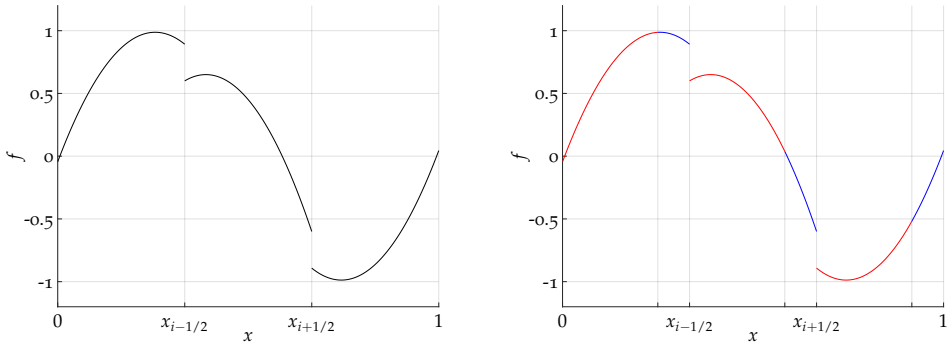
$$= \frac{S_0}{\Delta x} + 2\frac{S_x}{\Delta x^2} \left(x - \frac{\Delta x}{2}\right) + 6\frac{S_{xx}}{\Delta x^3} \left(x^2 - \Delta x\, x + \frac{\Delta x^2}{6}\right). \tag{2.87}$$

Thus S_0 stands for the zeroth moment, S_x for the first moment and S_{xx} for the second moment. The resulting coefficients from the final projection step in (2.85) in converted form are given by

$$\begin{aligned} S_0 = {} & \sigma S_{0,i-1} + (1-\sigma) S_{0,i} + \left(-\sigma^2 + \sigma\right) S_{x,i-1} + \left(\sigma^2 - \sigma\right) S_{x,i} \\ & + \left(2\sigma^3 - 3\sigma^2 + \sigma\right) S_{xx,i-1} + \left(-2\sigma^3 + 3\sigma^2 - \sigma\right) S_{xx,i}, \end{aligned} \tag{2.88}$$

$$\begin{aligned} S_x = {} & \left(3\sigma^2 - 3\sigma\right) S_{0,i-1} + \left(-3\sigma^2 + 3\sigma\right) S_{0,i} \\ & + \left(-2\sigma^3 + 6\sigma^2 - 3\sigma\right) S_{x,i-1} + \left(2\sigma^3 - 3\sigma + 1\right) S_{x,i} \\ & + \left(3\sigma^4 - 12\sigma^3 + 12\sigma^2 - 3\sigma\right) S_{xx,i-1} \\ & + \left(-3\sigma^4 + 6\sigma^2 - 3\sigma\right) S_{xx,i}, \end{aligned} \tag{2.89}$$

$$\begin{aligned} S_{xx} = {} & \left(10\sigma^3 - 15\sigma^2 + 5\sigma\right) S_{0,i-1} + \left(-10\sigma^3 + 15\sigma^2 - 5\sigma\right) S_{0,i} \\ & + \left(-5\sigma^4 + 20\sigma^3 - 20\sigma^2 + 5\sigma\right) S_{x,i-1} + \left(5\sigma^4 - 10\sigma^2 + 5\sigma\right) S_{x,i} \\ & + \left(6\sigma^5 - 30\sigma^4 + 50\sigma^3 - 30\sigma^2 + 5\sigma\right) S_{xx,i-1} \\ & + \left(-6\sigma^5 + 10\sigma^3 - 5\sigma + 1\right) S_{xx,i}, \end{aligned} \tag{2.90}$$



(a) quadratic polynomial for each grid cell.

(b) the polynomials are decomposed in a left and a right part.

(c) left and right parts are advected

(d) assembling of the new quadratic polynomial

Figure 2.1: Construction of the Second Order Moments method.

where σ is the CFL number. To show the equality of the method by van Leer to the results of the SOM method by Prather, we compare (2.87) with (2.41), and obtain the following relation of the coefficients

$$w_0 = \frac{S_0}{\Delta x} \tag{2.91}$$

$$w_x = 2\frac{S_x}{\Delta x} \tag{2.92}$$

$$w_{xx} = 12\frac{S_{xx}}{\Delta x}. \tag{2.93}$$

When we plug in these transformations in (2.88), (2.89), and (2.90) the equality of these two numerical methods can easily be seen.

2.3.2 Limiting

Prather proposes the following optional limits on the first and second moment to maintain a positive distribution in each cell,

$$\begin{aligned} [S_x]^{\text{MON}} &= \min\left(\frac{3}{2}S_0, \max\left(-\frac{3}{2}S_0, S_x\right)\right) \\ [S_{xx}]^{\text{MON}} &= \min\left(2S_0 - \frac{\left|[S_x]^{\text{MON}}\right|}{3}, \max\left(\left|[S_x]^{\text{MON}}\right| - S_0, S_{xx}\right)\right). \end{aligned} \tag{2.94}$$

To obtain these restrictions on the higher moments the polynomial distributions are cellwise examined, minima are located and from these the restrictions on the

higher order moments are determined. The limits (2.94) are computed in a such a way, that the local minimum for each cell is kept positive and therefore a positive tracer distribution is guaranteed within the cell.

The limiter given above is constructed in such a way to keep the numerical solution positive, i.e. above the limit zero. However, it can be useful to limit the solution from above as well and possibly from below by a different bound than zero. In [40] we found a simply way to first define the appropriate bounds and second limit the first moment, i.e. the slope, such that the numerical solution is kept between these bounds.

The bounds are defined by the values of the neighboring grid cells,

$$\begin{aligned} B_{\max} &= \max(m_{0,i-2}^{n+1}, m_{0,i-1}^{n+1}, m_{0,i}^{n+1}, m_{0,i+1}^{n+1}, m_{0,i+2}^{n+1}) \\ B_{\min} &= \min(m_{0,i-2}^{n+1}, m_{0,i-1}^{n+1}, m_{0,i}^{n+1}, m_{0,i+1}^{n+1}, m_{0,i+2}^{n+1}). \end{aligned} \tag{2.95}$$

The limits are given by

$$\begin{aligned} m_{1,i}^{\text{inter}} &= \min(B_{\max} - m_{0,i}^{n+1}, m_{0,i}^{n+1} - B_{\min}) \\ m_{1,i}^{\text{limited}} &= \min(m_{1,i}^{\text{inter}}, \max(-m_{1,i}^{\text{inter}}, m_{1,i}^{n+1})). \end{aligned} \tag{2.96}$$

This limiter is more restrictive than the one suggested by Prather as discussed in [40], but has the advantage to limit the numerical solution from above and below.

2.3.3 Variable velocity

The handling of a variable velocity coefficient in the advection equation of the SOM method is not explicitly described in [43]. It can be deduced from a Fortran implementation written by Prather that is available online on [42]. The comments stated below refer to that version.

First of all, there is a difference of the Fortran code to the analytical description in [43] that holds for variable and for constant velocity coefficient. Each advection step consists of a two step procedure in the Fortran code. First, the tracer is advected from all uneven grid cells to all even grid cells, and second, from all even to all uneven grid cells. This code structure causes unwanted effects as a tracer is advected not only to the next neighboring cell, but in addition to the one after that, with a Courant number of less than unity. In particular, it is striking in the case with the initial distribution of positive tracer in the first grid cell and in the second and third grid cell the tracer set to zero. After one time step with a CFL number of less than unity and positive velocity we expect the tracer to be also positive in the second grid cell and still be equal to zero in the third cell. However, with the two step procedure of Prather's implementation the tracer is also advected to the third grid cell within one advection step.

In the case of constant velocity the variable representing the density is equal to one over all time. If the velocity u changes in space, Prather approximates the velocity field assuming a piecewise constant velocity for each grid cell, i.e. u_i for the ith cell. In his algorithm the density variable ρ is adapted accordingly, thus during a time step with positive velocity the fraction $u_{i+1}\Delta t/\Delta x$ is subtracted from the density variable ρ_i and the fraction $u_i\Delta t/\Delta x$ is added to it. This approach is problematic. First, a procedure like that does not compute the correct evolution in time of the density, which should happen in order to solve the set of equations of (1.4) and (1.5). Second, the density variable for a certain grid cell is increased or decreased by the same amount for each time step. That means that in a one dimensional setting the density grows or drops unlimited for that cell. It can reach negative values, which is physically not possible. The same problem occurs in more space dimensions unless the velocity field is divergence-free as then the amounts that are added and subtracted add up to zero. In general, to solve the advection equation for the density and the tracer density with different numerical methods, as done in the SOM method, can lead to inconsistencies, see [30].

For these reasons, we implement and extend the idea of the SOM method with three main differences to Prather's Fortran code. First, the advection step is done within one single step as described originally in [43]. Second, the velocity is assumed to be piecewise linear and not approximated by piecewise constant functions. And last but not least, the density is treated and advected in the same way as the tracer density using the same numerical method.

2.3.4 Further remarks

We shift the discussion about the accuracy of the SOM method to Chapter 5, because it is identical with the SASLDG method derived in this thesis in the case of the linear advection equation with constant velocity.

However, the accurate numerical results of the SOM method are the reason why we follow this ansatz. The method is stable for CFL numbers up to one, which is to be extended for larger time steps in the SASLDG method.

2.4 DISCONTINUOUS GALERKIN

Discontinuous Galerkin (DG) methods were first developed to solve the linear advection equation [47] and then generalized, as e.g. described in [11], to solve hyperbolic problems of the type

$$y(x,t)_t + f(y(x,t))_x = 0, \tag{2.97}$$

with both, according initial values and boundary conditions. Main characteristics of DG methods are formal high order accuracy, high parallelizability, the ability to handle complex geometries and to capture discontinuities without generating spurious oscillations, see [11]. Another asset is the option of hp-adaptivity described in [10]. It enables the local adjustment of the grid cell sizes and the degree of the polynomials.

We begin the introduction with assuming constant velocity, $f(y) = uy$, and later comment on general hyperbolic problems.

2.4.1 Discontinuous Galerkin-space discretization

In DG methods, the exact solution y is approximated by y_h, that belongs to the space V_h,

$$V_h = V_h^r \equiv \{v \in L^1(0,1) \, : v|_{\Omega_i} \in P^r(\Omega_i), i = 1, \ldots, N\}, \tag{2.98}$$

where $P^r(I)$ is the space of polynomials in I with the highest degree r.

First, the linear advection equation (1.1) is rewritten in the weak formulation by multiplication of test functions $v(x)$, and integration by parts

$$\begin{aligned} &\int_{\Omega_i} \partial_t y(x,t) v(x) \, \mathrm{d}x - \int_{\Omega_i} uy(x,t) \partial_x v(x) \, \mathrm{d}x \\ &\quad + uy(x_{i+1/2},t) v(x_{i+1/2}^-) - uy(x_{i-1/2},t) v(x_{i-1/2}^+) = 0, \end{aligned} \tag{2.99}$$

$$\int_{\Omega_i} y(x,0) v(x) \, \mathrm{d}x = \int_{\Omega_i} y_0(x) v(x) \, \mathrm{d}x \,. \tag{2.100}$$

The notation of $x^+_{i-1/2}$ or $x^-_{i-1/2}$ refers to the one-sided limit approaching the point $x_{i-1/2}$ from the right or from the left side, respectively. Then, the functions y and v are discretized by functions of V_h,

$$\begin{aligned}&\int_{\Omega_i} \partial_t y_h(x,t) v_h(x)\,\mathrm{d}x - \int_{\Omega_i} u y_h(x,t)\partial_x v_h(x)\,\mathrm{d}x \\ &\quad + h(y_h)_{i+1/2}(t) v_h(x^-_{i+1/2}) - h(y_h)_{i-1/2}(t) v_h(x^+_{i-1/2}) = 0,\end{aligned} \tag{2.101}$$

$$\int_{\Omega_i} y_h(x,0) v_h(x)\,\mathrm{d}x = \int_{\Omega_i} y_0(x) v_h(x)\,\mathrm{d}x \tag{2.102}$$

which must hold for all $i = 1,\dots,N$ and for all $v_h \in P^r(\Omega_i)$. The numerical flux h is introduced, because the function y_h is discontinuous at the cell boundaries $x_{i+1/2}$. The function is determined from both neighboring values,

$$h(u)_{i+1/2}(t) = h(y(x^-_{i+1/2},t), y(x^+_{i+1/2},t)). \tag{2.103}$$

The selection of the numerical flux plays a role in the case of nonlinear problems, as we will show later. However, in the special case of constant velocity, i.e. $f(y) = uy$, it is given by

$$h(a,b) = u\,\frac{a+b}{2} - \frac{|u|}{2}(b-a). \tag{2.104}$$

Legendre polynomials P_j are chosen to describe the polynomials in the space V_h. They have the property to be L^2-orthogonal,

$$\int_{-1}^{1} P_j(s)\,P_{j'}(s)\,ds = \left(\frac{2}{2j+1}\right)\delta_{jj'}, \tag{2.105}$$

with the jth basis function given by

$$k_j(x) = P_j\left(\frac{2(x-x_i)}{\Delta x_i}\right). \tag{2.106}$$

The approximated solution y_h in the ith grid cell is of the form

$$y_h(x,t) = \sum_{j=0}^{r} y_i^j k_j(x). \tag{2.107}$$

Taking (2.101) and (2.102) and using the orthogonality of the basis functions we obtain the DG-space discretization for the linear advection equation for all grid cells $i = 1,\dots,N$ and for all degrees of freedom of the polynomial space $j = 0,\dots,r$

$$\begin{aligned}&\left(\frac{\Delta x_i}{2j+1}\right)\partial_t y_i^j(t) - \int_{\Omega_i} u y_h(x,t)\partial_x k_j(x)\;\mathrm{d}x \\ &\quad + \Big(h(y_h(x_{i+1/2}))(t) - (-1)^j h(y_h(x_{i-1/2}))(t)\Big) = 0,\end{aligned} \tag{2.108}$$

$$y_i^j(0) = \frac{2j+1}{\Delta x_i}\int_{\Omega_i} y_0(x) k_j(x)\;\mathrm{d}x\,. \tag{2.109}$$

Finally, (2.108) can be expressed as a system of ODEs

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t} y_h = L_h(y_h), \quad \text{in} \quad (0,T), \\ &y_h(t_0) = y_h(0),\end{aligned} \tag{2.110}$$

where $L_h(y_h)$ is the approximation to $-uy_x$ in V_h. The ODEs can be solved using a numerical time integrator such as Runge-Kutta methods, which is shown in the next section 2.4.2. Alternatively, (2.110) can be solved analytically. This is done in Section 2.4.3 to show the equivalence with the SOM method and van Leer's method, respectively.

2.4.2 Runge-Kutta Discontinuous Galerkin

The field of Runge-Kutta (RK) methods - and also of Runge-Kutta Discontinuous Galerkin (RKDG) methods - is very broad, we outline the idea of RK methods briefly, and name only three different schemes.

In a general setting of Runge-Kutta methods the numerical solution y_h^{n+1} at time t^{n+1} is given by the following procedure. We set $y_h^{(0)} = y_h^n$, then the time step is computed for intermediate time steps $i = 1, \dots, r+1$,

$$y_h^{(i)} = \sum_{l=0}^{i-1} \alpha_{il} y_h^{(l)} + \beta_{il} \Delta t^n L_h\left(u_h^l\right), \tag{2.111}$$

with the last step to set $y_h^{n+1} = y_h^{(r+1)}$. Possible choices for the coefficients α_{il} and β_{il} are e.g. listed in [54]. A first order method is the forward Euler method. The solution at t^{n+1} is obtained in one step

$$y_h^{n+1} = y_h^n + \Delta t L_h(y_h^n), \tag{2.112}$$

without any intermediate steps. The second-order Runge-Kutta method (also called Heun's method) is given by

$$\begin{aligned} y_h^{(1)} &= y_h^n + \Delta t L_h(y_h^n) \\ y_h^{n+1} &= \frac{1}{2} y_h^n + \frac{1}{2} y_h^{(1)} + \frac{1}{2} \Delta t L_h(y_h^{(1)}). \end{aligned} \tag{2.113}$$

We conclude the examples with a Runge-Kutta method introduced in [54], that is of order three,

$$\begin{aligned} y_h^{(1)} &= y_h^n + \Delta t L_h(y_h^n) \\ y_h^{(2)} &= \frac{3}{4} y_h^n + \frac{1}{4} y_h^{(1)} + \frac{1}{4} \Delta t L_h(y_h^{(1)}) \\ y_h^{n+1} &= \frac{1}{3} y_h^n + \frac{2}{3} y_h^{(2)} + \frac{2}{3} \Delta t L_h(y_h^{(2)}). \end{aligned} \tag{2.114}$$

The choice of the Runge-Kutta method is of great significance for the stability of the DG method. Even in the case of linear velocity $f(y) = uy$, attention must be paid to the time integrator that is used. In [8], Chavent and Cockburn discuss the case $r = 1$, i.e. the piecewise linear approximation in the DG-space combined with the forward Euler method. The resulting DG method is unconditionally unstable for any fixed ratio $\Delta t / \Delta x$. Cockburn and Shu proved in [12] that if a Runge-Kutta method of second order is applied in the same case, the stability holds if

$$u \frac{\Delta t}{\Delta x} \leq \frac{1}{3}. \tag{2.115}$$

A certain order of Runge-Kutta methods is mandatory in order to achieve stability. Numerical experiments by Cockburn indicate that stability holds, if Runge-Kutta methods of order $r+1$ are coupled with DG-space discretizations of order r and the following condition on the size of the time step must be met

$$u \frac{\Delta t}{\Delta x} \leq \frac{1}{2r+1}. \tag{2.116}$$

Hence,the more accurate the DG method ought to be, i.e. the higher the degree of the polynomials in the DG-space, the more restrictive is the size of the time step.

Note, that all of the above mentioned Runge-Kutta methods are total variation diminishing (TVD) methods.

2.4.3 Discontinuous Galerkin with exact time integration

The goal is to solve the ODEs in (2.110) analytically in time in order to avoid errors of numerical schemes such as RK methods. We start with (2.108) and add the time integral. We obtain,

$$\underbrace{\int_{t^n}^{t^{n+1}} \frac{\Delta x_i}{2j+1} \partial_t y_i^j(t)\,\mathrm{d}t}_{(i)} - \underbrace{\int_{t^n}^{t^{n+1}} \int_{\Omega_i} u y_h(x,t) \partial_x k_j(x)\;\mathrm{d}x\,\mathrm{d}t}_{(ii)} + \underbrace{\int_{t^n}^{t^{n+1}} \Big(h(y_h(x_{i+1/2}))(t) - (-1)^j h(y_h(x_{i-1/2}))(t) \Big)\,\mathrm{d}t}_{(iii)} = 0, \tag{2.117}$$

and proceed to examine each term individually. The exact time integration of the expression (i) is straight forward,

$$\int_{t^n}^{t^{n+1}} \frac{\Delta x_i}{2j+1} \partial_t y_i^j(t)\,\mathrm{d}t = \frac{\Delta x_i}{2j+1} \Big(y_i^j(t^{n+1}) - y_i^j(t^n) \Big). \tag{2.118}$$

We rewrite the integral of expression (ii) with new integration limits $[0, \Delta t]$. Then, we split the integral in a left part such that $y_h \in [x_{i-3/2}, x_{i-1/2}]$ for the whole time step and respectively a right part. We have,

$$\begin{aligned} &\int_{t^n}^{t^{n+1}} \int_{x_{i-1/2}}^{x_{i+1/2}} u y_h(x,t) \partial_x k_j(x)\,\mathrm{d}x\,\mathrm{d}t \\ &\quad = \int_0^{\Delta t} \int_{x_{i-1/2}}^{x_{i+1/2}} u y_h(x, t^n + \tau) \partial_x k_j(x)\,\mathrm{d}x\,\mathrm{d}\tau \end{aligned} \tag{2.119}$$

$$\begin{aligned} &= \int_0^{\Delta t} \int_{x_{i-1/2}}^{x_{i-1/2}+u\tau} u y_h(x, t^n + \tau) \partial_x k_j(x)\,\mathrm{d}x\,\mathrm{d}\tau \\ &\quad + \int_0^{\Delta t} \int_{x_{i-1/2}+u\tau}^{x_{i+1/2}} u y_h(x, t^n + \tau) \partial_x k_j(x)\,\mathrm{d}x\,\mathrm{d}\tau\,. \end{aligned} \tag{2.120}$$

The analytical solution of the linear advection equation (1.2) can be used to enable evaluation of y_h only at time level t^n, where the solution is known. Further, the

polynomial form of $y_h(x,t)$ (2.107) is plugged in. With that the term (ii) takes the form

$$\begin{aligned}
&\int_{t^n}^{t^{n+1}} \int_{x_{i-1/2}}^{x_{i+1/2}} u y_h(x,t)\partial_x k_j(x)\,\mathrm{d}x\,\mathrm{d}t \\
&\quad= \int_0^{\Delta t} \int_{x_{i-1/2}}^{x_{i-1/2}+u\tau} u y_h(x-u\tau,t^n)\partial_x k_j(x)\,\mathrm{d}x\,\mathrm{d}\tau \qquad (2.121)\\
&\qquad+ \int_0^{\Delta t} \int_{x_{i-1/2}+u\tau}^{x_{i+1/2}} u y_h(x-u\tau,t^n)\partial_x k_j(x)\,\mathrm{d}x\,\mathrm{d}\tau \\
&\quad= \sum_{l=0}^{r} u y_{i-1}^l(t^n) \int_0^{\Delta t} \int_{x_{i-1/2}}^{x_{i-1/2}+u\tau} k_{i-1}^l(x-u\tau)\partial_x k_j^{(i)}(x)\,\mathrm{d}x\,\mathrm{d}\tau \\
&\qquad+ u y_i^l(t^n) \int_0^{\Delta t} \int_{x_{i-1/2}+u\tau}^{x_{i+1/2}} k_i^l(x-u\tau,t^n)\partial_x k_j^{(i)}(x)\,\mathrm{d}x\,\mathrm{d}\tau\,. \qquad (2.122)
\end{aligned}$$

Now we consider the final term of (2.117). We assume a positive velocity u, thus, the numerical flux (2.103) equals $h(y_h(x_{i+1/2})) = u y_h(x_{i+1/2}^-,t)$. Then, the term (iii) yields

$$\begin{aligned}
&\int_{t^n}^{t^{n+1}} \Big(h(y_h(x_{i+1/2}))(t) - (-1)^j h(y_h(x_{i-1/2}))(t)\Big)\,\mathrm{d}t \\
&\quad= \int_{t^n}^{t^{n+1}} u y_h(x_{i+1/2}^-,t) - (-1)^j u y_h(x_{i-1/2}^-,t)\,\mathrm{d}t
\end{aligned} \qquad (2.123)$$

Further, the known analytical solution (1.2) to the linear advection equation and integration by substitution can be used to express an integral of y_h in time as a integral in space

$$\begin{aligned}
\int_0^{\Delta t} u y(x_{i+1/2},t^n+\tau)\,\mathrm{d}\tau &= \int_0^{\Delta t} u y(x_{i+1/2}-u\tau,t^n)\,\mathrm{d}\tau \qquad (2.124)\\
&= \int_{x_{i+1/2}}^{x_{i+1/2}-u\Delta t} u y(x,t)\left(-\frac{1}{u}\right)\mathrm{d}x \\
&= \int_{x_{i+1/2}-u\Delta t}^{x_{i+1/2}} y(x,t)\,\mathrm{d}x\,. \qquad (2.125)
\end{aligned}$$

This result can be applied to term (iii). Also, the polynomial form (2.107) is plugged in. Then, the expression is equal to

$$\begin{aligned}
&\int_{t^n}^{t^{n+1}} u y_h(x_{i+1/2}, t) - (-1)^j u y_h(x_{i-1/2}, t)\,\mathrm{d}t \\
&= \int_{x_{i+1/2}-u\Delta t}^{x_{i+1/2}} y_h(x, t^n)\,\mathrm{d}x - (-1)^j \int_{x_{i-1/2}-u\Delta t}^{x_{i-1/2}} y_h(x, t^n)\,\mathrm{d}x \qquad (2.126)\\
&= \sum_{l=0}^{r} y_i^l(t^n) \int_{x_{i+1/2}-a\Delta t}^{x_{i+1/2}} k_i^l(x)\,\mathrm{d}x \\
&\quad -(-1)^j \sum_{l=0}^{r} y_{i-1}^l(t^n) \int_{x_{i-1/2}-u\Delta t}^{x_{i-1/2}} k_{i-1}^l(x)\,\mathrm{d}x \qquad (2.127)\\
&= \sum_{l=0}^{r} \left[y_i^l(t^n) - (-1)^j y_{i-1}^l(t^n) \right] \int_{x_{i+1/2}-u\Delta t}^{x_{i+1/2}} k_i^l(x)\,\mathrm{d}x\,, \qquad (2.128)
\end{aligned}$$

since the integrals of the basis functions k_i and k_{i-1} coincide on the intervals $[x_{i+1/2} - u\Delta t, x_{i+1/2}]$ and $[x_{i-1/2} - u\Delta t, x_{i-1/2}]$, respectively. Altogether, the reformulated terms (i), (ii) and (iii) yield for $j = 0, \ldots, r$ and $i = 1, \ldots, N$,

$$\begin{aligned}
y_i^j(t^{n+1}) &= y_i^j(t^n) \\
&\quad + \frac{2j+1}{\Delta x} \Bigg(\sum_{l=0}^{r} u y_{i-1}^l(t^n) \int_0^{\Delta t} \int_{x_{i-1/2}}^{x_{i-1/2}+u\tau} k_{i-1}^l(x - u\tau) \partial_x k_j^{(i)}(x)\,\mathrm{d}x\,\mathrm{d}\tau \\
&\quad + u y_i^l(t^n) \int_0^{\Delta t} \int_{x_{i-1/2}+u\tau}^{x_{i+1/2}} k_i^l(x - u\tau, t^n) \partial_x k_j^{(i)}(x)\,\mathrm{d}x\,\mathrm{d}\tau \qquad (2.129)\\
&\quad - \sum_{l=0}^{r} \left[y_i^l(t^n) - (-1)^j y_{i-1}^l(t^n) \right] \int_{x_{i+1/2}-u\Delta t}^{x_{i+1/2}} k_i^l(x)\,\mathrm{d}x \Bigg).
\end{aligned}$$

The LHS of (2.129) yields the coefficients of the numerical solution at time t^{n+1}. The RHS consists of coefficients of the time level t^n only and the basis functions. The integrals can be computed and the solution is determined. If $r = 2$, the numerical solution in (2.129) equals exactly the solution of the SOM method (apart from a change of the basis) and van Leer's method.

2.4.4 Variable velocity

If we want to use DG methods to solve the more general hyperbolic problem (2.97), the numerical flux (2.103) has to be defined differently.

To obtain a monotone scheme, the flux must be a Lipschitz and consistent function, i.e. $h(u, u) = f(u)$, and it must be a monotone flux, which is fulfilled if it is a nondeacreasing function of its first argument and a nonincresing function of its second argument.

Two numerical fluxes are exemplarily given, that satisfy these properties, that is,

(i) the Godunov flux,

$$h^G(a, b) = \begin{cases} \min_{a \le y \le b} f(y), & \text{if} \quad a \le b \\ \max_{a \ge y \ge b} f(y), & \text{if} \quad a > b \end{cases} \qquad (2.130)$$

(ii) and the Lax-Friedrichs flux,

$$h^{LF}(a,b) = \frac{1}{2}[f(a) + f(b) - C\,(b-a)], \tag{2.131}$$

$$C = \max_{\min(a,b)\le s\le \max(a,b)} |f'(s)|. \tag{2.132}$$

Cockburn notes e.g. in [9], that the impact of the choice of the numerical flux decreases with increasing degree r of the polynomial space.

Slope limiting is inevitable to ensure a stable DG method for a general hyperbolic problem. Therefore, a generalized slope limiter $\Lambda\Pi_h$ has to be applied. For piecewise linear approximations the previously introduced limiter, see (2.60), by van Leer [31] can also be applied in the setting of DG methods. The linear function y_h is of the form

$$y_h|_{I_j} = \bar{y}_j + (x - x_j)v_{x,j}, \quad j = 1, \dots, N. \tag{2.133}$$

where $\bar{y}_j$ is the mean value of y_h in the jth cell. Then, the limiter is given by

$$y_h|_{I_j} = \bar{y}_j + (x - x_j)m\left(v_{x,j}, \frac{\bar{y}_{j+1} - \bar{y}_j}{\Delta x_j}, \frac{\bar{y}_j - \bar{y}_{j-1}}{\Delta x_j}\right) \tag{2.134}$$

with the minmod function m defined as

$$m(a_1, \dots, a_\nu) = \begin{cases} s \min_{1\le n\le \nu} |a_n|, & \text{if } s = sign(a_1) = \cdots = sign(a_\nu) \\ 0, & \text{otherwise.} \end{cases} \tag{2.135}$$

For higher order polynomials with degree $r > 1$ adequate slopes can be computed from the mean values and the function values at the cell boundaries. Then, these slopes can be limited.

The limiters are applied at every RK intermediate step, i.e. (2.111) becomes

$$y_h^{(i)} = \Lambda\Pi_h\left(\sum_{l=0}^{i-1} \alpha_{il} y_h^{(l)} + \beta_{il}\Delta t^n L_h\left(u_h^l\right)\right). \tag{2.136}$$

If the slope limiter has the property to be TVD in the means, then

$$|\Lambda\Pi_h(\bar{y}_h)|_{TV(0,1)} \le |\bar{y}_h|_{TV(0,1)}. \tag{2.137}$$

Further, if such a slope limiter is coupled with a TVD RK scheme it can be shown that the resulting DGRK method is TVD in the means. However, it is well-known that for TVD method the convergence rate degenerates to first order at local extremal points. Thus, less restrictive slope limiters were developed in [53] to be only TV bounded (TVB) in the means. With the weaker property stability can still be achieved while maintaining the higher order at extrema.

2.4.5 Convergence analysis

If the initial condition is smooth enough, DG methods can reach a high order accuracy. This is formally proven in [35] for the linear case.

Theorem 2.4.1 *Suppose that the initial condition y_0 belongs to $H^{k+2}(0,1)$. Let e be the approximation error $y - y_h$. Then we have,*

$$\|e(t)\|_{L^2(0,1)} =\le C|y_0|_{H^{k+2}(0,1)}(\Delta x)^{k+1}, \tag{2.138}$$

where C depends solely on k, $|u|$ and T.

It is also possible to show convergence for the nonlinear case, even though the result is not as strong. If a certain class of generalized slope limiters is applied, the numerical solution converges to the entropy solution of the problem, for details see e.g. [11]. Note that for convergence of nonlinear problems slope limiting is essential, whereas for the linear case it it optional to achieve properties as a TVD solution.

An advantage of RKDG methods is e.g. the hp-adaptivity. DGRK can be adapted to problems in two different ways. First, polynomials of higher degree can be chosen in a selection of cells or in the whole domain as needed. And second, the grid can easily be refined. This enables RKDG methods to achieve high order accuracy. Further, RKDG method are suitable for complicated geometries. Another important property is conservation of mass, which is ensured.

An obvious downside of RKDG methods is the strong time restriction to meet stability requirements, in particular if higher order polynomials are used in the DG-discretization for high accuracy.

In this section we have shown that DG methods can be equivalent to van Leer's method and the SOM method, respectively, in case of constant velocity with analytical time integration of the ODE arising in the DG method. This knowledge enables a different point of view for the SASLDG method developed in this thesis: It can be interpreted as a DG method with exact time integration.

2.5 SEMI-LAGRANGIAN METHODS

The evolution of a traced quantity can be observed from two different points of view. One possibility is to describe the quantity from a fixed point in space, as we have done so far. The advection equation is then given in Eulerian form as in (1.3). Another way to track the evolution of a fluid particle is to follow its motion. This is expressed in Lagrangian form

$$\frac{dy}{dt} = 0. \tag{2.139}$$

Both descriptions are equivalent with the link of the Lagrangian derivative (also called material derivative) defined by

$$\frac{dy}{dt} = \frac{\partial y(x,t)}{\partial t} + u(x,t)\frac{\partial y(x,t)}{\partial x}. \tag{2.140}$$

In a classical Lagrangian numerical scheme a set of fluid parcels is traced forward in time. A problem that arises from that is the highly nonuniform distribution of the parcels after some time. Semi-Lagrangian schemes resolve this difficulty by combining the idea of the purely Lagrangian schemes with Eulerian schemes. For every time step uniformly distributed points in space are used to trace fluid parcels.

A great advantage of Semi-Lagrangian schemes is that they overcome the shortcoming of many numerical methods that are restricted to small time steps due to the stability of the numerical schemes. Semi-Lagrangian methods enable time steps of arbitrary size.

To find the solution to (2.139), we note that along trajectories φ the solution is constant in time

$$y(x,t) = y_0(\varphi(0;t,x)). \tag{2.141}$$

These trajectories are the solution to the ODE with the respective initial data at time s given by

$$\begin{aligned} \frac{\partial}{\partial t}\varphi(t;s,x) &= u(\varphi(t;s,x),t), \\ \varphi(s;s,x) &= x. \end{aligned} \tag{2.142}$$

The solution to the initial value problem exists and is unique if $u(x,t)$ is Lipschitz continuous with respect to x, and continuous with respect to t on the time interval of interest, see e.g. [24].

To obtain the final numerical solution y two problems have to be solved. First, the solution to the initial value problem (2.142) must be computed. Then, we have to approximate $c_0(\varphi(0;t,x))$ in (2.141), if we assume that we know the numerical solution y only at certain given points.

In the following two sections we will outline possible ways to tackle these problems respectively. Section 2.5.3 deals with the extension of the semi-Lagrangian approach. Then, a mass conservative method.

2.5.1 Approximate solution to initial value problem

The theory of solving initial value problems is an exhaustive research topic and discussed profoundly in literature, e.g. [23], [61], [14] to name just a few. In the context of semi-Lagrangian schemes also a broad variety of methods is developed. An iterative process is used in [51], where the initial guess is made from preceding time steps. Common ODE solvers like Runge-Kutta schemes are applied for example in [46]. Another way to find the solution is to use a Taylor series expansion for the departure point $\varphi(t^n;t^{n+1},x)$ about the arrival point $\varphi(t^{n+1};t^{n+1},x)$ along the trajectory, e.g. [39], [20].

In the case of constant velocity $u(x,t)=u$ the ODE (2.142) can be solved analytically and the solution is given in the form of the trajectory by

$$\varphi(t;s,x) = x + u(t-s). \tag{2.143}$$

Thus, we have $\varphi(0;t,x) = x - ut$ and recover the analytical solution with (2.141), compare to (1.2).

2.5.2 Interpolation techniques

In the previous subsection we listed some possibilities on how to solve the ODE (2.142). However, the numerical solution is known only at certain points in space, e.g. at the grid points. The departure points of the trajectory $\varphi(t^n;t^{n+1},x_{i-1/2})$ at time t^n, that arrive at point $(t^{ni1},x_{i-1/2})$ are in general somewhere in between these given points. To obtain the values of the numerical solution at the departure points in (2.141) we need to interpolate the solution. To do so many options are feasible. If we choose linear interpolation and limit the size of the time steps by the Courant number smaller than one, we recover the upwind method, see [5]. Results of higher order can be achieved by applying for example quadratic Lagrange [2], cubic Lagrange [52] or cubic spline [3], [45] interpolation. A detailed discussion of properties like accuracy, stability and complexity of the resulting methods can be found e.g. in [60], [5] and references therein.

In general, higher order methods do not result in monotone methods. An idea how to monotonize the schemes is given in [4].

The non-conservation of mass is a serious problem for all semi-Lagrangian schemes using the above mentioned interpolation methods for general velocity fields (except for linear interpolation, i.e. the upwind method). The procedure of following the trajectories backwards in time starting at the grid points and computing the values at the departure points pointwise using interpolation methods does not guarantee conservation of mass. If the exact conservation of the advected quantity is desired, a slightly different approach must be chosen.

2.5.3 The semi-Lagrangian integrated-mass approach

If conservation of mass is an essential property of the numerical solution the above mentioned methods with interpolation of pointwise values must be replaced with a different approach that guarantees that feature like the semi-Lagrangian integrated-mass (SLIM) approach. To speak of conservation of the density ρ or some other quantity, it is reasonable to consider the advection equation in conservation form, as for ρ in (1.4).

Instead of the purely differential formulation (2.139), we use the integral formulation for the advection equation in semi-Lagrangian form, as described for example in [29], that is

$$\frac{d}{dt}\int_{A(t)}^{B(t)} \rho(x,t)\,\mathrm{d}x = 0, \tag{2.144}$$

where the limits $A(t)$ and $B(t)$ move in time along the fluid. These are defined by

$$\begin{aligned} \frac{\partial}{\partial t}A(t) &= u(A(t),t),\\ \frac{\partial}{\partial t}B(t) &= u(B(t),t). \end{aligned} \tag{2.145}$$

This ansatz has the advantage that the conservation property is satisfied and can be passed on to the discretization inherently. To show the equivalence of the integral form of the Lagrangian derivative (2.144) and the mass continuity equation (1.4) we make use of the Leibniz rule

$$\begin{aligned} &\frac{d}{dt}\int_{A(t)}^{B(t)} \rho(x,t)\,\mathrm{d}x\\ &\quad= \int_{A(t)}^{B(t)} \frac{\partial}{\partial t}\rho(x,t)\,\mathrm{d}x + \rho(B(t),t)\frac{\partial}{\partial t}B(t) - \rho(A(t),t)\frac{\partial}{\partial t}A(t). \end{aligned} \tag{2.146}$$

With (2.145) the time derivatives of $A(t)$ and $B(t)$ can be replaced with the velocity u. Then, (2.144) is equivalent to

$$\int_{A(t)}^{B(t)} \frac{\partial}{\partial t}\rho(x,t)\,\mathrm{d}x + \rho(B(t),t)u(B(t),t) - \rho(A(t),t)u(A(t),t) = 0, \tag{2.147}$$

which is equal to

$$\int_{A(t)}^{B(t)} \left(\frac{\partial}{\partial t}\rho(x,t) + \frac{\partial}{\partial x}(u(x,t)\rho(x,t))\right)\mathrm{d}x = 0. \tag{2.148}$$

The desired conservation property is proven. The integral of quantity ρ at the time level t^n between the boundaries $A(t^n)$ and $B(t^n)$ is exactly the same as the integral of ρ between the boundaries at the time level t^{n+1},

$$\int_{A(t^{n+1})}^{B(t^{n+1})} \rho(x,t^{n+1})\,\mathrm{d}x = \int_{A(t^n)}^{B(t^n)} \rho(x,t^n)\,\mathrm{d}x\,. \tag{2.149}$$

Next, we address the question how the integral formulation of the semi-Lagrangian method (2.144) tranfers into the discrete case resulting in a numerical method that

conserves the quantity inherently. We assume that the average of the distribution ρ is given for each grid cell,

$$M_i(t) = \frac{1}{\Delta x_i} \int_{x_{i-1/2}}^{x_{i+1/2}} \rho(x,t)\,\mathrm{d}x\,. \tag{2.150}$$

Thus, the function ρ is not given pointwise, but the mean values are stored. An integrable function is constructed from these given cell averages M_i with the property

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \rho_i(x,t)\,\mathrm{d}x = \Delta x_i M_i(t) \tag{2.151}$$

for each cell. It can be used as a foundation for a numerical method that is mass-conserving. The function is locally continuous within a grid cell, but not necessarily globally continuous over the whole computational domain Ω. After a function ρ is reconstructed, we can use numerical quadrature methods to evaluate the integral of the function at any time. We have

$$\begin{aligned}\int_{A(t)}^{B(t)} \rho(x,t)\,\mathrm{d}x = \int_{A(t)}^{x_{j+1/2}} \rho_j(x,t)\,\mathrm{d}x + \sum_{i=j+1}^{k-1} \Delta x_i M_i(t) \\ + \int_{x_{k-1/2}}^{B(t)} \rho_k(x,t)\,\mathrm{d}x\,,\end{aligned} \tag{2.152}$$

where $A(t)$ is within cell j, $B(t)$ is located in the kth cell and $j+1 < k$. In this way mass is conserved.

There is a wide scope of possibilities of how to reconstruct the function ρ from the averages M_i. The simpliest idea is to use just piecewise constant functions, that is

$$\rho_i(x,t) = M_i(t). \tag{2.153}$$

This results in a first order method and is equivalent to the upwind method. Another option is to apply the idea developed for the piecewise parabolic method [13] in a different context. To reconstruct the function ρ_i as a piecewise parabola the mean values of the neighboring cells are also taken into account to define the coefficients of the polynomial,

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \rho_k(x,t)\,\mathrm{d}x = \Delta x_i M_i(t) \tag{2.154}$$

for $i = k-1, k, k+1$.

The procedure for the SLIM approach is to first determine the limits of the integration either upstream or downstream. For the upstream variant the limit $A(t^{n+1})$ is set equal to the cell boundary,

$$A(t^{n+1}) = x_{i-1/2}, \tag{2.155}$$

and the departure point $A(t^n)$ at time t^n is computed solving (2.145). The computation for B follows respectively. Then, the function ρ is reconstructed with help of cell averages $M_i(t^n)$ at the time level t^n. Further, the reconstructed function ρ is integrated between the limits according to (2.152). Last, the new cell averages $M_i(t^{n+1})$ are stored.

The time step size of many numerical methods is limited by the CFL number equal to one - or even more strictly as seen for the DG methods described in Section 2.4 - for the computational stability of the scheme. The great advantage of semi-Lagrangian methods is to allow larger time steps. Some attention should be paid if conservation of the traced quantity is needed, and if so, an approach as the SLIM is advisable to be chosen.

2.6 MPDATA

MPDATA is introduced here, because we want to enable the possibility to combine the SASLDG method and MPDATA via operator splitting to deal with a certain class of numerical problems. That will be discussed thoroughly in Section 4.3. The basics of MPDATA are presented in the following.

The idea of the multidimensional positive definite advection transport algorithm (MPDATA) by Smolarkiewicz [56] is to avoid negative values for positive definite scalars in the numerical solution with reduced numerical diffusion using an iterative approach. Additionally, this numerical scheme operates with low computational cost. Because of these properties it is widely spread and used.

2.6.1 The algorithm

The first step of MPDATA is a simple FOU step, described in Example 2.1.1. The stability condition of the CFL number to be smaller or equal to one

$$\max_i \frac{|u_{i+1/2}|\Delta t}{\Delta x} \leq 1 \tag{2.156}$$

has to be met, see e.g. [36] for a detailed description. The requirement of the method to be positive definite, which means

$$\rho_i^0 \geq 0 \text{ for all } i \Rightarrow \rho_i^N \geq 0 \text{ for all } i \text{ and for all } N, \tag{2.157}$$

is fulfilled doing the FOU step. As first order methods have large diffusion error, the diffusion error of the FOU method is large, see again e.g. [36] for details. To reduce the numerical diffusion a corrective step is followed up in MPDATA. This corrective step can be iterated multiple times. The maximimum number of iterations is denoted by $IORD$. This so-called antidiffusive step based on the truncation error made in the first step is given by

$$\begin{aligned}\rho_i^{(*)^k} &= \rho_i^{(*)^{k-1}} \\ &\quad - \left(F(\rho_i^{(*)^{k-1}}, \rho_{i+1}^{(*)^{k-1}}, u_{i+1/2}^{(\sim)^{k-1}}) - F(\rho_{i-1}^{(*)^{k-1}}, \rho_i^{(*)^{k-1}}, u_{i-1/2}^{(\sim)^{k-1}}) \right),\end{aligned} \tag{2.158}$$

where $k = 1, \ldots, IORD$ is the number of iterations. Thus, we have

$$\rho_i^{(*)^0} = \rho_i^n, \qquad \rho_i^{(*)^{IORD}} = \rho_i^{n+1} \tag{2.159}$$

and the pseudo velocities are defined by

$$u_{i+1/2}^{(\sim)^{k+1}} = \tilde{u}(u_{i+1/2}^{(\sim)^k}, \rho_i^{(*)^k}), \qquad u^{(\sim)^0} = u \tag{2.160}$$

with

$$\tilde{u}(u_{i+1/2}, \rho_i) = \frac{\left(|u_{i+1/2}|\Delta x - \Delta t u_{i+1/2}^2 \right)(\rho_{i+1} - \rho_i)}{(\rho_{i+1} + \rho_i + \epsilon)\Delta x}. \tag{2.161}$$

Note, that for $IORD = 1$ the FOU method is recovered. The antidiffusive velocities are derived in such a way that the diffusion is reduced and the positive definiteness is maintained. Analysing the truncation error of (2.158) after the first iteration step using a Taylor expansion of $\rho_i^{(*)^1}$, $\rho_{i-1}^{(*)^1}$ and $\rho_{i+1}^{(*)^1}$ about the point (x, t^n), we obtain the modified equation. By means of that equation we can explicitly specify the term that is reponsible for the diffusion error. Considering this term as diffusion equation, we would like to reverse the diffusive effect in time, which is not possible in a direct way. However, it can be reformulated as an advection equation. Taking the velocity of that equation with opposite sign, we obtain the pseudo velocity given

in (2.161). Thus, when the next iterative step is computed using this pseudo velocity, it simulates the effect of reversing the diffusion equation, that describes the diffusive error made in the first iteration, in time. In [55] Smolarkiewicz describes that the number of iterations increases the order of accuracy in space.

A further component of MPDATA is a nonoscillatory option discussed in [58]. It enables the possibility to prevent oscillations in the numerical solution. The idea is taken from the flux corrected transport developed by Zalesak [66]. It sets limits on the higher order fluxes that cause the oscillations. These fluxes act as antidiffusive fluxes that are combined with the first order fluxes.

For each grid cell i minimal and maximal values for ρ are determined that must not be exceeded,

$$\rho_i^{\mathrm{MAX}} = \max_i(\rho_{i-1}^n, \rho_i^n, \rho_{i+1}^n, \rho_{i-1}^{(*)^{k-1}}, \rho_i^{(*)^{k-1}}, \rho_{i+1}^{(*)^{k-1}}), \tag{2.162}$$

$$\rho_i^{\mathrm{MIN}} = \min_i(\rho_{i-1}^n, \rho_i^n, \rho_{i+1}^n, \rho_{i-1}^{(*)^{k-1}}, \rho_i^{(*)^{k-1}}, \rho_{i+1}^{(*)^{k-1}}). \tag{2.163}$$

Further, bounds for the fluxes have to be defined which are then to be applied in every iteration of the method

$$\beta_i^{\uparrow} = \frac{\rho_i^{\mathrm{MAX}} - \rho_i^{(*)^{k-1}}}{\frac{\Delta t}{\Delta x}\left(\max(0, u_{i-1/2}^{(\sim)^k})\rho_{i-1}^{(*)^{k-1}} - \min(0, u_{i+1/2}^{(\sim)^k})\rho_{i+1}^{(*)^{k-1}}\right) + \epsilon}, \tag{2.164}$$

$$\beta_i^{\downarrow} = \frac{\rho_i^{(*)^{k-1}} - \rho_i^{\mathrm{MIN}}}{\frac{\Delta t}{\Delta x}\left(\max(0, u_{i+1/2}^{(\sim)^k})\rho_i^{(*)^{k-1}} - \min(0, u_{i-1/2}^{(\sim)^k})\rho_i^{(*)^{k-1}}\right) + \epsilon}. \tag{2.165}$$

The pseudo velocity for the flux corrected transport leading to a monotone, nonoscillatory solution is given by

$$\left[u_{i+1/2}^{(\sim)^k}\right]^{\mathrm{MON}} = \min(1, \beta_i^{\downarrow}, \beta_{i+1}^{\uparrow}) \max(0, u_{i+1/2}^{(\sim)^k}) \tag{2.166}$$

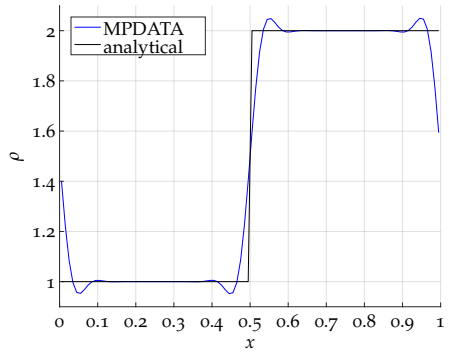
$$+ \min(1, \beta_i^{\uparrow}, \beta_{i+1}^{\downarrow}) \min(0, u_{i+1/2}^{(\sim)^k}). \tag{2.167}$$

To show the effect of the nonoscillatory option of MPDATA we compute the numerical solution of the step function using three different scenarios. The first one displays where oscillations can appear, the second shows the effect of the positive definiteness of MPDATA on oscillations and third scenario shows the effect of the nonoscillatory option. The initial data used for all examples consist of constant data with one single discontinuity. Higher order methods typically generate undesired wiggles next to the discontinuity using the step function as initial values. We compute one cycle with periodical boundary conditions, constant velocity equal to one and Courant number 0.5 on a grid with 100 grid cells. The results of the numerical and analytical solution are plotted in Figure 2.2.

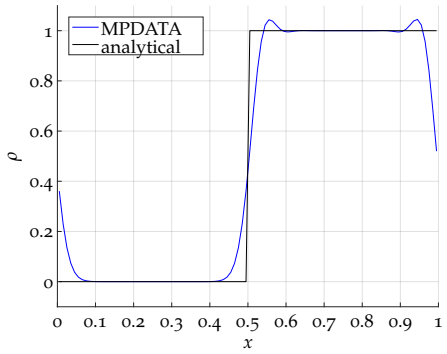
Figure 2.2(a) pictures the numerical solution without the nonoscillatory option. The initial values used for the computation are chosen such that the solution is strictly positive even without any restrictions of the algorithm. The typical wiggles near the discontinuity are visible.

As soon as the initial values are decreased such that a part of the values is equal to zero the property of preserving the positive definiteness of the solution of MPDATA comes into play. This is shown in Figure 2.2(b). The wiggles at the upper part of the solution still occur. The oscillations at the lower part are suppressed due to the requirement of a positive definite solution.

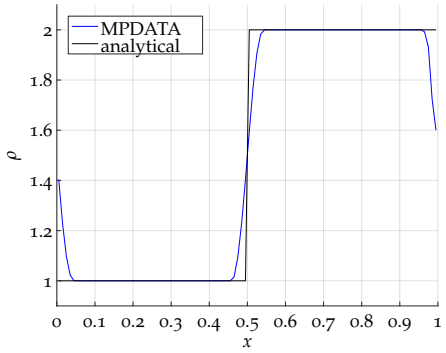
Figure 2.2(c) reveals the full effect of the nonoscillatory option of MPDATA. No oscillations occur neither at the lower or at the upper part. The effect to maintain a positive definite solution of the algorithm does not play a role here, because the initial values are again chosen large enough such that the solution maintains positive even with arising oscillations.



(a) Without nonoscillatory option.



(b) Without nonoscillatory option.



(c) With nonoscillatory option.

Figure 2.2: Different scenarios showing how oscillations can occur in the solution of MPDATA and ways to prevent those. Numerical solutions are computed applying two correction steps, $IORD = 3$ and Courant number 0.5 on a grid with 100 grid cells. Note the different scales of ρ. Analytical solution is shown for comparison.

Concluding, we add that there are many more options developed for MPDATA summarized in [59], as a third order accurate version, transporting tracers of variable sign and solving an advection-diffusion problem.

2.6.2 Error Analysis

For stability and consistency considerations we refer to [56].

Sprectral analysis

We want to gain insights on the diffusion and dispersion error of MPDATA. Therefore, we would like to conduct a spectral analysis of MPDATA to obtain information on the error in the amplitude and phase. Because of the nonlinearity of MPDATA the von Neumann analysis cannot be applied directly. The idea is to obtain the information of the diffusion and dispersion error derived from the numerical results.

We use sine functions $\sin(2\pi Kx)$ with all wave numbers $K = 1, \dots N/2 - 1$ as initial values $Q^0(K)$ that are resolved on a grid with N (even) grid cells. The sine functions are discretized on the unit interval $[0, 1]$ using N equidistant points, with grid length

$$\Delta x = \frac{1}{N} \tag{2.168}$$

and

$$x_j = (j-1)\Delta x, \quad \text{for } j = 1, \dots, N. \tag{2.169}$$

The vector X then contains the grid points, i.e.,

$$X = [x_1, x_2, \ldots, x_N]. \tag{2.170}$$

We compute the numerical solution $Q^1(K)$ after one time step with MPDATA for each K. In all computations N is chosen to be 360. Finally, we apply the DFT algorithm of MATLAB (2.22) to the initial data at time zero and to the data after the time step to both the numerical and the analytical reference solution, which is $\sin(2K\pi(x - u\Delta t))$.

$$\hat{c}_j(K) = \mathrm{DFT}(Q^0(K)), \tag{2.171}$$

$$\hat{c}_j^{\mathrm{ref}}(K) = \mathrm{DFT}(\sin(2K\pi(X - u\Delta t))), \quad \text{for } j = 1, \ldots, N. \tag{2.172}$$

From these Fourier coefficients we compute the amplitude and phase of reconstructed sine functions for each wave number. Thus, the spectral analysis is obtained and by comparing the Fourier coefficients $\hat{c}_j$ with their reference values $\hat{c}_j^{\mathrm{ref}}$ we derive the diffusion and dispersion error of MPDATA.

In the first step we select the first order terms of the Fourier coefficients, because they determine the error in amplitude and phase. For each wave number K different coefficients are of first order, i.e. the $K+1$th and $N-K+1$th coefficient are the relevant ones.

$$Q^1(K) = \sum_{j=1}^{N} \hat{c}_j e^{-2\pi i(K-1)x_j} \tag{2.173}$$

$$\approx \hat{c}_{K+1} e^{-2\pi i \frac{K}{N}(K-1)} + \hat{c}_{N-K+1} e^{-2\pi i \frac{N-K}{N}(K-1)} \tag{2.174}$$

Using the periodicity of $\exp(2\pi i N)$, we have

$$e^{-2\pi i(N-K)x_K} = e^{2\pi i K x_K}. \tag{2.175}$$

Altogether, applying some algebra and trigonometric laws lead to

$$\begin{aligned} Q^1(K) \approx\ & \hat{c}_{K+1}(\cos(2\pi K x_K) - i\sin(2\pi K x_K)) \\ & + \hat{c}_{N-K+1}(\cos(2\pi K x_K) + i\sin(2\pi K x_K)) \end{aligned} \tag{2.176}$$

$$\begin{aligned} =\ & \underbrace{(\hat{c}_{K+1} + \hat{c}_{N-K+1})}_{:=a} \cos(2\pi K x_K) \\ & + \underbrace{i(-\hat{c}_{K+1} + \hat{c}_{N-K+1})}_{:=b} \sin(2\pi K x_K) \end{aligned} \tag{2.177}$$

$$= a\cos(2\pi K x_K) + b\sin(2\pi K x_K) \tag{2.178}$$

$$= \sqrt{a^2+b^2}\left(\frac{a}{\sqrt{a^2+b^2}}\cos(2\pi K x_K) + \frac{b}{\sqrt{a^2+b^2}}\sin(2\pi K x_K)\right). \tag{2.179}$$

Because of

$$\left(\frac{a}{\sqrt{a^2+b^2}}\right)^2 + \left(\frac{b}{\sqrt{a^2+b^2}}\right)^2 = 1, \tag{2.180}$$

we can find some $\tilde{\phi}$ to express

$$\frac{a}{\sqrt{a^2+b^2}} = \sin(\tilde{\phi}) \tag{2.181}$$

and

$$\frac{b}{\sqrt{a^2+b^2}} = \cos(\tilde{\phi}). \tag{2.182}$$

Hence, we have

$$= \sqrt{a^2+b^2}\left(\sin(\tilde{\phi})\cos(2\pi K x_K) + \cos(\tilde{\phi})\sin(2\pi K x_K)\right) \tag{2.183}$$

$$= \sqrt{a^2+b^2}\left(\sin(2\pi K x_K + \tilde{\phi})\right) \tag{2.184}$$

$$= \sqrt{a^2+b^2}\left(\sin\left(2\pi K\left(x_K - \phi\right)\right)\right). \tag{2.185}$$

The amplitude is therefore given by

$$A(K) = \sqrt{a^2+b^2} \tag{2.186}$$

and the phase

$$\phi(K) = -\frac{\arcsin\left(\frac{a}{\sqrt{a^2+b^2}}\right)}{2K\pi}. \tag{2.187}$$

The amplitude and phase lead to the following reconstructed sine function for each wave number

$$f_K(x) := A(K)\sin(2K\pi(x-\phi(K))). \tag{2.188}$$

This result can be compared to the sine function shifted exactly

$$f_{\text{ref}}(x) = \sin(2K\pi(x-u\Delta t)). \tag{2.189}$$

The amplitude of the sine maintains the initial amplitude of the value one without any numerical diffusion. Hence the difference

$$e_{\text{diff}}(K) = 1 - A(K) \tag{2.190}$$

yields the numerical error of the amplitude for one time step, given as a function of the wave number K. The error of the amplitude e_{diff} relates to the diffusion error.

We know the exact phase difference of one time step, namely $u\Delta t$. The difference of the analytical phase difference and the numerical phase difference $\phi(K)$ leads to the numerical error of the phase

$$e_{\text{disp}}(K) = \phi(K) - u\Delta t, \tag{2.191}$$

which correlates to the absolute dispersion error. We follow [25] and show the relative error of the amplitude $\tilde{e}_{\text{diff}}$ and phase difference $\tilde{e}_{\text{disp}}$ which relate to the relative diffusion and dispersion error. We have

$$\tilde{e}_{\text{diff}}(K) = A(K) \tag{2.192}$$

for the relative error of the amplitude and

$$\tilde{e}_{\text{disp}}(K) = \frac{\phi(K)}{u\Delta t}. \tag{2.193}$$

for the relative error of the phase difference, respectively.

Before we show the results of the numerically obtained Fourier analysis we want to look at the FOU method and understand the diffusion and dispersion error made with this numerical scheme. As MPDATA is built upon the FOU method a comparison with respect to these errors is promising to reveal more information about MPDATA.

Analysis for the FOU method

Clearly, without any correction steps MPDATA is identical to the FOU method, so the diffusion and dispersion errors are also identical to the errors of the latter method. In the case without any corrections we can apply the von Neumann method to the FOU method to gain information on the exact errors of MPDATA.

We use the results of Example 2.1.1. The amplitude G_{FOU} and the phase Φ_{FOU} of the amplification factor of the FOU method are given by (2.39) and (2.40), respectively. The relative diffusion error $\tilde{e}_{\text{diff,FOU}}$ and the relative dispersion error $\tilde{e}_{\text{disp,FOU}}$ are obtained, their Taylor expansion about small wave numbers, i.e. $\alpha_k = 0$, is calculated and given by

$$|\tilde{e}_{\text{diff,FOU}}(k)| = \frac{|G_{\text{FOU}}(k)|}{|G_{\text{ana}}|} \tag{2.194}$$

$$\begin{aligned} &= 1 + \frac{1}{2}(\sigma - 1)\sigma\alpha_k^2 + \\ &\quad + \frac{1}{24}\left(-3\sigma^4 + 6\sigma^3 - 4\sigma^2 + \sigma\right)\alpha_k^4 + \mathcal{O}\left(\alpha_k^6\right), \end{aligned} \tag{2.195}$$

and

$$\tilde{e}_{\text{disp,FOU}}(k) = \frac{\arctan\left(\frac{\sigma\sin(\alpha_k)}{1-\sigma+\sigma\cos(\alpha_k)}\right)}{\sigma\alpha_k} \tag{2.196}$$

$$\begin{aligned} &= 1 + \frac{1}{6}\left(-2\sigma^2 + 3\sigma - 1\right)\alpha_k^2 + \\ &\quad + \frac{1}{120}\left(24\sigma^4 - 60\sigma^3 + 50\sigma^2 - 15\sigma + 1\right)\alpha_k^4 + \mathcal{O}\left(\alpha_k^6\right). \end{aligned} \tag{2.197}$$

The absolute errors enable conclusions about the order of the scheme.

$$e_{\text{diff,FOU}}(k) = \frac{|G_{\text{FOU}}(k)| - 1}{\Delta t} \tag{2.198}$$

$$\begin{aligned} &= \left(\frac{1}{2}(\sigma - 1)\sigma\alpha_k^2 + \frac{1}{24}\left(-3\sigma^4 + 6\sigma^3 - 4\sigma^2 + \sigma\right)\alpha_k^4 \right. \\ &\quad \left. + \mathcal{O}\left(\alpha_k^6\right)\right)/\Delta t \end{aligned} \tag{2.199}$$

and with $\alpha_k = \mathcal{O}(\Delta x)$ and $\sigma \in \mathcal{O}(1)$ we can determine the order of the diffusion error

$$e_{\text{diff,FOU}}(k) \sim \left(\frac{1}{2}(\sigma - 1)\sigma\Delta x^2 + \mathcal{O}\left(\Delta x^4\right)\right)/\Delta t \tag{2.200}$$

$$\sim \frac{1}{2}(\sigma - 1)\sigma\Delta x + \mathcal{O}\left(\Delta x^3\right) \tag{2.201}$$

$$\sim \mathcal{O}(\Delta x). \tag{2.202}$$

Similarly, we determine the absolute dispersion error of the FOU scheme.

$$e_{\text{disp,FOU}}(k) = \frac{\phi_{\text{FOU}} - \sigma\alpha_k}{\Delta t} \tag{2.203}$$

$$= \left(-\frac{1}{6}\alpha_k^3\left(\sigma(2\sigma^2 - 3\sigma + 1)\right)\right. \tag{2.204}$$

$$+ \frac{1}{120}\alpha_k^5\sigma(24\sigma^4 - 60\sigma^3 + 50\sigma^2 - 15\sigma + 1) \tag{2.205}$$

$$\left. + \mathcal{O}\left(\alpha_k^7\right)\right)/\Delta t. \tag{2.206}$$

With the same assumptions as above we obtain the order of the dispersion error

$$e_{\text{disp,FOU}} \sim \left(-\frac{1}{6}\Delta x^3\left(\sigma(2\sigma^2 - 3\sigma + 1)\right) + \mathcal{O}\left(\Delta x^5\right)\right)/\Delta t \tag{2.207}$$

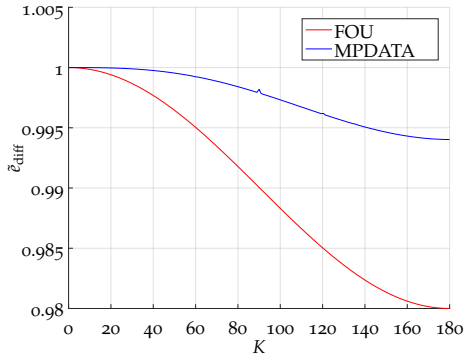
$$\sim -\frac{1}{6}\Delta x^2\left(\sigma(2\sigma^2 - 3\sigma + 1)\right) + \mathcal{O}\left(\Delta x^4\right) \tag{2.208}$$

$$\sim \mathcal{O}\left(\Delta x^2\right) \tag{2.209}$$

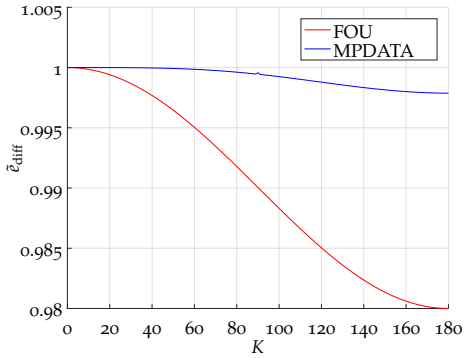
The order of convergence for the FOU method, namely order one, can be read off the results above. The diffusion error is the dominating one, because it is of lower order. Note that for $\sigma = 0.5$, the lowest order term of the dispersion error in (2.208) is cancelled out and the dispersion error reaches third order. In that case the overall order remains the same as the diffusion error is still of first order.

Plots of diffusion and dispersion error

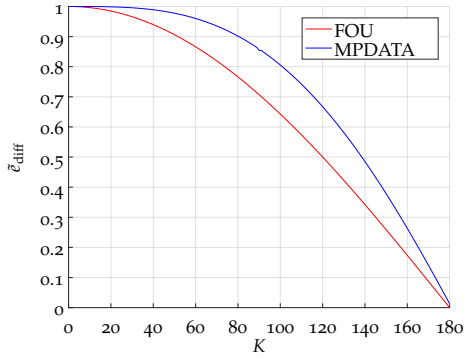
Figure 2.3 shows the relative diffusion error defined in (2.192) of MPDATA for the CFL numbers 0.01, 0.5 and 0.9. Each of these are computed with one and with two correction steps. The results are plotted as functions of the wave number k in comparison with the relative diffusion errors of the FOU method.



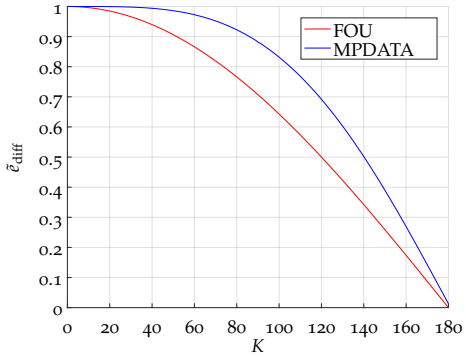
(a) $\sigma = 0.01$, MPDATA with $IORD = 2$



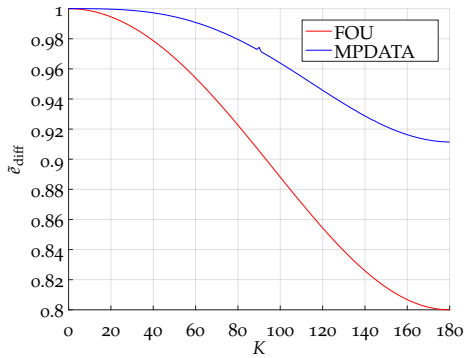
(b) $\sigma = 0.01$, MPDATA with $IORD = 3$



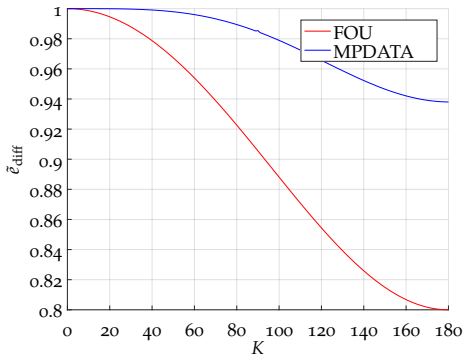
(c) $\sigma = 0.5$, MPDATA with $IORD = 2$



(d) $\sigma = 0.5$, MPDATA with $IORD = 3$



(e) $\sigma = 0.9$, MPDATA with $IORD = 2$



(f) $\sigma = 0.9$, MPDATA with $IORD = 3$

Figure 2.3: Relative diffusion error $\tilde{\epsilon}_{\text{diff}}$ of MPDATA and the FOU method for different CFL numbers. Note the different scales of $\tilde{\epsilon}_{\text{diff}}$.

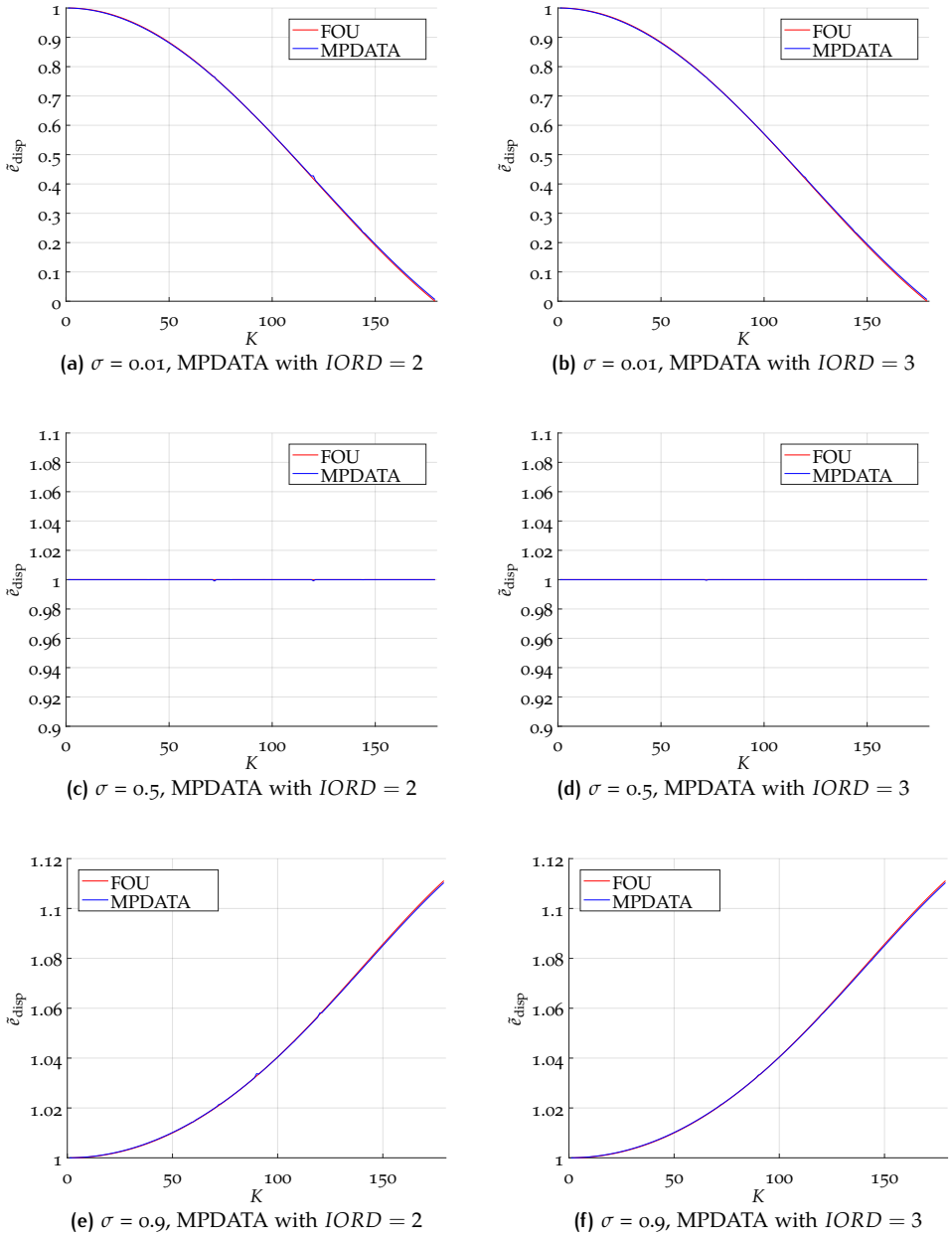
For Courant numbers 0.01 and 0.9 the second correction step decreases the diffusion error for all wave numbers, seen in Figure 2.3(a) and Figure 2.3(b), and in Figure 2.3(e) and Figure 2.3(f), respectively. For $\sigma = 0.5$ the highest oscillations tend to be damped out, when k tends to 180. In the limit $k = 180$, the oscillations are damped out completely. For lower wave numbers the diffusion error made with one corrective step (Figure 2.3(c)) is larger than the error with two corrections (Figure 2.3(d)), even though the effect is almost negligible.

From construction of MPDATA it is expected that the diffusive error is decreased while increasing the number of iterative steps. This can be confirmed by Figure 2.3.

The relative dispersion error is plotted in Figure 2.4. Similar to Figure 2.3 the panels show the numerical results obtained with $IORD = 2$ and 3 for different CFL numbers 0.01, 0.5 and 0.9. The relative dispersion errors of the FOU method are plotted as well.

Two main results can be derived from Figure 2.4. First, except for small numerical anomalies the dispersion error of MPDATA and the dispersion error of the FOU method coincide for all Courant numbers. And second, the corrective step in the algorithm of MPDATA leaves the dispersion unchanged.

Furthermore, for $\sigma = 0.01$ the dispersion error is a lagging error, for $\sigma = 0.9$ it is leading and for $\sigma = 0.5$ there is no dispersion error. Thus, the property of the FOU method - to have no dispersion error for $\sigma = 0.5$ - is inherited by MPDATA, independent from number of corrective iterations.



(a) σ = 0.01, MPDATA with $IORD = 2$

(b) σ = 0.01, MPDATA with $IORD = 3$

(c) σ = 0.5, MPDATA with $IORD = 2$

(d) σ = 0.5, MPDATA with $IORD = 3$

(e) σ = 0.9, MPDATA with $IORD = 2$

(f) σ = 0.9, MPDATA with $IORD = 3$

Figure 2.4: Relative dispersion error $\tilde{e}_{\text{disp}}$ of MPDATA and the FOU method for different CFL numbers. Note the different scaling of $\tilde{e}_{\text{disp}}$.

The equality of the diffusion error of MPDATA without corrections and the respective errors of the FOU method can be seen in Figure A.1 in the appendix.

Our results are in agreement with Smolarkiewicz and Clark [57]. There, the amplitude and phase of the numerical and analytical solution are compared, too. They came to the conlusion that the phase error of MPDATA is similar to the phase error of the FOU method.

Convergence rates

The L_∞, L_1 and L_2- error and the convergence rates are computed on different refinement levels of the grid. Table 2.1 shows the errors and convergence rates for the computations with $\sigma = 0.8$. The results computed with $\sigma = 0.5$ are shown in Table 2.2. In both computations, we use MPDATA with two correction steps ($IORD = 3$).

N	L_∞-error	L_∞-order	L_1-error	L_1-order	L_2-error	L_2-order
10	5.369e-02	-	3.364e-02	-	3.753e-02	-
20	1.306e-02	2.0391	8.019e-03	2.0685	8.973e-03	2.0642
40	3.174e-03	2.0413	1.981e-03	2.0169	2.207e-03	2.0239
80	7.832e-04	2.0189	4.939e-04	2.0042	5.490e-04	2.0068
160	1.947e-04	2.0081	1.234e-04	2.0010	1.371e-04	2.0018
320	4.856e-05	2.0036	3.084e-05	2.0002	3.426e-05	2.0005
640	1.212e-05	2.0017	7.711e-06	2.0001	8.565e-06	2.0001

Table 2.1: Convergence rates MPDATA $\sigma = 0.8$, $IORD = 3$.

For $\sigma = 0.8$ we observe a convergence rate of order two shown in Table 2.1. As the dispersion error of MPDATA is similar to the dispersion error of the FOU method for any Courant number and the latter error is of second order, see (2.208), the dispersion error of MPDATA is expected to be of second order, too. That is confirmed by our numerical results.

N	L_∞-error	L_∞-order	L_1-error	L_1-order	L_2-error	L_2-order
10	7.206e-02	-	4.676e-02	-	5.082e-02	-
20	9.504e-03	2.9225	6.526e-03	2.8410	7.082e-03	2.8431
40	1.196e-03	2.9898	8.445e-04	2.9502	9.157e-04	2.9512
80	1.526e-04	2.9713	1.072e-04	2.9780	1.158e-04	2.9831
160	1.922e-05	2.9891	1.346e-05	2.9933	1.454e-05	2.9933
320	2.409e-06	2.9960	1.686e-06	2.9969	1.822e-06	2.9970
640	3.015e-07	2.9982	2.110e-07	2.9985	2.279e-07	2.9986

Table 2.2: Convergence rates MPDATA $\sigma = 0.5$, $IORD = 3$.

For $\sigma = 0.5$ the dispersion error of both - MPDATA and the FOU method - vanishes. Thus, the dominating error is the diffusion error in that case. The diffusion error corresponds to the number of iterations. Therefore, the convergence rate of MPDATA with two or more corrective iteration steps reaches third order as seen in Table 2.2. If only one corrective iteration is applied, the convergence rates drop to order two. Without any correction steps the FOU method is recovered and the order of convergence is one.

Generally, the dispersion error limits the convergence rates in all cases $\sigma \neq 0.5$. For $\sigma = 0.5$, the dispersion error is equal to zero, which enables a higher convergence rate of order three.

In summary, MPDATA is a robust method, that operates with low numerical cost and can be configured with additional features like the nonoscillatory option if needed. However, the time step size for MPDATA is limited due to the CFL condition.

3 THE SEMI-ANALYTICAL SEMI-LAGRANGIAN DISCONTINUOUS GALERKIN METHOD

In the previous chapter numerical methods were introduced that form the basis for the new numerical scheme, the semi-analytical semi-Lagrangian discontinuous Galerkin (SASLDG) method, which will be developed in the following. First of all, the SASLDG method is intended to solve the linear advection equation with variable velocity. Furthermore, the properties of this method are mass conservation, high accuracy and no restrictions on the time step size. Thus, mass or more generally the integral of an advected quantity over the spatial domain must be conserved in time as in the SOM method, the DG methods and some of the SL schemes. The high accuracy of DG methods and the SOM method is adopted and implemented by using a polynomial of degree two. In addition, the property of semi-Lagrangian schemes to use time steps of arbitrary length without violating stability properties is incorporated.

To present the idea of how the numerical method works we begin with a short preview in Section 3.1. This simplifies the understanding of the technical parts of the implementation afterwards. To realize the requirements of the numerical method described above, the following structure is pursued: In Section 3.2, we derive the trajectory at which the quantity is advected in time in a semi-Lagrangian manner. The exact evolution of the advected tracer is shown in Section 3.3. The projection step to rebuild the polynomial for each grid cell is discussed in Section 3.4. An overview of the algorithm of the method is given in Section 3.5. A description of the limits of the integration that are needed for the projection step is provided in Section 3.6. This chapter concludes with the results of numerical tests of the SASLDG method in Section 3.7.

3.1 PREVIEW

We start with a reminder of the linear advection equation, rewritten as a system of two equations in conservation form, introduced in (1.4) and (1.5),

$$\rho_t + (u\rho)_x = 0, \tag{3.1}$$

$$(\rho y)_t + (u\rho y)_x = 0. \tag{3.2}$$

The equations for density ρ and tracer density ρy are solved in the same way. The solution of the tracer y is obtained by division at each time the solution of y is needed. As (3.2) is solved analogously to (3.1), we only discuss the numerical solution to the continuity equation (3.1) for density ρ from now on.

As mentioned before, the SASLDG method is an extension to the SOM method, whose idea is to advect the tracer, represented as polynomial distribution, analytically exact along the characteristics for one time step and project the resulting distributions onto polynomials. For details see Section 2.3. That notion is adopted for the SASLDG method with some adjustments due to shortcomings of the SOM method.

A drawback of the SOM method is the treatment of variable velocity that is approximated by piecewise constant velocity. The determination of the exact characteristics and thus the computation of the analytical solution to the linear advection

equation is done easily in that case. However, the idea of the SASLDG method is to compute the analytical solution even to a non-constant velocity field. Hence, the advected quantity and the surrounding fluid are allowed to be compressible, i.e. the density might change in space and time. Another downside is the limitation for the time step. Step sizes are only allowed up to CFL numbers equal to one. This is overcome in the SASLDG method by following the characteristics backward in time, not limited by the CFL number, but arbitrarily long in a semi-Lagrangian manner.

To find the trajectories we rewrite the linear advection equation as a system of ODEs with the help of the material derivative as described in Section 2.5. The advection equation (3.1) is equivalent to

$$\begin{aligned} &\frac{d}{dt}\rho(x(t),t) = -\rho(x(t),t)\frac{\partial}{\partial x}u(x(t),t), \\ &\rho(x(t_0),t_0) = \rho_0 \end{aligned} \tag{3.3}$$

with

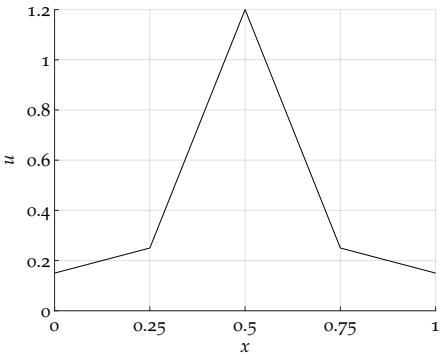
$$\begin{aligned} &\frac{d}{dt}x(t) = u(x(t),t), \\ &x(t_0) = x_0. \end{aligned} \tag{3.4}$$

The ODE (3.4) describes the trajectory depending on the velocity u and time. The information of the corresponding density is provided by the ODE (3.3).

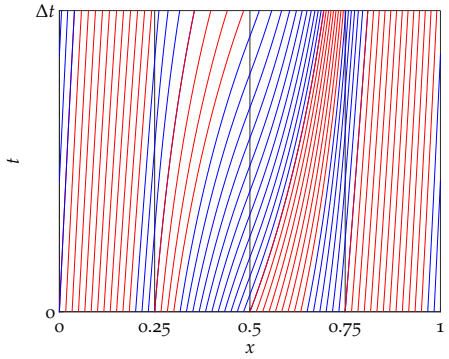
The SASLDG method consists of three main steps. First, the solution of (3.4) is computed analytically. The resulting trajectory φ describes the path along which the tracer is advected. Second, the evolution of the density is determined by solving (3.3) analytically. The solutions of (3.4) and (3.3) combined yield the exact solution to the linear advection equation (3.1). As the third and last step, we compute the projection of the exact solution. We introduce an error at that point, but we retrieve the polynomial structure that enables us to repeat the same steps again and again, and thus proceed in time. In this way, essential elements of the SOM method and of semi-Lagrangian schemes are incorporated into the SASLDG method. The tracer is advected exactly along the trajectory and then projected onto polynomials in each cell. In addition, time steps of any size are feasible, because we can follow the trajectories arbitrarily long in time.

