JET SCHEMES FOR ADVECTION PROBLEMS

BENJAMIN SEIBOLD, RODOLFO R. ROSALES, AND JEAN-CHRISTOPHE NAVE

Abstract. We present a systematic methodology to develop high order accurate numerical approaches for linear advection problems. These methods are based on evolving parts of the jet of the solution in time, and are thus called jet schemes. Through the tracking of characteristics and the use of suitable Hermite interpolations, high order is achieved in an optimally local fashion, i.e. the update for the data at any grid point uses information from a single grid cell only. We show that jet schemes can be interpreted as advect–and–project processes in function spaces, where the projection step minimizes a stability functional. Furthermore, this function space framework makes it possible to systematically inherit update rules for the higher derivatives from the ODE solver for the characteristics. Jet schemes of orders up to five are applied in numerical benchmark tests, and systematically compared with classical WENO finite difference schemes. It is observed that jet schemes tend to possess a higher accuracy than WENO schemes of the same order.

1. Introduction

In this paper we consider a class of approaches for linear advection problems that evolve parts of the jet of the solution in time. Therefore, we call them *jet schemes*.[1] As we will show, the idea of tracking derivatives in addition to function values yields a systematic approach to devise high order accurate numerical schemes that are very localized in space. The results presented are a generalization of gradient-augmented schemes (introduced in [21]) to arbitrary order.

In this paper we specifically consider the linear advection equation

$$\phi_t + \vec{v} \cdot \nabla \phi = 0 \tag{1}$$

for ϕ, with initial conditions $\phi(\vec{x}, 0) = \Phi(\vec{x})$. Furthermore, we assume that the (given) velocity field $\vec{v}(\vec{x}, t)$ is smooth. Equation (1) alone is rarely the central interest of a computational project. However, it frequently occurs as a part of a larger problem. One such example is the movement of a front under a given velocity field (in which case ϕ would be a level set function [22]). Another example is that of advection-reaction, or advection-diffusion, problems solved by fractional steps. Here, we focus solely on the advection problem itself, without devoting much attention to the background in which it may arise. However, we point out that generally one cannot, at any time, simply track back the solution to the initial data. Instead, the problem background generally enforces the necessity to advance the approximate solution in small time increments Δt.

The accurate (i.e. high-order) approximation of (1) on a fixed grid is a non-trivial task. Commonly used approaches can be put in two classes. One class comprises finite difference and finite volume schemes, which store function values or cell averages. These methods achieve high order accuracy by considering neighborhood information, with wide stencils in each coordinate direction. Examples are ENO [25] or WENO [18] schemes with strong stability preserving (SSP) time stepping [11, 24, 25] or flux limiter approaches [28, 29]. Due to their wide stencils, achieving high

2000 *Mathematics Subject Classification.* 65M25; 65M12; 35L04.

Key words and phrases. jet schemes, gradient-augmented, advection, cubic, quintic, high-order, superconsistency.

[1] The term "jet scheme" exists in the fields of algebra and algebraic geometry, introduced by Nash [20] and popularized by Kontsevich [16], as a concept to understand singularities. Since we are here dealing with a computational scheme for a partial differential equation, we expect no possibility for confusion.

order accurate approximations near boundaries can be challenging with these methods. In addition, the generalization of ENO/WENO methods to adaptive grids (quadtrees, octrees) [4, 19] is non-trivial. The other class of approaches is comprised by semi-discrete methods, such as discontinuous Galerkin (DG) [7, 14, 23]. These achieve high order accuracy by approximating the flux through cell boundaries based on high order polynomials in each grid cell. Generally, DG methods are based on a weak formulation of (1), and the flow between neighboring cells is determined by a numerical flux function. All integrals over cells and cell boundaries are approximated by Gaussian quadrature rules, and the time stepping is done by SSP Runge-Kutta schemes [11, 24, 25]. While the DG formalism can in principle yield any order of accuracy, its implementation requires some care in the design of the data structures (choice of polynomial basis, orientation of normal vectors, etc.). Furthermore, SSP Runge-Kutta schemes of an order higher than 4 are quite difficult to design [9, 10].

The approaches (jet schemes) considered here fall into the class of semi-Lagrangian approaches: they evolve the numerical solution on a fixed Eulerian grid, with an update rule that is based on the method of characteristics. Jet schemes share some properties with the methods from both classes described above. Like DG methods, they are based on a high order polynomial approximation in each grid cell, and store derivative information in addition to point values. However, instead of using a weak formulation, numerical flux functions, Gaussian quadrature rules, and SSP schemes, here characteristic curves are tracked — which can be done with simple, non-SSP, Runge-Kutta methods. Furthermore, unlike semi-discrete methods, jet schemes construct the solution at the next time step in a completely local fashion (point by point). Each of the intermediate stages of a Runge-Kutta scheme does not require the reconstruction of an approximate intermediate representation for the solution in space. This property is shared with Godunov-type finite volume methods. Fundamental differences with finite volume methods are that jet schemes are not conservative by design, and that higher derivative information is stored, rather than reconstructed from cell averages. Finally, like many other semi-Lagrangian approaches, jet schemes treat boundary conditions naturally (the distinction between an ingoing and an outgoing characteristic is built into the method), and they do not possess a Courant-Friedrichs-Lewy (CFL) condition that restricts stability. This latter property may be of relevance if the advection equation (1) is one step in a more complex problem that exhibits a separation of time scales.

For advection problems (1), jet schemes are relatively simple and natural approaches that yield high order accuracy with optimal locality: to update the data at a given grid point, information from only a single grid cell is used [21]. In addition, the high order polynomial approximation admits the representation of certain structures of subgrid size — see [21] for more details.

Jet schemes are based on the idea of advect–and–project: one time step in the solution of Equation (1) takes the form

$$\phi^{n+1} = P \circ A_{t+\Delta t,\, t}\, \phi^n, \tag{2}$$

where: (i) ϕ^m denotes the solution at time t_m — with $t_{n+1} = t_n + \Delta t$, (ii) $A_{t+\Delta t,\, t}$ is an approximate advection operator, as obtained by evolving the solution along characteristics using an appropriate ODE solver, and (iii) P is a projection operator, based on knowing a specified portion of the jet of the solution at each grid point in a fixed cartesian grid. At the end of the time step, the solution is represented by an appropriate cell based, polynomial Hermite interpolant — produced by the projection P. We use polynomial Hermite interpolants because they have a very useful stabilizing property: in each cell, the interpolant is a polynomial minimizer for the L^2 norm of a certain high derivative of the interpolated function. This controls the growth of the derivatives of the solution (e.g.: oscillations), and thus ensures stability.

Equation (2) is not quite a numerical scheme, since advecting the complete function ϕ^n would require a continuum of operations. However, it can be easily made into one. In order to be able to apply P, all we need to know is the values of some partial derivatives of $A_{t+\Delta t,\, t}\, \phi^n$ (the portion of the jet that P uses) at the grid points. These can be obtained from the ODE solver as follows: consider the formula (provided by using the ODE solver along characteristics) that gives the value

of $A_{t+\Delta t,\,t}\,\phi^n$ at any point (in particular, the grid points). Then take the appropriate partial derivatives of this formula; this gives an update rule that can be used to obtain the required data at the grid points. This process provides an implementable scheme that is fully equivalent to the continuum (functional) equation (2). We call such a scheme *superconsistent*, since it maintains functional consistency between the function and its partial derivatives.

Finally, we point out that no special restrictions on the ODE solver used are needed. This follows from the minimizing property of the Hermite interpolants mentioned earlier, and superconsistency — which guarantees that the property is not lost, as the time update occurs in the functional sense of Equation (2). Thus, for example, regular Runge-Kutta schemes can be used with superconsistent jet schemes — unlike WENO schemes, which require SSP ODE solvers to ensure total variation diminishing (TVD) stability [12].

This paper is organized as follows. The polynomial representation of the approximate solution is presented in § 2. There we show how, given suitable parts of the jet of a smooth function, a cell-based Hermite interpolant can be used to obtain a high order accurate approximation. This interpolant gives rise to a projection operator in function spaces, defined by evaluating the jet of a function at grid points, and then constructing the piecewise Hermite interpolant. In § 3, the jet schemes' advect–and–project approach sketched above is described in detail. In particular, we show that superconsistent jet schemes are equivalent to advancing the solution in time using the functional equation (2). This interpretation is then used to systematically inherit update rules for the solution's derivatives, from the numerical scheme used for the characteristics. Specific two-dimensional schemes of orders 1, 3, and 5 are constructed. These are then investigated numerically in § 4 for a benchmark test, and compared with classical WENO schemes of the same orders. Boundary conditions and stability are discussed in § 3.4 and § 3.5, respectively.

2. Interpolations and Projections

In this section we discuss the class of projections P that are used and required by jet schemes — see Equation (2). We begin, in § 2.1, by presenting the cell-based generalized Hermite interpolation problem of arbitrary order in any number of dimensions. In § 2.2 we construct, on an arbitrary cartesian grid, global interpolants — using the cell-based generalized Hermite interpolants of § 2.1. Then we introduce a stability functional, which is the key to the stability of the superconsistent jet schemes. Next, in § 2.3, two different types of portions of the jet of a function (corresponding to different kinds of projections) are defined. These are the total and the partial k-jets. In § 2.4 the global interpolants defined in § 2.2 are used to construct (in appropriate function spaces) the projections which are the main purpose of this section. Finally, in § 2.5 we discuss possible ways to decrease the number of derivatives needed by the Hermite interpolants, using cell based finite differences; the notion of an optimally local projection is introduced there.

2.1. Cell-Based Generalized Hermite Interpolation.

We start with a review of the generalized Hermite interpolation problem in one space dimension. Consider the unit interval $[0,1]$. For each boundary point $q \in \{0,1\}$ let a vector of data $(\phi_0^q, \phi_1^q, \ldots, \phi_k^q)$ be given, corresponding to all the derivatives up to order k of some sufficiently smooth function $\phi = \phi(x)$ — the zeroth order derivative is the function itself. Namely

$$\phi_\alpha^q = \left(\tfrac{d}{dx}\right)^\alpha \phi(q) \quad \forall\, q \in \{0,1\},\ \alpha \in \{0,1,\ldots,k\}.$$

In the class of polynomials of degree less than or equal to $n = 2k+1$, this equation can be used to define an interpolation problem with a unique solution. This solution, the n^{th} order Hermite interpolant, can be written as a linear superposition

$$\mathcal{H}_n(x) = \sum_{q\in\{0,1\}} \sum_{\alpha\in\{0,\ldots,k\}} \phi_\alpha^q\ w_{n,\,\alpha}^q(x)$$

of basis functions $w^q_{n,\,\alpha}$, each of which solves the interpolation problem

$$\left(\tfrac{d}{dx}\right)^{\alpha'} w^q_{n,\,\alpha}(q') = \delta_{\alpha,\,\alpha'}\,\delta_{q,\,q'} \quad \forall\, q' \in \{0,1\},\ \alpha' \in \{0,\,\ldots,\,k\},$$

where δ denotes Kronecker's delta. Hence, each of the $2(k+1) = n+1$ basis polynomials equals 1 on exactly one boundary point and for exactly one derivative, and equals 0 for any other boundary point or derivative up to order k. Notice that

$$w^0_{n,\,\alpha}(x) = (-1)^\alpha\, w^1_{n,\,\alpha}(1-x).$$

Example 1. The three lowest order cases of generalized Hermite interpolants are given by the following basis functions:

- linear ($k=0$, $n=1$):
$$w^1_{1,\,0}(x) = x,$$
- cubic ($k=1$, $n=3$):
$$w^1_{3,\,0}(x) = 3x^2 - 2x^3,$$
$$w^1_{3,\,1}(x) = -x^2 + x^3,$$
- quintic ($k=2$, $n=5$):
$$w^1_{5,\,0}(x) = 10x^3 - 15x^4 + 6x^5,$$
$$w^1_{5,\,1}(x) = -4x^3 + 7x^4 - 3x^5,$$
$$w^1_{5,\,2}(x) = \tfrac{1}{2}x^3 - x^4 + \tfrac{1}{2}x^5.$$

One dimensional Hermite interpolation can be generalized naturally to higher space dimensions by using a tensor product approach, as described next.

In $\mathbb{R}^p$, consider a *p-rectangle* (or simply "cell") $[a_1,\,b_1]\times\cdots\times[a_p,\,b_p]$. Let $\Delta x_i = b_i - a_i$, $1 \le i \le p$, denote the edge lengths of the p-rectangle, and call $h = \max_{i=1}^p \Delta x_i$ the *resolution*. In addition, we use the classical multi-index notation. For vectors $\vec{x} \in \mathbb{R}^p$ and $\vec{a} \in \mathbb{N}_0^p$, define: (i) $|\vec{a}| = \sum_{i=1}^p a_i$, (ii) $\vec{x}^{\,\vec{a}} = \prod_{i=1}^p x_i^{a_i}$, and (iii) $\partial^{\,\vec{a}} = \partial_1^{a_1}\,\ldots\,\partial_p^{a_p}$, where $\partial_i = \frac{\partial}{\partial x_i}$.

Definition 1. A *p-n polynomial* is a p-variate polynomial of degree less than or equal to n in each of the variables. Using multi-index notation, a p-n polynomial can be written as

$$\mathcal{H}_n(\vec{x}) = \sum_{\vec{\alpha}\in\{0,\,\ldots,\,n\}^p} c_{\vec{\alpha}}\;\vec{x}^{\,\vec{\alpha}},$$

with $(n+1)^p$ parameters $c_{\vec{\alpha}}$. Note that here we will consider only the case where n is odd.

Example 2. Examples of p-n polynomials are:

	$p=1$	$p=2$		$p=3$	
• $n=1$:	linear,	bi-linear,	and	tri-linear	functions.
• $n=3$:	cubic,	bi-cubic,	and	tri-cubic	functions.
• $n=5$:	quintic,	bi-quintic,	and	tri-quintic	functions.

Now let the p-rectangle's 2^p vertices be indexed by a vector $\vec{q} \in \{0,1\}^p$, such that the vertex of index $\vec{q}$ is at position $\vec{x}_{\vec{q}} = (a_1 + \Delta x_1\, q_1,\,\ldots,\,a_p + \Delta x_p\, q_p)$.

