ANALYSIS OF HIGH ORDER FAST INTERFACE TRACKING METHODS

OLOF RUNBORG

Abstract. Fast high order methods for the propagation of an interface in a velocity field are constructed and analyzed. The methods are generalizations of the fast interface tracking method proposed in [O. Runborg, *Commun. Math. Sci.* 7(2):365–398, 2009]. They are based on high order subdivision to make a multiresolution decomposition of the interface. Instead of tracking marker points on the interface the related wavelet vectors are tracked. Like the markers they satisfy ordinary differential equations (ODEs), but fine scale wavelets can be tracked with longer timesteps than coarse scale wavelets. This leads to methods with a computational cost of $O(\log N/\Delta t)$ rather than $O(N/\Delta t)$ for N markers and reference timestep Δt. These methods are proved to still have the same order of accuracy as the underlying direct ODE solver under a stability condition in terms of the order of the subdivision, the order of the ODE solver and the time step ratio between wavelet levels. In particular it is shown that with a suitable high order subdivision scheme any explicit Runge–Kutta method can be used. Numerical examples supporting the theory are also presented.

Keywords: interface tracking, multiresolution analysis, fast algorithms, high order methods

1. Introduction

In this paper we develop and analyze fast, high order methods for tracking a front in a given velocity field. The interface is a manifold of co-dimension at least one which moves according to a time-varying velocity field that does not depend on the front itself, i.e. the velocity of a point on the front depends only on the location of the point and the time. Applications include the tracking of wavefronts in high frequency wave propagation problems [18], iso-distance curves on a surface (front of geodesics) [4], fiber tract bundles in brain imaging [23] or the method of characteristics for the solution graph of hyperbolic partial differential equations (PDEs). We suppose that the front can be parameterized globally, so that for a fixed time t, the interface is described by the function $x(t,s) : \mathbb{R}^+ \times \mathbb{R}^n \to \mathbb{R}^d$, with the parameterization $s \in \Omega \subset \mathbb{R}^n$ and $n \leq d-1$. Then $x(t,s)$ satisfies the parameterized ordinary differential equation (ODE)

$$\frac{\partial x(t,s)}{\partial t} = F(t, x(t,s)), \qquad x(0,s) = \gamma(s), \qquad s \in \Omega, \tag{1.1}$$

where $F(t,x) : \mathbb{R}^+ \times \mathbb{R}^d \to \mathbb{R}^d$ is a given function representing the velocity field and $\gamma(s) : \mathbb{R}^n \to \mathbb{R}^d$ is the initial interface.

Date: February 1, 2014.

Department of Mathematics and Swedish e-Science Research Center (SeRC), KTH, 100 44, Stockholm, email: olofr@kth.se.

Numerical methods for this problem include the Lagrangian front tracking method, [12] which has been used extensively in e.g. multiphase flow [11, 24] and geophysics [25, 18]. The level set method [19] is a Eulerian approach for front tracking, particularly suitable for interfaces with topological changes, such as the merging or pinching off of interface parts. For flow problems we should also mention the marker-and-cell (MAC) [27] and volume of fluid (VOF) [16] methods.

In [22] we constructed a new fast front tracking method for solving (1.1). In standard front tracking, the interface is described by a set of marker points that are connected in a known topology. In one dimension one would approximate $x_j(t) \approx x(t, s_j)$ and use a numerical method for ODEs to solve