To visualize the procedure of one advection step using the SASLDG method we give an example in Figures 3.1 and 3.2. Given a certain velocity that is piecewise linear and depends only on space, as in Figure 3.1(a), we compute the corresponding solution φ to (3.4) and plot the trajectories in Figure 3.1(b) for one time step Δt. The different coloring indicates an important difference of the trajectories. The red colored trajectories remain within one grid cell, whereas the blue ones start at time zero in a different grid cell from where they end at time Δt. That means the blue trajectories cross at least one cell boundary. This fact is reflected in different forms of the solution trajectories.

The next steps can be observed in Figure 3.2. The initial distribution of ρ is chosen here to be a constant function equal to one, see Figure 3.2(a), but it could be any arbitrary quadratic polynomial. Figure 3.2(b) shows the exact solution of the linear advection equation after one time step computed from the solutions of (3.4) and (3.3). The colors indicate which part of the solution is advected within one grid cell (colored red) and which part of the solution is advected to its neighboring grid cell (colored blue). This corresponds to the same coloring of the trajectories in Figure 3.1(b). The exact solution pictured in Figure 3.2(b) is discontinuous in the first cell and must be piecewise defined within all grid cells. Hence, the solution is complicated to express. To simplify the form, the exact solution is projected onto the space of quadratic polynomials. Both functions, the exact solution and the result of the projection, that is the polynomial, can be seen in Figure 3.2(c). In Figure 3.2(d) the polynomial is displayed, which can now be the initial distribution for the next time step.

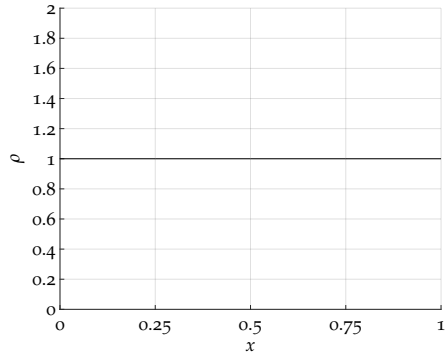


(a) Velocity field.

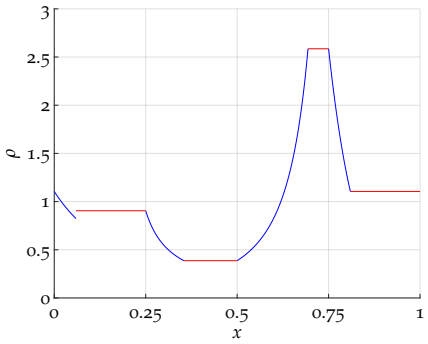


(b) Trajectories φ, computed from the velocity distribution in Figure 3.1(a)

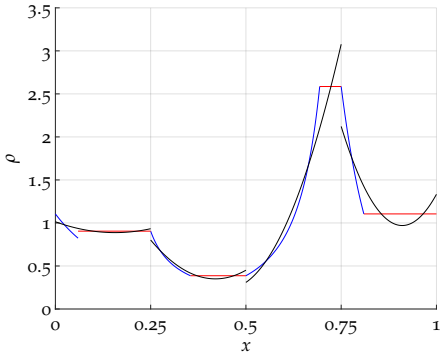
Figure 3.1: Example of a velocity field with corresponding trajectories.



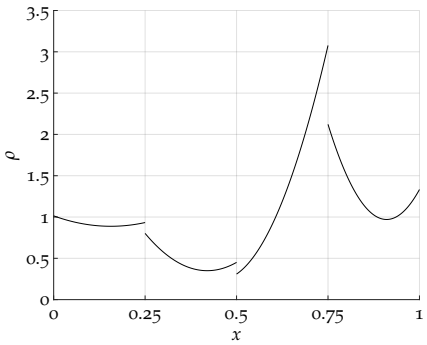
(a) Polynomial distributions are given for each grid cell.



(b) Compute exact solution.



(c) Project the discontinuous distributions onto continuous polynomial distributions.



(d) Start again with polynomial distributions given.

Figure 3.2: Construction of the SASLDG method.

In the following section we explain the derivation of the solution of (3.4), which is needed for the plots in Figure 3.1(b). In Section 3.3 the solution to (3.3) is discussed. The algorithm to understand the projection step shown Figure 3.2(c) is given in Section 3.4.

3.2 COMPUTING THE TRAJECTORIES

In this section we discuss the solution of the ODE in (3.4). The goal is to find an analytical solution carrying over the notion of the numerical methods by Prather and van Leer described in Section 2.2 and 2.3, respectively.

We first comment on a general solution for the ODE (3.4). Then, some simplifying assumptions have to be made in order to determine the analytical solution. For the detailed description of the trajectories, we differentiate between the trajectories that remain within one grid cell during a time step and the trajectories that cross at least one cell boundary. Trajectories that remain within one grid cell have the starting point and the endpoint after time Δt in the same grid cell. Trajectories that have an underlying positive velocity distinguish from the trajectories with negative velocity. The next section is devoted to the trajectories with positive velocity and the following section to the latter case. Both sections first examine the case of a trajectory that remains within one grid cell and then discuss the trajectories that cross cell boundaries. For these trajectories the treatment of coefficients that are near to or equal to zero is necessary and conducted subsequently.

The notation is introduced in the beginning of Chapter 2. In this chapter, the local coordinates $[0, \Delta x_i]$ are often used instead of the global coordinates $[x_{i-1/2}, x_{i+1/2}]$ for the ith cell, which simplifies the notation.

If the velocity field $u : \Omega \times \mathbb{R} \to \mathbb{R}$ is locally Lipschitz continuous for each $x \in \Omega$, then the solution of the ODE (3.4) exists on Ω for all time and is unique. We define this solution as the function φ:

Definition 3.2.1 *Let the function $\varphi : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be the solution to the ODE (3.4). Then, it satisfies*

$$\frac{d\varphi}{dt} = u(\varphi(x_0, t_0, t), t), \tag{3.5}$$

and the initial condition

$$\varphi(x_0, t_0, 0) = x_0. \tag{3.6}$$

The solution φ is defined in such a way that the first two arguments describe the starting point and time (x_0, t_0), using the initial condition of the ODE. The third argument t denotes the transformation in time t. The result of the transformation yields $\varphi(x_0, t_0, t)$.

Remark 3.2.1 *The function φ defined by 3.2.1 is the flow of the velocity field u and satisfies the following properties for $s, t \in \mathbb{R}$*

$$\varphi(x_0, t_0, s + t) = \varphi(\varphi(x_0, t_0, s), t_0 + s, t), \tag{3.7}$$

$$\varphi(x_0, t_0, 0) = x_0. \tag{3.8}$$

The result of the transformation is the same: if the flow first passes to the point $\varphi(x_0, t_0, s)$ during time s starting at time t_0 and then passes further for time t starting at time $t_0 + s$ or if the flow reaches directly the point $\varphi(x_0, t_0, s + t)$ during time $t + s$.

The time that passes can be reversed by the inverse of φ as shown in the following remark.

Remark 3.2.2 *The function φ is a diffeomorphism, so the inverse of φ exists*

$$\varphi^{-1}(x_0, t_0, t) = \varphi(x_0, t_0, -t), \tag{3.9}$$

and is differentiable, too. Thus, we have

$$\varphi^{-1}(\varphi(x_0, t_0, t), t_0 + t, t) = \varphi(\varphi(x_0, t_0, t), t_0 + t, -t) \tag{3.10}$$

$$= \varphi(x_0, t_0, t - t) \tag{3.11}$$

$$= x_0. \tag{3.12}$$

For a given initial starting point (x_0, t_0) we study the points $\varphi(x_0, t_0, t)$, which are passed during time t. The set of points that is obtained by this action for all $t \in \mathbb{R}$ is called trajectory of the flow φ of the point (x_0, t_0). The trajectory can be thought of the path at which particles of the tracer or the density move along.

Local Lipschitz-continuity of the velocity u is sufficient to know that a unique solution to the initial value problem (3.4) exists. However, if the goal is to solve the problem not only approximately but analytically, some assumptions have to be made. We assume that the velocity field is continuous, piecewise linear and depends only on space. Then, the velocity u takes the form

$$u|_{\Omega_i}(x) = a_i x + b_i \tag{3.13}$$

for each cell. The coefficient a_i describes the slope of the velocity of each grid cell. Thus, if a_i is equal to zero, the velocity is constant in the ith cell. The coefficients a_i and b_i for $i = 1, \dots, N$ are computed from values u_i given at the cell boundaries. Because of the continuity of u, it holds

$$\begin{aligned} u_i := u(x_{i-1/2}) &= a_i x_{i-1/2} + b_i, && (3.14)\\ &= a_{i-1} x_{i-1/2} + b_{i-1}, && (3.15) \end{aligned}$$

and expressed in local coordinates $[0, \Delta x_i]$

$$\begin{aligned} u|_{\Omega_i}(0) &= b_i = u_i && (3.16)\\ u|_{\Omega_i}(\Delta x_i) &= a_i \Delta x_i + b_i = u_{i+1}. && (3.17) \end{aligned}$$

The ODE (3.4) describing the trajectory for linear velocity for the ith cell simplifies to

$$\begin{aligned} \frac{d\varphi}{dt} &= a_i \varphi(x, t^n, t) + b_i, \\ \varphi(x, t^n, 0) &= x. \end{aligned} \tag{3.18}$$

Since the velocity u differs in general for each grid cell, ODEs with different coefficients have to be solved and put together once the trajectories cross cell boundaries. The solution φ takes a different form if it stays within one cell and does not cross any boundary, opposed to the form of the trajectory crossing at least one boundary within one time step. Further, trajectories resulting from positive velocity u differ from trajectories with negative velocity. The case of the velocity having a root is discussed in Section 3.5 and 3.6. We first consider trajectories with strictly positive velocity.

3.2.1 Trajectory with positive velocity

In the description of Figure 3.1 we mention the role of trajectories crossing cell boundaries or remaining within a grid cell. The trajectories that have the starting point and the endpoint after one time step of length Δt within one grid cell are plotted red colored. These trajectories are expressed by a different form than the blue colored trajectories that cross cell boundaries.

We describe the trajectory remaining within the kth grid cell, i.e. in the interval $[x_k, x_{k+1}]$ or locally $[0, \Delta x_k]$. The solution to the linear ODE of first order in the inhomogeneous case (3.18) is given at time t by

$$\varphi(x, t^n, t) = e^{a_k t} \left(\frac{b_k}{a_k} + x \right) - \frac{b_k}{a_k}. \tag{3.19}$$

The coefficient a_k is the slope of the velocity. In the special case of $a_k = 0$, and thus constant velocity, the ODE (3.18) reduces to

$$\begin{aligned} \frac{d\varphi}{dt} &= b_i, \\ \varphi(x, t^n, 0) &= x. \end{aligned} \tag{3.20}$$

and is solved by integration, which yields

$$\varphi(x, t^n, t) = x + b_k t \tag{3.21}$$

at time t. Note that we obtain the same solution as described for semi-Lagrangian methods in (2.143). Prather's SOM method and van Leer's scheme both rely on this trajectory, as well, even though this equation is not explicitly mentioned in [43] and [31].

We turn to the case for the trajectory crossing at least one cell boundary. An example of a trajectory crossing n cells during a time step of length Δt is plotted in Figure 3.3, illustrating the construction. We describe the assembling for the red colored trajectory with starting point x, which is located in the ith grid cell in the interval $[x_{i-1/2}, x_{i+1/2}]$. For simplification and a better understanding, we use local coordinates. Thus, the left and right cell boundary $x_{i-1/2}$ and $x_{i+1/2}$ are identified with x_0 and x_1, respectively. All n boundaries that the trajectory crosses are labeled incrementally with x_k for $k = 1, \ldots, n$. The endpoint of the trajectory labeled as $\varphi(x, t^n, \Delta t)$ lies in the $(i+n)$th grid cell, in the interval $[x_{(i+n)-1/2}, x_{(i+n)+1/2}]$ or $[x_n, x_{n+1}]$ in local coordinates, respectively.

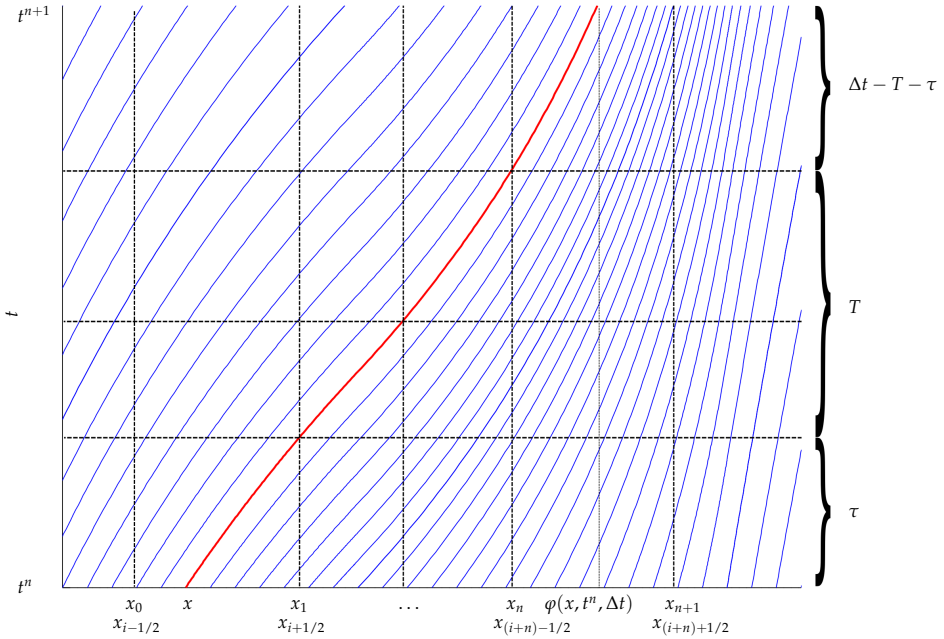


Figure 3.3: Example of trajectories computed from a positive velocity field. The red colored trajectory is $\varphi(x, t^n, t)$, with starting point x at time t^n. It crosses n grid cell boundaries (here $n = 3$).

We follow the trajectory with starting point x at time t^n to the right cell boundary x_1. This takes a certain time, defined as τ, i.e. $\varphi(x, t^n, \tau) = x_1$. Next, the time is computed that is needed to cross through the grid cell between x_1 and x_2, defined as t_1. In the same way we follow the trajectory φ from cell to cell. Generally, it takes time t_k to cross through cell k. The sum over all intermediate time intervals t_k is denoted by T,

$$T = \sum_{k=1}^{n-1} t_k. \tag{3.22}$$

Once the last boundary x_n is crossed, the arrival point $\varphi(x, t^n, \Delta t)$ at time t^{n+1} is computed. These steps assembled yield the construction of the trajectory,

$$\begin{aligned}
\varphi(x, t^n, \Delta t) &= \varphi(\varphi(x, t^n, \tau), t^n + \tau, \Delta t - \tau) && (3.23)\\
&= \varphi(x_1, t^n + \tau, \Delta t - \tau) && (3.24)\\
&= \varphi(\varphi(x_1, t^n + \tau, t_1), t^n + \tau + t_1, \Delta t - \tau - t_1) && (3.25)\\
&= \varphi(x_2, t^n + \tau + t_1, \Delta t - \tau - t_1) && (3.26)\\
&= \dots && (3.27)\\
&= \varphi(x_n, t^n + \tau + T, \Delta t - T - \tau). && (3.28)
\end{aligned}$$

The first step of advancing time t^n by τ is described by the right-hand side of (3.23). The trajectory proceeds to the point $t^n + \tau$ in time, arriving at the intermediate point $\varphi(x, t^n, \tau) = x_1$. To advance to the next cell boundary x_2 the time interval t_1 passes. It is determined by solving $\varphi(x_1, t^n + \tau, t_1) = x_2$ for t_1 in (3.25). Less time, i.e. $\Delta t - \tau - t_1$, has to pass to reach the final arrival point. In such a way, the trajectory proceeds from one cell boundary to the next until the last cell boundary x_n is reached. The point x_n is known as well as the time intervals τ and T. From this information $\varphi(x, t^n, \Delta t)$ is determined in (3.28).

For each individual grid cell the trajectory is given by (3.19) or (3.21), representing one substep of (3.23) to (3.28). We start with the computation of the time interval τ. It is the time interval in which the trajectory proceeds from the initial value x as starting point at t^n to the right boundary x_1,

$$\begin{aligned}
&\varphi(x, t^n, \tau) = x_1 && (3.29)\\
\iff &e^{a_0 \tau} \left(\frac{b_0}{a_0} + x \right) - \frac{b_0}{a_0} = x_1. && (3.30)
\end{aligned}$$

We write (3.30) in local coordinates, replacing x_1 with Δx_0, and obtain the solution for τ

$$\begin{aligned}
\tau &= -\frac{1}{a_0} \ln \left(\frac{x + \frac{b_0}{a_0}}{\Delta x_0 + \frac{b_0}{a_0}} \right) && (3.31)\\
&= -\frac{1}{a_0} \ln \left(\frac{a_0 x}{u_1} + \frac{b_0}{u_1} \right). && (3.32)
\end{aligned}$$

Next, we compute time t_k that describes the duration of the trajectory to cross the kth cell, from boundary x_k to boundary x_{k+1},

$$\begin{aligned}
&\varphi(x_k, t^n + \tau + t_1 + \dots + t_{k-1}, t_k) = x_{k+1} && (3.33)\\
\iff &e^{a_k t_k} \left(\frac{b_k}{a_k} + x_k \right) - \frac{b_k}{a_k} = x_{k+1} && (3.34)\\
\iff &e^{a_k t_k} = \frac{x_{k+1} + \frac{b_k}{a_k}}{\frac{b_k}{a_k} + x_k} && (3.35)\\
\iff &t_k = \frac{1}{a_k} \ln \left(\frac{a_k x_{k+1} + b_k}{a_k x_k + b_k} \right). && (3.36)
\end{aligned}$$

Solving for t_k, we have

$$t_k = \frac{1}{a_k} \ln \left(\frac{u_{k+1}}{u_k} \right). \qquad (3.37)$$

With the computations of τ and t_k for $i = 1, \dots, n-1$ we assemble the general formula for the trajectories crossing n cell boundaries ($n > 0$), using (3.28)

$$\begin{aligned}
&\varphi(x_n, t^n + T + \tau, \Delta t - \tau - T) \\
&\quad = \exp\left(a_n(\Delta t - \tau - T)\right)\left(\frac{b_n}{a_n} + x_n\right) - \frac{b_n}{a_n} && (3.38)\\
&\quad = \exp\left(a_n(\Delta t - T)\right)\left(\frac{a_0}{u_1}x + \frac{b_0}{u_1}\right)^{\frac{a_n}{a_0}} \frac{1}{a_n}\left(b_n + a_n x_n\right) - \frac{b_n}{a_n}. && (3.39)
\end{aligned}$$

Again, writing this expression in local coordinates with $x_n = 0$, we obtain

$$\varphi(x, t^n, \Delta t) = \frac{u_n}{a_n}\exp\left(a_n(\Delta t - T)\right)\left(\frac{a_0}{u_1}x + \frac{u_0}{u_1}\right)^{\frac{a_n}{a_0}} - \frac{u_n}{a_n}. \tag{3.40}$$

It will later turn out that a different form of φ is beneficial. Hence, we substitute $x \in [0, \Delta x_0]$ with

$$x = \Delta x_0(1 - \eta) = \Delta x_0 - \Delta x_0 \eta. \tag{3.41}$$

It yields for $\eta \in [0, 1]$

$$\widetilde{\varphi}(\eta, t^n, \Delta t) = \frac{u_n}{a_n}\exp\left(a_n(\Delta t - T)\right)\left(\frac{a_0}{u_1}(\Delta x_0 - \Delta x_0\eta) + \frac{u_0}{u_1}\right)^{\frac{a_n}{a_0}} - \frac{u_n}{a_n} \tag{3.42}$$

$$= \underbrace{\frac{u_n}{a_n}\exp\left(a_n(\Delta t - T)\right)}_{c_1}\left(\underbrace{-\frac{a_0}{u_1}\Delta x_0}_{c_2}\eta + 1\right)^{\overbrace{\frac{a_n}{a_0}}^{c_3}} - \underbrace{\frac{u_n}{a_n}}_{c_4} \tag{3.43}$$

In summary, the trajectories crossing $n \geq 1$ cell boundaries going forward in time for positive velocity can be expressed as

$$\widetilde{\varphi}(x, t^n, \Delta t) = c_1(c_2 x + 1)^{c_3} - c_4, \quad \text{for } x \in [0, 1]. \tag{3.44}$$

Recall that the indices of the coefficients refer to local values. The global cell $(i + n)$ where the trajectory arrives, is labeled with index n, the departure cell is denoted by 0, though globally it is the ith cell.

Note that φ is a diffeomorphism. We can follow the trajectory either forward or backward in time. In the projection step, both directions are needed, tracing the trajectories backward in time is needed for the computation of the limits of the integration discussed in Section 3.6.

We have to pay attention to coefficients a_i for $i = 0, \dots, n$ as they can be close or equal to zero. Thus, some approximations have to take place in order to avoid division by zero. First, we consider all cases of the coefficient a_k being small, beginning with $0 < k < n$, then, $a_0 < \epsilon$ and $a_n < \epsilon$ for some small value ϵ. The size of the boundary ϵ is discussed in Section 3.4.

Positive velocity: $|a_k| < \epsilon$

We consider the case that a_k is small, where $0 < k < n$. The coefficient a_k appears in the denominator in the computation of t_k in (3.37). If a_k is equal to zero, division must be avoided. In that case, the trajectory for cell $[x_k, x_{k+1}]$ takes the form given by (3.21). From that and the ansatz

$$\varphi(x_k, t^n + \tau + t_1 + \cdots + t_{k-1}, t_k) = x_{k+1}, \tag{3.45}$$

we obtain t_k

$$\begin{aligned}
t_k &= \frac{x_{k+1} - x_k}{b_k} && (3.46)\\
&= \frac{\Delta x_k}{u_k}. && (3.47)
\end{aligned}$$

The division by a_k being equal or nearly equal to zero can be circumvented by applying a Taylor expansion of (3.37) about the point $a_k = 0$. The following approximation remediates the problem,

$$t_k = \frac{1}{a_k} \ln\left(\frac{u_{k+1}}{u_k}\right) \tag{3.48}$$

$$= \frac{1}{a_k} \ln\left(\frac{a_k \Delta x_k + u_k}{u_k}\right) \tag{3.49}$$

$$= \frac{1}{a_k} \ln\left(1 + \frac{a_k \Delta x_k}{u_k}\right) \tag{3.50}$$

$$= \frac{\Delta x_k}{u_k} - \frac{a_k}{2}\left(\frac{\Delta x_k}{u_k}\right)^2 + \frac{a_k^2}{3}\left(\frac{\Delta x_k}{u_k}\right)^3 - \frac{a_k^3}{4}\left(\frac{\Delta x_k}{u_k}\right)^4 + \mathcal{O}\left(a_k^4\right). \tag{3.51}$$

If $a_k = 0$ is set to zero in the approximation (3.51), it equals the case (3.47). The approximation (3.51) converges if $|a_k|\Delta x_k < |u_k|$.

Positive velocity: $|a_0| < \epsilon$

We first look at the case when the coefficient a_0 is equal to zero. Hence, the velocity is constant in the first grid cell $[x_0, x_1]$ that the trajectory passes. We derive the trajectory φ for that particular case. First, we compute τ with the ansatz

$$\varphi(x, t^n, \tau) = x_1. \tag{3.52}$$

Using (3.21) as the piecewise definition for the trajectory in cell $[x_0, x_1]$, we obtain the time interval

$$\tau = \frac{\Delta x_0 - x}{u_0}. \tag{3.53}$$

Analogously to the construction of φ in (3.38) - (3.40), we can build the trajectory as follows

$$\varphi(x, t^n, \Delta t) = \varphi(x_n, t^n + \tau + T, \Delta t - T - \tau) \tag{3.54}$$

$$= \exp(a_n(\Delta t - T - \tau))\frac{b_n}{a_n} - \frac{b_n}{a_n} \tag{3.55}$$

$$= \exp(a_n(\Delta t - T))\exp\left(a_n \frac{x - \Delta x_0}{u_0}\right)\frac{b_n}{a_n} - \frac{b_n}{a_n} \tag{3.56}$$

$$= \frac{u_n}{a_n}\exp(a_n(\Delta t - T))\exp\left(a_n \frac{x - \Delta x_0}{u_1}\right) - \frac{u_n}{a_n}. \tag{3.57}$$

Note that $u_0 = u_1$, because $a_0 = 0$. Next we consider the case of a_0 being in the vicinity of zero. Based on the general form of a trajectory $\widetilde{\varphi}(x, t^n, \Delta t)$ crossing more than one grid cell (3.43), we determine an approximation of the trajectory with a series expansion about the point $a_0 = 0$.

$$\widetilde{\varphi}(x, t^n, \Delta t) = \frac{u_n}{a_n}\exp\left(a_n(\Delta t - T)\right)\left(-\frac{a_0}{u_1}\Delta x_0 \eta + 1\right)^{\frac{a_n}{a_0}} - \frac{u_n}{a_n} \tag{3.58}$$

$$= \frac{u_n}{a_n}\exp\left(a_n(\Delta t - T)\right)\exp\left(\frac{a_n}{a_0}\ln\left(-\frac{a_0}{u_1}\Delta x_0 \eta + 1\right)\right) - \frac{u_n}{a_n} \tag{3.59}$$

$$\begin{aligned} = \frac{u_n}{a_n}\exp\left(a_n(\Delta t - T)\right)\exp\left(-\frac{a_n \Delta x_0}{u_1}x\right)\Bigg(1 + \frac{a_0 a_n \Delta x_0^2}{2u_1^2}x^2 \\ + \frac{a_0^2 a_n \Delta x_0^3 x^3}{3u_1^3} + \frac{2a_0^3 a_n \Delta x_0^4 + a_0^2 a_n^2 \Delta x_0^4}{6u_1^4}x^4 \\ + \frac{a_0^3 a_n^2 \Delta x_0^5}{6u_1^5}x^5 + \frac{a_0^3 a_n^3 \Delta x_0^6}{48u_1^6}x^6 + \mathcal{O}\left(a_0^4\right)\Bigg) - \frac{u_n}{a_n} \end{aligned} \tag{3.60}$$

We omit the higher order terms and sum up the expression using suitable abbreviations. Then, the trajectory for $|a_0| < \epsilon$ crossing more than one grid cell is given by

$$\begin{aligned}\widetilde{\varphi}(x, t^n, \Delta t) = d_1 \exp(d_2 x)\Big(1 + d_3 x^2 + d_4 x^3 + d_5 x^4 + d_6 x^5 \\ + d_7 x^6\Big) + d_8.\end{aligned} \tag{3.61}$$

If we set the coefficient a_0 to zero in (3.60) apply the resubstitution

$$x = \frac{-\eta + \Delta x_0}{\Delta x_0} \tag{3.62}$$

we recover exactly the trajectory in Equation (3.57).

Positive velocity: $|a_n| < \epsilon$

The procedure of $a_n < \epsilon$ follows analogously to the case of a_0 being small. We consider the case of constant velocity in grid cell $[x_n, x_{n+1}]$, which is the cell with the endpoint of the trajectory.

Thus, we set a_n to zero and derive the trajectory φ for that case. The time interval τ for the first cell remains unchanged and is given by (3.31). We use the trajectory for constant velocity given by (3.21) for cell $[x_n, x_{n+1}]$ to assemble the whole trajectory

$$\varphi(x, t^n, \Delta t) = \varphi(x_n, t^n + \tau + T, \Delta t - T - \tau) \tag{3.63}$$

$$= b_n(\Delta t - T - \tau) \tag{3.64}$$

$$= u_n(\Delta t - T) + \frac{u_n}{a_0} \ln\left(\frac{a_0 x + u_0}{u_1}\right). \tag{3.65}$$

For the case of a_n being nearly but not equal to zero, we analyze the trajectory $\widetilde{\varphi}$ given in (3.43)

$$\widetilde{\varphi}(x, t^n, \Delta t) = \frac{u_n}{a_n} \exp\left(a_n(\Delta t - T)\right)\left(-\frac{a_0}{u_1}\Delta x_0 \eta + 1\right)^{\frac{a_n}{a_0}} - \frac{u_n}{a_n} \tag{3.66}$$

$$\begin{aligned}= \frac{u_n}{a_n} \exp\left(a_n(\Delta t - T)\right) \exp\left(\frac{a_n}{a_0} \ln\left(-\frac{a_0}{u_1}\Delta x_0 \eta + 1\right)\right) \\ - \frac{u_n}{a_n}.\end{aligned} \tag{3.67}$$

We apply a series expansion about the point $a_n = 0$ to approximate the trajectory,

$$\begin{aligned}\widetilde{\varphi}(x, t^n, \Delta t) = u_n(\Delta t - T) + \frac{1}{2}u_n a_n(\Delta t - T)^2 + \frac{1}{6}u_n a_n^2(\Delta t - T)^3 \\ + \frac{1}{24}u_n a_n^3(\Delta t - T)^4 + \frac{1}{6}\frac{u_n}{a_0}\Big((\Delta t - T)^3 a_n^3 + 3(\Delta t - T)^2 a_n^2 \\ + 6(\Delta t - T)a_n + 6\Big) \ln\left(-\frac{a_0}{u_1}\Delta x_0 x + 1\right) \\ + \frac{1}{4}\frac{u_n a_n}{a_0^2}\Big((\Delta t - T)^2 a_n^2 \\ + 2(\Delta t - T)a_n + 2\Big) \ln^2\left(-\frac{a_0}{u_1}\Delta x_0 x + 1\right) \\ + \frac{1}{6}\frac{u_n a_n^2}{a_0^3}\left((\Delta t - T)a_n + 1\right) \ln^3\left(-\frac{a_0}{u_1}\Delta x_0 x + 1\right) \\ + \frac{1}{24}\frac{u_n a_n^3}{a_0^4} \ln^4\left(-\frac{a_0}{u_1}\Delta x_0 x + 1\right) + \mathcal{O}\left(a_n^4\right).\end{aligned} \tag{3.68}$$

Further, for clarity and emphasis of the form of the resulting trajectory, we abbreviate some expressions and omit higher order terms

$$\begin{aligned}\widetilde{\varphi}(x,t^n,\Delta t) = d_1 + d_2 \ln(d_6 x + 1) + d_3 \ln^2(d_6 x + 1) \\ + d_4 \ln^3(d_6 x + 1) + d_5 \ln^4(d_6 x + 1).\end{aligned} \tag{3.69}$$

For the validation of the expansion we examine the case where we set a_n to zero in (3.68). After resubstitution of

$$x = \frac{-\eta + \Delta x_0}{\Delta x_0}, \tag{3.70}$$

we recover exactly the trajectory in (3.65).

Positive velocity: $|a_0| < \epsilon$ ***and*** $|a_n| < \epsilon$

After we examined the cases for a single cell with (nearly) constant velocity, and thus the coefficient $|a_k| < \epsilon$, for either $k = 0$, $k = n$ or some k in between, we now treat the case of $|a_0|$ and $|a_n|$ being smaller than ϵ.

We first treat the case that $n = 0$, which means that the trajectory remains within one grid cell. The trajectory is given by (3.19) and approximated in the following by a series expansions about $a_n = 0$,

$$\varphi(\Delta x_0 x, t^n, \Delta t) = e^{a_n t}\left(\frac{b_n}{a_n} + \Delta x_0 x\right) - \frac{b_n}{a_n} \tag{3.71}$$

$$\begin{aligned}&= \left(1 + a_n \Delta t + \frac{1}{2}a_n^2 \Delta t^2 + \frac{1}{6}a_n^3 \Delta t^3\right)\Delta x_0 x \\ &\quad + u_n \Delta t\left(1 + \frac{1}{2}a_i \Delta t + \frac{1}{6}a_n^2 \Delta t^2\right) + \mathcal{O}\left(a_n^4\right).\end{aligned} \tag{3.72}$$

We turn to the case of the trajectory crossing grid cell boundaries, i.e. $n > 0$. As a start, we construct the trajectory with a_0 and a_n set to zero. The time interval τ is computed as in (3.53), i.e. for small a_0. For the piecewise definition of trajectory for cell $[x_n, x_{n+1}]$, we use (3.21). Then, we can build the trajectory as follows

$$\varphi(x,t^n,\Delta t) = \varphi(x_n, t^n + \tau + T, \Delta t - T - \tau) \tag{3.73}$$

$$= b_n(\Delta t - T - \tau) \tag{3.74}$$

$$= b_n(\Delta t - T) - b_n\frac{\Delta x_0 - x}{u_0} \tag{3.75}$$

$$= u_n(\Delta t - T) - u_n\frac{\Delta x_0 - x}{u_1}. \tag{3.76}$$

For the derivation of the trajectory φ where both a_0 and a_n are not equal to zero but small, we start with the trajectory given in (3.60) for small a_0. In that equation the

division by a_0 is already avoided. Then, a series expansion about $a_n = 0$ is applied and we obtain the final solution for $\widetilde{\varphi}$

$$\begin{aligned}\widetilde{\varphi}(x,t^n,\Delta t) = {} & u_n(\Delta t - T) + \frac{1}{2}u_n a_n(\Delta t - T)^2 + \frac{1}{6}u_n a_n^2(\Delta t - T)^3 \\ & -\frac{1}{2}\frac{u_n \Delta x_0}{u_1}\left((\Delta t - T)^2 a_n^2 + 2(\Delta t - T)a_n + 2\right)x \\ & -\frac{u_n}{4u_1^2}\Delta x_0^2\Big((\Delta t - T)^2 a_0 a_n^2 + 2(\Delta t - T)a_0 a_n \\ & \quad -2(\Delta t - T)a_n^2 + 2a_0 - 2a_n\Big)x^2 \\ & -\frac{1}{6}\frac{u_n \Delta x_0^3}{u_1^3}\Big((\Delta t - T)^2 a_0^2 a_n^2 + 2(\Delta t - T)a_0^2 a_n \\ & \quad -3(\Delta t - T)a_0 a_n^2 + 2a_0^2 - 3a_0 a_n + a_n^2\Big)x^3 \\ & +\frac{1}{24}\frac{u_n a_n a_0 \Delta x_0^4}{u_1^4}\Big(11(\Delta t - T)a_0 a_n + 11a_0 - 6a_n\Big)x^4 \\ & -\frac{7}{24}\frac{u_n a_n^2 a_0^2 \Delta x_0^5}{u_1^5}x^5 + \mathcal{O}\left(a_0^3\right) + \mathcal{O}\left(a_n^3\right)\end{aligned} \tag{3.77}$$

For clarity, we sum up the expression with abbreviations, omit higher order terms and obtain

$$\widetilde{\varphi}(x,t^n,\Delta t) = d_1 + d_2 x + d_3 x^2 + d_4 x^3 + d_5 x^4 + d_6 x^5 \tag{3.78}$$

$$= \sum_{i=1}^{6} d_i x^{i-1}. \tag{3.79}$$

To show the relation of the (3.77) with small a_0 and a_n and (3.76) with these coefficients equal to zero, we set a_0 and a_n to zero in (3.77) and resubstitute with

$$x = \frac{-\eta + \Delta x_0}{\Delta x_0}. \tag{3.80}$$

We recover exactly the trajectory in (3.76).

So far, we have constructed trajectories for special cases of a_k, i.e. $|a_k| < \epsilon$, with either $0 < k < n$, or $k = 0$, or $k = n$, or with the last case $|a_0| < \epsilon$ and $|a_n| < \epsilon$, where the departure and arrival point of the trajectory lies within a cell with (nearly) constant velocity. All trajectories above are constructed for positive velocity u. The analog treatment of the trajectories with small a_k for negative velocity is shown subsequently.

3.2.2 Trajectory with negative velocity

The equations that describe trajectories which remain within one grid cell in (3.19) and (3.21) hold for positive and negative velocity and can therefore be used for this section as well.

The construction of the trajectories with negative velocity is carried out in the same way as for positive velocity, with small differences. We start with a point x, follow the trajectory to the cell boundary x_1, that is now the left boundary instead of to the right cell boundary as for positive velocity. We advance from cell boundary to cell boundary, computing the intermediate time intervals t_k, until the nth boundary. From there, the arrival point of the trajectory is determined. The procedure is illustrated in Figure 3.4.

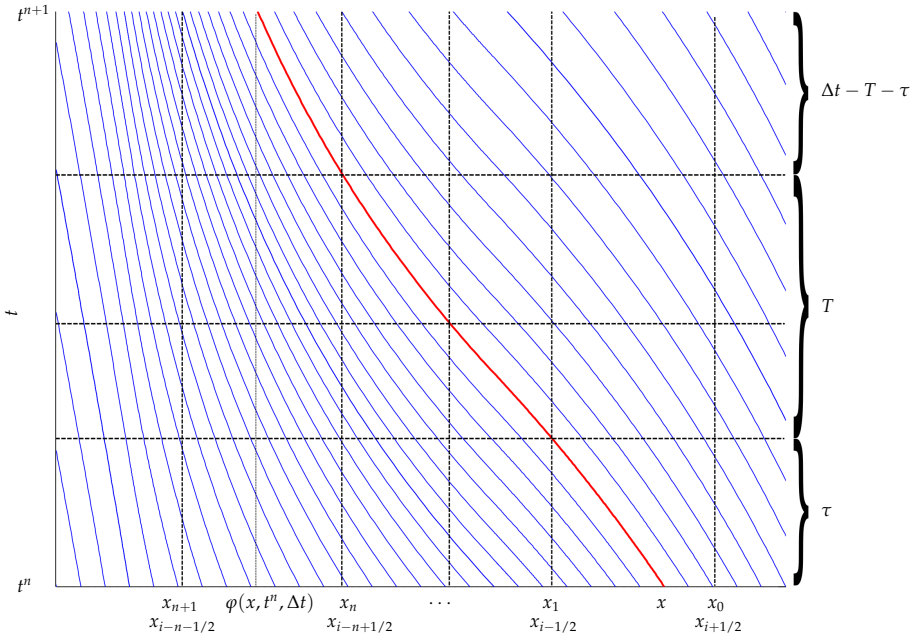


Figure 3.4: Example of trajectories computed from a negative velocity field. The red colored trajectory is $\varphi(x, t^n, t)$, with starting point x at time t^n. It crosses n grid cell boundaries (here $n = 3$).

The main difference of the construction of the trajectories with positive and negative velocity is the redefinition of the cell boundaries with local cell values. For positive velocity we have $x_k < x_{k+1}$ for $k = 0, \ldots, n$ and thus we identify x_k with zero and x_{k+1} with Δx_k. For negative velocity, $x_{k+1} < x_k$ and therefore the redefinition is vice versa, x_k is set to Δx_k and x_{k+1} to zero.

The time interval to proceed from starting point x to the first boundary x_1 is given by τ. We use (3.19) and obtain

$$\varphi(x, t^n, \tau) = x_1 \tag{3.81}$$

$$\iff e^{a_0 \tau} \left(\frac{b_0}{a_0} + x \right) - \frac{b_0}{a_0} = x_1. \tag{3.82}$$

We solve (3.82) for τ, set $x_1 = 0$ and obtain the solution

$$\tau = -\frac{1}{a_0} \ln \left(\frac{x + \frac{b_0}{a_0}}{\frac{b_0}{a_0}} \right) \tag{3.83}$$

$$= -\frac{1}{a_0} \ln \left(\frac{a_0}{u_0} x + 1 \right). \tag{3.84}$$

Next, we compute time t_k that describes the duration of the trajectory to cross the kth cell, from boundary x_k to boundary x_{k+1},

$$\varphi(x_k, t^n + \tau + t_1 + \cdots + t_{k-1}, t_k) = x_{k+1} \tag{3.85}$$

$$\iff e^{a_k t_k} \left(\frac{b_k}{a_k} + \Delta x_k \right) - \frac{b_k}{a_k} = 0. \tag{3.86}$$

Solving for t_k, we have

$$t_k = \frac{1}{a_k} \ln \left(\frac{u_k}{u_{k+1}} \right). \tag{3.87}$$

With the computations of τ and t_k for $i = 1, \dots, n-1$ we assemble the trajectory crossing n cell boundaries ($n > 0$), using (3.28)

$$\begin{aligned}
&\varphi(x_n, t^n + T + \tau, \Delta t - \tau - T) \\
&\quad = \exp\left(a_n(\Delta t - \tau - T)\right)\left(\frac{b_n}{a_n} + \Delta x_n\right) - \frac{b_n}{a_n} \\
&\quad = \frac{u_{n+1}}{a_n}\exp\left(a_n(\Delta t - T)\right)\left(\frac{a_0}{u_0}x + 1\right)^{\frac{a_n}{a_0}} - \frac{u_n}{a_n}.
\end{aligned} \tag{3.88}$$

Similar to the trajectory constructed from positive velocity, we use a substitution. Hence, we substitute $x \in [0, \Delta x_0]$ with

$$x = \Delta x_0 \eta. \tag{3.89}$$

It yields for $\eta \in [0, 1]$

$$\widetilde{\varphi}(\eta, t^n, \Delta t) = \underbrace{\frac{u_{n+1}}{a_n}\exp\left(a_n(\Delta t - T)\right)}_{c_1}\left(\underbrace{\frac{a_0}{u_0}\Delta x_0}_{c_2}\eta + 1\right)^{\overbrace{\frac{a_n}{a_0}}^{c_3}} - \underbrace{\frac{u_n}{a_n}}_{c_4}. \tag{3.90}$$

With the abbreviations defined above, the trajectory crossing n cell boundaries constructed from a negative velocity field is given by

$$\widetilde{\varphi}(x, t^n, \Delta t) = c_1(c_2 x + 1)^{c_3} - c_4, \quad \text{for } x \in [0, 1]. \tag{3.91}$$

The trajectories with positive and negative velocity are of the same form, compare to (3.44).

Negative velocity: $|a_k| < \epsilon$

Analog to the case $|a_k| < \epsilon$ for positive velocity, the case for negative velocity is treated. First, we derive the expression for t_k if $a_k = 0$, then look at t_k for nonzero but small a_k. Hence, for $a_k = 0$, we obtain the time interval t_k with the equation

$$\varphi(x_k, t^n + \tau + t_1 + \cdots + t_{k-1}, t_k) = x_{k+1}, \tag{3.92}$$

and by solving it for t_k

$$t_k = -\frac{\Delta x_k}{u_k}. \tag{3.93}$$

Further, for small a_k we have

$$t_k = \frac{1}{a_k}\ln\left(\frac{u_k}{u_{k+1}}\right) \tag{3.94}$$

$$= -\frac{1}{a_k}\ln\left(\frac{u_{k+1}}{u_k}\right) \tag{3.95}$$

$$= -\frac{1}{a_k}\ln\left(\frac{a_k\Delta x_k + u_k}{u_k}\right) \tag{3.96}$$

$$= -\frac{1}{a_k}\ln\left(1 + \frac{a_k\Delta x_k}{u_k}\right) \tag{3.97}$$

$$= -\frac{\Delta x_k}{u_k} + \frac{a_k}{2}\left(\frac{\Delta x_k}{u_k}\right)^2 - \frac{a_k^2}{3}\left(\frac{\Delta x_k}{u_k}\right)^3 + \frac{a_k^3}{4}\left(\frac{\Delta x_k}{u_k}\right)^4 - \mathcal{O}\left(a_k^4\right). \tag{3.98}$$

Setting a_k to zero in (3.98), we obtain (3.93).

Negative velocity: $|a_0| < \epsilon$

For $|a_0| < \epsilon$, we first consider the case of constant velocity in cell $[x_0, x_1]$. We compute τ with the ansatz

$$\varphi(x, t^n, \tau) = x_1. \tag{3.99}$$

Using (3.21) as the piecewise definition for the trajectory in cell $[x_0, x_1]$, we obtain the time interval

$$\tau = -\frac{x}{u_0}. \tag{3.100}$$

Analogously to the construction of φ that crosses more than one grid cell (3.38) - (3.40), we can build the trajectory as follows

$$\begin{aligned}
\varphi(x, t^n, \Delta t) &= \varphi(x_n, t^n, \Delta t - T - \tau) && (3.101)\\
&= \exp(a_n(\Delta t - T - \tau)) \left(\frac{b_n}{a_n} + \Delta x_n\right) - \frac{b_n}{a_n} && (3.102)\\
&= \frac{u_{n+1}}{a_n} \exp(a_n(\Delta t - T)) \exp\left(a_n \frac{x}{u_0}\right) - \frac{u_n}{a_n}. && (3.103)
\end{aligned}$$

Next, we address the case of nonzero, but small a_0. The trajectory φ for negative velocity is given by (3.90). We apply series expansion about $a_0 = 0$ and so we use the following trajectory for $a_0 < \epsilon$

$$\begin{aligned}
&\widetilde{\varphi}(x, t^n, \Delta t)\\
&= \frac{u_{n+1}}{a_n} \exp\left(a_n(\Delta t - T)\right) \exp\left(\frac{a_n \Delta x_0}{u_0} x\right) \left(1 - \frac{\Delta x_0^2 a_0 a_n}{2u_0^2} x^2\right.\\
&+ \frac{\Delta x_0^3 a_0^2 a_n}{3u_0^3} x^3 + \left(-\frac{\Delta x_0^4 a_0^3 a_n}{4u_0^4} + \frac{\Delta x_0^4 a_0^2 a_n^2}{8u_0^4}\right) x^4\\
&\left. - \frac{\Delta x_0^5 a_0^3 a_n^2}{6u_0^5} x^5 - \frac{\Delta x_0^6 a_0^3 a_n^3}{48u_0^6} x^6\right) - \frac{u_n}{a_n} + \mathcal{O}\left(a_0^4\right).
\end{aligned} \tag{3.104}$$

Using abbreviations and omitting higher order terms, we can rewrite the equation as

$$\begin{aligned}
\widetilde{\varphi}(x, t^n, \Delta t) =& d_1 \exp(d_2\, x)(1 + d_3 x^2 + d_4 x^3 + d_5 x^4 + d_6 x^5 + d_7 x^6)\\
&+ d8
\end{aligned} \tag{3.105}$$

If we set a_0 to zero in (3.104) and resubstitute x by division of Δx_0, we obtain the trajectory constructed above for that special case (3.103).

Negative velocity: $|a_n| < \epsilon$

We set a_n to zero and derive the trajectory φ for that case. The time interval τ for the first cell remains unchanged and is given by (3.83). We use the trajectory for constant velocity given by (3.21) for cell $[x_n, x_{n+1}]$ to assemble the whole trajectory

$$\begin{aligned}
\varphi(x, t^n, \Delta t) &= \varphi(x_n, t^n + \tau + T, \Delta t - T - \tau) && (3.106)\\
&= \Delta x_n + b_n(\Delta t - T - \tau) && (3.107)\\
&= \Delta x_n + u_n(\Delta t - T) + \frac{u_n}{a_0} \ln\left(\frac{a_0}{u_0} x + 1\right). && (3.108)
\end{aligned}$$

The trajectory φ for negative velocity is given by (3.90). As a_n goes to zero we apply series expansion about $a_n = 0$. We end with the form

$$\begin{aligned}
\widetilde{\varphi}(x,t^n,\Delta t) &= \frac{a_n^3 u_n}{24a_0^4}\ln\left(\frac{\Delta x_0 a_0}{u_0}x+1\right)^4 \\
&+\left(\frac{a_n^2 u_n}{6a_0^3}+\frac{((\Delta t-T)u_n+\Delta x_0)a_n^3}{6a_0^3}\right)\ln\left(\frac{\Delta x_0 a_0}{u_0}x+1\right)^3 \\
&+\left(\frac{a_n u_n}{2a_0^2}+\frac{((\Delta t-T)u_n+\Delta x_0)a_n^2}{2a_0^2}\right. \\
&\left.+\frac{\left(\frac{1}{2}u_n(\Delta t-T)^2+\Delta x_0(\Delta t-T)\right)a_n^3}{2a_0^2}\right)\ln\left(\frac{\Delta x_0 a_0}{u_0}x+1\right)^2 \\
&+\left(\frac{u_n}{a_0}+\frac{((\Delta t-T)u_n+\Delta x_0)a_n}{a_0}+\frac{\left(\frac{1}{2}u_n(\Delta t-T)^2+\Delta x_0(\Delta t-T)\right)a_n^2}{a_0}\right. \\
&\left.+\frac{\left(\frac{1}{6}u_n(\Delta t-T)^3+(1/2)\Delta x_0(\Delta t-T)^2\right)a_n^3}{a_0}\right)\ln\left(\frac{\Delta x_0 a_0}{u_0}x+1\right) \\
&+u_n(\Delta t-T)+\Delta x_0+\left(\frac{1}{2}u_n(\Delta t-T)^2+\Delta x_0(\Delta t-T)\right)a_n \\
&+\left(\frac{1}{6}u_n(\Delta t-T)^3+\frac{1}{2}\Delta x_0(\Delta t-T)^2\right)a_n^2 \\
&+\left(\frac{1}{24}u_n(\Delta t-T)^4+\frac{1}{6}\Delta x_0(\Delta t-T)^3\right)a_n^3+\mathcal{O}\left(a_n^4\right).
\end{aligned} \tag{3.109}$$

The short form of (3.109) with adequate abbreviations and d_k, and without higher order terms, is given by

$$\begin{aligned}
\widetilde{\varphi}(x,t^n,\Delta t) &= d_1+d_2\ln(d_6x+1)+d_3\ln^2(d_6x+1) \\
&\quad +d_4\ln^3(d_6x+1)+d_5\ln^4(d_6x+1).
\end{aligned} \tag{3.110}$$

$$= \sum_{k=1}^{5} d_k \ln(d_6x+1)^{k-1}. \tag{3.111}$$

If we set a_n to zero in (3.109) and resubstitute by division of Δx_0, we obtain the trajectory derived for constant velocity in the nth cell in (3.108).

Negative velocity: $|a_0| < \epsilon$ ***and*** $|a_n| < \epsilon$

The last case to be considered is the case of a_0 and a_n being small.

For the case $n = 0$, which means that the trajectory remains within one grid cell, we recall that the equation describing the trajectory that does not cross grid cell boundaries given in (3.19), holds for positive and negative velocity. Thus, we refer to the approximation given in (3.72).

We assume $n > 0$. We start to construct the trajectory if we set a_0 and a_n to zero. The time interval τ is computed as in (3.53). For the piecewise definition of trajectory for cell $[x_n, x_{n+1}]$, we use (3.21). Then we can build the trajectory as follows

$$\varphi(x,t^n,\Delta t) = \varphi(x_n, t^n+\tau+T, \Delta t-T-\tau) \tag{3.112}$$

$$= \Delta x_n + b_n(\Delta t-T-\tau) \tag{3.113}$$

$$= \Delta x_n + u_n(\Delta t-T)+u_n\frac{x}{u_0}. \tag{3.114}$$

Assuming the coefficients a_0 and a_n deviate from zero but are still small, we obtain based on (3.90) after series expansions about $a_0 = 0$ and $a_n = 0$ the following approximation,

$$\begin{aligned}
\widetilde{\varphi}(x,t^n,\Delta t) = &\left(\frac{7a_0^2a_n^2\Delta x^5u_n}{24u_0^5}\right)x^5 + \frac{a_0a_n\Delta x^4}{24u_0^4}\Bigg(11(\Delta t-T)a_0a_nu_n\\
&+11\Delta x a_0a_n + 11a_0u_n - 6a_nu_n\Bigg)x^4\\
&+\frac{\Delta x^3}{6u_0^3}\Bigg((\Delta t-T)^2a_0^2a_n^2u_n + 2(\Delta t-T)\Delta x a_0^2a_n^2 + 2(\Delta t-T)a_0^2a_nu_n\\
&-3(\Delta t-T)a_0a_n^2u_n + 2\Delta x a_0^2a_n - 3\Delta x a_0a_n^2 + 2a_0^2u_n\\
&-3a_0a_nu_n + a_n^2u_n\Bigg)x^3 - \frac{\Delta x^2}{4u_0^2}\Bigg((\Delta t-T)^2a_0a_n^2u_n\\
&+2(\Delta t-T)\Delta x a_0a_n^2 + 2(\Delta t-T)a_0a_nu_n - 2(\Delta t-T)a_n^2u_n\\
&+2\Delta x a_0a_n - 2\Delta x a_n^2 + 2a_0u_n - 2a_nu_n\Bigg)x^2\\
&+\frac{\Delta x}{2u_0}\Bigg((\Delta t-T)^2a_n^2u_n + 2(\Delta t-T)\Delta x a_n^2 + 2(\Delta t-T)a_nu_n\\
&+2\Delta x a_n + 2u_n\Bigg)x + (\Delta t-T)u_n + \Delta x + \frac{1}{6}a_n^2(\Delta t-T)^3u_n\\
&+\frac{1}{2}a_n(\Delta t-T)^2u_n + \frac{1}{2}a_n^2(\Delta t-T)^2\Delta x + a_n(\Delta t-T)\Delta x\\
&+\mathcal{O}\left(a_0^3\right) + \mathcal{O}\left(a_n^3\right).
\end{aligned} \tag{3.115}$$

With abbreviations of some expressions and omitting higher order terms, we can express the trajectory as

$$\widetilde{\varphi}(x,t^n,\Delta t) = \sum_{i=1}^{6} d_i x^{i-1}. \tag{3.116}$$

3.2.3 Zero velocity at grid points

In the preceding sections we examined the trajectories for small coefficients a_j, that is a small slope of the velocity distribution. However, the velocity itself can be small or zero as well. The impact of zero velocity on the trajectories is studied in this section.

Zero velocity at grid points influences the trajectories in different ways depending on the respective location. We examine the problem divided into three cases: first constant zero velocity, second we consider positive velocity except at the left or right grid cell boundary, where we set either $u(x_{i-1/2})$ or $u(x_{i+1/2})$ to zero, and third we observe $u(x_{i-1/2}) = 0$ or $u(x_{i+1/2}) = 0$ for otherwise negative velocity. The influence of zero velocity is exemplarily described for grid cell i, where either the starting or the endpoint of a trajectory lies, or both if the trajectory remains within the ith grid cell. A last example shows that the construction of a trajectory is possible that crosses grid cell boundaries, with a zero velocity cell boundary in the departure and arrival grid cell.

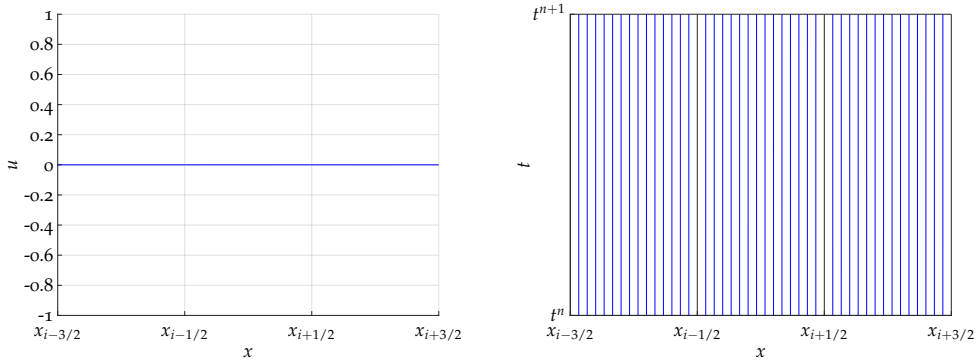
The first case of constant zero velocity leads to trajectories that remain within one grid cell. They are described by (3.19) for the ith grid cell and simplify to

$$\varphi(x,t^n,\Delta t) = \exp(a_i\Delta t)x \tag{3.117}$$

for $u_i = 0$. (3.19) contains the coefficient u_i only in the dividend. Thus, $u_i = 0$ is not critical and (3.117) is obtained directly. If u_{i+1} is equal to zero as well, which means constant zero velocity in grid cell i, the coefficient a_i equals zero, too. Then, the trajectories are simply

$$\varphi(x, t^n, \Delta t) = x. \tag{3.118}$$

With zero velocity the tracer remains at the same spot over time and is not advected to a different point in space.



(a) The velocity u is contantly equal to zero.

(b) The trajectories are straight lines perpendicular to the x-axis.

Figure 3.5: The velocity u equal to zero is shown with the corresponding trajectories.

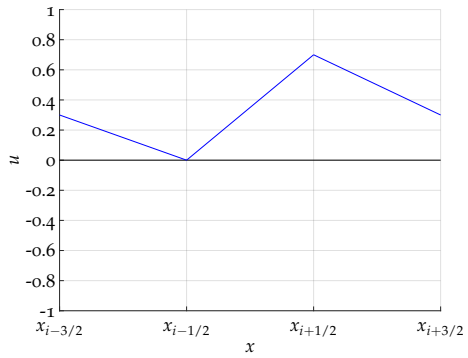
Figure 3.5 shows the case of constant zero velocity, here in three neighboring grid cells $i-1$, i and $i+1$. The according trajectories are straight lines perpendicular to the x-axis. In that way, a tracer that is advected along this path stays at the same point in space.

We turn to the second case of positive velocity except at either the left or right grid cell boundary of grid cell i. We remind of the trajectory given by (3.43) for positive velocity that crosses grid cell boundaries,

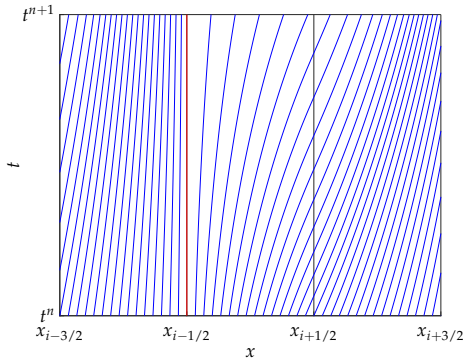
$$\widetilde{\varphi}(\eta, t^n, \Delta t) = \frac{u_n}{a_n} \exp\left(a_n(\Delta t - T)\right) \left(-\frac{a_0}{u_1}\Delta x_0 \eta + 1\right)^{\frac{a_n}{a_0}} - \frac{u_n}{a_n}.$$

The velocity coefficient u_j appears in the equation in the divisor. To avoid division by zero close consideration of the velocity is needed. The value u_1 must not be equal to zero. To understand which local index refers to which global index, and thus where the zero velocity plays a role, we look at Figure 3.6. The panels 3.6(a) and 3.6(c) show the velocity fields with positive velocity with $u(x_{i-1/2}) = 0$ and $u(x_{i+1/2}) = 0$, respectively. We study the corresponding trajectories.

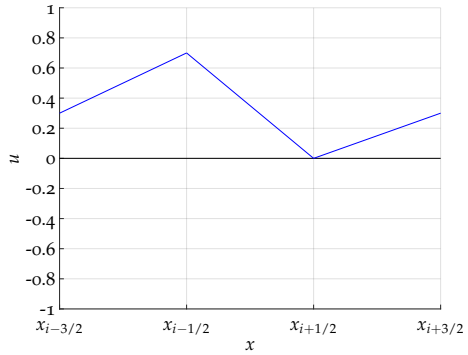
Figure 3.6(b) displays the trajectories that are associated with the velocity distribution given in Figure 3.6(a). The red colored trajectory is located at the zero velocity point, i.e. at $x_{i-1/2}$, and is given by $\varphi(x_{i-1/2}, t^n, \Delta t) = x_{i-1/2}$. When we look at grid cell i, we note that trajectories only depart from this grid cell. No trajectory crosses the left cell boundary to arrive in this cell. Obviously, the zero velocity point cannot be trespassed. The trajectories that remain within the ith grid cell need no further consideration as their case is discussed above. The trajectories that cross the right grid cell boundary are described by (3.43). Their departure cell is the ith grid cell and they end in the neighboring grid cell $i+1$. That means that the velocity at the left boundary of cell i is expressed in local notation as u_0, the right boundary as u_1. The velocity at the point $x_{i+1/2}$ is also denoted as u_n, because it is the last boundary that is crossed before the trajectory ends in grid cell $i+1$. The right grid cell of the arrival grid cell $i+1$ is assigned to u_{n+1}. The value $u_1 = u_n$ cannot take the value zero, as the trajectory crosses the boundary at $x_{i+1/2}$. The velocity coefficient



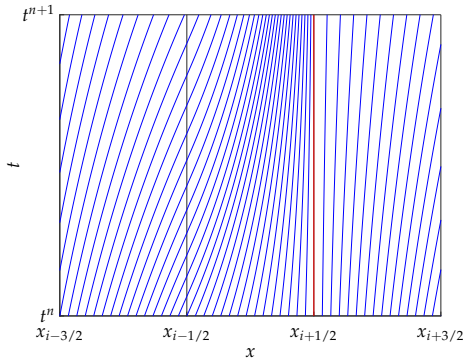
(a) Positive velocity, where $u(x_{i-1/2}) = 0$.



(b) Corresponding trajectories to 3.6(a). The red trajectory marks the zero velocity point.



(c) Positive velocity, where $u(x_{i+1/2}) = 0$.



(d) Corresponding trajectories to 3.6(c). The red trajectory marks the zero velocity point.

Figure 3.6: Different scenarios for the positive velocity u to approach the zero point.

u_0 which is equal to zero, does not affect the trajectory as it is not contained in the expression (3.43).

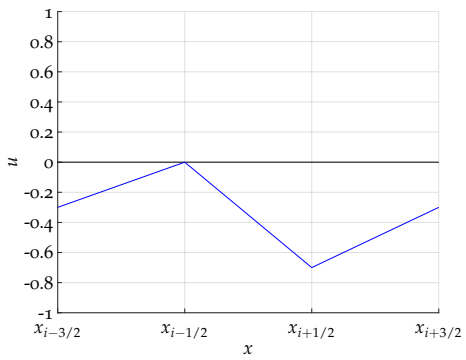
Studying the trajectories that cross a grid cell boundary in panel 3.6(d), we note that trajectories only end in the ith grid cell, but do not depart from it. The red colored trajectory corresponds to the zero velocity point at the right grid cell boundary at $x_{i+1/2}$. The assigment of the global to the local indices yields that the velocity at $x_{i-3/2}$, that is the left boundary of grid cell $i-1$, is matched to u_0, the global point $x_{i-1/2}$ denotes u_1 in local indices. This point is also the last boundary to be crossed, thus it is also marked as u_n. The right cell boundary of cell i is denoted as u_{n+1}, which is equal to zero. As in the case above the values $u_1 = u_n$ cannot be equal to zero. The coefficient $u(x_{i+1/2}) = u_{n+1}$, which is equal to zero, does not effect the trajectory because the coefficient does not appear in the equation of the trajectory (3.43).

Similarly, the procedure follows for the negative velocity. The trajectories that correspond to a negative velocity and cross through grid cells are described by (3.90), repeated here

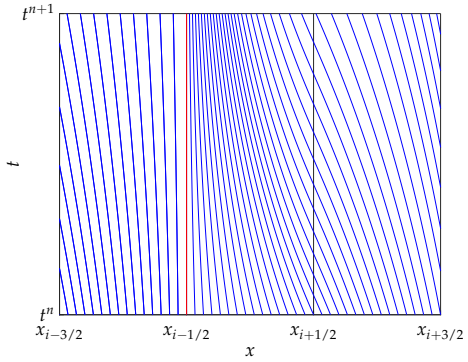
$$\widetilde{\varphi}(\eta, t^n, \Delta t) = \frac{u_{n+1}}{a_n} \exp\left(a_n(\Delta t - T)\right) \left(\frac{a_0}{u_0}\Delta x_0 \eta + 1\right)^{\frac{a_n}{a_0}} - \frac{u_n}{a_n}.$$

In this equation the value u_0 must not be equal to zero. We assign the global indices to the local ones by means of Figure 3.7. Figures 3.7(a) and 3.7(c) show the negative velocity distributions with the root at $x_{i-1/2}$ and $x_{i+1/2}$, respectively.

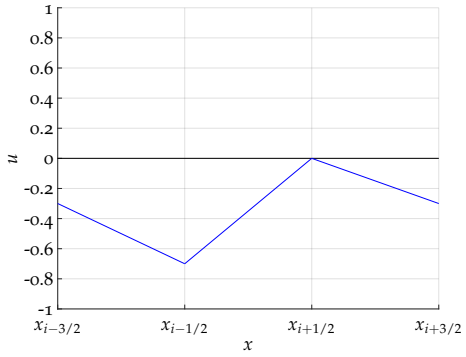
The corresponding trajectories to the velocity shown in Figure 3.7(a) are displayed in Figure 3.7(b). The trajectory that starts at the zero velocity point is plotted in red



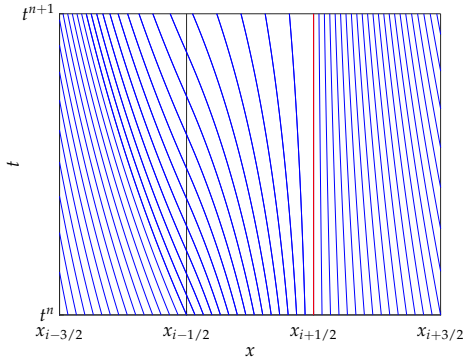
(a) Negative velocity, where $u(x_{i-1/2}) = 0$.



(b) Corresponding trajectories to 3.7(a). The red trajectory marks the zero velocity point.



(c) Negative velocity, where $u(x_{i+1/2}) = 0$.



(d) Corresponding trajectories to 3.7(c). The red trajectory marks the zero velocity point.

Figure 3.7: Different scenarios for the negative velocity u to approach the zero point.

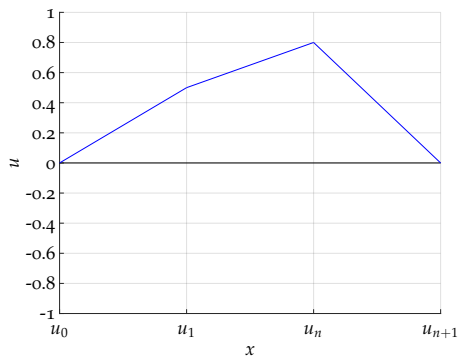
for emphasis. Studying the trajectories in the ith grid cell we note that all trajectories that start in this grid cell also end there. No trajectory with a starting point there crosses a boundary and is thus described by (3.19). However, the trajectories with the departure cell $i+1$ and arrival cell i are given by (3.90). We examine the indices for these trajectories. We point out that after matching the global to the local indices the relation of neighboring velocity values (3.17) still must hold locally for that grid cell. Therefore, the left boundary of the departure cell $i+1$ holds the value u_0 and the right boundary of the departure cell the value u_1. In that way, it is guaranteed that $u_1 = a_0\Delta x_0 + u_0$. In the case of negative velocity, the first boundary that is crossed is the one with velocity u_0. Analogously, the assigment follows for the arrival cell i. The left boundary takes the value u_n and the right boundary u_{n+1}. We can state that the value $u(x_{i+1/2}) = u_0 = u_{n+1}$ cannot be equal to zero, since the trajectory passes the point $x_{i+1/2}$. Note that the values of the velocity u_0 and u_{n+1} are equal for the case of the trajectory crossing one grid cell boundary, i.e. $n = 1$. Since these values are nonzero, the description of the trajectory (3.90) holds. The value u_n is set to zero in this example, but has no effect on the trajectories with arrival point in grid cell i.

In the second example shown in Figures 3.7(c) and 3.7(d), the velocity is set to zero at point $x_{i+1/2}$. The trajectory indicating the zero velocity starting at that point is colored in red. We examine the trajectories with departure point in grid cell i, passing the boundary at $x_{i-1/2}$ with arrival cell $i-1$. In this case, the left boundary of the ith cell, which is the departure cell, takes the value u_0 and the right one u_1. For the arrival cell $i-1$, the velocity at the left boundary at $x_{i-3/2}$ is equal to u_n

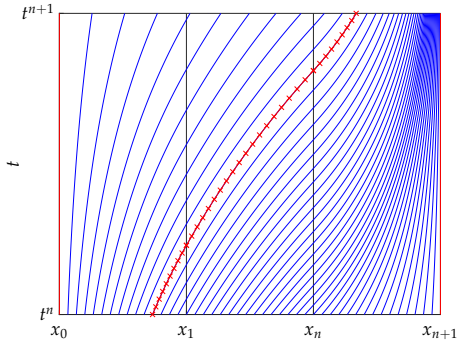
and at the right boundary u_{n+1}. Again, the values u_0 and u_{n+1} cannot be zero. The trajectory (3.90) is well-defined.

Summarized, we observe that for positive velocity only the left boundary of the departure cell and only the right boundary of the arrival cell can be zero. For negative velocity it is vice versa, in departure cells the right boundary can be of zero velocity and the left boundary of arrival cells. The trajectories for all scenarios can be described by (3.43) and (3.90), respectively, without the risk of division by zero.

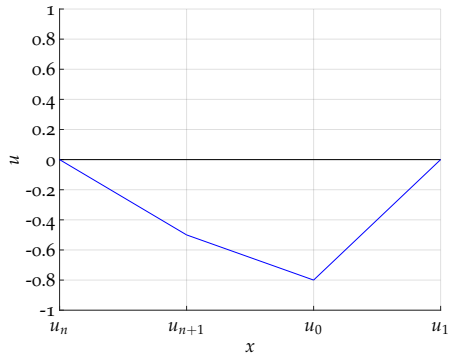
This section concludes with two examples to show the possibility that trajectories which pass through more than one grid cell can depart from a cell with a zero velocity cell boundary and arrive at such a cell - for positive and negative velocity.



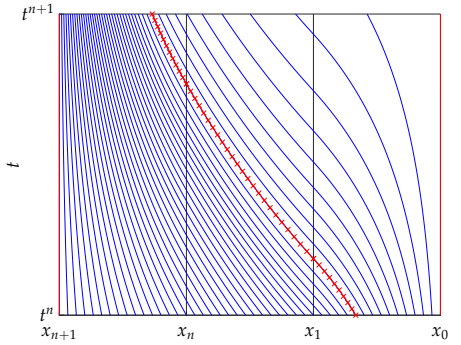
(a) Positive velocity distribution with zero velocity at the left and right boundary.



(b) Trajectories corresponding to 3.8(a).



(c) Negative velocity distribution with zero velocity at the left and right boundary.



(d) Trajectories corresponding to 3.8(c).

Figure 3.8: Examples for zero velocity at boundaries at departure and arrival grid cell.

Figure 3.8 shows the construction of these trajectories. Panel 3.8(a) shows the distribution for positive velocity, Figure 3.8(c) the negative velocity field. Both functions are equal to zero at the left boundary of the left grid cell shown on the panels and at the right boundary at the right cell. In the Figures 3.8(b) and 3.8(b) we can see the corresponding trajectories. At the very left and right the trajectories colored in red highlight the zero velocity points. The trajectories colored in red and marked with crosses, stand exemplarily for trajectories that start and end in a grid cell with a zero velocity boundary. The labels of the x-axis for the velocity as well as for the trajectories are written using the local indices, that are associated with these trajectories. All trajectories that cross cell boundaries are computed and plotted using (3.43) and (3.90), respectively, equations that are well-defined for all cases of zero velocity coefficients that can occur.

3.2.4 Overview of different types of trajectories

This section summarizes the types of trajectories that can occur. The trajectory (3.19) that remains within one grid cell holds for positive and negative velocity

$$\varphi(x,t^n,\Delta t) = e^{a_i\Delta t}\left(\frac{b_i}{a_i}+x\right)-\frac{b_i}{a_i}. \tag{3.119}$$

The trajectories crossing $n \geq 1$ cell boundaries can be expressed as (3.44) for positive velocity and for negative velocity (3.91) and are of the form

$$\widetilde{\varphi}(x,t^n,\Delta t) = c_1(c_2x+1)^{c_3}-c_4, \tag{3.120}$$

with adequate abbreviations c_k that differ for positive and negative velocity.

To avoid division by zero or small numbers, we derived approximations as a remedy. Thus, for small a_0 we have (3.61) for positive velocity and (3.105) for negative velocity, which are of the same type with different suitable abbreviations d_k,

$$\begin{aligned}\widetilde{\varphi}(x,t^n,\Delta t) &= d_1\exp(d_2x)\Big(1+d_3x^2+d_4x^3+d_5x^4+d_6x^5\\ &\quad +d_7x^6\Big)+d_8\\ &= d_1\exp(d_2x)\left(1+\sum_{k=1}^{5}d_{k+2}x^{k+1}\right)+d_8.\end{aligned} \tag{3.121}$$

If the coefficient a_n is small, the Taylor expansion about $a_n = 0$ leads to (3.69) for positive and to (3.111) for negative velocity. Again, the resulting trajectory is of the same type with respective abbreviations d_k,

$$\widetilde{\varphi}(x,t^n,\Delta t) = \sum_{k=1}^{5} d_k\ln(dx+1)^{k-1}. \tag{3.122}$$

If both coefficients a_0 and a_n are nearly or equal to zero we obtain (3.79) for positive velocity and (3.116) for negative velocity. These are of the type

$$\begin{aligned}\widetilde{\varphi}(x,t^n,\Delta t) &= d_1+d_2x+d_3x^2+d_4x^3+d_5x^4+d_6x^5\\ &= \sum_{i=1}^{6} d_ix^{i-1},\end{aligned} \tag{3.123}$$

with different d_k for positive and negative velocity.

In the Section 3.2 we computed the trajectories that are the solutions to the ODE (3.18). We obtained different expressions for the trajectories: for the trajectories that remain within one grid cell and for the trajectories that cross cell boundaries. For the latter case, we studied the solutions for small coefficients of the slope of the velocity to avoid division by zero. Further, we examined the trajectories for zero velocity at grid points and found out that zero velocity causes no restrictions on the expressions. Last of all, we summarized the different types of trajectories.

3.3 THE EXACT SOLUTION AND ITS INTEGRAL

The previous section is devoted to compute the trajectories that are the solution to (3.4). They are constructed from a positive and a negative velocity field. Further, the issue of division by (nearly) zero coefficients arising in the equations is addressed and resolved by series expansions. To find the solution to the linear advection equation, the initial value problem (3.3) must be solved.

Proposition 3.3.1 *By separation of variables we can determine the solution to the initial value problem (3.3) at time t^{n+1}*

$$\rho(x,t^{n+1}) = \rho(\varphi(x,t^{n+1},-\Delta t),t^n)\exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x}u(\varphi(x,t^{n+1},t-\Delta t),t)dt\right). \tag{3.124}$$

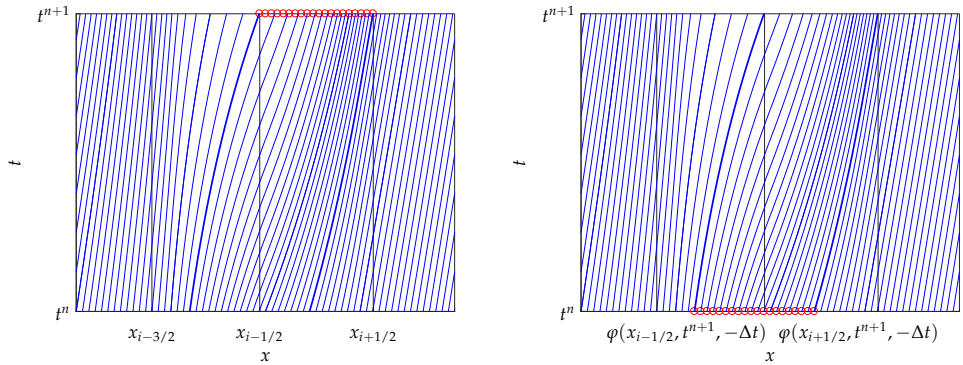
Details can be found in the appendix A.1.

As for all finite volume methods, we are interested in the integral value of ρ for each grid cell,

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \rho(x,t^{n+1}) = \int_{x_{i-1/2}}^{x_{i+1/2}} \rho(\varphi(x,t^{n+1},-\Delta t),t^n) \cdot \exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x}u(\varphi(x,t^{n+1},t-\Delta t),t)dt\right) \mathrm{d}x\,. \tag{3.125}$$

It is possible to compute the integral of the argument of the exponential in (3.125) and solve the initial value problem (3.3) that way. However, before we explicitly compute (3.125) we bring to mind that because of the conservation property the integral of ρ over the ith cell at time t^{n+1} is equal to the integral of ρ at time t^n over the interval spanned by the departure points of the trajectories starting at the cell boundaries of grid cell i.

Figure 3.9 shows the two possibilities. The first option is to compute the integral over the interval of the ith grid cell $[x_{i-1/2}, x_{i+1/2}]$ at time t^{n+1}. This is illustrated in Figure 3.9(a). The red dotted area corresponds to the interval that is integrated over. Since ρ is not known at t^{n+1}, the solution is obtained by (3.125). The second possibility is pictured in Figure 3.9(b). If the integral is transformed by substitution, the integral of ρ can be directly computed at time t^n. The integration interval changes to $[\varphi(x_{i-1/2},t^{n+1},-\Delta t),\varphi(x_{i+1/2},t^{n+1},-\Delta t)]$.



(a) Integration of ρ at time t^{n+1}. **(b)** Integration of ρ at time t^n.

Figure 3.9: Difference of time levels for the integration of ρ. The red dots symbolize the integration interval. The bold lines are the trajectories $\varphi(x_{i-1/2},t^{n+1},-t)$ and $\varphi(x_{i+1/2},t^{n+1},-t)$, respectively.

Under the additional assumption on u to be independent of time, we transform the integral in (3.125) by substitution with

$$x = \varphi(\eta,t^n,\Delta t). \tag{3.126}$$

The substitution leads to

$$\begin{aligned}\int_{x_{i-1/2}}^{x_{i+1/2}} \rho(x, t^{n+1})\,\mathrm{d}x = &\int_{\varphi^{-1}(x_{i-1/2},t^{n+1},\Delta t)}^{\varphi^{-1}(x_{i+1/2},t^{n+1},\Delta t)} \rho(\varphi(\varphi(\eta, t^n, \Delta t), t^{n+1}, -\Delta t), t^n) \\ &\cdot \exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x} u(\varphi(\varphi(\eta, t^n, \Delta t), t^{n+1}, t - \Delta t))dt\right) \\ &\cdot \frac{\partial}{\partial \eta}\varphi(\eta, t^n, \Delta t)\,\mathrm{d}\eta \end{aligned} \tag{3.127}$$

$$\begin{aligned}= &\int_{\varphi(x_{i-1/2},t^{n+1},-\Delta t)}^{\varphi(x_{i+1/2},t^{n+1},-\Delta t)} \rho(\eta, t^n) \exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x} u(\varphi(\eta, t^n, t))dt\right) \\ &\cdot \frac{\partial}{\partial \eta}\varphi(\eta, t^n, \Delta t)\,\mathrm{d}\eta\,. \end{aligned} \tag{3.128}$$

Now we use the fact that

$$\exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x} u(\varphi(\eta, t^n, t))dt\right) \frac{\partial}{\partial \eta}\varphi(\eta, t^n, \Delta t) = 1, \tag{3.129}$$

which is proven in the appendix in Section A.2.

Hence, with means of the substitution in (3.125) we have obtained

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \rho(x, t^{n+1})\,\mathrm{d}x = \int_{\varphi(x_{i-1/2},t^{n+1},-\Delta t)}^{\varphi(x_{i+1/2},t^{n+1},-\Delta t)} \rho(\eta, t^n)\,\mathrm{d}\eta\,. \tag{3.130}$$

With the trajectory φ we can express the integral of the exact solution to the linear advection equation. After the transformation ρ is evaluated at t^n and corresponds to Figure 3.9(b). Further, (3.130) shows that the integral of ρ is in fact conserved between the boundaries of two trajectories. That property is shared with mass conservative SL schemes. It corresponds to (2.149) for SL schemes.

This approach of computing the integral at time t^n with (3.130) has two advantages over using (3.125). The first one is that we do not need to compute the exponential factor responsible for the change of density as in (3.125). That factor drops out because it is the inverse of the derivative of φ. This improves the performance of the method, as less is to be calculated. The other asset is of importance for the conservation of the integral of density.

We assume that the density ρ at time t^n is of polynomial form, which will be the case in the following. Then, the integral of ρ at t^{n+1} is obtained by integrating over the simple form of a polynomial if (3.130) is used. Before the substitution in (3.125), the integral of ρ at t^{n+1} is computed from the integral over a function that consists of a product of ρ, which is distorted by the argument φ and the factor of the exponential function. The computation of the integral over a polynomial can be performed more accurately, which leads to more accurate conservation.

Note, that in (3.130) the trajectories in the limit of the integral go backward in time, compare to Remark 3.2.2. The details of these trajectories building the limits of the integral are discussed in Section 3.6.

In Section 3.2 we derived the exact solution of the trajectories φ for a piecewise linear velocity field. Thus, taking the exact solution φ to the ODE (3.18) with linear u and the exact solution (3.124) to the ODE (3.3) describing the evolution of the density, we obtain the result for the linear advection equation with piecewise linear velocity. (3.125) as well as (3.130) yield the integral of the exact solution to the linear advection equation (3.1).