Definition 2. For n odd, and a sufficiently smooth function $\phi = \phi(\vec{x})$, the *n-data* on the p-rectangle (defined on the vertices) is the set of $(n+1)^p$ scalars given by

$$\phi^{\vec{q}}_{\vec{\alpha}} = \partial^{\,\vec{\alpha}}\,\phi(\vec{x}_{\vec{q}}), \tag{3}$$

where $\vec{q} \in \{0,1\}^p$ and $\vec{\alpha} \in \{0,\ldots,k\}^p$, with $k = \frac{n-1}{2}$.

Lemma 1. *Two p-n polynomials with the same n-data on some p-rectangle, must be equal.*

Proof. Let ϕ be the difference between the two polynomials, which has zero data. We prove that $\phi \equiv 0$ by induction over p. For $p = 1$, we have a standard 1-D Hermite interpolation problem, whose solution is known to be unique. Assume now that the result applies for $p-1$. In the p-rectangle, consider the functions ϕ, $\partial_p \phi$, ..., $\partial_p^k \phi$ — both at the "bottom" hyperface ($x_p = a_p$, i.e. $q_p = 0$) and the "top" hyperface ($x_p = b_p$, i.e. $q_p = 1$). For each of these functions, zero data is given at all of the corner vertices of the two hyperfaces. Therefore, by the induction assumption, all of these functions vanish everywhere on the top and bottom hyperfaces. Consider now the "vertical" lines joining a point $(x_1, \ldots, x_{p-1}) \in [a_1, b_1] \times \ldots \times [a_{p-1}, b_{p-1}]$ in the bottom hyperface with its corresponding one on the top hyperface. For each of these lines we can use the $p = 1$ uniqueness result to conclude that $\phi = 0$ identically on the line. It follows that $\phi = 0$ everywhere. □

Theorem 2. *For any arbitrary n-data (n odd) on some p-rectangle, there exists exactly one p-n polynomial which interpolates the data.*

Proof. Lemma 1 shows that there exists at most one such polynomial. The interpolating p-n polynomial is explicitly given by

$$\mathcal{H}_n(\vec{x}) = \sum_{\vec{q} \in \{0,1\}^p} \; \sum_{\vec{\alpha} \in \{0,\ldots,k\}^p} \phi_{\vec{\alpha}}^{\vec{q}} \; W_{n,\,\vec{\alpha}}^{\vec{q}}(\vec{x}), \tag{4}$$

where the $W_{n,\,\vec{\alpha}}^{\vec{q}}(\vec{x})$ are p-n polynomial basis functions that satisfy

$$\partial^{\,\vec{\alpha}\,\prime} W_{n,\,\vec{\alpha}}^{\vec{q}}\,(\vec{x}_{\vec{q}\,\prime}) = \delta_{\vec{\alpha},\,\vec{\alpha}\,\prime}\;\delta_{\vec{q},\,\vec{q}\,\prime} \quad \forall\, \vec{q}\,' \in \{0,1\}^p,\; \vec{\alpha}\,' \in \{0,\ldots,k\}^p.$$

They are given by the tensor products

$$W_{n,\,\vec{\alpha}}^{\vec{q}}(\vec{x}) = \prod_{i=1}^{p} (\Delta x_i)^{\alpha_i}\; w_{n,\,\alpha_i}^{q_i}(\xi_i),$$

where $\xi_i = \frac{x_i - a_i}{\Delta x_i}$ is the relative coordinate in the p-rectangle, and the $w_{n,\,\alpha}^{q}$ are the univariate basis functions defined earlier. □

Next we show that the p-n polynomial interpolant given by Equation (4) is a $(n+1)^{st}$ order accurate approximation to any sufficiently smooth function ϕ it interpolates on a p-rectangle. For convenience, we consider a p-cube with $\Delta x_1 = \cdots = \Delta x_p = h$. In this case, the interpolant in (4) becomes

$$\mathcal{H}_n(\vec{x}) = \sum_{\vec{q} \in \{0,1\}^p} \; \sum_{\vec{\alpha} \in \{0,\ldots,k\}^p} \phi_{\vec{\alpha}}^{\vec{q}}\; h^{|\vec{\alpha}|} \prod_{i=1}^{p} w_{n,\,\alpha_i}^{q_i}(\xi_i). \tag{5}$$

Lemma 3. *Let the data determining the p-n polynomial Hermite interpolant be known only up to some error. Then Equation* (5) *yields the interpolation error*

$$\delta\mathcal{H}_n(\vec{x}) = \sum_{\vec{q} \in \{0,1\}^p} \; \sum_{\vec{\alpha} \in \{0,\ldots,k\}^p} \left(\prod_{i=1}^{p} w_{n,\,\alpha_i}^{q_i}(\xi_i) \right) h^{|\vec{\alpha}|}\; \delta\phi_{\vec{\alpha}}^{\vec{q}},$$

where the notation δu indicates the error in some quantity u. In particular, if the data $\phi_{\vec{\alpha}}^{\vec{q}}$ are known with $O\left(h^{n+1-|\vec{\alpha}|}\right)$ accuracy, then $\delta\left(\partial^{\vec{\alpha}}\mathcal{H}_n\right) = O\left(h^{n+1-|\vec{\alpha}|}\right)$.

Theorem 4. *Consider a sufficiently smooth function ϕ, and let $\mathcal{H}_n(\vec{x})$ be the p-n polynomial that interpolates the data given by ϕ on the vertices of a p-cube of size h. Then, everywhere inside the p-cube, one has*

$$\partial^{\vec{\alpha}}\mathcal{H}_n - \partial^{\vec{\alpha}}\phi = O\left(h^{n+1-|\vec{\alpha}|}\right), \tag{6}$$

where the constant in the error term is controlled by the $(n+1)^{st}$ derivatives of ϕ.

Proof. Let $\mathcal{G}$ be the degree n polynomial Taylor approximation to ϕ, centered at some point inside the p-cube. Then, by construction: (i) $\partial^{\vec{\alpha}}\mathcal{G} - \partial^{\vec{\alpha}}\phi = O\left(h^{n+1-|\vec{\alpha}|}\right)$. In particular, the data for $\mathcal{G}$ on the p-cube is related to the data for $\mathcal{H}_n$ (same as the data for ϕ) in the manner specified in Lemma 3. Thus: (ii) $\partial^{\vec{\alpha}}\mathcal{H}_G - \partial^{\vec{\alpha}}\mathcal{H}_n = O\left(h^{n+1-|\vec{\alpha}|}\right)$, where $\mathcal{H}_G$ is the p-n polynomial that interpolates the data given by $\mathcal{G}$ on the vertices of the p-cube. However, from Lemma 1: (iii) $\mathcal{H}_G = \mathcal{G}$, since $\mathcal{G}$ is a p-n polynomial. From (i), (ii), and (iii) Equation (6) follows. □

In conclusion, we have shown that: the p-n polynomial Hermite interpolant approximates smooth functions with $(n+1)^{st}$ order accuracy, and each level of differentiation lowers the order of accuracy by one level. Further: in order to achieve the full order accuracy, the data $\phi^{\vec{q}}_{\vec{\alpha}}$ must be known with accuracy $O\left(h^{n+1-|\vec{\alpha}|}\right)$.

2.2. **Global Interpolant.** Consider a rectangular computational domain $\Omega \subset \mathbb{R}^p$ in p spacial dimensions, equipped with a regular rectangular grid. Assume that at each grid point $\vec{x}_{\vec{m}}$, labeled by $\vec{m} \in \mathbb{Z}^p$, a vector of data $\phi^{\vec{m}}_{\vec{\alpha}}$ is given, where $\vec{\alpha} \in \{0, \ldots, k\}^p$ — for some $k \in \mathbb{N}_0$. Given this grid data, we define a global interpolant $\mathcal{H} : \Omega \to \mathbb{R}$, which is a piece-wise p-n polynomial (with $n = 2k+1$). On each grid cell $\mathcal{H}$ is given by the p-n polynomial obtained using Equation (4), with $\vec{q}$ related to the grid index $\vec{m}$ by: $\vec{q} = \vec{m} - \vec{m}_0$ — where $\vec{m}_0$ is the cell vertex with the lowest values for each component of $\vec{m}$. Note that $\mathcal{H}$ is C^∞ inside each grid cell, and C^k across cell boundaries. However, in general, $\mathcal{H}$ is not C^{k+1}.

Remark 1. The smoothness of $\mathcal{H}$ is biased in the coordinate directions. All the derivatives that appear in the data vectors, i.e. $\partial^{\vec{\alpha}}\mathcal{H}$ for $\vec{\alpha} \in \{0, \ldots, k\}^p$, are defined everywhere in Ω. Furthermore, at the grid points, $\partial^{\vec{\alpha}}\mathcal{H}(\vec{x}_{\vec{m}}) = \phi^{\vec{m}}_{\vec{\alpha}}$. In particular, all the partial derivatives up to order k are defined, and continuous. However, not all the partial derivatives of orders larger than k are defined. In general $\partial^{\vec{\alpha}}\mathcal{H}$, with $\vec{\alpha} \in \{0, \ldots, k+1\}^p$, is piece-wise smooth — with simple jump discontinuities across the grid hyperplanes which are perpendicular to any coordinate direction x_ℓ such that $\alpha_\ell = k+1$. Derivatives $\partial^{\vec{\alpha}}\mathcal{H}$, where $\alpha_\ell > k+1$ for some $1 \le \ell \le p$, generally exist only in the sense of distributions.

Definition 3. Any (sufficiently smooth) function $\phi : \Omega \to \mathbb{R}$ defines a global interpolant $\mathcal{H}_\phi$ as follows: at each grid point $\vec{x}_{\vec{m}}$, evaluate the derivatives of ϕ, as by Definition 2, to produce a data vector. Namely: $\phi^{\vec{m}}_{\vec{\alpha}} = \partial^{\vec{\alpha}}\phi(\vec{x}_{\vec{m}})$ $\forall \vec{\alpha} \in \{0, \ldots, k\}^p$. Then, use these values as data to define $\mathcal{H}_\phi$ everywhere.

Definition 4. For any sufficiently smooth function ϕ, define the *stability functional* by

$$\mathcal{F}[\phi] = \int_\Omega \left(\partial^{\vec{\beta}}\phi(\vec{x})\right)^2 \mathrm{d}\vec{x}, \tag{7}$$

where $\vec{\beta} = \vec{\beta}(k, p)$ is the p-vector $\vec{\beta} = (k+1, \ldots, k+1)$. Of course, $\mathcal{F}$ also depends on $k \in \mathbb{N}_0$ and $\Omega \subset \mathbb{R}^p$, but (to simplify the notation) we do not display these dependencies.

Theorem 5. *Replacing a sufficiently smooth function ϕ by the interpolant $\mathcal{H}_\phi$ does not increase the stability functional: $\mathcal{F}[\mathcal{H}_\phi] \le \mathcal{F}[\phi]$. In fact: $\mathcal{H}_\phi$ minimizes $\mathcal{F}$, subject to the constraints given by the data $\partial^{\vec{\alpha}}\phi(\vec{x}_{\vec{m}})$, and the requirement that the minimizer should be smooth in each grid cell.*

Remark 2. The minimizer is not unique if $p > 1$. To see this, let $f_j(x_j)$, $1 \le j \le p$ be nonzero smooth functions, with f_j and all its derivatives vanishing at the the grid points. Then $\psi = \mathcal{H}_\phi + \sum_{j=1}^p f_j(x_j) \ne \mathcal{H}_\phi$ has the same data as $\mathcal{H}_\phi$ and ϕ, and $\mathcal{F}[\psi] = \mathcal{F}[\mathcal{H}_\phi]$ if $p > 1$.

Proof. Note that $\partial^{\vec{\beta}}\mathcal{H}_\phi$ exists and it is piece-wise smooth (it may have simple discontinuities across cell boundaries — see Remark 1). Hence $\mathcal{F}[\mathcal{H}_\phi]$ is defined. Clearly, it is enough to show that $\mathcal{H}_\phi$ minimizes $\mathcal{F}$ in each grid cell Q. Define $\varphi = \phi - \mathcal{H}_\phi$. Since ϕ and $\mathcal{H}_\phi$ have the same data at the vertices of Q, φ has zero data at the vertices. Thus

$$I_p = \int_Q \left(\partial^{\vec{\beta}}\mathcal{H}_\phi(\vec{x})\right)\left(\partial^{\vec{\beta}}\varphi(\vec{x})\right) \mathrm{d}\vec{x} = 0, \tag{8}$$

as shown in Lemma 7. From this equality it follows that $\mathcal{F}[\phi] = \mathcal{F}[\mathcal{H}_\phi] + \mathcal{F}[\varphi]$. Now, since $\mathcal{F}[\varphi] \geq 0$, $\mathcal{F}[\mathcal{H}_\phi] \leq \mathcal{F}[\phi]$. For any other ϕ^* satisfying the theorem statement's constraints for the minimizing class, $\mathcal{H}_\phi = \mathcal{H}_{\phi^*}$. Hence $\mathcal{F}[\mathcal{H}_\phi] = \mathcal{F}[\mathcal{H}_{\phi^*}] \leq \mathcal{F}[\phi^*]$. □

Corollary 6. *$\phi \to \mathcal{H}_\phi$ is an orthogonal projection with respect to the positive semi-definite quadratic form associated with $\mathcal{F}$ — namely: the one used by Equation* (8)*.*

The reason for the name "stability functional", and the relevance of Theorem 5, will become clear later in § 3.5, where the inequality $\mathcal{F}[\mathcal{H}_\phi] \leq \mathcal{F}[\phi]$ is shown to play a crucial role in ensuring the stability of superconsistent jet schemes. Theorem 5 states that the Hermite interpolant is within the class of the least oscillatory functions that matches the data — where "oscillatory" is measured by the functional $\mathcal{F}$.