$$\frac{dx_j(t)}{dt} = F(t, x_j(t)), \qquad x_j(0) = \gamma(s_j), \tag{1.2}$$

where $s_0 < s_1 < \ldots < s_N$ is a discretization of Ω. For surfaces in three dimensions, the markers on the interface would typically be held together in a triangulation. Propagating one marker numerically with a timestep Δt to a fixed time cost $O(1/\Delta t)$ operations. Hence, if the interface is represented by N points the cost of standard front tracking is $O(N/\Delta t)$. In the new method in [22], wavelet vectors were used to describe the interface, instead of point values. The wavelet vectors correspond to the details of the interface on different scale levels. It was shown that the *time derivatives* of the wavelet vectors, just as the wavelet vectors themselves, decay exponentially with level of detail (cf. Theorem 5.5 below). The method exploits this fact by taking shorter timesteps for the coarse scales than for the fine scales. In this way, the computational cost is reduced to only $O(\log N/\Delta t)$ or even $O(1/\Delta t)$, without affecting the overall order of accuracy. We should stress that this is quite different from standard wavelet based adaptive schemes where shorter timesteps are often used for the *fine* details, i.e. the opposite of this method. With such a strategy the cost will be reduced, but it will just be the constant in the complexity estimate that is improved; the complexity itself remains the same. The reason is that there are comparatively few coarse scale wavelet vectors, where speed gain is achieved, and many fine scale wavelet vectors, where there is little gain. Another interpretation is that the method approximates the solution on a sparse grid [29, 2] in (t, s)-space to achieve efficiency.

In this paper we extend the methods in [22] to high order and give a thorough analysis of the one-dimensional case $n = 1$. It was observed in [22] that high order fast methods could not be achieved with the simple spatial averaging that was used there. A barrier at order two was identified. We show here rigorously that with higher order spatial averaging using general subdivision schemes, high order accuracy can still be combined with low computational cost, as was also discussed in [22]. In particular, it is possible to use high order Runge–Kutta schemes for the time stepping, if they are matched with sufficiently high order subdivision schemes. A more general stability relation is identified: that the factor m with which the time step increases in each level must satisfy $m < 2^{q/p}$ where p and q are the orders of the time and spatial approximations respectively.

In the proofs, new techniques for dealing with the high order subdivision schemes have been introduced. The main new difficulty compared to [22] comes from the fact that the norm of higher order subdivision operators is rarely bounded by one, as it is for the low order schemes in [22]. This makes the stability of the method harder to prove. Section 5.2 contains an all new part of the analysis dedicated to this issue. The more general stability consideration also has repercussions on the nonlinear recursions that must be analyzed in the final proof. For these, the next element in the recursion is bounded by a sum over all previous elements, not just by the previous one as in [22], see Lemma 5.8 and Lemma 6.4. In addition, the Lipschitz type condition (A2) that the numerical scheme must satisfy, is more involved to prove for schemes with multiple stages, like high order Runge–Kutta schemes.

This article is organized as follows. In Section 2 we introduce notation and some basic results for sequences and subdivision operators used later on. Section 3 explains the multiresolution representation of the interface and presents the governing ODEs. The fast numerical methods used to solve those ODEs are subsequently given in Section 4. The main part of the article is Section 5, where the precise analysis of the methods is carried out, and Section 6, where it is shown that the analysis is valid for a variety of Runge–Kutta time stepping schemes. Numerical experiments are presented in Section 7.

2. Sequences and Subdivision Schemes

In this section we give some definitions and basic properties of sequences and subdivision operators that will be used in the paper. Sequences will be written in bold face, and elements of sequences in normal font, $\boldsymbol{x} := \{x_k\}$. We use the usual sup-norm $|\boldsymbol{x}|_\infty = \sup_k |x_k|$ for $\boldsymbol{x} \in \ell_\infty =: L_\infty(\mathbb{Z};\mathbb{R})$. Scalar functions are applied to sequences component wise, so that $f(\boldsymbol{x}) = \{f(x_k)\}$. We use the special sequences $\boldsymbol{k} = \{k\}$, in which the k-th entry is k itself, and $\mathbf{0}$ (resp. $\mathbf{1}$), with all entries equal to 0 (resp. 1). The difference operator Δ is defined as

$$(\Delta \boldsymbol{x})_k = x_{k+1} - x_k. \tag{2.1}$$

Often a sequence itself is indexed by the level of detail j; then we use the convention that $\boldsymbol{x}_j := \{x_{j,k}\}$.