3.4 THE PROJECTION STEP

In the previous sections we derived the exact solution to the linear advection equation for one time step under the assumption of a linear velocity field. To obtain this solution we have studied the flow φ that describes the trajectories at which the tracer travels along. The trajectories are the solution to the ODE (3.18). The evolution of the density ρ can also be determined analytically by solving the ODE (3.3). No approximations had been made, except for the series expansion of the trajectories to avoid division by small numbers.

We assume ρ is given at time t^n as a polynomial function for each grid cell. The exact solution at time t^{n+1} can be computed. However, the resulting function is no longer of polynomial form in general. A non-constant velocity field distorts the originally polynomial distribution. Further, due to different grid cell sizes and an arbitrary CFL number, many trajectories with different departure cells can end up in one grid cell leading to a function that can only be piecewise defined. If such a function is used as initial distribution for the next time step, the computation of the exact solution is theoretically possible, but complicated. The projection step enables the reuse of the algorithm to determine the solution of the next time step. It transfers complicated piecewise defined distributions back into polynomial structure. The numerical solution deviates to the true analytical solution only because of the projection step.

The density ρ is given in grid cell i at time t^n in form of a polynomial of degree two following Prather's and van Leer's ansatz,

$$\rho_i(x, t^n) = m^n_{2,i} K^{(i)}_2(x) + m^n_{1,i} K^{(i)}_1(x) + m^n_{0,i} K^{(i)}_0(x), \tag{3.131}$$

where the Legendre polynomials are used as a basis. They are given expressed in local coordinates for the ith grid cell $[0, \Delta x_i]$ by

$$K^{(i)}_0(x) = 1, \tag{3.132}$$

$$K^{(i)}_1(x) = \frac{2x}{\Delta x_i} - 1, \tag{3.133}$$

$$K^{(i)}_2(x) = \frac{6x^2}{\Delta x_i^2} - \frac{6x}{\Delta x_i} + 1. \tag{3.134}$$

Remark 3.4.1 *Note that we discuss the numerical solution of the density ρ only. The computation of the numerical results for the tracer density ρy is carried out in the same way. Since ρ is defined as a polynomial function of degree two, the same holds for ρy. The tracer y, obtained by division of the tracer density by the density, is a rational function.*

If different numerical algorithms are applied for the advection of the density and the tracer, inconsistencies can arise, see e.g. [30]. These can lead to spurious changes in mass, if not treated additionally by special "fixing" as elaborated in [67].

The Legendre polynomials have the property to be L^2-orthogonal

$$\int_{x_{i-1/2}}^{x_{i+1/2}} K^{(i)}_j(x) K^{(i)}_j(x)\, \mathrm{d}x = \frac{\Delta x_i}{2j+1}. \tag{3.135}$$

The projection of the advected density ρ at time t^{n+1} for the interval $[x_{i-1/2}, x_{i+1/2}]$ takes the form

$$\begin{aligned}
&\mathbb{P}\rho_i(x, t^{n+1}) \\
&= \sum_{j=0}^{2} \frac{\int_{x_{i-1/2}}^{x_{i+1/2}} \rho(x, t^{n+1}) K_j^{(i)}(x)\,\mathrm{d}x}{\int_{x_{i-1/2}}^{x_{i+1/2}} K_j^{(i)}(x) K_j^{(i)}(x)\,\mathrm{d}x} K_j^{(i)}(x) \qquad (3.136)\\
&= \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{x_{i-1/2}}^{x_{i+1/2}} \rho(x, t^{n+1}) K_j^{(i)}(x)\,\mathrm{d}x\; K_j^{(i)}(x). \qquad (3.137)
\end{aligned}$$

To compute the integral in the projection (3.137), the unknown density $\rho(x, t^{n+1})$ at time t^{n+1} can either be replaced by (3.130) or it can be transferred to time t^n by applying the substitution (3.126), as shown in the previous section. The latter option is preferred, because the computation of the exponential factor that arises if the first choice is made, is avoided.

The result of the substitution inserted into the projection yields

$$\begin{aligned}
&\mathbb{P}\rho(x, t^{n+1}) \\
&= \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{\varphi(x_{i-1/2}, t^{n+1}, -\Delta t)}^{\varphi(x_{i+1/2}, t^{n+1}, -\Delta t)} \rho(x, t^n) K_j^{(i)}(\varphi(x, t^n, \Delta t))\,\mathrm{d}x\; K_j^{(i)}(x). \qquad (3.138)
\end{aligned}$$

As discussed in the previous section, the conservation of mass is guaranteed. We have for the zeroth coefficient

$$m_{0,i}^{n+1} = \frac{1}{\Delta x_i} \int_{\varphi(x_{i-1/2}, t^{n+1}, -\Delta t)}^{\varphi(x_{i+1/2}, t^{n+1}, -\Delta t)} \rho(x, t^n)\,\mathrm{d}x\,. \qquad (3.139)$$

Even if the computation of the trajectory φ is erroneous, the overall conservation of ρ over the domain Ω is maintained, as the exactness of the locations of the cell boundaries do not play a role for the integral over Ω. Only the accuracy of the integration is of importance, which is given because integration of a polynomial can be done with high accuracy.

In Section 3.2 we observe that the trajectories take different forms depending on the velocity field. The same cases that are derived above are applied in this section as well. We distinguish between the cases of trajectories that remain within a grid cell that cross at least one cell boundary without small coefficients and with small coefficients and according approximations for positive and negative velocity.

We again observe Figure 3.1(b). The trajectories colored in red remain within a grid cell and correspond to the red part of the analytical solution in Figure 3.2(b). Respectively, the blue colored trajectories correspond to the blue part of the solution. To compute the integral over a grid cell, the interval and thus the integral of that cell is divided into these parts. We examine trajectories with arrival points at t^{n+1} in grid cell i. If we assume that these trajectories have departure points in l different grid cells, thus the domain of dependence of cell i covers l grid cells, i.e. the integral of the ith cell is split up into l parts. Each of these integrals over subintervals contains one form of trajectory. The boundaries of the subintervals and hence the limits of the integrals are obtained by following the trajectories that start at cell boundaries that are located between $\varphi(x_{i-1/2, t^{n+1}, -\Delta t})$ and $\varphi(x_{i+1/2, t^{n+1}, -\Delta t})$. Hence, these points

are given by $\varphi(x_{i-1/2-r,t^n,\Delta t})$ for some r that depends on the number of grid cells crossed. The overall solution is given by assembling the solutions of all subintervals.

$$\begin{aligned}
&\mathbb{P}\rho_i(x,t^{n+1}) \\
&= \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \left(\int_{\varphi_0}^{\varphi_1} W_1 \,\mathrm{d}x + \int_{\varphi_1}^{\varphi_2} W_2 \,\mathrm{d}x + \cdots + \int_{\varphi_{l-1}}^{\varphi_l} W_l \,\mathrm{d}x \right) K_j^{(i)}(x) \qquad (3.140) \\
&= \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{\varphi_0}^{\varphi_1} W_1 \,\mathrm{d}x\, K_j^{(i)}(x) + \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{\varphi_1}^{\varphi_2} W_2 \,\mathrm{d}x\, K_j^{(i)}(x) \\
&\quad + \cdots + \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{\varphi_{l-1}}^{\varphi_l} W_l \,\mathrm{d}x\, K_j^{(i)}(x), \qquad (3.141)
\end{aligned}$$

where the integrand is abbreviated with $W_q = \rho(x,t^n)K_j^{(i)}(\varphi(x,t^n,\Delta t))$ for $q = 1,\ldots,l$. The limits of the integral defined as $\varphi_0 = \varphi(x_{i-1/2},t^{n+1},-\Delta t)$ and $\varphi_l = \varphi(x_{i+1/2},t^{n+1},-\Delta t)$ are the same limits as in the projection (3.138), the other limits in between are abbreviated accordingly.

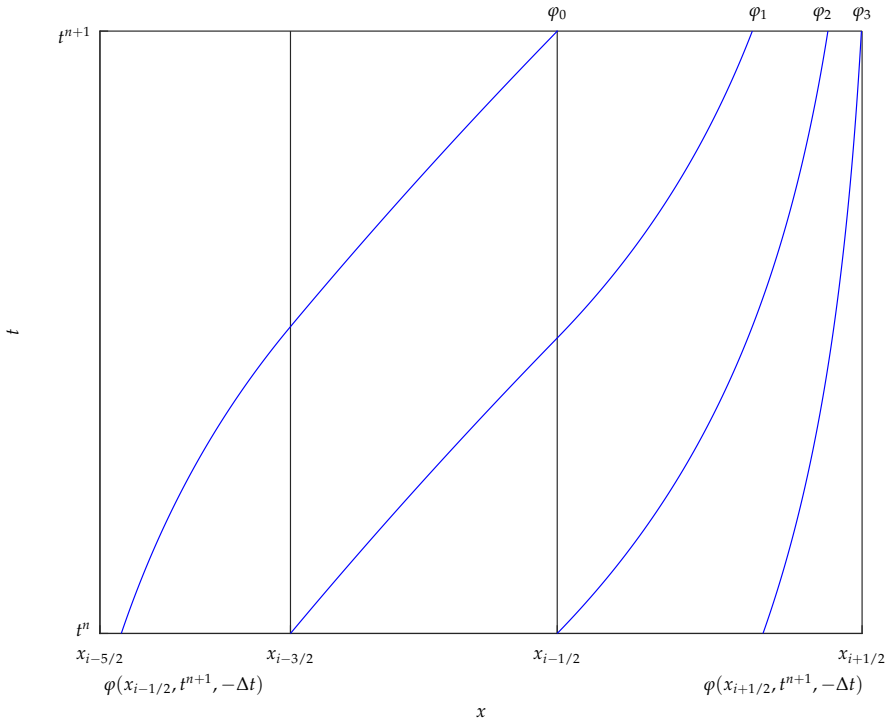


Figure 3.10: The interval $[x_{i-1/2},x_{i+1/2}]$ is divided into subintervals. The integral over each subinterval contains one type of trajectory. The trajectories shown in this figure form the boundaries for the subintervals.

Figure 3.10 shows how the interval of the ith grid cell is divided into subintervals. In this example the solution of grid cell i at t^{n+1} is obtained by the solution at time t^n of three different grid cells, i.e. cell $i, i-1$ and $i-2$. Therefore, the integral is split up into three integrals. The limits are given by $\varphi_q = \varphi(x_{i+1/2-q},t^n,\Delta t)$ for $q = 1,\ldots,3$, which are computed by following the trajectories that start at the cell boundaries $x_{i+1/2-q}$ at time t^n.

Each summand of (3.141) contributes a part of the solution for grid cell i. The summands are computed separately, because each of the trajectories in W_q is of a different type,

$$\sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{\varphi_0}^{\varphi_1} W_q \, \mathrm{d}x \, K_j^{(i)}(x) = \sum_{j=0}^{2} m_{j,i}^{n+1}\Big|_{\text{part}} K_j^{(i)}(x). \tag{3.142}$$

In the following section we consider integrals that contain trajectories for positive and negative velocity, which remain within one grid cell. The subsequent Section 3.4.2 examines the integral for the case when trajectories cross cell boundaries for positive and negative velocity as well. In these two sections the assumption is made that none of the coefficients, in particular the slope of the velocity, does not tend to zero. Those cases with small a_k are studied in Section 3.4.3 for positive velocity and negative velocity, respectively.

3.4.1 Trajectory remaining in one grid cell

We study the part of the solution that is obtained by following trajectories that remain within one grid cell, for example the red colored part shown in Figure 3.2(b).

The trajectories for positive velocity that end in the ith grid cell in the subinterval $[\varphi(x_{i-1/2}, t^n, \Delta t), x_{i+1/2}]$, or for negative velocity end in the subinterval $[x_{i-1/2}, \varphi(x_{i+1/2}, t^n, \Delta t)]$, respectively, and remain within that grid cell are given by (3.19). Inserted into (3.138), we obtain for cell i and the jth coefficient,

$$\begin{aligned} m_{j,i}^{n+1}\Big|_{\text{part}} &= \frac{2j+1}{\Delta x_i} \int \rho(x, t^n) K_j^{(i)}(\varphi(x, t^n, \Delta t)) \, \mathrm{d}x \\ &= \frac{2j+1}{\Delta x_i} \int \left(m_{2,i}^n K_2^{(i)}(x) + m_{1,i}^n K_1^{(i)}(x) + m_{0,i}^n K_0^{(i)}(x) \right) \\ &\qquad \cdot K_j^{(i)} \left(e^{a_k \Delta t} \left(\frac{b_k}{a_k} + x \right) - \frac{b_k}{a_k} \right) \mathrm{d}x \,. \end{aligned} \tag{3.143}$$

The integrand is a polynomial and can thus be easily integrated. This integral holds for positive and negative velocity.

The red part of the solution shown in Figure 3.2(b) is obtained by means of evaluating (3.143).

3.4.2 Trajectory crossing cell boundaries

We examine the part of the solution that is obtained by following trajectories that cross at least one cell boundary. An example is the blue colored part shown in Figure 3.2(b).

Trajectories that cross at least one cell boundary are described by (3.44) for positive velocity and by (3.91) for negative velocity. Both are of the form

$$\widetilde{\varphi}(\eta, t^n, \Delta t) = c_1 (c_2 \eta + 1)^{c_3} - c_4, \quad \text{for } \eta \in [0, 1]. \tag{3.144}$$

To obtain that form for positive velocity a substitution with the coordinate transformation (3.41), that is

$$x = \Delta x_0 - \Delta x_0 \eta, \tag{3.145}$$

is necessary. As described in Section 3.2, this form enables the series expansions for small coefficients.

The trajectory for negative velocity is rescaled by

$$x = \Delta x_0 \eta. \tag{3.146}$$

Because of these coordinate transformations, the trajectories for negative and positive velocity become of equal form and can be treated similarly.

To include the substitution applied in the case of positive velocity to the projection (3.138), we first apply the coordinate transformation to the density ρ and introduce suitable abbreviations. We then have

$$\begin{aligned}\widetilde{\rho}(\eta) &\equiv \rho(\Delta x_0 - \Delta x_0\eta, t^n)\\ &= m_2 K_2^{(0)}(\Delta x_0 - \Delta x_0\eta) + m_1 K_1^{(0)}(\Delta x_0 - \Delta x_0\eta)\\ &\quad + m_0 K_0^{(0)}(\Delta x_0 - \Delta x_0\eta)\\ &= \underbrace{6m_2}_{c_6}\eta^2 + \underbrace{(-2m_1 - 6m_2)}_{c_7}\eta + \underbrace{m_0 + m_1 + m_2}_{c_8}\\ &= c_6\eta^2 + c_7\eta + c_8.\end{aligned} \tag{3.147}$$

The application of the substitution to the integral expression of the projection (3.138) reads

$$\begin{aligned}&\mathbb{P}\rho(x, t^{n+1})\\ &= -\sum_{j=0}^{2}(2j+1)\frac{\Delta x_0}{\Delta x_i}\int_{-\frac{\varphi(x_{i-1/2},t^{n+1},-\Delta t)}{\Delta x_0}+1}^{-\frac{\varphi(x_{i+1/2},t^{n+1},-\Delta t)}{\Delta x_0}+1}\widetilde{\rho}(\eta)K_j^{(i)}(\widetilde{\varphi}(\eta, t^n, \Delta t))\,\mathrm{d}\eta\, K_j^{(i)}(x).\end{aligned} \tag{3.148}$$

For negative velocity, the density is transformed by the rescaled coordinate and given by

$$\begin{aligned}\widetilde{\rho}(\eta) &\equiv \rho(\Delta x_0\eta, t^n)\\ &= \underbrace{6m_2}_{c_6}\eta^2 + \underbrace{(2m_1 - 6m_2)}_{c_7}\eta + \underbrace{m_0 - m_1 + m_2}_{c_8}\\ &= c_6\eta^2 + c_7\eta + c_8.\end{aligned} \tag{3.149}$$

The limits of the integral change to $\varphi(x_{i+1/2,t^{n+1},-\Delta t})/\Delta x_0$ and $\varphi(x_{i-1/2,t^{n+1},-\Delta t})/\Delta x_0$, respectively. The structure of the integral is the same for positive and negative velocity, when we plug in the transformed density $\widetilde{\rho}$ and the trajectories $\widetilde{\varphi}$. We examine the structure of the integrals for each coefficient.

$$\begin{aligned}m_{0,i}^{n+1}\Big|_{\text{part}} &= -\frac{\Delta x_0}{\Delta x_i}\int \widetilde{\rho}(\eta)K_0^{(i)}(\widetilde{\varphi}(\eta))\,\mathrm{d}\eta\\ &= -\frac{\Delta x_0}{\Delta x_i}\int c_6\eta^2 + c_7\eta + c_8\,\mathrm{d}\eta\end{aligned} \tag{3.150}$$

$$\begin{aligned}m_{1,i}^{n+1}\Big|_{\text{part}} &= -\frac{3\Delta x_0}{\Delta x_i}\int \widetilde{\rho}(\eta)K_1^{(i)}(\widetilde{\varphi}(\eta))\,\mathrm{d}\eta\\ &= -\frac{3\Delta x_0}{\Delta x_i}\int \frac{2}{\Delta x}c_1\left(c_6\eta^2 + c_7\eta + c_8\right)(c_2\eta + 1)^{c_3}\\ &\quad + \left(-\frac{2c_4}{\Delta x} - 1\right)\left(c_6\eta^2 + c_7\eta + c_8\right)\mathrm{d}\eta\end{aligned} \tag{3.151}$$

$$\begin{aligned}m_{2,i}^{n+1}\Big|_{\text{part}} &= -\frac{5\Delta x_0}{\Delta x_i}\int \widetilde{\rho}(\eta)K_2^{(i)}(\widetilde{\varphi}(\eta))\,\mathrm{d}\eta\\ &= -\frac{5\Delta x_0}{\Delta x_i}\int \frac{6}{\Delta x^2}c_1^2(c_6\eta^2 + c_7\eta + c_8)(c_2\eta + 1)^{2c_3}\\ &\quad - \left(\frac{12}{\Delta x^2}c_1c_4 + \frac{6}{\Delta x}c_1\right)(c_6\eta^2 + c_7\eta + c_8)(c_2\eta + 1)^{c_3}\\ &\quad + \left(\frac{6}{\Delta x^2}c_4^2 + \frac{6}{\Delta x}c_4 + 1\right)\left(c_6\eta^2 + c_7\eta + c_8\right)\mathrm{d}\eta\end{aligned} \tag{3.152}$$

As mentioned before, the zeroth coefficient m_0 is obtained by integration of a polynomial. For the first and second coefficient, m_1 and m_2, the essential problem to solve integrals is of the type

$$\int_a^b \eta^k \, (1 + c_2\eta)^p \, \mathrm{d}\eta \qquad (k \in \{0, 1, 2\}), \tag{3.153}$$

for $p = jc_3$ with $j \in \{1, 2\}$. Recalling that $c_2 = -\Delta x_0 a_0 / u_1$, we observe that $c_2 \in \mathcal{O}(\Delta x_0)$. Therefore, for small grid cells, the parameter takes small values which can be enhanced near extremal points of the velocity where the slope a_0 is small. However, c_2 can take large values as well. The size of c_2 has influence on the algorithm for solving the integral as we will study in the following.

The solution can be assembled with help of the subsequent expressions I_1, I_2 and I_3. We have for $k = 0$,

$$\begin{aligned} I_0 &= \int_a^b (1 + c_2\eta)^p \, \mathrm{d}\eta \\ &= \frac{1}{c_2} \frac{(1 + c_2\eta)^{p+1}}{p+1} \Bigg|_a^b \end{aligned} \tag{3.154}$$

for the first expression. The multiplication of the integrand with η, i.e. $k = 1$ yields

$$I_1 = \int_a^b \eta (1 + c_2\eta)^p \, \mathrm{d}\eta \tag{3.155}$$

$$= -\frac{(1 + c_2\eta)^{p+2}}{c_2^2(p+1)(p+2)} + \frac{\eta(1 + c_2\eta)^{p+1}}{c_2(p+1)} \Bigg|_a^b \tag{3.156}$$

$$= \frac{1}{c_2^2} \frac{(1 + c_2\eta)^{p+2}}{p+2} \Bigg|_a^b - \frac{I_0}{c_2} \tag{3.157}$$

and the last type of integral for $k = 2$ that needs to be solved is given by

$$I_2 = \int_a^b \eta^2 (1 + c_2\eta)^p \, \mathrm{d}\eta \tag{3.158}$$

$$\begin{aligned} &= \frac{2(1 + c_2\eta)^{p+3}}{c_2^3(p+1)(p+2)(p+3)} - \frac{2\eta(1 + c_2\eta)^{p+2}}{c_2^2(p+1)(p+2)} \\ &\quad + \frac{\eta^2(1 + c_2\eta)^{p+1}}{c_2(p+1)} \Bigg|_a^b \end{aligned} \tag{3.159}$$

$$= \frac{1}{c_2^3} \frac{(1 + c_2\eta)^{p+3}}{p+3} \Bigg|_a^b - 2\frac{I_1}{c_2} + \frac{I_0}{c_2^2} \tag{3.160}$$

(3.156) and (3.159) are obtained by direct integration by parts. The subsequent formulations (3.157) and (3.160), respectively, are equivalent expressions and enable a more direct evaluation of the integral. The first term of the reformulated expressions can be computed by

$$\int_a^b (1 + cx)^{p-1} \, \mathrm{d}x = \frac{(cx + 1)^p}{cp} \Bigg|_a^b, \tag{3.161}$$

the following terms use the values computed beforehand. To avoid division by zero by the coefficient p, we obtain an approximation of the integral by series expansion

$$\begin{aligned} \int_a^b (1+cx)^{p-1} \, \mathrm{d}x &= \frac{\ln(cx+1)}{c} + \frac{p \ln^2(cx+1)}{2c} + \frac{p^2 \ln^3(cx+1)}{6c} \\ &\quad + \frac{p^3 \ln^4(cx+1)}{24c} \Bigg|_a^b + \mathcal{O}\left(p^5\right), \end{aligned} \tag{3.162}$$

for small p.

We have for each coefficient

$$m_{0,i}^{n+1}\Big|_{\text{part}} = \frac{c_6}{3}\eta^3 + \frac{c_7}{2}\eta^2 + c_8\eta \tag{3.163}$$

for I_k for $k = 0, 1, 2$, $p = c_3$ is used

$$\begin{aligned} m_{1,i}^{n+1}\Big|_{\text{part}} =& \frac{2}{\Delta x_i} c_1 \left(c_6 I_2 + c_7 I_1 + c_8 I_0\right) \\ &+ \left(-\frac{2c_4}{\Delta x_i} - 1\right)\left(\frac{c_6}{3}\eta^3 + \frac{c_7}{2}\eta^2 + \frac{c_8}{3}\eta\right) \\ =& \frac{2}{\Delta x_i} c_1 \left(c_6 I_2 + c_7 I_1 + c_8 I_0\right) + \left(-\frac{2c_4}{\Delta x_i} - 1\right) m_{0,i}^{n+1} \end{aligned} \tag{3.164}$$

for I'_k for $k = 0, 1, 2$, $p = 2c_3$ is used

$$\begin{aligned} m_{2,i}^{n+1}\Big|_{\text{part}} =& -\left(\frac{6}{\Delta x_i^2} c_1^2 \left(c_6 I'_2 + c_7 I'_1 + c_8 I'_0\right)\right. \\ &+ -\left(\frac{12}{\Delta x_i^2} c_1 c_4 + \frac{6}{\Delta x_i} c_1\right)\left(c_6 I_2 + c_7 I_1 + c_8 I_0\right)\Big)\Delta x_0 \\ &+ \left(\frac{6}{\Delta x_i^2} c_4^2 + \frac{6}{\Delta x_i} c_4 + 1\right) m_{0,i}^{n+1} \end{aligned} \tag{3.165}$$

We consider the role of c_2. As long as the value is not small, because either the grid cell Δx_0 is not small or the slope a_0 of the velocity in that cell is large, the evaluation of the terms I_0, I_1 and I_2 can be done as described above.

However, if c_2 is small, we notice significant numerical errors evaluating above expressions. We examine the term I_2. The key issue transpires when we compare (3.158) with (3.160): According to (3.158), $I_2 = \mathcal{O}(1)$ as $c_2 \to 0$. The equivalent formula in (3.160) instead involves terms of order $\mathcal{O}\left(c_2^{-3}\right)$, so that cancellation of such large terms to third order in c_2 is needed to get the net result down to order $\mathcal{O}(1)$. Cancellation of significant digits upon subtraction of almost identical terms seems pre-programmed this way.

Integrating (3.153) by parts once - the other way around as done above - we obtain

$$\int_a^b \eta^k (1 + c_2\eta)^p \,\mathrm{d}\eta = \frac{\eta^{k+1}}{k+1}(1 + c_2\eta)^p\Big|_a^b - c_2\frac{p}{k+1}\int_a^b \eta^{k+1}(1 + c_2\eta)^{p-1}\,\mathrm{d}\eta \ . \tag{3.166}$$

This expression is clearly of order $\mathcal{O}(1)$ and subtractions of almost equal terms are not involved any longer.

The last integral in (3.166) is of the same type as that in (3.153), albeit with a different range for k, and repeated integration by parts yields an N-term expression that reveals the c_2-dependence of the entire formula

$$\begin{aligned} &\int_a^b \eta^k (1 + c_2\eta)^p \,\mathrm{d}\eta \\ &\quad = \sum_{n=0}^{N-1} c_2^n (-1)^n \frac{1}{p-n}\left(\prod_{\nu=0}^{n}\frac{p-\nu}{k+1+\nu}\right)\eta^{k+1+n}(1 + c_2\eta)^{p-n}\Big|_a^b \\ &\quad + (-1)^N c_2^N \left(\prod_{\nu=0}^{N-1}\frac{p-\nu}{k+1+\nu}\right)\int_a^b \eta^{k+N}(1 + c_2\eta)^{p-N}\,\mathrm{d}\eta \ . \end{aligned} \tag{3.167}$$

One potential remedy to our problem may be evaluation of (3.167) for some sufficiently large N and neglect of the last term. This might appear like an asymptotic

expansion. Nevertheless, all terms still retain their c_2-dependence through the powers of $(1+c_2\eta)$. The expression $(1+c_2\eta)^p \in \mathcal{O}(1)$, and a as well as b are both in $\mathcal{O}(1)$, too. Thus I_0, I_1 and I_2 are in $\mathcal{O}(1)$. If these computations are done without cancellation of important digits, the remainder of the computations of the projection step is computed without difficulties. The values $c_6 \in \mathcal{O}\left(\Delta x^2\right), c_7 \in \mathcal{O}(\Delta x)$ and $c_8 \in \mathcal{O}(1)$. Thus, it holds $c_8 I_0 \in \mathcal{O}(1)$, $c_7 I_1 \in \mathcal{O}(\Delta x)$ and $c_6 I_2 \in \mathcal{O}\left(\Delta x^2\right)$ and these expressions are hence in different scales. Note, that for the implementation of (3.167), we avoid the division by $p-n$ as it can be equal to zero. This can be easily done as the last factor of

$$\prod_{\nu=0}^{n} \frac{p-\nu}{k+1+\nu} \tag{3.168}$$

is equal to $p-n$. Thus, we have

$$\frac{1}{p-n}\left(\prod_{\nu=0}^{n} \frac{p-\nu}{k+1+\nu}\right) = \frac{1}{k+1}\left(\prod_{\nu=0}^{n-1} \frac{p-\nu}{k+2+\nu}\right). \tag{3.169}$$

We have derived the projection for all cases where the velocity u or its slope does not tend to zero. The case of the trajectories to remain within one grid cell as well as the case of the trajectories crossing grid cell boundaries is discussed with numerical issues that can arise.

Analogously to the derivation of the trajectories in 3.2, we have to study the cases for the projection where coefficients of the trajectory tend to or are equal to zero. That is done in the following.

3.4.3 Integration with small coefficients

In Section 3.2 we have seen that the trajectory $\widetilde{\varphi}$ derived in (3.43) for positive velocity and in (3.90) for negative velocity is not defined for the coefficients $a_0 = 0$ and $a_n = 0$. Therefore, alternative trajectories are derived that approximate the original trajectory (see Section 3.2.4 for an overview of the different types of trajectories). The question at which point to switch between the trajectory and its approximations is open. If $|a_0|$ and $|a_n|$ are smaller than some value ϵ, the alternative algorithm is chosen. In numerical tests, the value $\epsilon = 0.5$ seems to be a suitable choice. It is used for all numerical computations of the SASLDG method in this thesis.

We study the coefficients $m_{0,i}^{n+1}$, $m_{1,i}^{n+1}$ and $m_{2,i}^{n+1}$ individually of the projection (3.148). The first coefficient is given by

$$\begin{aligned} m_{0,i}^{n+1}\Big|_{\text{part}} &= \int \widetilde{\rho}(x)\,\mathrm{d}x \\ &= \int c_6 x^2 + c_7 x + c_8\,\mathrm{d}x\,. \end{aligned} \tag{3.170}$$

The coefficients $m_{1,i}^{n+1}$ and $m_{2,i}^{n+1}$ yield

$$\begin{aligned} m_{1,i}^{n+1}\Big|_{\text{part}} &= \int \widetilde{\rho}(x) K_1(\widetilde{\varphi}(x,t^n,\Delta t))\,\mathrm{d}x \\ &= \int \left(c_6 x^2 + c_7 x + c_8\right)\left(\frac{2}{\Delta x_i}\widetilde{\varphi}(x,t^n,\Delta t) - 1\right)\mathrm{d}x\,, \end{aligned} \tag{3.171}$$

and

$$\begin{aligned} m_{2,i}^{n+1}\Big|_{\text{part}} &= \int \widetilde{\rho}(x) K_2(\widetilde{\varphi}(x,t^n,\Delta t))\,\mathrm{d}x \\ &= \int \left(c_6 x^2 + c_7 x + c_8\right)\left(\frac{6}{\Delta x_i^2}\widetilde{\varphi}^2(x,t^n,\Delta t)\right. \\ &\qquad \left. - \frac{6}{\Delta x_i}\widetilde{\varphi}(x,t^n,\Delta t) + 1\right)\mathrm{d}x\,. \end{aligned} \tag{3.172}$$

The first coefficient $m_{0,i}^{n+1}$ can be determined easily, because the necessary computation is integration over a quadratic polynomial. The coefficients $m_{1,i}^{n+1}$ and $m_{2,i}^{n+1}$ need more consideration. We note that $\widetilde{\varphi}$ is in the argument of K_2 and therefore $\widetilde{\varphi}^2$ is needed for the computations and needs to be determined as a first step for all cases of $|a_0| < \epsilon$, $|a_n| < \epsilon$ and the case $|a_0|$ and $|a_n|$ being small.

As mentioned above, all cases lead to different forms of trajectories. Therefore, the integrals are computed in different ways as well. The limits of integration are discussed in Section 3.6. In this part we use general placeholders x_L and x_R, which are in $[0, 1]$. Note that $\widetilde{\rho}$ is defined by (3.147) for positive velocity and by (3.149) for negative velocity.

In the following we will recall the type of the respective trajectory for each case. Then, we will derive an approximative form of the square of the trajectory. We discuss how the integral is solved. In the process we consider numerical difficulties that arise and show the resolutions how to deal with them.

Positive velocity: $|a_0| < \epsilon$

We derived the trajectory for positive velocity and small coefficient a_0 in Section 3.2.1. We found an abbreviated form that is given by (3.61),

$$\begin{aligned}\widetilde{\varphi}(x, t^n, \Delta t) = d_1 \exp(d_2 x)\Big(1 + d_3 x^2 + d_4 x^3 + d_5 x^4 + d_6 x^5 \\ + d_7 x^6\Big) + d_8.\end{aligned} \tag{3.173}$$

To obtain the square of the trajectory either the square of the expanded form of $\widetilde{\varphi}$ in (3.173) is determined or the original trajectory given in (3.43) is first squared and then approximated by a series expansion about $a_0 = 0$. The first choice leads to a multiplication by a polynomial factor with degree 12, the latter option yields a polynomial of degree 6, which is the same degree as in the factor in (3.173). Hence, the second choice is preferred. The derived expression of $\widetilde{\varphi}^2$ is found in the appendix in (A.20). We show the abbreviated form, where the abbreviations of (3.61) are reused. We obtain

$$\begin{aligned}\widetilde{\varphi}^2(x, t^n, \Delta t) = g_1 \exp(g_2 x)\left(1 + g_3 x^2 + g_4 x^3 + g_5 x^4 + g_6 x^5 + g_7 x^6\right) \\ + 2 d_8 d_1 \exp(d_2 x)\left(1 + d_3 x^2 + d_4 x^3 + d_5 x^4 + d_6 x^5 + d_7 x^6\right) + g_8\end{aligned} \tag{3.174}$$

for suitable coefficients g_j for $j = 1, \ldots, 8$.

For the computation of (3.171) and (3.172) we insert the abbreviated forms of $\widetilde{\varphi}$ and $\widetilde{\varphi}^2$. For the coefficient $m_{1,i}^{n+1}$ we obtain

$$\begin{aligned}m_{1,i}^{n+1}\Big|_{\text{part}} = \int \frac{2}{\Delta x_i}\left(c_6 x^2 + c_7 x + c_8\right)\Big(d_1 \exp(d_2 x)\left(1 + d_3 x^2\right. \\ \left. + d_4 x^3 + d_5 x^4 + d_6 x^5 + d_7 x^6\right)\Big) + \\ + \left(c_6 x^2 + c_7 x + c_8\right)\left(\frac{2}{\Delta x_i} d_8 - 1\right) \mathrm{d}x\,.\end{aligned} \tag{3.175}$$

The coefficient $m_{2,i}^{n+1}$ yields

$$\begin{aligned} m_{2,i}^{n+1}\Big|_{\text{part}} = \int \frac{6g_1}{\Delta x_i^2}(c_6x^2+c_7x+c_8)\exp(g_2x)\Big(1+g_3x^2+g_4x^3 \\ +g_5x^4+g_6x^5+g_7x^6\Big)+ \\ -\left(\frac{12}{\Delta x_i^2}\frac{u_n}{a_n}+\frac{6}{\Delta x_i}\right)(c_6x^2+c_7x+c_8)d_1\exp(d_2x) \\ \cdot\left(1+d_3x^2+d_4x^3+d_5x^4+d_6x^5+d_7x^6\right) \\ +\left(\frac{6}{\Delta x_i^2}+1\right)\left(c_6x^2+c_7x+c_8\right)\mathrm{d}x\,. \end{aligned} \tag{3.176}$$

In order to compute the integral of (3.175) and (3.176), the task is to compute integrals of the following form

$$\int_{x_L}^{x_R} x^k\exp(d_2x)\,\mathrm{d}x \tag{3.177}$$

for $k=0,1,\ldots,6$. When we solve the integral analytically, the solution is obtained by integration by parts - integrating the expression $\exp(d_2x)$ and differentiating the monomial factor x^k. That leads to the solution

$$\int_{x_L}^{x_R} x^k\exp(d_2x)\,\mathrm{d}x = \sum_{j=1}^{k+1}(-1)^{j+1}\left(\prod_{\nu=k-j+2}^{k}\nu\right)\frac{1}{d_2^j}x^{k-j+1}\exp(d_2x)\Big|_{x_L}^{x_R}. \tag{3.178}$$

The coefficient d_2 plays a role for the numerical evaluation of the integral. It can be of arbitrary size, small or large, depending on the given velocity,

$$d_2 = -\frac{a_n\Delta x_0}{u_1}. \tag{3.179}$$

The coefficient a_n is assumed to be not small in this case, otherwise both coefficients a_0 and a_n would be small ($|a_0|<\epsilon$ throughout this section). The problem of both coefficients being small is handled in the following section. However, the size of Δx_0 and u_1 are arbitrary, which determines the size of d_2.

If we want to find the behavior and the order of the solution of the integral for small d_2, we apply a Taylor expansion about the point $d_2=0$ to the exponential function. We have

$$\int_{x_L}^{x_R} x^k\exp(d_2x)\,\mathrm{d}x = \int_{x_L}^{x_R}\sum_{j=0}^{\infty}\frac{d_2^jx^{k+j}}{j!}\,\mathrm{d}x \tag{3.180}$$

$$= \mathcal{O}(1)\,, \tag{3.181}$$

for small d_2.

One problem becomes obvious when considering (3.178). The division by d_2^{k+1}, if d_2 is small, causes cancellation in the whole expression. We examine the solution of the integral for the summand $(k+1)$ thoroughly without the prefactor. It is given by

$$\begin{aligned} \frac{1}{d_2^{k+1}}\exp(d_2x)\Big|_{x_L}^{x_R} &= \frac{1}{d_2^{k+1}}\exp(d_2x_R)-\frac{1}{d_2^{k+1}}\exp(d_2x_L) \\ &= \frac{1}{d_2^{k+1}}\left(\exp(d_2x_R)-\exp(d_2x_L)\right). \end{aligned} \tag{3.182}$$

For small d_2 both numbers - the minuend and subtrahend - are both very large numbers and as the whole integral is of order one, they must be almost equally sized. Hence, loss of significance occurs. More precisely, it can be seen at which size of d_2 the problem of cancellation arises and of which order the error is.

To see that, we apply a Taylor expansion to the exponential functions about $d_2 = 0$ in (3.182) and obtain

$$\begin{aligned}\exp(d_2 x_R) - \exp(d_2 x_L) = &\left(1 + d_2 x_R + \frac{1}{2}(d_2 x_R)^2 + \dots \right.\\ &\left. - \left(1 + d_2 x_L + \frac{1}{2}(d_2 x_L)^2 + \dots\right)\right)\\ &= d_2\,(x_R - x_L) + d_2^2\left(\frac{1}{2}x_R^2 - \frac{1}{2}x_L^2\right) + \dots\end{aligned} \tag{3.183}$$

To estimate the numerical error, that is made in (3.182), we note on the number representation in MATLAB. The IEEE-754 norm is used and the double-precision floating points are stored in 64 bits. 52 bits for the mantissa, 11 for the exponent and one bit for the sign. This yields $N = 16$ significant digits for the representation of an arbitrary number in the decimal numeral system.

We write the factor d_2 as

$$d_2 = 0,\underbrace{0000}_{a-1}\underbrace{x_1 x_2 x_3 \dots}_{N-(a-1)} \tag{3.184}$$

$$= x_1, x_2 x_3 \dots \cdot 10^{-5} \tag{3.185}$$

$$= D \cdot 10^{-a} \tag{3.186}$$

with

$$D = x_1, x_2 x_3 \dots \tag{3.187}$$

and with x_k for $k = 1, 2, \dots$ are arbitrary digits. The number of zeros is $a - 1$.

Studying the first summand in (3.183), we note that the first $N - (a-1)$ digits of the difference $d_2(x_R - x_L)$ are correctly represented, but an error is introduced at the digit $N - (a-1) + 1$, because of the factor d_2. This error can be estimated by

$$\text{err} \sim 10^{-(N-(a-1)+1)}. \tag{3.188}$$

The other summands in (3.183) are of smaller order and therefore contribute less to the overall error. At (3.182) division by d_2^{k+1} enlarges the error even more to

$$\text{err} \sim \frac{10^{-(N-(a-1)+1)}}{d_2^{k+1}} = \frac{10^{-(N-(a-1)+1)}}{D^{k+1}10^{-a(k+1)}}. \tag{3.189}$$

To specify a reasonable bound for the error, we set

$$\text{err} = \text{TOL} \tag{3.190}$$

for some error tolerance given. We have

$$\text{TOL} = \frac{10^{-(N-(a_{\text{TOL}}-1)+1)}}{d_2^{k+1}} = \frac{10^{-(N-(a_{\text{TOL}}-1)+1)}}{D^{k+1}10^{-a_{\text{TOL}}(k+1)}}. \tag{3.191}$$

and solve for a, and obtain

$$a_{\text{TOL}} = \frac{\frac{\ln\left(\text{TOL}\cdot D^{k+1}\right)}{\ln(10)} + 2 + N}{k+2}. \tag{3.192}$$

For all $a > a_{\text{TOL}}$, the error tolerance is exceeded. Thus, the bound for d_2 is given by

$$\delta \sim 10^{-a_{\text{TOL}}}. \tag{3.193}$$

The numerical error introduced in (3.178) because of cancellation is bounded by TOL, if $d_2 > \delta$. Depending on how sharp and exact the bound should be computed, D can either be simply set to one or it can be computed by $D = d_2 10^a$, where $a = -\lfloor \log_{10}(d_2) \rfloor$ is the order of d_2. We can plug in $N = 16$ and the maximum number for k, which is $k = 6$. If we set $D = 1$ and $\mathrm{TOL} = 10^{-10}$, then the order of the bound is

$$a_{\mathrm{TOL}} = 1. \tag{3.194}$$

This yields a bound for d_2 of

$$\delta = 0.1. \tag{3.195}$$

If $d_2 < \delta$ it is not advisable to use the analytical formula given in (3.178) to evaluate the integral as the error exceeds a given tolerance. A different algorithm must be used.

The solution to the problem for small $d_2 < \delta$ is again integration by parts. However, this time we integrate x^k and differentiate $\exp(d_2 x)$. It is not determined how many times the integration by parts takes place. It is applied as many times until the cutoff error drops below Δx^2. We assume the iterative process is applied N times,

$$\begin{aligned}\int_{x_L}^{x_R} x^k \exp(d_2 x)\,\mathrm{d}x = &\sum_{j=1}^{N} (-1)^{j+1} \left(\prod_{\nu=1}^{j} \frac{1}{k+\nu} \right) d_2^{j-1} x^{k+j} \exp(d_2 x) \Big|_{x_L}^{x_R} \\ &+ (-1)^N \left(\prod_{\nu=1}^{N} \frac{1}{k+\nu} \right) d_2^N \int_{x_L}^{x_R} x^{k+N} \exp(d_2 x)\,\mathrm{d}x\,.\end{aligned} \tag{3.196}$$

The cutoff error of the last term, i.e. the integral, can be estimated by expanding the exponential function.

$$\begin{aligned}&\int_{x_L}^{x_R} \left(\prod_{\nu=1}^{N} \frac{1}{k+\nu} \right) d_2^N x^{k+N} \exp(d_2 x)\,\mathrm{d}x \\ &\quad = \int_{x_L}^{x_R} \left(\prod_{\nu=1}^{N} \frac{1}{k+\nu} \right) d_2^N x^{k+N} \left(\sum_{i=0}^{\infty} \frac{(d_2 x)^i}{i!} \right) \mathrm{d}x \\ &\quad = \int_{x_L}^{x_R} \left(\prod_{\nu=1}^{N} \frac{1}{k+\nu} \right) d_2^N \left(\sum_{i=0}^{\infty} \frac{d_2^i x^{k+N+i}}{i!} \right) \mathrm{d}x \\ &\quad = \left(\prod_{\nu=1}^{N} \frac{1}{k+\nu} \right) d_2^N \left(\sum_{i=0}^{\infty} \frac{d_2^i x^{k+N+i+1}}{i!(k+N+i+1)} \right) \Bigg|_{x_L}^{x_R} \\ &\quad \leq |d_2|^N \left(\sum_{i=0}^{\infty} |d_2|^i x^{k+N+i+1} \right) \Bigg|_{x_L}^{x_R} \\ &\quad = \mathcal{O}\left(d_2^N \right)\end{aligned} \tag{3.197}$$

Thus, N is chosen such that $|d_2^N| < \Delta x^2$ to maintain the order of accuracy of the numerical method.

Summarized, depending on the size of $|d_2|$, either the integrals are computed exactly or they are solved approximately while controlling the size of the approximation error and keeping it small enough. In both ways, the computations are not affected by cancellation of significance digits.

Positive velocity: $|a_n| < \epsilon$

The trajectory that crosses at least one grid cell boundary proceeds forward in time and is suitable for a small coefficient a_n is given in (3.69). We recall

$$\begin{aligned}\widetilde{\varphi}(x,t^n,\Delta t) &= d_1 + d_2 \ln(d_6 x+1) + d_3 \ln^2(d_6 x+1) \\ &\quad + d_4 \ln^3(d_6 x+1) + d_5 \ln^4(d_6 x+1) \qquad (3.198)\\ &= \sum_{i=1}^{5} d_i \ln^{i-1}(d_6 x+1)\,. \qquad (3.199)\end{aligned}$$

We have to determine the coefficients $m_{0,i}$, $m_{1,i}$ and $m_{2,i}$ at time t^{n+1}. For $m_{2,i}^{n+1}$, we need to determine $\widetilde{\varphi}^2$ for small a_n. We start with the form of $\widetilde{\varphi}$ given in (3.43), compute the square and carry out a Taylor expansion about $a_n = 0$. The complete formula is given in the appendix in (A.21). Using suitable abbreviation, we have

$$\begin{aligned}\widetilde{\varphi}^2(x,t^n,\Delta t) &= e_1 + e_2 \ln(d_6 x+1) + e_3 \ln(d_6 x+1)^2 \\ &\quad + e_4 \ln(d_6 x+1)^3 + e_5 \ln(d_6 x+1)^4 \qquad (3.200)\\ &\quad + e_6 \ln(d_6 x+1)^5 \\ &= \sum_{i=1}^{6} e_i \ln^{i-1}(d_6 x+1)\,. \qquad (3.201)\end{aligned}$$

We insert the approximations of $\widetilde{\varphi}$ and $\widetilde{\varphi}^2$ into the equations to compute (3.171) and (3.172), and obtain

$$\begin{aligned}m_{1,i}^{n+1}\Big|_{\text{part}} &= \int \left(c_6 x^2 + c_7 x + c_8\right) \\ &\quad \cdot \left(\frac{2}{\Delta x_i}\left(\sum_{i=1}^{5} d_i \ln^{i-1}(d_6 x+1)\right) - 1\right) \mathrm{d}x\,,\end{aligned} \qquad (3.202)$$

and

$$\begin{aligned}m_{2,i}^{n+1}\Big|_{\text{part}} &= \int \left(c_6 x^2 + c_7 x + c_8\right)\left(\frac{6}{\Delta x_i^2}\left(\sum_{i=1}^{6} e_i \ln^{i-1}(d_6 x+1)\right)\right. \\ &\quad \left. - \frac{6}{\Delta x_i}\left(\sum_{i=1}^{5} d_i \ln^{i-1}(d_6 x+1)\right) + 1\right) \mathrm{d}x\,.\end{aligned} \qquad (3.203)$$

This ends up to solve these types of integrals

$$\int_{x_L}^{x_R} x^k \ln^l(d_6 x+1)\,\mathrm{d}x\,, \text{ where } k \in \{0,1,2\},\ l \in \{1,2,\ldots,5\}. \qquad (3.204)$$

The case $l = 0$ is trivial and needs no further discussion. Before we compute the integrals, the coefficient d_6 needs some attention as it determines the way the integrals are solved. Generally, d_6 can take any value

$$d_6 = -\frac{a_0}{u_1}\Delta x_0. \qquad (3.205)$$

We can assume a_0 is not small. Because if otherwise, then both $|a_0|$ and $|a_n|$ would be small ($|a_n| < \epsilon$ is assumed for this section). The case of both coefficients being small is handled in the next section. However, even with a_0 assumed not to be small, Δx_0 can be small, which would result in a small $|d_6|$. The velocity u_1 at the grid cell boundary also influences the size of d_6.

If $|d_6| > \delta$ for some suitable δ, the integrals can be solved in the straight forward and stable way by integration by parts.

However, if $|d_6|$ is small, this algorithm leads to numerical instabilities caused by cancellation. We pick out one integral to discuss the possible cancellation exemplarily,

$$\int x^2 \ln(d_6 x+1)^2 \,\mathrm{d}x = \tag{3.206}$$

$$\left(\frac{18d_6^3x^3+18}{54d_6^3}\right)\ln(d_6x+1)^2 \tag{3.207}$$

$$-\frac{12d_6^3x^3-18d_6^2x^2+36d_6x+66}{54d_6^3}\ln(d_6x+1) \tag{3.208}$$

$$+\frac{4d_6^3x^3-15d_6^2x^2+66d_6x}{54d_6^3}. \tag{3.209}$$

A Taylor expansion of the integral in (3.206) about $d_6 = 0$ reveals the order of the result. We have

$$\begin{aligned}\int x^2 \ln(d_6x+1)^2 d_6x &= \frac{x^5d_6^2}{5}-\frac{x^6d_6^3}{6}+\frac{11x^7d_6^4}{84}-\frac{5x^8d_6^5}{48}+\mathcal{O}\left(d_6^6\right)\\ &= \mathcal{O}\left(d_6^2\right)\end{aligned} \tag{3.210}$$

Further expansions show the order of individual expressions. If we consider the last term in (3.208), we find

$$\begin{aligned}-\frac{66}{54\,d_6^3}\ln(d_6x+1) &= -\frac{11x}{9d_6^2}+\frac{11x^2}{18d_6}-\frac{11x^3}{27}+\frac{11x^4d_6}{36}\\ &\quad -\frac{11x^5d_6^2}{45}+\frac{11x^6d_6^3}{54}+\mathcal{O}\left(d_6^4\right)\\ &= \mathcal{O}\left(-\frac{1}{d_6^2}\right).\end{aligned} \tag{3.211}$$

It turns out to be the largest term for small $|d_6|$ in the integral and of order $\mathcal{O}\left(1/d_6^2\right)$. To get down to the final order of the whole integral (3.206), there is another large term of the same order, which is the last term in (3.209),

$$\frac{66\,d_6x}{54\,d_6^3} = \frac{11x}{9d_6^2} = \mathcal{O}\left(\frac{1}{d_6^2}\right). \tag{3.212}$$

For small d_6 the first cancellation occurs exactly at the subtraction of these two large terms. After these terms are cancelled, the second largest terms are of order $\mathcal{O}\left(1/d_6\right)$. The first one is revealed by a Taylor expansion of (3.207),

$$\begin{aligned}\frac{18\,d_6^3\,x^3+18}{54\,d_6^3}\ln\left(d_6x+1\right)^2 &= \frac{x^2}{3d_6}-\frac{x^3}{3}+\frac{11x^4d_6}{36}+\frac{x^5d_6^2}{18}\\ &\quad -\frac{43x^6d_6^3}{540}+\frac{13x^7d_6^4}{180}+\mathcal{O}\left(d_6^5\right)\end{aligned} \tag{3.213}$$

$$= \mathcal{O}\left(\frac{1}{d_6}\right). \tag{3.214}$$

After a Taylor expansion of the sum of (3.208) and (3.209), the corresponding term of the same order appears

$$
\begin{aligned}
&-\frac{12d_6^3x^3-18\,d_6^2x^2+36\,d_6x+66}{54\,d_6^3}\ln\left(d_6x+1\right)\\
&\quad+\frac{4\,d_6^3x^3-15\,d_6^2x^2+66\,d_6x}{54\,d_6^3} \qquad (3.215)\\
&=-\frac{x^2}{3d_6}+\frac{x^3}{3}-\frac{11x^4d_6}{36}+\frac{13x^5d_6^2}{90}-\frac{47x^6d_6^3}{540}+\mathcal{O}\left(d_6^4\right)\\
&=\mathcal{O}\left(\frac{1}{d_6}\right). \qquad (3.216)
\end{aligned}
$$

The results of (3.213) and (3.215) are equal up to order $\mathcal{O}\left(d_6^2\right)$ with opposite signs. The sum of these terms yields the complete indefinite integral. Thus, for each summand, cancellation takes place.

A solution to avoid these instabilities due to loss of significant digits is to use d_6 as a bound. If $|d_6| \leq \delta$, we apply the reversed integration by parts to compute the integrals. The choice of $\delta = 0.005$ turned out to be a practical bound.

We show the procedure for $l = 1$ and for $l = 2$, all other cases follow analogously. We have for $l = 1$,

$$
\begin{aligned}
\int x^k\ln\left(d_6x+1\right)\mathrm{d}x=&\frac{1}{k+1}x^{k+1}\ln(d_6x+1)\\
&-\sum_{j=1}^{r}\frac{1}{(k+1+j)!}x^{k+1+j}\frac{(j-1)!d_6^j}{(d_6x+1)^j} \qquad (3.217)\\
&-\int\frac{1}{(k+1+r)!}x^{k+1+r}\frac{r!d_6^{r+1}}{(d_6x+1)^{r+1}}\,\mathrm{d}x
\end{aligned}
$$

To approximate the integral in (3.217), we build the sum over the first r summands and cut off the remaining integral. The error estimation of omitting the integral is given by

$$
\begin{aligned}
&\int_{x_L}^{x_R}\frac{1}{(k+1+r)!}x^{k+1+r}\frac{r!d_6^{r+1}}{(d_6x+1)^{r+1}}\,\mathrm{d}x \qquad (3.218)\\
&\qquad\leq\int_{x_L}^{x_R}\left|\frac{1}{(k+1+r)!}x^{k+1+r}\right|\left|\frac{r!d_6^{r+1}}{(d_6x+1)^{r+1}}\right|\mathrm{d}x\\
&\qquad\leq\int_{0}^{1}\left|\frac{r!d_6^{r+1}}{(d_6x+1)^{r+1}}\right|\mathrm{d}x \qquad (3.219)\\
&\qquad=\left|\frac{(r-1)!d_6^r}{(d_6+1)^r}\right|-\left|(r-1)!d_6^r\right| \qquad (3.220)\\
&\qquad=\mathcal{O}\left(d_6^r\right). \qquad (3.221)
\end{aligned}
$$

Thus, the error that occurs is $\mathcal{O}\left(d_6^r\right)$. Therefore, the number of summands r is chosen in such a way to guarantee a cut off error of $d_6^r < \Delta x^2$. Further, a general

minimum amount of summands is used to compute the integral. We examine the case $l = 2$.

$$\begin{aligned}
&\int x^k \ln^2(d_6x+1)\,\mathrm{d}x \\
&= \frac{1}{k+1}x^{k+1}\ln^2(d_6x+1) \\
&+ \sum_{j=1}^{r}(-1)^j \frac{1}{(k+1+j)!}x^{k+1+j}\frac{d_6^j(a_j\ln(d_6x+1)+b_j)}{(d_6x+1)^j} \\
&+ \int (-1)^{r+1}\frac{1}{(k+1+r)!}x^{k+1+r}\frac{d_6^{r+1}(a_{r+1}\ln(d_6x+1)+b_{r+1})}{(d_6x+1)^{r+1}}\,\mathrm{d}x\,,
\end{aligned} \tag{3.222}$$

with

$$a_j = 2(-1)^{j+1}(j-1)! \tag{3.223}$$

$$b_j = 2(-1)^j\sum_{i=1}^{j-1}\frac{(j-1)!}{i}. \tag{3.224}$$

A similar estimation as above leads to the following error estimate,

$$\begin{aligned}
&\int_{x_L}^{x_R}\frac{1}{(k+1+r)!}x^{k+1+r}\frac{d_6^{r+1}(a_{r+1}\ln(d_6x+1)+b_{r+1})}{(d_6x+1)^{r+1}}\,\mathrm{d}x \\
&\leq \int_{x_L}^{x_R}\left|\frac{1}{(k+1+r)!}x^{k+1+r}\right|\left|\frac{d_6^{r+1}(a_{r+1}\ln(d_6x+1)+b_{r+1})}{(d_6x+1)^{r+1}}\right|\mathrm{d}x
\end{aligned} \tag{3.225}$$

$$\leq \int_0^1\left|\frac{d_6^{r+1}(a_{r+1}\ln(d_6x+1)+b_{r+1})}{(d_6x+1)^{r+1}}\right|\mathrm{d}x \tag{3.226}$$

$$= \left|-\frac{d_6^r(a_{r+1}\ln(d_6x+1)+rb_{r+1}+a_{r+1})}{r^2(d_6x+1)^r}\right| \tag{3.227}$$

$$= \mathcal{O}(d_6^r)\,. \tag{3.228}$$

The computation for all other integrals for the cases for $k \in \{0,1,2\}$ and $l \in \{1,2,\ldots,5\}$ follow analogously.

Positive velocity: $|a_0| < \epsilon$ ***and*** $|a_n| < \epsilon$

The trajectory derived for positive velocity, and small coefficients $|a_0|$ and $|a_n|$ in the departure and arrival cell, is given by (3.79) and repeated

$$\widetilde{\varphi}(x,t^n,\Delta t) = \sum_{i=1}^{6} d_i x^{i-1}. \tag{3.229}$$

Similarly, the trajectory $\widetilde{\varphi}^2$ is determined and approximated by a series expansions about $a_0 = 0$ and $a_n = 0$. The complete form is given in the appendix in (A.23) and abbreviated by

$$\widetilde{\varphi}^2(x,\Delta t) = \sum_{i=1}^{7} e_i x^{i-1}. \tag{3.230}$$

Thus, the computation needed for the coefficient $m_{1,i}^{n+1}$ leads to the integral

$$m_{1,i}^{n+1}\Big|_{\text{part}} = \int \left(c_6x^2+c_7x+c_8\right)\left(\frac{2}{\Delta x_i}\left(\sum_{i=1}^{6} d_i x^{i-1}\right)-1\right)\mathrm{d}x\,. \tag{3.231}$$

For the coefficient $m_{2,i}^{n+1}$, we have to solve

$$m_{2,i}^{n+1}\Big|_{\text{part}} = \int \left(c_6x^2 + c_7x + c_8\right)\left(\frac{6}{\Delta x_i^2}\left(\sum_{i=1}^{7} e_i x^{i-1}\right) - \frac{6}{\Delta x_i}\left(\sum_{i=1}^{6} d_i x^{i-1}\right) + 1\right) \mathrm{d}x\,. \tag{3.232}$$

The computation of the integrals in (3.231) and (3.232) is integration over a polynomial of degree nine.

Negative velocity: $|a_0| < \epsilon$

The case of negative velocity and a small coefficient $|a_0|$ in the starting cell of a trajectory $\widetilde{\varphi}$ is treated analogously to the case of small $|a_0|$ and positive velocity.

The trajectories take the same form with different coefficients. The trajectory for negative velocity is described in (3.105) and given by

$$\widetilde{\varphi}(x, t^n, \Delta t) = d_1 \exp(d_2\, x)(1 + d_3x^2 + d_4x^3 + d_5x^4 + d_6x^5 + d_7x^6) + d8. \tag{3.233}$$

The approximation to the square of the trajectory is listed in the appendix in (A.26). The abbreviation with the respective coefficients d_k and g_k reads

$$\widetilde{\varphi}^2(x, t^n, \Delta t) = g1 \exp(g_2x)(1 + g_3x^2 + g_4x^3 + g_5x^4 + g_6x^5 + g_7x^6) + 2d_8d_1 \exp(d_2x)(1 + d_3x^2 + d_4x^3 + d_5x^4 + d_6x^5 + d_7x^6) + g_8. \tag{3.234}$$

The coefficients d_2 and g_2 that determine the way of the evaluation of the integral are given by

$$d_2 = \frac{a_n \Delta x_0}{u_0} \tag{3.235}$$

and

$$g_2 = 2\frac{a_n \Delta x_0}{u_0}. \tag{3.236}$$

Thus, depending on their size, the coefficients $m_{0,i}^{n+1}$, $m_{1,i}^{n+1}$ and $m_{2,i}^{n+1}$ are solved using direct integration by parts as in (3.178) or repeated reversed integration by parts described in (3.196), where the last integral term is omitted, when the cutoff error is small enough.

Negative velocity: $|a_n| < \epsilon$

The trajectory for negative velocity and a small coefficient $|a_n|$ in the arrival grid cell is given by (3.111). The abbreviation of the trajectory reads

$$\widetilde{\varphi}(x, t^n, \Delta t) = \sum_{k=1}^{5} d_k \ln(d_6x + 1)^{k-1}. \tag{3.237}$$

with according coefficients d_k. The square of the trajectory is approximated by a series expansion as well as written in the appendix in (A.28). The short form is given by

$$\widetilde{\varphi}^2(x, t^n, \Delta t) = e_1 + e_2 \ln(d_6x + 1) + e_3 \ln(d_6x + 1)^2 + e_4 \ln(d_6x + 1)^3 + e_5 \ln(d_6x + 1)^4 + e_6 \ln(d_6x + 1)^5. \tag{3.238}$$

To determine the solution of the coefficients $m_{0,i}^{n+1}$, $m_{1,i}^{n+1}$ and $m_{2,i}^{n+1}$ for the solution of the SASLDG time step the integrals in (3.170), (3.171) and (3.172) have to be computed. To do so, depending on the size of the coefficient d_6, given by

$$d_6 = \frac{\Delta x_0 a_0}{u_0}, \tag{3.239}$$

the integrals are solved by integration by parts either directly, leading to the analytically correct solution or in reversed direction as shown in (3.217), which approximates the solution. The numerical problem, which arises because of cancellation, the exact procedure of the computation and the discussion of the remedy are elucidated in the previous section for positive velocity and small coefficient a_n.

Negative velocity: $|a_0| < \epsilon$ ***and*** $|a_n| < \epsilon$

The trajectory derived for negative velocity, and small coefficients $|a_0|$ and $|a_n|$ in the starting and arrival cell, is given by (3.116) and repeated

$$\widetilde{\varphi}(x, t^n, \Delta t) = \sum_{i=1}^{6} d_i x^{i-1}. \tag{3.240}$$

Similarly, the trajectory $\widetilde{\varphi}^2$ determined and approximated by a series expansions about $a_0 = 0$ and $a_n = 0$. The complete form is given in the appendix in (A.30) and abbreviated by

$$\widetilde{\varphi}^2(x, t^n, \Delta t) = \sum_{i=1}^{7} e_i x^{i-1} \tag{3.241}$$

Thus, the computation needed for the coefficient $m_{1,i}^{n+1}$ leads to the integral

$$m_{1,i}^{n+1}\Big|_{\text{part}} = \int \left(c_6 x^2 + c_7 x + c_8\right) \left(\frac{2}{\Delta x_i}\left(\sum_{i=1}^{6} d_i x^{i-1}\right) - 1\right) \mathrm{d}x\,. \tag{3.242}$$

For the coefficient $m_{2,i}^{n+1}$, we have to solve

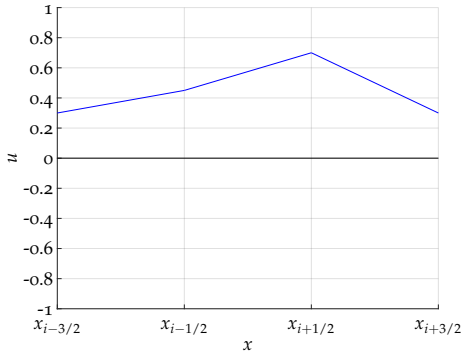
$$\begin{aligned} m_{2,i}^{n+1}\Big|_{\text{part}} = \int \left(c_6 x^2 + c_7 x + c_8\right) \Bigg(\frac{6}{\Delta x_i^2}\left(\sum_{i=1}^{7} e_i x^{i-1}\right) \\ - \frac{6}{\Delta x_i}\left(\sum_{i=1}^{6} d_i x^{i-1}\right) + 1\Bigg) \mathrm{d}x\,. \end{aligned} \tag{3.243}$$

The computation of the integrals in (3.242) and (3.243) is integration over a polynomial of degree nine.

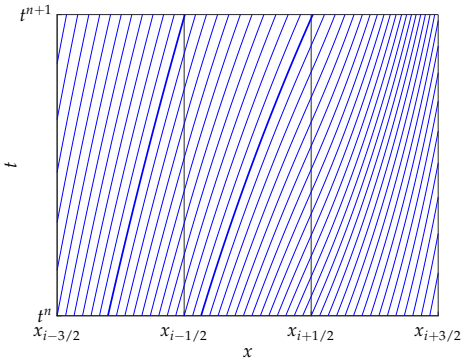
3.5 THE OVERALL ALGORITHM

As mentioned in the preview in Section 3.1, the SASLDG method consists of the main steps computation of the trajectories, determination of the analytical solution, and the projection step. The assembling of the trajectories is discussed in Section 3.2, the analytical solution is found in 3.3, and the projection step is elucidated in Section 3.4. In this Section we explain the procedure of the algorithm. The limits of integration are provided in Section 3.6.

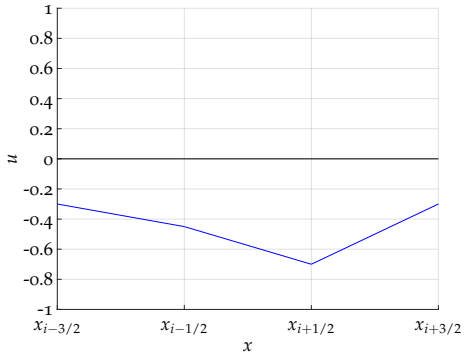
Before the solution to the advection equation can be computed, the trajectories need to be assembled and the domain of dependence must be known. In order to determine these, the type of the trajectories must be known for the current grid cell. The trajectories take different forms depending on the velocity distribution,



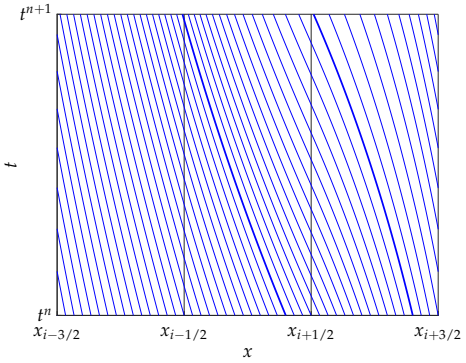
(a) Positive velocity distribution.



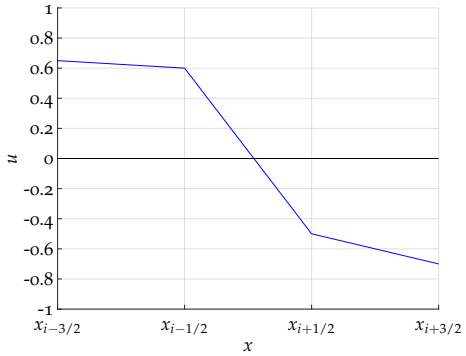
(b) Domain of dependence consists of the $i-1$th and the ith grid cell.



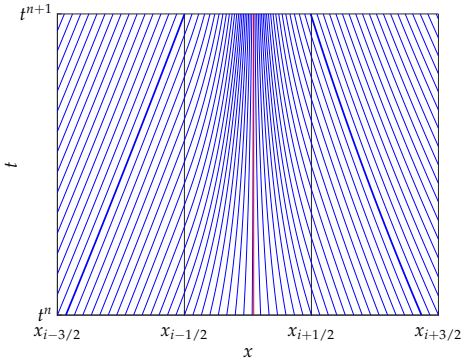
(c) Negative velocity distribution.



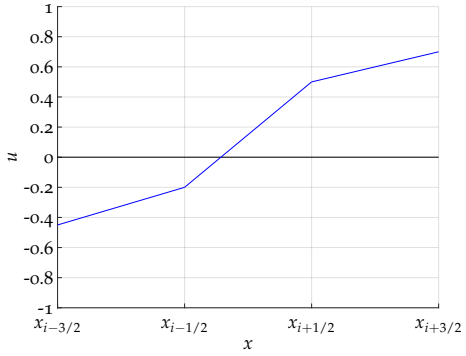
(d) Domain of dependence consists of the ith and the $i+1$th grid cell.



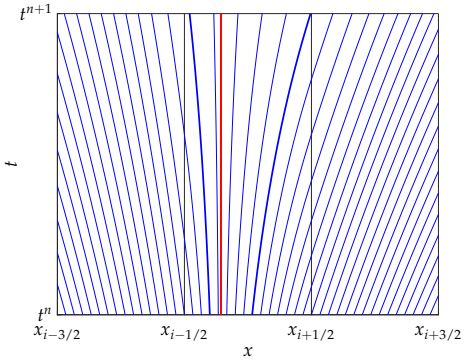
(e) Velocity distribution changing from positive to negative.



(f) Domain of dependence consists of the $i-1$th, the ith and the $i+1$th grid cell. The red trajectory marks the zero velocity point.



(g) Velocity distribution changing from negative to positive.



(h) Domain of dependence consists of the ith grid cell only. The red trajectory marks the zero velocity point.

Figure 3.11: Different scenarios of the velocity distributions with according trajectories, resulting in different domains of dependence between the bold blue printed trajectories.

i.e. positive or negative velocity, and on the number of involved grid cells. To find the correct type of trajectory, the algorithm of the SASLDG method checks the velocity of each grid cell as a first step: the velocity can be positive, negative, positive changing to negative and negative changing to positive. These possibilities along with the according trajectories are depicted in Figure 3.11. If the velocity is strictly positive in grid cell i as plotted in Figure 3.11(a), then the according trajectories in Figure 3.11(b) go to the left when followed backward in time. In this example the domain of dependence consists of the grid cells i and $i-1$. For strictly negative velocity, as shown in Figure 3.11(c), the trajectories go to the right, if followed backward in time. This implies a domain of dependence of grid cells i and $i+1$ in this example. The velocity in grid cell $[x_{i-1/2}, x_{i+1/2}]$ can change from positive to negative values, see Figure 3.11(e). The trajectories with endpoint in such a grid cell depart from the left and right neighboring grid cells. In the example illustrated in Figure 3.11(f) the domain of dependence consists of three grid cells, i.e. grid cell $i-1$, i and $i+1$. The scenario shown in Figure 3.11(g) gives an example for a velocity distribution changing from negative to positive velocity. All according trajectories remain within the ith grid cell as depicted in Figure 3.11(h). Thus, the domain of dependence lies within this grid cell.

After the determination of the domain of dependence and the corresponding trajectories the projection step takes place. The thorough discussion of the determination of the limits of integration needed in the projection step is conducted in the next section.

The algorithm for each time step of the SASLDG method can be summarized as follows.

(i) Determine the type of trajectory. Check, if
  (i) the velocity is strictly positive.
  (ii) the velocity is strictly negative.
  (iii) the velocity has positive slope and a root.
  (iv) the velocity has negative slope and a root.

(ii) Find domain of dependence of the ith grid cell.

(iii) Compute the departure points

$$x_{0,\text{left}} = \varphi(x_{i-1/2}, t^{n+1}, -\Delta t), \tag{3.244}$$

and

$$x_{0,\text{right}} = \varphi(x_{i+1/2}, t^{n+1}, -\Delta t). \tag{3.245}$$

(iv) Loop over all grid cells between the points $x_{0,\text{left}}$ and $x_{0,\text{right}}$:
  (i) Construct trajectory with departure point in current grid cell.
  (ii) Evolve density of the current grid cell along the trajectory to end point at time Δt.
  (iii) Determine limits of integration of the current grid cell for the projection step.
  (iv) Project density distribution onto a polynomial of degree two.

3.6 LIMITS OF INTEGRATION

The limits of integration that are part of the projection step which is described in Section 3.4, depend on the velocity field u and on the time step size. As illustrated

in Figure 3.10, to compute the result of the projection step for the ith grid cell, the interval $[x_{i-1/2}, x_{i+1/2}]$ is divided into subintervals. The projection is computed from values of the density or the tracer at time t^n. Therefore, the trajectories are followed backward in time and the complete interval that is integrated over, which possibly contains multiple grid cells, is given by $[\varphi(x_{i-1/2}, t^{n+1}, -\Delta t), \varphi(x_{i+1/2}, t^{n+1}, -\Delta t)]$. The form of the trajectory depends on the number of grid cells that are passed and whether coefficients a_k are close or equal to zero. In this section, we describe all possible scenarios and list the according limits of integration.

Positive velocity

We begin with the two principal types of trajectories that can occur, i.e. the trajectory that remains within a grid cell and the trajectory that crosses at least one grid cell boundary. So far, these were derived for following a tracer forward in time. The inverse case, which means going backward in time, is given here. A trajectory that remains within the ith grid cell which is the inverse to (3.19), takes the form

$$\varphi(x, t^{n+1}, -\Delta t) = e^{-a_i \Delta t} \left(\frac{b_i}{a_i} + x \right) - \frac{b_i}{a_i}. \tag{3.246}$$

The equation describing the trajectory going backward in time is the same as going forward in time apart, from the change of the sign. The existence is provided by Remark 3.2.2.

The trajectory that passes through at least two grid cells and proceeds in backward direction is given by

$$\varphi(x, t^{n+1}, -\Delta t) = \frac{u_1}{a_0} \exp\left(a_0 (T - \Delta t)\right) \left(\frac{a_n}{u_n} x + 1 \right)^{\frac{a_0}{a_n}} - \frac{u_0}{a_0}, \tag{3.247}$$

for $x \in [0, \Delta x_n]$. The trajectory φ in (3.247) forms the inverse to φ in (3.40). It can be computed directly by inversion of (3.40), or similarly to the derivation in Section 3.2.1 with a different ansatz proceeding in the reversed direction. Note that the labeling of the local indices does not change its order when the inverse is computed. That means that the trajectory going backward in time starts at the departure cell labeled with index n and ends in the arrival cell with index zero.

The algorithm of the SASLDG method detects if a limit of the integration lies within the arrival cell. If that is the case, the limits are given by

$$\begin{aligned} \varphi_L &= 0 \\ \varphi_R &= \frac{1}{\Delta x_0} \left(\frac{u_{n+1}}{a_n} \exp(-a_n \Delta t) - \frac{u_n}{a_n} \right). \end{aligned} \tag{3.248}$$

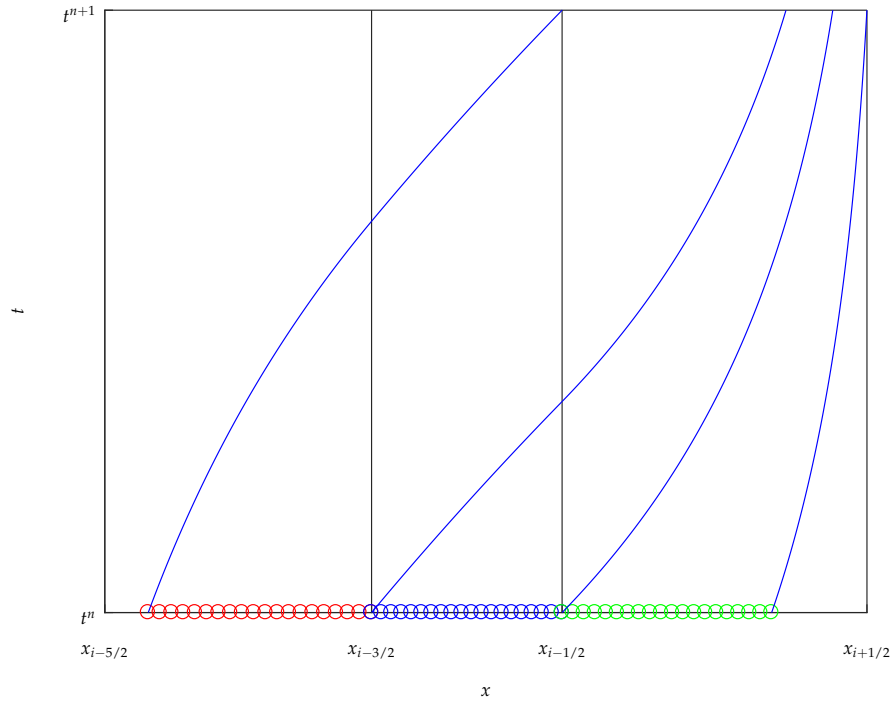
If a_n is small or equal to zero, a series expansion about the point $a_n = 0$ is carried out, exactly as for the computation of the trajectories in general. Then, the resulting limits yield

$$\begin{aligned} \varphi_L &= 0 \\ \varphi_R &= \frac{1}{\Delta x_0} \Big(\Delta x_0 + (a_n \Delta x_0 + u_n) \Big(-\Delta t + \frac{1}{2} a_n \Delta t^2 \\ &\quad - \frac{1}{6} a_n^2 \Delta t^3 + \frac{1}{24} a_n^3 \Delta t^4 \Big) \Big), \end{aligned} \tag{3.249}$$

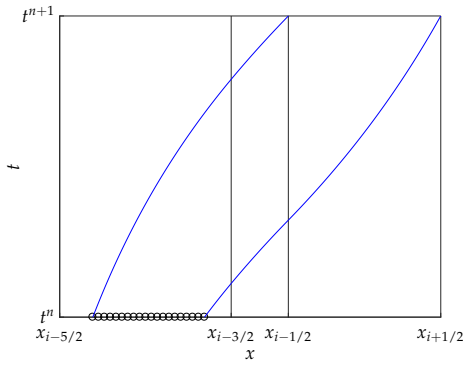
where the higher order terms are omitted. In the limiting case of $a_n \to 0$, the upper limit φ_R equals

$$\varphi_R = \Delta x_0 - u_n \Delta t. \tag{3.250}$$

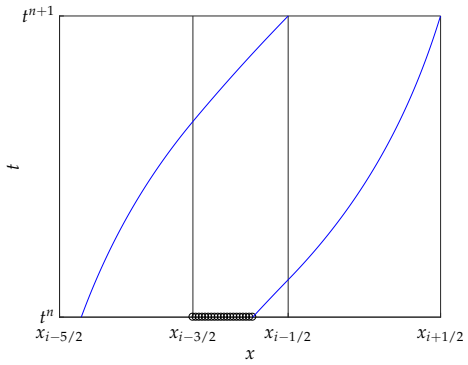
This case of constant velocity in grid cell i equals the assumptions of the SOM method. The interval $[0, \Delta x_0 - u_n \Delta t]$ is the same as the interval denoted by V^L of the SOM method, which represents the part that remains within the ith grid cell.



(a) The limits of integration of the red and blue marked intervals are computed from (3.247), the green one from (3.246). The three intervals build the domain of dependence for the ith grid cell.



(b) The limits of integration are computed from (3.247). The domain of dependence is included in one cell.



(c) The limits of integration are computed from (3.247). The marked interval is the last part of the domain of dependence.

Figure 3.12: Limits of integration.

An example for a limit of integration that lies in grid cell i is shown in Figure 3.12. The interval marked with green circles in Figure 3.12(a) is described by the limits in (3.248).

All other limits of integration are defined by trajectories that cross at least one grid cell boundary given by (3.247). These are determined by the following case environment. Let l be the total number of grid cells that are included in the domain of dependence for the ith grid cell. The variable $k = 1, \ldots, l$ denotes the counter that indicates the cell of which the integral is computed. All limits of integration are listed in transformed form because of the substitution applied in (3.41).

They yield

- if $l = 1$, i.e. the domain of dependence is included within one grid cell, illustrated in Figure 3.12(b)

$$\begin{aligned} \varphi_L &= -\frac{1}{\Delta x_0}\left(\frac{u_1}{a_0}\exp(a_0\,(T_L - \Delta t)) - \frac{u_0}{a_0}\right) + 1 \\ \varphi_R &= -\frac{1}{\Delta x_0}\left(\frac{u_1}{a_0}\exp(a_0\,(T_R - \Delta t)) - \frac{u_0}{a_0}\right) + 1, \end{aligned} \tag{3.251}$$

- if k = 1, i.e. the first part of the integral is computed, an example is the interval marked by red circles in Figure 3.12(a)

$$\begin{aligned} \varphi_L &= -\frac{1}{\Delta x_0}\left(\frac{u_1}{a_0}\exp(a_0\,(T_L - \Delta t)) - \frac{u_0}{a_0}\right) + 1 \\ \varphi_R &= 0, \end{aligned} \tag{3.252}$$

- if $k = l$, i.e. the last part of the integral is computed, an example is illustrated in Figure 3.12(c)

$$\begin{aligned} \varphi_L &= 1 \\ \varphi_R &= -\frac{1}{\Delta x_0}\left(\frac{u_1}{a_0}\exp(a_0\,(T_R - \Delta t)) - \frac{u_0}{a_0}\right) + 1, \end{aligned} \tag{3.253}$$

- else, i.e. a middle part of the integral is computed, an example is the interval marked by blue circles in Figure 3.12(a)

$$\begin{aligned} \varphi_L &= 1 \\ \varphi_R &= 0, \end{aligned} \tag{3.254}$$

where T_L and T_R are the sums of the intermediate time intervals determined from (3.22) from the starting point $(x_{i-1/2}, t^{n+1})$ or $(x_{i+1/2}, t^{n+1})$, respectively.

The equations (3.251), (3.252) and (3.253) are not defined for $a_0 = 0$. Therefore, the trajectories are approximated for small a_0 by means of a series expansion about $a_0 = 0$, the expressions for φ_L and φ_R without higher order terms change respectively to

$$\begin{aligned} \varphi_L = &-\frac{1}{\Delta x_0}(\Delta x_0 + (a_0\Delta x_0 + u_0)((T_L - \Delta t) + \frac{1}{2}a_0(T_L - \Delta t)^2 \\ &+ \frac{1}{6}a_0^2(T_L - \Delta t)^3 + \frac{1}{24}a_0^3(T_L - \Delta t)^4)) + 1 \end{aligned} \tag{3.255}$$

$$\begin{aligned} \varphi_R = &-\frac{1}{\Delta x_0}(\Delta x_0 + (a_0\Delta x_0 + u_0)((T_R - \Delta t) + \frac{1}{2}a_0(T_R - \Delta t)^2 \\ &+ \frac{1}{6}a_0^2(T_R - \Delta t)^3 + \frac{1}{24}a_0^3(T_R - \Delta t)^4)) + 1. \end{aligned} \tag{3.256}$$

Figure 3.12 shows the different scenarios for the limits of integration. Panel 3.12(a) illustrates three possibilities. The domain of dependence of grid cell i contains three grid cells, the cell i itself and the two neighboring cells to the left. The relevant part $[\varphi(x_{i-1/2}, t^{n+1}, \Delta t), \Delta x_{i-2}]$ of grid cell $i-2$, is marked with red circles. It gives an example for the lower limit of integration that is computed from a trajectory that crosses more than one grid cell and determined by (3.252). The complete interval of the $i-1th$ grid cell $[0, \Delta x_{i-1}]$ contributes to the integral marked with blue circles, described in the transformed version by (3.254). The last part of the domain of dependence for this example is part of the ith grid cell and marked with green circles. This part represents a limit of integration that is computed from a trajectory which remains within one grid cell, given by (3.248). Figure 3.12(b) shows the case of a domain of dependence that is included in one grid cell but outside of the arrival cell i. It is described by (3.251). The last case that can occur is pictured in Figure 3.12(c). This example represents a last interval in the domain of dependence, which does not lie in the arrival cell, i.e. it differs from (3.248). The limit of integration is derived from the trajectory in (3.247) and given by (3.253).