Lemma 7. *The equality in* (8) *applies.*

Proof. The proof is by induction over p. Without loss of generality, assume $Q = [0,\, 1]^p$ is the unit p-cube. For $p = 1$ the result holds, since $k+1$ integrations by parts can be used to obtain:

$$I_1 = (-1)^{k+1} \int_0^1 \left(\partial_1^{2k+2}\, \mathcal{H}_\phi(x_1)\right)\, \varphi(x_1)\, \mathrm{d}x_1 = 0,$$

where there are no boundary contributions because the data for φ vanishes, and we have used that $\partial_1^{2k+2}\mathcal{H}_\phi(x) = 0$. Assume now that the result is true for $p-1 > 0$, and do $k+1$ integrations by parts over the variable x_p. This yields

$$I_p = (-1)^{k+1} \int_Q \left(\partial^{\,\vec{\beta}\,\prime}\, \partial_p^{2k+2}\, \mathcal{H}_\phi(\vec{x})\right) \left(\partial^{\,\vec{\beta}\,\prime} \varphi(\vec{x})\right)\, \mathrm{d}\vec{x} + \mathrm{BTC} = \mathrm{BTC},$$

where BTC stands for "Boundary Terms Contributions", $\vec{\beta}\,' = (k+1, \ldots, k+1)$ is a $(p-1)$-vector, and we have used that $\partial_p^{2k+2}\mathcal{H}_\phi(x) = 0$. Furthermore the BTC are a sum over terms of the form

$$I_{p-1,\,j,\,q} = \int_{Q_q} \left(\partial^{\,\vec{\beta}\,\prime}\, \partial_p^{k+j+1}\, \mathcal{H}_\phi(\vec{x})\right) \left(\partial^{\,\vec{\beta}\,\prime}\, \partial_p^{k-j}\, \varphi(\vec{x})\right)\, \mathrm{d}\vec{x}, \quad 0 \leq j \leq k, \quad 0 \leq q \leq 1,$$

where Q_q stands for the $(p-1)$-cube obtained from Q by setting either $x_p = 0$ ($q = 0$) or $x_p = 1$ ($q = 1$). Now: the data for φ is, precisely, the union of the data for $\{\partial_p^{k-j}\, \varphi\}_{j=0}^{k}$ in both Q_0 and Q_1. Furthermore $\partial_p^{k+j+1}\, \mathcal{H}_\phi$, restricted to either Q_0 or Q_1, is a $(p-1)$-n polynomial. Hence we can use the induction hypothesis to conclude that $I_{p-1,\,j,\,q} = 0$, for all choices of j and q. Thus $I_p = \mathrm{BTC} = 0$, which concludes the inductive proof. □

2.3. **Total and Partial k-Jets.** The *jet of a smooth function* $\phi : \Omega \to \mathbb{R}$ is the collection of all the derivatives $\partial^{\,\vec{\alpha}}\phi$, where $\vec{\alpha} \in \mathbb{N}_0^p$. The interpolants defined in § 2.2 are based on parts of the jet of a function, evaluated at grid points. Next we introduce two different notions characterizing parts of jet.

Definition 5. The *total k-jet* is the collection of all the derivatives $\partial^{\,\vec{\alpha}}\phi$, where $|\vec{\alpha}| \leq k$.

Definition 6. The *partial k-jet* is the collection of all derivatives $\partial^{\,\vec{\alpha}}\phi$, where $\vec{\alpha} \in \{0, \ldots, k\}^p$.

Thus the total k-jet contains all the derivatives up to the order k, while the partial k-jet contains all the derivatives for which the partial derivatives with respect to each variable are at most order k. For any given $k \geq 0$, the total k-jet is contained within the partial k-jet (strictly if $k > 0$), while the partial k-jet is a subset of the total n-jet, with $n = pk$.

2.4. **Projections in Function Spaces.** The aim of this subsection is to use the global interpolant introduced in § 2.2 to define projections of functions — which are needed for the projection step in the general class of advect–and–project methods specified in § 3.2. This requires the introduction of appropriate spaces where the projections operate. Further, because of the advection that occurs between projection steps in the § 3.2 methods, it is necessary that these spaces be invariant under diffeomorphisms. Unfortunately, the coordinate bias (see Remark 1) that the generalized Hermite interpolants exhibit causes difficulties with this last requirement, as explained next.

The functions that we are interested in projecting are smooth advections, over one time step, of some global interpolant $\mathcal{H}$. Let ψ be such a function. From Remark 1, it follows that ψ is C^k and piece-wise smooth, with singularity hypersurfaces given by the advection of the grid hyperplanes. Inside each of the regions that result from advecting a single grid cell, ψ is smooth all the way to the boundary of the region. Furthermore, at a singularity hypersurface:

(i) Partial derivatives of ψ involving differentiation normal to the hypersurface of order k or lower are defined and continuous.
(ii) Partial derivatives of ψ involving differentiation normal to the hypersurface of order $k+1$ are defined, but (generally) have a simple jump discontinuity.
(iii) Partial derivatives of ψ involving differentiation normal to the hypersurface of order higher than $k+1$ are (generally) not defined as functions. However, they are defined on each side, and are continuous up to the hypersurface.

Imagine now a situation where a grid point is on one of the singularity hypersurfaces. Then the interpolant $\mathcal{H}_\psi$, as given by Definition 3, may not exist — because one, or more, of the partial derivatives of ψ needed at the grid point does not exist. This is a serious problem for the advect–and–project strategy advocated in § 3.2. This leads us to the considerations below, and a recasting of Definition 3 (i.e.: Definition 7) that resolves the problem.

We are interested in solutions to Equation (1) that are are sufficiently smooth. Let then ψ be an approximation to such a solution. In this case, from Theorem 4, we should add to (i – iii) above the following

(iv) The jumps in any partial derivative of ψ (across a singularity hypersurface) cannot be larger than $O(h^{n+1-s})$, where $n = 2k+1$ and s is the order of the derivative.

Then, from Lemma 3, we see that these jumps are below the numerical resolution — hence, essentially, not present from the numerical point of view. This motivates the following generalization of Definition 3:

Definition 7. For functions ψ that satisfy the restrictions in items (i – iv) above, define the global interpolant $\mathcal{H}_\psi$ as in Definition 3, with one exception: whenever a partial derivative $\psi_{\vec{\alpha}}^{\vec{m}} = \partial^{\vec{\alpha}}\psi(\vec{x}_{\vec{m}})$ which is needed for the data at a grid point does not exist (because the grid point is on a singularity hypersurface), supply a value using the formula

$$\partial^{\vec{\alpha}}\psi(\vec{x}_{\vec{m}}) = \frac{1}{2}\left(\limsup_{\vec{x}\to\vec{x}_{\vec{m}}} \partial^{\vec{\alpha}}\psi(\vec{x}) + \liminf_{\vec{x}\to\vec{x}_{\vec{m}}} \partial^{\vec{\alpha}}\psi(\vec{x})\right). \tag{9}$$

Remark 3. The value in equation (9) for $\partial^{\vec{\alpha}}\psi(\vec{x}_{\vec{m}})$ ignores the "distribution component" of $\partial^{\vec{\alpha}}\psi(\vec{x}_{\vec{m}})$ — i.e. Dirac's delta functions and derivatives, with support on the singularity hypersurface. These components originate purely because of approximation errors: small mismatches in the derivatives of the Hermite interpolants across the grid hyperplanes. Thus we expect Equation (9) to provide an accurate approximation to the corresponding partial derivative of the smooth function that ψ approximates. Note that, while we are unable to supply a rigorous justification for this argument, the numerical experiments that we have conducted provide a validation — see § 4.

Remark 4. An alternative to Definition 7, that also allows the construction of global interpolants from advected global interpolants, is introduced in Definition 12 below.

Motivated by the discussion above, we now consider two types of projections (and corresponding function spaces). Note that in the theoretical formulation that follows below, the "small jumps in the partial derivatives" property is not built into the definitions of the spaces. This is not untypical for numerical methods. For instance, finite difference discretizations are only meaningful for sufficiently smooth functions. However, the finite differences themselves are defined for any function that lives on the grid, even though the notion of smoothness does not make sense for such grid-functions.

Definition 8. Let S^ν denote the space of of all the L^∞ functions $\psi : \Omega \to \mathbb{R}$ which are ν-times differentiable a.e., with derivatives in L^∞.

Here differentiable is meant in the classical sense: $\psi(\vec{x}+\vec{u}) = \psi(\vec{x}) + (D\,\psi)\,[\vec{u}] + \frac{1}{2}\left(D^2\,\psi\right)[\vec{u},\vec{u}] + \ldots + \frac{1}{\nu!}\,(D^\nu\,\psi)\,[\vec{u},\ldots,\vec{u}] + o\left(\|\vec{u}\|^\nu\right)$ near a point where the function is differentiable, where $(D^\mu\,\psi)$, $1 \le \mu \le \nu$, is a μ-linear form.

Definition 9. Let $S^{k,+}$ be defined by $S^{k,+} = S^\nu \cap C^k$, with $\nu = pk$.

Definition 10. For any function $\psi \in S^\nu$, and any $\vec{\alpha} \in \mathbb{N}_0^p$ with $|\vec{\alpha}| \le \nu$, define $\partial^{\vec{\alpha}}\psi$ at every point $\vec{x}_{\vec{m}}$ in the rectangular grid, via

$$\partial^{\vec{\alpha}}\psi(\vec{x}_{\vec{m}}) = \frac{1}{2}\left(\operatorname*{ess\,lim\,sup}_{\vec{x}\to\vec{x}_{\vec{m}}} \partial^{\vec{\alpha}}\psi(\vec{x}) + \operatorname*{ess\,lim\,inf}_{\vec{x}\to\vec{x}_{\vec{m}}} \partial^{\vec{\alpha}}\psi(\vec{x})\right). \tag{10}$$

where ess lim sup and ess lim inf denote the essential upper and lower limits [2], respectively. Notice that this last formula differs from (9) by the use of essential limits only.

The definition in Equation (10) is not the only available option. One could use the upper limit only, or the lower limit only, or some intermediate value between these two — since numerically the various alternatives lead to differences that are of the order of the truncation errors. In particular, a single sided evaluation is more efficient, so this is what we use in our numerical implementations. In what follows we assume that a choice has been made, e.g.: the one given by Equation (10), or the one given in Remark 9.

Definition 11. In $S^{k,+}$, define the projection $P_k^+ : S^{k,+} \to S^{k,+}$ as the application of the interpolant defined in § 2.2, i.e. $P_k^+\psi = \mathcal{H}_\psi$.

Definition 12. In S^k, define the projection $P_k : S^k \to S^k$ by the following steps.

(1) Given a function $\psi \in S^k$, evaluate the total k-jet at the grid points $\vec{x}_{\vec{m}}$.
(2) Approximate the remaining derivatives in the partial k-jet by a process like the one described in § 2.5. Notice that, if the approximation of the derivatives of order $\ell > k$ is done with $O(h^{n+1-\ell})$ errors, then (by Lemma 3) the full accuracy of the projection is preserved.
(3) Define $P_k\psi$ as the global interpolant (see § 2.2) based on this approximate partial k-jet.

2.5. **Construction of the Partial k-jet from the Total k-jet.** For the advect–and–project methods introduced in § 3.2, the advection of the solution's derivatives is a significant part of the computational cost. Furthermore, the higher the derivative, the higher the cost. For example, when using P_k^+ with $k = 2$ and the approach in § 3.3.2, the cost of obtaining the $|\vec{\alpha}| > k$ derivatives in the partial k-jet is (at the lowest) about three times that of obtaining the total k-jet. Further: the cost ratio grows, roughly, linearly with k. Thus it is tempting to design more efficient approaches that are based on advecting the total k-jet only, and then recovering the higher derivatives in the partial k-jet by a finite difference approximation of the total k-jet data at grid points. We call such approaches *grid-based finite differences*, in contrast to the *ε-finite differences* introduced in § 3.3.2.

Unfortunately, jet schemes based on grid-based finite difference reconstructions of the higher derivatives in the partial k-jet have one major drawback: the nice stabilizing properties that the Hermite interpolants possess (see Theorem 5 and § 3.5) are lost. Hence the stability of these schemes is not ensured. Nevertheless, we explore this idea numerically (see § 4.2 for some

examples). As expected, in the absence of a theoretical underpinning that would allow us to distinguish stable schemes from their unstable counterparts, many approaches that are based on grid-based finite differences turn out to yield unstable schemes. However

- In the case $k = 1$, a stable scheme using a grid-based finite difference reconstruction of the partial k-jet, can be given. It is described in Example 3. The particular reconstruction used has some interesting properties, which motivate Definition 13 below.
- In unstable versions of schemes that are based on grid-based reconstructions of higher derivatives, the instabilities are, in general, observed to be very weak: even for fairly small grid sizes, the growth rate of the instabilities is rather small — an example of this phenomenon is shown in § 4.2. Thus it may be possible to stabilize these schemes (e.g. by some form of jet scheme artificial viscosity) without seriously diminishing their accuracy.

Definition 13. We say that a projection is *optimally local* if, at each grid node, the recovered data (e.g. derivatives with $|\vec{\alpha}| > k$ in the partial k-jet) depends solely on the known data (e.g. the total k-jet) at the vertices of a single grid cell.

Of course, the projections P_k^+ in Definition 11 are by default optimally local (since there is no data to be recovered). However, as pointed out above, the idea of optimally local projections that are based on the total k-jet is that they produce more efficient jet schemes than using P_k^+. In Example 3 below we present an optimally local projection, which corresponds to P_1 in Definition 12.

Remark 5. An important question is: *why is optimal locality desirable?* The reason is that optimally local projections do not involve communication between cells. This has, at least, three advantageous consequences. First, near boundaries, a local formulation simplifies the enforcement of boundary conditions (e.g. see § 3.4). Second, in situations where adaptive grids are used, a local formulation can make the implementation considerably simpler than it would be for non-local alternatives. And third, locality is a desirable property for parallel implementations since communication boundaries between processors are reduced to lines of single cells.

Example 3. In two space dimensions, we can define an optimally local version of the projection P_1, as follows below. In this projection we presume that the total 1-jet is given at each grid point. To simplify the description, we work in the cell $Q = [0,\, h] \times [0,\, h]$, with $\vec{q} \in \{0,\, 1\}^2$ corresponding to the vertex $\vec{x}_{\vec{q}} = \vec{q}\, h$. Thus $\psi^{\vec{q}}$ indicates the value $\psi(\vec{x}_{\vec{q}})$. We also use the convention that when $q_i = \frac{1}{2}$, the quantity is defined or evaluated at $x_i = \frac{h}{2}$, for $i \in \{1,\, 2\}$.