Subdivision is a mechanism for iteratively creating smooth curves and surfaces, [5, 6, 7, 10, 9]. A local, stationary subdivision scheme is characterized by a bounded linear operator $S : \ell_\infty \mapsto \ell_\infty$, defined by a finite sequence $\boldsymbol{a}$ as follows:

$$(S\boldsymbol{x})_k = \sum_\ell a_{k-2\ell} x_\ell.$$

The width B of S is defined by $B = 2\max\{|k|; a_k \neq 0\}$. Thus, the sum is finite: for each k there are only terms with $\ell \in I_k = [\lceil (k-B)/2) \rceil, \lfloor (k+B)/2 \rfloor]$. The max norm of S can be expressed as

$$|S|_\infty = \sup_{|\boldsymbol{x}|_\infty = 1} |S\boldsymbol{x}|_\infty = \max\left(\sum_{k \text{ odd}} |a_k|, \sum_{k \text{ even}} |a_k|\right).$$

To build a smooth function we start from a sequence $\boldsymbol{x}_0$ and associate to it a function $f_0(x)$ which is piecewise linear and interpolates $x_{0,k}$ on the integer grid, $f_0(k) = x_{0,k}$. We can then apply the subdivision scheme S iteratively and define $\boldsymbol{x}_j$ for all $j > 0$ by

$$\boldsymbol{x}_{j+1} = S\boldsymbol{x}_j.$$

Table 1. Masks for the Lagrange subdivision schemes of different orders q.

q	Mask
2	$\left[\ \frac{1}{2}\ ,\ \frac{1}{2}\ \right]$
4	$\left[\ -\frac{1}{16}\ ,\ \frac{9}{16}\ ,\ \frac{9}{16}\ ,\ -\frac{1}{16}\ \right]$
6	$\left[\ \frac{3}{256}\ ,\ -\frac{25}{256}\ ,\ \frac{150}{256}\ ,\ \frac{150}{256}\ ,\ -\frac{25}{256}\ ,\ \frac{3}{256}\ \right]$
8	$\left[\ -\frac{5}{2048}\ ,\ \frac{49}{2048}\ ,\ -\frac{245}{2048}\ ,\ \frac{1225}{2048}\ ,\ \frac{1225}{2048}\ ,\ -\frac{245}{2048}\ ,\ \frac{49}{2048}\ ,\ -\frac{5}{2048}\ \right]$

Sequences at level $j > 0$ similarly represent samples of piecewise linear functions $f_j(x)$ on grids with the increasingly fine spacing 2^{-j}. More precisely, $\boldsymbol{x}_j$ is associated with the grid $\boldsymbol{s}_j$, where $s_{j,k} = k2^{-j}$, and $x_{j,k} = f_j(s_{j,k})$. For a large class of S the process converges, $f_j \to f$ where f is a smooth function.

A subdivision scheme is *interpolating* if $a_{2\ell} = \delta_{\ell,0}$, implying $x_{j+1,2k} = x_{j,k}$ for all j, k; in this case f_{j+1} thus interpolates f_j on the coarser grid: $f_{j+1}(s_{j,k}) = f_j(s_{j,k})$ since $s_{j+1,2k} = s_{j,k}$. Interpolating schemes are described by the *mask* used for computing odd (in k) points, i.e. $[a_{-B}, a_{-B+2}, \ldots, a_B]$

The *order* of an interpolating subdivision scheme S is the largest q such that $SP(\boldsymbol{k}) = P(\boldsymbol{k}/2)$ for all polynomials P of degree $p < q$. We always assume that q is at least one so that $S\mathbf{1} = \mathbf{1}$.