Negative velocity

The results for negative velocity can be determined analogously and are listed in the appendix in Section A.4.

Changing velocity from negative to positive

In the case all trajectories remain within one cell, the ith cell. That makes it very simple as it is given by (3.19). No determination is needed of how many grid cells are involved - it is just one.

Also, the polynomial character is preserved after the projection step, which makes the integration easy and exact without any approximations.

The left and right limit of the integration are given by

$$\varphi_L = \frac{1}{\Delta x_i}\left(\exp(-a_i\Delta t)\frac{b_i}{a_i} - \frac{b_i}{a_i}\right), \tag{3.257}$$

$$\varphi_R = \frac{1}{\Delta x_i}\left(\exp(-a_i\Delta t)\left(\frac{b_i}{a_i} + \Delta x\right) - \frac{b_i}{a_i}\right). \tag{3.258}$$

It is not possible that the coefficient a_i is small enough to create numerical problems during division. If $a_i < \epsilon$, it means that the velocity would be nearly constant in the whole cell. Further, u_i is negative and u_{i+1} positive, so the velocity would be nearly constant zero. That case is handled before.

Changing velocity from positive to negative

This case is separated into two parts: first the subinterval of positive velocity is treated exactly like the case for strictly positive velocity and second the subinterval of negative velocity is treated exactly as the case for purely negative velocity. However, there is one difference, the limits of integration must be adjusted with respect to the boundary of the two subintervals, which is the root of the velocity given by $R := -b_i/a_i$.

First, the subinterval of positive velocity is treated: The domain of dependence of the positive subinterval is bounded by $x_{0,\text{left}} = \varphi(0, t^{n+1}, -\Delta t)$ and $x_{0,\text{right}} = \varphi(R, t^{n+1}, -\Delta t)$. Note, that $x_{0,\text{right}} = R$, because $u(R) = 0$. Let l be the number of grid cells in the domain of dependence . If $l = 1$, the velocity would be (almost) equal to zero. That problem is captured and treated before, thus this case can be skipped. The algorithm loops over all grid cells in the domain of dependence. Let k be the counter. The case $k = 1$ is the same case as for strictly positive velocity. The limits of integration are given by

$$\varphi_L = -\frac{1}{\Delta x_0}\left(\frac{u_1}{a_0}\exp(a_0(T_L - \Delta t)) - \frac{u_0}{a_0}\right) + 1, \tag{3.259}$$

$$\varphi_R = 0. \tag{3.260}$$

All cases for $0 < k < l$ are treated in the same way as for strictly positive velocity

$$\varphi_L = 1, \tag{3.261}$$

$$\varphi_R = 0. \tag{3.262}$$

If $k = l$, the trajectories remain within grid cell i as $x_{0,\text{right}} = R \in [0, \Delta x_i]$. The integration interval is bounded by zero and the root R of the velocity distribution of that cell. The transformed limits of integration for the trajectories that remain within the ith grid cell are given by

$$\varphi_L = 0, \tag{3.263}$$

$$\varphi_R = \frac{R}{\Delta x_i}. \tag{3.264}$$

Analogously, the subinterval of negative velocity follows: The domain of dependence for this subinterval is bounded by $x_{0,\text{left}} = \varphi(R, t^{n+1}, -\Delta t)$ and $x_{0,\text{right}} = \varphi(\Delta x_i, t^{n+1}, -\Delta t)$. In this case we have $x_{0,\text{left}} = R$. Let l be the total number of grid cells contained in the domain of dependence. Note, that number of grid cells in the domains of dependence of the subintervals can differ. The case $l = 1$ can be skipped for the same reason as above. The counter k loops over the grid cells in the domain of dependence.

The case $k = 1$ leads to the following limits of integration

$$\varphi_L = 0, \tag{3.265}$$

$$\varphi_R = \frac{1}{\Delta x_0}\left(\frac{u_0}{a_0}\exp(a_0(T_R - \Delta t)) - \frac{u_0}{a_0}\right). \tag{3.266}$$

For the cases $0 < k < l$, it holds

$$\varphi_L = 0, \tag{3.267}$$

$$\varphi_R = 1. \tag{3.268}$$

For $k = l$, we have

$$\varphi_L = \frac{R}{\Delta x_i}, \tag{3.269}$$

$$\varphi_R = 1. \tag{3.270}$$

Attention must be paid to small coefficients a_0 or a_n. While running through the grid cells, the algorithm checks each time if small coefficients are present in the current cell. If so, it switches to the respective treatment. When the algorithm detects a small coefficient a_0, it works in the same way as for strictly positive or negative velocity. It changes the computation of the trajectories and the associated limits of integration accordingly. In grid cell i, where the root is present, the coefficients cannot be small or would mean a nearly constant velocity close to zero in this grid cell, which is handled before. For the same reason, the case of both coefficients being small cannot exist.

3.7 NUMERICAL RESULTS

In this section we show the numerical results of selected test cases computed by the SASLDG method. All tests are limited to one space dimension, results in two dimensions are described in Chapter 4.

The test cases are constructed in such a way to show the properties of the SASLDG method, in particular the extensions to the SOM method introduced in Section 2.3. These properties are the ability to handle velocity distributions with constant velocity as well as variable velocity in space, the capability to deal with Courant numbers of arbitrary size and an irregular grid. The option to apply a slope limiter should be given. Further, the numerical test results reveal the long time behavior of the shape conservation of initial data.

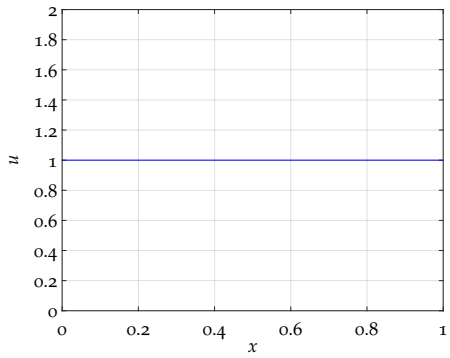
The test interval is the unity interval. We use periodic boundary conditions for all tests.

Two different velocity fields shown in Figure 3.13 are used for the numerical test cases. To simulate constant velocity we use the function

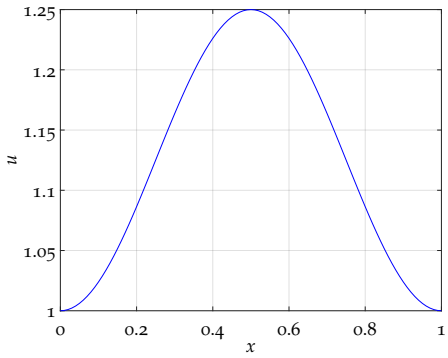
$$u(x) = 1 \tag{3.271}$$

displayed in Figure 3.13(a). For variable velocity we use the function

$$u(x) = \left(\frac{1}{2}\sin(\pi x)\right)^2 + 1 \tag{3.272}$$



(a) Constant velocity defined in (3.271).



(b) Variable velocity defined in (3.272).

Figure 3.13: The velocity fields used for all test cases.

shown in Figure 3.13(b).

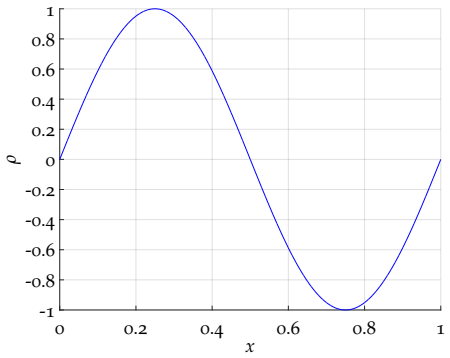
The initial values in form of the coefficients $m^0_{0,i}$, $m^0_{1,i}$ and $m^0_{2,i}$ are obtained from the projection of two functions onto the polynomial space $P^2(\Omega_i)$: First, we use a smooth sine function shown in Figure 3.14(a)

$$f(x) = \sin(2\pi x), \tag{3.273}$$

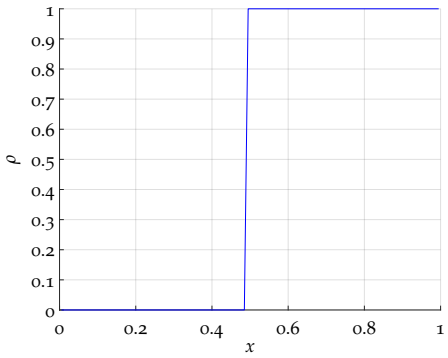
and second, a step function, illustrated in Figure 3.14(b), given by

$$f(x) = \begin{cases} 0, & \text{if} \quad x \leq 0.5 \\ 1, & \text{if} \quad x > 0.5 \end{cases} \tag{3.274}$$

is employed. Both functions reveal the performance of the SASLDG method on a different setting. For the smooth sine function it is important that the extrema are conserved and not damped because of the numerical diffusion. The numerical dispersion can cause wiggles and small oscillations in the vicinity of sharp discontinuities, which are to be avoided or to be kept small, simulated by the step function.



(a) Sine function defined in (3.273).



(b) Step function defined in (3.274).

Figure 3.14: Functions that are used for initial values.

We start to show the result of test cases solving the linear advection equation of the form (1.4) for the density ρ,

$$\rho_t + (u\rho)_x = 0. \tag{3.275}$$

Since the solution to ρy described in (1.5) is determined in the same way, we can restrict the numerical tests to solving the equation for the density. Concluding this

section, we discuss one test case, in which all variables the density ρ, tracer density ρy and tracer y are illustrated.

For each test case, we compute the maximum error l_∞ and the l_1-error from the mean values, i.e. we have

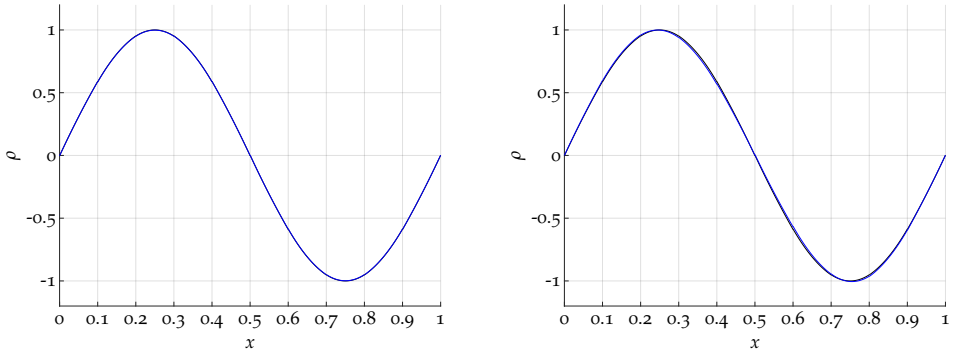
$$l_\infty = \max_i \left| m_{0,i}^N - m_{0,i}^0 \right|, \tag{3.276}$$

and

$$l_1 = \sum_i \Delta x_i \left| m_{0,i}^N - m_{0,i}^0 \right|, \tag{3.277}$$

where N is the total number of time steps taken to obtain $T_{\max} = N\Delta t$.

The first test case shows the results of the SASLDG method computed on a grid that consists of only ten grid cells. We apply the initial data obtained from (3.273). The Courant number is chosen to be 0.45. Figure 3.15 displays the results. Panel 3.15(a) shows the results computed with the constant velocity from (3.271), Panel 3.15(b) shows the respective results for variable velocity from (3.272).



(a) Results with constant velocity u.

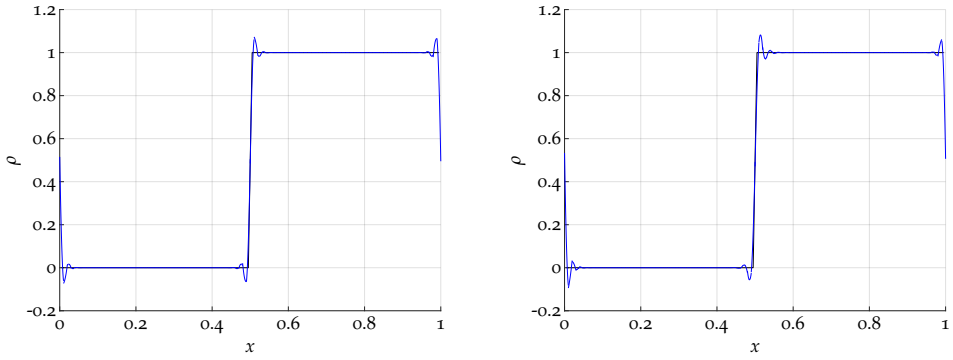
(b) Results with variable velocity u.

Figure 3.15: Numerical solution computed on ten grid cells, shown after 100 circulations.

The grid displayed in the figures equals the actual grid used for the computations. Both results are shown after 100 circulations, i.e. for the constant velocity case we have the maximum time $T_{\max} = 100$ to complete 100 circulations, for the variable velocity case it is $T_{\max} \approx 89.4$. The results are barely distinguishable from the initial values at time zero. The extremal points drop from 0.99997 of the initial distribution by 0.15 percent to the value 0.9984 for constant velocity, and by 0.07 percent to 0.9993 for variable velocity. The largest error occurs at the extremal points for constant velocity and is given by $l_\infty = 1.508 \times 10^{-3}$. The l_1-error takes the value 9.759×10^{-4}. Applying the variable velocity leads to larger errors of $l_\infty = 1.634 \times 10^{-2}$ and $l_1 = 8.088 \times 10^{-3}$. The maximum error is not located at the extrema of the numerical solution, but in the neighboring grid cells due to slight deformation of the solution.

The second test case, displayed in Figure 3.16, reveals the performance of the SASLDG method with respect to discontinuities in the initial values. Therefore, the function defined in (3.274) is used to determine the initial values. We compute the solution on a grid of 100 grid cells, because in that way the SASLDG method can be compared to the results of MPDATA shown in 2.6, where the same number of cells is used. For the same reason, the Courant number is chosen to be 0.5. Again, constant and variable velocity fields are applied to obtain the results. One circulation is computed, i.e. $T_{\max} = 1$ and $T_{\max} \approx 0.89$, respectively.

The result for constant velocity is shown in Figure 3.16(a). Both numerical solutions contain oscillations in the vicinity of the discontinuity. The amplitude of the



(a) Results with constant velocity u.

(b) Results with variable velocity u.

Figure 3.16: Advection of initial values with a sharp discontinuity for one circulation.

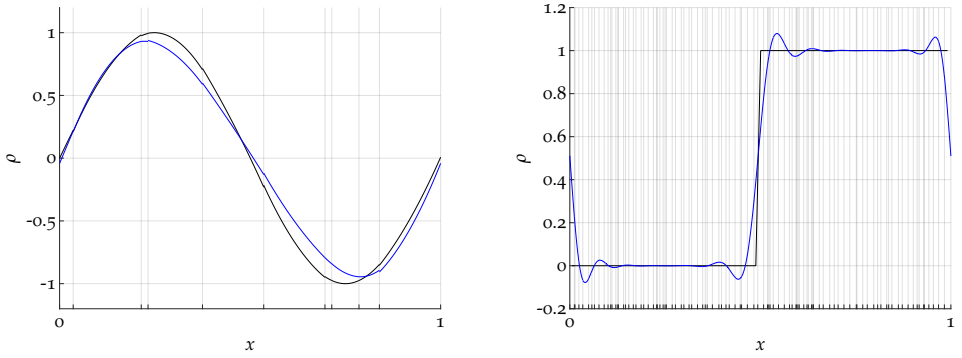
oscillations is measured as the maximum and minimum value of the whole polynomial distribution of the numerical solution. It is given for the constant velocity case by 1.0729 and -0.0729, respectively. The value of the amplitudes are of smaller magnitude if we measure maximum and minimum mean value (1.0390 and -0.0390). In comparison with the results of MPDATA shown in Figure 2.2(a) we observe that the maximum and minimum of the solution of MPDATA, given by 2.0493 and 0.9522, respectively, are smaller when compared to the whole polynomial distribution of SASLDG. However, if we consider mean values for the minimum and maximum point of the SASLDG solution, the oscillations of MPDATA are larger. The maximum error of MPDATA is given by 4.0647×10^{-1}, the l_1-error by 3.4876×10^{-2}. These errors are larger than the errors of the SASLDG method, which take the values $l_\infty = 1.555 \times 10^{-1}$ and $l_1 = 8.387 \times 10^{-3}$. Additionally, the oscillations of MPDATA affect more grid cells, i.e. they are stretched over a larger region, whereas the oscillations in the result of the SASLDG method are limited to few grid cells of a more narrow region.
The numerical solution computed with variable velocity is shown in Figure 3.16(b). The maximum takes the value 1.0819 and the minimum is given by -0.0945. Both values are larger as in the constant velocity case. The same holds for the errors, which are given by $l_\infty = 1.985 \times 10^{-1}$ and $l_1 = 9.131 \times 10^{-3}$.

The third test case shows that the SASLDG method is constructed in such a way that it is suitable for any one dimensional regular or irregular grid. The numerical results along with the analytical solution are shown in Figure 3.17 computed with CFL number 0.45, after 100 circulations and variable velocity, i.e. $T_{\max} \approx 89.4$. Note, that the time step size Δt is determined such that the local Courant number, defined by

$$\sigma_i = \frac{\max\limits_{0 \le x \le \Delta x} u_i(x) \Delta t}{\Delta x_i}, \tag{3.278}$$

is at most 0.45 for any grid cell. The irregular grid used in the computations is plotted in the respective figure. The first Panel 3.17(a) displays the numerical solution on an irregular grid of only ten grid cells with the sine function as initial values. The result is comparable with the test case computed on a regular grid shown in Figure 3.17(a). The maximum value is 0.9381, the minimum is reached at -0.9453. The shape of the numerical solution is more distorted than the corresponding solution on regular grid. Thus, the maximum error given by 1.520×10^{-1} is located at the largest distortion. The l_1-error takes the value 6.612×10^{-2}.

The numerical solution computed on an irregular grid of 100 cells using the step function as initial values shown in Figure 3.17(b) corresponds to the solu-

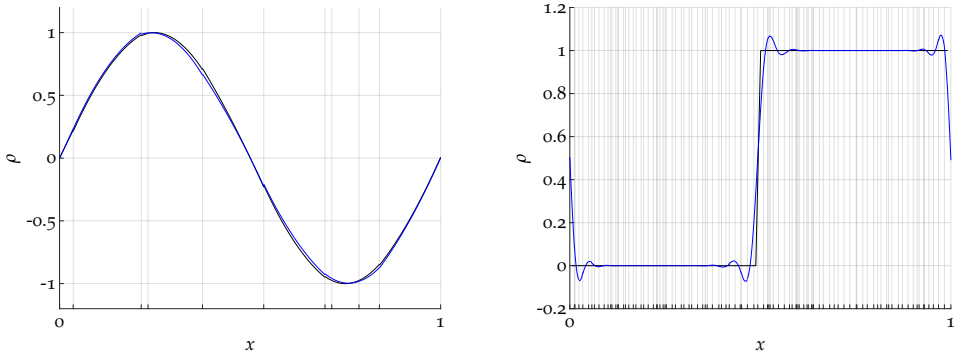


(a) Computations on ten grid cells. **(b)** Computations on 100 grid cells.

Figure 3.17: Numerical and analytical solution after 100 circulations computed on an irregular grid, using CFL number 0.45 and variable velocity.

tion in Figure 3.16(b) with the difference of computing 100 circulations and the Courant number 0.45. Note, the oscillations strech over a larger area. The maximum and minimum peak of the oscillations takes the value 1.0794 and -0.0779, respectively. The l_∞-error is given by 3.919×10^{-1} and the l_1-error by 3.542×10^{-2}. The same test with 100 circulations computed on regular grid yields smaller errors $l_\infty = 3.549 \times 10^{-1}$ and a l_1-error given by 2.197×10^{-2}.

In the last test case we have shown the numerical solution computed on an irregular grid. However, the time step size Δt was chosen in such a way that the local CFL number of 0.45 was the maximum CFL number for all cells. This means that the trajectories departing in cell i would reach at most to the neighboring grid cell $i+1$ (for positive velocity). In the following and fourth test case, we choose a large CFL number of value 10. In this way, most trajectories will go further than to the direct neighboring cell depending on the velocity and grid cell size. We compute the numerical solution with the same setting used for Figure 3.17, with the difference of the CFL number changed to 10.



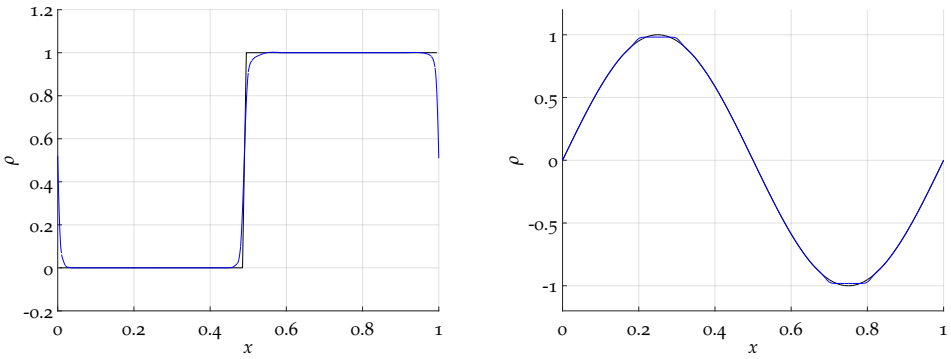
(a) Computations on ten grid cells. **(b)** Computations on 100 grid cells.

Figure 3.18: Numerical and analytical solution after 100 circulations computed on an irregular grid, using CFL number 10 and variable velocity.

The results correspond to the results with a smaller CFL number of 0.45 as shown in Figure 3.17. The distortion of the shape of the numerical solution seen in Figure 3.17(a) is larger than that of the result in Figure 3.18(a). It maintains better the form of the initial distribution. Because of the larger Courant number, the errors drop to $l_\infty = 2.558 \times 10^{-2}$ and $l_1 = 1.573 \times 10^{-2}$. The same test on a regular grid leads to decreased errors $l_\infty = 4.348 \times 10^{-4}$ and $l_1 = 1.795 \times 10^{-4}$.

A similar result can be observed in Figure 3.18(b). The maximum and minimum value of the oscillations given by 1.0714 and -0.0725, respectively. Thus, the oscillations are of smaller amplitude than in the case of CFL number 0.45. Furthermore, the oscillations remain in a smaller area of approximately half as many grid cells, which leads to a better shape preservation. Also, the maximum error given by 3.472×10^{-1} and the l_1-error of 2.127×10^{-2} are smaller than in the corresponding case with smaller CFL number. In fact, these errors are smaller than the errors on regular grid with CFL number 0.45. The case of applying CFL number 10 on regular grid leads to a smaller error of $l_\infty = 2.433 \times 10^{-1}$ and $l_1 = 1.192 \times 10^{-2}$.

The next test case is listed to illustrate the applicability of limiters to the SASLDG method shown in Figure 3.19. A possibility is given by the limiters introduced for the SOM method in Chapter 2.3. In this test case we use the step function given in (3.274) as initial values to demonstrate the ability to avoid oscillations. Further, we use the smooth sine function of (3.273) to show that the extremal points are not clipped. The numerical solution is computed on a grid of 100 cells and with a Courant number of 0.5 for both cases.



(a) Solution computed with variable velocity, shown after one circulation.

(b) Solution computed with constant velocity, shown after 100 circulations.

Figure 3.19: Numerical and analytical solution with the application of a limiter, computed on 100 grid cells and Courant number 0.5.

In this test case we use the limiter described in (2.96) in combination with the bounds defined in (2.95). Because of the variable velocity used in Figure 3.19(a), the density ρ does not remain between the bounds zero and one. It reaches the maximum value of 1.24 during the whole test, as shown in the subsequent test without the usage of a limiter. Therefore, it is reasonable to define the bounds as in (2.95), which takes the minimum and maximum of ρ of the neighboring grid cells, respectively. Figure 3.19(a) shows the results of the numerical solution after one circulation. As required, the oscillations are not existent. The numerical solution preserves positivity since the minimum value equals zero. The maximum takes the value 1.0022. The maximum error is given by 2.919×10^{-1} and the l_1-error by 1.253×10^{-2}. These errors are larger than in the corresponding test case shown in Figure 3.16(b). Because of the slope limiting, the oscillations are avoided, however, the sharpness of the discontinuity is not preserved as well.
In Figure 3.19(b) we can see the effect of the limiter on extrema after 100 circulations. We can observe that the extremal points are flattened, but their amplitude does not decrease significantly. The maximum of the solution is given by 0.9816. The maximum- and l_1-error is given by 1.775×10^{-2} and 3.409×10^{-3}, respectively.

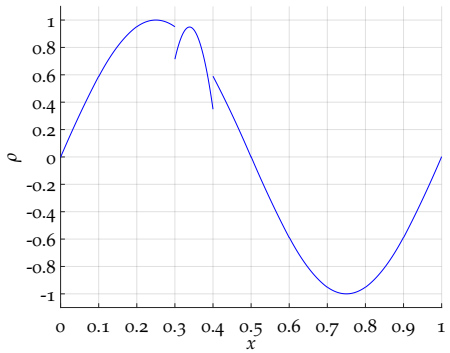
Test case number six reveals an interesting property of the algorithm. If we use smooth initial data and perturb the slope or the curvature of a single coefficient, this pertubation is damped out after a few time steps. In this example, we use the total number of 10 grid cells and the Courant number 0.45. We perturb the third

coefficient in the 4th grid cell, i.e. $m_{2,4}$, and enlarge the curvature by a factor of ten. The results of the development of the pertubation are shown in Figure 3.20.

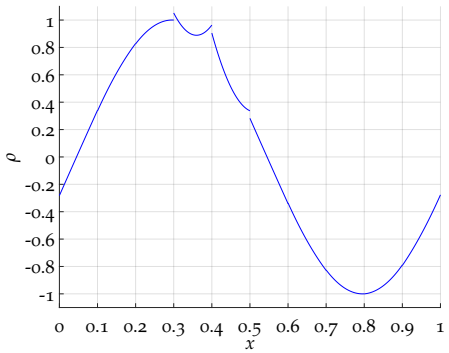
We plot the initial data with the pertubation in Figure 3.20(a). Figures 3.20(b) and 3.20(c) show the evolution of the pertubation after one and three time steps, respectively. After one time step, the advected pertubation is visible in the forth and fifth grid cell, after two more time steps it is present in grid cells five and six accordingly to the time step size. It is remarkable that the pertubation seems to vanish in time. In particular, after one circulation at time $T_{max} = 1$, displayed in Figure 3.20(d), the numerical result barely shows any sign of the initial pertubation. The main difference is the difference of the maximum values. The maximum of the numerical solution without the pertubation would take the value 0.99993, whereas with the initial pertubation it reaches the value 0.98879.

As mentioned above, the concluding test case of the numerical results illustrates the correlation of the density ρ, tracer density ρy and the tracer y. This is done by means of figures showing the evolution in time for one circulation. The initial density ρ is chosen to be a function constant to one and the tracer density ρy is the step function defined in (3.274). The tracer y is obtained by the quotient of density and tracer density. The velocity field is variable in space and given by (3.272). We compute the numerical solution on a grid of 100 grid cells and choose the Courant number to be 0.45. The results are shown at time $T_{\max} \approx 0.89$, which means after one circulation, in Figure 3.21.

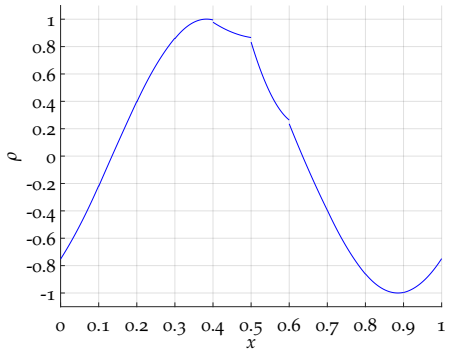
The panels on the left side, i.e. Figures 3.21(a), 3.21(d), 3.21(g), 3.21(j) and 3.21(m) display the evolution of one circulation for the density ρ. The panels in the middle as there are Figures 3.21(b), 3.21(e), 3.21(h), 3.21(k) and 3.21(n) show the numerical results for the tracer density ρy. The panels on the right side (see Figures 3.21(c), 3.21(f)) 3.21(i), 3.21(l) and 3.21(o), display the quotient of tracer density and density,



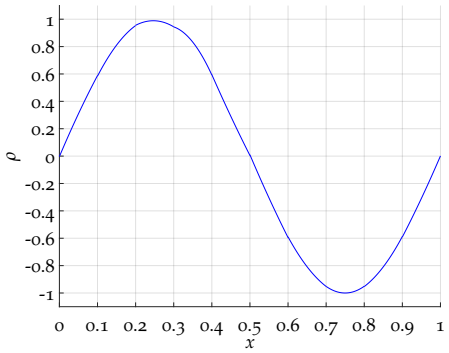
(a) Initial pertubation at time $t = 0$.



(b) Numerical solution after one time step.

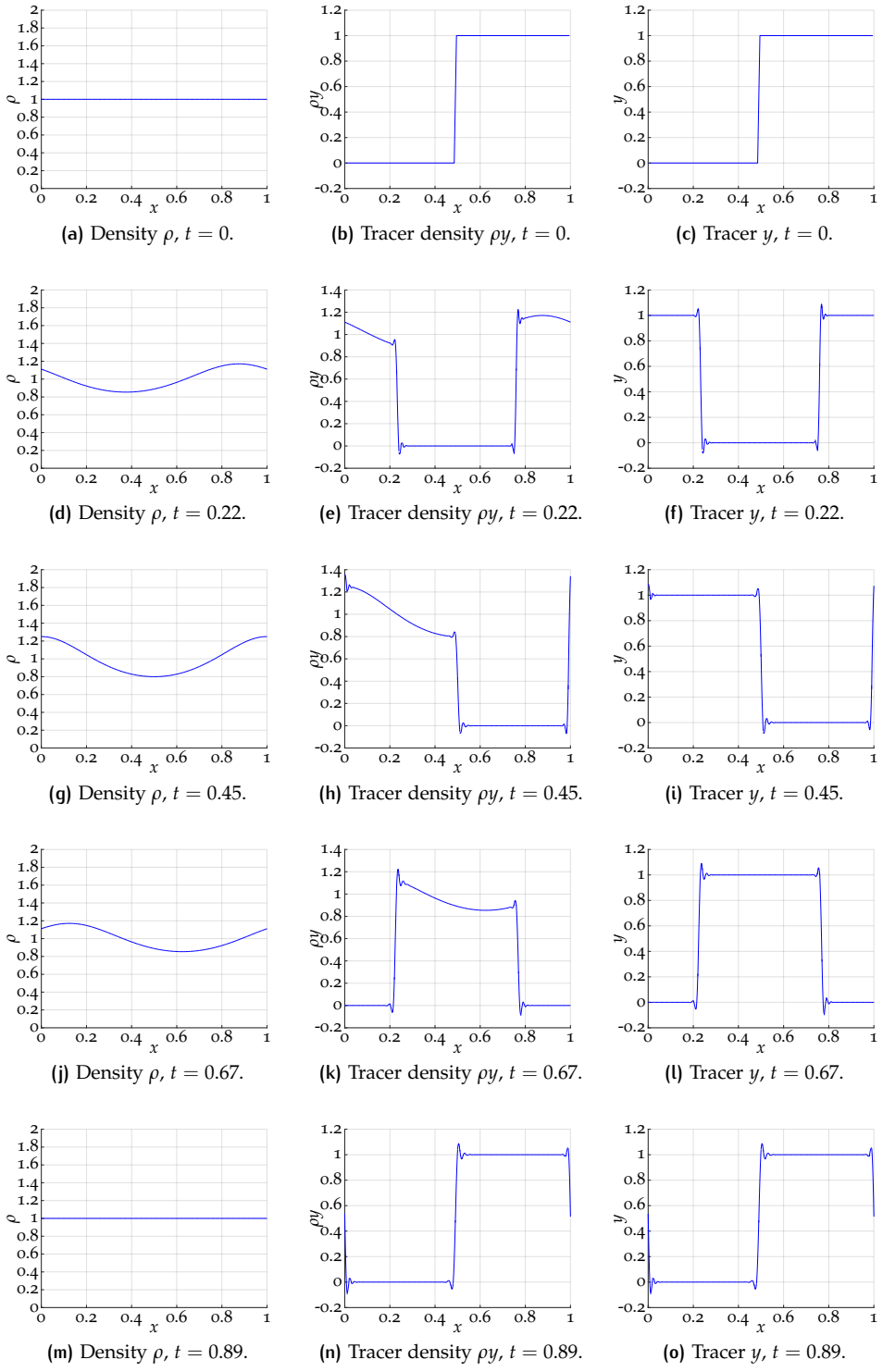


(c) Numerical solution after three time steps.



(d) Numerical solution after one circulation.

Figure 3.20: Evolution in time of a single initial pertubation of the coefficient $m_{2,4}$. The solution is computed on a grid of 10 cells with CFL number 0.45.



(a) Density ρ, $t = 0$. **(b)** Tracer density ρy, $t = 0$. **(c)** Tracer y, $t = 0$.

(d) Density ρ, $t = 0.22$. **(e)** Tracer density ρy, $t = 0.22$. **(f)** Tracer y, $t = 0.22$.

(g) Density ρ, $t = 0.45$. **(h)** Tracer density ρy, $t = 0.45$. **(i)** Tracer y, $t = 0.45$.

(j) Density ρ, $t = 0.67$. **(k)** Tracer density ρy, $t = 0.67$. **(l)** Tracer y, $t = 0.67$.

(m) Density ρ, $t = 0.89$. **(n)** Tracer density ρy, $t = 0.89$. **(o)** Tracer y, $t = 0.89$.

Figure 3.21: Evolution in time of the numerical solution of the density, the tracer density and the tracer. The solution is computed on a grid of 100 cells, with variable velocity and CFL number 0.45 for one circulation.

which yields the tracer. The respective results are plotted at time zero, after a quarter circulation at $t \approx 0.22$, after half a circulation at $t \approx 0.45$, after three quarters at $t \approx 0.67$ and a full circulation at $t \approx 0.89$. The numerical solution of ρy is similar to the result shown in Figure 3.16(b) but with a change of the CFL number from 0.5 to 0.45. Note the different scales in the panels. The distribution of the tracer remains in the interval $[0, 1]$ apart from the oscillations. The scales of the density and tracer density vary.

We conclude the section of the numerical tests with test series that show the influence of the CFL number on the maximum error l_∞ as well as the l_1-error and on the computational time. The CFL numbers smaller than one 0.2, 0.4, 0.5, 0.6, 0.8 show the results for advection to the direct neighboring grid cell within one time step. The larger CFL numbers 1.6, 3.2 and 6.4 reveal the results when more grid cells are affected. The last CFL number is chosen such that final time $T_{\max}$ is obtained with one single time step, i.e. $\Delta t = T_{\max}$. All performance measurements are taken on a machine with the MATLAB bench characteristics listed in the appendix in Section A.7.

CFL	l_∞-error	l_1-error	time (s)
0.2	4.047e-03	2.620e-03	9.31
0.4	1.799e-03	1.164e-03	4.83
0.5	1.322e-03	8.554e-04	4.09
0.6	1.220e-03	7.895e-04	3.15
0.8	1.109e-03	7.180e-04	2.58
1.6	4.302e-04	2.784e-04	1.49
3.2	2.774e-04	1.796e-04	0.90
6.4	1.566e-04	1.014e-04	0.38
1000.0 ($\Delta t = T_{\max}$)	8.279e-12	5.358e-12	0.14

Table 3.1: Errors and computational time of the solution after 100 circulations computed with constant velocity on 10 cells of a regular grid and the sine function as IV.

Table 3.1 shows the errors and the respective computational time for the computation of the numerical solution after 100 circulations, constant velocity on a regular grid of ten cells. The sine function is used as initial distribution.

CFL	l_∞-error	l_1-error	time (s)
0.2	3.267e-02	1.745e-02	7.29
0.4	1.898e-02	9.581e-03	3.55
0.5	1.404e-02	6.854e-03	2.96
0.6	1.113e-02	5.557e-03	2.56
0.8	1.089e-02	5.176e-03	1.85
1.6	6.016e-03	2.615e-03	1.30
3.2	2.346e-03	9.585e-04	0.79
6.4	1.089e-03	4.944e-04	0.48
1117.59 ($\Delta t = T_{\max}$)	1.324e-06	8.128e-07	0.10

Table 3.2: Errors and computational time of the solution after 100 circulations computed with variable velocity on 10 cells of a regular grid and the sine function as IV.

Table 3.2 shows the corresponding results to Table 3.1, but with variable instead of constant velocity. Thus, the last CFL number of the series is changed to 1117.59.

The same setting except for the regularity of the grid is used for the next test series shown in Table 3.3. The grid consists of 10 cells, which are randomly distributed. It was used before in the test shown in Figures 3.17(a) and 3.18(a).

Table 3.4 shows the l_∞-error, the l_1-error and the computational time of the respective numerical solution after one circulation computed with the step function

CFL	l_∞-error	l_1-error	time (s)
0.2	1.396e-01	6.505e-02	33.95
0.4	1.562e-01	6.887e-02	17.15
0.5	1.351e-01	6.308e-02	13.67
0.6	1.602e-01	6.992e-02	11.46
0.8	1.643e-01	7.107e-02	8.66
1.6	1.190e-01	5.810e-02	4.64
3.2	9.322e-02	4.781e-02	2.46
6.4	4.945e-02	2.450e-02	1.42
6240.18 ($\Delta t = T_{\max}$)	1.561e-06	8.112e-07	0.10

Table 3.3: Errors and computational time of the solution after 100 circulations computed with variable velocity on 10 cells of an irregular grid and the sine function as IV.

CFL	l_∞-error	l_1-error	time (s)
0.2	2.081e-01	1.102e-02	9.14
0.4	1.756e-01	9.631e-03	4.54
0.5	1.555e-01	8.387e-03	3.69
0.6	1.601e-01	8.724e-03	3.07
0.8	1.517e-01	7.977e-03	2.38
1.6	1.207e-01	5.705e-03	1.24
3.2	9.859e-02	4.147e-03	0.69
6.4	8.411e-02	3.470e-03	0.53
100.0 ($\Delta t = T_{\max}$)	4.441e-14	8.882e-16	0.17

Table 3.4: Errors and computational time of the solution after one circulation computed with constant velocity on 100 cells of a regular grid and the step function as IV.

as initial values. The grid used for this test consists of 100 cells. Constant velocity is applied.

Table 3.5 shows the corresponding minimum and maximum values of the numer-

CFL	min (mean)	max (mean)	min	min
0.2	1.0695	-0.0695	1.0854	-0.0854
0.4	1.0575	-0.0575	1.0828	-0.0828
0.5	1.0390	-0.0390	1.0729	-0.0729
0.6	1.0474	-0.0474	1.0916	-0.0916
0.8	1.0416	-0.0416	1.1047	-0.1047
1.6	1.0177	-0.0177	1.1004	-0.1004
3.2	1.0068	-0.0068	1.0991	-0.0991
6.4	1.0026	-0.0026	1.0899	-0.0899
100.0 ($\Delta t = T_{\max}$)	1.0000	0.0000	1.0000	-4.025e-14

Table 3.5: Minimum and maximum of means of the solution and of whole distribution after one circulation computed with constant velocity on 100 cells of a regular grid and the step function as IV.

ical solution in two different versions. First, the minimum and maximum is given computed from the mean values of each grid cell, i.e. only from the coefficients $m_{0,i}$ (marked by mean). Since the numerical solution is described by piecewise polynomial functions, these can be used as well for the determination of the minimum and maximum. The whole distribution is therefore used for the determination of the second information given in the table.

4 EXTENSION OF THE SASLDG METHOD IN 2D

In two space dimensions, the linear advection with variable velocity coefficient is given by

$$\frac{\partial y}{\partial t} + \mathbf{u} \cdot \nabla y = 0, \tag{4.1}$$

extending the one-dimensional equation described in (1.3). Analogously to the procedure in Chapter 1, the equation in non-conservative form is transferred into a system of two equations that are in conservation form. This is also done by introducing the variable ρ, which represents the density,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{4.2}$$

$$\frac{\partial \rho y}{\partial t} + \nabla \cdot (\rho y \mathbf{u}) = 0 \tag{4.3}$$

These equations correspond to the one-dimensional equations (1.4) and (1.5).

To extend a numerical method that is constructed to solve a one-dimensional problem to more space dimensions, a dimensional splitting approach can be applied. The more dimensional advection equation is solved along each coordinate direction alternately as a one-dimensional problem. This technique is described by Strang in [62].

The rest of this chapter is organized as follows. After a brief description of the Strang splitting in the next section (4.1), the necessary changes of the 2D extension of the SASLDG method are discussed in Section 4.2. Numerical tests of the SASLDG method in two space dimensions are carried out. A hybrid version of the operator splitting, which combines the two schemes MPDATA and a special form of the SASLDG method, is described in the concluding section (4.3) of this chapter. The performance of this hybrid method is examined and compared with solutions of MPDATA and the SASLDG method. All performance measurements are taken on a machine with the MATLAB bench characteristics listed in the appendix in Section A.7.

4.1 OPERATOR SPLITTING

Strang splitting [62] relies on the idea to divide a certain problem into subproblems, solve these easier subproblems and assemble the solutions of the subproblems to the solution of the original problem. This applies to the reduction of space dimensions as well as to other problems as e.g. the advection equation with source terms shown for example in [36].

We define the solution operator $S_{\Delta t}$ of the two-dimensional problem (4.2),

$$\rho^{n+1} = S_{\Delta t} \rho^n. \tag{4.4}$$

Then, $L^x_{\Delta t}$ is the fractional solution operator in x-direction, i.e. in one dimension, solving (1.4),

$$\rho^{n+1} = L^x_{\Delta t} \rho^n. \tag{4.5}$$

The fractional solution operator $L^y_{\Delta t}$ is defined analogously. A straight forward way to combine the solution operator for both space dimensions would be of the form

$$S_{\Delta t} = L^x_{\Delta t} L^y_{\Delta t}. \tag{4.6}$$

However, this approach can only be first-order accurate. It can be shown that if we use Strang splitting of the form

$$S_{\Delta t} = L^x_{\Delta t/2} L^y_{\Delta t} L^x_{\Delta t/2}, \tag{4.7}$$

second-order accuracy can be attained, if both fractional operators are at least of order two. Thus, to compute a full 2D time step with a numerical method, first, half a time step is made into x-direction, then a full time step into y-direction and concludingly, again half a time step into x-direction. Certainly, the directions can be switched, that is first a half a time step in y-direction and so on. Thus, the space dimension with more varying velocity can be chosen to be one with half time steps and thus gain more accuracy.

4.2 THE SASLDG METHOD IN 2D

The representation of the density ρ_i for the ith grid cell in polynomial form of degree two in two space dimensions takes the form

$$\begin{aligned}\rho_i(x,y,t^n) = m^n_{0,i} K^{(i)}_0(x,y) + m^n_{x,i} K^{(i)}_x(x,y) + m^n_{xx,i} K^{(i)}_{xx}(x,y) \\ + m^n_{y,i} K^{(i)}_y(x,y) + m^n_{yy,i} K^{(i)}_{yy}(x,y) + m^n_{xy,i} K^{(i)}_{xy}(x,y),\end{aligned} \tag{4.8}$$

with coefficients describing the distribution in x and in y-direction. Additionally, there is one cross term to fully define the polynomial in two dimensions. Accordingly, the Legendre basis functions are extended to

$$K^{(i)}_0(x,y) = 1 \tag{4.9}$$

$$K^{(i)}_x(x,y) = \frac{2x - \Delta x}{\Delta x} \tag{4.10}$$

$$K^{(i)}_{xx}(x,y) = \frac{1}{2}\left(3\left(\frac{2x-\Delta x}{\Delta x}\right)^2 - 1\right) \tag{4.11}$$

$$K^{(i)}_y(x,y) = \frac{2y - \Delta y}{\Delta y} \tag{4.12}$$

$$K^{(i)}_{yy}(x,y) = \frac{1}{2}\left(3\left(\frac{2y-\Delta y}{\Delta y}\right)^2 - 1\right) \tag{4.13}$$

$$K^{(i)}_{xy}(x,y) = \frac{2x - \Delta x}{\Delta x}\frac{2y - \Delta y}{\Delta y}. \tag{4.14}$$

According to the Strang splitting defined in (4.7), we solve the advection equation in both space dimensions in turn. The idea and the main parts of the algorithm for the 2D version of the SASLDG method remain unchanged in comparison with the version in one space dimension. However, the projection step is adapted, since a polynomial of form (4.8) must be obtained. Note that we assume a grid of equidistantly distributed grid cells in each space dimension because it simplifies the projection step. The usage of an irregular grid - as long as it is rectilinear - is possible, however, the costs of the computations of the integrals increase.

The projections for the time step in x-direction yield the new coefficients $m^{n*}_{0,i}$, $m^{n*}_{x,i}$, $m^{n*}_{xx,i}$, $m^{n*}_{y,i}$, $m^{n*}_{yy,i}$ and $m^{n*}_{xy,i}$ at the intermediate time level $n*$ after the fractional step operator $L^x_{\Delta t/2}$ is applied to the numerical solution at the old time level n.

Analogously to (3.138) in one space dimension, we obtain the projection formula,

$$\begin{aligned}&\mathbb{P}\rho(x,y,t^{n*}) \\ &= \sum_{j=1}^{6} \frac{\int_0^{\Delta y}\int_{x_L}^{x_R} \rho(x,y,t^n) K^{(i)}_j(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}x\,\mathrm{d}y}{\int_0^{\Delta y}\int_0^{\Delta x} K^{(i)}_j(x) K^{(i)}_j(x)\,\mathrm{d}x\,\mathrm{d}y} K^{(i)}_j(x,y),\end{aligned} \tag{4.15}$$

where we use $x_L =: \varphi(x_{i-1/2}, t^{n*}, -\Delta t/2)$ and $x_R =: \varphi(x_{i+1/2}, t^{n*}, -\Delta t/2)$ to abbreviate the expression. Note that we make only half a time step in x-direction. In constrast to the one dimensional projection in (3.138), the numerical solution in (4.15) describes an intermediate solution at an intermediate time level, denoted by t^{n*}. The integrals consist of subintervals that lie in different grid cells. We refer to Section 3.4 for details. Thus, we compute the subintervals separately. The limits of the integration that can occur for each subinterval is discussed in Section 3.5.

A part of the solution is computed for grid cell i, with the departure cell d, where the trajectory $\varphi(x, t^{n*}, -\Delta t/2)$ ends at time t^n. We have,

$$\begin{aligned} m^{n*}_{0,i}\big|_{\text{part}} &= \frac{1}{\Delta x \Delta y} \int_0^{\Delta y} \int_{x_1}^{x_2} \rho_d(x,y,t^n) K_0^{(i)}(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{1}{\Delta x} \int_{x_1}^{x_2} \left(m^n_{0,d} K_0^{(d)} + m^n_{x,d} K_x^{(d)} + m^n_{xx,d} K_{xx}^{(d)} \right) \mathrm{d}x\,, \end{aligned} \tag{4.16}$$

$$\begin{aligned} m^{n*}_{x,i}\big|_{\text{part}} &= \frac{3}{\Delta x \Delta y} \int_0^{\Delta y} \int_{x_1}^{x_2} \rho_d(x,y,t^n) K_x^{(i)}(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{3}{\Delta x} \int_{x_1}^{x_2} K_x^{(i)}(\varphi(x,t^n,\Delta t/2),y) \left(m^n_{0,d} K_0^{(d)} + \right. \\ &\quad \left. + m^n_{x,d} K_x^{(d)} + m^n_{xx,d} K_{xx}^{(d)} \right) \mathrm{d}x\,, \end{aligned} \tag{4.17}$$

$$\begin{aligned} m^{n*}_{xx,i}\big|_{\text{part}} &= \frac{5}{\Delta x \Delta y} \int_0^{\Delta y} \int_{x_1}^{x_2} \rho_d(x,y,t^n) K_{xx}^{(i)}(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{5}{\Delta x} \int_{x_1}^{x_2} K_{xx}^{(i)}(\varphi(x,t^n,\Delta t/2),y) \left(m^n_{0,d} K_0^{(d)} + \right. \\ &\quad \left. + m^n_{x,d} K_x^{(d)} + m^n_{xx,d} K_{xx}^{(d)} \right) \mathrm{d}x\,, \end{aligned} \tag{4.18}$$

$$\begin{aligned} m^{n*}_{y,i}\big|_{\text{part}} &= \frac{3}{\Delta x \Delta y} \int_0^{\Delta y} \int_{x_1}^{x_2} \rho_d(x,y,t^n) K_y^{(i)}(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{3}{\Delta x \Delta y} \int_{x_1}^{x_2} \int_0^{\Delta y} m^n_{y,d} K_y^{(d)}(x,y) K_y^{(i)}(\varphi(x,t^n,\Delta t/2),y) \\ &\quad + m^n_{xy,d} K_{xy}^{(d)}(x,y) K_y^{(i)}(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}y\,\mathrm{d}x \\ &= \frac{3}{\Delta x \Delta y} \int_{x_1}^{x_2} m_y \frac{\Delta y}{3} + m_{xy} \frac{\Delta y}{3} K_x^{(d)}(x,y)\,\mathrm{d}x \\ &= \frac{1}{\Delta x} \int_{x_1}^{x_2} m_y + m_{xy} K_x^{(d)}(x,y)\,\mathrm{d}x\,, \end{aligned} \tag{4.19}$$

$$\begin{aligned} m^{n*}_{yy,i}\big|_{\text{part}} &= \frac{5}{\Delta x \Delta y} \int_0^{\Delta y} \int_{x_1}^{x_2} \rho_d(x,y,t^n) K_{yy}^{(i)}(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{5}{\Delta x \Delta y} \int_{x_1}^{x_2} \int_0^{\Delta y} m^n_{yy,d} K_{yy}^{(d)}(x,y) K_{yy}^{(i)}(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}y\,\mathrm{d}x \\ &= \frac{1}{\Delta x} \int_{x_1}^{x_2} m^n_{yy,d}\,\mathrm{d}x\,, \end{aligned} \tag{4.20}$$

and

$$\begin{aligned} m^{n*}_{xy,i}\big|_{\text{part}} &= \frac{9}{\Delta x\Delta y}\int_0^{\Delta y}\int_{x_1}^{x_2}\rho_d(x,y,t^n)K^{(i)}_{xy}(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{9}{\Delta x\Delta y}\int_{x_1}^{x_2}\int_0^{\Delta y} m^n_{y,d}K^{(d)}_y(x,y)K^{(i)}_{xy}(\varphi(x,t^n,\Delta t/2),y) \\ &\quad + m^n_{xy,d}K^{(d)}_{xy}(x,y)K^{(i)}_{xy}(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}y\,\mathrm{d}x \\ &= \frac{9}{\Delta x\Delta y}\int_{x_1}^{x_2} m^n_{y,d}\frac{\Delta y}{3}K^{(i)}_x(\varphi(x,t^n,\Delta t/2),y) \\ &\quad + m^n_{xy,d}\frac{\Delta y}{3}K^{(d)}_x(x,y)K^{(i)}_x(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}x \\ &= \frac{3}{\Delta x}\int_{x_1}^{x_2} m^n_{y,d}K^{(i)}_x(\varphi(x,t^n,\Delta t/2),y) \\ &\quad + m^n_{xy,d}K^{(d)}_x(x,y)K^{(i)}_x(\varphi(x,t^n,\Delta t/2),y)\,\mathrm{d}x\,. \end{aligned} \tag{4.21}$$

Similarly, the projections of the fractional step $L^y_{\Delta t}$ are determined. We abbreviate $y_L = \varphi(y_{i-1/2}, t^{n**}, -\Delta t)$ and $x_R = \varphi(y_{i+1/2}, t^{n**}, -\Delta t)$. For the full time step Δt in y-direction, we have

$$\begin{aligned} &\mathbb{P}\rho(x,y,t^{n**}) \\ &= \sum_{j=0}^{6}\frac{\int_{y_L}^{y_R}\int_0^{\Delta x}\rho_d(x,y,t^{n*})K^{(i)}_j(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}x\,\mathrm{d}y}{\int_0^{\Delta y}\int_0^{\Delta x}K^{(i)}_j(x)K^{(i)}_j(x)\,\mathrm{d}x\,\mathrm{d}y}K^{(i)}_j(x,y). \end{aligned} \tag{4.22}$$

Then, the respective parts of the coefficients $m^{n**}_{0,i}$, $m^{n**}_{x,i}$, $m^{n**}_{xx,i}$, $m^{n**}_{y,i}$, $m^{n**}_{y,i}$ and $m^{n**}_{xy,i}$ at the intermediate time level t^{n**} are given by,

$$\begin{aligned} m^{n**}_{0,i}\big|_{\text{part}} &= \frac{1}{\Delta x\Delta y}\int_{y_1}^{y_2}\int_0^{\Delta x}\rho_d(x,y,t^{n*})K^{(i)}_0(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{1}{\Delta y}\int_{y_1}^{y_2} m^{n*}_{0,d}K^{(d)}_0 + m^{n*}_{y,d}K^{(d)}_y + m^{n*}_{yy,d}K^{(d)}_{yy}\,\mathrm{d}y\,, \end{aligned} \tag{4.23}$$

$$\begin{aligned} m^{n**}_{x,i}\big|_{\text{part}} &= \frac{3}{\Delta x\Delta y}\int_{y_1}^{y_2}\int_0^{\Delta x}\rho_d(x,y,t^{n*})K^{(i)}_x(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{1}{\Delta y}\int_{y_1}^{y_2}\left(m^{n*}_{x,d} + m^{n*}_{xy,d}K^{(d)}_y\right)\mathrm{d}y\,, \end{aligned} \tag{4.24}$$

$$\begin{aligned} m^{n**}_{xx,i}\big|_{\text{part}} &= \frac{5}{\Delta x\Delta y}\int_{y_1}^{y_2}\int_0^{\Delta x}\rho_d(x,y,t^{n*})K^{(i)}_{xx}(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{1}{\Delta y}\int_{y_1}^{y_2} m^{n*}_{xx,d}\,\mathrm{d}y\,, \end{aligned} \tag{4.25}$$

$$\begin{aligned} m^{n**}_{y,i}\big|_{\text{part}} &= \frac{3}{\Delta x\Delta y}\int_{y_1}^{y_2}\int_0^{\Delta x}\rho_d(x,y,t^{n*})K^{(i)}_y(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{3}{\Delta y}\int_{y_1}^{y_2}\Big(m^{n*}_{0,d}K^{(d)}_0 + m^{n*}_{y,d}K^{(d)}_y + \\ &\quad + m^{n*}_{yy,d}K^{(d)}_{yy}\Big)K^{(i)}_y(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}y\,, \end{aligned} \tag{4.26}$$

$$\begin{aligned} m^{n**}_{yy,i}\big|_{\text{part}} &= \frac{5}{\Delta x\Delta y}\int_{y_1}^{y_2}\int_0^{\Delta x}\rho_d(x,y,t^{n*})K^{(i)}_{yy}(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{5}{\Delta y}\int_{y_1}^{y_2}\Big(m^{n*}_{0,d}K^{(d)}_0 + m^{n*}_{y,d}K^{(d)}_y + \\ &\quad + m^{n*}_{yy,d}K^{(d)}_{yy}\Big)K^{(i)}_{yy,d}(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}y\,, \end{aligned} \tag{4.27}$$

and

$$\begin{aligned} m_{xy,i}^{n**}\Big|_{\text{part}} &= \frac{9}{\Delta x \Delta y}\int_{y_1}^{y_2}\int_0^{\Delta x}\rho_d(x,y,t^{n*})K_{xy}^{(i)}(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{9}{\Delta x \Delta y}\int_{y_1}^{y_2} m_{x,d}^{n*}\frac{\Delta x}{3}K_y^{(i)}(x,\varphi(y,t^{n*},\Delta t)) \\ &\quad + m_{xy,d}^{n*}\frac{\Delta x}{3}K_y^{(d)}(x,y)K_y^{(i)}(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}y \\ &= \frac{3}{\Delta y}\int_{y_1}^{y_2} m_{x,d}^{n*}K_y^{(i)}(x,\varphi(y,t^{n*},\Delta t)) \\ &\quad + m_{xy,d}^{n*}K_y^{(d)}(x,y)K_y^{(i)}(x,\varphi(y,t^{n*},\Delta t))\,\mathrm{d}y\,. \end{aligned} \tag{4.28}$$

To complete a full time step of the 2D SASLDG method, a last projection of the fractional operator $L^x_{\Delta t/2}$ in x-direction is made,

$$\begin{aligned} &\mathbb{P}\rho(x,y,t^{n+1}) \\ &= \sum_{j=1}^{6}\frac{\int_0^{\Delta y}\int_{x_L}^{x_R}\rho(x,y,t^{n**})K_j^{(i)}(\varphi(x,t^{n**},\Delta t/2),y)\,\mathrm{d}x\,\mathrm{d}y}{\int_0^{\Delta y}\int_0^{\Delta x}K_j^{(i)}(x)K_j^{(i)}(x)\,\mathrm{d}x\,\mathrm{d}y}K_j^{(i)}(x,y). \end{aligned} \tag{4.29}$$

The integrals of the parts of the coefficients are equal to (4.16)-(4.21) apart from the change of the density at the new time level $\rho_d(x,y,t^{n**})$ and the trajectories $\varphi(x,t^{n**},\Delta t/2)$. Thus, the formulas for the new coefficients $m_{0,i}^{n+1}$, $m_{x,i}^{n+1}$, $m_{xx,i}^{n+1}$, $m_{y,i}^{n+1}$, $m_{y,i}^{n+1}$ and $m_{xy,i}^{n+1}$ follow analogously to (4.16)-(4.21) and are not listed here.

The projection step of the SASLDG method in 2D discussed above is the only difference to the one-dimensional scheme. The computation of the trajectories and the overall procedure are the same. Because of the differences in the computations of the projections, the computer code developed for the one-dimensional problem cannot be reused. For testing purposes the integrals of the 2D method are solved with the quadrature function of MATLAB.

Below, we conduct two standard test cases for the linear advection in two space dimensions. We carry out the solid body rotation test and a deformational flow test.

4.2.1 Solid body rotation test

The solid body rotation test case is a translational examination of a sine hill. The analytical solution is know at every point in time. We compute the solution for a full rotation. In that way we can compare the numerical solution after the rotation with the initial data. The velocity field of this translational test is given by

$$\begin{aligned} u(x,y) &= y \\ v(x,y) &= -x. \end{aligned} \tag{4.30}$$

Figure 4.1 shows the velocity field corresponding to (4.30). For clarity, the velocity field is plotted on a grid of only 20×20 grid cells.

The initial data consists of a sine hill located in the upper right half of a plane, pictured in Figure 4.2(a). The resolution is chosen to be 51×51.

In this test, it suffices to solve (4.3) only. The density is set equal to one at the beginning and remains constant to one throughout the whole test. The velocity is constant along each one-dimensional problem that is solved. The constant initial distribution equal to one in combination with the constant velocity results in a numerical solution equal to one computed by the SASLDG method. Therefore, the computation of the density is dispensable.

The numerical solution after one rotation is shown in Figure 4.2(b). The computations are carried out with the CFL number 0.5. The absolute value of the difference

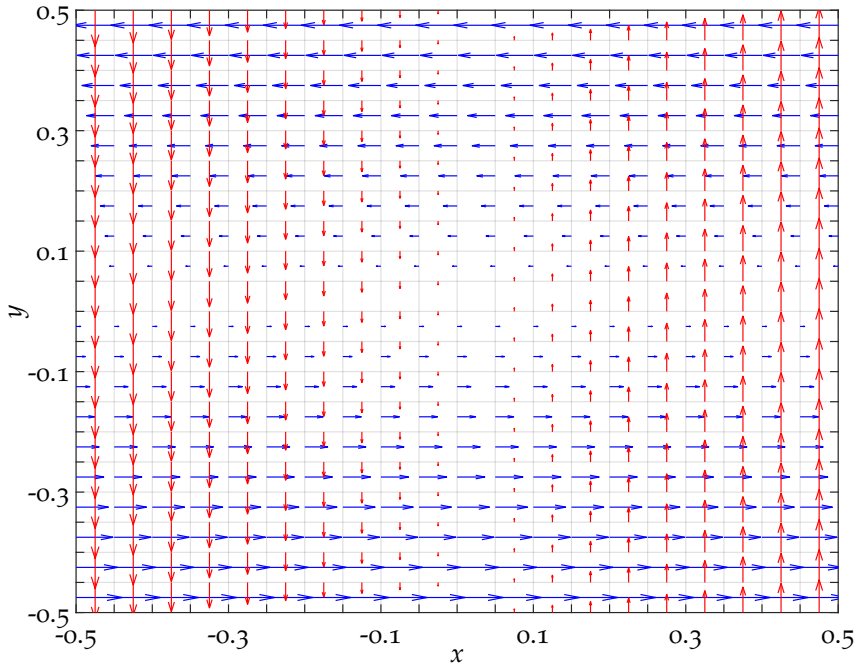


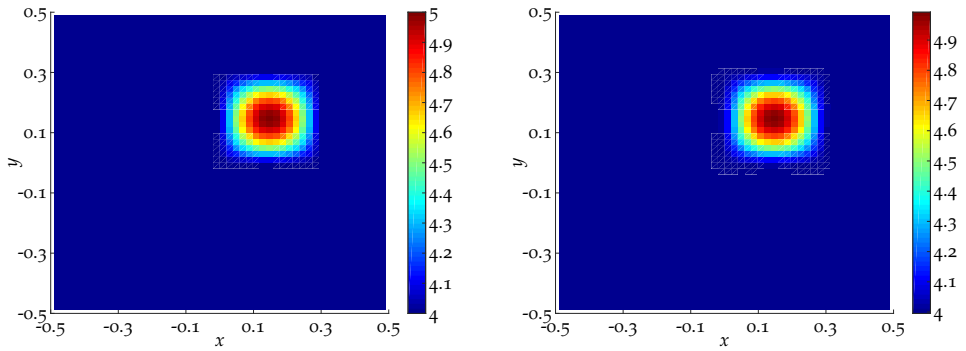
Figure 4.1: The velocity field used for the solid body rotation test.

of the initial values and the numerical solution after one rotation is the error plotted in Figure 4.3.

The cross section of the peak of the sine hill reveals more of the error in detail. Figure 4.4(a) shows the cross section of the initial data (printed in black) and of the numerical solution after one rotation (printed in red). The peak of the initial data is equal to one. Its value decreases to the value 0.992 after one rotation. The steep sides of the hill broaden and some oscillations are visible. The numerical solution takes negative values in the absence of a limiter.

The areas of the largest error are at the edge of the sine hill as depicted in Figure 4.4(b), which shows the absolute value of the difference of the cross sections. Thus, the maximum error of 0.014 is reached not at the peak of the sine hill, but at its edges.

The next test case uses a velocity field that consists of varying velocity in each direction in contrast to the field used in the solid body rotation test case.



(a) Sine hill used as initial values. **(b)** Numerical solution after one revolution.

Figure 4.2: The solid body rotation test for the SASLDG method.

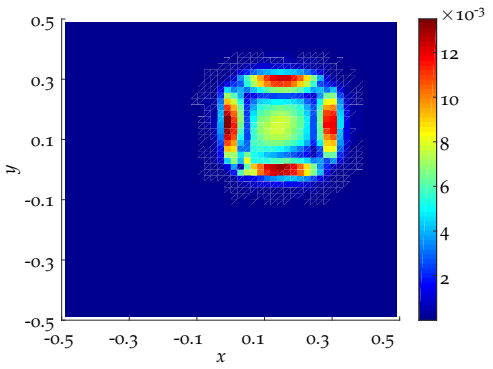
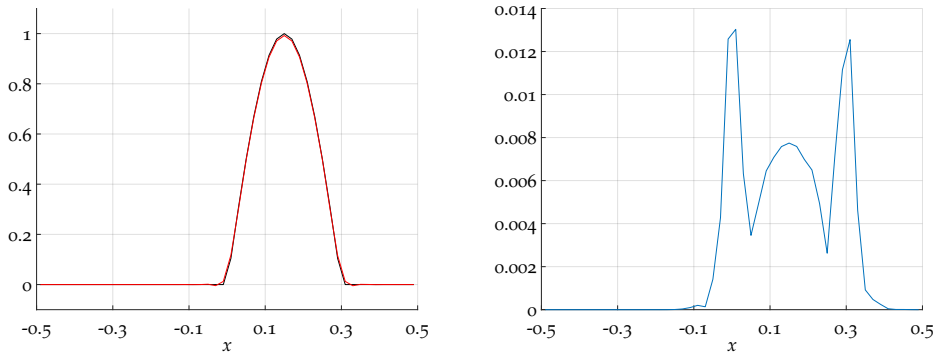


Figure 4.3: The absolute error of the numerical solution after one rotation.



(a) Cross section of the numerical solution: The black line is the IV, the red one shows the solution after one rotation.

(b) Cross sections of the absolute error.

Figure 4.4: Cross sections of the peak of the sine hill.

4.2.2 Deformational flow test

We examine the numerical results of a deformational flow test similarly to the idealized vortex problem described in [15]. The velocity field is given by

$$\begin{aligned} u(x,y) &= -\frac{y}{r}\operatorname{sech}^2(r)\tanh(r) \\ v(x,y) &= \frac{x}{r}\operatorname{sech}^2(r)\tanh(r), \end{aligned} \tag{4.31}$$

where r is the radius

$$r = \sqrt{x^2+y^2}. \tag{4.32}$$

Figure 4.5 shows the velocity field on a grid of 20×20 cells defined by (4.31). The velocity field is non-divergent. Hence, as in the solid body rotation test, it should be unnecessary to compute the density. A constant density distribution equal to one should be preserved in time. However, the density does non remain constant as we will show below.

Using a velocity field as defined in (4.31) a non-constant initial distribution curls to a vortex and long filaments are built. This can be best seen if we use initial data of the form of a staircase function as pictured in Figure 4.6(a). The lowest step takes the value 0.1, each step increases by 0.1. Thus, the highest step of a total of five steps is equal to 0.5. The initial distribution of the density ρ is chosen to be constantly equal to one.

We compute the numerical solution of ρ and ρy for 346 time steps on a grid of 100×100 cells. We use a CFL number equal to 0.5. The time step size is de-

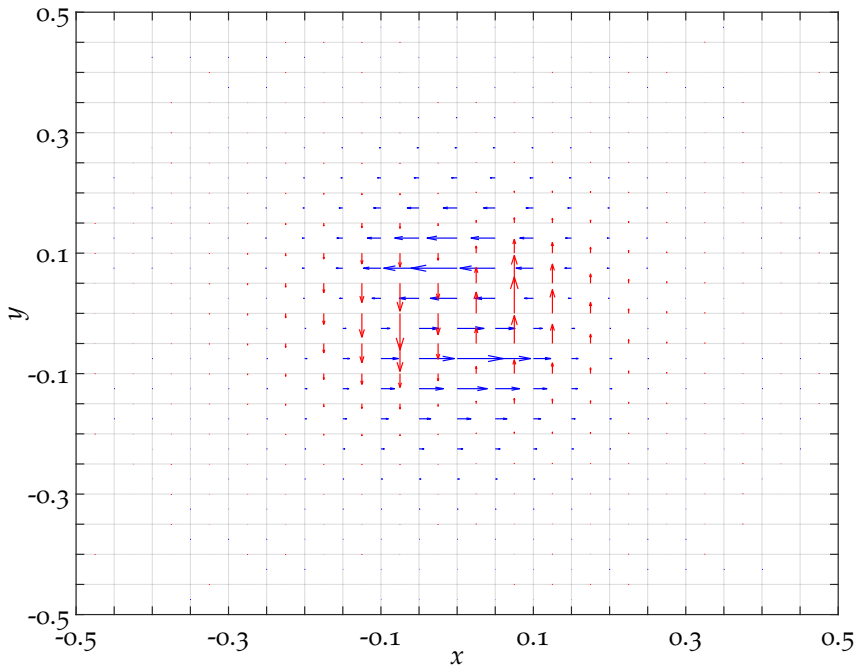


Figure 4.5: The velocity field used for the static vortex test.

termined from the maximum velocity value of all grid cells, i.e. we have $\Delta t = \text{CFL}\,\Delta x / \max(u)$.

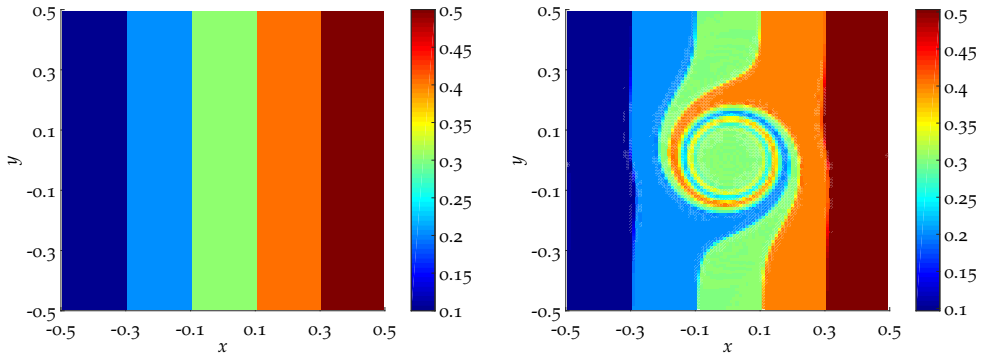
The numerical solution of the tracer y after 347 time steps is shown in Figure 4.6(b). The vortex is visible. The filaments in the middle of the vortex are as thin as one to two grid cell length.

After that, we determine the inverse velocity field and apply it to the vortex shown in Figure 4.6(b) as initial data. Again, we compute the numerical solution for 346 time steps. If the numerical scheme was exact, we would obtain the original staircase function as shown in Figure 4.6(a). The result of the numerical solution of this turned back vortex is pictured in Figure 4.6(c).

The absolute value of the difference of the numerical solution and the initial staircase function is plotted in Figure 4.6(d). The largest error of approximately 0.079 occurs at the boundaries of the step in the middle of the staircase at $x = -0.1$ and $x = 0.1$. This is the area where the largest deformation took place during the test. Smaller errors arise at each boundary of a step.

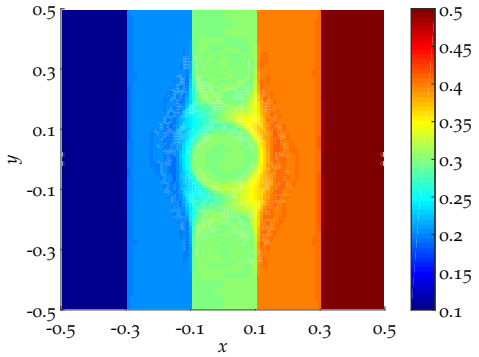
As mentioned above, a constant density distribution should be preserved in time, Figure 4.7 shows that this is not the case in this test. Figure 4.7(a) displays the density distribution after 346 time steps. The maximum deviation from the initial distribution is about 0.0023.

After reversing the velocity, we could expect that the deformed density is advected back to the constant function to one. However, Figure 4.7(b) pictures the density after 346 time steps with the inverse velocity and shows that the error increases. The maximum deviation takes the value 0.0045.
Thus, for constant velocity as in the solid body rotation test case, the constant density is preserved in contrast to the deformational test case, where errors are introduced.

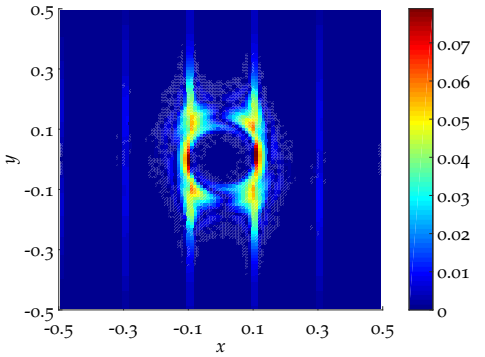


(a) Staircase function used as IV.

(b) Numerical solution after 347 time steps.

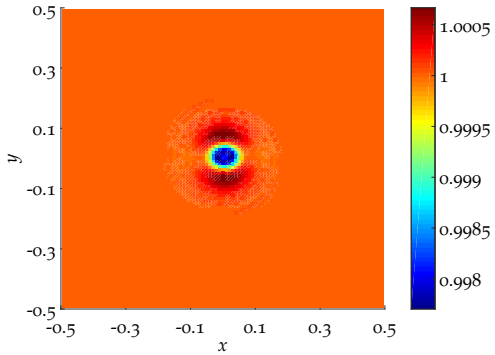


(c) Numerical solution at time $T_{\max}$.

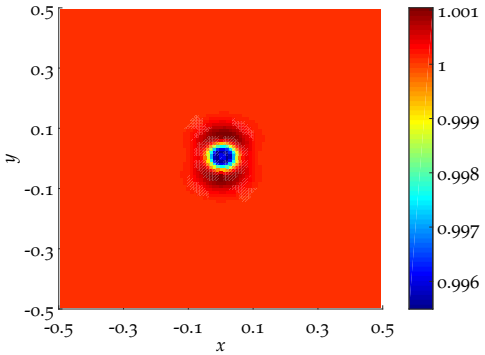


(d) Absolute error of solution at time $T_{\max}$.

Figure 4.6: The static vortex test for the SASLDG method.



(a) The density after 347 time steps.



(b) The density at time $T_{\max}$.

Figure 4.7: Density ρ of the static vortex test.

4.3 HYBRID OPERATOR SPLITTING: MPDATA AND THE SASLDG METHOD

Geophysical processes, in particular meteorological phenomena, occur at different scales in horizontal and vertical direction. For example, the horizontal scale of planetary waves of $\mathcal{O}(1000) - \mathcal{O}(10000)$ km is much larger compared to the thickness of the troposhere of $\mathcal{O}(10)$ km. Processes on the synoptic scale are of the order $\mathcal{O}(100)$ km to $\mathcal{O}(1000)$ km, which is again larger than the depth of the troposhere. Thus, when simulating these processes, the vertical direction must be resolved with a fine grid to grasp the phenomena on this scale. To use the same resolution for the horizontal grid would result in too much computational cost. Therefore, a coarser

grid is used for horizontal and a finer grid for the vertical scale. This causes a large grid aspect ratio of $\mathcal{O}(1) : \mathcal{O}(100)$. For more details, see [41].

If we choose to solve the linear advection equation in two space dimensions via operator splitting, the time step size is the same for horizontal and vertical direction. If the Courant number of the numerical method is limited for stability reasons, e.g. by one as for MPDATA, the size of the time step is determined by the smallest grid cell of both directions (for relatively uniform velocity in both directions).

We assume the following setting: Horizontally, i.e. in x-direction the grid cell sizes are large. Oppositely, in the vertical y-direction the length of the grid cells are small as shown in Figure 4.8.

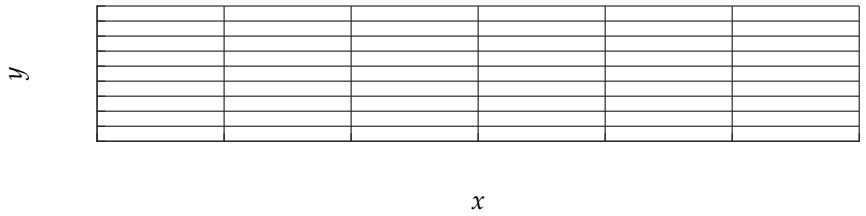


Figure 4.8: Example of a 2D grid with flat grid cells in vertical direction.

In this example, the size of a grid cell in y-direction determines the size of the time step. This means that the time step restriction is adequate for stepping in the y-direction, however for the x-direction Δt could be chosen larger without violating stability properties.

In the following we suggest a possible remedy for that problem. Further, we test the resulting method in particular with respect to different CFL numbers and grid aspect ratios.

4.3.1 Description of the procedure

An idea to circumvent the strict time step restriction for operator splitting in one space dimension for grids with high aspect ratio is to use different numerical methods for stepping into the two directions, which we call "hybrid operator splitting" or hybrid method. For solving the advection equation in the direction with a large scale, a numerical method with low computational cost, but with time step restriction, can be used. A more expensive and accurate method without time step restriction can be applied for the direction of smaller scale. We use the robust, second-order accurate method MPDATA (introduced in 2.6) for stepping in the x-direction of large grid cells. The part of the method with higher accuracy and without any limitations in regard to the time step is taken by a modified version of the SASLDG method.

For the realization we have to consider that the SASLDG method and MPDATA need different input data. MPDATA operates using cell averages whereas the SASLDG method besides the cell averages also uses information on higher coefficients as slope and curvature to express the polynomials. To get this information for the SASLDG method in an efficient manner, we give up a part of the resolution in y-direction to gain the information for the polynomials. To do so, we combine three grid cells into one and compute the missing information for the resulting polynomial in that combined grid cell from the cell averages. The modified version of the SASLDG method is called SASLDG-3c (three cells are combined to one).

The advection step of SASLDG-3c in vertical direction takes the following form.

(i) In each vertical column, three grid cells are combined into one new grid cell. From the three cell averages a quadratic polynomial is reconstructed. These reconstructions serve as initial values for the time step of the SASLDG method.

(ii) The SASLDG method computes the solution of the advection step on a three times coarser grid.

(iii) We determine the cell averages of the three original grid cells exactly and pass on the data for the horizontal advection step.

Step (i) and (iii) must be described. The SASLDG method used in step (ii) is discussed in Chapter 3. Thus, specifying the first step we must consider the following substeps. First, the coarse grid is defined, then the coefficients for the polynomials are computed from the cell averages and the given velocity must be adapted to the coarse grid. Last, the cell averages are computed as part of the third step.

The definition of the coarse grid is straight forward. Three grid cells, e.g. the cells $i-1$, i and $i+1$, are merged into one cell. The length of the new grid cell Δx is the sum of the three old cells, i.e. $\Delta x = \Delta x_{i-1} + \Delta x_i + \Delta x_{i+1}$. Note that this algorithm works only for grids with the total number of grid cells divisible by three in vertical direction.

The polynomial $f(x) = m_2 k_2(x) + m_1 k_1(x) + m_0 k_0(x)$ is expressed with the Legendre polynomials $k_j(x)$ as basis functions given in (3.132) to (3.134). The coefficients m_0, m_1 and m_2 are computed fulfilling the property

$$\int_{x_{k-1/2}}^{x_{k+1/2}} f(x)\,\mathrm{d}x = \Delta x_k Q_k, \tag{4.33}$$

for $k = i-1, i, i+1$, where Q_k is the cell average for the kth grid cell. That results in the coefficients given by

$$m_0 = \frac{Q_{i-1}\Delta x_{i-1} + Q_{i+1}\Delta x_{i+1} + Q_i \Delta x_i}{\Delta x_{i-1} + \Delta x_{i+1} + \Delta x_i}, \tag{4.34}$$

$$\begin{aligned} m_1 = \Big(& Q_{i-1}\Delta x_{i+1}^2 - Q_{i+1}\Delta x_{i-1}^2 - Q_{i-1}\Delta x_i^2 + Q_{i+1}\Delta x_i^2 \\ & + \Delta x_{i-1}^2 Q_i - \Delta x_{i+1}^2 Q_i + 3\Delta x_{i-1} Q_i \Delta x_i \\ & - 3\Delta x_{i+1} Q_i \Delta x_i - 3Q_{i-1}\Delta x_{i-1}\Delta x_i \\ & + 3Q_{i+1}\Delta x_{i+1}\Delta x_i - 3Q_{i-1}\Delta x_{i-1}\Delta x_{i+1} \\ & + 3Q_{i+1}\Delta x_{i-1}\Delta x_{i+1}\Big) \Big/ \Big(2(\Delta x_{i-1} + \Delta x_i)(\Delta x_{i+1} + \Delta x_i)\Big), \end{aligned} \tag{4.35}$$

$$\begin{aligned} m_2 = & ((\Delta x_{i-1} + \Delta x_{i+1} + \Delta x_i)(Q_{i-1}\Delta x_i + Q_{i+1}\Delta x_i \\ & - \Delta x_{i-1} Q_i - \Delta x_{i+1} Q_i + Q_{i-1}\Delta x_{i+1} + Q_{i+1}\Delta x_{i-1} \\ & - 2Q_i \Delta x_i))/(2(\Delta x_{i-1} + \Delta x_i)(\Delta x_{i+1} + \Delta x_i)). \end{aligned} \tag{4.36}$$

Before we can compute the advective step in vertical direction in step (ii), we have to consider the velocity field and adapt it to the coarser grid. The velocity is given at any grid point of the fine grid. To adjust it to the coarse grid, every third point is taken. These points coincide with the cell boundaries of the newly created coarse grid.