Define P_1 as follows (this is the projection introduced, and implemented, in [21]). To construct the bi-cubic interpolant, at each vertex of Q the derivative ψ_{xy} must be approximated with second order accuracy, using the given data at the vertices. To do so: (i) Use centered differences to write second order approximations to ψ_{xy} at the midpoints of each of the edges of Q. For example: $\psi_{xy}^{(0,\frac{1}{2})} = \frac{1}{h}\left(\psi_x^{(0,1)} - \psi_x^{(0,0)}\right)$ and $\psi_{xy}^{(\frac{1}{2},0)} = \frac{1}{h}\left(\psi_y^{(1,0)} - \psi_y^{(0,0)}\right)$. (ii) Use the four values obtained in (i) to define a bi-linear (relative to the rotated coordinate system $x + y$ and $x - y$) approximation to ψ_{xy}. (iii) Evaluate the bi-linear approximation obtained in (ii) at the vertices of Q. Three dimensional versions of these formulas are straightforward, albeit somewhat cumbersome.

We do not know whether stable analogs of P_1 exist for P_k with $k > 1$. It is plausible that a construction similar to the one given in Example 3 (i.e. based on using lower order Hermite interpolants in rotated coordinate systems) may yield the desired stability properties. This is the subject of current research.

Remark 6. The optimally local projection P_1 with "single cell approximations" uses, at each grid point, values for ψ_{xy} that are different for each of the adjoining cells. This means that (generally) $P_1\psi \notin C^1$ — though $P_1\psi \in C^0$ always. However, for sufficiently smooth functions ψ, the discontinuities in the derivatives of $P_1\,\psi$ across the grid lines cannot involve jumps larger than $O(h^{4-s})$, where s is the order of the derivative. This is a small variation of the situation discussed from the beginning of § 2.4 through Remark 3. The same arguments made there apply here.

Remark 7. The lack of smoothness of the optimally local projection P_1 (i.e. Remark 6) could be eliminated as follows. At each grid node consider all the possible values obtained for ψ_{xy} by cell based approximations (one per cell adjoining the node). Then assign to ψ_{xy} the average of these values. This eliminates the multiple values, and yields an interpolant $P_1\psi$ that belongs to C^1. Formally this creates a dependence of the partial 1-jet on the total 1-jet that is not optimally local. However, all the operations done are solely cell-based (i.e.: the reconstruction of the partial 1-jet) or vertex based (i.e.: the averaging). Hence, such an approach can easily be applied at boundaries or in adaptive meshes.

3. Advection and Update in Time

The characteristic form of Equation (1) is

$$\frac{d\,\vec{x}}{dt} = \vec{v}(\vec{x},\, t), \tag{11}$$

$$\frac{d\,\phi}{dt} = 0. \tag{12}$$

Let $\vec{X}(\vec{x},\, \tau,\, t)$ denote the solution of the ordinary differential equation (11) at time t, when starting with initial conditions $\vec{x}$ at time $t = \tau$. Hence $\vec{X}$ is defined by

$$\frac{\partial}{\partial t}\vec{X}(\vec{x},\, \tau,\, t) = \vec{v}(\vec{X}(\vec{x},\, \tau,\, t),\, t), \quad \text{with} \quad \vec{X}(\vec{x},\, \tau,\, \tau) = \vec{x}.$$

Due to (12), the solution of the partial differential equation (1) satisfies

$$\phi(\vec{x},\, t + \Delta t) = \phi(\vec{X}(\vec{x},\, t + \Delta t,\, t),\, t).$$

That is: the solution at time $t + \Delta t$ and position $\vec{x}$ is found by tracking the corresponding characteristic curve, given by (11), backwards to time t, and evaluating the solution at time t there. Introduce now the solution operator $S_{\tau,\,t}$, which maps the solution at time t to the solution at time τ. Then $S_{\tau,\,t}$ acts on a function $g(\vec{x})$ as follows

$$(S_{\tau,\,t}\, g)(\vec{x}) = g(\vec{X}(\vec{x},\, \tau,\, t)). \tag{13}$$

Hence, the solution of (1) satisfies

$$\phi(\vec{x},\, t + \Delta t) = S_{t+\Delta t,\,t}\, \phi(\vec{x},\, t).$$

In particular, the solution at time $t = n\,\Delta t$ can be obtained by applying successive advection steps to the initial conditions

$$\phi(\vec{x},\, t) = S_{t,\,t-\Delta t} \circ \cdots \circ S_{2\Delta t,\,\Delta t} \circ S_{\Delta t,\,0}\, \Phi(\vec{x}).$$

3.1. Approximate Advection.

Let $\vec{\mathcal{X}}$ be an approximation to the exact solution $\vec{X}$, as arising from a numerical ODE solver — e.g. a high order Runge-Kutta method. By analogy to the true advection operator (13), introduce an approximate advection operator, defined as acting on a function $g(\vec{x})$ as follows

$$(A_{\tau,\,t}\, g)(\vec{x}) = g(\vec{\mathcal{X}}(\vec{x},\, \tau,\, t)). \tag{14}$$

The approximate characteristic curves, given by $\vec{\mathcal{X}}$, motivate the following definition.

Definition 14. The point $\vec{x}_{\text{foot}} = \vec{\mathcal{X}}(\vec{x},\, t + \Delta t,\, t)$ is called the foot of the (approximate) characteristic through $\vec{x}$ at time $t + \Delta t$. Note that $\vec{x}_{\text{foot}} = \vec{x}_{\text{foot}}(\vec{x},\, t,\, \Delta t)$, but we do not display these dependencies when obvious.

Let now $\vec{\mathcal{X}}(\vec{x},\, t + \Delta t,\, t)$ represent a single ODE solver step for (11), from $t + \Delta t$ to t, starting from $\vec{x}$ — i.e. let Δt be the ODE solver time step. Then the successive application of approximate advection operators

$$A_{t,\,t-\Delta t} \circ \cdots \circ A_{2\Delta t,\,\Delta t} \circ A_{\Delta t,\,0}\, \Phi(\vec{x}) \tag{15}$$

yields an approximation to the solution of (1) at time $t = n\Delta t$. In principle this formula provides a way to (approximately) solve (11) by taking (for each point $\vec{x}$ at which the solution is desired at time $t = n\,\Delta t$) n ODE solver steps from time t back to time 0, and then evaluating the initial conditions at the resulting position. However, as described in § 1, we are interested in approaches that allow access to the solution at each time step (represented on a numerical grid by a finite amount of data), and then advance it forward to the next time step. Hence the method provided by expression (15) is not adequate.

3.2. **Advect–and–Project Approach.** In this approach, an appropriate projection is applied at the end of every time step, after the advection. The projection allows the representation of the solution, at each of a discrete set of times, with a finite amount of data. Here, we consider projections based on function and derivative evaluations at the grid points, as in § 2.4. Thus, let P denote a projection operator — e.g. see Definitions 11, 12, or 13. Then the approximate solution method is defined by

$$\phi_{\text{approx}}(\vec{x}, t) = (P \circ A_{t,\,t-\Delta t}) \circ \cdots \circ (P \circ A_{2\Delta t,\,\Delta t}) \circ (P \circ A_{\Delta t,\,0})\ \Phi(\vec{x}). \tag{16}$$

Namely, to advance from time t to $t+\Delta t$, apply $P \circ A_{t+\Delta t,\,t}$ to the (approximate) solution at time t. The key simplification introduced by adding the projection step is that: in order to define the (approximate) solution at time $t + \Delta t$, only the data at the grid points is needed. At this point, all that is missing to make this approach into a computational scheme, is a method to obtain the grid point data at time $t + \Delta t$ from the (approximate) solution at time t. This is done in § 3.3.

As shown in § 2.1, the P_k projection is an $O(h^{n+1})$ accurate approximation for sufficiently smooth functions, where $n = 2k + 1$ and $h = \max_i \Delta x_i$. Thus, with $\Delta t \propto h$ the use of a locally $(n + 1)^{st}$ order time stepping scheme ensures that the full accuracy is preserved. As usual, the global error is in general one order less accurate, since $O(1/h)$ time steps are required to reach a given final time. Hence, a k-jet scheme can be up to n^{th} order accurate.

3.3. **Evolution of the k-Jet.** The projections defined in § 2.4 require knowledge of parts of the jet of the (approximate) solution at the grid points. Specifically, P_k^+ (see Definition 11) requires the partial k-jet, and P_k (see Definition 12) requires the total k-jet.

A natural way to find the jet at time $t + \Delta t$ is to consider the approximate advection operator (14). Since it defines an approximate solution everywhere, it also defines the solution's spacial derivatives. Thus, by differentiating (14), update rules for all the elements of the k-jet can be systematically inherited from the numerical scheme for the characteristics.

Definition 15. Jet schemes for which the update rule for the whole (total or partial) k-jet is derived from one single approximate advection scheme are called *superconsistent*.

Remark 8. Non-superconsistent jet schemes can be constructed, by first writing an evolution equation along the characteristics for each of the relevant derivatives — by differentiating Equation (1), and then applying some approximation scheme to each equation. On the one hand, these schemes can be less costly than superconsistent ones, because an ℓ^{th} order derivative only needs an accuracy that is ℓ orders below that of the function value — see Lemma 3. On the other hand, the benefits of an interpretation in function spaces are lost, such as the optimal coherence between all the entries in the k-jet, and the stability arguments of § 3.5.

Notice that a superconsistent scheme has a very special property: even though it is a fully discrete process, it updates the solution in time in a way that is equivalent to carrying out the process given by Equation (16) in the function space where the projection is defined. Below we present two possible ways to find the update rule for the k-jet: analytical differentiation and ε-finite differences. Up to a small approximation error, both approaches are equivalent. However, they can make a difference with the ease of implementation.

3.3.1. *Analytical differentiation.* Let $\mathcal{H}$ denote the approximate solution at time t. Since each time step ends with a projection step, $\mathcal{H}$ is a piecewise p-n polynomial, defined by the data at time t. Using the short notation $\vec{\mathcal{X}} = \vec{\mathcal{X}}(\vec{x},\, t+\Delta t,\, t)$, update formulas for the derivatives (here up to second order) are given by:

$$\begin{aligned}
A_{t+\Delta t,\,t}\,\phi(\vec{x},\,t) &= \mathcal{H}(\vec{\mathcal{X}},\,t),\\
\frac{\partial}{\partial x_i} A_{t+\Delta t,\,t}\,\phi(\vec{x},\,t) &= \frac{\partial \vec{\mathcal{X}}}{\partial x_i}\cdot D\,\mathcal{H}(\vec{\mathcal{X}},\,t),\\
\frac{\partial^2}{\partial x_i \partial x_j} A_{t+\Delta t,\,t}\,\phi(\vec{x},\,t) &= \frac{\partial^2 \vec{\mathcal{X}}}{\partial x_i\,\partial x_j}\cdot D\,\mathcal{H}(\vec{\mathcal{X}},\,t) + \left(\frac{\partial \vec{\mathcal{X}}}{\partial x_i}\right)^T \cdot D^2\,\mathcal{H}(\vec{\mathcal{X}},\,t)\cdot\frac{\partial \vec{\mathcal{X}}}{\partial x_j},
\end{aligned}$$

where $D\mathcal{H}$ is the Jacobian and $D^2\mathcal{H}$ the Hessian of $\mathcal{H}$. It should be clear how to continue this pattern for higher derivatives. Since $\mathcal{H}$ is a p-n polynomial, its derivatives are easy to compute analytically. The partial derivatives of $\vec{\mathcal{X}}$ follow from the ODE solver formulas, as the following example (using a Runge-Kutta scheme) illustrates.

Example 4. When tracking the total 1-jet (this would be called a *gradient-augmented scheme*, see [21]), the Hermite interpolant is fourth order accurate. Hence, in order to achieve full accuracy when scaling $\Delta t \propto h$, the advection operator should be approximated with a locally fourth order accurate time-stepping scheme. An example is the Shu-Osher scheme [25], which in superconsistent form looks as follows.

$$\begin{aligned}
\vec{x}_1 &= \vec{x} - \Delta t\;\vec{v}\,(\vec{x},\,t+\Delta t),\\
\nabla\,\vec{x}_1 &= I - \Delta t\;\nabla\,\vec{v}\,(\vec{x},\,t+\Delta t),\\
\vec{x}_2 &= \tfrac{3}{4}\,\vec{x} + \tfrac{1}{4}\,\vec{x}_1 - \tfrac{1}{4}\,\Delta t\;\vec{v}\,(\vec{x}_1,\,t),\\
\nabla\vec{x}_2 &= \tfrac{3}{4}\,I + \tfrac{1}{4}\,\nabla\vec{x}_1 - \tfrac{1}{4}\,\Delta t\;\nabla\,\vec{x}_1\cdot\nabla\,\vec{v}\,(\vec{x}_1,\,t),\\
\vec{x}_{\text{foot}} &= \tfrac{1}{3}\,\vec{x} + \tfrac{2}{3}\,\vec{x}_2 - \tfrac{2}{3}\,\Delta t\;\vec{v}\,(\vec{x}_2,\,t+\tfrac{1}{2}\,\Delta t),\\
\nabla\,\vec{x}_{\text{foot}} &= \tfrac{1}{3}\,I + \tfrac{2}{3}\,\nabla\,\vec{x}_2 - \tfrac{2}{3}\,\Delta t\;\nabla\,\vec{x}_2\cdot\nabla\,\vec{v}\,(\vec{x}_2,\,t+\tfrac{1}{2}\,\Delta t),\\
\phi(\vec{x},\,t+\Delta t) &= \mathcal{H}(\vec{x}_{\text{foot}},\,t),\\
(\nabla\phi)(\vec{x},\,t+\Delta t) &= \nabla\,\vec{x}_{\text{foot}}\cdot\nabla\,\mathcal{H}(\vec{x}_{\text{foot}},\,t).
\end{aligned}$$

Remark 9. When $\vec{x}_{\text{foot}}$ is on a grid hyperplane, Equation (10) in Definition 10 would require several ODE solves for derivatives that are discontinuous across the hyperplane (one for each of the local Hermite interpolants used in the adjacent grid cells). This would augment the computational cost without any increase in accuracy. Hence, in our implementations we do only one update, as follows: (i) A convention is selected that assigns the points on the grid hyperplanes as belonging to (a single) one of the cells that they are adjacent to. (ii) This convention then uniquely defines which Hermite interpolant should be used whenever $\vec{x}_{\text{foot}}$ is at a grid hyperplane.