Since sequences often corresponds to smooth functions we also define the divided difference operator $D_j = 2^j\Delta$. The divided differences of a sequence $\boldsymbol{x}_j$ then are

$$\boldsymbol{x}_j^{[r]} = D_j^r \boldsymbol{x}_j \qquad r > 0.$$

For these sequences it frequently useful to consider the *derived* subdivision schemes, defined as

$$S^{[0]} = S, \qquad S^{[r]} = 2\Delta S^{[r-1]}\Delta^{-1}, \qquad r > 0.$$

This is implies that if $\boldsymbol{x}_{j+1} = S\boldsymbol{x}_j$ then $\boldsymbol{x}_{j+1}^{[r]} = S^{[r]}\boldsymbol{x}_j^{[r]}$. Note that $S^{[r]}$ is only well-defined for $r \le q$.

A special example of a subdivision scheme is the midpoint interpolating scheme S_2 where $(S_2\boldsymbol{x})_{2k+1} = (x_k + x_{k+1})/2$. Hence the mask is $[1/2, 1/2]$. This scheme has order $q = 2$ and yields piecewise linear limit functions. Examples of masks for higher order interpolating subdivision schemes are given in Table 1. These *Lagrange* subdivision schemes are used in the numerical examples in Section 7.

The generalization to higher dimensions is straightforward with sequences of several components $\boldsymbol{x} = (\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^d) \in \ell_\infty^d =: L_\infty(\mathbb{Z}; \mathbb{R}^d)$, i.e. $\boldsymbol{x}^j \in \ell_\infty$. We can equivalently view this as each sequence element belonging to $\mathbb{R}^d$, so that $\boldsymbol{x} = \{x_k\}$ with $x_k = (x_k^1, x_k^2, \ldots, x_k^d)^T \in \mathbb{R}^d$. For scalar functions $f : \mathbb{R}^d \mapsto \mathbb{R}$ applied to such sequences we write $f(\boldsymbol{x}) = \{f(x_k^1, x_k^2, \ldots, x_k^d)\} = \{f(x_k)\}$ and for vectorial functions $F = (f_1, \ldots, f_r)^T : \mathbb{R}^d \mapsto \mathbb{R}^r$ we write $F(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_r(\boldsymbol{x}))^T$. Operators are always applied elementwise to the sequences, e.g. $\Delta\boldsymbol{x} = (\Delta\boldsymbol{x}^1, \ldots, \Delta\boldsymbol{x}^d)$ and $S\boldsymbol{x} = (S\boldsymbol{x}^1, \ldots, S\boldsymbol{x}^d)$. The sup norm is defined as

$$|\boldsymbol{x}|_\infty = \max_j \left|\boldsymbol{x}^j\right|_\infty = \sup_k |x_k|_\infty .$$

Sequences with several components are needed to approximate curves in $\mathbb{R}^d$.

We end this section with some basic results on subdivision which will be used later on. First we have a local error estimate for a subdivision scheme.

Proposition 1 (Proposition 3.1 in [8]). For a local subdivision scheme S of order $q \geq 1$ and $\boldsymbol{x} \in \ell_\infty$ we have the estimate

$$\max_{\ell \in I_k} |x_\ell - (S\boldsymbol{x})_k| \leq C \max_{\ell \in I_k} |(\Delta \boldsymbol{x})_\ell| \leq C \left|\Delta \boldsymbol{x}\right|_\infty = C 2^{-j} \left|D_j \boldsymbol{x}\right|_\infty . \tag{2.2}$$

Furthermore, we have two results showing different versions of an approximate commutation property between the subdivision operator and application of a function. Let the space $C^k(\Omega_1; \Omega_2)$ denote all measurable functions from Ω_1 to Ω_2 with continuous derivatives upto order k. We then define $C_b^k(\Omega_1; \Omega_2)$ as the functions $f \in C^k(\Omega_1; \Omega_2)$ whose derivatives upto order k are all bounded.