Then, the advective step can be computed with SASLDG, where the polynomials are used as initial distribution and the coarse velocity field is given. After that, step (iii) concludes the advection in vertical direction. We determine the cell averages for the fine grid from the polynomials of the coarse grid. The formula computing the averages Q_k reverses (4.33) and yields

$$Q_k = \frac{1}{\Delta x_k} \int_{x_{k-1/2}}^{x_{k+1/2}} f(x)\,\mathrm{d}x\,, \tag{4.37}$$

for $k = i-1, i, i+1$. The function f is the new polynomial distribution after the advective step in vertical direction. The cell averages of the three cells computed from the quadratic polynomial on one coarse grid cell are given by

$$\begin{aligned} Q_{i-1} =& m_0 - m_1 + m_2 + \frac{2m_2\Delta x_{i-1}^2}{\Delta x^2} + \frac{m_1\Delta x_{i-1}}{\Delta x} \\ & - \frac{3m_2\Delta x_{i-1}}{\Delta x}, \end{aligned} \tag{4.38}$$

$$\begin{aligned} Q_i =& m_0 - m_1 + m_2 + \frac{6m_2\Delta x_{i-1}^2}{\Delta x^2} + \frac{m_1\Delta x_i}{\Delta x} - \frac{3m_2\Delta x_i}{\Delta x} \\ & + \frac{2m_1\Delta x_{i-1}}{\Delta x} - \frac{6m_2\Delta x_{i-1}}{\Delta x} + \frac{2m_2\Delta x_i^2}{\Delta x^2} + \frac{6m_2\Delta x_{i-1}\Delta x_i}{\Delta x^2}, \end{aligned} \tag{4.39}$$

$$\begin{aligned} Q_{i+1} =& m_0 - m_1 + m_2 + \frac{6m_2\Delta x_{i-1}^2}{\Delta x^2} + \frac{2m_2\Delta x_{i+1}^2}{\Delta x^2} + \frac{2m_1\Delta x_i}{\Delta x} \\ & - \frac{6m_2\Delta x_i}{\Delta x} + \frac{2m_1\Delta x_{i-1}}{\Delta x} + \frac{m_1\Delta x_{i+1}}{\Delta x} - \frac{6m_2\Delta x_{i-1}}{\Delta x} \\ & - \frac{3m_2\Delta x_{i+1}}{\Delta x} + \frac{6m_2\Delta x_i^2}{\Delta x^2} + \frac{12m_2\Delta x_{i-1}\Delta x_i}{\Delta x^2} \\ & + \frac{6m_2\Delta x_{i+1}\Delta x_i}{\Delta x^2} + \frac{6m_2\Delta x_{i-1}\Delta x_{i+1}}{\Delta x^2}. \end{aligned} \tag{4.40}$$

The old fine grid is stored. The cell averages for all grid cells of the fine grid are available after the vertical advection step. The horizontal advection step follows next.

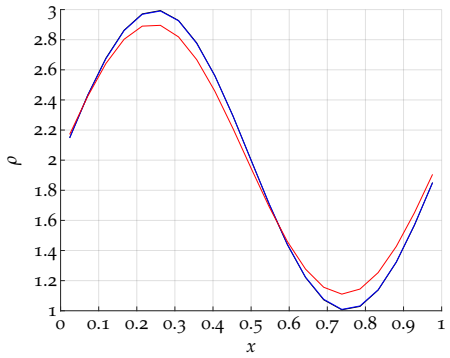
4.3.2 One-dimensional SASLDG-3c vs. MPDATA

The first numerical test is limited to one space dimension. In this way the performance of SASLDG-3c can be studied without the effect of the operator splitting, which influences the results of all two dimensional tests. We compare the results of SASLDG-3c and MPDATA because these two methods will be used for the hybrid operator splitting.

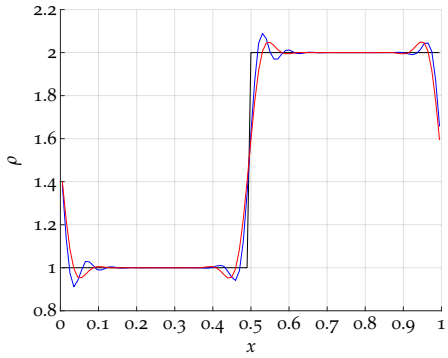
We conduct two of the test cases, that we have shown in Section 3.7. Both use constant velocity field defined in (3.271) with the Courant number 0.45 and 0.5, respectively. The first test case uses the smooth initial distribution of the sine function, given in (3.273), added by two in order to obtain positive initial values needed by MPDATA. We compute ten revolutions of the solution on a grid of 21 grid cells. The second test examines the numerical methods with respect to a discontinuity in the initial data, defined in (3.274), once again added by one for positivity. The numerical solution is determined after one revolution on 99 grid cells. The results are shown in Figure 4.9.

Note that the CFL numbers 0.45 and 0.5 for the SASLDG-c3 method refers to the fine grid. The actual advection step is conducted on a three times coarser grid of 7 grid cells for the first test and on a grid of 33 grid cells for the latter test. Thus, the CFL numbers 0.15 and 0.167, respectively, are the actual numbers used for the computations of SASLDG-3c.

The numerical solution of MPDATA, printed in red, and of SASLDG-3c, printed in blue, of the first test case is shown in Figure 4.9(a) along with the analytical solution. The maximum of the analytical solution takes the value 2.9935, the maximum of SASLDG-3c drops to 2.9904 and of MPDATA to 2.8954. The solution of SASLDG-3c barely alters the sinusoidal form, compared to MPDATA, which distorts the solution after some time. The deformation is particularly noticeable at the point $x = 0.5$, where the analytical function takes the value two. The maximum error and the l_1-error of MPDATA are given by 0.1160 and 0.0723, respectively. The corresponding errors of the SASLDG-3c method are 0.0037 and 0.0020, respectively. We recall of the performance of the SASLDG method in Section 3.7. We tested on a



(a) Solution with smooth IV after ten revolutions on 21 cells with CFL number 0.45.



(b) Solution with discontinuous IV after one revolution on 150 cells with CFL number 0.5.

Figure 4.9: Results of MPDATA (red) and SASLDG-3c (blue) are shown computed with constant velocity. The analytical solution (black) is plotted for comparison.

grid of ten cells and examined the solution after 100 revolutions, see Figure 3.15(b). The comparable errors made in the computation on 21 cells and after only 10 cycles are $6.48 \cdot 10^{-6}$ (the maximum error) and $4.13 \cdot 10^{-6}$ (the l_1-error). These values are obtained by initial values that are projected onto polynomial functions. If the test is conducted with piecewise constant initial values, as for MPDATA and the SASLDG-3c method, the maximum error increases to 0.0074 and the l_1-error to 0.0047. All errors are determined from the absolute difference of the numerical solution and the respective initial values.

The results of the second test case with discontinuous initial values are shown in Figure 4.9(b). The most distinctive errors of this test are the over- and undershoots in the numerical solutions of both methods, since the option of limiting is disabled. The value of the largest undershoot in the solution of MPDATA is 0.9524, the overshoot takes the value 2.0492. The largest under- and overshoot of the SASLDG-3c method is of 0.9114 and 2.0889, respectively, and thus both of larger amplitude. However, the maximal absolute error of MPDATA of value 0.4062 is larger than the largest absolute deviation of SASLDG-3c from the analytical solution, which is 0.3868. The SASLDG-3c method succeds in maintaining a steeper jump discontinuity, which cause the maximal absolute error. For the same reason, the l_1- error of SASLDG-3c of value 0.0288 is smaller than the l_1-error of MPDATA, that is 0.0351. The corresponding test case on 100 grid cells of the SASLDG method in Section 3.7 is shown in Figure 3.16(a). The maximum error computed for comparison on a grid of 99 grid cells and with the upwards shifted initial distribution is about 0.1552. The l_1-error takes the value 0.0084. Both errors are remarkably smaller than than the errors of MPDATA and the SASLDG-3c method. Additionally, note that the area of the oscillations in Figure 3.16(a) by SASLDG is limited to a much more narrow region than the affected area of the oscillations in the solution by MPDATA and SASLDG-3c. The maximal under- and overshoot also are of smaller amplitude and are given by the values 0.9612 and 2.0388, respectively.

In summary, the smallest errors are caused by the SASLDG method, followed by its modified version, i.e. SASLDG-3c. The largest errors (absolute and l_1-error) arise using MPDATA in both test cases, although the over- and undershoot of the solution of MPDATA are of smaller amplitude than of the SASLDG-3c method.

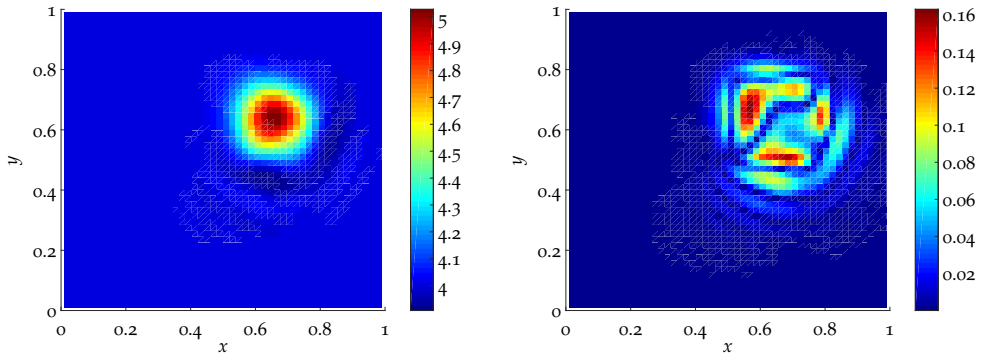
4.3.3 Solid body rotation test

The solid body rotation test is conducted for the two dimensional SASLDG method in Section 4.2.1. We repeat the same 2D test case using operator splitting for MP-

DATA , SASLDG-3c and the hybrid method. Thus we can examine the solution of each individual method and compare the result with the hybrid version. The same velocity field is employed, defined in (4.30) and shown in Figure 4.1. We compute the solution for one rotation. The CFL number of 0.5 used for this test case holds for both space dimensions, since the grid cells are of the same size and number in horizontal and vertical direction.

MPDATA

We compute the numerical solution with MPDATA via operator splitting in both space directions. The result of the computation is shown in Figure 4.10(a) after one rotation. The peak of the sine hill increases compared to the initial hill and takes the value 5.028. Because of oscillations, the minimum value drops to 3.907.



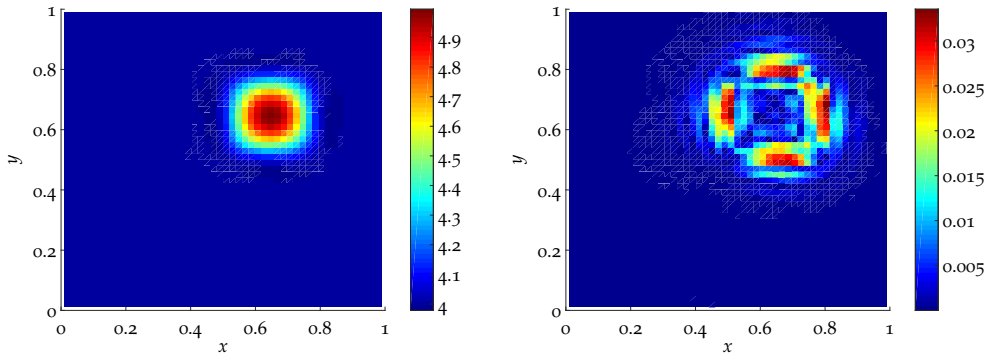
(a) Numerical solution after one rotation. **(b)** Absolute error of numerical solution.

Figure 4.10: The solid body rotation test for MPDATA.

Figure 4.10(b) shows the absolute values of the difference of the numerical solution after one rotation and the initial values. The maximum error is approximately 0.163. The error is not symmetrical, which indicates that the position after one rotation of the sine hill deviates from the initial position.

SASLDG-3c

Figure 4.11(a) shows the numerical solution using SASLDG-3c in both space directions. It pictures the solution after one rotation. The maximum of the numerical solution, located in the sine hill, reaches the value 4.995. The minimum of the numerical solution of the whole computational domain is 3.978.



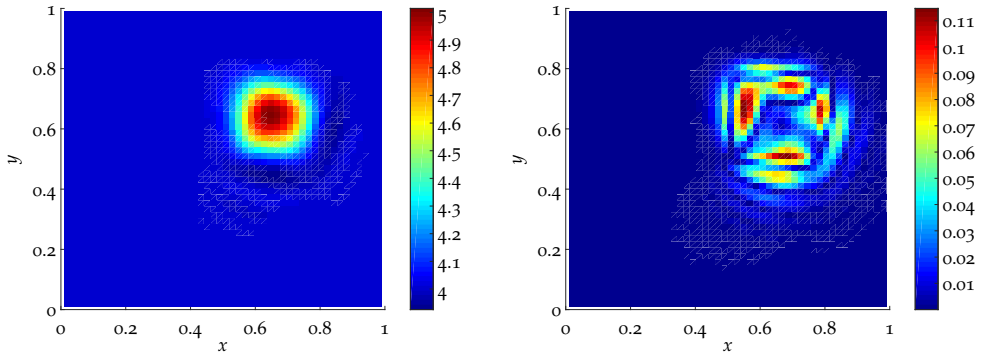
(a) Numerical solution after one rotation. **(b)** Absolute error of numerical solution.

Figure 4.11: The solid body rotation test for the SASLDG-3c method.

The absolute value of the difference of the solution after one rotation and the initial distribution is plotted in Figure 4.11(b). The maximum error takes the value 0.034. Note the maximum error of SASLDG (without the combination of three grid cells to one) is approximately 0.014. The error is quite symmetrical, which means that the location after the rotation matches the position of the initial values. The flattening of the sides of the sine hill is the main source of the error.

Hybrid method

The last test case of the solid body rotation test combines MPDATA and SASLDG-3c in the hybrid version of the operator splitting. The horizontal advection is computed by MPDATA, the vertical advection by SASLDG-3c. The numerical solution is shown after one rotation in Figure 4.12(a). The maximum increases to the value 5.015, the minimum is reached at the value 3.926.



(a) Numerical solution after one rotation.

(b) Absolute error of numerical solution.

Figure 4.12: The solid body rotation test for the hybrid method.

The absolute error of the hybrid method is shown in Figure 4.12(b). Its maximum reaches the value of 0.115, which is between the error of using MPDATA only and SASLDG-3c only. Also, the structure of the error is visibly a combination of both schemes applied purely.

A summary of characteristic values of the numerical solutions is given in Table 4.1. It lists the respective minimum and maximum of the solution after one rotation. The l_∞-error and the l_1-error are displayed. Furthermore, the computational time is also given in the table.

	min	max	l_∞-error	l_1-error	time (s)
SASLDG	3.995	4.992	0.014	0.0008	8344.76
MPDATA	3.907	5.028	0.163	0.0103	45.3
SASLDG-3c	3.978	4.995	0.034	0.0021	147.2
hybrid	3.926	5.015	0.115	0.0067	81.25

Table 4.1: Characteristic values for the cone revolution test computed on a 51×51 grid with CFL number 0.5.

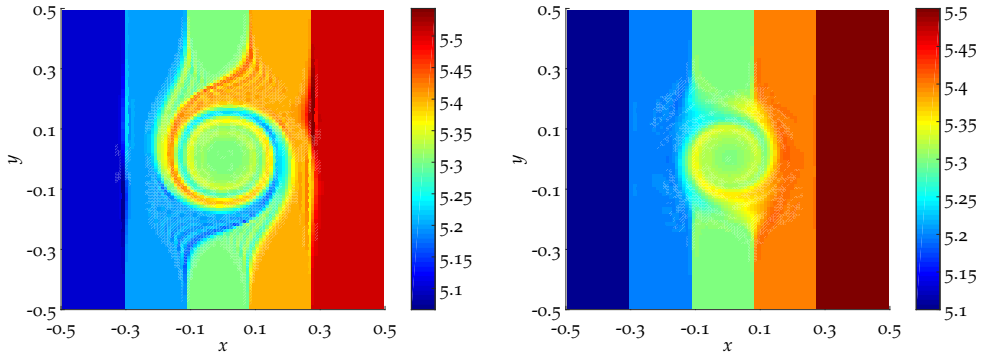
4.3.4 Deformational flow test: static vortex

We continue the deformational flow test, where we have shown the results for the SASLDG method in Section 4.2.2. We proceed with testing MPDATA, SASLDG-3c and the hybrid method. The velocity field for the first half of the test is defined as above, see (4.31), and plotted in Figure 4.5. For the second half of the test, the velocity case is inverted. We decrease the number of grid cells from 100 by one

to guarantee a number divisible by three. The Courant number 0.5 holds for both directions, since the size of the grid cells are equal as well as the maximum velocity in both dimensions. The initial staircase function, displayed in Figure 4.6(a), is used as initial distribution with one difference. The function is increased by the value five everywhere. Developing oscillations in the hybrid method caused by the SASLDG-3c scheme could reach negative numbers, which is allowed for this method but not tolerated by MPDATA.

MPDATA

The numerical solution of the 2D advection equation is computed for 346 time steps using MPDATA in both space dimensions. The vortex, that has built up after that time, is shown in Figure 4.13(a). The filaments that were generated are roughly two to three grid cells thick. The inner filament of the vortex is barely distinguishable from the surrounding step. When zooming in on the filaments, we note that at the location, where the filaments start to curl from the step, oscillations arise towards the edge of the filament. Using that vortex as initial values, we again compute the solution for 346 time steps with the reversed velocity field. The result is pictured in Figure 4.13(b). A vortex is still visible, though less curled up than in the first half.



(a) Numerical solution after 347 time steps. **(b)** Numerical solution at time $T_{\max}$.

Figure 4.13: Numerical solution of MPDATA for the static vortex test.

The analytical solution of this back and forth test case is the initial staircase function shown in Figure 4.6(a). Therefore, the error is the difference of the numerical solution pictured in Figure 4.13(b) and the staircase function. The absolute value of the difference is shown in Figure 4.14.

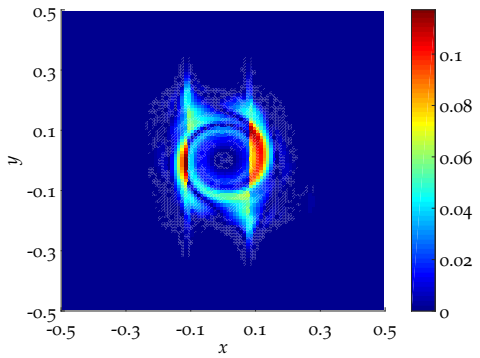
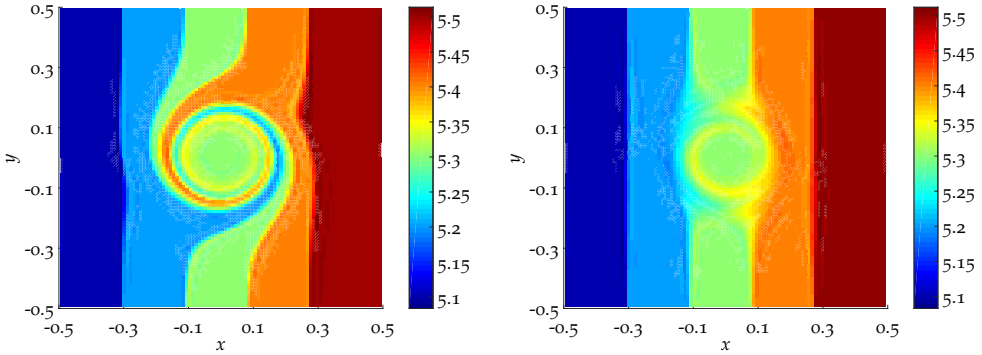


Figure 4.14: Absolute error of the solution of MPDATA for the static vortex test.

The maximal error is approximately 0.117 and occurs at the intersection of the edges of the steps and the vortex. The error shown in the figure is limited to the region, where the vortex originated.

SASLDG-3c

Figure 4.15(a) shows the numerical solution after the first half, that is 346 time steps, computed with SASLDG-3c via operator splitting. Equally to the results computed with MPDATA, the thickness of the filaments is two to three grid cells. Reversing the velocity and continuation of the computations for 346 time steps leads to the numerical solution plotted in Figure 4.15(b). The oscillations that occur near the edge of the filaments in the results from MPDATA above, do not arise using SASLDG-3c.



(a) Numerical solution after 347 time steps.

(b) Numerical solution at time $T_{\max}$.

Figure 4.15: Numerical solution of the SASLDG-3c method for the static vortex test.

The error, i.e. the absolute difference of the solution shown in Figure 4.15(b) and the staircase function, is shown in Figure 4.16.

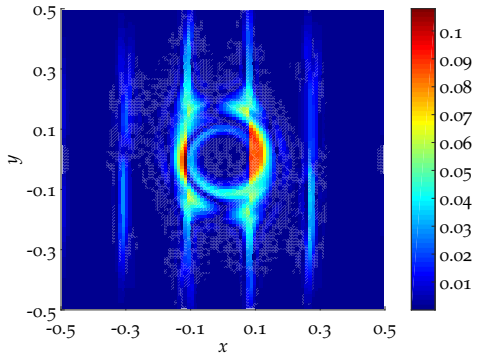


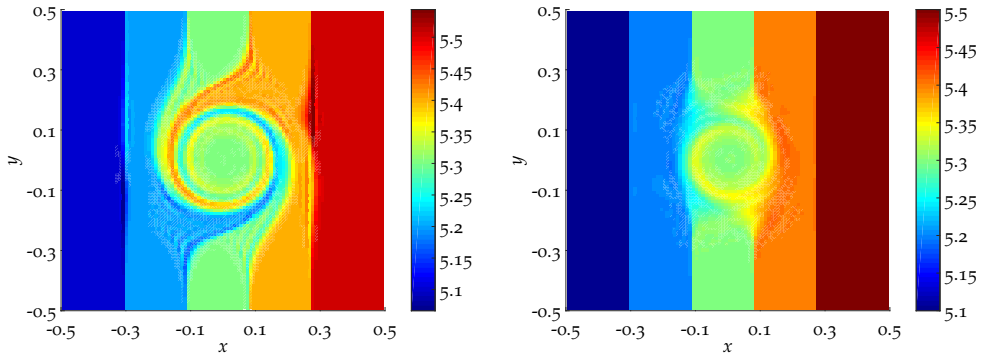
Figure 4.16: Absolute error of the solution of the SASLDG-3c method for the static vortex test.

The maximum of the error takes the value 0.108 at a similar location as the error of the MPDATA computation. Additionally, errors of a smaller order arise at all edges of the steps in the whole computational domain. This can be explained by the combination of three grid cells to one in the algorithm of the SASLDG-3c method.

Hybrid method

After we examined the results for the 2D linear advection equation with pure MPDATA operator splitting and SASLDG-3c only, we study the results of the hybrid version. Again, the horizontal dimension is solved using MPDATA, and the vertical dimension with SASLDG-3c. The numerical solution of the static vortex test after 346 time steps is shown in Figure 4.17(a). In fact, the result of the hybrid method is a mixture of the results from pure MPDATA and pure SASLDG-3c. The oscillations

mentioned above do occur in the hybrid version, but of less amplitude than for using MPDATA only. The numerical solution of the turned back vortex after 346 time steps with the reversed velocity field is pictured in Figure 4.17(b).



(a) Numerical solution after 347 time steps. **(b)** Numerical solution at time $T_{\max}$.

Figure 4.17: Numerical solution of the hybrid method for the static vortex test.

The error of the hybrid method is plotted in Figure 4.18. The error shows evidence for using the hybrid method, i.e. shows similarities to the error of MPDATA and SASLDG-3c.

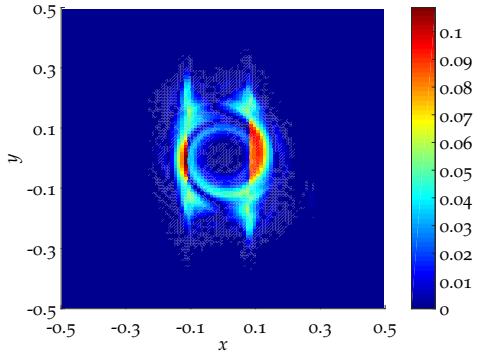


Figure 4.18: Absolute error of the solution of the hybrid method for the static vortex test.

The maximum of the absolute error reaches the value 0.109, which is between the error of using both methods purely. Since MPDATA is applied for the horizontal advection, the error that arises at the edges of the steps from using the SASLDG-3c method visible in Figure 4.16, is not noticeable in Figure 4.18.

Table 4.2 gives an overview of the maximal error, that occurs in the different numerical methods. Additionally, the l_1-error is given.

	l_∞-error	l_1-error
SASLDG	0.079	0.0029
MPDATA	0.117	0.0040
SASLDG-3c	0.108	0.0056
hybrid	0.109	0.0038

Table 4.2: Characteristic values for the static vortex test computed on a 99×99 grid (for SASLDG 100×100) with CFL number 0.5.

4.3.5 Deformational flow test: Rider Kothe

In this test, case we use a deformational flow velocity field proposed by Rider and Kothe in [50]. It is given by

$$\begin{aligned} u(x,y) &= \sin(2\pi(x+d))\sin(2\pi(y/2+d)) \\ v(x,y) &= \cos(2\pi(x+d))\cos(2\pi(y/2+d)), \end{aligned} \tag{4.41}$$

with $d = -0.5$. The velocity field is shown in Figure 4.19.

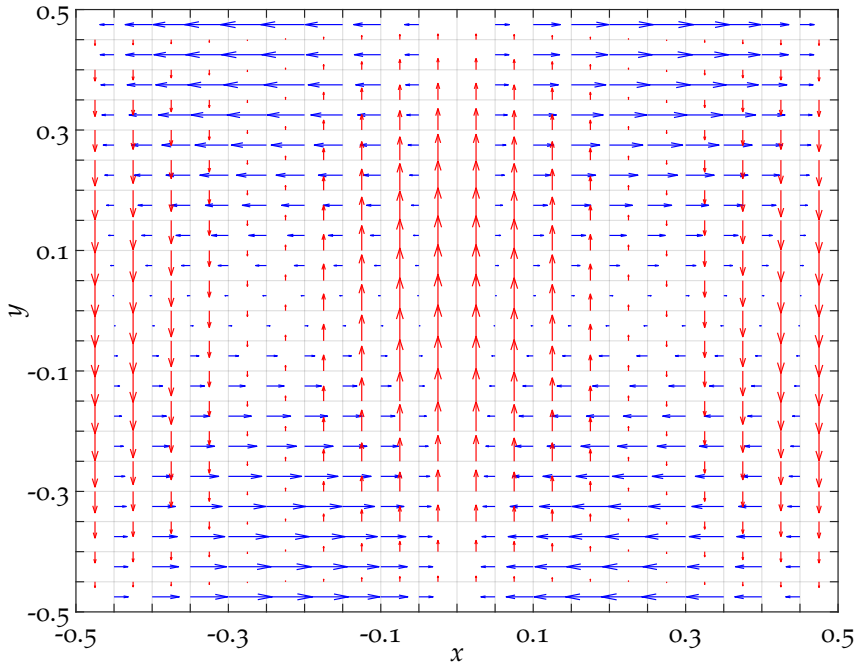


Figure 4.19: The velocity field used for the deformational flow test Rider Kothe.

The velocity field is determined such that it creates two symmetrical vortices that point in opposite directions. We choose smooth initial data, i.e. a gaussian hill, as shown in Figure 4.20.

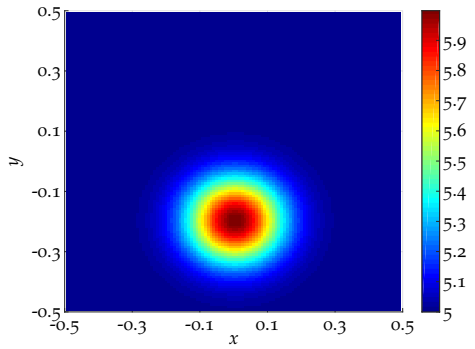
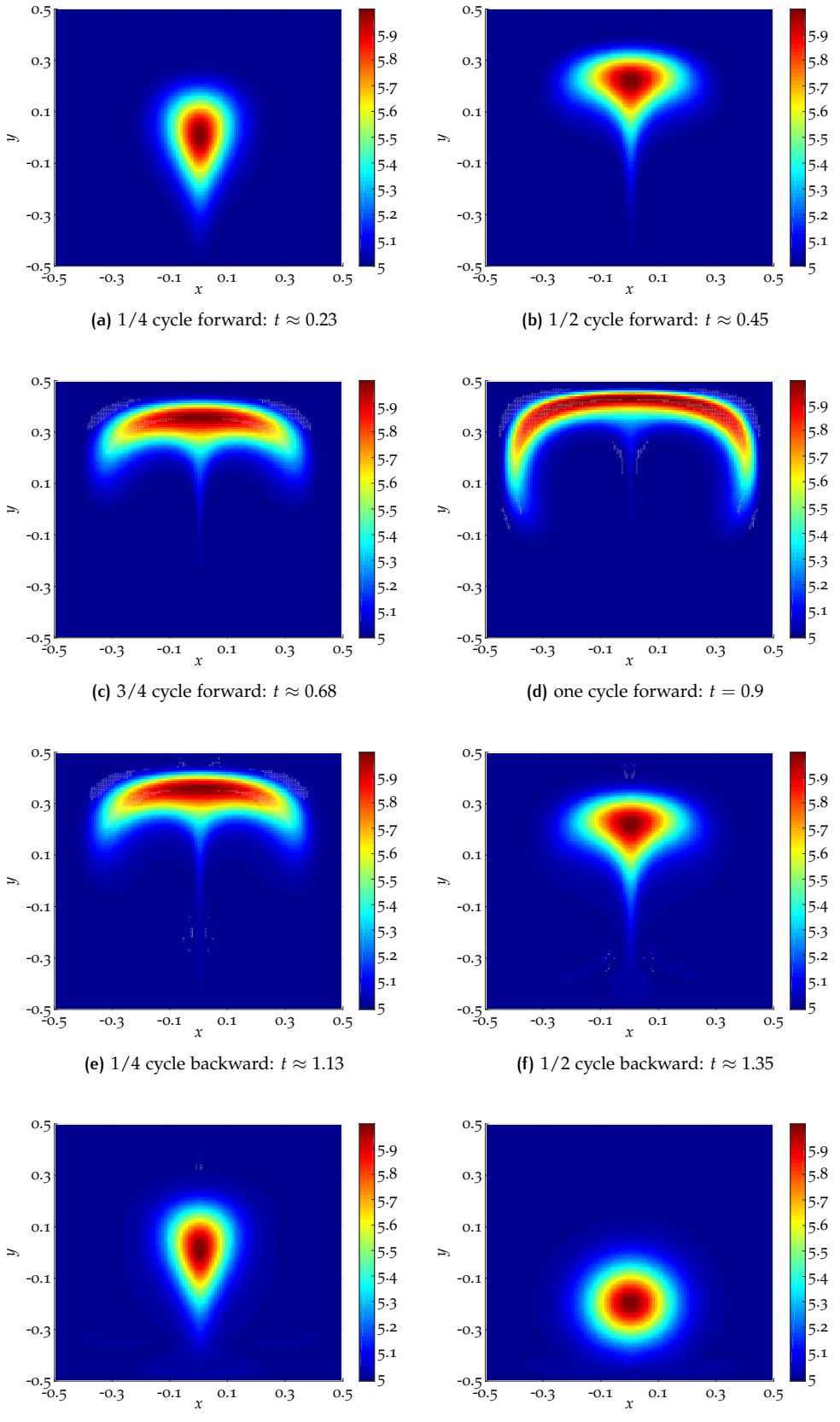


Figure 4.20: The initial values used for the deformational flow test Rider Kothe.

It is defined by

$$f(x,y) = \exp\left(-\frac{1}{2}\left(\left(\frac{x-\mu_x}{\sigma}\right)^2 + \left(\frac{y-\mu_y}{\sigma}\right)^2\right)\right), \tag{4.42}$$



(a) $1/4$ cycle forward: $t \approx 0.23$

(b) $1/2$ cycle forward: $t \approx 0.45$

(c) $3/4$ cycle forward: $t \approx 0.68$

(d) one cycle forward: $t = 0.9$

(e) $1/4$ cycle backward: $t \approx 1.13$

(f) $1/2$ cycle backward: $t \approx 1.35$

(g) $3/4$ cycle backward: $t \approx 1.58$

(h) one cycle backward: $t = 1.8$

Figure 4.21: Evolution of the deformational test using the Rider-Kothe velocity field. The numerical solution is obtained by SASLDG-3c on a 120×120 grid with the Courant number 0.5.

where $\mu_x = 0$, $\mu_y = -0.2$ and $\sigma = 0.1$.

Using the velocity field defined in (4.41) up to time $t = 0.9$, the gaussian hill gets deformed and advected upwards. It then gets drawn apart and takes the form of an arch. If we continued with this velocity field, thin filaments would start to form and curl to vortices. Instead, we reverse the velocity field and start with the numerical data at time $t = 0.9$ and perform the computations for the same time, i.e. we have $T_{\max} = 1.8$. Ideally, the initial data would be attained after that time.

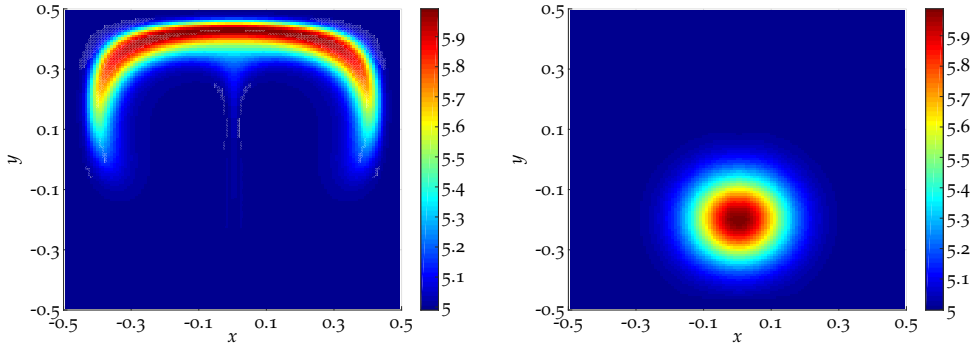
The whole evolution of the test case is illustrated in Figure 4.21 by snapshots at different points in time. The data is obtained from using SASLDG-3c on a grid of 120×120 cells with Courant number 0.5. The panels show the numerical solution after a quarter, a half and three quarters towards to largest deformation, which is pictured as well. Then, with reversed velocity, the solution is plotted after a quarter, a half and three quarters back towards the initial distribution. The last figure shows the numerical solution at time $T_{\max} = 1.8$.

With this test we want to examine how the different numerical methods perform using smooth initial values in a deformational flow test. Furthermore, we conduct the test on two different sets of grids. First, we use a grid of 120×120 cells, i.e. a grid with aspect ratio 1:1. Second, a grid of 120×360 cells and thus with aspect ratio 1:3 is applied. Two different Courant numbers are tested for the hybrid method on the latter grid. Additionally, we measure the elapsed time for the computation of each individual numerical test. This enables us to compare the computational cost for the different methods with varying settings.

For all tests, we plot the numerical solution at the largest deformation and at the end, where the form of the initial values should be attained. The respective subsequent figure shows the absolute error, i.e. the absolute difference of the initial data and the numerical solution.

MPDATA - grid ratio 1:1

The numerical solution is computed with MPDATA on a grid of 120×120 grid cells. The Courant number applied for this test is 0.5 with respect to the vertical grid, written shortened by 0.5 (y). The magnitude of the maximal velocity is similar in horizontal and vertical direction. Since the grid cells are equally sized in both directions, the Courant number holds for the x-direction as well as the y-direction. The numerical solution of the largest deformation at time $t = 0.9$ is shown in Figure 4.22(a). The maximum of 5.993 is reached at the point $[0.09, 0.42]$, slightly shifted to the right from $x = 0$. The minimum drops to 4.990 at the point $[-0.01, -0.01]$, caused by small oscillations. Figure 4.22(b) pictures the numerical solution at the



(a) The numerical solution at time $t = 0.9$.

(b) The numerical solution at time $t = 1.8$.

Figure 4.22: The numerical solution of the Rider Kothe test computed with MPDATA on a 120×120 grid with CFL number 0.5 (y) w.r.t. the vertical direction.

end of the test at time $T_{\max} = 1.8$. The maximum of the gaussian hill is advected

exactly to its initial point, i.e. $[0.0, -0.20]$. It takes the value 5.988. The minimum 4.998 of the numerical solution is obtained at $[0.0, 0.15]$.

The difference of the numerical solution at time $T_{\max} = 1.8$ and the initial values at time $t = 0$ describes the error made in this test. The absolute difference is shown in Figure 4.23.

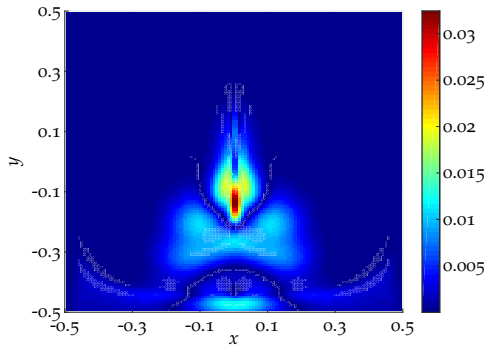


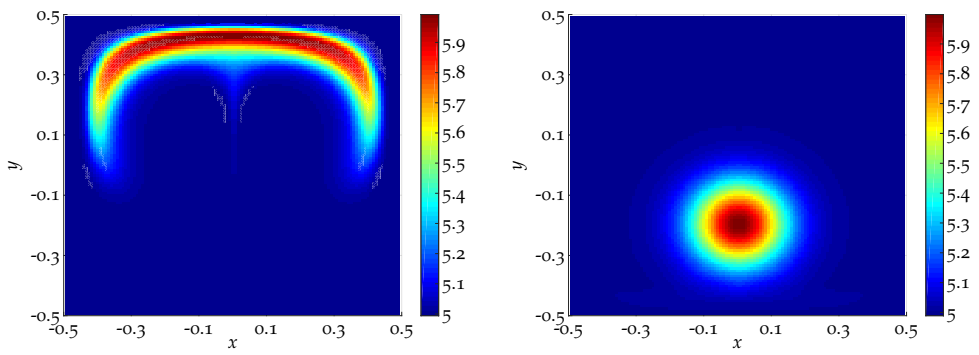
Figure 4.23: Absolute error of the solution of MPDATA shown in Figure 4.22(b).

The maximum of the error is approximately 0.0324. The area of the largest error is a narrow stripe that stretches from where the peak of the gaussian hill is located up to point $[0.0, 0.0]$. The maximum is located at the point $[0.0, -0.14]$. Errors of smaller magnitude of approximately 0.011 are present at the region of the lower half of the gaussian hill. Another error peak of approximately 0.013 is located at $[0.0, -0.48]$, near the boundary of the computational domain. Small oscillations occur in this area around that peak up to the left and right side of the computational domain. The l_1-error is given by 0.0014.

The elapsed time for the computation of the test case with MPDATA on a grid of 120×120 grid cells and CFL number 0.5 (y) is 314.9s.

SASLDG-3c - grid ratio 1:1

We compute the numerical solution on a grid of 120×120 grid cells using the SASLDG-3c method with the Courant number 0.5 (y) with respect to the fine grid. The actual CFL number corresponding to the coarse grid of 40 gird cells is 0.167. The numerical solution at time $t = 0.9$ at the largest deformation is shown in Figure 4.24(a). The maximum of the solution is kept at the middle of the x-axis, i.e. at $[0.0, 0.43]$. It takes the value 5.993. A minimum of 4.994 is located at $[0.02, 0.11]$.



(a) The numerical solution at time $t = 0.9$.

(b) The numerical solution at time $t = 1.8$.

Figure 4.24: The numerical solution of the Rider Kothe test computed with SASLDG-3c on a 120×120 grid with CFL number 0.5 (y) w.r.t. the fine vertical grid.

The numerical solution at time $T_{\max}$ is displayed in Figure 4.24(b). The peak of the gaussian hill is advected to its initial location at $[0.0, -0.2]$ and takes the value 5.999. The minimum of 4.989 is located at $[0.46, -0.34]$. It is part of oscillations of the numerical solution that become visible in the error plot.

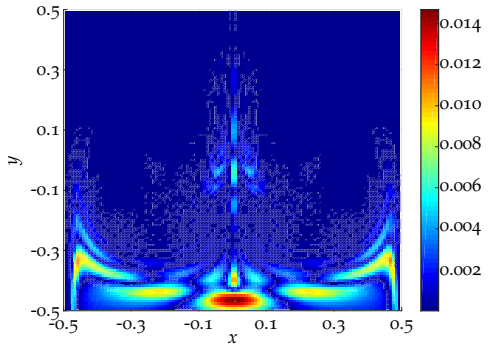


Figure 4.25: Absolute error of the solution of SASLDG-3c shown in Figure 4.24(b).

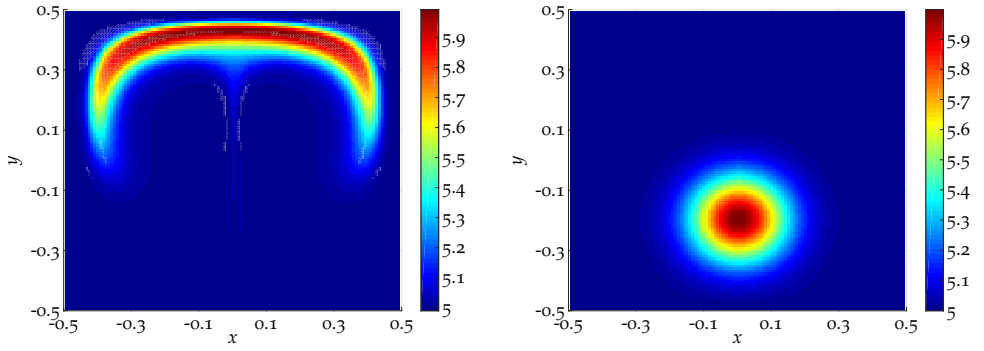
The largest error occurs near the lower boundary of the computational domain. Its peak of 0.0147 is located at the point $[0.0, -0.47]$. At the same spot, an error of smaller magnitude is present for MPDATA. Further errors arise because of the oscillations in the numerical solution. Thus, an error of 0.01 is located at the location of the minimum of the solution. Small errors of 0.006 are present in the area where the largest error of MPDATA occurs. The l_1-error is given by 0.0008.

The elapsed time for the computation of the test case with SASLDG-3c on a grid of 120×120 grid cells and CFL number 0.5 (y) with respect to the fine grid is 556.0s.

Hybrid method - grid ratio 1:1

The next test examines the numerical solution of the hybrid method on a grid of 120×120 grid cells. The size of the time step is determined by the Courant number 0.5 with respect to the fine grid in both space dimensions. Since the advection step of the SASLDG-3c method is done on a grid of 40 grid cells, the actual CFL number decreases to 0.167. The result of the computation after time $t = 0.9$ and after time $t = 1.8$ can be seen in Figure 4.26(a) and Figure 4.26(b), respectively. The minimum of the solution at the end of the forward cycle of value 4.990 is at the same point as the minimum of the solution of MPDATA, i.e. $[-0.01, -0.01]$. The maximum of 5.993 is at the same location as the maximum of the SASLDG-3c method at $[0.0, 0.43]$. The minimum of 4.999 of the numerical solution at time $T_{\max}$ shows resemblance with the minimum of MPDATA. It is located at the point $[-0.01, 0.26]$, which is shifted towards the upper boundary of the domain in comparison to the minimum of MPDATA. As for the methods above, the maximum of value 5.998 is located at the point of the initial peak of the gaussian hill.

The absolute error of the numerical solution at time $t = 1.8$ is shown in the error plot in Figure 4.27. The overall structure of the error of the hybrid method bears resemblance with the error of SASLDG-3c. The largest error of 0.0131 is located at $[0.0, -0.48]$, which is the same point as for SASLDG-3c. Furthermore, smaller error peaks are situated at $x = 0$, between approximately $y = 0$ and the upper boundaries of the domain with diminishing magnitude towards $y = 0.5$. The oscillations that are present for the SASLDG-3c method are still visible for the hybrid method. However, their maximal amplitude of approximately 0.004 is significantly smaller. The maximum error and the l_1-error of 0.0007 of the solution of the hybrid method are smaller than the respective errors of the individual methods.
Thus, the combination of MPDATA and SASLDG-3c in this particular way improves the overall result. Note that if we assemble the hybrid method the other way around, i.e. if we use MPDATA for the vertical and SASLDG-3c for the horizontal advection,



(a) The numerical solution at time $t = 0.9$.

(b) The numerical solution at time $t = 1.8$.

Figure 4.26: The numerical solution of the Rider Kothe test computed with the hybrid method on a 120×120 grid with CFL number 0.5 (y) w.r.t. the vertical direction.

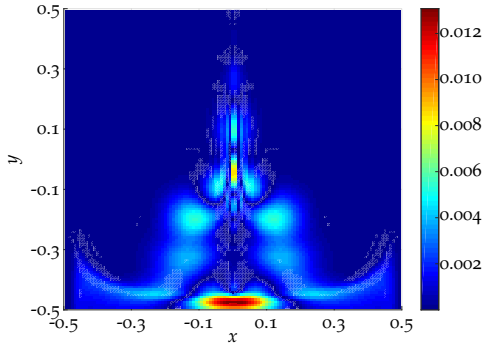


Figure 4.27: Absolute error of the solution of the hybrid method shown in Figure 4.26(b).

the errors of the numerical solution increase. In that case, the l_∞- and the l_1-error are given by 0.0335 and 0.0015, respectively.

The time for the computation of the test with the hybrid method on a grid of 120×120 grid cells and CFL number 0.5 (y) with respect to the fine grid is 406.5s.

Table 4.3 gives an overview of the minimum and maximum values of the respective numerical solutions at time $T_{\max}$, as well as the l_∞-error and the l_1-error. Additionally, the computational time is listed. The CFL number used in all tests refers to the grid in vertical direction, which is denoted by (y) in the table.

	CFL	min	max	l_∞-error	l_1-error	time (s)
MPDATA	0.5 (y)	4.998	5.988	0.0324	0.0014	314.9
SASLDG-3c	0.5 (y)	4.989	5.999	0.0147	0.0008	556.0
hybrid	0.5 (y)	4.999	5.998	0.0131	0.0007	406.5

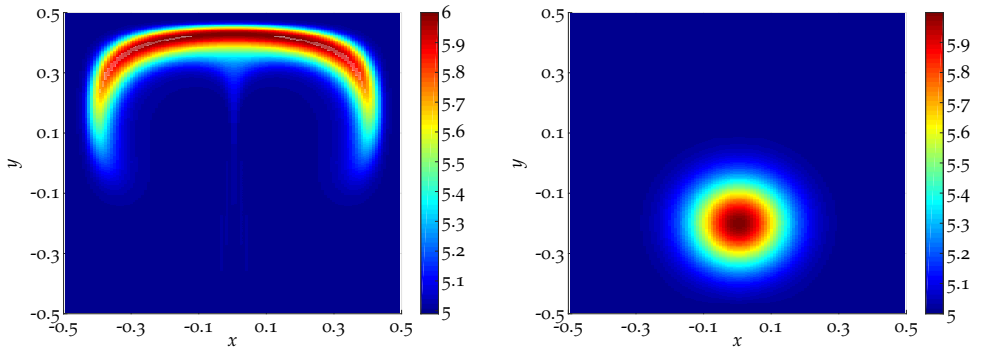
Table 4.3: Characteristic values for the Rider Kothe test computed on a 120×120 grid.

MPDATA - grid ratio 1:3

We change the grid aspect ratio to 1:3 for this and the succeeding tests. Thus, we increase the vertical resolution to 360 grid cells. The horizontal resolution is kept at 120 cells to obtain the desired aspect ratio. We establish the test environment of flat grid cells, as described in the introduction of this section. It is exemplarily illustrated in Figure 4.8.

We use MPDATA to solve the advection equation in both space dimensions and choose to limit the time step such that the CFL number of 0.5 (y) is met in vertical direction. Consequently, the CFL number for the x-direction is approximately 0.167 (x). Considering the horizontal dimension only, we could enlarge the time step by a factor of about six and still meet the stability requirements of MPDATA. In vertical direction, the CFL number could be increased by a factor of two at most.

The numerical solution computed with MPDATA after half the test at the largest deformation at time $t = 0.9$ is plotted in Figure 4.28(a). The minimum and the maximum of the solution are located at approximately the same points as on the 120×120 cells grid. The value 4.990 of the minimum is the same as before. The maximum value of the solution at time $t = 0$ increases to 6.001. The final solution at



(a) The numerical solution at time $t = 0.9$.

(b) The numerical solution at time $t = 1.8$.

Figure 4.28: The numerical solution of the Rider Kothe test computed with MPDATA on a 120×360 grid with CFL number 0.5 (y) w.r.t. the vertical direction.

time $t = 1.8$ is given in Figure 4.28(b). The maximum is reached at the analytically correct point of $[0.0, -0.2]$ and takes the value 5.999. The minimum of 5.0 is the same as the minimum of the initial distribution.

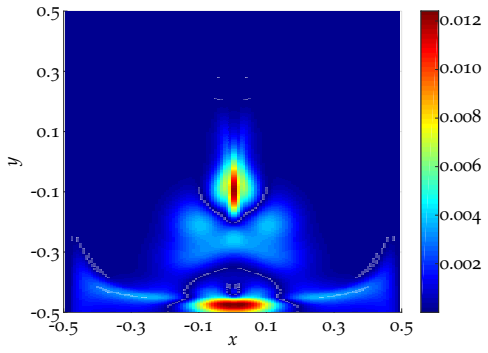


Figure 4.29: Absolute error of the solution of MPDATA shown in Figure 4.28(b).

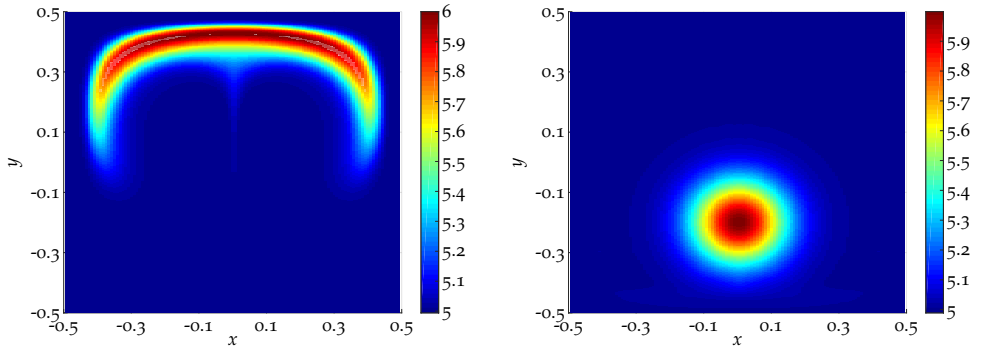
The error plot is given in Figure 4.29. The structure of the error plot resembles the error of MPDATA on the 120×120 grid with one exception. The maximal error, which takes the value 0.0124, is located near the lower boundary of the computational domain at $[-0.01, -0.48]$. A smaller error peak is visible at the same spot for the solution of MPDATA on the 120×120 grid as well. Using the resolution 120×360 leads to an increase of that error. The largest error peak on the previous grid is also present on this grid approximately at point $[0.0, -0.1]$. It takes the value 0.012. The l_1-error is given by 0.0006.

The time for the computation of the test with MPDATA on a grid of 120×360 grid cells and CFL number 0.5 (y) with respect to the vertical direction is 2820.3s.

SASLDG-3c - grid ratio 1:3

In this test we compute the numerical solution with the SASLDG-3c method in both space directions. As for the test case with MPDATA above, the grid has the aspect ratio of 1:3. The CFL number is chosen with respect to the fine grid, i.e. the time step is restricted to meet the CFL number 0.5 (y) in vertical direction. This yields the CFL number of 0.167 (x) for the horizontal direction. Because of combining three grid cells to one in both space directions, the actual CFL numbers related to the coarse grid of 40×120 cells are 0.167 (y) and 0.056 (x), respectively.

Figure 4.30(a) shows the solution at time $t = 0.9$. Figure 4.30(b) plots the results at the end of the backward cycle at time $t = 1.8$. The locations of the minimum and the maximum are at the same points as for the SASLDG-3c method on the previous grid of ratio 1:1. The minimum takes the same value of 4.994. The maximum is given by 6.0. Similarly, the minimum and the maximum of the numerical solution



(a) The numerical solution at time $t = 0.9$.

(b) The numerical solution at time $t = 1.8$.

Figure 4.30: The numerical solution of the Rider Kothe test computed with SASLDG-3c on a 120×360 grid with CFL number 0.5 (y) w.r.t. the vertical direction.

at time $T_{\max}$ are located at the same spots and in fact take the same values as on the grid of 120×120 grid cells. Thus, the minimum of 4.989 is situated at $[0.46, -0.34]$. As before, it is part of oscillations that arise in the solution. The maximum at the point $[0, -0.2]$ takes the value 5.999.

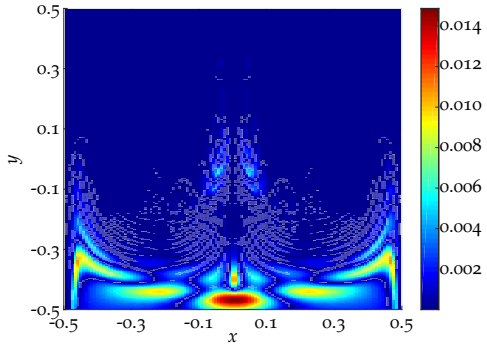


Figure 4.31: Absolute error of the solution of SASLDG-3c shown in Figure 4.30(b).

The error of this test is shown in Figure 4.31. Its structure is comparable with the error structure of the test of the SASLDG-3c method on the 120×120 grid. The maximal error is located at the same location, i.e. $[0.0, -0.47]$, and takes the value 0.0149. The l_1-error is given by 0.0008.

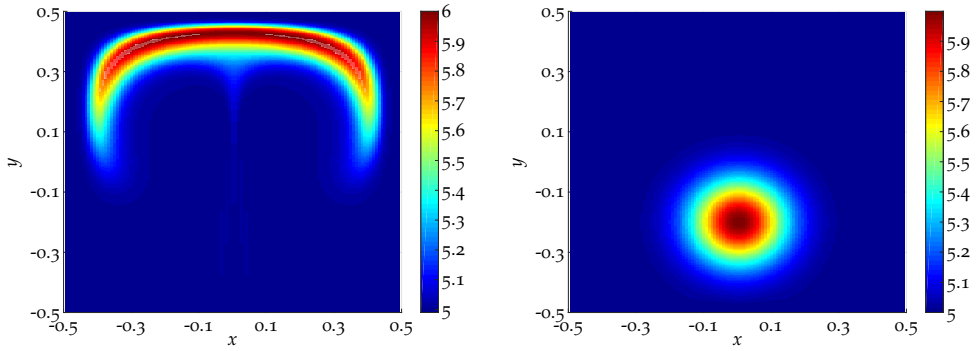
The time for the computation of the test with the SASLDG-3c method on a grid of 120×360 grid cells and CFL number 0.5 (y) with respect to the fine grid in vertical direction is 5058.7s.

Hybrid method - ratio grid 1:3

We examine the numerical result of the hybrid method on the grid with 120×360 cells, which yields a grid aspect ratio of 1:3. Using this grid we will study the influence of two different CFL numbers on the numerical solution.

The first choice of the Courant number is the same as in the previous test, that is the Courant number equals to 0.5 (y) in vertical direction with respect to the fine grid and to approximately 0.167 (x) in horizontal direction. The actual CFL number for the vertical direction decreases by the factor of three to 0.167 (y), when it refers to the coarser grid of 120 cells used for the SASLDG-3c method.

The results of the hybrid scheme at time $t = 0.9$ and at time $t = 1.8$ are shown in Figure 4.32(a) and Figure 4.32(b), respectively. The minimum and maximum of the solution at time $t = 0.9$ are located at the same points, i.e. at $[0.01, -0.01]$ and $[0.0, 0.43]$, respectively, as in the results of the hybrid method on the 120×120 cells grid. The minimum takes the values 4.990, the maximum is given by 6.001. The solution after the backward cycle at time $T_{\max}$ has a minimum of 5.0 and a



(a) The numerical solution at time $t = 0.9$.

(b) The numerical solution at time $t = 1.8$.

Figure 4.32: The numerical solution of the Rider Kothe test computed with the hybrid method on a 120×360 grid with CFL number 0.5 (y) w.r.t. the vertical direction.

maximum of 5.999, located at the analytically correct position at the peak of the gaussian hill.

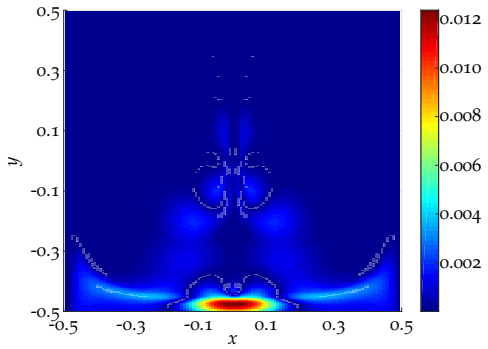


Figure 4.33: Absolute error of the solution of the hybrid method shown in Figure 4.32(b).

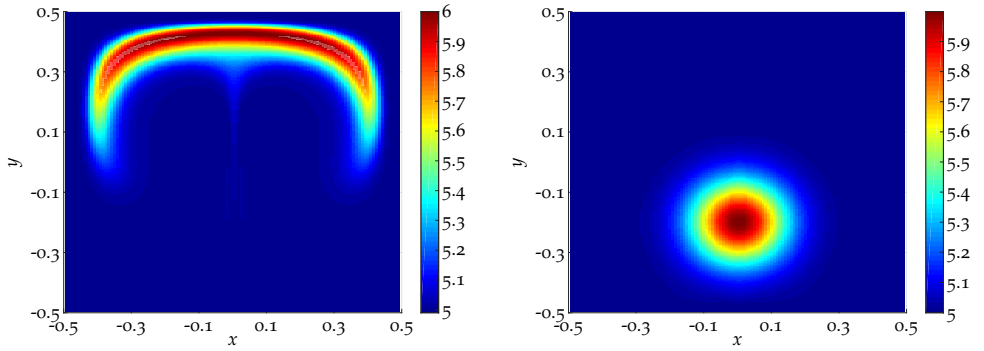
The error of the hybrid method is displayed in Figure 4.33. The error is a hybrid version of the errors of the individual methods. The maximum of 0.0124, located near the bottom of the domain at $[0.01, -0.48]$, takes the same value as the maximal error of MPDATA. The error structure of MPDATA is given again in the area above $y = -0.4$, where larger error peaks of the SASLDG-3c method are located. In contrast, the region with errors of smaller magnitude of the SASLDG-3c method is

taken at the position of the second largest error peak for MPDATA (at $[0.0, -0.14]$). Summarized, the solution of the hybrid method takes the respective smaller error of the methods MPDATA and SASLDG-3c. Apart from the maximal error peak, the other errors remain below the value 0.002. Hence, the l_1-error of the hybrid method, given by 0.0004, is smaller than for the individual methods.

The time for the computation of the test with the hybrid method on a grid of 120×360 grid cells and CFL number 0.5 (y) with respect to the fine grid in vertical direction is 3698.8s.

For the concluding test case we retain the grid resolution and the method but change the CFL number. The size of the time step is determined from the CFL number of 1.0 (x) in horizontal direction. For the fine grid of 360 grid cells, the CFL number yields the value 2.99 (y) for the vertical direction. If we refer the CFL number to the coarser vertical grid used for the advection step in the SASLDG-3c scheme, the Courant number is approximately 1.0 (x).

The results from this test can be found in Figure 4.34(a) for the largest deformation at time $t = 0.9$. The minimum and maximum of the numerical solution for CFL 1.0 (x) shows the same values at the same position as the solution for CFL 0.5 (y) above. Figure 4.34(b) displays the numerical solution after the backward cycle at



(a) The numerical solution at time $t = 0.9$.

(b) The numerical solution at time $t = 1.8$.

Figure 4.34: The numerical solution of the Rider Kothe test computed with the hybrid method on a 120×360 grid with CFL number 1.0 (x) w.r.t. the horizontal direction.

time $T_{\max}$. Again, the values of minimum and maximum bear a resemblance with the solution computed with CFL 0.5 (y). The maximum at the point $[0.0, -0, .2]$ takes the value 5.999. The minimum of 4.999 at the point $[-0.03, 0.18]$ is located similarly to the minimum of the solution of MPDATA on the 120×120 grid.

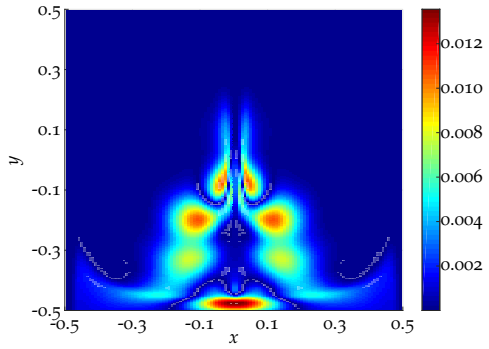


Figure 4.35: Absolute error of the solution of the hybrid method shown in Figure 4.34(b).

The error of the numerical solution is shown in Figure 4.35. The maximal error of 0.135 is located at the same position as for the tests above. Though, the magnitude

of the maximal error is enlarged by a small amount, the errors at the other erroneous areas increase by a large factor and reach values above 0.01. Therefore, the l_1-error takes the value 0.0011, which is the largest of all tests conducted on this grid.

The time for the computation of the test with the hybrid method on a grid of 120×360 grid cells and CFL number 1.0 (x) with respect to the grid in horizontal direction is 617.1s.

	CFL	min	max	l_∞-error	l_1-error	time (s)
MPDATA	0.5 (y)	5.000	5.999	0.0124	0.0006	2820.3
SASLDG-3c	0.5 (y)	4.989	5.999	0.0149	0.0008	5058.7
hybrid	0.5 (y)	5.000	5.999	0.0124	0.0004	3698.8
hybrid	1.0 (x)	4.999	5.999	0.0135	0.0011	617.1

Table 4.4: Characteristic values for the Rider Kothe test computed on a 120×360 grid.

Table 4.4 gives an overview of the CFL numbers used in the tests, the corresponding minimum and maximum values of the respective numerical solutions at time $T_{\max}$, as well as the l_∞-error and the l_1-error. Additionally, the computational time is listed. The (x) and the (y) written in the column of the CFL number give the space dimension the CFL number refers to.

4.3.6 Wavelike flow test

In the test we examine the numerical results computed from discontinuous non-smooth initial data and a time dependend velocity field that is wavelike and not as much deforming as in the deformational flow tests above. The velocity field is given by

$$\begin{aligned} u(x,y,t) &= -\frac{1}{2Hk}\cos\Big(\pi\big(\frac{y}{H}\big)\Big)\sin(2\pi k(x-ct)) \\ v(x,y,t) &= \sin\Big(\pi\frac{y}{H}\Big)\cos(2\pi k(x-ct)), \end{aligned} \tag{4.43}$$

where H is the height, k the frequency and c a coefficient which determines the degree of deformation. We set $H = 1$, $k = 1$ and $c = 4$ in the tests. The velocity field forces upwards and downwards advection as well as advection to the left and right in turn in a wavelike form. Note that the maximum magnitude of the vertical velocity component v is twice as large as of the horizontal velocity component u. A plot of the velocity field at time $t = 0$ is shown in Figure 4.36. The largest magnitude of the horizontal velocity is located at the top and at the bottom of the computational domain at the initial time. The largest component of the vertical velocity acts in the middle region and at the left and right boundary of the computational domain. The velocity field is shifted to the right in time. A full cycle of the vertical movement is reached at time $t = 1/c$. However, the overall advection deforms the solution such that the initial distribution is not recovered at that time. To simulate a longer period of time, the solution is computed for five cycles applying the given velocity field (4.43) up to the point in time $t = 5/c$. In order to obtain a state of the solution for which the analytical solution is known, the velocity field is reversed. Using the reversed velocity field, the numerical solution is again computed for five cycles. Therefore, the chosen maximum time for this test is $T_{\max} = 10/c$.

The initial values are shown in Figure 4.37. The distribution contains a sharp jump discontinuity. It can represent a sharp boundary layer in either space direction of any tracer in the atmosphere or the ocean. For example, one region can represent the structure of a cloud with water droplets as tracer, the other region stands for an area of less humidity. Another example is the salinity in the ocean. There are layers that consist of a high salt concentration. The correct advection of these tracers is of

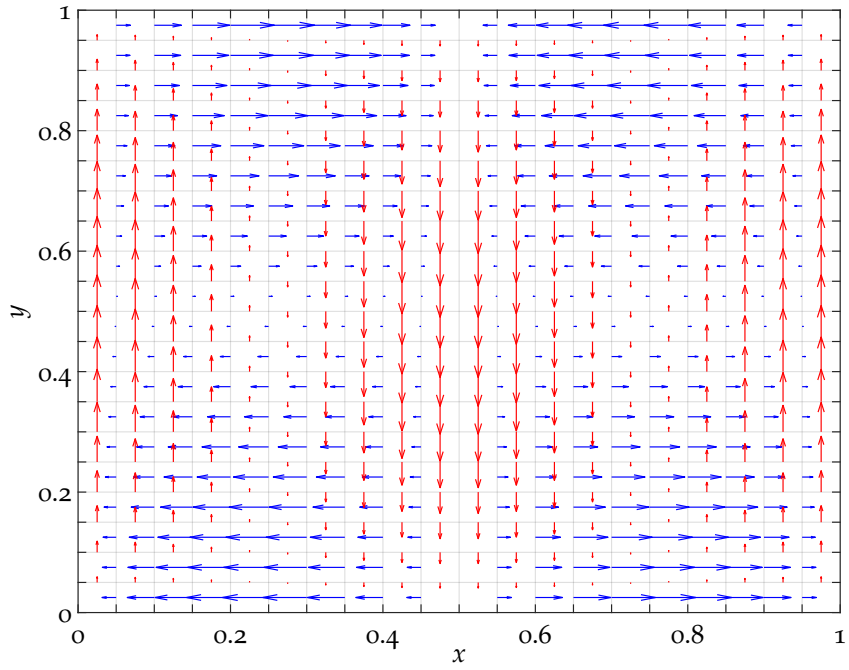
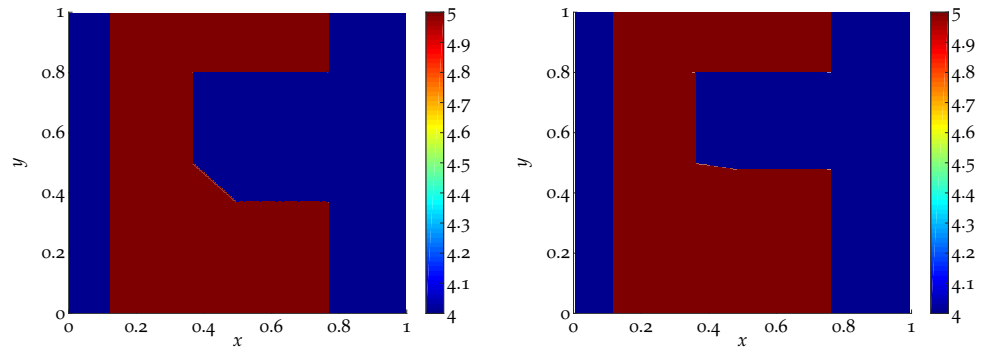


Figure 4.36: The velocity field used for the wavelike flow test at time $t = 0$.

importance, see e.g. [26]. Thus, this test studies the performance of the numerical schemes with focus on the maintenance of the sharp boundary.

We conduct numerical tests on two different grids. The first grid has an aspect ratio of 1:1 with 150 grid cells in both space directions. The second grid is of 90×540 grid cells and hence a grid with aspect ratio of 1:6. On each of these grids, we compute the numerical solution with operator splitting with MPDATA in both dimensions, with the SASLDG-3c method in both dimensions and with the hybrid method, where MPDATA is used for the horizontal advection and SASLDG-3c for the vertical one. Furthermore, the Courant number is increased in the latter test case of the hybrid method to show its influence on the numerical solution. Also, the computational time is measured to compare the cost of the different algorithms and settings.

The evolution of this test is shown in Figure 4.38 to see the maximal deformation of one vertical cycle. It pictures snapshots at different points in time of the first

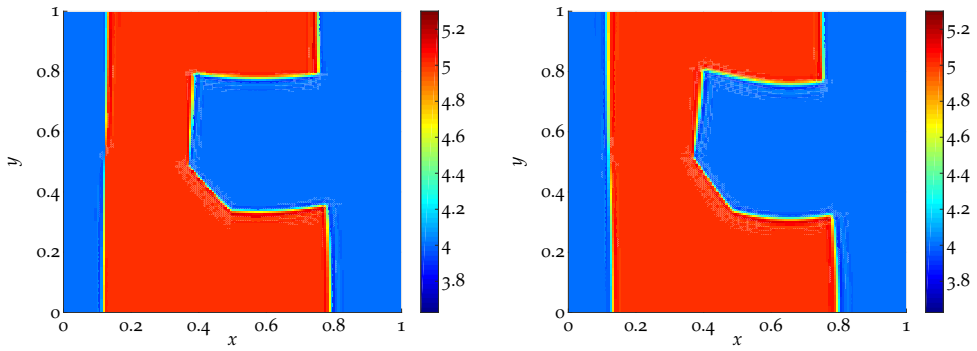


(a) The initial values for the 150×150 grid.

(b) The initial values for the 90×540 grid.

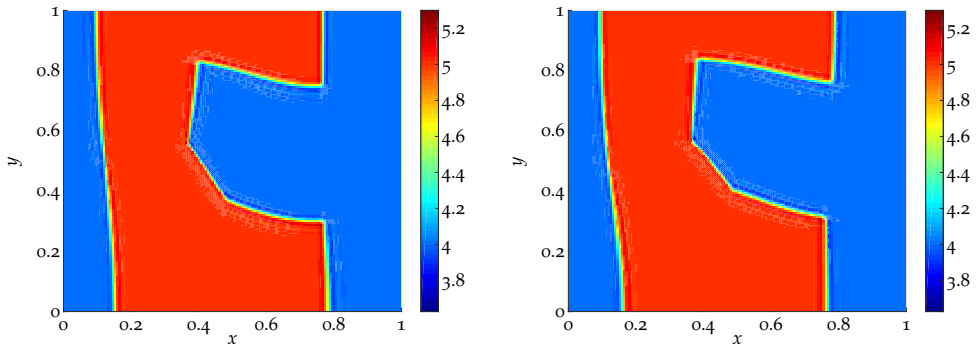
Figure 4.37: The initial values used for the wavelike flow test.

cycle. The data is obtained from the SASLDG-3c method on a grid of 150×150 cells. The CFL number used for the computation is 0.5 with respect to the grid in vertical direction, which we abbreviate by 0.5 (y).



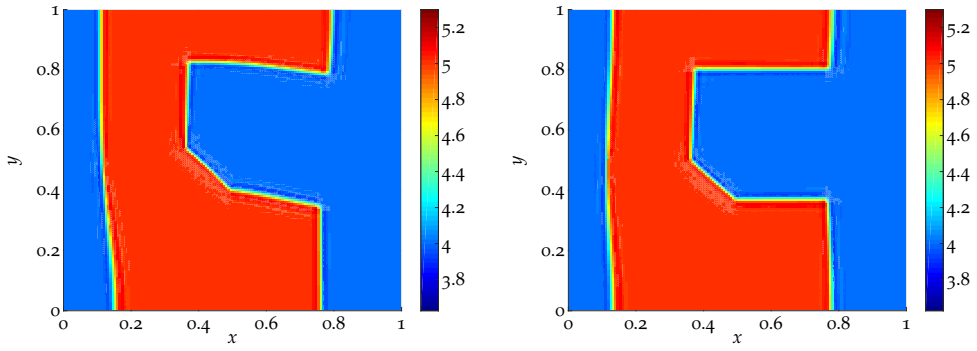
(a) 1/6 cycle: time step 12

(b) 1/3 cycle: time step 24



(c) 1/2 cycle: time step 37

(d) 2/3 cycle: time step 50



(e) 5/6 cycle: time step 62

(f) one cycle: time step 75

Figure 4.38: Evolution of one cycle of the wavelike test. The plotted numerical solution is obtained by SASLDG-3c on a 150×150 grid with the Courant number 0.5 (y).

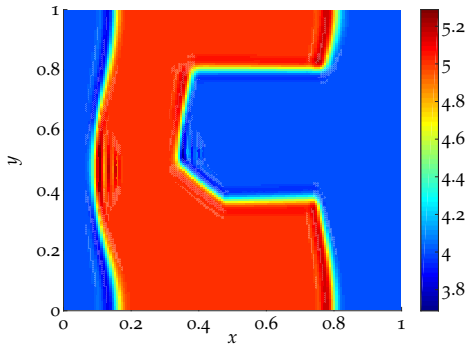
We use the initial distribution shown in Figure 4.37(a). The snapshots are taken at every sixth of the cycle. The main impact of the velocity is visible at the cut-out corner. In particular, it reflects the vertical advection. The effect of the horizontal velocity components is primarily noticeable at the top and at the bottom of the vertical edges. The largest vertical velocity in downwards direction starts to act on the cells at $x = 0.5$ at time $t = 0$.
After 12 time steps, the point of the largest negative velocity is shifted by approximately 0.16 to the right. Thus, it is is located approximately at $x = 0.66$. The impact can be seen in Figure 4.38(a), which displays the numerical solution after 12 time steps. The solution shows a deformation of the horizontal edges of the cut-out cor-

ner.
The solution after 24 time steps is plotted in Figure 4.38(b). The effect of the horizontal velocity starts to be visible, too. The upper part of the left edge is advected to the left, whereas the lower parts tends to the right. The vertical upwards velocity acts on the left part of the cut-out corner.
After 37 time steps, half of the cycle is processed, which can be seen in Figure 4.38(c). The largest deformation of the left edge being rotated counterclockwise is reached. The left part of the cut-out corner has reached the highest point.
Figure 4.38(d) shows the numerical solution after 50 time steps. The point of the largest upwards velocity has moved to approximately $x = 0.65$ and thus acts on the middle part of the cut-out corner.
The solution after 62 time steps is pictured in Figure 4.38(e). The left edge starts to visibly move back clockwise. The largest upwards advection is located at $x \approx 0.83$, whereas the largest downwards advection takes place at $x \approx 0.33$. As a consequence, the left part of the cut-out corner starts moving downwards, whereas the right part of the corner still is advected upwards.
The numerical solution after 75 time steps, which is a vertical cycle, is shown in Figure 4.38(f). The overall deformation, which increases in time, is barely visible after one cycle. The vertical discontinuities begin to form a curve. The solution after five cycles is shown for each respective test case. The deformation becomes obvious on these plots.

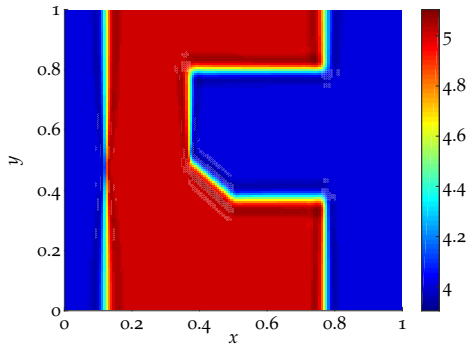
MPDATA - grid ratio 1:1

We begin with the numerical solution computed with MPDATA on the grid of 150×150 cells shown in Figure 4.39. The CFL number is chosen to be 0.5 with respect to the vertical grid, i.e. 0.5 (y). The corresponding CFL number for horizontal advection is approximately 0.25 (x). The numerical solution after five cycles is pictured in Figure 4.39(a). It reveals over- and undershoots, as it takes values between 3.688 and 5.289. The over- and undershoots decrease in the numerical solution after five more cycles in backward direction (plotted in Figure 4.39(b)) since the minimum and maximum take the value 3.905 and 5.104, respectively.

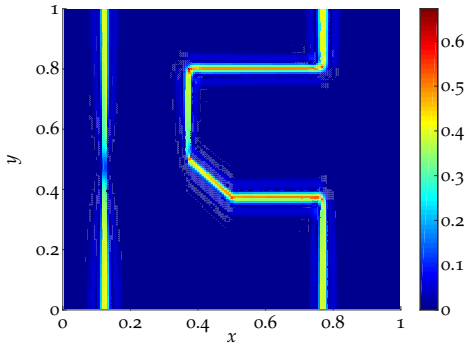
The error is plotted in Figure 4.39(c), which consists of the absolute difference of the numerical solution and the initial values. The main error occurs along the discontinuity. The maximal error of 0.673 is a narrow peak located at the left corner in the top of the cut-out corner. The error of the horizontal edges as well as the error of the whole diagonal part of the the cut-out corner is of roughly 0.5 and larger than the error of the vertical discontinuities, which is approximately 0.4.
When measuring the width of the error along the discontinuity, we count the grid cells where the error is above the value 0.1. With this definition, the width of the error of the left vertical edge is between two and four grid cells. The width of four grid cells occurs at the top and at the bottom of the left edge, where the largest horizontal velocities are applied. These cause oscillations which enlarge the region of the error. Smaller oscillations appear also in the region in the middle of the left vertical edge. These are of smaller amplitude because the horizontal velocity is smaller within this area. The large vertical velocity that acts on this region does not cause horizontal oscillations. The error in the cut-out corner stretches over two to four grid cells at the vertical edge and seven and five grid cells along the horizontal bottom and top edge, respectively.
In general, when we study the left vertical edge, it is significant that the error is larger in middle and at the very top and bottom. In between it is of smaller magnitude. This can be explained by the velocity field. The maximal vertical velocity is reached in the middle region, the maximal horizontal velocity at the top and bottom. Thus, the advection is larger in these regions which leads to the enhanced error.
The l_1-error computed from the sum of the cell averages is approximately 0.026.



(a) The numerical solution after five cycles.



(b) The numerical solution after ten cycles.



(c) Absolute error of numerical solution.

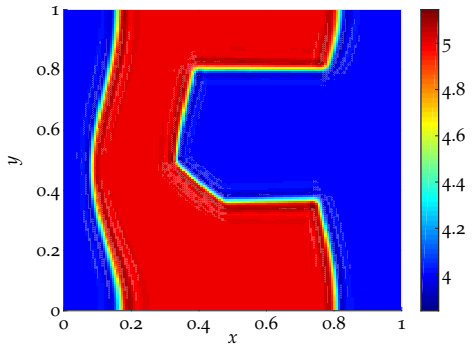
Figure 4.39: The wavelike test computed with MPDATA on a 150×150 grid with CFL number 0.5 (y).

The time for the computation of this test case with MPDATA on a grid with 150×150 grid cells and CFL number 0.5 (y) is 874.6s.

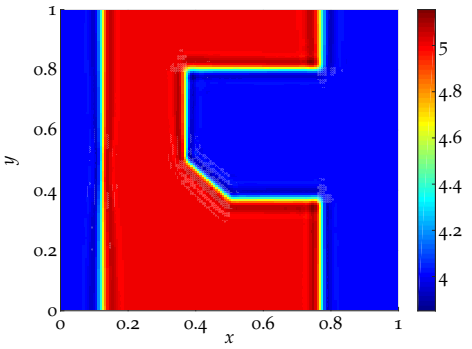
SASLDG-3c - grid ratio 1:1

The result from the computation of the wavelike flow advection test of ten cycles with the SASLDG-3c method on a grid of 150×150 grid cells is shown in Figure 4.40. The CFL number applied for this test is 0.5 (y) with respect to the fine grid in vertical direction. The corresponding Courant number in horizontal direction is 0.25 (x). The CFL numbers with respect to the coarse grid, which is used for the advection steps, are given by 0.16 (y) and 0.08 (x), respectively.
A plot of the solution after five cycles is shown in Figure 4.40(a). The over- and undershoot are of less amplitude than in the case of MPDATA. The minimum and maximum of the numerical solution at that time take the value 3.844 and 5.146, respectively. The solution after five more cycles with the reversed velocity is pictured in Figure 4.40(b). The minimum takes the value 3.852, which is smaller than the corresponding value of the MPDATA solution. The maximum of the SASLDG-3c solution given by 5.156 is larger than the maximum of the corresponding MPDATA solution.