Finally, we point out that

- The SSP property [12] is not required for the tracking of the characteristics, since this is not a process in which TVD stability plays any role. The use of the Shu-Osher scheme [25] in Example 4 is solely to illustrate the analytical differentiation process; any Runge-Kutta scheme of the appropriate order will do the job. In particular, when using P_2^+ a 5^{th} order Runge-Kutta scheme (e.g. the 5^{th} order component of the Cash-Karp method [5]) yields a globally 5^{th} order jet scheme.
- Jet schemes that result from using P_0^+ (p-linear interpolants) do not need any updates of derivatives, since they are based on function values only. In the one dimensional case, the CIR method [8] by Courant, Isaacson, and Rees is recovered. For these schemes a simple forward Euler ODE solver is sufficient.
- Using analytical differentiation, the update of a single partial derivative is almost as costly as updating all the partial derivatives of the same order. In particular, in order to update

the partial k-jet all the derivatives of order less than or equal to pk have to be updated — even though only a few with order greater than k are needed. Therefore, it is worthwhile to employ alternative ways to approximate the $|\vec{\alpha}| > k$ derivatives in the partial k-jet. One strategy is to use the grid-based finite differences described in § 2.5, assuming that a stable version can be found. Another strategy, which is essentially equivalent to analytical differentiation, is presented next.

3.3.2. *ε-finite differences.* Here we present a way to update derivatives in a superconsistent fashion that avoids the problem discussed in the third item at the end of § 3.3.1. In ε-finite differences the definition of a superconsistent scheme is applied by directly differentiating the approximately advected function $A_{t+\Delta t,\,t}\,\phi(\vec{x},\,t)$, or its derivatives. This is done using finite difference formulas, with a separation $\varepsilon \ll h$ chosen so that maximum accuracy is obtained. As shown below, ε-finite differences can be either employed singly, or in combination with analytic differentiation. In particular, in order to update the partial k-jet, one could update the total k-jet using analytical differentiation, and use ε-finite differences to obtain the remaining partial derivatives. The approach is best illustrated with a few examples.

Example 5. Here we present examples of how to implement the advect–and–project approach given by Equation (16) in two dimensions, with $P = P_1^+$ or $P = P_2^+$ (see Definition 11), using a combination of analytical differentiation and ε-finite differences. The error analysis below estimates the difference between the derivatives obtained using ε-finite differences and the values required by superconsistency. This difference can be made much smaller than the numerical scheme's approximation errors, so that stability (see § 3.5) is preserved. In this analysis, $\delta > 0$ characterizes the accuracy of the floating point operations. Namely, δ is the smallest positive number such that $1 + \delta \neq 1$ in floating point arithmetic.

- Advect–and–project using P_1^+. At time $t + \Delta t$ and at every grid point $(x_m,\, y_m)$, the values for $(\phi,\, D\phi,\, \phi_{xy})$ must be computed from the solution at time t. To do so, use the approximate advection solver $\vec{\mathcal{X}}$ to compute the approximately advected solution $\phi^{\vec{q}}$ at the four points $(x_m + q_1\,\varepsilon,\, y_m + q_2\,\varepsilon)$, where $\vec{q} \in \{-1,\, 1\}^2$. From these values $\phi^{(1,1)}$, $\phi^{(-1,1)}$, $\phi^{(1,-1)}$, and $\phi^{(-1,-1)}$ we obtain the desired derivatives at the grid point using the $O(\varepsilon^2)$ accurate stencils
$$\begin{aligned}
\phi &= \tfrac{1}{4}\,(\phi^{(1,1)} + \phi^{(-1,1)} + \phi^{(1,-1)} + \phi^{(-1,-1)})\,,\\
\phi_x &= \tfrac{1}{4\varepsilon}\,(\phi^{(1,1)} - \phi^{(-1,1)} + \phi^{(1,-1)} - \phi^{(-1,-1)})\,,\\
\phi_y &= \tfrac{1}{4\varepsilon}\,(\phi^{(1,1)} + \phi^{(-1,1)} - \phi^{(1,-1)} - \phi^{(-1,-1)})\,,\\
\phi_{xy} &= \tfrac{1}{4\varepsilon^2}(\phi^{(1,1)} - \phi^{(-1,1)} - \phi^{(1,-1)} + \phi^{(-1,-1)})\,.
\end{aligned}$$
- Advect–and–project using P_2^+. At time $t + \Delta t$ and at every grid point $(x,\, y) = (x_m,\, y_m)$, the values for $(\phi,\, D\phi,\, D^2\phi,\, \phi_{xxy},\, \phi_{xyy},\, \phi_{xxyy})$ must be computed from the solution at time t. To do so, use the approximate advection solver $\vec{\mathcal{X}}$ with analytic differentiation, to compute $(\phi,\, D\phi,\, D^2\phi)$ at the three points $(x_m,\, y_m)$ and $(x_m \pm \varepsilon,\, y_m)$. From this obtain $(\phi_{xxy},\, \phi_{xyy},\, \phi_{xxyy})$ at the grid point using $O(\varepsilon^2)$ accurate centered differences in x.

Error analysis. In both cases P_1^+ and P_2^+, the errors in the approximations are: $\mathcal{E} = O(\varepsilon^2) + O(\delta)$ for averages, $\mathcal{E} = O(\varepsilon^2) + O(\delta/\varepsilon)$ for first order centered ε-differences, and $\mathcal{E} = O(\varepsilon^2) + O(\delta/\varepsilon^2)$ for second order centered ε-differences. Thus the optimal choice for ε is $\varepsilon = O(\delta^{1/4})$, which yields $\mathcal{E} = O(\delta^{1/2})$ for all approximations.

Remark 10. In Remark 9 the scenario of a characteristic's foot point falling on a cell boundary is addressed. For ε-finite differences the analogous situation is more critical, since several characteristics, separated by $O(\varepsilon)$ distances, are used. Whenever the foot points for these characteristics fall in different cells, the interpolant corresponding to one and the same cell must be used in the ε-finite differences. This is always possible, since the interpolants are p-n-polynomials, and thus defined outside the cells.

Finally, we point out that, when computing derivatives with ε-finite differences, round-off error becomes important — the more so the higher the derivative. The error for an s^{th} order derivative, with an order $O(\varepsilon^r)$ accurate ε-finite difference, has the form $\mathcal{E} = O(\varepsilon^r) + O(\delta/\varepsilon^s)$. The best choice is then $\varepsilon = O(\delta^{1/(r+s)})$, which yields $\mathcal{E} = O(\delta^{r/(s+r)})$. Hence, under some circumstances it may be advisable to use high precision arithmetic and thus make δ very small. Given their relatively simple implementation, ε-finite differences can be the best choice in many situations.

3.4. **Boundary Conditions.** In this section we illustrate the fact that, due to their use of the method of characteristics, jet schemes treat boundary conditions naturally. For simplicity, we consider the two dimensional computational domain $\Omega = [0,1]^2$, equipped with a regular square grid of resolution $h = \frac{1}{N}$ — i.e. the grid points are $\vec{x}_{\vec{m}} = h\,\vec{m}$, where $\vec{m} \in \{0, \ldots, N\}^2$. In Ω we consider the advection equation

$$\phi_t + u\,\phi_x + v\,\phi_y = 0, \quad \text{with initial conditions} \quad \phi(x,\, y, 0) = \Phi(x,\, y). \tag{17}$$

Let us assume that $u > 0$ on both the left and right boundaries, $v < 0$ on the bottom boundary, and $v > 0$ on the top boundary. Hence the characteristics enter the square domain of computation Ω only through the left boundary, where boundary conditions are needed. For example: Dirichlet $\phi(0,\, y, t) = \xi(y,\, t)$ or Neumann $\phi_x(0,\, y, t) = \zeta(y,\, t)$. We will also consider the special situation where $u \equiv 0$ on the left boundary, in which case no boundary conditions are needed.

We now approximate (17) with a jet scheme, with the time step restricted by $\Delta t < h/\Lambda$, where Λ is an upper bound for $\sqrt{u^2 + v^2}$ — so that for each grid point $\|\vec{x}_{\text{foot}} - \vec{x}\| < h$. Then $\vec{x}_{\text{foot}} \in \Omega$ for any node $(x_m,\, y_n)$ with $m \neq 0$. Hence: at these nodes the data defining the solution can be updated by the process described earlier in this paper. We only need to worry about how to determine the data on the nodes for which $x = 0$. There are three cases to describe.

(i) Dirichlet boundary conditions. Then, on $x = 0$ we have: $\phi_t = \xi_t$ and $\phi_y = \xi_y$, while ϕ_x follows from the equation: $\phi_x = -(\phi_t + v\,\phi_y)/u$. Formulas for the higher order derivatives of ϕ at the left boundary can be obtained by differentiating the equation. For example, from

$$\phi_{ty} + u_y\,\phi_x + u\,\phi_{xy} + v_y\,\phi_y + v\,\phi_{yy} = 0$$

we can obtain ϕ_{xy}.

(ii) Neumann boundary conditions. Then, on $x = 0$, ϕ satisfies $\phi_t + v\,\phi_y = -u\,\zeta$. This is a lower dimensional advection problem, which can be solved[2] to obtain $\xi = \phi(0,\, y, t)$. Higher order derivatives follow in a similar fashion.

(iii) $u \equiv 0$ on the left boundary. Then $\vec{x}_{\text{foot}} \in \Omega$ for the nodes with $x = 0$, so that the data defining the solution can be updated in the same way as for the nodes with $x > 0$.

3.5. **Advection and Function Spaces: Stability.** In this section we show how the interpretation of jet schemes as a process of advect–and–project in function spaces, together with the fact that Hermite interpolants minimize the stability functional in Equation (7), lead to stability for jet schemes. We provide a rigorous proof of stability in one space dimension for constant coefficients. Furthermore, we outline (i) how the arguments could be extended to higher space dimensions, and (ii) how the difficulties in the variable coefficients case could be overcome. Note that, even though no rigorous stability proofs in higher space dimensions are given here, the numerical tests shown in § 4 indicate that the presented jet schemes are stable in the 2-D variable coefficients case.

3.5.1. *Control over averages in 1-D Hermite interpolants.* Let $k \in \mathbb{N}_0$, with $n = 2k+1$. Let $\mathcal{H}$ be an arbitrary 1-n-polynomial, and define

$$r_k = r_k(x) = \frac{1}{(n+1)!}\, x^{k+1}\,(1-x)^{k+1} \quad \text{and} \quad \mu_k = \mu_k(x) = r_k^{(k+1)}, \tag{18}$$

[2]Note that jet schemes can be easily generalized to problems with a source term. That is, replace Equation (1) by $\phi_t + \vec{v} \cdot \nabla\phi = S$, where S is a known function.

where $f^{(j)}$ indicates the j^{th} derivative of a function $f = f(x)$. Integration by parts then yields

$$\int_0^1 \mu_k(x)\, \mathcal{H}^{(k+1)}(x)\, \mathrm{d}x = (-1)^{k+1} \int_0^1 r_k(x)\, \mathcal{H}^{(n+1)}(x)\, \mathrm{d}x = 0, \tag{19}$$

where the boundary contributions vanish because of the $(k+1)^{st}$ order zeros of r_k at $x = 0$ and at $x = 1$. Integration by parts also shows that, for any sufficiently smooth function $\psi = \psi(x)$

$$\int_0^1 \mu_k(x)\, \psi^{(k+1)}(x)\, \mathrm{d}x = \left(\mu_k^{(0)}\, \psi^{(k)}\right)\Big|_{x=0}^{x=1} - \left(\mu_k^{(1)}\, \psi^{(k-1)}\right)\Big|_{x=0}^{x=1} + \cdots + \int_0^1 \psi(x)\, \mathrm{d}x,$$

where we have used that $\mu_k^{(k+1)} \equiv (-1)^{k+1}$. This can also be written in the form

$$\begin{aligned} \int_0^1 \mu_k(x)\, \psi^{(k+1)}(x)\, \mathrm{d}x - \int_0^1 \psi(x)\, \mathrm{d}x \quad &= \quad \left(\mu_k^{(0)}\, \psi^{(k)}\right)\Big|_{x=0}^{x=1} - \left(\mu_k^{(1)}\, \psi^{(k-1)}\right)\Big|_{x=0}^{x=1} \\ &\quad + \cdots + (-1)^k \left(\mu_k^{(k)}\, \psi^{(0)}\right)\Big|_{x=0}^{x=1}. \end{aligned} \tag{20}$$

In particular, assume now that $\mathcal{H}$ is the n^{th} Hermite interpolant for ψ in the unit interval, and apply (20) to both $\mathcal{H}$ and ψ. Then the resulting right hand sides in the equations are equal, so that the left hand sides must also be equal. Thus, from (19), it follows that

$$\int_0^1 \mathcal{H}(x)\, \mathrm{d}x = \int_0^1 \psi(x)\, \mathrm{d}x - \int_0^1 \mu_k(x)\, \psi^{(k+1)}(x)\, \mathrm{d}x. \tag{21}$$

This can be scaled, to yield a formula that applies to the n^{th} Hermite interpolant for ψ on any 1-D cell $x_n \le x \le x_{n+1} = x_n + h$

$$\int_{x_n}^{x_{n+1}} \mathcal{H}(x)\, \mathrm{d}x = \int_{x_n}^{x_{n+1}} \psi(x)\, \mathrm{d}x - h^{k+1} \int_{x_n}^{x_{n+1}} \mu_k\left(\frac{x - x_n}{h}\right) \psi^{(k+1)}(x)\, \mathrm{d}x. \tag{22}$$

The following theorem is a direct consequence of this last formula

Theorem 8. *In one dimension, consider the computational domain $\Omega = \{x \,|\, a \le x \le b\}$ and the grid: $x_n = a + n\,h$ — where $0 \le n \le N$, $h = (b-a)/N$, and $N > 0$ is an integer. Then, for any function $\psi \in C^k$, with integrable $(k+1)^{st}$ derivative*

$$\int_a^b (P_k^+ \psi)(x)\, \mathrm{d}x = \int_a^b \psi(x)\, \mathrm{d}x - \, h^{k+1} \int_a^b E_k\left(\frac{x-a}{h}\right) \psi^{(k+1)}(x)\, \mathrm{d}x,$$

where $E_k = E_k(z)$ is the periodic function of period one defined by $E_k(z) = \mu_k(z \bmod 1)$.

3.5.2. *Stability for 1-D constant advection.* Using Theorems 8 and 5 we can now prove stability for jet schemes in one dimension, and with a constant advection velocity. For simplicity we will also assume periodic boundary conditions.

Theorem 9. *Under the same hypothesis as in Theorem 8, consider the constant coefficients advection equation $\phi_t + v\,\phi_x = 0$ in the computational domain Ω, with periodic boundary conditions $\phi(b,\, t) = \phi(a,\, t)$, and initial condition $\phi(x,\, 0) = \Phi(0)$. Approximate the solution by the jet scheme defined by the advect–and–project process of § 3.2 and the projection P_k^+, with a time step Δt satisfying $\Delta t \ge O(h^{k+1})$. Then this scheme is stable, in the sense described next. Let ϕ_n be the numerical solution at time $t_n = n\,\Delta t$, and consider a fixed time interval $0 \le t_n \le T$ — for some fixed initial condition. Then:*

$$\left\|\phi_n^{(\ell)}\right\|_\infty \text{ and } \left\|\phi_n^{(k+1)}\right\|_2 \text{ remain bounded as } h \to 0, \text{ where } 0 \le \ell \le k.$$

Notation: here $\|\cdot\|_p$ is the L^p norm in Ω, and $f^{(\ell)}$ denotes the ℓ^{th} derivative of a function f.

Proof. For any Runge-Kutta scheme (and many other types of ODE solvers), the approximate advection operator is given by the shift operator

$$(A\,\phi)(x,\,t) = \phi(x - v\,\Delta t,\,t), \tag{23}$$

where we use the notation $A = A_{t+\Delta t,\,t}$ — since $A_{t+\Delta t,\,t}$ is independent of time. Let the numerical solution at time $t = t_n = n\,\Delta t$ be denoted by ϕ_n. Note that $\phi_n = (P_k^+ \circ A)^n\,\Phi \in C^k$.

From Theorem 5, Equation 23, and $\phi_{n+1} = (P_k^+ \circ A)\,\phi_n$ it follows that

$$\left\|\phi_{n+1}^{(k+1)}\right\|_2 \leq \left\|\phi_n^{(k+1)}\right\|_2 \quad \Longrightarrow \quad \left\|\phi_n^{(k+1)}\right\|_2 \leq \left\|\Phi^{(k+1)}\right\|_2. \tag{24}$$

Furthermore, from Theorem 8

$$\mathcal{M}(\phi_{n+1}) = \mathcal{M}(\phi_n) - h^{k+1} \int_a^b E_k\left(\frac{x-a}{h}\right)\,\phi_n^{(k+1)}(x - v\,\Delta t)\,\mathrm{d}x,$$

where $\mathcal{M}$ denotes the integral from a to b. From this, using the Cauchy-Schwarz inequality, we obtain the estimate

$$|\mathcal{M}(\phi_{n+1})| \leq |\mathcal{M}(\phi_n)| + h^{k+1}\,C\,\left\|\phi_n^{(k+1)}\right\|_2, \tag{25}$$

where

$$C^2 = \int_a^b \left(E_k\left(\frac{x-a}{h}\right)\right)^2\,\mathrm{d}x = (b-a)\int_0^1 \left(E_k(x)\right)^2\,\mathrm{d}x = \frac{b-a}{n+2}\,\left(\frac{(k+1)!}{(n+1)!}\right)^2.$$

From (25), using (24), it follows that

$$|\mathcal{M}(\phi_n)| \leq |\mathcal{M}(\Phi)| + \frac{t_n}{\Delta t}\,h^{k+1}\,C\,\left\|\Phi^{(k+1)}\right\|_2. \tag{26}$$

Now, for any of the (periodic) functions $g = \phi_n^{(\ell)}$ — $0 \leq \ell \leq k$, we can write

$$g(x) = \overline{g} + \int_a^b G(x-y)\,g^{(1)}(y)\,\mathrm{d}y, \tag{27}$$

where $\overline{g}$ is the average value for g and G is the Green's function defined by: G is periodic of period $b-a$, and $G(x) = \frac{1}{2} - \frac{x}{b-a}$ for $0 < x < b-a$. Note that $\overline{g} = 0$ for $1 \leq \ell \leq k$, and $\overline{g} = \frac{1}{b-a}\,\mathcal{M}(\phi_n)$ for $\ell = 0$.

Now use (24), (27), and the Cauchy-Schwarz inequality, to find an h-independent bound for $\left\|\phi_n^{(k)}\right\|_\infty$ — hence also for $\left\|\phi_n^{(k)}\right\|_2$. Repeat the process for $\phi_n^{(k-1)}$, and so on — all the way down to $\phi_n^{(0)}$. In the last step, (26) is needed. □

Notice that Theorem 9 says nothing about the L^∞ norm of $\phi_n^{(k+1)}$. A natural question is then: what can we say about $\phi_n^{(k+1)}$ — beyond the statement in Equation (24)? In particular, notice that there are stricter bounds on the growth of derivatives than the one given by Theorem 5 , since $\mathcal{F}$ is actually minimized in each cell individually. Can these, as well results similar to the one in Theorem 8, be exploited to obtain more detailed information about the behavior of the solutions (and derivatives) given by jet schemes? This is something that we plan to explore in future work.

3.5.3. *Stability for 1-D variable advection.* An extension of this proof to the case of variable advection will be considered in future work.

3.5.4. *Stability for higher dimensions and constant coefficients.* In several space dimensions, assume that the advection velocity does not involve rotation — specifically: the advected grid hyperplanes remain parallel to the coordinate hyperplanes. Then all the derivatives needed to compute the stability functional $\mathcal{F}$ (as well as all the derivatives needed in the proof of Theorem 5) remain defined after advection. Thus Theorem 5 can be used, as in the proof of Theorem 9, to control the growth of $\left\| \partial^{\vec{\beta}} \phi \right\|_2$ — where $\vec{\beta}$ is as in Equation (7). In particular, when the advection velocity is constant,

$$\left\| \partial^{\vec{\beta}} \phi \right\|_2 \leq \left\| \partial^{\vec{\beta}} \Phi \right\|_2 \tag{28}$$

follows. Just as in the one dimensional case, (28) alone is not enough to guarantee stability. For example: in the case with periodic boundary conditions, control over the growth of $\partial^{\vec{\beta}} \phi$ still leaves the possibility of unchecked growth of partial means, i.e. components of the solution of the form $\phi = \sum f_j(\vec{x})$, where f_j does not depend on the variable x_j. However, the Hermite interpolant for any such component reduces to the Hermite interpolant of a lower dimension, to which a lower dimensional version of the stability functional $\mathcal{F}$ applies — thus bounding their growth. The appropriate generalization of Theorem 8 to control the partial means of ϕ is the subject of current research.

3.5.5. *Stability for higher dimensions and variable coefficients.* In the presence of rotation, Theorem 5 fails. Then, after advection, the derivatives involved are only defined in the sense of Definition 10, and the integrations by parts used to prove Theorem 5 are no longer justified. On the other hand (see the discussion at the beginning of § 2.4), the jumps in the derivatives that cause the failure are small. Hence we conjecture that, in this case, a "corrected" version of Theorem 5 will state that $\mathcal{F}[\mathcal{H}_\phi] \leq \mathcal{F}[\phi] + C(h)$, where $C(h)$ is a small correction that depends on the smallness of the jumps. In this case a simultaneous proof of stability and convergence might be possible — simultaneous because the "small jumps" property is valid only as long as the numerical solution is close to the actual solution.

4. Numerical Results

As a test for both the accuracy and the performance of jet schemes, we consider a version of the classical "vortex in a box" flow [3, 17], adapted as follows. On the computational domain $(x,\, y) \in [0,1]^2$, and for $t \in [0,\, t_{\text{final}}]$, we consider the linear advection equation (1) with the velocity field

$$\vec{v}(x,\, y,\, t) = \cos\left(\tfrac{\pi t}{T}\right) \begin{pmatrix} \sin^2(\pi x)\, \sin(2\pi y) \\ -\sin(2\pi x)\, \sin^2(\pi y) \end{pmatrix}. \tag{29}$$

This velocity field at $t = 0$ is shown in Figure 1. The maximum speed that ever occurs is 1. We prescribe periodic boundary conditions on all sides of the computational domain, and provide periodic initial conditions — note that the velocity field (29) is C^∞ everywhere when repeated periodically beyond $[0,\, 1]^2$. This test is a mathematical analog of the famous "unmixing" experiment, presented by Heller [13], and popularized by Taylor [27]. It models the passive advection of a solute concentration by an incompressible fluid motion, on a time scale where diffusion can be neglected. The velocity field swirls the concentration forth and back (possibly multiple times) around the center point $(\frac{1}{2},\, \frac{1}{2})$ in such a fashion that the equi-concentration contours of the solution become highly elongated at maximum stretching. The parameters T and t_{final} determine the amount of maximum deformation and the number of swirls. In all tests, we choose $t_{\text{final}} = \ell\, T$, where $\ell \in \mathbb{N}$. This implies that $\phi(x, y, t_{\text{final}}) = \phi(x,\, y,\, 0)$, i.e. at the end of each computation, the solution returns to its initial state. Below, we compare the accuracy and convergence rates of jet schemes with WENO methods (§ 4.1), investigate the accuracy and stability of different versions of jet schemes (§ 4.2), demonstrate the performance of jet schemes on a benchmark test (§ 4.3), and finally compare the computational cost and efficiency of all the numerical schemes considered (§ 4.4).

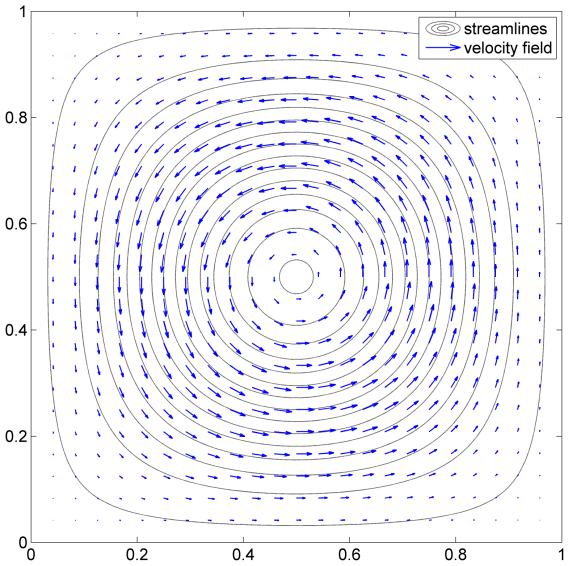

Figure 1. Streamlines and quiver plot of the velocity field used for the test cases.

4.1. **Test of Accuracy and Convergence Rate.** We first test the relative accuracy and convergence rate of jet schemes in comparison with classical WENO approaches. For this test, we choose $t_{\text{final}} = T = 1$ and initial conditions $\phi(x, y, 0) = \cos(2\pi x)\cos(4\pi y)$. This choice of parameters yields a moderate deformation, and the smallest structures in the solution are well resolved for $h \leq 1/50$. In all error convergence tests, the error with respect to the true solution at t_{final} is measured in the L^∞ norm.

The results of the numerical error analysis are shown in Figure 2. The jet schemes considered here are the ones based on the projection P_k^+, given in Definition 11. Specifically, we consider a bi-linear scheme ($k = 0$), denoted by black dots; a bi-cubic scheme ($k = 1$), implemented using analytical differentiation for the partial 1-jet, as described in § 3.3.1, marked by black squares; and the bi-quintic scheme ($k = 2$) described in the second bullet point in Example 5, denoted by black triangles.

As reference schemes, we consider the classical WENO finite difference schemes described in [15]. We use schemes of orders 1, 3, and 5, constructed as follows. Unless otherwise noted, time stepping is done with the maximum time step that stability admits. The first order version is the simple 2-D upwind scheme, using forward Euler in time (with a time step $\Delta t = \frac{1}{\sqrt{2}}h$). The time stepping of the third order WENO is done using the Shu-Osher scheme [25] (with $\Delta t = h$). Lacking a simple fifth order SSP Runge-Kutta scheme [9, 10], the fifth order WENO is advanced forward in time using the same third order Shu-Osher scheme, however, with a time step $\Delta t = h^{\frac{5}{3}}$. This yields a globally order $O(h^5)$ scheme. For both WENO3 and WENO5, the parameters $\varepsilon = 10^{-6}$ and $p = 2$ (in the notation of [15]) are used. These choices are commonly employed when WENO is used in a black-box fashion (i.e. without adapting the parameters to the specific solution). In Figure 2, the numerical errors incurred with the WENO schemes are shown by gray curves and symbols, namely: dots for upwind; squares for WENO3; and triangles for WENO5.

This test demonstrates the potential of jet schemes in a remarkable fashion. While the first order jet scheme is very similar to the 2-D upwind method, the jet schemes of orders 3 and 5 are strikingly more accurate than their WENO counterparts. For equal resolution h, the errors for the bi-cubic jet scheme are about 100 times smaller than with WENO3, and for the bi-quintic jet scheme, the errors are about 1000 times smaller than with WENO5. This implies that a high order jet scheme achieves the same accuracy as a WENO scheme (of the same order) with a resolution that is about 4 times as coarse. This reflects the fact that jet schemes possess subgrid

resolution, i.e. structures of size less than h can be represented due to the high degree polynomial interpolation [21].

4.2. **Accuracy and Stability of Grid-Based Finite Difference Schemes.** In a second test, the accuracy and stability of schemes that use grid-based finite difference reconstructions of the higher derivatives are tested. As in the test described in § 4.1, we choose the deformation $T = 1$, however, now $t_{\text{final}} = 20$, i.e. the solution is moved back and forth multiple times. Through this approach, enough time steps are taken so that unstable approaches in fact show their instabilities.