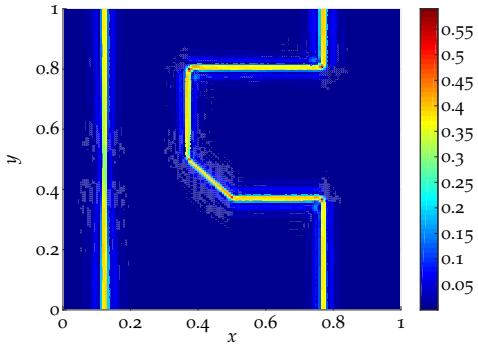
The absolute error is shown in Figure 4.40(c). Even though, the over- and undershoots are of larger magnitude than the ones in the MPDATA test above, the maximal error is smaller in the SASLDG-3c test. It takes the value 0.5903. The smaller error indicates a sharper representation of the discontinuities. The largest errors occur at the corners at the cut-out corner. The errors along the discontinuities apart from the corners do not exceed the value 0.4..
The width of the error along the discontinuity is measured, where the error is larger



(a) The numerical solution after five cycles.



(b) The numerical solution after ten cycles.



(c) Absolute error of numerical solution.

Figure 4.40: The wavelike test computed with SASLDG-3c on a 150×150 grid with CFL number 0.5 (y) w.r.t. the fine grid.

than the value 0.1. The error of the left vertical edge spans over two to four grid cells, just as the result for MPDATA. The error along the edges of the cut-out corner has a width of four grid cells along the horizontal parts and a width of two to three grid cells along the vertical parts of the edges. Thus, the error region of the horizontal discontinuities of the cut-out corner, where the largest velocities are applied, is kept to a more narrow area than for MPDATA. The l_1-error is 0.024 and thus smaller than the l_1-error of MPDATA.

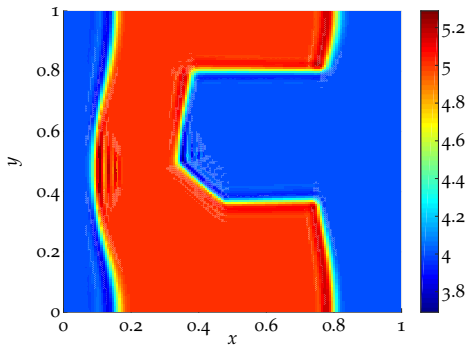
The time for the computation of this test case with SASLDG-3c on a grid with 150×150 grid cells and CFL number $0.5(y)$ with respect to the vertical fine grid is 1653.8s.

Hybrid method - grid ratio 1:1

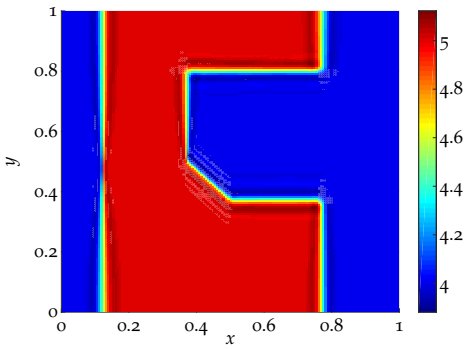
The last scheme that we examine on the grid with 150×150 cells and hence a aspect ratio of 1:1, is the hybrid method. It consists of using MPDATA in horizontal and SASLDG-3c in vertical direction. The Courant number is chosen to be $0.5(y)$ with respect to the vertical fine grid, as done in the preceding tests. Thus, the actual CFL number on the coarse grid in the advection step of the SASLDG-3c method is approximately 0.16 (y).

Figure 4.41 shows the numerical results. The solution of the hybrid method after five cycles is plotted in Figure 4.41(a). The over- and undershoots are very similar to the MPDATA version. The maximum and the minimum of the numerical solution take the value 5.288 and 3.692, respectively. The solution after five more cycles is shown in Figure 4.41(b). The amplitude of the maximal over- and undershoots reduces. The maximum and the minimum are given by 5.122 and 3.882, respec-

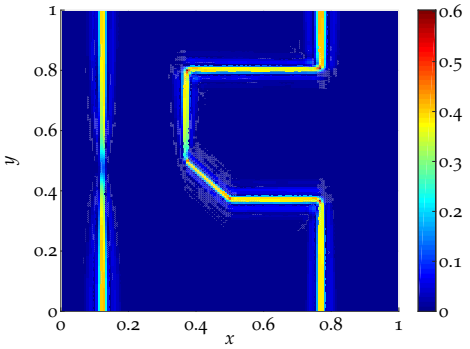
tively. These values are in between the results of using MPDATA and SASLDG-3c individually.



(a) The numerical solution after five cycles.



(b) The numerical solution after ten cycles.



(c) Absolute error of numerical solution.

Figure 4.41: The wavelike test computed with the hybrid method on a 150×150 grid with CFL number 0.5 (y) w.r.t. the fine grid.

The absolute error is displayed in Figure 4.41(c). The error plot reveals the resemblance to both methods MPDATA and SASLDG-3c. The horizontal velocities that force the horizontal advection solved with MPDATA act mainly on the vertical edges. The vertical edges in the error plot of the hybrid method is of the same structure and magnitude of approximately 0.4 as the error of MPDATA. Respectively, the error of the horizontal edges of the hybrid method is similar to the error of approximately 0.4 of the SASLDG-3c scheme.
The width of the error along the discontinuity for this test case also reflects the combination of the two methods. The error covers a region of the width of two to four grid cells at the left vertical edge. At the cut-out corner, the error width is four grid cells along the horizontal edge and two to four along the vertical edge.
The maximal absolute error takes the value 0.603. The l_1-error is 0.024. Thus, the errors of hybrid method is in between the errors of MPDATA and the SASLDG-3c method.

The time for the computation of this test case with the hybrid method on a grid with 150×150 grid cells and CFL number $0.5(y)$ with respect to the vertical fine grid is 1125.8s.

Table 4.5 summarizes characteristic values of the previous tests. It lists the CFL numbers used in the tests, the corresponding minimum and maximum values of the respective numerical solutions at time $T_{\max}$, and the l_∞-error and the l_1-error. Additionally, the computational time is given.

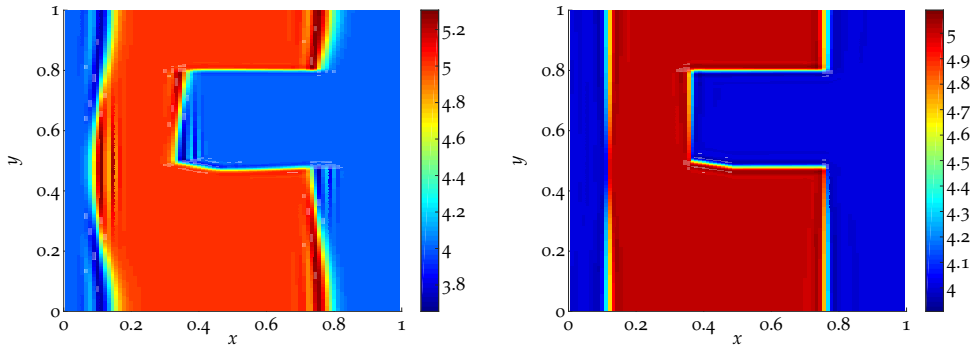
	CFL	min	max	l_∞-error	l_1-error	time (s)
MPDATA	0.5 (y)	3.905	5.104	0.673	0.026	874.6
SASLDG-3c	0.5 (y)	3.852	5.156	0.590	0.024	1653.8
hybrid	0.5 (y)	3.882	5.122	0.603	0.024	1125.8

Table 4.5: Characteristic values for the wavelike test computed on a 150×150 grid.

MPDATA - grid ratio 1:6

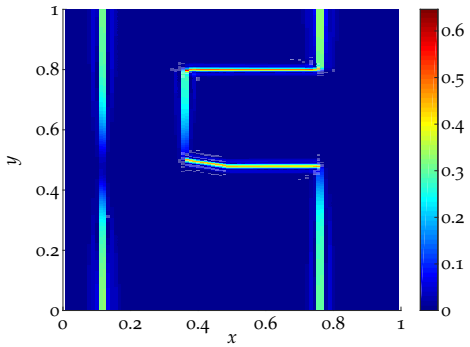
For the next series of tests we change the grid to 90×540 cells. This corresponds to a grid aspect ratio of 1:6. As for the previous test, we compute the numerical solution with MPDATA, the SASLDG-3c method and the hybrid method. The hybrid method is examined with regard to two different Courant numbers.

Figure 4.42 shows the results of the computation using the Courant number 0.5 (y). The corresponding CFL number for the horizontal direction is given by 0.04 (x). The numerical solution after five cycles computed by MPDATA on the grid of 90×540 cells is displayed in Figure 4.42(a). The maximum and the minimum of the numerical solution at time $t = 5/c$ take the value 5.306 and 3.655, respectively. The solution at time $t = T_{\max}$ is show in Figure 4.42(b). The maximal overshoot takes the value 5.091, the maximal undershoot is given by 3.908.



(a) The numerical solution after five cycles.

(b) The numerical solution after ten cycles.



(c) Absolute error of numerical solution.

Figure 4.42: The wavelike test computed with MPDATA on a 90×540 grid with CFL number 0.5 (y).

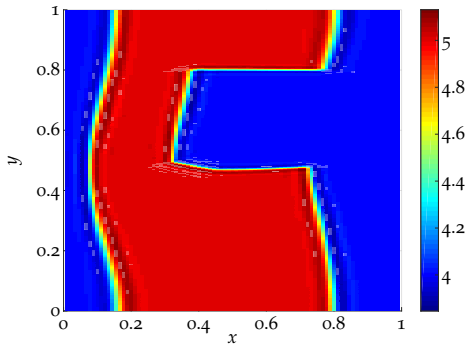
The absolute error is plotted in Figure 4.42(c). The maximal error is a single peak, which takes the value 0.646. It is located at the upper left corner of the cut-out corner. The area of the largest error apart from the peak consists of the horizontal edges of the cut-out corner. This erroneous area is approximately of value 0.5. The error at the vertical edges is of less amplitude and takes approximately the value

0.3.
We define the width of the error as for the tests before. The number of grid cells that have an error of more than 0.1 contribute to the width of the respective error location. So, the error width of the left vertical edge is two grid cells. The error at the cut-out corner stretches over eight and seven grid cells along the lower and upper horizontal edge, respectively, and over two grid cells along the vertical edge. The l_1-error takes the value 0.017.

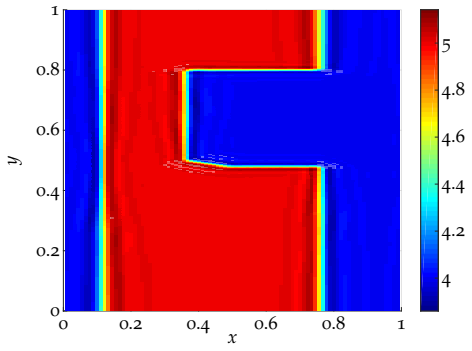
The time for the computation of this test case with MPDATA on a grid with 90×540 grid cells and CFL number 0.5 (y) is 6826.4s.

SASLDG-3c - grid ratio 1:6

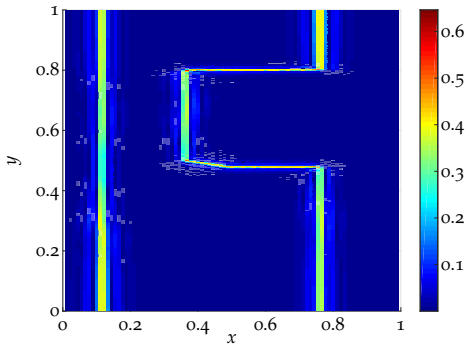
The results computed with SASLDG-3c on the grid of 90×540 grid cells is displayed in Figure 4.43. The CFL number 0.5 (y) is used for the computation and thus 0.04 (x) is the CFL number for the horizontal advection. The numbers change to 0.16 (y) and 0.014 (x), respectively, with respect to the coarse grid. Figure 4.43(a) shows the numerical solution after five cycles. The maximum and minimum of the solution take the value 5.130 and 3.846, respectively. After another five cycles with the reversed velocity, the maximum and minimum are given by 5.142 and 3.860, respectively. The solution at time $T_{\max}$ is displayed in Figure 4.43(b).



(a) The numerical solution after five cycles.



(b) The numerical solution after ten cycles.



(c) Absolute error of numerical solution.

Figure 4.43: The wavelike test computed with SASLDG-3c on a 90×540 grid with CFL number 0.5 (y) w.r.t. the vertical fine grid.

The absolute error of SASLDG-3c is pictured in Figure 4.43(c). The maximal error of 0.646 is located at the upper left corner of the cut-out corner, which forms a narrow peak. The error along all edges of the discontinuity is of the same order and takes approximately the value 0.4.
The maximum width of error along the edges is the same for all edges. The errors located along the left vertical edge, as well as along the edges in the cut-out-corner

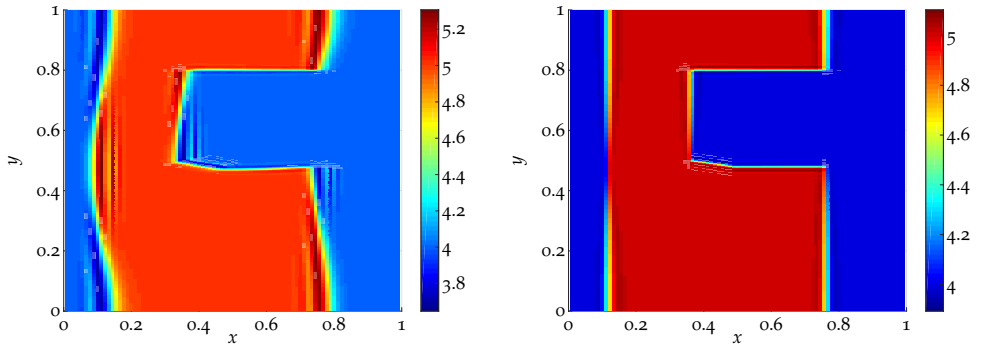
have the width of four grid cells.
The l_1-error takes the value 0.029.

The time for the computation of this test case with SASLDG-3c on a grid with 90×540 grid cells and CFL number 0.5 (y) with respect to the fine vertical grid is 13256.7s.

Hybrid method - grid ratio 1:6

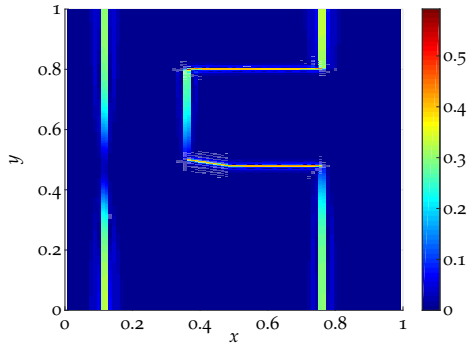
The concluding test case is the computation of the numerical solution with the hybrid method on the grid of 90×540 grid cells, and thus the aspect ratio of 1:6. We conduct tests with two different Courant numbers. The numerical solution is determined with the CFL number 0.5 with respect to the vertical and the horizontal grid, i.e. CFL 0.5 (y) and 0.5 (x), respectively.

The solution of the first test is displayed in Figure 4.44. We compute the numerical solution with the hybrid method on the grid of 90×540 cells. We apply the CFL number 0.5 (y) with respect to the vertical fine grid, which corresponds to the number 0.16 (y) w.r.t. the coarse grid of the advection step used for the SASLDG-3c method. The solution at time $t = 5/c$ is pictured in Figure 4.44(a). The minimum of the solution of 3.648 and the maximum of 5.306 resemble the values of MPDATA. Also, the minimum and maximum of the solution after five more cycles, which take the value 3.892 and 5.103, respectively, are similar to the respective values of MPDATA. The solution at time $T_{\max}$ is shown in Figure 4.44(b).



(a) The numerical solution after five cycles.

(b) The numerical solution after ten cycles.



(c) Absolute error of numerical solution.

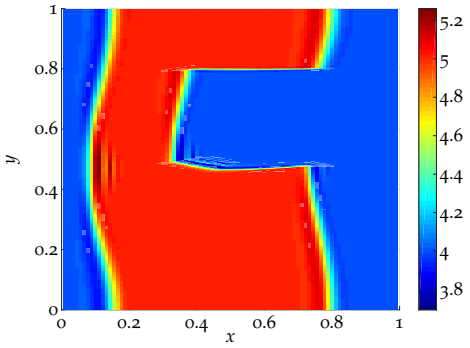
Figure 4.44: The wavelike test computed with the hybrid method on a 90×540 grid with CFL number 0.5 (y) w.r.t. the fine vertical grid.

The error of this test is shown in Figure 4.44(c). The overall error of the hybrid method is composed of errors of the two methods MPDATA and SASLDG-3c. The vertical edges are driven mainly by the horizontal velocity and thus advected with MPDATA. The error of the magnitude of approximately 0.3 developed at these

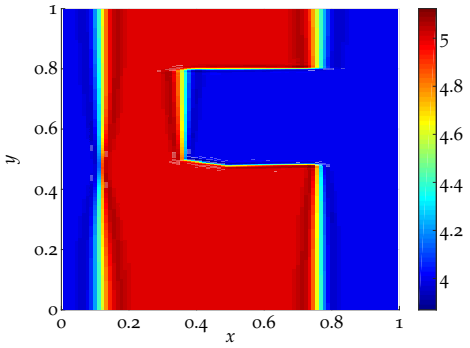
edges bears a resemblance with the error of MPDATA. Correspondingly, the error of approximately 0.4 of the horizontal edges is similar to the error of SASLDG-3c. The same composition holds for the width of the error. The width of the error along all vertical edges is two grid cells, just as for MPDATA. The error of the horizontal discontinuities stretches over five and four grid cells of the lower and the upper edge of the cut-out corner, respectively.
The combination of the respective small errors for the horizontal edges advected with SASLDG-3c and the vertical edges computed by MPDATA lead to a smaller maximal error of 0.592 and to a smaller l_1-error of 0.016 than for the individual methods.

The time for the computation of this test case with the hybrid method on a grid with 90×540 grid cells and CFL number 0.5 (y) is 8949.4s.

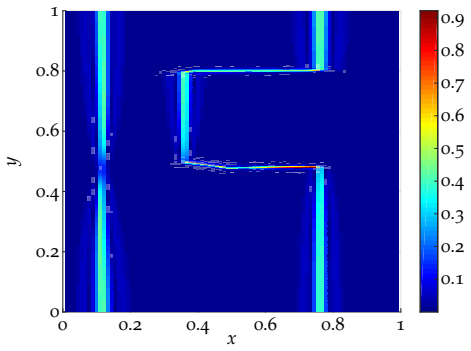
The next test case shows the results of the hybrid method computed with the CFL number 0.5 (x) with respect to the horizontal grid. This corresponds to the approximate CFL number 6.0 (y) in vertical direction w.r.t. the fine grid. Thus, the Courant number applied in the vertical advection step on the coarse grid of 180 grid cells for the SASLDG-3c method is 2.0 (y). The time step used in this setting is approximately 12 times larger than for the tests conducted above.
The results for this computation are shown in Figure 4.45. The numerical solution after five cycles is displayed in Figure 4.45(a). The minimum and the maximum of the numerical solution at time $t = 5/c$ take the value 3.694 and 5.261, respectively. These values are comparable to the previous test with CFL number 0.5 (y). The solution after the next five cycles is shown in Figure 4.45(b). The maximal over- and undershoot are given by 5.123 and 3.870 and are similar to the results above.



(a) The numerical solution after five cycles.



(b) The numerical solution after ten cycles.



(c) Absolute error of numerical solution.

Figure 4.45: The wavelike test computed with the hybrid method on a 90×540 grid with CFL number 0.5 (x) w.r.t. the horizontal direction.

The error plot is given in Figure 4.45(c). Even though the maximum and minimum of the numerical solution is comparable to the tests described before, the

maximum absolute error of 0.921 is larger. The main error is located at the horizontal edges of the cut-out corner. In the respective middle at approximately $x = 0.5$ the error takes the value 0.4 and increases towards the left and right. The error at the lower edge is larger than at the upper edge and reaches the maximal error at the right corner of the lower edge. The errors at the vertical edges are smaller and take approximately the value 0.4, which is larger than in the previous tests. Since the maximal over-and undershoot 5.123 and 3.870 at time $T_{\max}$ deviates only by 0.123 and 0.130, respectively, from the analytical solution, the large maximal error of 0.921 origins from a displacement of the discontinuity. The lower edge of the cut-out corner is not advected to the correct position parallel to the x-axis.
The width of the error is barely enlarged. Hence, the error of horizontal edges in the cut-out corner stretches over four to five grid cells. The error of the vertical edges stretches over two to four grid cells.
The l_1-error takes the value 0.030.

The time for the computation of this test case with the hybrid method on a grid with 90×540 grid cells and CFL number 0.5 (x) with respect to the horizontal fine grid is 791.4s.

	CFL	min	max	l_∞-error	l_1-error	time (s)
MPDATA	0.5 (y)	3.908	5.091	0.646	0.017	6826.4
SASLDG-3c	0.5 (y)	3.860	5.142	0.646	0.029	13256.7
hybrid	0.5 (y)	3.892	5.103	0.592	0.016	8949.4
hybrid	0.5 (x)	3.870	5.123	0.921	0.030	791.4

Table 4.6: Characteristic values for the wavelike test computed on a 90×540 grid.

Table 4.6 shows characteristic values of the tests computed on a grid of 90×540 cells. It lists the CFL numbers used in the tests, the corresponding minimum and maximum values of the respective numerical solutions at time $T_{\max}$, and the l_∞-error and the l_1-error. Additionally, the computational time is displayed. The (x) and the (y) written in the column of the CFL number gives the space dimension the CFL number refers to.

5 ANALYSIS

In this chapter we want to examine the theoretical properties of the numerical method introduced in Chapter 3. We start with showing the consistency of the numerical method. We explicitly determine the error between the analytical and numerical solution that results after one time step. With the information of the error we can show the theoretical order of convergence. The next property that we examine is the stability. We find estimates for the L^1- and L^2-norm of the numerical solution and use the result for showing the conservation of mass. A von Neumann stability analysis is conducted additionally as it gives further insight into the properties of the numerical method. The convergence rates that are obtained by numerical tests are analyzed in the last section of this chapter.

5.1 CONSISTENCY

To show the consistency of the SASLDG method we analyze the error after one time step between the analytical solution and the numerical solution for Δx and Δt tending to zero. The order of the error reveals additionally the order of accuracy of the method.

The goal of this section is to explicitly compute the error, i.e. the difference between the numerical and analytical solution, that is done in one time step. To obtain the information on the error we first compute the exact solution to the linear advection equation after one single time step. To make the solution comparable to the numerical solution we use a Taylor expansion for the grid cell sizes going to zero to obtain a polynomial form of the exact solution. The second step consists of the computation of the numerical solution. This is carried out under the assumption of a Courant number of less than one and equidistantly distributed grid points. A general CFL number complicates the explicit specification of the coefficients of the numerical solution, because the trajectories could cross several grid cells. In the case of a CFL number of less than one and a strictly positive velocity, the solution for a grid i after one time step consists always of the information of the solution in grid cell $i-1$ and i before the time step. The procedure for negative velocity would be carried out analogously, hence we restrict the analysis to positive velocity. From the expanded analytical and numerical solution the error is obtained and the consistency verified.

For the determination of the order of accuracy one last step is needed. We find out that the error after one time step at time t^{n+1} in the ith grid cell depends on the coefficients of ρ_i given in (3.131), that is $m_{j,i}^n$ and $m_{j,i-1}^n$ for $j = 0, 1, 2$ of the former time step at time t^n and differences between them. To determine the order of accuracy we need to determine the order of all $m_{j,i}^n$ as Δx goes to zero, which is conducted by induction. The information on the coefficients and the error yield the order of accuracy.

Before we start to examine the analytical solution, we define the local Courant number σ_i, which is needed throughout this section,

$$\sigma_i = \frac{\max\limits_{0 \le x \le \Delta x} u_i(x) \Delta t}{\Delta x}. \tag{5.1}$$

We remind of the definition of u given in (3.13). Because of the piecewise linearity of u we assume w.l.o.g.

$$\sigma_i = \frac{b_i \Delta t}{\Delta x}. \tag{5.2}$$

This gives us the opportunity to rewrite the coefficient b_i, and thus introduce Δt and Δx in certain expressions, which can be used for series expansions.

To relate neigboring local CFL numbers we make use of (3.17), which states

$$a_{i-1}\Delta x + b_{i-1} = b_i. \tag{5.3}$$

Thus, we can rewrite σ_{i-1} as

$$\sigma_{i-1} = \sigma_i - a_{i-1}\Delta t. \tag{5.4}$$

which simplifies the computations in the following.

Courant numbers of less than one imply $\Delta t \in \mathcal{O}(\Delta x)$ and vice versa $\Delta x \in \mathcal{O}(\Delta t)$, which makes them interchangeable with respect to the limiting case of small Δx or Δt. Note that the quotient $\Delta t / \Delta x$ could converge to a constant for $\Delta t \to 0$ and $\Delta x \to 0$. However, either Δt and Δx occur separately or the term consists of the quotient multiplied with x, i.e. $\Delta t / \Delta x \cdot x$. Since $x \in [0, \Delta x]$ the expression tends to zero in the limit. Furthermore, it holds $x / \Delta x \in \mathcal{O}(1)$ for the same reason.

5.1.1 Analytical solution after one time step

The analytical solution at time t^{n+1} is given by Proposition 3.3.1

$$\rho(x, t^{n+1}) = \rho(\varphi(x, t^{n+1}, -\Delta t), t^n) \exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x} u(\varphi(x, t^{n+1}, t - \Delta t)) dt \right) \tag{5.5}$$

We determine the analytical solution using (5.5) for the ith grid cell with the given velocity distribution of this and the neigboring cells. We approximate the solution with a polynomial of degree two with the help of Taylor expansions. Each of the expressions appearing in the above equation are expanded about the point $\Delta x = 0$ or $\Delta t = 0$ as this is of interest for the consistency and convergence analysis.

Remembering the distinct types of trajectories summarized in Section 3.2.4 we have to differentiate between the case of the trajectory φ remaining in cell i and the case of φ crossing the boundary of cell $i-1$ and the ith cell.

The first case yields the analytical solution for the right interval of the ith grid cell $[\varphi(x_{i-1/2}, t^n, \Delta t), x_{i+1/2}]$. The trajectory, that remains within the ith grid cell going backwards in time is given by

$$\varphi(x, t^{n+1}, -\Delta t) = e^{-a_i \Delta t} \left(\frac{b_i}{a_i} + x \right) - \frac{b_i}{a_i}. \tag{5.6}$$

We reformulate the expression for the trajectory using (5.2) as follows

$$\varphi(x, t^{n+1}, -\Delta t) = e^{-a_i \Delta t} \left(\frac{\sigma_i \Delta x}{a_i \Delta t} + x \right) - \frac{\sigma_i \Delta x}{a_i \Delta t}. \tag{5.7}$$

This can be approximated by using a Taylor expansion about $\Delta t = 0$. The result yields

$$\varphi(x, t^{n+1}, -\Delta t) = \left(1 - a_i \Delta t + \frac{{a_i}^2 \Delta t^2}{2} \right) x + \sigma_i \Delta x \left(-1 + \frac{a_i \Delta t}{2} \right) + \mathcal{O}\left(\Delta t^3 \right). \tag{5.8}$$

The next term that we approximate is the expansion factor. In the case of the trajectory remaining within the ith cell it is of easy structure. The factor simplifies to

$$\exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x} u(\varphi(x, t^{n+1}, t-\Delta t))dt\right) = \exp(-a_i \Delta t) \tag{5.9}$$

and its approximation is given by

$$\exp(-a_i\Delta t) = 1 - a_i\Delta t + \frac{{a_i}^2\Delta t^2}{2} + \mathcal{O}\left(\Delta t^3\right). \tag{5.10}$$

The approximations of the terms can be inserted in (5.5) and then displayed in polynomial form with the Legendre basis functions $k_j(x)$, $j \in \{0,1,2\}$

$$\begin{aligned}\rho(x, t^{n+1}) = &\left(\left(\frac{(6m_{2,i})}{\Delta x^2}\right)\varphi(x, t^{n+1}, -\Delta t)^2 + \right.\\ &+\left(\frac{2m_{1,i}}{\Delta x} - \frac{6m_{2,i}}{\Delta x}\right)\varphi(x, t^{n+1}, -\Delta t) + m_{0,i} \\ &\left.-m_{1,i} + m_{2,i}\right)\left(1 - a_i\Delta t + \frac{{a_i}^2\Delta t^2}{2}\right) + \mathcal{O}\left(\Delta t^3\right) \qquad (5.11)\\ = &\, m^{n+1}_{2,i,ana,R}k_2(x) + m^{n+1}_{1,i,ana,R}k_1(x) + m^{n+1}_{0,i,ana,R}k_0(x) \qquad (5.12)\end{aligned}$$

with the coefficients $m^{n+1}_{j,i,ana,R}, j = \{0,1,2\}$ given by

$$\begin{aligned}m^{n+1}_{0,i,ana,R} = &\left(\frac{{a_i}^2 m_{0,i}}{2} + \frac{3{a_i}^2 m_{1,i}}{2} + \frac{7{a_i}^2 m_{2,i}}{2} - \frac{7\sigma {a_i}^2 m_{1,i}}{3}\right.\\ &\left.-12\sigma {a_i}^2 m_{2,i} + \frac{25\sigma^2 {a_i}^2 m_{2,i}}{2}\right)\Delta t^2\\ &+\left(3\sigma a_i m_{1,i} - a_i m_{1,i} - a_i m_{2,i} - a_i m_{0,i}\right.\\ &\left.+6\sigma a_i m_{2,i} - 12\sigma^2 a_i m_{2,i}\right)\Delta t \qquad (5.13)\\ &+6m_{2,i}\sigma^2 - 2m_{1,i}\sigma + m_{0,i} + \mathcal{O}\left(\Delta t^3\right),\\ m^{n+1}_{1,i,ana,R} = &\left(15\sigma a_i m_{2,i} - 3a_i m_{2,i} - 2a_i m_{1,i}\right)\Delta t + m_{1,i} - 6\sigma m_{2,i}\\ &+\mathcal{O}\left(\Delta t^3\right), \qquad (5.14)\\ m^{n+1}_{2,i,ana,R} = &\, m_{2,i} + \mathcal{O}\left(\Delta t^3\right). \qquad (5.15)\end{aligned}$$

The index *ana* indicates that the coefficients belong to the analytical solution, the index R points out that the coefficients build the polynomial for the right interval $[\varphi(x_{i-1/2}, t^n, \Delta t), \Delta x]$ of the ith grid cell.

Now, we turn to the left part of the grid cell $[0, \varphi(x_{i-1/2}, t^n, \Delta t)]$. In this case all trajectories cross the grid cell boundary $x_{i-1/2}$. Equation (3.247) describes these trajectories. With the help of (5.2) and (5.4) it can be transformed to

$$\begin{aligned}&\varphi(x, t^{n+1}, -\Delta t)\\ &= \frac{u_i}{a_{i-1}}\exp\left(-a_{i-1}\Delta t\right)\left(\frac{a_i}{u_i}x + 1\right)^{\frac{a_{i-1}}{a_i}} - \frac{u_{i-1}}{a_{i-1}} \qquad (5.16)\\ &= \frac{\sigma_i \Delta x}{a_{i-1}\Delta t}\left(\exp\left(-a_{i-1}\Delta t\right)\exp\left(\frac{a_{i-1}}{a_i}\ln\left(\frac{a_i\Delta t}{\sigma_i\Delta x}x + 1\right)\right) - 1\right) + \Delta x. \qquad (5.17)\end{aligned}$$

The Taylor expansion about $\Delta t = 0$ yields an approximation to φ,

$$\begin{aligned}
&\varphi(x,t^{n+1},-\Delta t) \\
&\quad = \left(\left(\frac{a_{i-1}\,a_i}{2\,\sigma_i\,\Delta x} - \frac{{a_{i-1}}^2}{2\,\sigma_i\,\Delta x}\right)\Delta t^2 + \left(\frac{a_{i-1}}{2\,\sigma_i\,\Delta x} - \frac{a_i}{2\,\sigma_i\,\Delta x}\right)\Delta t\right) x^2 \\
&\qquad + \left(\frac{\Delta t^2\,{a_{i-1}}^2}{2} - \Delta t\,a_{i-1} + 1\right) x \\
&\qquad + \Delta x - \sigma_i\,\Delta x + \frac{\sigma_i\,\Delta t\,\Delta x\,a_{i-1}}{2} + \mathcal{O}\left(\Delta t^3\right).
\end{aligned} \tag{5.18}$$

Approximating the expansion factor we have to consider the piecewise definition of the velocity $u(x)$, which might differ from cell $i-1$ to cell i. Therefore, the time interval $[0, \Delta t]$ is split up into two pieces,

$$\exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x} u(\varphi(x, t^{n+1}, t-\Delta t))dt\right) = \exp\left(\int_0^{\tau} -a_{i-1}dt + \int_{\tau}^{\Delta t} -a_i dt\right). \tag{5.19}$$

The value τ defined in (3.31) refers to the time interval that passes while the flow starting in point x in the ith cell reaches the grid cell boundary $x_{i-1/2}$, i.e. $\varphi(x, t^{n+1}, -\tau) = x_{i-1/2}$. It is given by,

$$\tau = \frac{1}{a_i}\ln\left(\frac{a_i x}{b_i} + 1\right) \tag{5.20}$$

$$= \frac{1}{a_i}\ln\left(\frac{a_i \Delta t x}{\sigma_i \Delta x} + 1\right), \tag{5.21}$$

where (5.2) is used to replace b_i. Hence, the Taylor expansion about $\Delta t = 0$ of the expansion factor is given by

$$\begin{aligned}
&\exp\left(\int_0^{\tau} -a_{i-1}dt + \int_{\tau}^{\Delta t} -a_i dt\right) \\
&\quad = 1 - \left(a_{i-1} + \frac{x a_i}{\sigma_i \Delta x} - \frac{x a_{i-1}}{\sigma_i \Delta x}\right)\Delta t \\
&\qquad + \left(\frac{a_{i-1}^2}{2} + \frac{x a_i a_{i-1}}{\sigma_i \Delta x} - \frac{x a_{i-1}^2}{\sigma_i \Delta x} + \frac{x^2 a_i^2}{\sigma_i^2 \Delta x^2}\right. \\
&\qquad \left. - \frac{3x^2 a_i a_{i-1}}{2\sigma_i^2 \Delta x^2} + \frac{x^2 a_{i-1}^2}{2\sigma_i^2 \Delta x^2}\right)\Delta t^2 + \mathcal{O}\left(\Delta t^3\right).
\end{aligned} \tag{5.22}$$

Finally, we can determine the approximation of the analytical solution for the left part of the ith cell

$$\rho(x, t^{n+1}) = m_{2,i,ana,L}^{n+1} K_2(x) + m_{1,i,ana,L}^{n+1} K_1(x) + m_{0,i,ana,L}^{n+1} K_0(x). \tag{5.23}$$

The coefficients $m^{n+1}_{j,i,ana,L}, j = \{0,1,2\}$ are given below. The first coefficient $m^{n+1}_{0,i,ana,L}$ at time t^{n+1} reads

$$
\begin{aligned}
m^{n+1}_{0,i,ana,L} = \Big(\Big(& 2a_i^2 - \frac{a_i a_{i-1}}{2\sigma_i^2} + \frac{19 a_i a_{i-1}}{2\sigma_i} + 6\sigma_i a_i a_{i-1} + 36 a_{i-1}^2 \\
& + \frac{25\sigma_i^2 a_{i-1}^2}{2} - 32\sigma_i a_{i-1}^2 - \frac{2a_i^2}{\sigma_i} - \frac{15 a_{i-1}^2}{2\sigma_i} + \frac{a_i^2}{3\sigma_i^2} + \frac{a_{i-1}^2}{6\sigma_i^2} \\
& - \frac{45 a_i a_{i-1}}{2} \Big) m_{2,i-1} + \Big(\frac{7 a_i a_{i-1}}{2\sigma_i} - \frac{a_i a_{i-1}}{2\sigma_i^2} + 4a_{i-1}^2 - \frac{\sigma_i a_{i-1}^2}{3} - \frac{3 a_i a_{i-1}}{2} \\
& - \frac{17 a_{i-1}^2}{6\sigma_i} - \frac{2a_i^2}{3\sigma_i} + \frac{a_{i-1}^2}{6\sigma_i^2} + \frac{a_i^2}{3\sigma_i^2} \Big) m_{1,i-1} + \Big(\frac{a_{i-1}^2}{2} + \frac{a_i a_{i-1}}{2\sigma_i} - \frac{a_i a_{i-1}}{2\sigma_i^2} \\
& + \frac{a_{i-1}^2}{6\sigma_i^2} - \frac{a_{i-1}^2}{2\sigma_i} + \frac{a_i^2}{3\sigma_i^2} \Big) m_{0,i-1} \Big) \Delta t^2 \\
& + \Big(\Big(-12\sigma_i^2 a_{i-1} - 3\sigma_i a_i + 27\sigma_i a_{i-1} + 9a_i - 22 a_{i-1} - \frac{7a_i}{2\sigma_i} \\
& + \frac{7 a_{i-1}}{2\sigma_i} \Big) m_{2,i-1} + \Big(3\sigma_i a_{i-1} + a_i - 4a_{i-1} - \frac{3a_i}{2\sigma_i} + \frac{3a_{i-1}}{2\sigma_i} \Big) m_{1,i-1} \\
& + \Big(-\frac{a_i}{2\sigma_i} + \frac{a_{i-1}}{2\sigma_i} - a_{i-1} \Big) m_{0,i-1} \Big) \Delta t \\
& + 6(\sigma_i - 1)^2 m_{2,i-1} - 2(\sigma_i - 1) m_{1,i-1} + m_{0,i-1},
\end{aligned}
\tag{5.24}
$$

the second coefficient $m^{n+1}_{1,i,ana,L}$ is given by

$$
\begin{aligned}
m^{n+1}_{1,i,ana,L} = \Big(\Big(& -\frac{3a_i^2}{\sigma_i} + \frac{a_i^2}{2\sigma_i^2} + \frac{a_{i-1}^2}{4\sigma_i^2} - \frac{63 a_i a_{i-1}}{2} - \frac{11 a_{i-1}^2}{\sigma_i} - 25\sigma_i a_{i-1}^2 \\
& + 48 a_{i-1}^2 + 3a_i^2 + \frac{14 a_i a_{i-1}}{\sigma_i} - \frac{3 a_i a_{i-1}}{4\sigma_i^2} + 6\sigma_i a_i a_{i-1} \Big) m_{2,i-1} + \Big(\frac{a_i^2}{2\sigma_i^2} \\
& + \frac{a_{i-1}^2}{4\sigma_i^2} - \frac{a_i^2}{\sigma_i} - \frac{4a_{i-1}^2}{\sigma_i} - \frac{3 a_i a_{i-1}}{2} + \frac{7 a_{i-1}^2}{2} + \frac{5 a_i a_{i-1}}{\sigma_i} - \frac{3 a_i a_{i-1}}{4\sigma_i^2} \Big) m_{1,i-1} \\
& + \Big(\frac{a_i^2}{2\sigma_i^2} + \frac{a_{i-1}^2}{4\sigma_i^2} - \frac{a_{i-1}^2}{2\sigma_i} - \frac{3 a_i a_{i-1}}{4\sigma_i^2} + \frac{a_i a_{i-1}}{2\sigma_i} \Big) m_{0,i-1} \Big) \Delta t^2 \\
& + \Big(\Big(-\frac{5a_i}{\sigma_i} + \frac{5 a_{i-1}}{\sigma_i} - 3\sigma_i a_i + 18\sigma_i a_{i-1} + 12 a_i - 27 a_{i-1} \Big) m_{2,i-1} \\
& + \Big(-\frac{2a_i}{\sigma_i} + \frac{2a_{i-1}}{\sigma_i} - 3a_{i-1} + a_i \Big) m_{1,i-1} + \Big(-\frac{a_i}{2\sigma_i} + \frac{a_{i-1}}{2\sigma_i} \Big) m_{0,i-1} \Big) \Delta t \\
& - 6(\sigma_i - 1) m_{2,i-1} + m_{1,i-1},
\end{aligned}
\tag{5.25}
$$

and the third coefficient $m^{n+1}_{2,i,ana,L}$ for the left interval of the ith cell is

$$
\begin{aligned}
m^{n+1}_{2,i,ana,L} = \Big(\Big(& -\frac{a_i^2}{\sigma_i} + \frac{a_i^2}{6\sigma_i^2} + \frac{a_{i-1}^2}{12\sigma_i^2} - 9 a_i a_{i-1} - \frac{7 a_{i-1}^2}{2\sigma_i} + a_i^2 + \frac{25 a_{i-1}^2}{2} \\
& - \frac{a_i a_{i-1}}{4\sigma_i^2} + \frac{9 a_i a_{i-1}}{2\sigma_i} \Big) m_{2,i-1} + \Big(\frac{a_i^2}{6\sigma_i^2} + \frac{a_{i-1}^2}{12\sigma_i^2} - \frac{a_i^2}{3\sigma_i} - \frac{7 a_{i-1}^2}{6\sigma_i} - \frac{a_i a_{i-1}}{4\sigma_i^2} \\
& + \frac{3 a_i a_{i-1}}{2\sigma_i} \Big) m_{1,i-1} + \Big(\frac{a_i^2}{6\sigma_i^2} + \frac{a_{i-1}^2}{12\sigma_i^2} - \frac{a_i a_{i-1}}{4\sigma_i^2} \Big) m_{0,i-1} \Big) \Delta t^2 \\
& + \Big(\Big(-\frac{3a_i}{2\sigma_i} + \frac{3 a_{i-1}}{2\sigma_i} + 3a_i - 6a_{i-1} \Big) m_{2,i-1} \\
& + \Big(-\frac{a_i}{2\sigma_i} + \frac{a_{i-1}}{2\sigma_i} \Big) m_{1,i-1} \Big) \Delta t + m_{2,i-1}.
\end{aligned}
\tag{5.26}
$$

The index *ana* indicates the reference to the analytical solution. The index L refers to the left interval of the cell $[0, \varphi(x_{i-1/2}, t^n, \Delta t)]$.

We have explicitly derived the solution and approximated it by Taylor expansions for small Δx or small Δt. The resulting coefficients of the polynomial are given for the left and right part of the ith grid cell.

5.1.2 Numerical solution after one time step

As described in Chapter 3 the numerical method uses the analytical solution of the linear advection equation and computes the projection of it. Therefore, the numerical solution consists of the coefficients $m_{j,i}^{n+1}$ for the ith grid cell and for $j = 0, 1, 2$. These coefficients $m_{j,i}^{n+1}$ have to be calculated at time t^{n+1} and are given after the projection step by (3.138), i.e.

$$\begin{aligned}&\mathbb{P}\rho(x, t^{n+1}) \\ &= \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{\varphi(x_{i-1/2}, t^{n+1}, -\Delta t)}^{\varphi(x_{i+1/2}, t^{n+1}, -\Delta t)} \rho(x, t^n) K_j(\varphi(x, t^n, \Delta t))\, \mathrm{d}x\ K_j(x).\end{aligned} \tag{5.27}$$

The jth coefficient for the interval $[x_{i-1/2}, x_{i+1/2}]$ at the time step t^{n+1} is given by

$$m_{j,i}^{n+1} = \frac{2j+1}{\Delta x_i} \int_{\varphi(x_{i-1/2}, t^{n+1}, -\Delta t)}^{\varphi(x_{i+1/2}, t^{n+1}, -\Delta t)} \rho(x, t^n) K_j(\varphi(x, t^n, \Delta t))\, \mathrm{d}x\,. \tag{5.28}$$

The trajectories that define the limits of the integral as well as the trajectory in the integrand, that need to be specified in the following, contain expressions as the exponential or logarithmic function. A series expansion of these terms make the numerical solution comparable to the analytical solution.

The trajectory φ for the compution of the limits of the integral is rewritten using (5.2) similarly as in the previous section,

$$\varphi(x_{i+1/2}, t^{n+1}, -\Delta t) = \exp(-a_i \Delta t) \left(\frac{b_i}{a_i} + \Delta x \right) - \frac{b_i}{a_i} \tag{5.29}$$

$$= \frac{\sigma_i \Delta x_i}{a_i \Delta t} \left(\exp(-a_i \Delta t) - 1 \right) + \exp(-a_i \Delta t) \Delta x. \tag{5.30}$$

This form of the trajectory is expanded about the point $\Delta t = 0$ and yields the result for the upper limit of the integral,

$$\begin{aligned}\varphi(x_{i+1/2}, t^{n+1}, -\Delta t) = &\left(\frac{\Delta x\, {a_i}^2}{2} - \frac{\sigma_i\, \Delta x\, {a_i}^2}{6} \right) \Delta t^2 \\ &+ \left(\frac{\sigma_i\, \Delta x\, a_i}{2} - \Delta x\, a_i \right) \Delta t + \Delta x - \sigma_i\, \Delta x + \mathcal{O}\left(\Delta t^3 \right).\end{aligned} \tag{5.31}$$

The same is done for the lower limit of the integral with according changes. The coefficients b_{i-1} are replaced using the relation (5.2) and (5.4). We have,

$$\varphi(x_{i-1/2}, t^{n+1}, -\Delta t) = \exp(-a_{i-1} \Delta t) \left(\frac{b_{i-1}}{a_{i-1}} + \Delta x \right) - \frac{b_{i-1}}{a_{i-1}} \tag{5.32}$$

$$= \frac{\sigma_i \Delta x}{a_{i-1} \Delta t} \left(\exp(-a_{i-1} \Delta t) - 1 \right) + \Delta x. \tag{5.33}$$

We obtain the result for the lower limit of the integral by the Taylor expansion about $\Delta t = 0$,

$$\begin{aligned}\varphi(x_{i-1/2}, t^{n+1}, -\Delta t) = &\frac{\sigma_i\, \Delta x\, \Delta t^3\, {a_{i-1}}^3}{24} - \frac{\sigma_i\, \Delta x\, \Delta t^2\, {a_{i-1}}^2}{6} \\ &+ \frac{\sigma_i\, \Delta x\, \Delta t\, a_{i-1}}{2} + \Delta x - \sigma_i\, \Delta x.\end{aligned} \tag{5.34}$$

The trajectory φ, which remains within the ith grid cell, goes forward in time, and starts at an arbitrary point x, takes the form

$$\varphi(x, t^{n+1}, \Delta t) = \exp(a_i \Delta t) \left(\frac{b_i}{a_i} + x \right) - \frac{b_i}{a_i} \tag{5.35}$$

$$= \frac{\sigma_i \Delta x}{a_i \Delta t} \left(\exp(a_i \Delta t) - 1 \right) + \exp(a_i \Delta t) x. \tag{5.36}$$

Series expansion about $\Delta t = 0$ leads to

$$\begin{aligned} \varphi(x, t^{n+1}, \Delta t) = &\left(\frac{x\, {a_i}^2}{2} + \frac{\sigma_i\, \Delta x\, {a_i}^2}{6} \right) \Delta t^2 + \left(x\, a_i + \frac{\sigma_i\, \Delta x\, a_i}{2} \right) \Delta t \\ &+ x + \sigma_i\, \Delta x + \mathcal{O}\left(\Delta t^3 \right). \end{aligned} \tag{5.37}$$

The trajectory departing in cell $i-1$ and ending in the ith cell is described by

$$\varphi(x, t^n, \Delta t) = \frac{b_i}{a_i} \left(\exp(a_i \Delta t) \left(\frac{a_{i-1}}{b_i} x + \frac{b_{i-1}}{b_i} \right)^{\frac{a_i}{a_{i-1}}} - 1 \right). \tag{5.38}$$

To estimate the behavior for small Δx we start analyzsing and rewriting the term with (5.2) and (5.4),

$$\left(\frac{a_{i-1}}{b_i} x + \frac{b_{i-1}}{b_i} \right)^{\frac{a_i}{a_{i-1}}} = \exp\left(\frac{a_i}{a_{i-1}} \ln\left(\frac{a_{i-1} \Delta t}{\sigma_i \Delta x} x + \frac{\sigma_{i-1}}{\sigma_i} \right) \right) \tag{5.39}$$

$$= \exp\left(\frac{a_i}{a_{i-1}} \ln\left(\frac{a_{i-1} \Delta t}{\sigma_i \Delta x} x - \frac{a_{i-1} \Delta t}{\sigma_i} + 1 \right) \right) \tag{5.40}$$

$$= \exp\left(\frac{a_i}{a_{i-1}} \ln\left(\Delta t \left(\frac{a_{i-1}}{\sigma_i \Delta x} x - \frac{a_{i-1}}{\sigma_i} + 1 \right) \right) \right). \tag{5.41}$$

A series expansion of this reformulated expression ends in polynomial structure. Thus, the Taylor expansion of (5.38) using the formulation of (5.41) leads to

$$\begin{aligned} \varphi(x, t^{n+1}, \Delta t) = &\left(-\frac{x^2 {a_{i-1}}^2}{\sigma_i^2 \Delta x} + \frac{x^2 a_i^2}{2\sigma_i \Delta x} - \frac{x^2 a_i^2}{2\sigma_i^2 \Delta x} - \frac{x^2 a_{i-1} a_i}{2\sigma_i \Delta x} + \frac{3x^2 a_{i-1} a_i}{2\sigma_i^2 \Delta x} \right. \\ &+ \frac{x a_{i-1} a_i}{\sigma_i} - \frac{3x a_{i-1} a_i}{2\sigma_i^2} + \frac{x a_i^2}{2} - \frac{x a_i^2}{\sigma_i} + \frac{x a_i^2}{2\sigma_i^2} + \frac{x {a_{i-1}}^2}{\sigma_i^2} \Bigg) \Delta t^2 \\ &+ \left(\frac{x^2 a_i}{2\sigma_i \Delta x} - \frac{x^2 a_{i-1}}{2\sigma_i \Delta x} + x a_i - \frac{x a_i}{\sigma_i} + \frac{x a_{i-1}}{\sigma_i} - \Delta x a_i + \frac{\Delta x a_i}{2\sigma_i} \right. \\ &\left. - \frac{\Delta x a_{i-1}}{2\sigma_i} + \frac{\sigma_i \Delta x a_i}{2} \right) \Delta t + x - \Delta x + \sigma_i \Delta x + \mathcal{O}\left(\Delta t^3 \right). \end{aligned} \tag{5.42}$$

Inserting all approximations for the limits of the integral and the trajectories going forward in time into (5.28) yield the approximated numerical solution for the coefficients $m_{0,i}^{n+1}$, $m_{1,i}^{n+1}$ and $m_{2,i}^{n+1}$.

When the superindex n in $m_{j,i}^n$ is omitted, the coefficients refers to the old time level n. The first coefficient $m_{0,i}^{n+1}$ is given by

$$\begin{aligned} m_{0,i}^{n+1} = &\left(\left(\frac{\sigma_i\, {a_{i-1}}^2}{6} \right) \Delta t^2 + \left(-\frac{\sigma_i\, a_{i-1}}{2} \right) \Delta t + \sigma_i \right) m_{0,i-1} \\ &+ \left(\left(\frac{{a_i}^2}{2} - \frac{\sigma_i\, {a_i}^2}{6} \right) \Delta t^2 + \left(\frac{\sigma_i\, a_i}{2} - a_i \right) \Delta t + 1 - \sigma_i \right) m_{0,i} \\ &+ \left(\left({\sigma_i}^2\, a_{i-1} - \frac{\sigma_i\, a_{i-1}}{2} \right) \Delta t + \sigma_i - {\sigma_i}^2 \right) m_{1,i-1} \\ &+ \left(\left(-a_i\, {\sigma_i}^2 + \frac{5\, a_i\, \sigma_i}{2} - a_i \right) \Delta t + {\sigma_i}^2 - \sigma_i \right) m_{1,i} \\ &+ \left(2\, {\sigma_i}^3 - 3\, {\sigma_i}^2 + \sigma_i \right) m_{2,i-1} \\ &+ \left(-2\, {\sigma_i}^3 + 3\, {\sigma_i}^2 - \sigma_i \right) m_{2,i} + \mathcal{O}\left(\Delta x^3 \right). \end{aligned} \tag{5.43}$$

The second coefficient $m_{1,i}^{n+1}$ reads

$$\begin{aligned} m_{1,i}^{n+1} = & \left(\left(\frac{\sigma_i^2 \, a_{i-1}^2}{4} - \frac{\sigma_i^2 \, a_{i-1} \, a_i}{4} + \frac{\sigma_i^2 \, a_i^2}{4} - \frac{\sigma_i \, a_{i-1}^2}{2} \right) \Delta t^2 \right. \\ & + \left(\frac{3\,\sigma_i \, a_{i-1}}{2} - \sigma_i^2 \, a_{i-1} + \sigma_i^2 \, a_i \right) \Delta t + 3\,\sigma_i^2 - 3\,\sigma_i \Big) \, m_{0,i-1} \\ & + \left(\left(\frac{\sigma_i \, a_i^2}{2} - \frac{\sigma_i^2 \, a_i^2}{4} \right) \Delta t^2 + \left(-\frac{3\,\sigma_i \, a_i}{2} \right) \Delta t + 3\,\sigma_i - 3\,\sigma_i^2 \right) m_{0,i} \\ & + \left(\left(\frac{3\,\sigma_i^3 \, a_{i-1}}{2} - 4\,\sigma_i^2 \, a_{i-1} + \frac{3\,\sigma_i \, a_{i-1}}{2} + \sigma_i^2 \, a_i - \frac{\sigma_i^3 \, a_i}{2} \right) \Delta t \right. \\ & \left. -2\,\sigma_i^3 + 6\,\sigma_i^2 - 3\,\sigma_i \right) m_{1,i-1} \\ & + \left(\left(-a_i \, \sigma_i^3 + 3\,a_i \, \sigma_i^2 + \frac{3\,a_i \, \sigma_i}{2} - 2\,a_i \right) \Delta t + 2\,\sigma_i^3 - 3\,\sigma_i + 1 \right) m_{1,i} \\ & + \left(3\,\sigma_i^4 - 12\,\sigma_i^3 + 12\,\sigma_i^2 - 3\,\sigma_i \right) m_{2,i-1} \\ & + \left(-3\,\sigma_i^4 + 6\,\sigma_i^2 - 3\,\sigma_i \right) m_{2,i} + \mathcal{O}\left(\Delta x^3 \right). \end{aligned} \tag{5.44}$$

The third coefficient $m_{2,i}^{n+1}$ yields

$$\begin{aligned} m_{2,i}^{n+1} = & \left(\left(\frac{\sigma_i^3 a_{i-1}^2}{2} - \frac{3\sigma_i^3 a_{i-1} a_i}{2} + \frac{7\sigma_i^3 a_i^2}{2} - \frac{5\sigma_i^2 a_{i-1}^2}{4} \right. \right. \\ & \left. + \frac{5\sigma_i^2 a_{i-1} a_i}{4} - \frac{5\sigma_i^2 a_i^2}{4} + \frac{5\sigma_i a_{i-1}^2}{6} \right) \Delta t^2 + \left(5\sigma_i^2 a_{i-1} - \frac{5\sigma_i^3 a_{i-1}}{2} \right. \\ & \left. - \frac{5\sigma_i a_{i-1}}{2} - 5\sigma_i^2 a_i + \frac{15\sigma_i^3 a_i}{2} \right) \Delta t + 10\sigma_i^3 - 15\sigma_i^2 + 5\sigma_i \Big) \, m_{0,i-1} \\ & + \left(\left(-\frac{5\sigma_i^3 a_i^2}{2} + \frac{5\sigma_i^2 a_i^2}{4} - \frac{5\sigma_i a_i^2}{6} \right) \Delta t^2 + \left(\frac{5\sigma_i a_i}{2} - 5\sigma_i^3 a_i \right) \Delta t \right. \\ & \left. -10\sigma_i^3 + 15\sigma_i^2 - 5\sigma_i \right) m_{0,i} \\ & + \left(\left(10\sigma_i^2 a_{i-1} - 10\sigma_i^3 a_{i-1} + 3\sigma_i^4 a_{i-1} - \frac{5\sigma_i a_{i-1}}{2} - 5\sigma_i^2 a_i \right. \right. \\ & \left. \left. +10\sigma_i^3 a_i - 3\sigma_i^4 a_i \right) \Delta t - 5\sigma_i^4 + 20\sigma_i^3 - 20\sigma_i^2 + 5\sigma_i \right) m_{1,i-1} \\ & + \left(\left(10 a_i \sigma_i^3 - 5 a_i \sigma_i^2 - \frac{5 a_i \sigma_i}{2} \right) \Delta t + 5\sigma_i^4 - 10\sigma_i^2 + 5\sigma_i \right) m_{1,i} \\ & + \left(6\sigma_i^5 - 30\sigma_i^4 + 50\sigma_i^3 - 30\sigma_i^2 + 5\sigma_i \right) m_{2,i-1} \\ & + \left(-6\sigma_i^5 + 10\sigma_i^3 - 5\sigma_i + 1 \right) m_{2,i} + \mathcal{O}\left(\Delta x^3 \right). \end{aligned} \tag{5.45}$$

Above, we have approximated the numerical solution, which is determined by the approximative coefficients $m_{0,i}^{n+1}$, $m_{1,i}^{n+1}$ and $m_{2,i}^{n+1}$ given by (5.43) to (5.45).

5.1.3 Order of coefficients

Studying the coefficients $m_{j,i}^{n+1}$ for $j = 0, 1, 2$ of the analytical and the numerical solution we notice that the order of the coefficients and their differences for small Δx and Δt are of importance to find the order of accuracy. Therefore, in this section we determine the order of the coefficients $m_{j,i}^{n+1}$. This is carried out by induction. For the induction basis we show the order for small Δx for the initial values, i.e. at

time t^n with $n = 0$. The inductive step reveals, that assuming a certain order of the coefficients after n time steps at time t^n, the same order holds after one more time step at time t^{n+1} as well.

Induction basis n=0

We assume a smooth function $f \in H^s, s \geq 3$, H^s being a Sobolev space, that is approximated by a polynomial for the initial values. We use a Taylor expansion to approximate f in two neighboring cells, i.e. the $(i-1)$th and ith cell, at their common grid cell boundary $x_{i-1/2}$. We obtain the approximation f_i to f for the ith grid by an expansion about $x = x_{i-1/2}$, which is the left boundary and thus $x = 0$ in local coordinates,

$$f_i(x) = \sum_{j=0}^{3} \frac{f^{(j)}(0)}{j!} x^j + \mathcal{O}\left(\Delta x_i^4\right) \tag{5.46}$$

$$= f_i(0) + f_i^{(1)}(0)x + \frac{1}{2} f_i^{(2)}(0)x^2 + \frac{1}{6} f_i^{(3)}(0)x^3 + \mathcal{O}\left(\Delta x^4\right). \tag{5.47}$$

Analogously, we have a representation of f_{i-1} that approximates f in grid cell $i-1$. It is determined by an expansion about its right boundary $x = x_{i-1/2}$ or $x = \Delta x$ expressed in local coordinates,

$$f_{i-1}(x) = \sum_{j=0}^{3} \frac{f^{(j)}(\Delta x)}{j!} (x - \Delta x)^j + \mathcal{O}\left(\Delta x_{i-1}^4\right) \tag{5.48}$$

$$\begin{aligned} &= \frac{1}{6} f_{i-1}^{(3)}(\Delta x)x^3 + \left(\frac{1}{2} f_{i-1}^{(2)}(\Delta x) - \frac{1}{2} f_{i-1}^{(3)}(\Delta x)\Delta x\right) x^2 \\ &\quad + \left(\frac{1}{2} f_{i-1}^{(3)}(\Delta x)\Delta x^2 - f_{i-1}^{(2)}(\Delta x)\Delta x + f_{i-1}^{(1)}(\Delta x)\right) x \\ &\quad - \frac{1}{6} f_{i-1}^{(3)}(\Delta x)\Delta x^3 + \frac{1}{2} f_{i-1}^{(2)}(\Delta x)\Delta x^2 - f_{i-1}^{(1)}(\Delta x)\Delta x \\ &\quad + f_{i-1}(\Delta x) + \mathcal{O}\left(\Delta x^4\right). \end{aligned} \tag{5.49}$$

Figure 5.1 displays the function f, which is exemplarily chosen to be a polynomial of degree five, and two approximating polynomials f_{i-1} and f_i of degree two, obtained by Taylor expansion about the point $x_{i-1/2}$.

The very first step of the numerical method is the projection step. It is carried out before any time stepping is done and enables an arbitrary function to be transformed into polynomial structure as initial values. The projection of a polynomial function as f_i and f_{i-1} equals a change of basis - changing from the monomial basis to the Legendre basis.

The polynomial with the Legendre basis as defined in (3.131) takes the form

$$\rho(x, t^n) = m_{2,i}^n K_2(x) + m_{1,i}^n K_1(x) + m_{0,i}^n K_0(x). \tag{5.50}$$

The transformed coefficients of f_i for cell i are given by

$$m_{0,i}^0 = \frac{f_i^{(2)}(0)\,\Delta x^2}{6} + \frac{f_i^{(1)}(0)\,\Delta x}{2} + f_i(0), \tag{5.51}$$

$$m_{1,i}^0 = \frac{f_i^{(2)}(0)\,\Delta x^2}{4} + \frac{f_i^{(1)}(0)\,\Delta x}{2}, \tag{5.52}$$

$$m_{2,i}^0 = \frac{f_i^{(2)}(0)\,\Delta x^2}{12}, \tag{5.53}$$

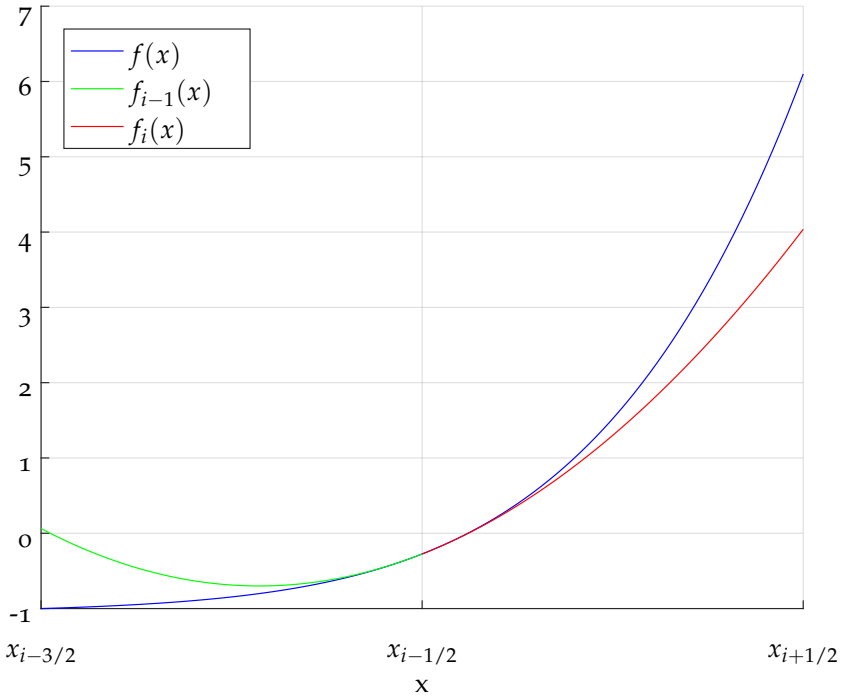


Figure 5.1: The functions f_i and f_{i-1} approximate f by Taylor expansions about the point $x_{i-1/2}$.

and of f_{i-1} for cell $i-1$,

$$m^0_{0,i-1} = -\frac{f^{(3)}_{i-1}(\Delta x)\,\Delta x^3}{12} + \frac{f^{(2)}_{i-1}(\Delta x)\,\Delta x^2}{6} - \frac{f^{(1)}_{i-1}(\Delta x)\,\Delta x}{2} + f_{i-1}(\Delta x), \tag{5.54}$$

$$m^0_{1,i-1} = \frac{f^{(1)}_{i-1}(\Delta x)\,\Delta x}{2} - \frac{f^{(2)}_{i-1}(\Delta x)\,\Delta x^2}{4}, \tag{5.55}$$

$$m^0_{2,i-1} = \frac{f^{(2)}_{i-1}(\Delta x)\,\Delta x^2}{12} - \frac{f^{(3)}_{i-1}(\Delta x)\,\Delta x^3}{12}, \tag{5.56}$$

respectively. The order of the coefficients $m^0_{j,k}$ for small Δx can be read off the above expressions (5.51) - (5.56). Thus, the coefficients $m^0_{j,k}$ for $j = 0, 1, 2$ and $k \in \{i, i-1\}$ have the following order

$$m^0_{0,k} = \mathcal{O}(1), \tag{5.57}$$

$$m^0_{1,k} = \mathcal{O}(\Delta x), \tag{5.58}$$

$$m^0_{2,k} = \mathcal{O}\left(\Delta x^2\right). \tag{5.59}$$

As $f \in H^s$, the approximations $f_i(0)$ and $f_{i-1}(\Delta x)$ are equal and their derivatives evaluated at $x_{i-1/2}$ as well,

$$f^{(j)}_{i-1/2} := f^{(j)}(x_{i-1/2}) = f^{(j)}_i(0) = f^{(j)}_{i-1}(\Delta x), \tag{5.60}$$

for $j = 1, 2$. That enables us to analyze the order of differences of the coefficients. We obtain,

$$m^0_{0,i} - m^0_{0,i-1} = f^{(1)}_{i-1/2}\Delta x + \frac{1}{12} f^{(3)}_{i-1/2}\Delta x^3 = \mathcal{O}\left(\Delta x\right), \tag{5.61}$$

$$m^0_{1,i} - m^0_{1,i-1} = \frac{1}{2} f^{(2)}_{i-1/2}\Delta x^2 = \mathcal{O}\left(\Delta x^2\right), \tag{5.62}$$

$$m^0_{2,i} - m^0_{2,i-1} = \frac{1}{12} f^{(3)}_{i-1/2}\Delta x^3 = \mathcal{O}\left(\Delta x^3\right), \tag{5.63}$$

$$m^0_{0,i} - m^0_{0,i-1} - 2m^0_{1,i} = -\frac{1}{2} f^{(2)}_{i-1/2}\Delta x^2 + \frac{1}{12} f^{(3)}_{i-1/2}\Delta x^3 = \mathcal{O}\left(\Delta x^2\right), \tag{5.64}$$

$$m^0_{0,i} - m^0_{0,i-1} - 2m^0_{1,i} + 6m^0_{2,i} = \frac{1}{12} f^{(3)}_{i-1/2}\Delta x^3 = \mathcal{O}\left(\Delta x^3\right), \tag{5.65}$$

$$m^0_{0,i} - m^0_{0,i-1} - 2m^0_{1,i-1} - 6m^0_{2,i-1} = \frac{7}{12} f^{(3)}_{i-1/2}\Delta x^3 = \mathcal{O}\left(\Delta x^3\right), \tag{5.66}$$

$$m^0_{1,i} - m^0_{1,i-1} - 6m^0_{2,i} = \mathcal{O}\left(\Delta x^3\right). \tag{5.67}$$

As the basis of the induction at time $t = 0$, at time step $n = 0$, we found the order of the coefficients $m^0_{j,k}$ and of the correlation between the coefficients.

Induction hypothesis

The induction hypothesis assumes that the behavior of the coefficients for small Δx and their correlations are still true after n time steps are made. We have,

$$m^n_{0,i} = \mathcal{O}\left(1\right), \tag{5.68}$$

$$m^n_{1,i} = \mathcal{O}\left(\Delta x\right), \tag{5.69}$$

$$m^n_{2,i} = \mathcal{O}\left(\Delta x^2\right), \tag{5.70}$$

$$m^n_{0,i} - m^n_{0,i-1} = \mathcal{O}\left(\Delta x\right), \tag{5.71}$$

$$m^n_{1,i} - m^n_{1,i-1} = \mathcal{O}\left(\Delta x^2\right), \tag{5.72}$$

$$m^n_{2,i} - m^n_{2,i-1} = \mathcal{O}\left(\Delta x^3\right), \tag{5.73}$$

$$m^n_{0,i} - m^n_{0,i-1} - 2m^n_{1,i} = \mathcal{O}\left(\Delta x^2\right), \tag{5.74}$$

$$m^n_{0,i} - m^n_{0,i-1} - 2m^n_{1,i} + 6m^n_{2,i} = \mathcal{O}\left(\Delta x^3\right), \tag{5.75}$$

$$m^n_{0,i} - m^n_{0,i-1} - 2m^n_{1,i-1} - 6m^n_{2,i-1} = \mathcal{O}\left(\Delta x^3\right), \tag{5.76}$$

$$m^n_{1,i} - m^n_{1,i-1} - 6m^n_{2,i} = \mathcal{O}\left(\Delta x^3\right). \tag{5.77}$$

Inductive step

With the assumptions made in the induction hypothesis, we show in the inductive step, that the assumptions hold after one more time step is carried out with the numerical method, i.e.

$$m_{0,i}^{n+1} = \mathcal{O}(1), \tag{5.78}$$
$$m_{1,i}^{n+1} = \mathcal{O}(\Delta x), \tag{5.79}$$
$$m_{2,i}^{n+1} = \mathcal{O}\left(\Delta x^2\right), \tag{5.80}$$
$$m_{0,i}^{n+1} - m_{0,i-1}^{n+1} = \mathcal{O}(\Delta x), \tag{5.81}$$
$$m_{1,i}^{n+1} - m_{1,i-1}^{n+1} = \mathcal{O}\left(\Delta x^2\right), \tag{5.82}$$
$$m_{2,i}^{n+1} - m_{2,i-1}^{n+1} = \mathcal{O}\left(\Delta x^3\right), \tag{5.83}$$
$$m_{0,i}^{n+1} - m_{0,i-1}^{n+1} - 2m_{1,i}^{n+1} = \mathcal{O}\left(\Delta x^2\right), \tag{5.84}$$
$$m_{0,i}^{n+1} - m_{0,i-1}^{n+1} - 2m_{1,i}^{n+1} + 6m_{2,i}^{n+1} = \mathcal{O}\left(\Delta x^3\right), \tag{5.85}$$
$$m_{0,i}^{n+1} - m_{0,i-1}^{n+1} - 2m_{1,i-1}^{n+1} - 6m_{2,i-1}^{n+1} = \mathcal{O}\left(\Delta x^3\right), \tag{5.86}$$
$$m_{1,i}^{n+1} - m_{1,i-1}^{n+1} - 6m_{2,i}^{n+1} = \mathcal{O}\left(\Delta x^3\right). \tag{5.87}$$

The numerical solution is determined for the coefficients $m_{j,i}^{n+1}$ in section 5.1.2 and given by (5.43) for the first coefficient $m_{0,i}^{n+1}$, by (5.44) for the second coefficient $m_{1,i}^{n+1}$ and the solution for the third $m_{2,i}^{n+1}$ coefficient is written in (5.45). The coefficients for the neigboring grid cell $i-1$ are equal apart from the shift of all indices that refer to the grid cell.

Some algebra and the exploitation of the induction hypothesis lead to and confirm the order of the coefficients and the order of the correlations of the coefficients.

5.1.4 Order of errors

To determine the error and the order of the error for small Δx and Δt we compute the difference of the approximated analytical solution and the approximated numerical solution.

Then, the error is explicitly given for the left part of the cell $[0, \varphi(0, t^n, \Delta t)]$ by

$$|m_{j,i,ana,L}^{n+1} - m_{0,i}^{n+1}| \tag{5.88}$$

and for the right part of the cell $[\varphi(0, t^n, \Delta t), \Delta x]$

$$|m_{j,i,ana,R}^{n+1} - m_{0,i}^{n+1}|. \tag{5.89}$$

These differences are computed and examined. The next section studies the error for constant velocity. The subsequent section generalizes the result to a non-constant, however smooth velocity field.

Constant velocity

The coefficients of the analytical and the numerical solution simplify in the case of constant velocity. The variables a_i and a_{i-1}, which indicate the slope of the velocity, are set to zero.

To obtain the analytical solution for the left interval $[0, \varphi(0, t^n, \Delta t)]$ of the ith grid cell for constant velocity, we set a_i and a_{i-1} to zero in (5.24) to (5.26). We have,

$$\begin{aligned} m^{n+1}_{0,i,ana,L} &= 6\sigma^2 m_{2,i-1} + m_{0,i-1} + 2m_{1,i-1} + 6m_{2,i-1} - 2\sigma m_{1,i-1} \\ &\quad -12\sigma m_{2,i-1}, \end{aligned} \tag{5.90}$$

$$m^{n+1}_{1,i,ana,L} = m_{1,i-1} + 6m_{2,i-1} - 6\sigma m_{2,i-1}, \tag{5.91}$$

$$m^{n+1}_{2,i,ana,L} = m_{2,i-1}. \tag{5.92}$$

Analogously, the coefficients $m^{n+1}_{j,i,ana,R}$ for $j = 0, 1, 2$ given by (5.13) to (5.15) take the form

$$m^{n+1}_{0,i,ana,R} = 6m_{2,i}\sigma^2 - 2m_{1,i}\sigma + m_{0,i}, \tag{5.93}$$

$$m^{n+1}_{1,i,ana,R} = m_{1,i} - 6\sigma m_{2,i}, \tag{5.94}$$

$$m^{n+1}_{2,i,ana,R} = m_{2,i} \tag{5.95}$$

for constant velocity.

The numerical solution simplifies as well for constant velocity. We list the coefficients $m^{n+1}_{j,i}$ below, first with the variables a_i and a_{i-1} set to zero and then in a second version, with rearranged terms that indicate the benefit of the induction hypothesis, i.e. the correlation between the coefficients. The first coefficient $m^{n+1}_{0,i}$ is given by

$$\begin{aligned} m^{n+1}_{0,i} &= \sigma m_{0,i-1} + (1-\sigma)\, m_{0,i} \\ &\quad + \left(\sigma - \sigma^2\right) m_{1,i-1} + \left(\sigma^2 - \sigma\right) m_{1,i} \\ &\quad + \left(2\,\sigma^3 - 3\,\sigma^2 + \sigma\right) m_{2,i-1} \\ &\quad + \left(-2\,\sigma^3 + 3\,\sigma^2 - \sigma\right) m_{2,i} + \mathcal{O}\left(\Delta x^3\right) \end{aligned} \tag{5.96}$$

$$\begin{aligned} &= -\sigma(m_{0,i} - m_{0,i-1} - 2m_{1,i} + 6m_{2,i}) \\ &\quad + \left(-\sigma^2 + \sigma\right)(m_{1,i} - m_{1,i-1} - 6m_{2,i}) \\ &\quad + \left(-2\,\sigma^3 + 3\,\sigma^2 - \sigma\right)(m_{2,i} - m_{2,i-1}) \\ &\quad + m_{0,i} - 2\sigma m_{1,i} + 6\sigma^2 m_{2,i} + \mathcal{O}\left(\Delta x^3\right). \end{aligned} \tag{5.97}$$

The second coefficient $m^{n+1}_{1,i}$ of the numerical solution reads

$$\begin{aligned} m^{n+1}_{1,i} &= \left(3\,\sigma^2 - 3\,\sigma\right) m_{0,i-1} + \left(3\,\sigma - 3\,\sigma^2\right) m_{0,i} \\ &\quad + \left(-2\,\sigma^3 + 6\,\sigma^2 - 3\,\sigma\right) m_{1,i-1} + \left(2\,\sigma^3 - 3\,\sigma + 1\right) m_{1,i} \\ &\quad + \left(3\,\sigma^4 - 12\,\sigma^3 + 12\,\sigma^2 - 3\,\sigma\right) m_{2,i-1} \\ &\quad + \left(-3\,\sigma^4 + 6\,\sigma^2 - 3\,\sigma\right) m_{2,i} + \mathcal{O}\left(\Delta x^3\right) \end{aligned} \tag{5.98}$$

$$\begin{aligned} &= \left(3\,\sigma - 3\,\sigma^2\right)(m_{0,i} - m_{0,i-1} - 2m_{1,i} + 6m_{2,i}) \\ &\quad + \left(2\,\sigma^3 - 6\,\sigma^2 + 3\,\sigma\right)(m_{1,i} - m_{1,i-1} - 6m_{2,i}) \\ &\quad + \left(-3\,\sigma^4 + 12\,\sigma^3 - 12\,\sigma^2 + 3\,\sigma\right)(m_{2,i} - m_{2,i-1}) \\ &\quad + m_{1,i} - 6\sigma m_{2,i} + \mathcal{O}\left(\Delta x^3\right), \end{aligned} \tag{5.99}$$

and the third one $m_{2,i}^{n+1}$ yields

$$\begin{aligned} m_{2,i}^{n+1} = & \left(10\sigma^3 - 15\sigma^2 + 5\sigma\right) m_{0,i-1} \\ & + \left(-10\sigma^3 + 15\sigma^2 - 5\sigma\right) m_{0,i} \\ & + \left(-5\sigma^4 + 20\sigma^3 - 20\sigma^2 + 5\sigma\right) m_{1,i-1} \\ & + \left(5\sigma^4 - 10\sigma^2 + 5\sigma\right) m_{1,i} \\ & + \left(6\sigma^5 - 30\sigma^4 + 50\sigma^3 - 30\sigma^2 + 5\sigma\right) m_{2,i-1} \\ & + \left(-6\sigma^5 + 10\sigma^3 - 5\sigma + 1\right) m_{2,i} + \mathcal{O}\left(\Delta x^3\right) \end{aligned} \tag{5.100}$$

$$\begin{aligned} = & \left(-10\sigma^3 + 15\sigma^2 - 5\sigma\right) (m_{0,i} - m_{0,i-1} - 2m_{1,i} + 6m_{2,i}) \\ & + \left(5\sigma^4 - 10\sigma^2 + 5\sigma\right) (m_{1,i} - m_{1,i-1} - 6m_{2,i}) \\ & + \left(-6\sigma^5 + 30\sigma^4 - 50\sigma^3 + 30\sigma^2 - 5\sigma\right) (m_{2,i} - m_{1,i-1}) \\ & + m_{2,i} + \mathcal{O}\left(\Delta x^3\right). \end{aligned} \tag{5.101}$$

Exemplarily, the computation of the error is shown for the coefficient $m_{0,i}^{n+1}$ in the left interval.

$$|m_{0,i,ana,L}^{n+1} - m_{0,i}^{n+1}| = 6\sigma^2 m_{2,i-1} + m_{0,i-1} + 2m_{1,i-1} + 6m_{2,i-1} \tag{5.102}$$

$$-2\sigma m_{1,i-1} - 12\sigma m_{2,i-1} \tag{5.103}$$

$$- \left(m_{0,i} - 2\sigma m_{1,i} + 6\sigma^2 m_{2,i}\right) + \mathcal{O}\left(\Delta x^3\right) \tag{5.104}$$

$$= - \left(m_{0,i} - m_{0,i-1} - 2m_{1,i-1} - 6m_{2,i-1}\right) \tag{5.105}$$

$$+2\sigma \left(m_{1,i} - m_{1,i-1} - 6m_{2,i-1}\right) + \mathcal{O}\left(\Delta x^3\right) \tag{5.106}$$

$$= \mathcal{O}\left(\Delta x^3\right), \tag{5.107}$$

where expressions that are of order Δx^3 and higher are omitted making use of the induction hypothesis.

Similarly, the errors for the other coefficients of the numerical solution for the left and right interval can be determined and their order derived. Thus, for the left interval $[0, \varphi(0, t^n, \Delta t)]$ we have,

$$|m_{0,i,ana,L}^{n+1} - m_{0,i}^{n+1}| = \mathcal{O}\left(\Delta x^3\right), \tag{5.108}$$

$$|m_{1,i,ana,L}^{n+1} - m_{1,i}^{n+1}| = \mathcal{O}\left(\Delta x^3\right), \tag{5.109}$$

$$|m_{2,i,ana,L}^{n+1} - m_{2,i}^{n+1}| = \mathcal{O}\left(\Delta x^3\right), \tag{5.110}$$

and for the right part of the cell $[\varphi(0, t^n, \Delta t), \Delta x]$ the errors yield

$$|m_{0,i,ana,R}^{n+1} - m_{0,i}^{n+1}| = \mathcal{O}\left(\Delta x^3\right), \tag{5.111}$$

$$|m_{1,i,ana,R}^{n+1} - m_{1,i}^{n+1}| = \mathcal{O}\left(\Delta x^3\right), \tag{5.112}$$

$$|m_{2,i,ana,R}^{n+1} - m_{2,i}^{n+1}| = \mathcal{O}\left(\Delta x^3\right). \tag{5.113}$$

We can conclude that the error of the coefficients $m_{j,i}^{n+1}$ for the left and right interval is of order Δx^3, and therefore the overall error for the polynomial built from these coefficients, representing the numerical solution, is of third order accuracy for constant velocity.

Variable velocity

The procedure for variable velocity is carried out similarly as for the case with constant velocity.

We assume $a_i = a_{i-1} + \mathcal{O}(\Delta x)$ to represent a varying velocity, but with only small changes in the slope. Since $a_i = \frac{\partial}{\partial x} u|_{\Omega_i}(x)$, the condition on a_i requires the existence of a second derivative of u, i.e. $u \in C^2(\Omega)$. We conduct the respective computations as for the constant velocity case shown above allowing a variable velocity u. We find that the numerical solution is second order accurate, i.e. one order of accuracy is lost, when we allow variable velocity.

If we assume the equality $a_i = a_{i-1}$, that are not necessarily equal to zero, we maintain the third order accuracy.