We compare jet schemes of order 3 ($k = 1$) and order 5 ($k = 2$), each in two versions: first, we consider the approaches based on the projection P_k^+ (see Definition 11), to which the stability arguments in § 3.5 apply. Second, we implement schemes that construct the partial k-jet from the total k-jet, using optimally local grid-based finite difference approximations. As described in § 2.5, these types of schemes are generally less costly than approaches based on P_k^+, and therefore worth investigating (see also § 4.4).

Specifically, we consider a bi-cubic scheme that tracks the total 1-jet, and uses the grid-based finite difference approximation of ϕ_{xy} described in Example 3. Furthermore, we implement a bi-quintic scheme that tracks the total 2-jet, and approximates ϕ_{xxy}, ϕ_{xyy}, and ϕ_{xxyy} using grid-based finite differences, as follows. In a cell $Q = [0,\, h]^2$, with the index $\vec{q} \in \{0,\, 1\}^2$ corresponding to the vertex $\vec{x}_{\vec{q}} = \vec{q}\,h$, the derivatives at the vertex $(0, 0)$ are approximated by:

$$
\begin{aligned}
\phi^{(0,0)}_{xxy} &= \tfrac{1}{h^2}\left(-\phi^{(0,0)}_{x} + \phi^{(1,0)}_{x} + \phi^{(0,1)}_{x} - \phi^{(1,1)}_{x}\right) + \tfrac{6}{h^2}\left(-\phi^{(0,0)}_{y} + \phi^{(1,0)}_{y}\right) \\
&\quad + \tfrac{1}{2h}\left(-\phi^{(0,0)}_{xx} - \phi^{(1,0)}_{xx} + \phi^{(0,1)}_{xx} + \phi^{(1,1)}_{xx}\right) + \tfrac{1}{h}\left(-4\,\phi^{(0,0)}_{xy} - 2\,\phi^{(1,0)}_{xy}\right), \\
\phi^{(0,0)}_{xyy} &= \tfrac{6}{h^2}\left(-\phi^{(0,0)}_{x} + \phi^{(0,1)}_{x}\right) + \tfrac{1}{h^2}\left(-\phi^{(0,0)}_{y} + \phi^{(1,0)}_{y} + \phi^{(0,1)}_{y} - \phi^{(1,1)}_{y}\right) \\
&\quad + \tfrac{1}{h}\left(-4\,\phi^{(0,0)}_{xy} - 2\,\phi^{(0,1)}_{xy}\right) + \tfrac{1}{2h}\left(-\phi^{(0,0)}_{yy} + \phi^{(1,0)}_{yy} - \phi^{(0,1)}_{yy} + \phi^{(1,1)}_{yy}\right), \\
\phi^{(0,0)}_{xxyy} &= \tfrac{6}{h^3}\left(\phi^{(0,0)}_{x} - \phi^{(1,0)}_{x} - \phi^{(0,1)}_{x} + \phi^{(1,1)}_{x}\right) + \tfrac{6}{h^3}\left(\phi^{(0,0)}_{y} - \phi^{(1,0)}_{y} - \phi^{(0,1)}_{y} + \phi^{(1,1)}_{y}\right) \\
&\quad + \tfrac{1}{h^2}\left(7\,\phi^{(0,0)}_{xy} - \phi^{(1,0)}_{xy} - \phi^{(0,1)}_{xy} - 5\,\phi^{(1,1)}_{xy}\right).
\end{aligned}
$$

It can be verified by Taylor expansion that these are $O(h^{6-s})$ accurate approximations to the respective derivatives of order s. The corresponding approximations at the vertices $(1, 0)$, $(0, 1)$ and $(1, 1)$ are obtained by symmetry.

The convergence errors for these jet schemes are shown in Figure 3. As already observed in § 4.1, the bi-cubic (shown by black squares) and bi-quintic (denoted by black triangles) jet schemes that are based on tracking the partial k-jet are stable and $(2k+1)^{st}$ order accurate. The bi-cubic jet scheme that approximates ϕ_{xy} by grid-based finite differences (indicated by open squares) is stable and third order accurate as well, albeit with a larger error constant than the "pure" version based on P_1^+. In contrast, the bi-quintic jet scheme that approximates ϕ_{xxy}, ϕ_{xyy}, and ϕ_{xxyy} by grid-based finite differences (denoted by open triangles) is unstable. Interestingly, the instability is extremely mild: even though many time steps are taken in this test, the instability only shows up for $h < 0.008$. This phenomenon is both alarming and promising. Alarming, because it demonstrates that with jet schemes that are not based on the partial k-jet, stability is not assured, and generalizations of the stability results given in § 3.5 are needed. Promising, because it is plausible that these jet schemes could be stabilized without significantly having to diminish their accuracy.

4.3. **Test of Performance on a Level-Set-Type Example.** In this test we assess the practical accuracy of the considered numerical approaches, by following a curve of equi-concentration of the solution. The parameters are now $t_{\text{final}} = T = 10$, and the initial conditions are given by a periodic Gaussian hump: $\phi(x,\, y,\, 0) = \sum_{i,\, j \in \mathbb{Z}} g(x-i,\, y-j)$, where $g(x,\, y) = \exp(-10((x-x_0)^2+(y-y_0)^2))$ with $x_0 = 0.5$ and $y_0 = 0.75$. We examine the time-evolution of the contour $\Gamma(t) = \{(x,\, y) \in$

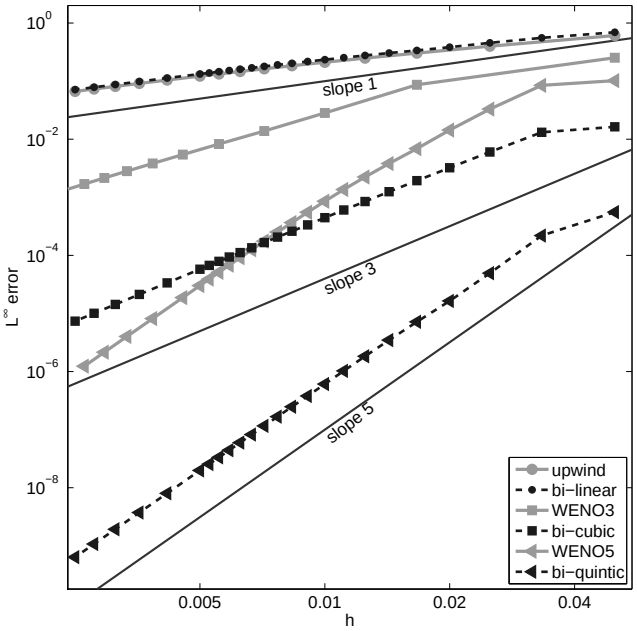

Figure 2. Numerical convergence rates for jet schemes of orders 1, 3, and 5, in comparison with WENO schemes of the same orders.

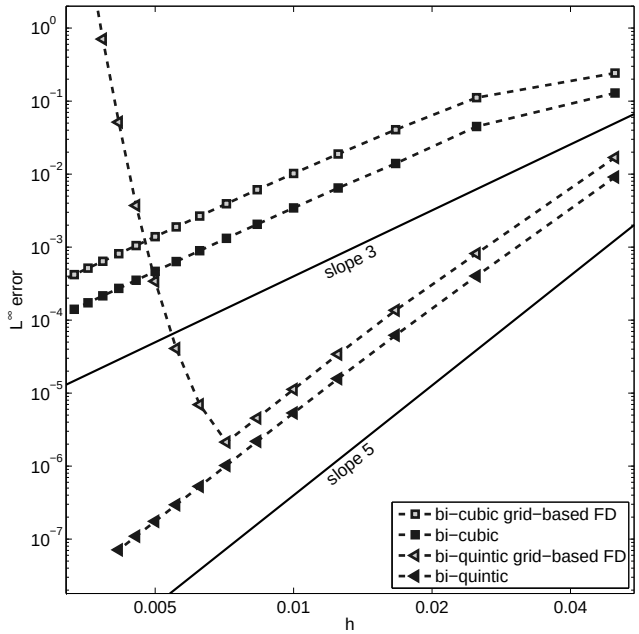

Figure 3. Numerical convergence rates for jet schemes of orders 3 and 5, comparing schemes based on the full partial jet with schemes using grid-based finite differences. The latter type of approach turns out unstable for the order 5 jet scheme.

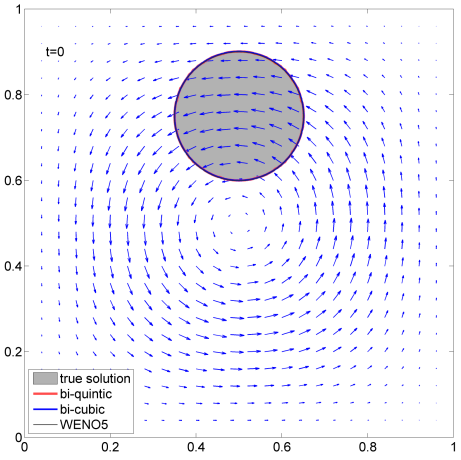

Figure 4. Swirl test: velocity field and initial conditions.

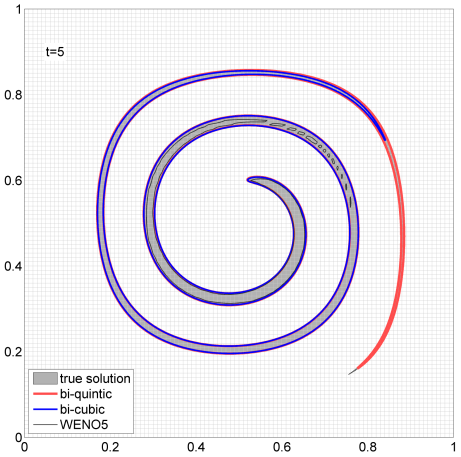

Figure 5. Swirl test: $\phi = \phi_c$ contour at maximum deformation ($t = \frac{T}{2}$).

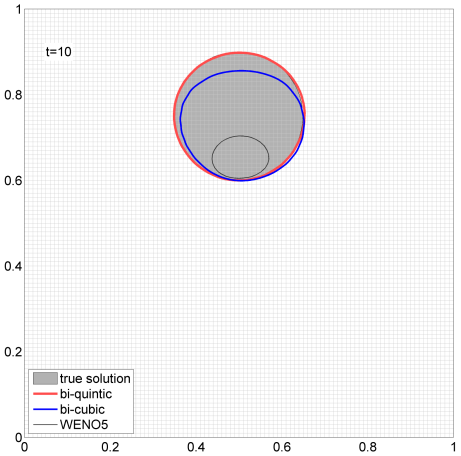

Figure 6. Swirl test: $\phi = \phi_c$ contour at the final time ($t = T$).

$\Omega : \phi(x, y, t) = \phi_c\}$ that corresponds to the concentration $\phi_c = \exp(-10\, r^2)$, where $r = 0.15$. The initial (and final) contour $\Gamma(0)$ is almost an exact circle of radius r, centered at (x_0, y_0). At maximum deformation, $\Gamma(\frac{T}{2})$ is highly elongated.

This test is well-known in the area of level set approaches [22]. For these, only one specific contour is of interest, and it is common practice to modify the other contours, e.g. by adding a reinitialization equation [26], or by modifying the velocity field away from the contour of interest, using extension velocities [1]. Since here the interest lies on more general advection problems (1), such as the convection of concentration fields, no level-set-method-specific modifications are considered. However, this does not mean that these procedures cannot be applied in the context of jet schemes. In fact, proper combinations of jet schemes with reinitialization are the subject of current research.

We apply the bi-cubic and the bi-quintic jet schemes, as well as the WENO5 scheme (all as described in § 4.1), on a grid of resolution $h = {}^1/_{100}$. The resulting equi-concentration contours are shown in Figure 4 (initial conditions), Figure 5 (maximum deformation), and Figure 6 (final state). The contour obtained with WENO5 is thin and black, the contour obtained with the bi-cubic jet scheme is medium thick and blue, and the contour obtained with the bi-quintic jet scheme is thick and red. The true solution (approximated with high accuracy using Lagrangian markers), is shown as a gray patch. For the considered resolution, both jet schemes yield more accurate results than WENO5. In fact, the fifth order jet scheme yields an almost flawless approximation to the true solution, on the scale of interest. As alluded to in § 4.2, the subgrid resolution of the jet schemes is of great benefit in resolving the thin elongated structure.

4.4. **Computational Cost and Efficiency.** In order to get an impression of the relative computational cost and efficiency of the presented schemes, we measure the CPU times that various versions of jet schemes and WENO require to perform given tasks. We use the same test as in § 4.1, and apply the following versions of jet schemes: (a) jet schemes of orders 1, 3, and 5, that are based on the partial k-jet; and (b) jet schemes of orders 3 and 5, that are based on the total k-jet and use grid-based finite difference approximations for the missing derivatives of the partial k-jet (see § 4.2 for a description of these schemes). In addition, we consider the following versions of WENO and finite difference schemes to serve as reference methods: simple linear 2-D upwinding with forward Euler in time (with $\Delta t = \frac{1}{\sqrt{2}} h$), WENO3 [15] advanced with the third order Shu-Osher method [25], and WENO5 with the Cash-Karp method [5] used for the time stepping. Note that unlike the case where convergence is tested (see § 4.1), for a fair comparison of CPU times a true fifth order time stepping scheme must be used. While the Cash-Karp method is not SSP, for the considered test case no visible oscillations are observed. Furthermore, in order to give a fair treatment to WENO, one must be careful with the choice of the parameter ε that WENO schemes require [15]. The meaning of the parameter is that for $\varepsilon \ll h^2$, spurious oscillations are minimized at the expense of a degradation of convergence rate, while for $\varepsilon \gg h^2$, the schemes reduce to (unlimited) linear finite differences of the respective order. We therefore consider two extremal cases for both WENO3 and WENO5: one with a tiny $\varepsilon = 10^{-10}$, and one that is linear finite differences (called "linear FD3" and "linear FD5") with upwinding (with all limiting routines removed). Practical WENO implementations will be somewhere in between these two extremal cases.

As a first comparison of the computational costs of these numerical approaches, we run all of them for a fixed resolution ($h = {}^1/_{150}$), and record the resulting L^∞ errors and CPU times, as shown in Table 1. One can see a clear separation between methods of different orders both in terms of accuracy and in terms of CPU times. When comparing approaches of the same order one can observe that for identical resolutions, jet schemes incur a significantly larger computational effort than WENO and linear finite difference schemes. However, this increased cost comes at the benefit of a significant increase in accuracy. Similarly, it is apparent that the use of grid-based finite difference approximations for parts of the k-jet can reduce the cost of jet schemes significantly. However, this tends to come at an expense in accuracy, in addition to the fact that

Order	Approach	L^∞ error	CPU time/sec.
1	bi-linear jet scheme	1.69×10^{-1}	0.400
1	upwind	1.92×10^{-1}	0.456
3	bi-cubic jet scheme	1.35×10^{-4}	7.91
3	bi-cubic jet scheme grid-based FD	2.31×10^{-4}	5.99
3	WENO3/RK3 ($\varepsilon = 10^{-10}$)	1.21×10^{-2}	1.61
3	linear FD3/RK3	1.54×10^{-3}	1.15
5	bi-quintic jet scheme	8.23×10^{-8}	103
5	bi-quintic jet scheme grid-based FD	1.32×10^{-7}	36.3
5	WENO5/RK5 ($\varepsilon = 10^{-10}$)	1.25×10^{-4}	19.3
5	linear FD5/RK5	2.15×10^{-5}	19.1

Table 1. CPU times for a fixed resolution $h = {}^1/_{150}$ with various jet schemes and WENO schemes.

Order	Approach	Res. h	L^∞ error	time/sec.
1	bi-linear jet scheme	${}^1/_{1600}$	1.89×10^{-2} (*)	441 (*)
1	upwind	${}^1/_{1600}$	2.28×10^{-2} (*)	482 (*)
3	bi-cubic jet scheme	${}^1/_{167}$	9.86×10^{-5}	11.4
3	bi-cubic jet scheme grid-based FD	${}^1/_{200}$	9.86×10^{-5}	13.0
3	WENO3/RK3 ($\varepsilon = 10^{-10}$)	${}^1/_{1350}$	9.61×10^{-5}	1110
3	linear FD3/RK3	${}^1/_{380}$	9.65×10^{-5}	18.7
5	bi-quintic jet scheme	${}^1/_{35}$	9.76×10^{-5}	1.55
5	bi-quintic jet scheme grid-based FD	${}^1/_{40}$	8.62×10^{-5}	0.821
5	WENO5/RK5 ($\varepsilon = 10^{-10}$)	${}^1/_{158}$	9.71×10^{-5}	21.4
5	linear FD5/RK5	${}^1/_{110}$	9.87×10^{-5}	11.7

Table 2. CPU times required to achieve an L^∞ error of less than 10^{-4}. Note the lower accuracy of the first order schemes.

the resulting scheme could become unstable, as demonstrated in § 4.2. In fact, the CPU times for the bi-quintic jet scheme with grid-based finite differences shown in Table 1 must be interpreted as a guideline only, since without further stabilization this version of the method is of no practical use.

As a second test we compare the schemes in terms of their actual efficiency. To that end, we measure the CPU times that the schemes require to achieve a given accuracy, i.e. for each scheme we choose the coarsest resolution such that the L^∞ error is below 10^{-4}, and for this resolution we report the required CPU time. The results of this test are shown in Table 2. Note that the first order schemes were computed at a maximum resolution of $h = {}^1/_{1600}$. A simple extrapolation reveals that in order to reach an accuracy of 10^{-4}, CPU times of more than 50 years would be required. In contrast, the third and fifth order methods yield reasonable CPU times, except for WENO3 with $\varepsilon = 10^{-10}$, which due to the degraded convergence rate requires an unreasonably fine resolution to reach the target accuracy. In fact, for this very smooth test case, linear finite differences present themselves as rather efficient methods, which achieve descent accuracies at a very low cost per time step. However, the results show that jet schemes are actually more efficient than their WENO/finite difference counterparts. Their higher cost per time step is more than outweighed by the significant increase in accuracy. Specifically, third order jet schemes are observed to be more efficient than finite differences by a factor of 1.5, and fifth order jet schemes by a factor of at least 7.5.

The computational efficiency of jet schemes goes in addition to their structural advantages, such as optimal locality and subgrid resolution. A more detailed efficiency comparison of jet schemes with WENO, and also with Discontinuous Galerkin approaches, is given in [6].

5. Conclusions and Outlook

The jet schemes introduced in this paper form a new class of numerical methods for the linear advection equation. They arise from a conceptually straightforward formalism of advect–and–project in function spaces, and provide a systematic methodology for the construction of high (arbitrary) order numerical schemes. They are based on two ingredients: an ODE solver to track characteristic curves, and Hermite interpolation. Unlike finite volume methods or discontinuous Galerkin approaches, jet schemes have (currently) a limited area of application: linear advection equations that are approximated on rectangular grids. However, under these circumstances, they have several interesting advantages: a natural treatment of the flow nature of the equation and of boundary conditions, no CFL restrictions, no requirement for SSP ODE solvers, and optimal locality, i.e. the update rule for the data on a grid point uses information in only a single grid cell.

Jet schemes achieve high order by carrying a portion of the jet of the solution as data. To derive an update rule in time for the data, the notion of superconsistency is introduced. This concept provides a way to systematically inherit update procedures for the derivatives from an approximation scheme (ODE solver) for the characteristic curves. Various alternative approaches are presented for how to track different portions of the jet of the solution, and for how to specifically implement superconsistent schemes. One example is the grid-based reconstruction of higher derivatives, which on the one hand can dramatically reduce the computational cost, but on the other hand could destabilize the scheme. Another example is the idea of ε-finite differences, which requires diligent considerations of round-off errors, but can lead to simpler implementations and also reduce the computational cost.

The formal equivalence of superconsistent schemes to advect–and–project approaches in function spaces gives rise to a notion of stability: the projection step (which is based on a piece-wise application of Hermite interpolation) minimizes a stability functional, while the increase by the advection step can be appropriately bounded. The stability functional gives control over certain derivatives of the numerical solution, and thus bounds the occurrence of oscillations. In this paper, a full proof of stability for the 1-D constant coefficient case is provided.

Investigations of the numerical error convergence and performance of high order jet schemes show that they tend to be more costly, but also strikingly more accurate than classical WENO schemes of the same respective orders. It is observed that, for the same resolution, a jet scheme incurs 2–7 times the computational cost of a WENO scheme. However, the numerical experiments also show that a jet scheme achieves the same accuracy as a WENO scheme (of the same order) at a 3–4 times coarser grid resolution. Efficiency comparisons reveal that jet schemes tend to achieve the same accuracy as an efficient WENO scheme at a 1.5–7 times smaller computational cost. In comparison with WENO schemes with a sub-optimal parameter choice, the efficiency gains of jet schemes are factors of 14–85. An application to a model of the passive advection of contours demonstrates the practical accuracy of jet schemes. A particular feature of jet schemes is their potential to represent structures that are smaller than the grid resolution.

This paper should be seen as a first step into the class of jet schemes. Their general interpretation as advect–and–project approaches, their conceptual simplicity, and their apparent accuracy make them promising candidates for numerical methods for certain important types of advection problems. However, many questions remain to be answered, and many aspects remain to be investigated, before one could consider them generally "competitive". The listing below gives a few examples of some of the aspects where further research and development is still needed.

Analysis. A general proof of stability in arbitrary space dimensions, and for variable coefficients, is one important goal for jet schemes. A fundamental understanding of the stability of these methods is particularly important when combined with grid-based finite difference approximations to higher derivatives.

Efficiency. Some ideas presented here (e.g. grid-based or ε-finite differences) lead to significant reductions in implementation effort and computational cost, but they also indicate that the stability of such new approaches is not automatically guaranteed. One can safely expect that there are many other ways to implement jet schemes more easily and more efficiently. The investigation and analysis of such ideas is an important target for future research.

Applicability. The field of linear advection problems on rectangular grids is not as limited as it may seem. Many examples of interface tracking fall into this area. Of particular interest is the combination of jet schemes with adaptive mesh refinement techniques. The optimal locality of jet schemes promises a straightforward generalization to adaptive meshes.

Generality. Since all that jet schemes need are a method for tracking the characteristics, and an interpolation procedure, they are not fundamentally limited to rectangular grids. Similarly, they could be generalized to allow for non-conforming boundaries, and thus apply to general unstructured domains. Furthermore, it is plausible that the idea of tracking derivatives can be advantageous for nonlinear Hamilton-Jacobi equations, hyperbolic conservation laws, or diffusion problems as well. In all these cases new challenges arise, such as the occurrence of shocks or the absence of characteristics.

Acknowledgments

The authors would like to acknowledge support by the National Science Foundation through grant DMS–0813648. In addition, R. R. Rosales and B. Seibold wish to acknowledge partial support by the National Science Foundation through through grants DMS–1007967 and DMS–1007899, as well as grants DMS–1115269 and DMS–1115278, respectively. Further, J.-C. Nave wishes to acknowledge partial support by the NSERC Discovery Program.

References

[1] D. Adalsteinsson and J. A. Sethian. The fast construction of extension velocities in level set methods. *J. Comput. Phys.*, 148:2–22, 1999.

[2] A. V. Arutyunov. *Optimality conditions: Abnormal and degenerate problems*. Kluwer Academic Publishers, The Netherlands, 2000.

[3] J. B. Bell, P. Colella, and H. Glaz. A second-order projection method for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 85:257–283, 1989.

[4] M. J. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53:484–512, 1984.

[5] J. R. Cash and A. H. Karp. A variable order Runge-Kutta method for initial value problems with rapidly varying right-hand sides. *ACM T. Math. Software*, 16:201–222, 1990.

[6] P. Chidyagwai, J.-C. Nave, R. R. Rosales, and B. Seibold. A comparative study of the efficiency of jet schemes. *Int. J. Numer. Anal. Model.-B*, (under review), 2011.

[7] B. Cockburn and C.-W. Shu. The local Discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM J. Numer. Anal.*, 35(6):2440–2463, 1988.

[8] R. Courant, E. Isaacson, and M. Rees. On the solution of nonlinear hyperbolic differential equations by finite differences. *Comm. Pure Appl. Math.*, 5:243–255, 1952.

[9] S. Gottlieb. On high order strong stability preserving Runge-Kutta and multi step time discretizations. *Journ. Scientif. Computing*, 25(1):105–128, 2005.

[10] S. Gottlieb, D. I. Ketcheson, and C.-W. Shu. High order strong stability preserving time discretizations. *Journ. Scientif. Computing*, 38(3):251–289, 2009.

[11] S. Gottlieb and C.-W. Shu. Total variation diminishing Runge-Kutta schemes. *Math. Comp.*, 67(221):73–85, 1998.

[12] S. Gottlieb, C.-W. Shu, and E. Tadmor. Strong stability preserving high order time discretization methods. *SIAM Review*, 43(1):89–112, 2001.

[13] J. P. Heller. An unmixing demonstration. *Am. J. Phys.*, 28:348–353, 1960.

[14] W. Hesthaven and T. Warburton. *Nodal Discontinuous Galerkin methods: Algorithms, analysis, and applicationss*, volume 54 of *Texts in Applied Mathematics*. Springer, New York, 2008.

[15] G.-S. Jiang and C.-W. Shu. Efficient implementation of weighted ENO schemes. *J. Comput. Phys.*, 126(1):202–228, 1996.

[16] M. L. Kontsevich. Lecture at Orsay, December 1995.

[17] R. LeVeque. High-resolution conservative algorithms for advection in incompressible flow. *SIAM J. Numer. Anal.*, 33:627–665, 1996.
[18] X.-D. Liu, S. Osher, and T. Chan. Weighted essentially non-oscillatory schemes. *J. Comput. Phys.*, 115:200–212, 1994.
[19] C.-H. Min and F. Gibou. A second order accurate level set method on non-graded adaptive cartesian grids. *J. Comput. Phys.*, 225(1):300–321, 2007.
[20] J. F. Nash Jr. Arc structure of singularities. *Duke Math. J.*, 81(1):31–38, 1995. (Written in 1966).
[21] J.-C. Nave, R. R. Rosales, and B. Seibold. A gradient-augmented level set method with an optimally local, coherent advection scheme. *J. Comput. Phys.*, 229:3802–3827, 2010.
[22] S. Osher and J. A. Sethian. Fronts propagating with curvature–dependent speed: Algorithms based on Hamilton–Jacobi formulations. *J. Comput. Phys.*, 79:12–49, 1988.
[23] W. H. Reed and T. R. Hill. Triangular mesh methods for the neutron transport equation. Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.
[24] C.-W. Shu. Total-variation diminishing time discretizations. *SIAM J. Sci. Stat. Comp.*, 9(6):1073–1084, 1988.
[25] C.-W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *J. Comput. Phys.*, 77:439–471, 1988.
[26] M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible two-phase flow. *J. Comput. Phys.*, 114(1):146–159, 1994.
[27] G. I. Taylor. Low Reynolds number flow. Movie, 1961. U.S. National Committee for Fluid Mechanics Films (NCFMF).
[28] B. van Leer. Towards the ultimate conservative difference scheme I. The quest of monoticity. *Springer Lecture Notes in Physics*, 18:163–168, 1973.
[29] B. van Leer. Towards the ultimate conservative difference scheme V. A second-order sequel to Godunov's method. *J. Comput. Phys.*, 32:101–136, 1979.

(Benjamin Seibold) Department of Mathematics, Temple University,
1805 North Broad Street, Philadelphia, PA 19122

E-mail address: seibold@temple.edu

URL: http://www.math.temple.edu/~seibold

(Rodolfo R. Rosales) Department of Mathematics, Massachusetts Institute of Technology,
77 Massachusetts Avenue, Cambridge, MA 02139

E-mail address: rrr@math.mit.edu

(Jean-Christophe Nave) Department of Mathematics and Statistics, McGill University,
805 Sherbrooke W., Montreal, QC, H3A 2K6, Canada

E-mail address: jcnave@math.mcgill.ca

URL: http://www.math.mcgill.ca/jcnave