In this section we studied the consistency and the order of accuracy of the numerical method. We examined the explicit error between the analytical and the numerical solution. To make these solutions comparable both were approximated by means of Taylor expansions about Δt and Δx tending to zero. The error was reduced to certain correlations of the coefficients $m^n_{j,i-1}$ and $m^n_{2,i}$ for $j = 0, 1, 2$. The relations between the coefficients and their order were found with the help of an induction. From the order of the coefficients and their correlations the error and its order could be implied. The error goes to zero for small time steps and small grid cells. Therefore, the numerical method is consistent. The order of accuracy is three for constant velocity and two for smooth variable velocity.

5.2 STABILITY

We first show the L^1- and the L^2-stability by finding suitable estimates for the growth of the numerical solution. Further, we apply the von Neumann stability analysis, because it firstly examines the diffusion and dispersion error separately and secondly it gives further insight into the theoretical order of convergence.

5.2.1 L^1-Stability

To show the L^1-stability of the SASLDG method we have to find a bound for the integral of the absolute value of the advected quantity. We separate the problem in two parts. First, we examine how the L^1-norm changes when the given distribution is moved along the trajectories. Then, the effect of the L^2-projection on the L^1-norm is studied.

Part 1: Effect of the exact solution on the L^1-norm

We write the L^2-norm of the exact solution $\rho(x, t^{n+1})$ using (3.124) for the ith cell $[x_{i-1/2}, x_{i+1/2}]$ and show that we can find an estimate for the bound. We have,

$$\left\|\rho(x,t^{n+1})\right\|_{L^1(\Omega_i)} = \int_{x_{i-1/2}}^{x_{i+1/2}} \left|\rho(x,t^{n+1})\right| \mathrm{d}x \tag{5.114}$$

$$\begin{aligned} &= \int_{x_{i-1/2}}^{x_{i+1/2}} \Bigg|\rho(\varphi(x,t^{n+1},-\Delta t),t^n) \\ &\quad \cdot \exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x} u(\varphi(x,t^{n+1},t-\Delta t))dt\right)\Bigg| \mathrm{d}x\,. \end{aligned} \tag{5.115}$$

Then, we use the same substitution as in (3.126), i.e. $x = \varphi(\eta, t^n, \Delta t)$ and the property (3.129). This leads to

$$\begin{aligned}
&\left\|\rho(x, t^{n+1})\right\|_{L^1(\Omega_i)} \\
&= \int_{\varphi(x_{i-1/2}, t^{n+1}, -\Delta t)}^{\varphi(x_{i+1/2}, t^{n+1}, -\Delta t)} \left| \rho(\eta, t^n) \exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x} u(\varphi(\eta, t^n, t)) dt \right) \frac{\partial}{\partial \eta} \varphi(\eta, t^n, \Delta t) \right| \mathrm{d}\eta \qquad (5.116) \\
&= \int_{\varphi(x_{i-1/2}, t^{n+1}, -\Delta t)}^{\varphi(x_{i+1/2}, t^{n+1}, -\Delta t)} |\rho(\eta, t^n)| \, \mathrm{d}\eta \, . \qquad (5.117)
\end{aligned}$$

Hence, the L^1-norm of the exact solution within one grid cell remains the same during one time step. Therefore, the same holds for the whole computational domain. We have,

$$\begin{aligned}
\left\|\rho(x, t^{n+1})\right\|_{L^1(\Omega)} &= \sum_{i=1}^{N} \int_{x_{i-1/2}}^{x_{i+1/2}} \left|\rho(x, t^{n+1})\right| \mathrm{d}x \qquad (5.118) \\
&= \sum_{i=1}^{N} \int_{\varphi(x_{i-1/2}, t^{n+1}, -\Delta t)}^{\varphi(x_{i+1/2}, t^{n+1}, -\Delta t)} |\rho(\eta, t^n)| \, \mathrm{d}\eta \qquad (5.119) \\
&= \|\rho(\eta, t^n)\|_{L^1(\Omega)}. \qquad (5.120)
\end{aligned}$$

The effect of the projection on the L^1-norm is studied in the following.

Part 2: Effect of the projection of the exact solution on the L^1-norm

The numerical solution is given by (3.138). We differentiate between two cases. First, we study positive numerical solutions. We can show that the L^1-norm remains the same after applying the L^2-projection. Second, we allow solutions with changing sign. In that case, the L^1-norm of the solution is bounded. We start with the first case,

$$\begin{aligned}
&\int_{x_{i-1/2}}^{x_{i+1/2}} \left|\mathbb{P}\rho(x, t^{n+1})\right| \mathrm{d}x = \int_{x_{i-1/2}}^{x_{i+1/2}} \mathbb{P}\rho(x, t^{n+1}) \, \mathrm{d}x \\
&= \int_{x_{i-1/2}}^{x_{i+1/2}} \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{\varphi(x_{i-1/2}, t^{n+1}, -\Delta t)}^{\varphi(x_{i+1/2}, t^{n+1}, -\Delta t)} \rho(x, t^n) K_j(\varphi(x, t^n, \Delta t)) \, \mathrm{d}x \; K_j(x) \, \mathrm{d}x \qquad (5.121) \\
&= \int_{x_{i-1/2}}^{x_{i+1/2}} \frac{1}{\Delta x_i} \int_{\varphi(x_{i-1/2}, t^{n+1}, -\Delta t)}^{\varphi(x_{i+1/2}, t^{n+1}, -\Delta t)} \rho(x, t^n) K_0(\varphi(x, t^n, \Delta t)) \, \mathrm{d}x \; K_0(x) \, \mathrm{d}x \qquad (5.122) \\
&= \int_{x_{i-1/2}}^{x_{i+1/2}} \frac{1}{\Delta x_i} \int_{x_{i-1/2}}^{x_{i+1/2}} \rho(x, t^{n+1}) \, \mathrm{d}x \, \mathrm{d}x \qquad (5.123) \\
&= \int_{x_{i-1/2}}^{x_{i+1/2}} \rho(x, t^{n+1}) \, \mathrm{d}x \, . \qquad (5.124)
\end{aligned}$$

For the second case, we primarily use the fact that the basis function $K_j(x)$ are bounded, such that

$$\left|K_j(x)\right| \leq 1 \qquad (5.125)$$

for $j = 0, 1, 2$ and $x \in [0, \Delta x_j]$. For simplicity, we assume grid cells of equal size, i.e. $\Delta_i = \Delta_j$. We use the projection in the form (3.137) for the estimation,

$$\begin{aligned}
&\left\| \mathbb{P}\rho_i(x, t^{n+1}) \right\|_{L^1(\Omega_i)} \\
&= \int_{x_{i-1/2}}^{x_{i+1/2}} \left| \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{x_{i-1/2}}^{x_{i+1/2}} \rho(x, t^{n+1}) K_j^{(i)}(x)\, \mathrm{d}x \; K_j^{(i)}(x) \right| \mathrm{d}x && (5.126)\\
&\leq \int_{x_{i-1/2}}^{x_{i+1/2}} \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{x_{i-1/2}}^{x_{i+1/2}} \left| \rho(x, t^{n+1}) \right| \left| K_j^{(i)}(x) \right| \mathrm{d}x \; \left| K_j^{(i)}(x) \right| \mathrm{d}x && (5.127)\\
&\leq \int_{x_{i-1/2}}^{x_{i+1/2}} \sum_{j=0}^{2} \frac{2j+1}{\Delta x_i} \int_{x_{i-1/2}}^{x_{i+1/2}} \left| \rho(x, t^{n+1}) \right| \mathrm{d}x\, \mathrm{d}x && (5.128)\\
&= \int_{x_{i-1/2}}^{x_{i+1/2}} \frac{1+3+5}{\Delta x_i} \int_{x_{i-1/2}}^{x_{i+1/2}} \left| \rho(x, t^{n+1}) \right| \mathrm{d}x\, \mathrm{d}x && (5.129)\\
&= 9 \int_{x_{i-1/2}}^{x_{i+1/2}} \left| \rho(x, t^{n+1}) \right| \mathrm{d}x && (5.130)\\
&= 9 \left\| \rho_i(x, t^{n+1}) \right\|_{L^1(\Omega_i)} && (5.131)
\end{aligned}$$

Building the sum over all grid cells shows that the L^1-norm is bounded in the case of solutions with negative sign as well. In summary, we could show that the L^1-norm remains unchanged if the solution is positive and is bounded in the case of a negative solution. Thus, the method is L^1-stable.

5.2.2 L^2-Stability

As for the L^1-norm we separate the problem in two parts. First, we examine how the L^2-norm of the exact solution changes in time. Then, the effect of the L^2-projection on the L^2-norm is studied.

Part 1: Effect of the exact solution on the L^2-norm

We write the L^2-norm of the exact solution $\rho(x, t^{n+1})$ using (3.124) for the ith cell $[x_{i-1/2}, x_{i+1/2}]$ and show that we can find an estimate for the bound. We have,

$$\left\| \rho(x, t^{n+1}) \right\|_{L^2(\Omega_i)}^2 = \int_{x_{i-1/2}}^{x_{i+1/2}} \left| \rho(x, t^{n+1}) \right|^2 \mathrm{d}x \tag{5.132}$$

$$\begin{aligned}
&= \int_{x_{i-1/2}}^{x_{i+1/2}} \rho^2(\varphi(x, t^{n+1}, -\Delta t), t^n) \\
&\quad \cdot \exp\left(2 \int_0^{\Delta t} -\frac{\partial}{\partial x} u(\varphi(x, t^{n+1}, t - \Delta t))\, dt \right) \mathrm{d}x\,.
\end{aligned} \tag{5.133}$$

Then, we use the same substitution as in (3.126), i.e. $x = \varphi(\eta, t^n, \Delta t)$ and the property (3.129). This leads to

$$\begin{aligned}
&\left\|\rho(x,t^{n+1})\right\|^2_{L^2[x_{i-1/2},x_{i+1/2}]} \\
&= \int_{\varphi(x_{i-1/2},t^{n+1},-\Delta t)}^{\varphi(x_{i+1/2},t^{n+1},-\Delta t)} \rho^2(\eta,t^n) \exp\left(2\int_0^{\Delta t} -\frac{\partial}{\partial x}u(\varphi(\eta,t^n,t))dt\right) \frac{\partial}{\partial \eta}\varphi(\eta,t^n,\Delta t)\,\mathrm{d}\eta && (5.134)\\
&= \int_{\varphi(x_{i-1/2},t^{n+1},-\Delta t)}^{\varphi(x_{i+1/2},t^{n+1},-\Delta t)} \rho^2(\eta,t^n) \exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x}u(\varphi(\eta,t^n,t))dt\right) \mathrm{d}\eta && (5.135)\\
&\leq \exp\left(\int_0^{\Delta t} \max_{0\leq x\leq \Delta x}\left|\frac{\partial}{\partial x}u(\varphi(\eta,t^n,t))\right|dt\right) \int_{\varphi(x_{i-1/2},t^{n+1},-\Delta t)}^{\varphi(x_{i+1/2},t^{n+1},-\Delta t)} \rho^2(\eta,t^n)\,\mathrm{d}\eta && (5.136)\\
&\leq \exp\left(\Delta t\left\|\frac{\partial}{\partial x}u\right\|_\infty\right) \|\rho(x,t^n)\|^2_{L^2[\varphi(x_{i-1/2},t^{n+1},-\Delta t),\varphi(x_{i+1/2},t^{n+1},-\Delta t)]}. && (5.137)
\end{aligned}$$

Let $T_{\max} = K\Delta t$, where K is the number of the time steps, then we have

$$\begin{aligned}
\left\|\rho(x,t^K)\right\|_{L^2} &\leq \exp\left(\frac{1}{2}K\Delta t\left\|\frac{\partial}{\partial x}u\right\|_\infty\right) \left\|\rho(x,t^0)\right\|_{L^2} && (5.138)\\
&\leq C(T_{\max})\left\|\rho(x,t^0)\right\|_{L^2}. && (5.139)
\end{aligned}$$

Thus, for any $T_{\max}$ the local L^2-norm is bounded. As this holds for any grid cell, the global L^2-norm is bounded as well.

There is one special case, where $C(T_{\max}) = 1$. The condition is that the velocity is strictly positive or negative and to have periodic boundary conditions. In that case, there exists a time interval T_{rev} with

$$\varphi(x,t^n,T_{\text{rev}}) = x. \tag{5.140}$$

Then, $T_{\text{rev}} = T_{\max} > 0$ equals the time interval that a tracer needs for one revolution. Note, the time interval T_{rev} is the same for all points in the domain. The time interval of crossing one single cell k is given by

$$t_k = \frac{1}{a_k}\ln\left(\frac{b_{k+1}}{b_k}\right), \tag{5.141}$$

where $b_{k+N} = b_1$. The sum over all t_k equals T_{rev}. The exponential expression in (5.135), that in the general case was estimated to be bounded, can now be simplified,

$$\begin{aligned}
\exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x}u(\varphi(\eta,t^n,t))dt\right) &= \exp\left(\sum_{k=1}^N \int_0^{t_k} -a_k dt\right) && (5.142)\\
&= \exp\left(-\sum_{k=1}^N \ln\left(\frac{b_{k+1}}{b_k}\right)\right) && (5.143)\\
&= \exp\left(-\ln\left(\prod_{k=1}^N \frac{b_{k+1}}{b_k}\right)\right) && (5.144)\\
&= 1. && (5.145)
\end{aligned}$$

It follows that the L^2-norm is bounded with $C(T_{\text{rev}}) = 1$,

$$\left\|\rho(x,t^{n+1})\right\|^2_{L^2[x_{i-1/2},x_{i+1/2}]} = \int_{x_{i-1/2}}^{x_{i+1/2}} \rho^2(\eta,t^n)\exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x}u(\varphi(\eta,t^n,t))dt\right)\mathrm{d}\eta \tag{5.146}$$

$$= \|\rho(x,t^n)\|^2_{L^2[x_{i-1/2},x_{i+1/2}]}. \tag{5.147}$$

This implies that for arbitrary $T_{\max}$ the L^2-norm may grow up to some bound depending on $T_{\max}$, but if $T_{\max} \approx T_{\text{rev}}$, then we know $\left\|\rho(x,t^K)\right\|_{L^2} \approx \left\|\rho(x,t^0)\right\|_{L^2}$. Hence, the L^2-norm is periodically bounded by the norm of the initial values.

Part 2: Effect of the projection of the exact solution on the L^2-norm

For the second part we need two theorems to obtain the answer. The first one is the Weierstrass approximation theorem, the proof can be found in e.g. [64].

Theorem 5.2.1 (Weierstrass approximation theorem) *The subspace $P[a,b]$ of polynomials on $[a,b] \subset \mathbb{R}$, is dense in $(C[a,b], \|\cdot\|_\infty)$.*

Furthermore, we recall the following theorem (see e.g. [64]).

Theorem 5.2.2 *$C[a,b]$ is dense in $L^p[a,b]$ for $1 \leq p < \infty$.*

With the above theorems we can conclude that for any function $f \in L^p$ there exists a sequence of polynomials p_n with $\|f - p_n\|_p \to 0$, $n \to \infty$. Actually, we can write

$$f = \sum_{i=0}^{\infty} c_i p_i(x), \tag{5.148}$$

where c_i are coefficients computed by

$$c_i = \int_a^b f(x)p_i(x)\,\mathrm{d}x \tag{5.149}$$

and p_i are orthonormal basis functions of $P[a,b]$. Then, (5.150) is the L^2-projection onto $P^N[a,b]$. To obtain the projection $\mathbb{P}$ onto $P^2[a,b]$, only the first three summands are taken,

$$\mathbb{P}f = \mathbb{P}\sum_{i=0}^{N} c_i p_i(x) \tag{5.150}$$

$$= \sum_{i=0}^{2} c_i p_i(x). \tag{5.151}$$

Now, we can compute the L^2-norm of f and $\mathbb{P}f$.

$$\|f\|_2 = \left(\int_a^b \left(\sum_{i=0}^{\infty} c_i p_i(x)\right)^2 \mathrm{d}x\right)^{1/2} = \left(\sum_{i=0}^{\infty} c_i^2\right)^{1/2} \tag{5.152}$$

and

$$\|\mathbb{P}f\|_2 = \left(\int_a^b \left(\sum_{i=0}^{2} c_i p_i(x)\right)^2 \mathrm{d}x\right)^{1/2} = \left(\sum_{i=0}^{2} c_i^2\right)^{1/2}. \tag{5.153}$$

From (5.152) and (5.153) it follows that

$$\|\mathbb{P}f\|_2 \leq \|f\|_2, \tag{5.154}$$

hence the L^2-norm is bounded when the projection step is applied.

This result and the finding that the L^2-norm of the analytical solution of the linear advection equation for a specific velocity field is bounded, yield the L^2-stability of the numerical method.

5.2.3 Von Neumann stability analysis

In the sections 5.2.1 and 5.2.2 we have shown that the numerical method is stable. Yet, we further investigate the stability property of the scheme by means of the von Neumann stability analysis. It gives a deeper understanding of the errors that occur as it examines the diffusion and dispersion error separately. The von Neumann stability analysis offers additionally the possibility to derive the order of accuracy from a certain point of view.

The procedure of a von Neumann stability analysis applied to numerical methods is introduced in Section 2.1.2. Exemplarily, the analysis is carried out for the FOU method in Example 2.1.1.

The application of the von Neumann stability analysis to the SASLDG method must be altered, because it was intented for the analysis of finite difference methods or finite volume methods, which have the property that for each grid cell or grid node one value is computed. For each grid cell of the SASLDG method the information of a polynomial is stored, i.e. the coefficients $m_{0,i}^n$, $m_{1,i}^n$ and $m_{2,i}^n$ for the ith grid cell. To simplify the computations in the following we examine the numerical method for the constant velocity case only. Additionally, equidistantly distributed grid points are assumed. The method can be represented by

$$v^{n+1} = Av^n, \tag{5.155}$$

where A is a suitable matrix and the vector v^n contains the coefficients $m_{0,j}^n$, $m_{1,j}^n$ and $m_{2,j}^n$

$$v^n = \begin{pmatrix} m_{0,j}^n \\ m_{1,j}^n \\ m_{2,j}^n \end{pmatrix}, \tag{5.156}$$

and v^{n+1} at time step $n+1$, respectively.

We follow closely the analysis conducted by van Leer in [31], where he has shown the stability properties for the second-order scheme described in Section 2.2. The coefficients given at time n in his scheme can be updated in time for the new time level $n+1$ using a 2×2- matrix. That approach is extended to a 3×3- matrix. We use the coefficients derived in Chapter 3, but in the simplified version for constant velocity. These are written in the previous section in (5.96), (5.98) and (5.100).

Each row of (5.155) yields the respective update for the coefficients in time and is represented as finite Fourier series using the notation introduced in Section 2.1.2. We display the result for wave number k. The equations are shown in the form of

$$v^{n+1} = Mv^n, \tag{5.157}$$

where v^n is defined in (5.156). The eigenvalues of the matrix M determine the stability properties of the numerical method and can be interpreted as amplification factors. The matrix M is given by

$$M = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}, \tag{5.158}$$

with the first row

$$a_{11} = \sigma e^{-i\alpha_k} + 1 - \sigma, \tag{5.159}$$

$$a_{12} = \left(\sigma - \sigma^2\right) e^{-i\alpha_k} + \sigma^2 - \sigma, \tag{5.160}$$

$$a_{13} = \left(2\sigma^3 - 3\sigma^2 + \sigma\right) e^{-i\alpha_k} - 2\sigma^3 + 3\sigma^2 - \sigma, \tag{5.161}$$

the second row

$$a_{21} = \left(3\sigma^2 - 3\sigma\right) e^{-i\alpha_k} + 3\sigma - 3\sigma^2, \tag{5.162}$$

$$a_{22} = \left(-2\sigma^3 + 6\sigma^2 - 3\sigma\right) e^{-i\alpha_k} + 2\sigma^3 - 3\sigma + 1, \tag{5.163}$$

$$a_{23} = \left(3\sigma^4 - 12\sigma^3 + 12\sigma^2 - 3\sigma\right) e^{-i\alpha_k} - 3\sigma^4 + 6\sigma^2 - 3\sigma, \tag{5.164}$$

and the third row

$$a_{31} = \left(10\sigma^3 - 15\sigma^2 + 5\sigma\right) e^{-i\alpha_k} - 10\sigma^3 + 15\sigma^2 - 5\sigma, \tag{5.165}$$

$$\begin{aligned} a_{32} = &\left(-5\sigma^4 + 20\sigma^3 - 20\sigma^2 + 5\sigma\right) e^{-i\alpha_k} \\ &+ 5\sigma^4 - 10\sigma^2 + 5\sigma, \end{aligned} \tag{5.166}$$

$$\begin{aligned} a_{33} = &\left(6\sigma^5 - 30\sigma^4 + 50\sigma^3 - 30\sigma^2 + 5\sigma\right) e^{-i\alpha_k} \\ &- 6\sigma^5 + 10\sigma^3 - 5\sigma + 1. \end{aligned} \tag{5.167}$$

From this matrix we can study the stability properties as well as the diffusion and dispersion error of the numerical method. The eigenvalues of M, needed for the analysis, are e_1, e_2 and e_3 displayed in the appendix (A.39) to (A.49) in Section A.5.1.

Stability

Similar to the von Neumann stability condition in (2.26) the method is stable, if the absolute value of the eigenvalues is less than one. The absolute value of all eigenvalues depends on the Courant number. It is not obvious to see, at which number the absolute value takes the maximum.

Van Leer showed in [31] that for the second-order case the diffusion error, which is computed from the absolute value of the eigenvalues, has a maximum at $\sigma = 1/2$. Therefore, when he calculates the diffusion error he uses that value of σ. In our case the Courant number, where the maximum error is reached, cannot be computed directly and is only approximated. We plot a series expansion of $e_1(\sigma)$ for fixed value $\alpha_k = \pi/4$ about the point $\sigma = 1/2$ up to order eight as seen in Figure 5.2. From the figure we can see that the approximate minimum absolute value of the e_1, i.e. the maximum diffusion error, is reached in the vicinity of $\sigma = 0.25$ and $\sigma = 0.75$.

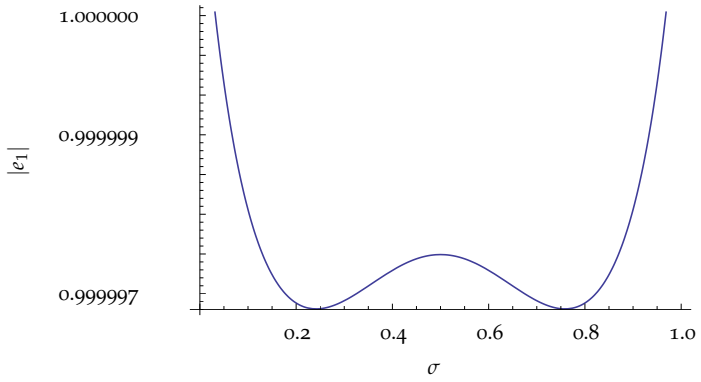


Figure 5.2: Series expansion of $|e_1(\sigma)|$ about $\sigma = \frac{1}{2}$, $\alpha_k = \frac{\pi}{4}$,

Another observation that we make when looking at Figure 5.2 is that the absolute value of the first eigenvalue is symmetrical at point 1/2. This can be formally proven, again in the same way as van Leer did it. It can be shown that $e_1(1-\sigma) = e^{-i\alpha_k}\overline{e_1(\sigma)}$ holds, where $\overline{e_1}$ denotes the complex conjugate of e_1. From this equation the requested symmetry relation follows directly

$$|e_1(1-\sigma)| = |e_1(\sigma)|. \tag{5.168}$$

To study the properties of e_1, we evaluate the absolute value nevertheless for $\sigma = 1/2$ even though the maximum error is reached at a different value for σ as seen in Figure 5.2, because the computations are simplified algebraically for that value.

Hence, the absolute value of the first eigenvalue at the point $\sigma = 1/2$ is given by

$$\left|e_1\left(\frac{1}{2}\right)\right| = \left|-\frac{1}{8}\cos\left(\frac{\alpha_k}{2}\right) + \frac{7\cdot 3^{2/3}}{2\cdot 2^{2/3}\sqrt[3]{p}} + \frac{1}{16}\sqrt[3]{\frac{3}{2}}\sqrt[3]{p}\right|, \tag{5.169}$$

where

$$\begin{aligned} p = {} & 426\cos\left(\frac{\alpha_k}{2}\right) + 6\cos\left(\frac{3\alpha_k}{2}\right) \\ & + \sqrt{3}\sqrt{31098\cos(\alpha_k) + 852\cos(2\alpha_k) + 6\cos(3\alpha_k) - 57556}. \end{aligned} \tag{5.170}$$

To examine the behavior for small α_k, we expand the absolute eigenvalues about $\alpha_k = 0$ and obtain the following series expansion for e_1 for $\sigma = 1/2$

$$\left|e_1\left(\frac{1}{2}\right)\right| = \left|1 - \frac{\alpha_k^6}{92160} + \mathcal{O}\left(\alpha_k^8\right)\right|. \tag{5.171}$$

Similarly, Taylor expansions about the point $\sigma = 1/2$ are obtained for the second and third eigenvalue e_2 and e_3, and further the expressions are expanded about $\alpha_k = 0$. The results yield

$$\left|e_2\left(\frac{1}{2}\right)\right| = \left|-\frac{1}{2} + \frac{3}{32}\alpha_k^2 + \mathcal{O}\left(\alpha_k^4\right)\right| \tag{5.172}$$

and

$$\left|e_3\left(\frac{1}{2}\right)\right| = \left|-\frac{7}{8} - \frac{3}{64}\alpha_k^2 + \mathcal{O}\left(\alpha_k^4\right)\right|. \tag{5.173}$$

The absolute values of all eigenvalues are less than or equal to one, which shows the stability of the method as expected from the results of the stability analysis of the previous sections.

Diffusion error

As described in Section 2.1.2 the absolute and relative diffusion error is given by (2.29) and (2.30). To understand the impact of the eigenvalues that contribute to the diffusion error, we examine the leading order of the respective eigenvalues. The leading order of eigenvalue e_1 is one, of e_2 it is $1/2$ and of e_3 it is $7/8$. The leading order of e_2 and e_3 is less than one, thus their contribution to the total solution is damped out after a few time steps. Therefore, the long time behavior is only driven by the first eigenvalue e_1.

For this reason the absolute diffusion error made per time step is approximately

$$e_{\text{diff}} = 1 - |e_1(\sigma)| \tag{5.174}$$

and the relative diffusion error

$$\tilde{e}_{\text{diff}} = |e_1(\sigma)|. \tag{5.175}$$

To determine the diffusion error and its order we use (5.171). The chosen Courant number of $1/2$ does not play a role for the determination of the order of consistency of the diffusion error.

That means that the diffusive error of the SASLDG method for constant velocity is of fifth order with respect to its eigenfunctions that correspond to (5.157). The actual order of convergence is of order three as shown in Section 5.1.4.

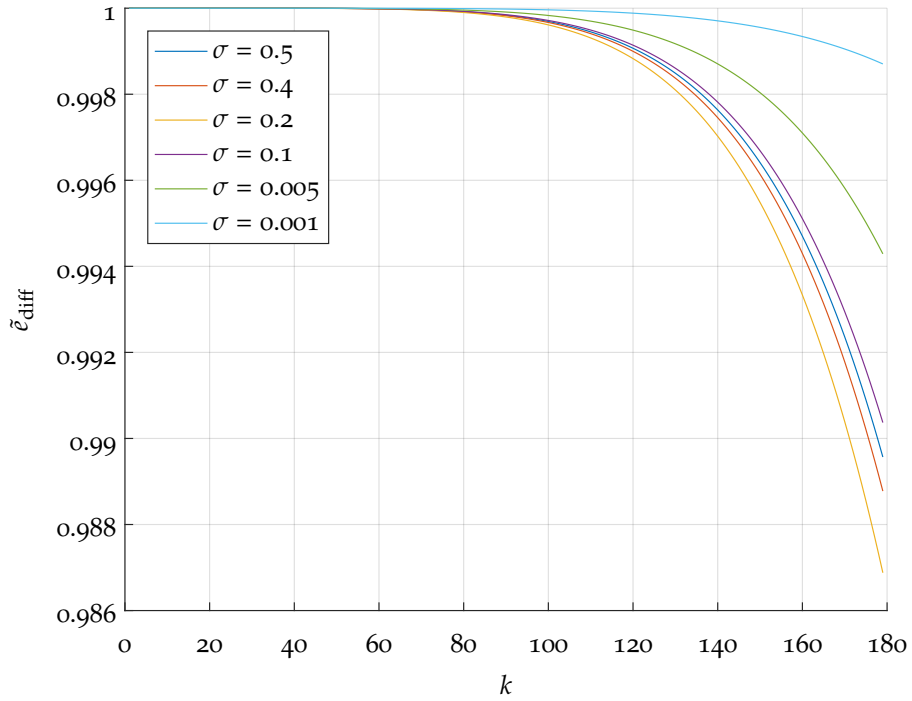


Figure 5.3: Relative diffusion error for different values of σ.

The diffusion error depends on the value of the Courant number σ. How much and in what way the error is influenced by σ is illustrated in Figure 5.3. It shows the diffusion error for different CFL numbers. The functions shown here are series expansions of $e_1(\sigma)$ at $\sigma = 0.5$, $\sigma = 0.4$, $\sigma = 0.2$, $\sigma = 0.1$, $\sigma = 0.005$ and $\sigma = 0.001$ for small value α_k. Because of the symmetry relation (5.168), we only plot CFL-numbers smaller than 0.5 since larger Courant numbers lead to equal results. As expected from Figure 5.2 the largest diffusion error occurs for $\sigma = 0.2$ among the plotted graphs. The closer the CFL number is to zero or to one, the smaller is the diffusive error.

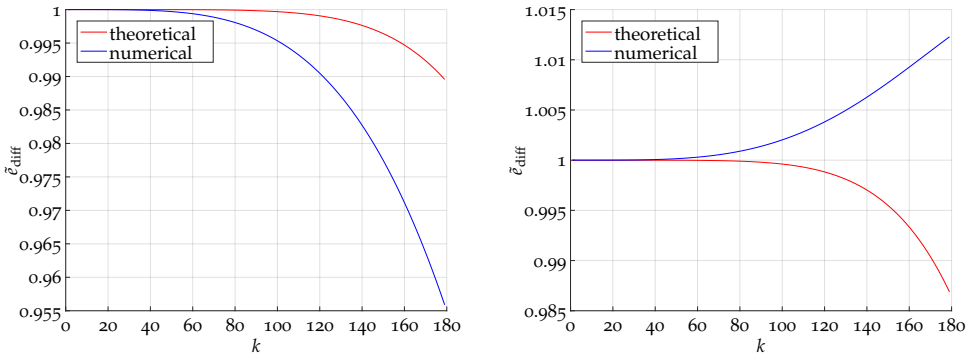
Note, that these are theoretical results that are only based on computations based on the first eigenvalue e_1. The other eigenvalues influence the numerical solution as well.

To study the impact of all eigenvalues we can examine the numerically derived diffusion error. In the same way as we conducted the Fourier analysis for MPDATA, we carry out the procedure for the SASLDG method. For details of the numerical Fourier analysis see Section 2.6.2. The deviation from only considering e_1, which is theoretically derived to the impact of all eigenvalues as seen in the numerical Fourier analysis can be seen in the Figure 5.4.

While Figure 5.4(a) shows that the diffusive error of the numerical analysis for CFL =0.5 is only larger than the theoretical diffusive error computed from e_1, Figure 5.4(b) seems to reveal an catastrophic result of the numerically determined diffusion error for CFL = 0.8: it displays an amplification factor larger than one, in particular for large wave numbers. Numerical methods with such an amplification factor are expected to be unstable and let the numerical solution grow indefinitely in time. For the SASLDG method this is not the case - it is stable.

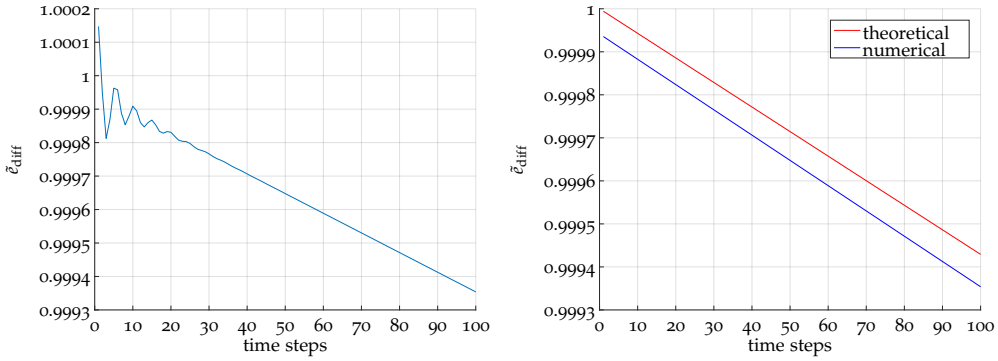
Therefore, we examine the numercially derived diffusive error more closely in particular for $\sigma = 0.8$.

We look at the evolution in time of the diffusive error for fixed wave number $K = 50$. This is depicted in Figure 5.5(a). We can see for the first time step that the diffusive error in this particular case is larger than one, seconded by the results in Figure 5.4(b). For the next few time steps the amplification factor drops below



(a) $\sigma = 0.5$ **(b)** $\sigma = 0.8$

Figure 5.4: Comparison of relative theoretical versus relative numerical diffusion error for different values of σ.



(a) Numerically determined relative diffusion error in time.

(b) Comparison of slopes of numerically determined diffusion without oscillation and theoretically derived diffusion.

Figure 5.5: Diffusion development for 100 time steps in time, for fixed wave number $k = 50$ and Courant number $\sigma = 0.8$.

one and starts oscillating in time. After approximately 20 to 30 time steps the oscillations are damped and the plotted function is roughly of linear form. This can be explained by the impact of the second and third eigenvalue. They cause the oscillations until the components are damped out. After that only the first eigenvalue e_1 governs the numerical solution and thus the diffusive behavior.

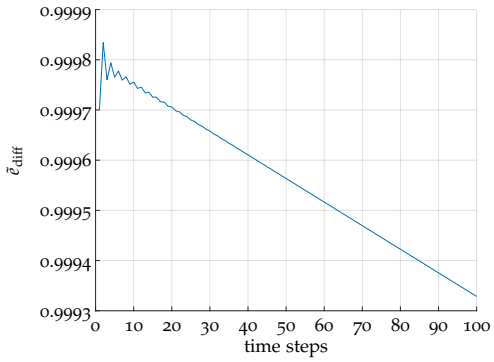
To show the result without the oscillations we compute the average slope between the 50th and 200th time step. This is plotted in Figure 5.5(b). The slope determined from the theoretical amplification factor for CFL = 0.8 computed from only e_1 is given as Taylor expansion about $\alpha_k = 0$ by

$$f(\alpha_k, n) = \left(1 - \frac{2687}{196875000}\alpha_k^6 + \mathcal{O}\left(\alpha_k^7\right)\right)^n. \tag{5.176}$$

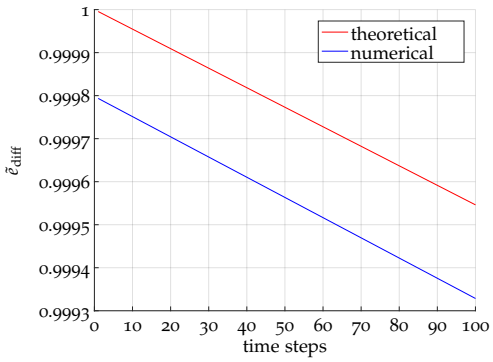
The graph of $f(50, n)$ is shown in comparison to the numerical values. The numerical and theoretical slope coincide approximately aside from a shift, which is also due to the influence of e_2 an e_3.

A similar in depth analysis is done in Figure 5.6 for CFL = 0.5 and again $k = 50$ to see the influence of two different Courant numbers on the numerical solutions and their respective diffusive behavior.

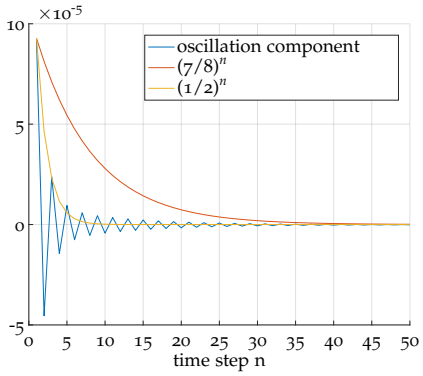
Figures 5.6(a) and 5.6(b) correspond to Figures 5.5(a) and 5.5(b) and show the respective results for Courant number 0.5. Again, the form of the graph in Figure 5.6(a) showing the diffusion in time can be described as a linear function with



(a) Numerically determined diffusion relative error in time.



(b) Comparison of slopes of numercially determined diffusion without oscillation and theoretically derived diffusion.



(c) Oscillation with mean slope substracted.

Figure 5.6: Diffusion development for 100 time steps in time, for fixed wave number $k = 50$ and Courant number $\sigma = 0.5$.

negative slope and visible oscillations occuring in the first twenty to thirty time steps. In this case the numerically determined amplification factor remains below one for all time steps. When we compare the functions without the oscillations we note that the diffusion after one time step for CFL = 0.5 takes approximately the value 0.9998, whereas for CFL = 0.8 it is ca 0.99995 (see Figures 5.6(b) and 5.5(b)). However, as the slope for CFL = 0.8 is steeper, after some time steps the diffusive error for CFL = 0.8 is larger than the error for CFL = 0.5. This can also be seen for the theoretically derived diffusion in Figure 5.3. The graph plotted for CFL number 0.2, which equals the one for CFL = 0.8, takes smaller values than the function for CFL = 0.5 for all wave numbers.

Figure 5.6(c) describes the decay of the oscillations. The component of the linear function is substracted from the amplification factor and only the pure oscillations are shown. The functions $f(n) = (7/8)^n$ and $f(n) = (1/2)^n$ are also depicted in this figure, which describe roughly the decay of the impact of the eigenvalues e_2 and e_3 (compare (5.172) and (5.173)). The absolute value of the extracted numerical oscillations lie between these two functions.

Dispersion

The absolute and relative dispersion error of a numerical method derived by means of the von Neumann analysis is given by (2.31) and (2.32), respectively. For the

SASLDG method the first eigenvalue is interpreted as amplification factor. Its argument yields the dispersion error. Then, the relative dispersion error is given by

$$\tilde{e}_{\text{disp}} = \frac{\tan^{-1}\left(-\frac{\operatorname{Im} e_1}{\operatorname{Re} e_1}\right)}{\sigma\alpha_k}. \tag{5.177}$$

To determine the order of the dispersion error we compute the error in the limit of small Δx, when Δt is fixed. Thus, we let σ go to zero,

$$\lim_{\sigma\to 0} \tilde{e}_{\text{disp}} = \lim_{\sigma\to 0} \frac{\arctan\left(-\frac{\operatorname{Im} e_1}{\operatorname{Re} e_1}\right)}{\sigma\alpha_k}. \tag{5.178}$$

The expression (5.178) is difficult to solve. No symbolic mathematical computation program seems to be able so solve that equation directly. Hence, we need to simplify the expression. This is shown in the appendix in Section A.5.2.

The result is given as a series expansion about wave number $\alpha_k = 0$,

$$\lim_{\sigma\to 0} \tilde{e}_{\text{disp}} = 1 + \frac{\alpha_k^6}{42000} + \mathcal{O}\left(\alpha_k^7\right). \tag{5.179}$$

Therefore, the theoretically determined order of the dispersion error for the SASLDG method for constant velocity is order five with respect to the eigenfunctions of 5.157. As noted for the diffusion error: this finding holds only for the eigenfunctions and the order of convergence is of order three.

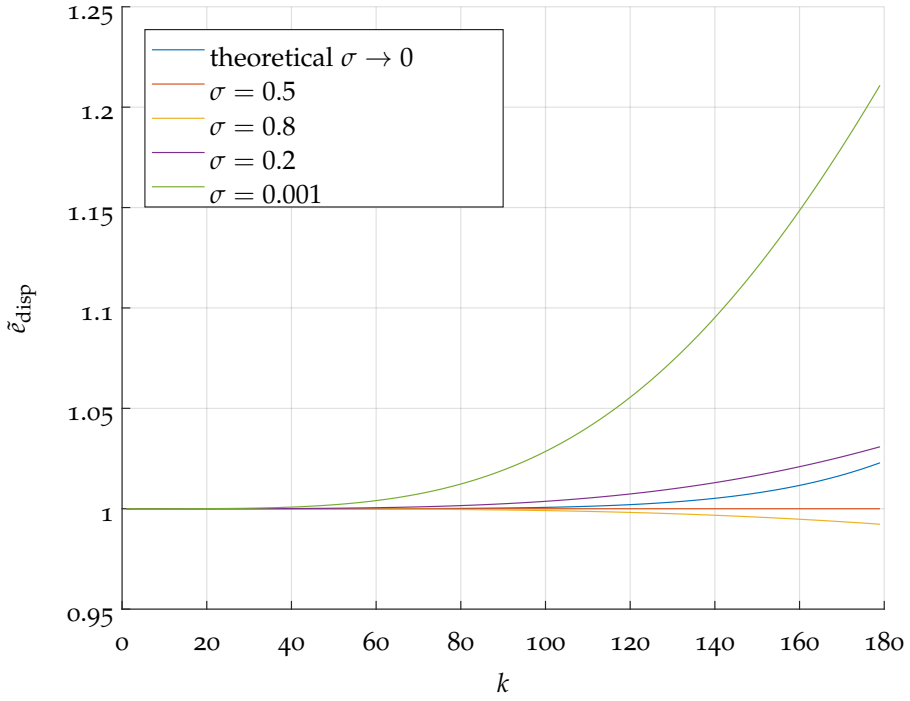


Figure 5.7: Relative dispersion error.

The relative dispersion error is theoretically derived for Courant number $\sigma = 0$ only. However, the error for other CFL numbers can be determined in the same way as we did for the diffusion error. In Figure 5.7 the relative dispersion error is plotted for different CFL numbers as well as the theoretical result for $\sigma \to 0$. Depending on the Courant number the error is larger or smaller than one. That means that the numerical solution propagates either faster or slower than the analytical solution. For CFL numbers lager than 0.5 the error is a lagging error, for numbers smaller than 0.5 the error is leading. For CFL = 0.5 the error is aside from small numerical

errors equal to zero. Note, that the dispersion error is not symmetric with respect to the CFL number unlike the diffusion error.

The theoretical dispersion error determined for the limiting case of σ going to zero is also shown in Figure 5.8. The numerically calculated dispersion errors seem to converge to a different limiting function for a small Courant number. The numerical dispersion error is larger than the theoretically derived one. The plot suggests that it is limited as σ goes to zero. The dispersion error made using $\sigma = 0.001$, barely differs from the error with Courant number equal to 10^{-7}. For example, the difference for wave number 180 for the latter named CFL numbers is in the order of 10^{-3}. The difference for CFL 10^{-7} and 10^{-6} again for wave number 180 is in the order of 10^{-8}.

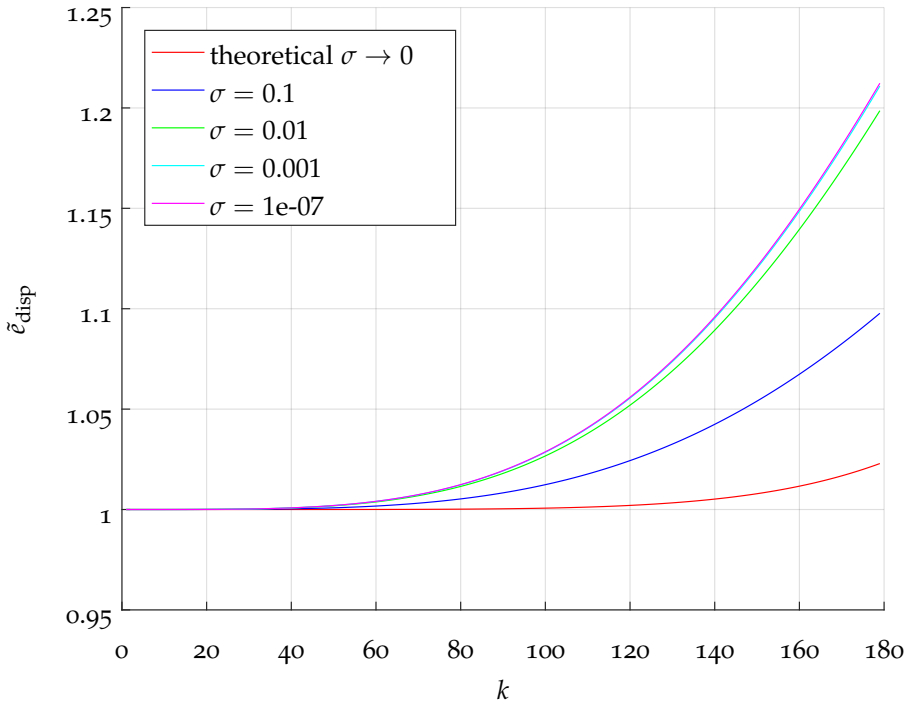


Figure 5.8: Dispersion to zero

The deviation of the theoretically and numercially determined dispersion for small Courant numbers can be explained by the interaction of all three eigenvalues in the numercially derived dispersion error from the numerical solution, whereas for the theoretical determined dispersion error (5.179) the first eigenvalue only plays a role.

In summary we can say that the long time behavior is governed only by the first eigenvector and thus it is suitable to use e_1 for the analysis. The theoretically derived amplification factors suggest a smaller diffusive error than the effective numerical amplification factors, but are still beneficial approximations.

5.3 CONVERGENCE RATES

In this section we determine numerically the convergence rate of the SASLDG method. To obtain the initial data the sine function is projected cellwise onto a polynomial of degree two. The sine function used as initial data for $x \in [0,1]$ is given by

$$f_{IV}(x) = \sin(2\pi x), \tag{5.180}$$

which takes positive and negative values. Additionally, it is used in a modified version, such that the convergence rates can be computed from purely positive initial data as well

$$f_{IV}(x) = \sin(2\pi x) + 1.5. \tag{5.181}$$

Further, two different velocity fields are used for the computations. The first one is constant, i. e.

$$u(x) = 1. \tag{5.182}$$

The second one is of variable velocity and given by

$$u(x) = \left(\frac{1}{2}(\sin(\pi x))\right)^2 + 1. \tag{5.183}$$

The velocity fields are chosen to be strictly positive, such that after some finite time the initial distribution is advected to the initial position. We compute the numerical solution after one revolution. The error is computed cellwise between the reconstructed polynomial function of the numerical solution and the sine function, which is the analytical solution. The difference of the numerical and analytical solution is used to assemble an error function for each refinement level r and for each grid cell j

$$f^r_{\text{err,j}}(x) = \left| f^r_{\text{num,j}}(x) - f^r_{\text{ana,j}}(x) \right|. \tag{5.184}$$

From the function $f^r_{\text{err,j}}(x)$ we compute the $L^r_{\max}$, L^r_1 and L^r_2 error

$$L^r_{\max} = \max_j \left(\max_x f^r_{\text{err,j}}(x) \right), \tag{5.185}$$

$$L^r_1 = \sum_j \int f^r_{\text{err,j}}(x)\, \mathrm{d}x \tag{5.186}$$

and

$$L^r_2 = \sqrt{\sum_j \int {f^r_{\text{err,j}}}^2(x)\, \mathrm{d}x}. \tag{5.187}$$

The convergence rate can be computed from the errors of the different refinement levels, here shown for the $L_{\max}$ error,

$$c^r_{\max} = \frac{\ln(L^{r+1}_{max}) - \ln(L^r_{max})}{\ln(g^{r+1}) - \ln(g^r)}, \quad \text{for } r \in \{1, \ldots, 9\} \tag{5.188}$$

where $g^r = 2^{r+2}$ is the number of grid cells used for refinement level r. The convergence rates for the L_1 and L_2 errors are determined analogously.

We compute the convergence rates for different scenarios. In the case of constant velocity we apply two different Courant numbers to see the influence of the CFL number 1/2. In the case of variable velocity we vary first the initial data, second the CFL number and third the switch of the numerical method for small coefficients a_k of the slope of the velocity field.

Tables 5.1 and 5.2 show the convergence rates for the constant velocity case (5.182). For both computations the initial function (5.181) is used. Table 5.1 displays the results for CFL = 0.45, Table 5.2 for CFL = 0.5, respectively. Both tables confirm the convergence rate of order three for the constant velocity case as shown theoretically in 5.1.4.

Though both tables indicate the third order accuracy, Table 5.2 shows a rate closer to order three. This can be explained by the dispersion error, which is equal to zero for the Courant number 0.5. Only the diffusion error causes numericals errors.

N	L_{max}-error	L_{max}-order	L_1-error	L_1-order	L_2-error	L_2-order
8	1.47e-03	-	3.22e-04	-	4.28e-04	-
16	7.37e-04	2.8330	1.03e-04	3.0027	1.49e-04	2.9568
32	1.05e-04	2.8102	1.44e-05	2.8360	2.05e-05	2.8583
64	1.30e-05	3.0104	1.73e-06	3.0607	2.49e-06	3.0383
128	1.53e-06	3.0936	2.04e-07	3.0862	2.97e-07	3.0719
256	1.64e-07	3.2163	2.46e-08	3.0522	3.54e-08	3.0679
512	2.11e-08	2.9603	3.08e-09	2.9962	4.45e-09	2.9911
1024	2.86e-09	2.8855	3.91e-10	2.9767	5.69e-10	2.9672
2048	4.03e-10	2.8265	5.51e-11	2.8284	7.83e-11	2.8616

Table 5.1: Convergence rates for constant velocity defined in (5.182) and the CFL number 0.45. The strictly positive initial values are given by (5.181).

N	L_{max}-error	L_{max}-order	L_1-error	L_1-order	L_2-error	L_2-order
8	1.08e-03	-	3.16e-04	-	4.16e-04	-
16	6.25e-04	2.9578	1.01e-04	3.0263	1.44e-04	2.9943
32	7.87e-05	2.9895	1.26e-05	3.0065	1.80e-05	2.9986
64	9.85e-06	2.9974	1.57e-06	3.0017	2.26e-06	2.9997
128	1.23e-06	2.9993	1.96e-07	3.0004	2.82e-07	2.9999
256	1.54e-07	2.9998	2.45e-08	3.0001	3.52e-08	3.0000
512	1.93e-08	3.0000	3.06e-09	3.0000	4.41e-09	3.0000
1024	2.41e-09	3.0000	3.83e-10	3.0000	5.51e-10	3.0000
2048	3.01e-10	2.9999	4.79e-11	3.0003	6.88e-11	3.0000

Table 5.2: Convergence rates for constant velocity defined in (5.182) and the CFL number 0.5. The strictly positive initial values are given by (5.181).

We expect a convergence rate of order two from the theoretical results derived in Section 5.1.4, when a variable velocity field as given in (5.183) is used. Some factors affect the convergence rates for the new method. The positivity of the initial values, the CFL number, and the switch for the algorithm for a small slope of the velocity coefficients causes slight changes in the rates.

N	L_{max}-error	L_{max}-order	L_1-error	L_1-order	L_2-error	L_2-order
8	4.29e-03	-	7.61e-04	-	1.07e-03	-
16	1.27e-03	2.4983	2.06e-04	2.4471	2.72e-04	2.4730
32	2.48e-04	2.3633	4.62e-05	2.1567	5.99e-05	2.1828
64	5.49e-05	2.1747	1.12e-05	2.0387	1.45e-05	2.0516
128	1.29e-05	2.0851	2.76e-06	2.0229	3.57e-06	2.0161
256	3.13e-06	2.0454	6.89e-07	2.0047	8.91e-07	2.0041
512	7.71e-07	2.0238	1.72e-07	2.0019	2.23e-07	2.0011
1024	2.06e-07	1.9024	4.30e-08	1.9985	5.58e-08	1.9973
2048	1.42e-07	0.5404	1.21e-08	1.8291	1.63e-08	1.7775

Table 5.3: Convergence rates for variable velocity defined in (5.183) and the CFL number 0.45. The initial values are given by (5.180).

The influence of the positivity of the initial values is studied in the Tables 5.3 and 5.4. The initial values, that are used for the computation are given by (5.180), which takes positive and negative values, and (5.181), which is strictly positive, respectively. Both tables show the rate of order two up to the refined grid with 1024 grid cells. The convergence rates drop for the next refinement level due to cancellation of significant digits in the computations. The L_{max}-order shows the significant drop of the rate at the refinement level of 2048 grid cells ,the L_1-order

N	L_{max}-error	L_{max}-order	L_1-error	L_1-order	L_2-error	L_2-order
8	9.85e-03	-	2.13e-03	-	2.80e-03	-
16	3.01e-03	2.1765	5.23e-04	2.0815	7.13e-04	2.0700
32	7.18e-04	2.0704	1.29e-04	2.0249	1.76e-04	2.0196
64	1.74e-04	2.0407	3.20e-05	2.0044	4.39e-05	2.0018
128	4.34e-05	2.0055	8.15e-06	1.9745	1.11e-05	1.9787
256	1.08e-05	2.0026	2.05e-06	1.9923	2.81e-06	1.9898
512	2.71e-06	2.0015	5.15e-07	1.9938	7.04e-07	1.9945
1024	6.78e-07	1.9972	1.29e-07	1.9953	1.77e-07	1.9958
2048	4.47e-07	0.6004	3.44e-08	1.9061	4.74e-08	1.8980

Table 5.4: Convergence rates for variable velocity defined in (5.183) and the CFL number 0.45. The strictly positive initial values are given by (5.181).

and the L_2-order display that issue at the next refinement level, which is not shown here. However, the rates in Table 5.4 computed from strictly positive initial values reveal a slighly better result and seem to be more robust than the rates in Table 5.3 with initial data of changing sign.

N	L_{max}-error	L_{max}-order	L_1-error	L_1-order	L_2-error	L_2-order
8	1.07e-02	-	2.19e-03	-	2.86e-03	-
16	3.05e-03	2.2435	5.34e-04	2.0826	7.27e-04	2.0698
32	7.23e-04	2.0746	1.30e-04	2.0342	1.78e-04	2.0278
64	1.75e-04	2.0478	3.22e-05	2.0207	4.41e-05	2.0161
128	4.36e-05	2.0057	8.18e-06	1.9745	1.12e-05	1.9784
256	1.09e-05	2.0027	2.06e-06	1.9903	2.82e-06	1.9890
512	2.71e-06	2.0023	5.17e-07	1.9949	7.06e-07	1.9954
1024	6.79e-07	1.9986	1.29e-07	2.0000	1.77e-07	2.0003
2048	4.36e-07	0.6399	3.38e-08	1.9329	4.62e-08	1.9331

Table 5.5: Convergence rates for variable velocity defined in (5.183) and the CFL number 0.5. The strictly positive initial values are given by (5.181).

In Table 5.5 we see the convergence rates for the SASLDG method computed with the same settings as in Table 5.4 apart from a change of the Courant number from 0.45 to 0.5. The results are similar with slightly smaller errors in Table 5.5. Just as for constant velocity, the CFL number 0.5 reduces the errors by a small amount.

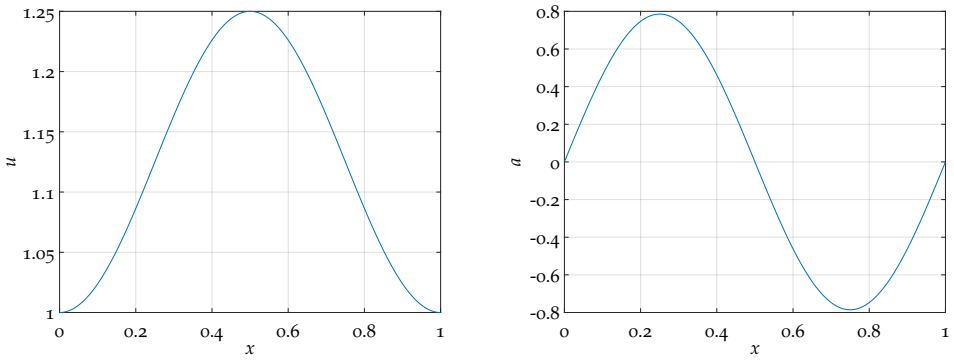
The convergence rates shown in all tables above are computed with the possibility enabled to switch between the algorithms for small or large coefficients a_k that occur in the slope of the velocity. The bound is chosen to be 0.5. The slope of the velocity is equal to zero in the case for constant velocity. Hence, the relevant coefficient is below the bound everywhere and the according branch of the algorithm is applied. The convergence rates for the constant velocity case yields results without numerical cancellation. Thus, the branch of the algorithm is well suited for a fine grid and small time steps. The other branch of the algorithm for coefficients larger than the bound reveals problems because of the cancellation. If we examine the method closely, we note that wherever coefficients a_k appears in the equations there is the always the product of a_k and Δt given. The branch of the algorithm was intented for small a_k and according series expansion are applied about the point $a_k = 0$. However, for small Δt the same expansions hold. In the study of the convergence rates small Δx and small Δt are assumed. Thus, the requirements for choosing the branch for small a_k is also fulfilled for small Δt.

Table 5.6 shows the results if we increase the bound of the switch such that that algorithm for small a_k is applied for all a_k. The convergence rates show the order two for all refinement levels. Cancellation of significant digits does not occur.

N	L_{max}-error	L_{max}-order	L_1-error	L_1-order	L_2-error	L_2-order
8	4.29e-03	-	7.61e-04	-	1.07e-03	-
16	1.27e-03	2.4984	2.06e-04	2.4471	2.72e-04	2.4731
32	2.48e-04	2.3633	4.62e-05	2.1569	5.99e-05	2.1831
64	5.50e-05	2.1725	1.12e-05	2.0379	1.45e-05	2.0503
128	1.29e-05	2.0879	2.77e-06	2.0220	3.57e-06	2.0172
256	3.13e-06	2.0450	6.89e-07	2.0064	8.91e-07	2.0044
512	7.70e-07	2.0238	1.72e-07	2.0018	2.23e-07	2.0011
1024	1.91e-07	2.0121	4.30e-08	2.0005	5.56e-08	2.0003
2048	4.75e-08	2.0062	1.07e-08	2.0001	1.39e-08	2.0001

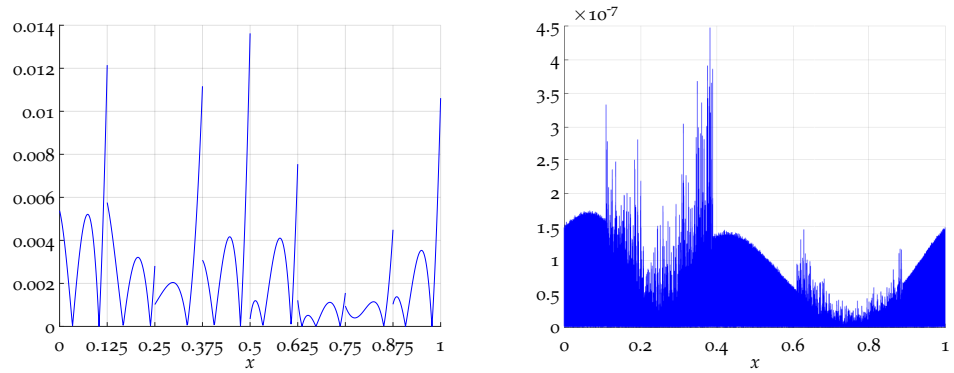
Table 5.6: Convergence rates for variable velocity defined in (5.183) and the CFL number 0.45. The initial values are given by (5.180). The branch of the algorithm for small a_0 and a_n is applied only.

Figure 5.9 and 5.10 picture the influence of the choice of the bound to switch the algorithms on the error in the solution. Figure 5.9(a) displays the velocity distribution defined in (5.183), Figure 5.9(b) shows the slope of the velocity, i.e. the coefficient a_k. These figures become of interest, when studied in combination with Figure 5.10, which shows the error of the numerical solution computed with variable velocity.



(a) The velocity u. **(b)** Slope of u.

Figure 5.9: The velocity u described in (5.183) and its slope a.



(a) Computed on 8 grid cells. **(b)** Computed on 2048 grid cells.

Figure 5.10: Error plot of numerical solution computed with variable velocity and CFL number 0.45.

Figure 5.10(a) demonstrates the error plot for a coarse grid of eight grid cells. For each grid cell j the error function $f^r_{err,j}(x)$ is plotted, with $r = 1$ as the refinement

level. Figure 5.10(b) shows the error for a grid of 2048 cells, i.e. the refinement level $r = 9$. Note the different scaling.

Because of the amount of grid cells, it is not possible to see the distribution of a error function for a particular grid cell. However, the plot reveals the areas with the largest error. These areas correspond exactly with the interval where the absolute value of the slope a_k is above the bound of 0.5. This means wherever the branch of the algorithm is used for large a_k, the numerical solution becomes inaccurate. The other branch for small a_k has the property to damp the errors, which is illustrated by the smooth areas wherever $|a_k| \leq 0.5$. This damping effect is also discussed in one test case in Section 3.7.

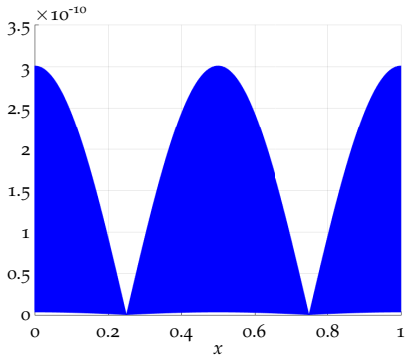


Figure 5.11: Error plot of numerical solution computed with constant velocity and CFL number 0.5.

Figure 5.11 shows the error plot for the same refinement level as used in Figure 5.10(b), that is 2048 grid cells. The numerical solution is computed with constant velocity. Hence, the algorithm for small a_k is applied everywhere. The error plot is correspondingly smooth.

In summary, the numerically computed convergence rates confirm the theoretical results. The SASLDG method is third order accurate, when applied to a constant velocity field and of second order accuracy for variable velocity.

6 DISCUSSION

We discuss the findings of the previous chapters, the construction and development of the SASLDG method, the extension of the method to two space dimensions and the analysis. In particular, the numerical results of the SASLDG method are considered thoroughly in this discussion. Furthermore, we highlight possibilities to extend and enhance the SASLDG method in future work.

The SASLDG method strives for the goal of solving the compressible linear advection equation with as many steps computed analytically as possible while retaining a suitable form of the solution, i.e. polynomial form. The derivation of the SASLDG method reveals the difficulties of that objective. The advected quantity is followed analytically along trajectories and projected onto polynomial distributions for each time step. The velocity field given by piecewise linear functions sets constraints on the form of the trajectory: We must differentiate between strictly positive velocity, strictly negative velocity, and a velocity distribution with a root for each grid cell. Time step size, the length of a grid cell and the velocity influence the number of grid cells that a trajectory crosses. The form of the trajectory alters if it remains within one grid cell or if it crosses grid cell boundaries. Furthermore, if the slope of the velocity equals zero in any of the grid cells that a trajectory crosses, the trajectory takes a different form. This can occur at three occasions, namely in the departure cell, in the arrival cell of the trajectory or in both cells. The algorithm of the SASLDG method takes account of all these possible forms of trajectories. In the subsequent projection step, the analytically correct solution is cast into polynomial form. The integrals that are solved analytically for the projection differ depending on the form of the trajectories. Additionally, in the computation of the integrals, cancellation of significant digits can occur and must be handled. Thus, on the one hand the algorithm is complicated and branched due to many if statements, on the other hand it yields a semi-analytical solution to the linear advection equation.

The numerical tests conducted in Chapter 3 show the results of the SASLDG method for test cases in one space dimension. We tested two different initial values. First, a smooth initial distribution, i.e. the sine function. Second, we chose the step function as initial data with a jump discontinuity. We examined the numerical solution with these initial values with a constant and a variable velocity field, on a regular grid and on an irregular grid with grid cells of random size. Furthermore, tests with a series of CFL numbers smaller than one as well as larger than one were conducted. Additionally, the application of a slope limiter is tested.
In general, the SASLDG method is very accurate with a small maximum- and l_1-error in all test cases. In particular, smooth initial values, which can be well approximated by the polynomials, are advected without introducing large numerical errors. In contrast, if the step function is chosen as initial distribution, the jump discontinuity within a grid cell cannot be fitted to quadratic polynomial functions as well.
An irregular grid with varying grid cell sizes can be handled without any issues. However, the maximum- and the l_1-error are larger than on a regular grid. The tests with CFL number 10 show a reduction of the error compared to the tests with CFL number 0.45.
This is also confirmed by the test series with different increasing CFL numbers. When we consider the results with a smooth initial distribution and double the CFL number, the errors roughly decrease by the factor of two. Additionally, the computational time is approximately reduced by half as well. This is an obvious fact for CFL numbers of less than one, since the amount of time steps is reduced by the fac-

tor two as well. However, for CFL numbers larger than one, the trajectories become more complicated as they cross potentially several grid cells. Thus, the tracking of the trajectories for a longer time step is less expensive than the computation of the projection. We carry the increase of the CFL number to extremes and choose it in such a way that the time step size equals the time $T_{\max}$ and the numerical solution after one circulation is computed with one single time step. For variable velocity, the errors are further decreased and of the same order on the regular and irregular grid. The l_∞- and l_1-error of the solution on a regular grid reach almost machine accuracy for constant velocity.
Analysing the test series with discontinuous initial values and constant velocity, we observe a general decline (or small momentarily increase) of the l_∞- and l_1-error if the CFL number is increased. The same holds for the over- and undershoots that arise in the numerical solution for that test series. The dispersion error influences the numerical results for this test series, in particular noticeable at the over- and undershoots. If the CFL number is larger than 0.5, the error is lagging. If the CFL number is smaller than 0.5, the error is leading, which means an increase of the oscillations in the vicinity of the discontinuity either ahead or behind the jump. The dispersion error vanishes for the CFL number equal to 0.5. Therefore, the results show smaller errors for 0.5 than for slightly larger CFL numbers as 0.6, even though less time steps are made and consequently less projections have to be computed. If the time step size is chosen such that the numerical solution at time $T_{\max}$ is obtained with one single time step, the l_∞- and l_1-error are in the order of machine accuracy.

Some phenomena, e.g. in the dynamics of the atmosphere or the ocean, occur at different scales in vertical and horizontal direction. If we assume a problem of large horizontal but small vertical scale and have to choose a high resolution for the vertical direction, the same resolution often cannot be applied to the horizontal direction because of the computational cost. A grid with high aspect ratio is the consequence.
Furthermore, the structure of the grid, in particular its increment, also depends on on the initial values and the velocity field. The regularity of the initial data enables a higher or lower resolution. Smooth initial values can be resolved on a coarser grid, whereas for the representation of sharp discontinuities a relatively fine grid is needed. Since the regularity and the variation of the initial values might vary in the space dimensions, different grid aspect ratios might be suitable. In addition, the velocity field must be considered for the choice of the aspect ratio, i.e. the different resolution in both space dimensions. A strongly varying velocity field must be resolved on an adequate fine grid.
Independent of the origin for the grid with high aspect ratio, methods that can only apply CFL numbers up to number one, are subject to a strict time step restriction.

The SASLDG method is extended to solve the two-dimensional linear advection equation via operator splitting. The general procedure of the 2D-version is equal to the method in one space dimension. However, six coefficients describe the polynomial that represents the numerical solution. Although the 2D algorithm works in the same way as in 1D, the integrals of the projection step have to be determined differently because of different representation of the solution. The implementation of the analytical computation is restricted to the 1D case. However, for testing purposes the integrals are solved using the quadrature of MATLAB. Because of the complexity of the algorithm, the computational cost to determine the numerical solution is high.
For that reason another method is developed and introduced to make use of the SASLDG algorithm and solve the advection equation in 2D. The so-called hybrid method combines the complex, but high-accurate SASLDG method and the robust MPDATA, with low computational cost. To adjust the SASLDG method to an operation with cell mean values only, a modified version is developed. This so-called SASLDG-3c method combines three grid cells to one. It uses the information of the three mean values to build the coefficients that describe the polynomial for the

larger cell. Via operator splitting it is constructed to use MPDATA in horizontal and the SASLDG-3c method in vertical direction. A feature of the hybrid method is that there is no time step restriction in vertical direction. As a consequence, the hybrid method can handle problems with high aspect ratio with a reasonable time step size.

Four test cases are studied in two space dimensions. First, the solid body rotation test and second, a deformational test, i.e. the static vortex test case are carried out. The 2D SASLDG method, MPDATA, the SASLDG-3c method and the hybrid method are tested with these test cases. Furthermore, two additional test cases, another deformational test using the Rider Kothe velocity field and a wavelike flow test, offer more insights into the performance of the MPDATA, SASLDG-3c and the hybrid method.
The tests have several objectives. The test cases solid body rotation and the Rider Kothe test in which a sine hill is rotated and a gaussian hill is deformed, respectively, examine the methods with respect to smooth initial data. Initial distributions that contain discontinuities are used and tested in the other two test cases: a staircase function for the static vortex test and a distribution that only consists of sharp edges for the wavelike flow test. The solid body rotation test is driven from a velocity field of constant velocity for each one-dimensional splitting operator. The velocity fields of the other test cases are all of variable velocity. Thus, constant and variable velocity is applied in the 2D tests. The capability of the methods to handle grids with high aspect ratio is of interest as well. Furthermore, the computational cost is measured for the solid body rotation, the Rider Kothe and the wavelike flow test.

Considering the solid body rotation, the result of the SASLDG method stands out with respect to two aspects. First, the l_∞- and the l_1-error are small compared to the errors of the other methods. However, the second feature is the large computational time. Comparing the extremes MPDATA and SASLDG, we find that the computational time of SASLDG is approximately 184 times larger than of MPDATA. The error of the solution of SASLDG is 12 times smaller than of MPDATA. The higher accuracy of the SASLDG method does not justify the high cost. The performance of the modified version of the SASLDG method is remarkable. The computation of the numerical solution of the SASLDG-3c method takes roughly three times longer than of MPDATA. Yet, the errors are approximately five times smaller. The good result of SASLDG-3c with respect to smooth initial data, i.e. the sine function, is also shown in the 1D test case. The errors of SASLDG and its modified version are comparable.

The superiority of the SASLDG method with respect to the quality of the solution with smooth initial values does not apply to the solution computed from initial data with discontinuities. We examine the outcome of the static vortex test case. The l_∞- and l_1-error of the solution of SASLDG are smaller than of the other methods, but not to that extent. The errors of the SASLDG method are approximately by a factor of 0.7 smaller than the errors of MPDATA. The hybrid method yields an astonishing error behavior in this test case. The l_∞-error is very close to the error of SASLDG-3c, which is smaller than the error of MPDATA, and the l_1-error of the SASLDG-3c method is even smaller than of both individual methods. Thus, the combination of both methods resulting in the hybrid method yields the best result. When studying the one-dimensional comparison of MPDATA and the SASLDG-3c method for the test with the step function as initial data, we find that the l_∞- as well as the l_1-error of the SASLDG-3c method is smaller, but the oscillations reach a larger amplitude. Since MPDATA is applied in horizontal advection, it is responsible for advection across the steps of the staircase function. The SASLDG-3c method advects the smooth solution in vertical direction and thus not across the step but within, without any discontinuities. In that way the result of the hybrid method can lead to better results than the individual methods.

The deformational test case using the Rider Kothe velocity field starts with smooth initial values, i.e. a gaussian hill. Though, the initial data is comparable to the data of the solid body rotation test, the test is of more complicated structure. The velocity applied to the data is variable in space. The initial gaussian hill is deformed into an arch during the test. The upwards and downwards advection of the hill deforms the hill relatively little compared to the horizontal advection, where the most deformation takes place.
On the grid of 120×120 grid cells the SASLDG-3c method yields better results than MPDATA in terms of l_∞- and l_1-error. Similar to the static vortex test case, the result with the smallest errors - smaller than for the individual methods - is given by the hybrid method. Again, the vertical advection computed by the SASLDG-3c method leads to smaller errors than computed by MPDATA. If MPDATA is used for horizontal advection instead of the SASLDG-3c method, the errors remain smaller. Thus, the hybrid method has the smallest errors. If we changed the roles of MPDATA and SASLDG-3c for the hybrid method and used MPDATA for the vertical and SASLDG-3c for the horizontal advection, the l_∞- and the l_1-error would take larger values. A reason for the better performance of MPDATA with respect to the horizontal advection could be the higher resolution of the grid. This could be helpful to capture the horizontally more complicated velocity field. The velocity field describes two vortices in horizontal direction, in contrast to the vertical vortex that stretches over the whole computational domain.
Considering the results of the Rider Kothe test computed on the 120×360 grid, the conclusion for the hybrid method, which yields the smallest errors of all methods, holds as well. It is interesting to note that the error by MPDATA decreases switching to this grid, whereas the error of the SASLDG-3c method remains about the same. This can be explained by the fact that the horizontal resolution of 40 grid cells used by SASLDG-3c limits a further reduction of an error for that particular problem. As soon as the horizontal resolution is chosen higher, the error of the SASLDG-3c method diminishes. (We conducted this test, however the result is not shown in this thesis.)
The hybrid method yields a suitable result if a larger CFL number of 0.5 with respect to the horizontal grid is applied. The time step chosen for this test is approximately six times larger than for the other tests. This corresponds to the CFL number 2.99 in vertical direction, which is only possible for the SASLDG-3c method. The l_∞- and l_1-error increase by a factor of approximately 1.1 and 1.8, respectively, compared to the values of MPDATA. In return, the speed up for the computation with a larger time step is about a factor of 4.6.

The goal of the wavelike test is to study the long time behavior of a tracer with initial values consisting of sharp discontinuities that is advected in a wavelike manner. The transport takes place in upwards and downwards as well as in left and right direction, in which the vertical movement is predominant. The numerical methods are tested with respect to the qualitative conservation of the discontinuities. Since the initial values are constant apart from the discontinuities, the errors occur only in their neighborhood. The tests on both grids, the 150×150 and 90×540, confirm once more similarly to the results to the previous test cases: The numerical solution computed by the hybrid method is a hybrid combination of MPDATA and the SASLDG-3c method. We consider the width of the error (the area, where the error is larger than the value 0.1), the absolute error at vertical and horizontal edges, and the l_1-error. The l_∞-error is not as meaningful for the conservation of the discontinuities because it consists of single peaks at the corners of the cut-out corner. The solution of the hybrid method inherits the features of the individually computed solutions.
The wavelike test shows the capability to handle large CFL numbers in vertical direction without restriction on the time step size. In the example on the 90×540 grid, i.e. a grid of aspect ratio 1:6, the time step is chosen 12 times larger than in the tests prior to that. Even though the errors grow about a factor 1.2 to 1.7, the run-

time decreases about a factor 8.6. Depending on the application, this trade-off can serve a purpose. Summarized, the wavelike test confirms that the hybrid method can be used on a grid with high aspect ratio and relatively large CFL numbers and still obtain reasonable results. However, the large oscillations that arise at the discontinuities must be captured in order to reduce this error source. Thus, the development of a limiter for the SASLDG-3c method might be worthwile for future work.

The benefit of the SASLDG method to handle arbitrary CFL numbers and allow large time steps applies in the one-dimensional test cases with smooth initial values. If we solve the linear advection equation in two space dimensions via operator splitting, this advantage does not apply directly since large time steps are not applicable for operator splitting. However, with regard to grids with high aspect ratio, the possibility to make large time steps is reasonable. The time step restriction for some numerical methods up to CFL number one can limit these methods to very small time steps in one space dimension if the grid aspect ratio is high.
The test cases that are conducted for the hybrid method with two CFL numbers of different sizes indicate that an increase of the CFL number results in a growth of the l_1- and the l_∞-error. A possible way to make use of the nonexistent time step restriction for the SASLDG method in 2D is the development of a full multi-dimensional Lagrangian or semi-Lagrangian method that are not based on operator splitting.

The accuracy of the SASLDG method is second order for variable velocity, i.e. for a continuous, piecewise linear velocity field. If the advection velocity is constant, the SASLDG method is third order accurate as shown analytically. These results are confirmed by numerical convergence tests. A problem that arises within these tests is cancellation of significant digit. If the grid is refined to 2048 grid cells, the convergence rates drop. A possible remedy is to use the algorithm for small coefficients $|a_0|$ and $|a_n|$ which is an adequate approximation. The Taylor expansion used to obtain this approximation about $a_0 = 0$ and $a_n = 0$ holds for $\Delta t = 0$ as well.
In order to increase the accuracy, the representation of the velocity field and of the numerical solution must be adapted. If the velocity field is given as a continuous, piecewise polynomial of degree two (instead of one), the computation of the trajectories must be altered. The analytical solution of the resulting ODE exists and can be determined. However, its computation requires even more distinction of cases and is of more complicated form. Thus, the remaining algorithm becomes more complex, too.
The increase of the degree of the polynomials for the representation of the numerical solution leaves the determination of the trajectories unaffected. Instead, the projection step must be adapted. The solution of all integrals is explicitly implemented in the algorithm that is needed for the projection of the analytical solution. The polynomials are part of the integrand. Thus, when the order of the polynomials change, the solution of the altered integrals must be implemented. The solution with a higher order accuracy can be determined in the same semi-analytical way, but with higher computational cost.

An asset of DG methods is the hp-refinement. It enables the local adjustment of the grid cell sizes and the degree of the polynomials. In that way, complex geometries can be adequately represented. This feature can be implemented for the SASLDG method, too. A local change of the the grid cell sizes can be realized by a projection. The split of one grid cell into two cells, leaves the polynomial distribution unchanged. Two sets of new coefficients are determined that describe the polynomial for each grid cell. To combine two or more grid cells into one cell, a projection is applied. The possibility to adjust the degree of the polynomials is mentioned above. It can be implemented, but with higher computational cost. However, in opposite to DG methods, the size of time step is not affected by the degree of the polynomials and can be arbitrarly large for the SASLDG method.

An open problem of the current implementation of the SASLDG method is the existence of cancellation of significant digits if the grid cells are small. The con-

vergence test, conducted for variable velocity, reveals the problem. However, the cancellation of significant digits does not occur for the convergence analysis if the algorithm branch for small $|a_0|$ and $|a_n|$ is used. This ansatz of using the alternative algorithm or a similar approximation might solve the problem of cancellation.

The high computational cost of the algorithm is another problem to be resolved in future work. The exact determination of the trajectories with the associated distinction of cases for the different scenarios of the velocity (strictly positive/negative/roots etc.) leads to as many cases for the computation of the integrals. In addition, the cases of small coefficients in the respective departure and arrival cell must be considered. The possible cancellation of significant digits and its treatment complicate the algorithm. In principle, the trajectories are smooth functions as they are diffeomorphisms. Yet, the computation of the integral, which consists of the product of a polynomial and the Legendre polynomials with the trajectories as argument, is costly. Adequate approximations at deliberate parts of the algorithm can be a remedy for the high computational cost. Possible approaches could be the usage of numerical quadrature methods and approximations of the trajectories. An application of quadrature schemes has the additional benefit to be independent of the degree of the polynomial, which simplifies the implementation of hp-refinement. A suggestion for the approximation to the trajectory is provided by the existent branch for small $|a_0|$ and $|a_n|$. This approximation is also valid for a small grid cell and time step size.

The SASLDG method yields promising results in 1D as well as in 2D of high accuracy. A remedy for the downside of high computational cost and the cancellation of significant digits at certain points of the algorithm can be the usage of approximations.

The version of SASLDG elaborated in this thesis can serve as a reference for further developments. An enhancement of this method can lead to (nearly) as accurate numerical results of the compressible linear advection equation, but more robust and with lower computational cost.

APPENDIX

A APPENDIX

A.1 DETAILS OF PROPOSITION 3.3.1

We are interested in the solution of the ODE 3.3, that reads

$$\begin{aligned} &\frac{d}{dt}\rho(x(t),t) = -\rho(x(t),t)\frac{\partial}{\partial x}u(x(t),t), \\ &\rho(x(t_0),t_0) = \rho_0. \end{aligned} \tag{A.1}$$

We want to determine the solution for $\rho(x, t^{n+1})$ at the time level t^{n+1}. However, the function ρ is only known at time t^n. It turns out to be convenient to choose the trajectory $x(t)$ to be of the form $\varphi(x, t^{n+1}, t - \Delta t)$. By doing so, we obtain for $t = t^n$ some point $\varphi(x, t^{n+1}, -\Delta t)$, that is followed backward from (x, t^{n+1}) to end at time level t^n. For $t = t^{n+1}$, we recover the value $\varphi(x, t^{n+1}, 0) = x$ at time level t^{n+1}. The ODE is then given by

$$\begin{aligned} &\frac{d}{dt}\rho(\varphi(x,t^{n+1},t-\Delta t),t) \\ &\qquad = -\rho(\varphi(x,t^{n+1},t-\Delta t),t)\frac{\partial}{\partial x}u(\varphi(x,t^{n+1},t-\Delta t),t). \end{aligned} \tag{A.2}$$

The separation of variables and integration over time yields

$$\begin{aligned} &\int_0^{\Delta t} \frac{1}{\rho(\varphi(x,t^{n+1},t-\Delta t),t)}d\rho \\ &\qquad = \int_0^{\Delta t} -\frac{\partial}{\partial x}u(\varphi(x,t^{n+1},t-\Delta t)),t)dt. \end{aligned} \tag{A.3}$$

The integral on the lefthandside can be computed

$$\begin{aligned} &\ln\Big(\rho(\varphi(x,t^{n+1},0),\Delta t)\Big) - \ln\Big(\rho(\varphi(x,t^{n+1},-\Delta t),0)\Big) = \\ &\qquad \int_0^{\Delta t} -\frac{\partial}{\partial x}u(\varphi(x,t^{n+1},t-\Delta t)),t)dt, \end{aligned} \tag{A.4}$$

which equals

$$\begin{aligned} &\rho(x,t^{n+1}) = \\ &\rho(\varphi(x,t^{n+1},-\Delta t),t^n)\exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x}u(\varphi(x,t^{n+1},t-\Delta t),t)dt\right). \end{aligned} \tag{A.5}$$

A.2 ADDITIONAL COMPUTATIONS FOR 3.3

We want to show that the following property holds

$$\exp\left(\int_{t^n}^{t^{n+1}} -\frac{\partial}{\partial x}u(\varphi(\eta,t^n,t))dt\right)\frac{\partial}{\partial \eta}\varphi(\eta,t^n,\Delta t) = 1. \tag{A.6}$$

The product arises in (3.128) because the determinant of the Jacobian matrix is introduced from the substitution.

Because of Definition 3.2.1, we have

$$\int_0^\tau u(\varphi(\eta, t^n, t))dt = \int_0^\tau \frac{d\varphi}{dt}dt \tag{A.7}$$

$$= \varphi(\eta, t^n, \tau) - \varphi(\eta, t^n, 0) \tag{A.8}$$

$$= \varphi(\eta, t^n, \tau) - \eta. \tag{A.9}$$

We take the derivative of (A.9) with respect to η and obtain

$$\int_0^\tau \frac{\partial}{\partial x}u(\varphi(\eta, t^n, t))\frac{\partial}{\partial \eta}\varphi(\eta, t^n, t)dt = \frac{\partial}{\partial \eta}\varphi(\eta, t^n, \tau) - 1. \tag{A.10}$$

Taking another derivate, now with respect to τ, yields

$$\frac{\partial}{\partial x}u(\varphi(\eta, t^n, \tau))\frac{\partial}{\partial \eta}\varphi(\eta, t^n, \tau) = \frac{\partial}{\partial \tau}\frac{\partial}{\partial \eta}\varphi(\eta, t^n, \tau). \tag{A.11}$$

This implies the following

$$\frac{\frac{\partial}{\partial \tau}\frac{\partial}{\partial \eta}\varphi(\eta, t^n, \tau)}{\frac{\partial}{\partial \eta}\varphi(\eta, t^n, \tau)} = \frac{\partial}{\partial x}u(\varphi(\eta, t^n, \tau)). \tag{A.12}$$

Because of the form of the derivative of the logarithmic function, we have

$$\frac{\partial}{\partial \tau}\ln\left(\frac{\partial}{\partial \eta}\varphi(\eta, t^n, \tau)\right) = \frac{\partial}{\partial x}u(\varphi(\eta, t^n, \tau)). \tag{A.13}$$

Integration of (A.13) with respect to τ in the limits from 0 to Δt yields

$$\ln\left(\frac{\frac{\partial}{\partial \eta}\varphi(\eta, t^n, \Delta t)}{\frac{\partial}{\partial \eta}\varphi(\eta, t^n, 0)}\right) = \int_0^{\Delta t}\frac{\partial}{\partial x}u(\varphi(\eta, t^n, t))dt. \tag{A.14}$$

Exponentation of (A.14) together with the definition of φ in (3.6) yields the final result of

$$\frac{\partial}{\partial \eta}\varphi(\eta, t^n, \Delta t) = \frac{\partial}{\partial \eta}\varphi(\eta, t^n, 0)\exp\left(\int_0^{\Delta t}\frac{\partial}{\partial x}u(\varphi(\eta, t^n, t))dt\right) \tag{A.15}$$

$$= \exp\left(\int_0^{\Delta t}\frac{\partial}{\partial x}u(\varphi(\eta, t^n, t))dt\right). \tag{A.16}$$

Thus, we have

$$\frac{\partial}{\partial \eta}\varphi(\eta, t^n, \Delta t)\exp\left(\int_0^{\Delta t} -\frac{\partial}{\partial x}u(\varphi(\eta, t^n, t))dt\right) = 1. \tag{A.17}$$

A.3 TRAJECTORIES

A.3.1 Positive velocity: $|a_0| < \epsilon$

We compute the square of the trajectory given in (3.43). Then we determine an approximation by a series expansion about $a_0 = 0$. We obtain

$$\begin{aligned}
&\widetilde{\varphi}^2(x, t^n, \Delta t) \\
&= \frac{u_n^2}{a_n^2} \exp\left(2a_n(\Delta t - T)\right) \exp\left(2\frac{a_n}{a_0} \ln\left(-\frac{a_0}{u_1}\Delta x_0 \eta + 1\right)\right) \\
&\quad -2\frac{u_n}{a_n}\widetilde{\varphi}(x, \Delta t) + \frac{u_n^2}{a_n^2}
\end{aligned} \tag{A.18}$$

$$\begin{aligned}
&= \frac{u_n^2}{a_n^2}(\exp(a_n(\Delta t - T)))^2 \exp\left(-2\frac{a_n \Delta x_0}{u_1}x\right)\left(-\frac{a_0^3 a_n^3 \Delta x_0^6}{6u_1^6}x^6\right. \\
&\quad + \frac{2a_0^3 a_n^2 \Delta x_0^5}{3u_1^5}x^5 - \frac{1}{6u_1^6}\left(3a_0^3 a_n u_1^2 \Delta x_0^4 - 3a_0^2 a_n^2 u_1^2 \Delta x_0^4\right)x^4 \\
&\quad \left. - \frac{2a_0^2 a_n \Delta x_0^3}{3u_1^3}x^3 - \frac{a_0 a_n \Delta x_0^2 x^2}{u_1^2} + 1\right) \\
&\quad -2\exp\left(\frac{-a_n \Delta x_0}{u_1}x\right)\frac{u_n^2}{a_n^2}\exp\left(a_n(\Delta t - T)\right)\left(-\frac{a_0^3 a_n^3 \Delta x_0^6}{48u_1^6}x^6\right. \\
&\quad + \frac{a_0^3 a_n^2 \Delta x_0^5}{6u_1^5}x^5 - \frac{a_0^3 a_n \Delta x_0^4}{4u_1^4}x^4 + \frac{a_0^2 a_n^2 \Delta x_0^4}{8u_1^4}x^4 - \frac{a_0^2 a_n \Delta x_0^3}{3u_1^3}x^3 \\
&\quad \left. - \frac{a_0 a_n \Delta x_0^2}{2u_1^2}x^2 + 1\right) + \frac{u_n^2}{a_n^2} + \mathcal{O}\left(a_0^4\right)
\end{aligned} \tag{A.19}$$

$$\begin{aligned}
&= g1\exp(g_2 x)(1 + g_3 x^2 + g_4 x^3 + g_5 x^4 + g_6 x^5 + g_7 x^6) \\
&\quad +2d_8 d_1 \exp(d_2 x)(1 + d_3 x^2 + d_4 x^3 + d_5 x^4 + d_6 x^5 + d_7 x^6) \\
&\quad +g_8 + \mathcal{O}\left(a_0^4\right).
\end{aligned} \tag{A.20}$$

A.3.2 Positive velocity: $|a_n| < \epsilon$

We find an approximation of the square of $\widetilde{\varphi}$ given in (3.43). We first compute the square and then carry out a Taylor expansion about $a_n = 0$,

$$\begin{aligned}
\widetilde{\varphi}^2(x, t^n, \Delta t) = {} & \frac{u_n^2 a_n^3}{4a_0^5} \ln\left(-\frac{a_0 \Delta x_0}{u_1} x + 1\right)^5 \\
& + \frac{u_n^2 a_n^2}{12 a_0^4} \left(15(\Delta t - T) a_n + 7\right) \ln\left(-\frac{a_0 \Delta x_0}{u_1} x + 1\right)^4 \\
& + \frac{u_n^2 a_n}{6 a_0^3} \Bigg(15(\Delta t - T)^2 a_n^2 \\
& \qquad + 14(\Delta t - T) a_n + 6\Bigg) \ln\left(-\frac{a_0 \Delta x_0}{u_1} x + 1\right)^3 \\
& + \frac{u_n^2}{2 a_0^2} \Bigg(5(\Delta t - T)^3 a_n^3 + 7(\Delta t - T)^2 a_n^2 \\
& \qquad + 6(\Delta t - T) a_n + 2\Bigg) \ln\left(-\frac{a_0 \Delta x_0}{u_1} x + 1\right)^2 \\
& + \frac{u_n^2}{12 a_0} \Bigg(\Delta t - T)(15(\Delta t - T)^3 a_n^3 + 28(\Delta t - T)^2 a_n^2 \\
& \qquad + 36(\Delta t - T) a_n + 24)\Bigg) \ln\left(-\frac{a_0 \Delta x_0}{u_1} x + 1\right) \\
& + \frac{1}{4} u_n^2 a_n^3 (\Delta t - T)^5 + \frac{7}{12} u_n^2 a_n^2 (\Delta t - T)^4 + u_n^2 a_n (\Delta t - T)^3 \\
& + u_n^2 (\Delta t - T)^2 + \mathcal{O}\left(a_n^4\right)
\end{aligned} \tag{A.21}$$

$$\begin{aligned}
= {} & e_1 + e_2 \ln(d_6 x + 1) + e_3 \ln(d_6 x + 1)^2 + e_4 \ln(d_6 x + 1)^3 \\
& + e_5 \ln(d_6 x + 1)^4 + e_6 \ln(d_6 x + 1)^5 + \mathcal{O}\left(a_n^4\right).
\end{aligned} \tag{A.22}$$

A.3.3 positive velocity: $|a_0| < \epsilon$ and $|a_n| < \epsilon$

The approximation to the square of the trajectory derived for positive velocity, and small coefficients $|a_0|$ and $|a_n|$ in the departure and arrival cell is determined by first computing the square of $\widetilde{\varphi}$ given in (3.43). Then the approximation is determined by a series expansions about $a_0 = 0$ and $a_n = 0$,

$$
\begin{aligned}
\widetilde{\varphi}^2(x, t^n, \Delta t) = {} & \frac{119}{72}\frac{u_n^2}{u_1^6} a_n^2 \Delta x_0^6 a_0^2 x^6 \\
& - \frac{7}{12}\frac{u_n^2}{u_1^5} a_n \Delta x_0^5 a_0 \Big(7(\Delta t - T) a_0 a_n + 3a_0 - 2a_n \Big) x^5 \\
& + \frac{1}{24}\frac{u_n^2}{u_1^4} \Delta x_0^4 \Big(77(\Delta t - T)^2 a_0^2 a_n^2 + 66(\Delta t - T) a_0^2 a_n \\
& - 84(\Delta t - T) a_0 a_n^2 + 22 a_0^2 - 36 a_0 a_n + 14 a_n^2 \Big) x^4 \\
& - \frac{1}{18}\frac{u_n^2}{u_1^3} \Delta x_0^3 \Big(14(\Delta t - T)^3 a_0^2 a_n^2 + 18(\Delta t - T)^2 a_0^2 a_n \\
& - 63(\Delta t - T)^2 a_0 a_n^2 + 12(\Delta t - T) a_0^2 - 54(\Delta t - T) a_0 a_n \\
& + 42(\Delta t - T) a_n^2 - 18 a_0 + 18 a_n \Big) x^3 \\
& - \frac{1}{6}\frac{u_n^2}{u_1^2} \Delta x_0^2 \Big(7(\Delta t - T)^3 a_0 a_n^2 + 9(\Delta t - T)^2 a_0 a_n \\
& - 21(\Delta t - T)^2 a_n^2 + 6(\Delta t - T) a_0 - 18(\Delta t - T) a_n - 6 \Big) x^2 \\
& - \frac{1}{3}\frac{u_n^2}{u_1} (\Delta t - T) \Delta x_0 \Big(7(\Delta t - T)^2 a_n^2 + 9(\Delta t - T) a_n + 6 \Big) x \\
& + (a_n(\Delta t - T)^3 + (\Delta t - T)^2 + (7/12) a_n^2 (\Delta t - T)^4) u_n^2 + \mathcal{O}\left(a_0^3\right) + \mathcal{O}\left(a_n^3\right) \qquad \text{(A.23)} \\
= {} & \sum_{i=1}^{7} e_i x^{i-1} + \mathcal{O}\left(a_0^3\right) + \mathcal{O}\left(a_n^3\right) \qquad \text{(A.24)}
\end{aligned}
$$

A.3.4 Negative velocity: $|a_0| < \epsilon$

We compute the square of the trajectory given in (3.90). Then we determine an approximation by a series expansion about $a_0 = 0$. We obtain

$$\begin{aligned}
\widetilde{\varphi}^2(x, t^n, \Delta t) = \frac{u_{n+1}^2}{a_n^2} \exp\left(2a_n(\Delta t - T)\right) \Bigg(& 1 - \frac{\Delta x_0^2}{u_0^2} x^2 a_0 a_n \\
& + \frac{2}{3}\frac{\Delta x_0^3}{u_0^3} x^3 a_0^2 a_n + \frac{1}{2}\frac{\Delta x_0^4}{u_0^4}\left(-a_0^3 a_n + a_0^2 a_n^2\right) x^4 \\
& - \frac{2}{3}\frac{\Delta x_0^5}{u_0^5} a_0^3 a_n^2 x^5 - \frac{1}{6}\frac{\Delta x_0^6}{u_0^6} a_0^3 a_n^3 x^6 \Bigg) \exp\left(2a_n \frac{\Delta x_0}{u_0} x\right) \\
& + 2\left(-\frac{u_n}{a_n}\right)\frac{u_{n+1}}{a_n} \exp\left(a_n(\Delta t - T)\right) \Bigg(1 - \frac{1}{2}\frac{\Delta x_0^2}{u_0^2} x^2 a_0 a_n \\
& + \frac{1}{3}\frac{\Delta x_0^3}{u_0^3} x^3 a_0^2 a_n + \frac{1}{8}\frac{\Delta x_0^4}{u_0^4}\left(-2a_0^3 a_n + a_0^2 a_n^2\right) x^4 \\
& - \frac{1}{6}\frac{\Delta x_0^5}{u_0^5} x^5 a_0^3 a_n^2 - \frac{1}{48}\frac{\Delta x_0^6}{u_0^6} x^6 a_0^3 a_n^3 \Bigg) \exp\left(a_n \frac{\Delta x_0}{u_0} x\right) \\
& + \frac{u_n^2}{a_n^2} + \mathcal{O}\left(a_0^4\right)
\end{aligned} \tag{A.25}$$

$$\begin{aligned}
& = g1 \exp(g_2 x)(1 + g_3 x^2 + g_4 x^3 + g_5 x^4 + g_6 x^5 + g_7 x^6) \\
& + 2 d_8 d_1 \exp(d_2 x)(1 + d_3 x^2 + d_4 x^3 + d_5 x^4 + d_6 x^5 + d_7 x^6) \\
& + g_8 + \mathcal{O}\left(a_0^4\right)
\end{aligned} \tag{A.26}$$

A.3.5 Negative velocity: $|a_n| < \epsilon$

We find an approximation of the square of $\widetilde{\varphi}$ given in (3.90). First, we compute the square and then carry out a Taylor expansion about $a_n = 0$,

$$
\begin{aligned}
\widetilde{\varphi}^2(x, t^n, \Delta t) = {} & \frac{1}{4} a_n^3 \frac{u_n^2}{a_0^5} \ln\left(\frac{\Delta x_0 a_0}{u_0} x + 1\right)^5 \\
& + \left(\frac{7}{12} a_n^2 \frac{u_n^2}{a_0^4} + \frac{5}{4} a_n^3 (\Delta t - T) \frac{u_n^2}{a_0^4} + \frac{5}{4} a_n^3 \Delta x_0 \frac{u_n}{a_0^4}\right) \ln\left(\frac{\Delta x_0 a_0}{u_0} x + 1\right)^4 \\
& + \left(a_n \frac{u_n^2}{a_0^3} + \frac{4}{3} a_n^3 \frac{\Delta x_0^2}{a_0^3} + \frac{7}{3} a_n^2 \frac{u_n^2}{a_0^3} (\Delta t - T) + \frac{7}{3} a_n^2 u_n \frac{\Delta x_0}{a_0^3}\right. \\
& \left. + \frac{5}{2} a_n^3 (\Delta t - T)^2 \frac{u_n^2}{a_0^3} + 5 a_n^3 (\Delta t - T) \Delta x_0 \frac{u_n}{a_0^3}\right) \ln\left(\frac{\Delta x_0 a_0}{u_0} x + 1\right)^3 \\
& + \left(\frac{u_n^2}{a_0^2} + 2 a_n^2 \frac{\Delta x_0^2}{a_0^2} + 3 a_n \frac{u_n^2}{a_0^2} (\Delta t - T) + 3 a_n u_n \frac{\Delta x_0}{a_0^2}\right. \\
& + \frac{7}{2} a_n^2 \frac{u_n^2}{a_0^2} (\Delta t - T)^2 + \frac{5}{2} a_n^3 (\Delta t - T)^3 \frac{u_n^2}{a_0^2} + 4 a_n^3 (\Delta t - T) \frac{\Delta x_0^2}{a_0^2} \\
& \left. + 7 a_n^2 u_n \frac{\Delta x_0}{a_0^2} (\Delta t - T) + \frac{15}{2} a_n^3 (\Delta t - T)^2 \frac{\Delta x_0 u_n}{a_0^2}\right) \ln\left(\frac{\Delta x_0 a_0}{u_0} x + 1\right)^2 \qquad \text{(A.27)} \\
& + \left(2 \frac{u_n^2}{a_0} (\Delta t - T) + 2 u_n \frac{\Delta x_0}{a_0} + 2 a_n \frac{\Delta x_0^2}{a_0} + 3 a_n \frac{u_n^2}{a_0} (\Delta t - T)^2\right. \\
& + \frac{7}{3} a_n^2 \frac{u_n^2}{a_0} (\Delta t - T)^3 + 4 a_n^2 \frac{\Delta x_0^2}{a_0} (\Delta t - T) + \frac{5}{4} a_n^3 (\Delta t - T)^4 \frac{u_n^2}{a_0} \\
& + 4 a_n^3 (\Delta t - T)^2 \frac{\Delta x_0^2}{a_0} + 6 a_n u_n \frac{\Delta x_0}{a_0} (\Delta t - T) + 7 a_n^2 \frac{u_n}{a_0} \Delta x_0 (\Delta t - T)^2 \\
& \left. + 5 a_n^3 (\Delta t - T)^3 \Delta x_0 \frac{u_n}{a_0}\right) \ln\left(\frac{\Delta x_0 a_0}{u_0} x + 1\right) \\
& + u_n^2 (\Delta t - T)^2 + a_n u_n^2 (\Delta t - T)^3 + 2 a_n \Delta x_0^2 (\Delta t - T) \\
& + \frac{7}{12} a_n^2 u_n^2 (\Delta t - T)^4 + 2 a_n^2 \Delta x_0^2 (\Delta t - T)^2 + \frac{7}{3} a_n^2 u_n (\Delta t - T)^3 \Delta x_0 \\
& + \Delta x_0^2 + \frac{1}{4} a_n^3 (\Delta t - T)^5 u_n^2 + \frac{4}{3} a_n^3 (\Delta t - T)^3 \Delta x_0^2 + 2 u_n (\Delta t - T) \Delta x_0 \\
& + 3 a_n u_n (\Delta t - T)^2 \Delta x_0 + \frac{5}{4} a_n^3 (\Delta t - T)^4 \Delta x_0 u_n + \mathcal{O}\left(a_n^4\right) \\
= {} & e_1 + e_2 \ln(d_6 x + 1) + e_3 \ln(d_6 x + 1)^2 + e_4 \ln(d_6 x + 1)^3 \\
& + e_5 \ln(d_6 x + 1)^4 + e_6 \ln(d_6 x + 1)^5 + \mathcal{O}\left(a_n^4\right) \qquad \text{(A.28)}
\end{aligned}
$$

A.3.6 Negative velocity: $|a_0| < \epsilon$ and $|a_n| < \epsilon$

The approximation to the square of the trajectory derived for negative velocity, and small coefficients $|a_0|$ and $|a_n|$ in the departure and arrival cell is determined by first computing the square of $\widetilde{\varphi}$ given in (3.90). Then, the approximation is determined by a series expansions about $a_0 = 0$ and $a_n = 0$,

$$
\begin{aligned}
\widetilde{\varphi}^2(x, t^n, \Delta t) = &\left(\frac{119}{72} a_0^2 a_n^2 \frac{\Delta x^6}{u_0^6} u_n^2\right) x^6 \\
&+ \frac{7}{12} a_0 a_n \frac{\Delta x^5}{u_0^5} u_n \left(7(\Delta t - T) a_0 a_n u_n + 7\Delta x a_0 a_n + 3 a_0 u_n - 2 a_n u_n\right) x^5 \\
&+ \frac{1}{24} \frac{\Delta x^4}{u_0^4} \Big(77(\Delta t - T)^2 a_0^2 a_n^2 u_n^2 + 154(\Delta t - T)\Delta x a_0^2 a_n^2 u_n \\
&+ 66(\Delta t - T) a_0^2 a_n u_n^2 - 84(\Delta t - T) a_0 a_n^2 u_n^2 + 44\Delta x^2 a_0^2 a_n^2 \\
&+ 66\Delta x a_0^2 a_n u_n - 84\Delta x a_0 a_n^2 u_n + 22 a_0^2 u_n^2 - 36 a_0 a_n u_n^2 + 14 a_n^2 u_n^2\Big) x^4 \\
&+ \frac{1}{18} \frac{\Delta x^3}{u_0^3} \Big(14(\Delta t - T)^3 a_0^2 a_n^2 u_n^2 + 42(\Delta t - T)^2 \Delta x a_0^2 a_n^2 u_n \\
&+ 18(\Delta t - T)^2 a_0^2 a_n u_n^2 - 63(\Delta t - T)^2 a_0 a_n^2 u_n^2 + 24(\Delta t - T)\Delta x^2 a_0^2 a_n^2 \\
&+ 36(\Delta t - T)\Delta x a_0^2 a_n u_n - 126(\Delta t - T)\Delta x a_0 a_n^2 u_n \\
&+ 12(\Delta t - T) a_0^2 u_n^2 - 54(\Delta t - T) a_0 a_n u_n^2 + 42(\Delta t - T) a_n^2 u_n^2 \\
&+ 12\Delta x^2 a_0^2 a_n - 36\Delta x^2 a_0 a_n^2 + 12\Delta x a_0^2 u_n - 54\Delta x a_0 a_n u_n \\
&+ 42\Delta x a_n^2 u_n - 18 a_0 u_n^2 + 18 a_n u_n^2\Big) x^3 \\
&- \frac{1}{6} \frac{\Delta x^2}{u_0^2} \Big(7(\Delta t - T)^3 a_0 a_n^2 u_n^2 + 21(\Delta t - T)^2 \Delta x a_0 a_n^2 u_n \\
&+ 9(\Delta t - T)^2 a_0 a_n u_n^2 - 21(\Delta t - T)^2 a_n^2 u_n^2 + 12(\Delta t - T)\Delta x^2 a_0 a_n^2 \\
&+ 18(\Delta t - T)\Delta x a_0 a_n u_n - 42(\Delta t - T)\Delta x a_n^2 u_n + 6(\Delta t - T) a_0 u_n^2 \\
&- 18(\Delta t - T) a_n u_n^2 + 6\Delta x^2 a_0 a_n - 12\Delta x^2 a_n^2 + 6\Delta x a_0 u_n \\
&- 18\Delta x a_n u_n - 6 u_n^2\Big) x^2 \\
&+ \frac{1}{3} \frac{\Delta x}{u_0} \Big(7(\Delta t - T)^3 a_n^2 u_n^2 + 21(\Delta t - T)^2 \Delta x a_n^2 u_n \\
&+ 9(\Delta t - T)^2 a_n u_n^2 + 12(\Delta t - T)\Delta x^2 a_n^2 + 18(\Delta t - T)\Delta x a_n u_n \\
&+ 6(\Delta t - T) u_n^2 + 6\Delta x^2 a_n + 6\Delta x u_n\Big) x \\
&+ \frac{7}{3} a_n^2 (\Delta t - T)^3 \Delta x u_n + \frac{7}{12} a_n^2 (\Delta t - T)^4 u_n^2 + 2 a_n^2 (\Delta t - T)^2 \Delta x^2 \\
&+ \Delta x^2 + (\Delta t - T)^2 u_n^2 + 3 a_n (\Delta t - T)^2 \Delta x u_n + 2(\Delta t - T)\Delta x u_n \\
&+ a_n (\Delta t - T)^3 u_n^2 + 2 a_n (\Delta t - T)\Delta x^2 + \mathcal{O}\left(a_0^3\right) + \mathcal{O}\left(a_n^3\right)
\end{aligned}
\tag{A.29}
$$

$$
= \sum_{i=1}^{7} e_i x^{i-1} + \mathcal{O}\left(a_0^3\right) + \mathcal{O}\left(a_n^3\right) \tag{A.30}
$$

A.4 LIMITS OF INTEGRATION FOR NEGATIVE VELOCITY

The limits of integration in the case of the trajectories that remain within one cell for negative velocity are given by

$$\begin{aligned} \varphi_L &= \frac{1}{\Delta x_0}\left(\frac{u_n}{a_n}\exp(-a_n\Delta t) - \frac{u_n}{a_n}\right) \\ \varphi_R &= 1. \end{aligned} \tag{A.31}$$

If the coefficient a_n is nearly or equal to zero cell, the limits are approximated by

$$\begin{aligned} \varphi_L &= \frac{u_n}{\Delta x_0}\left(-\Delta t + \frac{1}{2}a_n\Delta t^2 - \frac{1}{6}a_n^2\Delta t^3 + \frac{1}{24}a_n^3\Delta t^4\right) \\ \varphi_R &= 1. \end{aligned} \tag{A.32}$$

A similar case environment as for positive velocity (3.251) - (3.254) leads to the limits of integration for negative velocity if trajectories cross more than one grid cell boundary

- if number of cells = 1

$$\begin{aligned} \varphi_L &= \frac{1}{\Delta x_0}\left(\frac{u_0}{a_0}\exp(a_0\,(T_L - \Delta t)) - \frac{u_0}{a_0}\right) \\ \varphi_R &= \frac{1}{\Delta x_0}\left(\frac{u_0}{a_0}\exp(a_0\,(T_R - \Delta t)) - \frac{u_0}{a_0}\right), \end{aligned} \tag{A.33}$$

- if k = 1

$$\begin{aligned} \varphi_L &= 0 \\ \varphi_R &= \frac{1}{\Delta x_0}\left(\frac{u_0}{a_0}\exp(a_0\,(T_R - \Delta t)) - \frac{u_0}{a_0}\right), \end{aligned} \tag{A.34}$$

- if k = no cells

$$\begin{aligned} \varphi_L &= \frac{1}{\Delta x_0}\left(\frac{u_0}{a_0}\exp(a_0\,(T_L - \Delta t)) - \frac{u_0}{a_0}\right) \\ \varphi_R &= 1, \end{aligned} \tag{A.35}$$

- else

$$\begin{aligned} \varphi_L &= 0 \\ \varphi_R &= 1. \end{aligned} \tag{A.36}$$

If a_0 is small, these cases are also approximated by series expansion

$$\begin{aligned} \varphi_L =& \frac{u_0}{\Delta x_0}((T_L - \Delta t) + \frac{1}{2}a_0(T_L - \Delta t)^2 + \frac{1}{6}a_0^2(T_L - \Delta t)^3 \\ &+ \frac{1}{24}a_0^3(T_L - \Delta t)^4), \end{aligned} \tag{A.37}$$

$$\begin{aligned} \varphi_R =& \frac{u_0}{\Delta x_0}((T_R - \Delta t) + \frac{1}{2}a_0(T_R - \Delta t)^2 + \frac{1}{6}a_0^2(T_R - \Delta t)^3 \\ &+ \frac{1}{24}a_0^3(T_R - \Delta t)^4). \end{aligned} \tag{A.38}$$

A.5 DETAILS OF THE STABILITY ANALYSIS OF THE SASLDG METHOD FOR CONSTANT VELOCITY

A.5.1 Eigenvalues

We list the eigenvalues e_1, e_2 and e_3 for matrix M given in (5.158):
The first eigenvalue e_1 is given by

$$e_1 = A_1 + \frac{B_1}{C_1^{1/3}} + e^{-ik}\sigma D_1^{1/3}, \tag{A.39}$$

where the abbreviations are listed in the following,

$$A_1 := 1 - 3\sigma + 4\sigma^3 - 2\sigma^5 + e^{-ik}\sigma(1 - 8\sigma + 16\sigma^2 - 10\sigma^3 + 2\sigma^4) \tag{A.40}$$

$$\begin{aligned} B_1 :=& e^{-ik}(-1+\sigma)^2\sigma\Big(1 - 14\sigma + 65\sigma^2 - 122\sigma^3 + 95\sigma^4 - 32\sigma^5 + 4\sigma^6 \\ &- 2e^{ik}(7 - 34\sigma + 17\sigma^2 + 30\sigma^3 - 5\sigma^4 - 12\sigma^5 + 4\sigma^6) \\ &+ e^{2ik}(-3 + 6\sigma + 9\sigma^2 - 18\sigma^3 - 5\sigma^4 + 8\sigma^5 + 4\sigma^6)\Big), \end{aligned} \tag{A.41}$$

$$\begin{aligned} C_1 :=& \Big((-1+\sigma)^6\Big(- (1 - 14\sigma + 65\sigma^2 - 122\sigma^3 + 95\sigma^4 - 32\sigma^5 + 4\sigma^6 \\ &- 2e^{ik}(7 - 34\sigma + 17\sigma^2 + 30\sigma^3 - 5\sigma^4 - 12\sigma^5 + 4\sigma^6) \\ &+ e^{2ik}(-3 + 6\sigma + 9\sigma^2 - 18\sigma^3 - 5\sigma^4 + 8\sigma^5 + 4\sigma^6))^3 \\ &+ (1 - 21\sigma + 171\sigma^2 - 694\sigma^3 + 1518\sigma^4 - 1839\sigma^5 + 1250\sigma^6 - 477\sigma^7 \\ &+ 96\sigma^8 - 8\sigma^9 + 3e^{ik}(-7 + 83\sigma - 310\sigma^2 + 401\sigma^3 + 18\sigma^4 \\ &- 238\sigma^5 + 9\sigma^6 + 117\sigma^7 - 56\sigma^8 + 8\sigma^9) - 3e^{2ik}(-25 + 45\sigma + 67\sigma^2 \\ &- 96\sigma^3 - 146\sigma^4 + 145\sigma^5 + 68\sigma^6 - 43\sigma^7 - 16\sigma^8 + 8\sigma^9) \\ &+ e^{3ik}(-3 + 27\sigma - 60\sigma^2 + 3\sigma^3 + 90\sigma^4 - 12\sigma^5 - 73\sigma^6 - 3\sigma^7 \\ &+ 24\sigma^8 + 8\sigma^9))^2\Big)\Big)^{1/2} - (-1+\sigma)^3\Big(1 - 21\sigma + 171\sigma^2 \\ &- 694\sigma^3 + 1518\sigma^4 - 1839\sigma^5 + 1250\sigma^6 - 477\sigma^7 + 96\sigma^8 - 8\sigma^9 \\ &+ 3e^{ik}(-7 + 83\sigma - 310\sigma^2 + 401\sigma^3 + 18\sigma^4 - 238\sigma^5 + 9\sigma^6 \\ &+ 117\sigma^7 - 56\sigma^8 + 8\sigma^9) - 3e^{2ik}(-25 + 45\sigma + 67\sigma^2 - 96\sigma^3 \\ &- 146\sigma^4 + 145\sigma^5 + 68\sigma^6 - 43\sigma^7 - 16\sigma^8 + 8\sigma^9) \\ &+ e^{3ik}(-3 + 27\sigma - 60\sigma^2 + 3\sigma^3 + 90\sigma^4 - 12\sigma^5 - 73\sigma^6 - 3\sigma^7 \\ &+ 24\sigma^8 + 8\sigma^9)\Big), \end{aligned} \tag{A.42}$$

$$
\begin{aligned}
D_1 :=& \Big((-1+\sigma)^6 \Big(-(1 - 14\sigma + 65\sigma^2 - 122\sigma^3 + 95\sigma^4 - 32\sigma^5 + 4\sigma^6 \\
&- 2e^{ik}(7 - 34\sigma + 17\sigma^2 + 30\sigma^3 - 5\sigma^4 - 12\sigma^5 + 4\sigma^6) \\
&+ e^{2ik}(-3 + 6\sigma + 9\sigma^2 - 18\sigma^3 - 5\sigma^4 + 8\sigma^5 + 4\sigma^6))^3 \\
&+ (1 - 21\sigma + 171\sigma^2 - 694\sigma^3 + 1518\sigma^4 - 1839\sigma^5 + 1250\sigma^6 \\
&- 477\sigma^7 + 96\sigma^8 - 8\sigma^9 + 3e^{ik}(-7 + 83\sigma - 310\sigma^2 + 401\sigma^3 \\
&+ 18\sigma^4 - 238\sigma^5 + 9\sigma^6 + 117\sigma^7 - 56\sigma^8 + 8\sigma^9) \\
&- 3e^{2ik}(-25 + 45\sigma + 67\sigma^2 - 96\sigma^3 - 146\sigma^4 + 145\sigma^5 + 68\sigma^6 \\
&- 43\sigma^7 - 16\sigma^8 + 8\sigma^9) + e^{3ik}(-3 + 27\sigma - 60\sigma^2 + 3\sigma^3 \\
&+ 90\sigma^4 - 12\sigma^5 - 73\sigma^6 - 3\sigma^7 + 24\sigma^8 + 8\sigma^9))^2 \Big) \Big)^{1/2} \\
&- (-1+\sigma)^3 \Big(1 - 21\sigma + 171\sigma^2 - 694\sigma^3 + 1518\sigma^4 - 1839\sigma^5 \\
&+ 1250\sigma^6 - 477\sigma^7 + 96\sigma^8 - 8\sigma^9 + 3e^{ik}(-7 + 83\sigma - 310\sigma^2 \\
&+ 401\sigma^3 + 18\sigma^4 - 238\sigma^5 + 9\sigma^6 + 117\sigma^7 - 56\sigma^8 + 8\sigma^9) \\
&- 3e^{2ik}(-25 + 45\sigma + 67\sigma^2 - 96\sigma^3 - 146\sigma^4 + 145\sigma^5 + 68\sigma^6 \\
&- 43\sigma^7 - 16\sigma^8 + 8\sigma^9) + e^{3ik}(-3 + 27\sigma - 60\sigma^2 + 3\sigma^3 \\
&+ 90\sigma^4 - 12\sigma^5 - 73\sigma^6 - 3\sigma^7 + 24\sigma^8 + 8\sigma^9) \Big).
\end{aligned}
\tag{A.43}
$$

The second eigenvalue e_2 yields

$$
e_2 = A_2 + 2^{-2/3}\frac{B_2}{C_2^{1/3}} - e^{-ik}\frac{1 - i\sqrt{3}}{6 \cdot 2^{1/3}} D_2^{1/3}, \tag{A.44}
$$

with the abbreviations

$$
A_2 := 1 - 3\sigma + 4\sigma^3 - 2\sigma^5 + e^{-ik}\sigma(1 - 8\sigma + 16\sigma^2 - 10\sigma^3 + 2\sigma^4) \tag{A.45}
$$

$$
\begin{aligned}
B_2 :=& -3(1 + i\sqrt{3})(-1+\sigma)^2\sigma^2 \Big(1 - 14\sigma + 65\sigma^2 - 122\sigma^3 \\
&+ 95\sigma^4 - 32\sigma^5 + 4\sigma^6 - 2e^{ik}(7 - 34\sigma + 17\sigma^2 + 30\sigma^3 - 5\sigma^4 - 12\sigma^5 \\
&+ 4\sigma^6) + e^{2ik}(-3 + 6\sigma + 9\sigma^2 - 18\sigma^3 - 5\sigma^4 + 8\sigma^5 + 4\sigma^6) \Big),
\end{aligned}
\tag{A.46}
$$

$$
\begin{aligned}
C_2 :=& \Big(2916(-1+\sigma)^6\sigma^6\Big(-(1-14\sigma+65\sigma^2-122\sigma^3+95\sigma^4-32\sigma^5+4\sigma^6 \\
&-2e^{ik}(7-34\sigma+17\sigma^2+30\sigma^3-5\sigma^4-12\sigma^5+4\sigma^6) \\
&+e^{2ik}(-3+6\sigma+9\sigma^2-18\sigma^3-5\sigma^4+8\sigma^5+4\sigma^6))^3 \\
&+(1-21\sigma+171\sigma^2-694\sigma^3+1518\sigma^4-1839\sigma^5+1250\sigma^6-477\sigma^7 \\
&+96\sigma^8-8\sigma^9+3e^{ik}(-7+83\sigma-310\sigma^2+401\sigma^3+18\sigma^4-238\sigma^5 \\
&+9\sigma^6+117\sigma^7-56\sigma^8+8\sigma^9)-3e^{2ik}(-25+45\sigma+67\sigma^2-96\sigma^3 \\
&-146\sigma^4+145\sigma^5+68\sigma^6-43\sigma^7-16\sigma^8+8\sigma^9)+e^{3ik}(-3+27\sigma \\
&-60\sigma^2+3\sigma^3+90\sigma^4-12\sigma^5-73\sigma^6-3\sigma^7+24\sigma^8+8\sigma^9))^2\Big)\Big)^{1/2} \\
&-54(-1+\sigma)^3\sigma^3\Big(1-21\sigma+171\sigma^2-694\sigma^3+1518\sigma^4-1839\sigma^5 \\
&+1250\sigma^6-477\sigma^7+96\sigma^8-8\sigma^9+3e^{ik}(-7+83\sigma-310\sigma^2+401\sigma^3 \\
&+18\sigma^4-238\sigma^5+9\sigma^6+117\sigma^7-56\sigma^8+8\sigma^9)-3e^{2ik}(-25+45\sigma \\
&+67\sigma^2-96\sigma^3-146\sigma^4+145\sigma^5+68\sigma^6-43\sigma^7-16\sigma^8+8\sigma^9) \\
&+e^{3ik}(-3+27\sigma-60\sigma^2+3\sigma^3+90\sigma^4-12\sigma^5-73\sigma^6-3\sigma^7 \\
&+24\sigma^8+8\sigma^9)\Big),
\end{aligned}
\tag{A.47}
$$

$$
\begin{aligned}
D_2 :=& \Big(2916(-1+\sigma)^6\sigma^6\Big(-(1-14\sigma+65\sigma^2-122\sigma^3+95\sigma^4-32\sigma^5 \\
&+4\sigma^6-2e^{ik}(7-34\sigma+17\sigma^2+30\sigma^3-5\sigma^4-12\sigma^5+4\sigma^6) \\
&+e^{2ik}(-3+6\sigma+9\sigma^2-18\sigma^3-5\sigma^4+8\sigma^5+4\sigma^6)\Big)^3 \\
&+(1-21\sigma+171\sigma^2-694\sigma^3+1518\sigma^4-1839\sigma^5+1250\sigma^6 \\
&-477\sigma^7+96\sigma^8-8\sigma^9+3e^{ik}(-7+83\sigma-310\sigma^2+401\sigma^3 \\
&+18\sigma^4-238\sigma^5+9\sigma^6+117\sigma^7-56\sigma^8+8\sigma^9) \\
&-3e^{2ik}(-25+45\sigma+67\sigma^2-96\sigma^3-146\sigma^4+145\sigma^5+68\sigma^6 \\
&-43\sigma^7-16\sigma^8+8\sigma^9)+e^{3ik}(-3+27\sigma-60\sigma^2+3\sigma^3+90\sigma^4 \\
&-12\sigma^5-73\sigma^6-3\sigma^7+24\sigma^8+8\sigma^9))^2)\Big)^{1/2} \\
&-54(-1+\sigma)^3\sigma^3\Big(1-21\sigma+171\sigma^2-694\sigma^3+1518\sigma^4-1839\sigma^5 \\
&+1250\sigma^6-477\sigma^7+96\sigma^8-8\sigma^9 \\
&+3e^{ik}(-7+83\sigma-310\sigma^2+401\sigma^3+18\sigma^4-238\sigma^5+9\sigma^6+117\sigma^7 \\
&-56\sigma^8+8\sigma^9)-3e^{2ik}(-25+45\sigma+67\sigma^2-96\sigma^3-146\sigma^4 \\
&+145\sigma^5+68\sigma^6-43\sigma^7-16\sigma^8+8\sigma^9)+e^{3ik}(-3+27\sigma-60\sigma^2 \\
&+3\sigma^3+90\sigma^4-12\sigma^5-73\sigma^6-3\sigma^7+24\sigma^8+8\sigma^9)\Big).
\end{aligned}
\tag{A.48}
$$

The third eigenvalue e_3 is given by

$$
e_3 = A_3 + 2^{-2/3}\frac{B_3}{C_3^{1/3}} - \frac{(1+i\sqrt{3})e^{-ik}}{6\cdot 2^{1/3}}D_3^{1/3}, \tag{A.49}
$$

where we have used the abbreviations

$$
A_3 := 1+\sigma-8\sigma^2+16\sigma^3-10\sigma^4+2\sigma^5+e^{ik}(-3\sigma+4\sigma^3-2\sigma^5) \tag{A.50}
$$

$$
\begin{aligned}
B_3 :=& -3(1-i\sqrt{3})e^{-ik}(-1+\sigma)^2\sigma^2\Big(1-14\sigma+65\sigma^2-122\sigma^3+95\sigma^4\\
&-32\sigma^5+4\sigma^6-2e^{ik}(7-34\sigma+17\sigma^2+30\sigma^3-5\sigma^4-12\sigma^5+4\sigma^6)\\
&+e^{2ik}(-3+6\sigma+9\sigma^2-18\sigma^3-5\sigma^4+8\sigma^5+4\sigma^6)\Big),
\end{aligned}
\tag{A.51}
$$

$$
\begin{aligned}
C_3 :=& \Big(2916(-1+\sigma)^6\sigma^6\Big(-(1-14\sigma+65\sigma^2-122\sigma^3+95\sigma^4-32\sigma^5\\
&+4\sigma^6-2e^{ik}(7-34\sigma+17\sigma^2+30\sigma^3-5\sigma^4-12\sigma^5+4\sigma^6)\\
&+e^{2ik}(-3+6\sigma+9\sigma^2-18\sigma^3-5\sigma^4+8\sigma^5+4\sigma^6))^3\\
&+(1-21\sigma+171\sigma^2-694\sigma^3+1518\sigma^4-1839\sigma^5+1250\sigma^6\\
&-477\sigma^7+96\sigma^8-8\sigma^9+3e^{ik}(-7+83\sigma-310\sigma^2+401\sigma^3\\
&+18\sigma^4-238\sigma^5+9\sigma^6+117\sigma^7-56\sigma^8+8\sigma^9)\\
&-3e^{2ik}(-25+45\sigma+67\sigma^2-96\sigma^3-146\sigma^4+145\sigma^5+68\sigma^6\\
&-43\sigma^7-16\sigma^8+8\sigma^9)+e^{3ik}(-3+27\sigma-60\sigma^2+3\sigma^3+90\sigma^4\\
&-12\sigma^5-73\sigma^6-3\sigma^7+24\sigma^8+8\sigma^9))^2\Big)\Big)^{1/2}\\
&-54(-1+\sigma)^3\sigma^3\Big(1-21\sigma+171\sigma^2-694\sigma^3+1518\sigma^4-1839\sigma^5\\
&+1250\sigma^6-477\sigma^7+96\sigma^8-8\sigma^9\\
&+3e^{ik}(-7+83\sigma-310\sigma^2+401\sigma^3+18\sigma^4-238\sigma^5+9\sigma^6+117\sigma^7\\
&-56\sigma^8+8\sigma^9)-3e^{2ik}(-25+45\sigma+67\sigma^2-96\sigma^3-146\sigma^4+145\sigma^5\\
&+68\sigma^6-43\sigma^7-16\sigma^8+8\sigma^9)+e^{3ik}(-3+27\sigma-60\sigma^2+3\sigma^3\\
&+90\sigma^4-12\sigma^5-73\sigma^6-3\sigma^7+24\sigma^8+8\sigma^9)\Big),
\end{aligned}
\tag{A.52}
$$

$$
\begin{aligned}
D_3 :=& \Big(2916(-1+\sigma)^6\sigma^6\Big(-(1-14\sigma+65\sigma^2-122\sigma^3+95\sigma^4-32\sigma^5\\
&+4\sigma^6-2e^{ik}(7-34\sigma+17\sigma^2+30\sigma^3-5\sigma^4-12\sigma^5+4\sigma^6)\\
&+e^{2ik}(-3+6\sigma+9\sigma^2-18\sigma^3-5\sigma^4+8\sigma^5+4\sigma^6))^3\\
&+(1-21\sigma+171\sigma^2-694\sigma^3+1518\sigma^4-1839\sigma^5+1250\sigma^6\\
&-477\sigma^7+96\sigma^8-8\sigma^9+3e^{ik}(-7+83\sigma-310\sigma^2+401\sigma^3\\
&+18\sigma^4-238\sigma^5+9\sigma^6+117\sigma^7-56\sigma^8+8\sigma^9)\\
&-3e^{2ik}(-25+45\sigma+67\sigma^2-96\sigma^3-146\sigma^4+145\sigma^5+68\sigma^6\\
&-43\sigma^7-16\sigma^8+8\sigma^9)+e^{3ik}(-3+27\sigma-60\sigma^2+3\sigma^3\\
&+90\sigma^4-12\sigma^5-73\sigma^6-3\sigma^7+24\sigma^8+8\sigma^9))^2\Big)\Big)^{1/2}\\
&-54(-1+\sigma)^3\sigma^3\Big(1-21\sigma+171\sigma^2-694\sigma^3+1518\sigma^4\\
&-1839\sigma^5+1250\sigma^6-477\sigma^7+96\sigma^8-8\sigma^9\\
&+3e^{ik}(-7+83\sigma-310\sigma^2+401\sigma^3+18\sigma^4-238\sigma^5+9\sigma^6\\
&+117\sigma^7-56\sigma^8+8\sigma^9)-3e^{2ik}(-25+45\sigma+67\sigma^2-96\sigma^3\\
&-146\sigma^4+145\sigma^5+68\sigma^6-43\sigma^7-16\sigma^8+8\sigma^9)+e^{3ik}(-3+27\sigma\\
&-60\sigma^2+3\sigma^3+90\sigma^4-12\sigma^5-73\sigma^6-3\sigma^7+24\sigma^8+8\sigma^9)\Big).
\end{aligned}
\tag{A.53}
$$

A.5.2 Reformulation of the dispersion error

When we compute the limit in 5.178, which is

$$\lim_{\sigma\to 0} \tilde{e}_{\text{disp}} = \lim_{\sigma\to 0} \frac{\arctan\left(-\frac{\operatorname{Im} e_1}{\operatorname{Re} e_1}\right)}{\sigma\alpha_k}. \tag{A.54}$$

we find that both numerator and denominator both tend to zero. Hence, we can apply L'Hôpital's rule. We have

$$\lim_{\sigma\to 0} \tilde{e}_{\text{disp}} = \lim_{\sigma\to 0} \frac{\frac{\partial}{\partial\sigma}\arctan\left(-\frac{\operatorname{Im} e_1}{\operatorname{Re} e_1}\right)}{\frac{\partial}{\partial\sigma}\sigma\alpha_k} \tag{A.55}$$

$$= \lim_{\sigma\to 0} \frac{\frac{\partial}{\partial\sigma}\arctan\left(-\frac{\operatorname{Im} e_1}{\operatorname{Re} e_1}\right)}{\alpha_k}. \tag{A.56}$$

Further, for the arctan it holds

$$\frac{\partial}{\partial\sigma}\arctan\left(-\frac{\operatorname{Im} e_1}{\operatorname{Re} e_1}\right) = \frac{\frac{\partial}{\partial\sigma}\left(-\frac{\operatorname{Im} e_1}{\operatorname{Re} e_1}\right)}{1+\frac{\operatorname{Im} e_1^2}{\operatorname{Re} e_1^2}}. \tag{A.57}$$

The quotient rule yields

$$\frac{\partial}{\partial\sigma}\left(-\frac{\operatorname{Im} e_1}{\operatorname{Re} e_1}\right) = -\frac{\operatorname{Re} e_1\left(\frac{\partial}{\partial\sigma}\operatorname{Im} e_1\right) - \operatorname{Im} e_1\left(\frac{\partial}{\partial\sigma}\operatorname{Re} e_1\right)}{\operatorname{Re} e_1^2} \tag{A.58}$$

$$= \frac{\operatorname{Im} e_1\left(\frac{\partial}{\partial\sigma}\operatorname{Re} e_1\right) - \operatorname{Re} e_1\left(\frac{\partial}{\partial\sigma}\operatorname{Im} e_1\right)}{\operatorname{Re} e_1^2}. \tag{A.59}$$

Taking all parts together, we obtain

$$\frac{\partial}{\partial\sigma}\arctan\left(-\frac{\operatorname{Im} e_1}{\operatorname{Re} e_1}\right) = \frac{\frac{\operatorname{Im} e_1\left(\frac{\partial}{\partial\sigma}\operatorname{Re} e_1\right) - \operatorname{Re} e_1\left(\frac{\partial}{\partial\sigma}\operatorname{Im} e_1\right)}{\operatorname{Re} e_1^2}}{1+\frac{\operatorname{Im} e_1^2}{\operatorname{Re} e_1^2}} \tag{A.60}$$

$$= \frac{\operatorname{Im} e_1\left(\frac{\partial}{\partial\sigma}\operatorname{Re} e_1\right) - \operatorname{Re} e_1\left(\frac{\partial}{\partial\sigma}\operatorname{Im} e_1\right)}{\operatorname{Re} e_1^2 + \operatorname{Im} e_1^2}. \tag{A.61}$$

With the help of a symbolic mathematical computation program we find that

$$\lim_{\sigma\to 0} \operatorname{Re} e_1^2 + \operatorname{Im} e_1^2 = 1, \tag{A.62}$$

$$\lim_{\sigma\to 0} \operatorname{Im} e_1 = 0, \tag{A.63}$$

$$\lim_{\sigma\to 0} \operatorname{Re} e_1 = 1. \tag{A.64}$$

So, after plugging in these results the computation of the dispersion error boils down to

$$\lim_{\sigma\to 0} \tilde{e}_{\text{disp}} = -\frac{\lim_{\sigma\to 0}\frac{\partial}{\partial\sigma}\operatorname{Im} e_1}{\alpha_k}. \tag{A.65}$$

Unfortunately, the expression is still to complicated to be solved in this form. A remedy is found in Theorem 7 in [28] by Lancaster. It opens a way to change the order of the computations, namely to first differentiate matrix M with respect to σ, then find the limit as $\sigma \to 0$ and afterwards determine the eigenvalue. Theorem 7 states that if $\mu(\sigma)$ depends on parameter σ and it is an eigenvalue of matrix $A(\sigma)$,

then under certain assumptions the derivative of eigenvalue $\mu(\sigma)$ with respect to σ is an eigenvalue of the derivative $A(\sigma)$.

The requirements are described in Theorem A.5.1 and Theorem A.5.2, the main statement in given in Theorem A.5.3.

Theorem A.5.1 (Lancaster, Theorem 4) *Let μ_1 be an eigenvalue of $A(\lambda_0)$ and X_1 be the subspace of right eigenvectors of μ_1 while $\mu(\lambda)$ is an eigenvalue of $A(\lambda)$ for which $\mu(\lambda) \to \mu_1$ as $\lambda \to \lambda_0$. If the elements of $A(\lambda)$ are regular in some neighborhood of λ_0 while $A(\lambda_0)$ is similar to a diagonal matrix then, given $\epsilon > 0$, there exists a δ such that for any right eigenvector $\mathbf{x}$ of $\mu(\lambda)$, $\mathbf{x} = \vartheta_1 + \vartheta_2$ where $\vartheta_1 \in X_1$ and $\left(\overline{\vartheta_2}'\vartheta_2\right)^{\frac{1}{2}} < \epsilon$ provided $|\lambda - \lambda_0| < \delta$.*

Theorem A.5.2 (Lancaster, Theorem 6) *If, in the hypothesis of theorem 4, $A^{(q)}(\lambda_0)$ is the first non-vanishing derivative of $A(\lambda)$ at $\lambda = \lambda_0$, then the n eigenvalues $\mu(\lambda)$ of $A(\lambda)$ are differentiable at least q times at λ_0 and their first $q-1$ derivatives all vanish at λ_0.*

Theorem A.5.3 (Lancaster, Theorem 7) *With the assumption of theorem 6, let $\mu(\lambda_0)$ be an eigenvalue of $A(\lambda_0)$ with multiplicity α and let the columns of the $n \times \alpha$ matrices X_α, Y_α span X_1, Y_1 respectively. If these matrices are chosen so that $Y_\alpha' X_\alpha = I_\alpha$, then the α derivatives $\mu^{(q)}(\lambda_0)$ (of the α eigenvalues which coincide at λ_0) are the eigenvalues of the matrix $Y_\alpha' A^{(q)} X_\alpha$.*

According to Theorem A.5.3 we now can to tackle the computation of

$$\lim_{\sigma \to 0} \frac{\partial}{\partial \sigma} \operatorname{Im} e_1 \tag{A.66}$$

differently. As stated above we first differentiate matrix M with respect to σ, then let σ go to zero and eventually compute the eigenvalues of

$$M_D := \lim_{\sigma \to 0} \frac{\partial}{\partial \sigma} M \tag{A.67}$$

$$= \begin{pmatrix} -1 + e^{-i\alpha_k} & -1 + e^{-i\alpha_k} & -1 + e^{-i\alpha_k} \\ 3 - 3e^{-i\alpha_k} & -3 - 3e^{-i\alpha_k} & -3 - 3e^{-i\alpha_k} \\ -5 + 5e^{-i\alpha_k} & 5 + 5e^{-i\alpha_k} & -5 + 5e^{-i\alpha_k} \end{pmatrix} \tag{A.68}$$

The imaginary part can be determined at the very end, because this operator is linear. The first eigenvalue of M_D is given by

$$e_D = e^{-i\alpha_k} \left(\sqrt[3]{p} - \frac{126e^{i\alpha_k} + 27e^{2i\alpha_k} - 9}{9\sqrt[3]{p}} - 3e^{i\alpha_k} + 1 \right), \tag{A.69}$$

where

$$\begin{aligned} p = {} & 2\sqrt{3e^{2i\alpha_k} - 166e^{3i\alpha_k} + 1872e^{4i\alpha_k} - 18e^{5i\alpha_k} + 9e^{6i\alpha_k}} \\ & -21e^{i\alpha_k} + 75e^{2i\alpha_k} - 3e^{3i\alpha_k} + 1. \end{aligned} \tag{A.70}$$

For this expression a symbolic mathematical computation program can finally apply a Taylor expansion about $\alpha_k = 0$. We obtain

$$e_D = -i\alpha_k - \frac{\alpha_k^6}{7200} - \frac{i\alpha_k^7}{42000} + \mathcal{O}\left(\alpha_k^8\right). \tag{A.71}$$

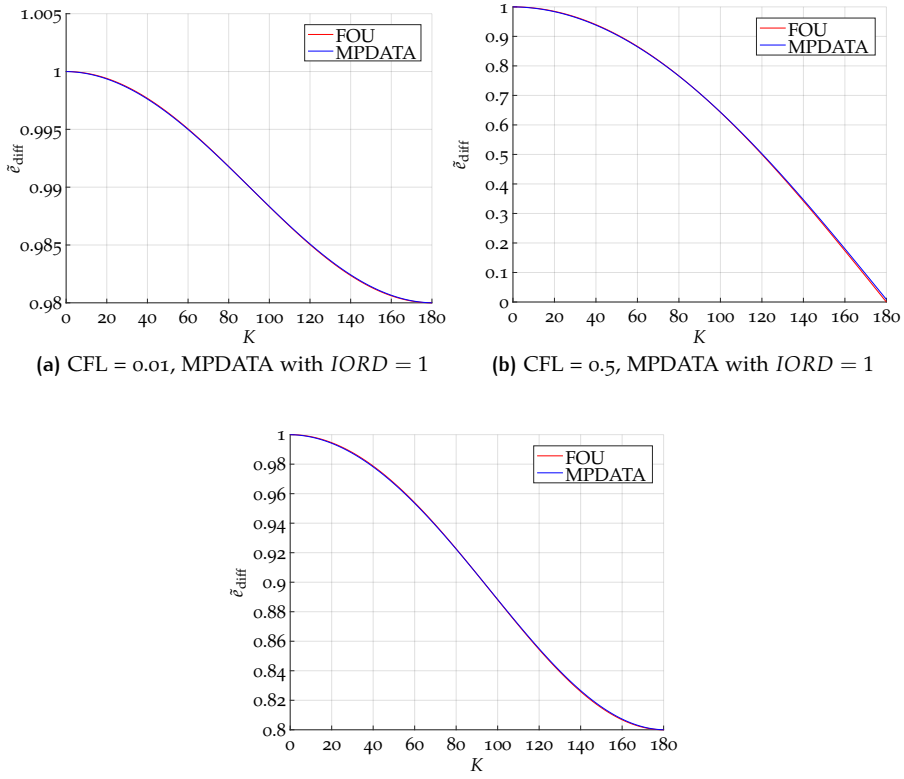
This result can be plugged in (A.65). The dispersion error computed from the first eigenvalue yields the result

$$\lim_{\sigma \to 0} \tilde{e}_{\text{disp}} = -\operatorname{Im}\left(\frac{-\frac{i\alpha_k^7}{42000} - \frac{\alpha_k^6}{7200} - i\alpha_k}{\alpha_k} \right) \tag{A.72}$$

$$= 1 + \frac{\alpha_k^6}{42000} + \mathcal{O}\left(\alpha_k^7\right). \tag{A.73}$$

A.6 MPDATA SPECTRAL ANALYSIS

The equality of the diffusion error of MPDATA without corrections and the respective errors of the FOU method can be seen in Figure A.1.



(a) CFL = 0.01, MPDATA with $IORD = 1$

(b) CFL = 0.5, MPDATA with $IORD = 1$

(c) CFL = 0.9, MPDATA with $IORD = 1$

Figure A.1: Relative diffusion error $\tilde{e}_{\text{diff}}$ of MPDATA without corrections and the FOU method for different CFL numbers. Note the different scales of $\tilde{e}_{\text{diff}}$.

A.7 MATLAB BENCHMARK TEST RESULT

test	LU	FFT	ODE	Sparse	2-D	3-D
	0.2078	0.1251	0.0596	0.0961	0.1678	0.1397
	0.2083	0.1250	0.0608	0.0959	0.1674	0.1397
	0.2097	0.1262	0.0609	0.0956	0.1675	0.1387
	0.2062	0.1250	0.0604	0.0955	0.1677	0.1394
	0.2065	0.1234	0.0597	0.0958	0.1680	0.1435
	0.2073	0.1259	0.0605	0.0963	0.1673	0.1385
	0.2087	0.1255	0.0603	0.0959	0.1668	0.1383
	0.2081	0.1250	0.0598	0.0958	0.1672	0.1409
	0.2081	0.1246	0.0608	0.0960	0.1679	0.1379
	0.2077	0.1251	0.0604	0.0959	0.1672	0.1401

Table A.1: The measurements of the execution speed of six different tasks for 10 runs of the MATLAB benchmark test.

SUMMARY

The main objective of this thesis is to develop a numerical method, known as the SASLDG method, that solves the compressible linear advection equation in a semi-analytical way using a semi-Lagrangian approach and applying ideas from the discontinuous Galerkin (DG) methods. In order to achieve physically meaningful solutions, mass conservation is required. For the same reason, the option of slope limiting and thus positivity preservation are also necessary. Furthermore, high-order accuracy and no time step restriction are required.

First, the basic methods underpinning the SASLDG method are introduced. We describe a method developed independently by Prather and van Leer. This so-called second-order moments method is a preliminary stage of the SASLDG method. Both methods are equivalent for CFL numbers less than or equal to one and with constant velocity. We then introduce DG methods to generalize the concept of high-order accuracy of numerical solutions using a polynomial representation for each grid cell. One drawback of this class of methods is the strict time step restriction. In contrast, semi-Lagrangian methods allow time steps of arbitrary size without violating stability properties. We also introduce and analyze the numerical scheme multidimensional positive definite advection transport algorithm (MPDATA).

Following the semi-Lagrangian notion, the advected quantity is tracked analytically along the trajectories. It is represented as piecewise polynomials of degree two. A condition for this approach is the restriction of the given velocity field to a piecewise linear distribution. This enables the derivation of the exact solution, which is in general not in polynomial form. Therefore, much like with DG methods, a projection onto the polynomial space is carried out. A deviation from the solution only occurs in the projection step. The SASLDG method involves rigorously computing every step of the solution to the linear advection equation analytically. However, this concept places many conditions on the algorithm, which require the application of different branches of the SASLDG method. To prevent the cancellation of significant digits, further exceptions have to be made and resolved by alternative computations.

A thorough analysis of the SASLDG method proves its consistency and stability. Third-order accuracy for advection with constant velocity and second-order accuracy for variable velocity is demonstrated analytically and confirmed by numerical convergence tests.

Test cases in one- and two- space dimensions are conducted to show the performance of the SASLDG method. High accuracy is shown in tests using smooth and discontinuous initial values. The method's ability to handle an irregular grid and arbitrary CFL numbers is assessed. Tests in 1D show that an increase of the CFL number results in smaller maximum- and l_1-errors. This feature cannot be transferred to the 2D case since the extension is done via operator splitting, where accuracy increases with decreasing time step sizes. Numerous physical phenomena, e.g. in the dynamics of the atmosphere or the ocean, can occur on different scales of the space dimensions. This can lead to computational grids with higher resolution and grid cells sizes in the vertical direction than in the horizontal direction. To address this scenario, we develop a hybrid method which employs MPDATA and a modified version of the SASLDG method. This scheme can compute the solution to the advection equation on grids with high aspect ratio without strict time step restrictions in the vertical direction. Overall, the SASLDG method shows promising results and is worthy of further development.

ZUSAMMENFASSUNG

Das Ziel dieser Arbeit ist die Entwicklung einer numerischen Methode, die die kompressible lineare Advektionsgleichung auf eine semi-analytische Weise mit einem semi-Lagrangian Ansatz unter Verwendung von Ideen der Discontinuous Galerkin (DG) Methoden löst, hier als SASLDG Methode bezeichnet. Eine Vorgabe für das Verfahren ist die Massenerhaltung, um eine physikalisch sinnvolle Lösung zu garantieren. Aus diesem Grund soll auch "slope limiting" möglich sein und die Positivität der Lösung erhalten bleiben. Hohe Genauigkeit und unbeschränkte Zeitschrittgröße sind weitere Bedingungen.

Zunächst werden die Methoden erläutert, die der SASLDG Methode zugrunde liegen. Eine von Prather und van Leer unabhängig voneinander entwickelte Methode wird beschrieben. Diese sogennante Second-Order Moments Methode ist eine Vorstufe zur SASLDG Methode. Für CFL Zahlen, die kleiner gleich eins sind, und für eine konstante Advektionsgeschwindigkeit sind die Verfahren identisch. DG Methoden verallgemeinern das Konzept der hohen Genauigkeit durch Verwendung von Polynomen höherer Ordnung. Diese haben jedoch den Nachteil strikter Zeitschrittbeschränkung. Semi-Lagrangian Verfahren lassen Zeitschritte beliebiger Größe zu, ohne Stabilitätsbedingungen zu verletzen. Schließlich wird die Methode Multidimensional Positive Definite Advection Transport Algorithm (MPDATA) vorgestellt und analysiert.

Im Sinne des semi-Lagrangian Ansatzes wird die advektierte Größe analytisch exakt entlang von Trajektorien transportiert. Sie wird durch ein Polynom zweiten Grades dargestellt. Eine Bedingung für dieses Vorgehen ist die Beschränkung des Geschwindigkeitsfeldes auf stückweise lineare Funktionen. So wird die Herleitung der analytischen Lösung ermöglicht, die im Allgemeinen nicht polynomiell ist. Durch einen Projektionsschritt, ähnlich dem Vorgehen bei DG Verfahren, wird die Lösung auf ein Polynom projiziert. Nur dadurch entstehen Abweichungen zur analytischen Lösung. Die Idee der SASLDG Methode besteht darin, alle Schritte rigoros analytisch durchzuführen. Diese Methodik stellt allerdings komplexe Bedingungen an den Algorithmus und führt stellenweise zum Problem der Auslöschung. Dadurch sind Verzweigungen notwendig sowie alternative Berechnungswege.

Konsistenz und Stabilität wird durch eine Analyse des SASLDG Verfahrens gezeigt. Das Verfahren konvergiert mit zweiter Ordnung für variable Geschwindigkeit und mit dritter Ordnung für konstante Geschwindigkeit. Dies wird analytisch hergeleitet und mithilfe numerischer Konvergenztests bestätigt.

Die hohe Genauigkeit des SASLDG Verfahrens wird durch numerische Tests in 1D und 2D gezeigt. Dabei werden glatte Anfangswerte sowie Daten mit Diskontinuitäten verwendet. Tests in 1D liefern Ergebnisse auf regel- und unregelmäßigem Gitter sowie mit verschiedenen CFL Zahlen. Daraus resultiert, dass mit größerer CFL Zahl l_∞- und l_1-Fehler sinken. Dieses Ergebnis gilt jedoch nicht für 2D-Testfälle aufgrund der Verwendung von "operator splitting", für das die Genauigkeit bei kleineren Schrittweiten steigt. Zahlreiche physikalische Phänomene, beispielsweise im Bereich der Dynamik der Atmosphäre und des Ozeans, erstrecken sich über unterschiedliche Raumskalen. Das kann zu Rechengittern führen, die in vertikaler Richtung eine viel höhere Auflösung mit kleinerer Gitterweite als in horizontaler Richtung erfordern. Eine dafür entwickelte hybride Methode, eine Kombination aus MPDATA und einer angepassten Variante des SASLDG Verfahrens, kann auf diesen Gittern ohne strikte Zeitschrittbeschränkung in vertikaler Richtung Lösungen berechnen. Insgesamt zeigt die in dieser Arbeit hergeleitete SASLDG Methode vielversprechende Resultate, die eine Weiterentwicklung des Verfahrens erstrebenswert machen.

BIBLIOGRAPHY

[1] D. G. Andrews, J. R. Holton, and C. B. Leovy. *Middle Atmosphere Dynamics*. International geophysics series. Academic Press, 1987 (cit. on p. 2).

[2] J. R. Bates and A. McDonald. "Multiply-Upstream, semi-Lagrangian Advective Schemes: Analysis and Application to a Multi-Level Primitive Equation Model". In: *Mon. Wea. Rev.* 110.12 (1982), pp. 1831–1842 (cit. on p. 30).

[3] R. Bermejo. "On the Equivalence of semi-Lagrangian Schemes and Particle-in-Cell Finite Element Methods". In: *Mon. Wea. Rev.* 118.4 (1990), pp. 979–987 (cit. on p. 30).

[4] R. Bermejo and A. Staniforth. "The Conversion of semi-Lagrangian Advection Schemes to Quasi-Monotone Schemes". In: *Monthly Weather Review* 120.11 (1992), pp. 2622–2632 (cit. on p. 30).

[5] L. Bonaventura. "An introduction to semi-Lagrangian methods for geophysical scale flows". In: *Lecture Notes, ERCOFTAC Leonhard Euler Lectures, SAM-ETH Zurich* (2004) (cit. on p. 30).

[6] J. Boris and D. Book. "Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works". In: *Journal of Computational Physics* 11.1 (1973), pp. 38–69 (cit. on p. 5).

[7] V. M. Canuto, Y. Cheng, and A. M. Howard. "Vertical diffusivities of active and passive tracers". In: *Ocean Modelling* 36.3 (2011), pp. 198–207 (cit. on p. 2).

[8] G. Chavent and B. Cockburn. "The local projection $P^0 - P^1$-discontinuous-Galerkin finite element method for scalar conservation laws". eng. In: *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique* 23.4 (1989), pp. 565–592 (cit. on p. 24).

[9] B. Cockburn. "An introduction to the Discontinuous Galerkin method for convection-dominated problems". In: *Advanced Numerical Approximation of Nonlinear Hyperbolic Equations: Lectures given at the 2nd Session of the Centro Internazionale Matematico Estivo (C.I.M.E.) held in Cetraro, Italy, June 23–28, 1997*. Ed. by A. Quarteroni. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 150–268 (cit. on p. 28).

[10] B. Cockburn, G. Karniadakis, and C.-W. Shu. *Discontinuous Galerkin Methods: Theory, Computation and Applications*. Springer Publishing Company, Incorporated, 2000 (cit. on pp. 6, 22).

[11] B. Cockburn and C.-W. Shu. "Runge-Kutta Discontinuous Galerkin Methods for Convection-Dominated Problems". In: *Journal of Scientific Computing* 16.3 (2001), pp. 173–261 (cit. on pp. 22, 29).

[12] B. Cockburn and C.-W. Shu. "The Runge-Kutta local projection P^1-discontinuous-Galerkin finite element method for scalar conservation laws". eng. In: *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique* 25.3 (1991), pp. 337–361 (cit. on p. 24).

[13] P. Colella and P. Woodward. "The Piecewise Parabolic Method (PPM) for gas-dynamical simulations". In: *Journal of computational Physics* 54.1 (1984), pp. 174–201 (cit. on pp. 5, 32).

[14] P. Deuflhard and F. Bornemann. *Numerische Mathematik II*. de-Gruyter, 2002 (cit. on p. 30).

[15] C. Doswell III. "A Kinematic Analysis of Frontogenesis Associated with a Nondivergent Vortex". In: *Journal of the Atmospheric Sciences* 41.7 (1984), pp. 1242–1248 (cit. on p. 107).

[16] D. R. Durran. *Numerical Methods for Fluid Dynamics: With Applications to Geophysics*. Texts in Applied Mathematics. Springer New York, 2012 (cit. on pp. 1, 2).

[17] D. Etling. *Theoretische Meteorologie: Eine Einführung*. Vieweg+Teubner Verlag, 2013 (cit. on p. 1).

[18] R. Gerdes, C. Köberle, and J. Willebrand. "The influence of numerical advection schemes on the results of ocean general circulation models". In: *Climate Dynamics* 5.4 (1991), pp. 211–226 (cit. on p. 2).

[19] F. Giraldo. "Lagrange-Galerkin Methods on Spherical Geodesic Grids". In: *Journal of Computational Physics* 136.1 (1997), pp. 197–213 (cit. on p. 6).

[20] F. Giraldo. "Trajectory calculations for spherical geodesic grids in Cartesian space". In: *Monthly Weather Review* (1999) (cit. on p. 30).

[21] S. K. Godunov. "A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics". In: *Matematicheskii Sbornik* 89.3 (1959), pp. 271–306 (cit. on p. 5).

[22] W. Guo, R. Nair, and J.-M. Qiu. "A Conservative semi-Lagrangian Discontinuous Galerkin Scheme on the Cubed Sphere". In: *Monthly Weather Review* 142.1 (2014), pp. 457–475 (cit. on p. 6).

[23] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I (2Nd Revised. Ed.): Nonstiff Problems*. New York, NY, USA: Springer-Verlag New York, Inc, 1993 (cit. on p. 30).

[24] P. Hartman. *Ordinary Differential Equations*. Baltimore, Md., 1973 (cit. on p. 30).

[25] C. Hirsch. *Numerical computation of internal and external flows*. 2. ed. Amsterdam and Heidelberg [u.a.]: Elsevier Butterworth-Heinemann, 2007 (cit. on pp. 11, 37).

[26] M. Hofmann and M. A. Morales Maqueda. "Performance of a second-order moments advection scheme in an Ocean General Circulation Model". In: *Journal of Geophysical Research: Oceans* 111.C5 (2006) (cit. on pp. 6, 130).

[27] R. Klein. "Asymptotics, structure, and integration of sound-proof atmospheric flow equations". In: *Theoretical and Computational Fluid Dynamics* 23.3 (May 2009), pp. 161–195 (cit. on p. 2).

[28] P. Lancaster. "On Eigenvalues of Matrices Dependent on a Parameter." In: *Numerische Mathematik* 6 (1964), pp. 377–387 (cit. on p. 194).

[29] R. Laprise and A. Plante. "A Class of semi-Lagrangian Integrated-Mass (SLIM) Numerical Transport Algorithms". In: *Mon. Wea. Rev.* 123.2 (1995), pp. 553–565 (cit. on p. 31).

[30] P. Lauritzen, P. Ullrich, and R. Nair. "Atmospheric Transport Schemes: Desirable Properties and a semi-Lagrangian View on Finite-Volume Discretizations". In: *Numerical Techniques for Global Atmospheric Models*. Ed. by P. Lauritzen et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 185–250 (cit. on pp. 21, 67).

[31] B. van Leer. "Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection". In: *Journal of Computational Physics* 23.3 (Mar. 1977), pp. 276–299 (cit. on pp. 6, 13, 28, 48, 160, 161).

[32] B. van Leer. "Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method". In: *Journal of Computational Physics* 32.1 (1979), pp. 101–136 (cit. on pp. 5, 17).

[33] B. van Leer. "Upwind and High-Resolution Methods for Compressible Flow: From Donor Cell to Residual-Distribution Schemes". In: *Communications in Computational Physics* 1.2 (Apr. 2006), pp. 192–206 (cit. on p. 17).

[34] B. P. Leonard. "The Ultimate Conservative Difference Scheme Applied to Unsteady One-dimensional Advection". In: *Comput. Methods Appl. Mech. Eng.* 88.1 (June 1991), pp. 17–74 (cit. on p. 3).

[35] P. Lesaint and P. A. Raviart. "On a Finite Element Method for Solving the Neutron Transport Equation". eng. In: *Publications mathématiques et informatique de Rennes* S4 (1974), pp. 1–40 (cit. on pp. 5, 28).

[36] R. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002 (cit. on pp. 5, 33, 101).

[37] H. Lewy, K. Friedrichs, and R. Courant. "Über die partiellen Differenzengleichungen der mathematischen Physik". In: *Mathematische Annalen* 100 (1928), pp. 32–74 (cit. on p. 4).

[38] G. L. Manney et al. "Lagrangian Transport Calculations Using UARS Data. Part II: Ozone". In: *Journal of the Atmospheric Sciences* 52.17 (1995), pp. 3069–3081 (cit. on p. 2).

[39] J. L. McGregor. "Economical determination of departure points for semi-Lagrangian models". In: *Monthly Weather Review* 121:1 (1993) (cit. on p. 30).

[40] L. Michalk. "Numerical methods for the linear advection equation: plateaus vs. extrema". Diplomarbeit. Freie Universität Berlin, 2011 (cit. on p. 21).

[41] S. Nishizawa et al. "Influence of grid aspect ratio on planetary boundary layer turbulence in large-eddy simulations". In: *Geoscientific Model Development* 8.10 (2015), pp. 3393–3419 (cit. on pp. 6, 110).

[42] M. Prather. *Implementation of the SOM method*. 2007. URL: http://ess.uci.edu/researchgrp/prather/files/SOM_2007.zip (cit. on p. 21).

[43] M. Prather. "Numerical Advection by Conservation of Second-Order Moments". In: *Journal of Geophysical Research* 91.D6 (May 1986), pp. 6671–6681 (cit. on pp. 6, 17, 21, 22, 48).

[44] J. Prusa, P. Smolarkiewicz, and A. Wyszogrodzki. "EULAG, a computational model for multiscale flows". In: *Computers & Fluids* 37.9 (2008), pp. 1193–1207 (cit. on p. 2).

[45] D. K. Purnell. "Solution of the Advective Equation by Upstream Interpolation with a Cubic Spline". In: *Mon. Wea. Rev.* 104.1 (1976), pp. 42–48 (cit. on p. 30).

[46] A. Quarteroni and A. Valli. *Numerical approximation of partial differential equations*. Vol. 23. Springer series in computational mathematics. Berlin and Heidelberg: Springer, 2008 (cit. on p. 30).

[47] W. H. Reed and T. R. Hill. "Triangular mesh methods for the neutron transport equation". In: Oct. 1973 (cit. on pp. 5, 22).

[48] M. Restelli, L. Bonaventura, and R. Sacco. "A semi-Lagrangian Discontinuous Galerkin Method for Scalar Advection by Incompressible Flows". In: *J. Comput. Phys.* 216.1 (July 2006), pp. 195–215 (cit. on p. 7).

[49] L. F. Richardson. *Weather prediction by numerical process*. University Press, 1922 (cit. on p. 4).

[50] W. Rider and D. Kothe. "Reconstructing Volume Tracking". In: *Journal of Computational Physics* 141.2 (1998), pp. 112–152 (cit. on p. 119).

[51] A. Robert. "A stable numerical integration scheme for the primitive meteorological equations". In: *Atmosphere-Ocean* 19.1 (1981), pp. 35–46 (cit. on p. 30).

[52] J. S. Sawyer. "A semi-Lagrangian method of solving the vorticity advection equation". In: *Tellus* 15.4 (1963), pp. 336–342 (cit. on p. 30).

[53] C.-W. Shu. "TVB uniformly high-order schemes for conservation laws". In: *Mathematics of Computation* 49.179 (1987), pp. 105–121 (cit. on p. 28).

[54] C.-W. Shu and S. Osher. "Efficient Implementation of Essentially Non-oscillatory Shock-capturing Schemes". In: *J. Comput. Phys.* 77.2 (Aug. 1988), pp. 439–471 (cit. on p. 24).

[55] P. Smolarkiewicz. "A fully multidimensional positive definite advection transport algorithm with small implicit diffusion". In: *Journal of Computational Physics* 54.2 (1984), pp. 325–362 (cit. on p. 34).

[56] P. Smolarkiewicz. "A Simple Positive Definite Advection Scheme with Small Implicit Diffusion". In: *Monthly Weather Review* 111.3 (1983), pp. 479–486 (cit. on pp. 5, 33, 35).

[57] P. Smolarkiewicz and T. Clark. "The multidimensional positive definite advection transport algorithm: Further development and applications". In: *Journal of Computational Physics* 67.2 (1986), pp. 396–438 (cit. on p. 41).

[58] P. Smolarkiewicz and W. Grabowski. "The multidimensional positive definite advection transport algorithm: Nonoscillatory option". In: *Journal of Computational Physics* 86.2 (1990), pp. 355–375 (cit. on p. 34).

[59] P. Smolarkiewicz and L. Margolin. "MPDATA: A Finite-Difference Solver for Geophysical Flows". In: *Journal of Computational Physics* 140.2 (1998), pp. 459–480 (cit. on p. 35).

[60] A. Staniforth and J. Côté. "Semi-Lagrangian Integration Schemes for Atmospheric Models—A Review: Monthly Weather Review". In: *Mon. Wea. Rev* 119.9 (1991), pp. 2206–2223 (cit. on p. 30).

[61] J. Stoer et al. *Introduction to numerical analysis*. Texts in applied mathematics. New York: Springer, 2002 (cit. on p. 30).

[62] G. Strang. "On the Construction and Comparison of Difference Schemes". In: *SIAM Journal on Numerical Analysis* 5.3 (1968), pp. 506–517 (cit. on pp. 6, 101).

[63] G. K. Vallis. *Atmospheric and Oceanic Fluid Dynamics: Fundamentals and Large-scale Circulation*. Cambridge University Press, 2006 (cit. on p. 1).

[64] D. Werner. *Funktionalanalysis*. Springer, Berlin, 2004, 540 p. (Cit. on p. 159).

[65] Wikimedia Commons. *File:Green Chicago River on Saint Patricks Day 2009.jpg — Wikimedia Commons, the free media repository*. 2016. URL: https://commons.wikimedia.org/w/index.php?title=File:Green_Chicago_River_on_Saint_Patricks_Day_2009.jpg&oldid=197238274 (cit. on p. 2).

[66] S. Zalesak. "Fully multidimensional flux-corrected transport algorithms for fluids". In: *Journal of Computational Physics* 31.3 (1979), pp. 335–362 (cit. on pp. 5, 34).

[67] K. Zhang et al. "Consistency problem with tracer advection in the Atmospheric Model GAMIL". In: *Advances in Atmospheric Sciences* 25.2 (Mar. 2008), pp. 306–318 (cit. on p. 67).

ERKLÄRUNG

Hiermit erkläre ich, dass ich alle Hilfsmittel und Hilfen angeben habe und versichere, auf dieser Grundlage die Arbeit selbständig verfasst zu haben. Die Arbeit wurde nicht schon einmal in einem früheren Promotionsverfahren eingereicht.

Berlin, den 1. Februar 2018

Linda Michalk