Proposition 2. Suppose $F \in C_b^1(\mathbb{R}^d; \mathbb{R}^r)$. For a local subdivision scheme S of order $q \geq 1$, $\boldsymbol{x} \in \ell_\infty^d$ and $\boldsymbol{y} \in \ell_\infty^r$, we have the estimate

$$\left|F(S\boldsymbol{x})^T S\boldsymbol{y} - SF(\boldsymbol{x})^T \boldsymbol{y}\right|_\infty \leq C \left|DF\right|_\infty \left|\Delta \boldsymbol{x}\right|_\infty \left|\boldsymbol{y}\right|_\infty . \tag{2.3}$$

Proof. Let $\boldsymbol{z} = F(S\boldsymbol{x})^T S\boldsymbol{y} - SF(\boldsymbol{x})^T \boldsymbol{y}$. Then, since

$$z_k = \sum_{\ell \in I_k} a_{k-2\ell} \left[F\left((S\boldsymbol{x})_k\right) - F(x_\ell)\right]^T y_\ell,$$

we can bound

$$|z_k|_\infty \leq |DF|_\infty \max_{\ell \in I_k} |x_\ell - (S\boldsymbol{x})_k|_\infty \sum_{\ell \in I_k} |a_{k-2\ell}| \, |y_\ell|_\infty ,$$

and the result follows from applying Proposition 1 componentwise to $x_\ell - (S\boldsymbol{x})_k \in \mathbb{R}^d$. □

Theorem 2.1 (Theorem 3.4 in [8])**.** *Let S be a subdivision scheme of order q and let $\{\boldsymbol{x}_j\}$, with $\boldsymbol{x}_j \in \ell_\infty$, be generated by S. Suppose $x_{j,k} \in \Omega$ for all j, k and that $f \in C_b^{r+1}(\Omega)$ with $r \in \mathbb{Z}$ and $1 \leq r < q$; suppose also that $\left|\boldsymbol{x}_j^{[\ell]}\right|_\infty \leq C$ for $1 \leq \ell \leq r$ and for $j > 0$. Then*

$$|f(S\boldsymbol{x}_j) - Sf(\boldsymbol{x}_j)|_\infty \leq C \sup_{0 \leq \ell \leq k+1} \left|f^{(\ell)}\right|_\infty 2^{-j(r+1)},$$

where C is independent of j and f.

This is a slightly modified version of the theorem in [8]. Here we have made the dependence on f of the constant more precise. The result follows directly from the corresponding proof in [8].

Remark 2.2. In the description above the subdivision operators act on infinite sequences. In numerical computations one must restrict them to finite length sequences, which then corresponds to samples of functions on bounded intervals or periodic functions. For the former case, the subdivision operators can be modified near the boundaries, such that a different sequence $\boldsymbol{a}$ is used in (2) for those points. For interpolatory schemes, a modification is necessary when the width B exceeds two. By using an appropriate $\boldsymbol{a}$, high order schemes can then still be constructed. However, the formulation of Proposition 1, Proposition 2 and Theorem 2.1, as well as the definition of derived schemes must be changed. To simplify we will therefore just consider periodic (closed) interfaces, corresponding to periodic sequences, in the main analysis in Section 5.

3. Multiresolution description of the interface propagation

We consider the case of a one-dimensional interface ($n = 1$) in d-dimensional space, where $s \in \Omega = [0,1]$ or $\Omega = \mathbb{T}$, the one-periodic torus, and $x(t,s) \in \mathbb{R}^d$. In standard front tracking algorithms marker points are used to represent the interface. We will instead consider a multiresolution representation, which is often a more efficient way to describe curves and surfaces. In our case the curve $x(t,s)$ will be described as follows. We introduce the parameter indices $s_{j,k} = k2^{-j}$ and define

$$x_{j,k}(t) := x(t, s_{j,k}), \qquad 0 \le k \le 2^j.$$

Note that $x_{j+1,2k} = x_{j,k}$ and that for a fixed j the markers $\{x_{j,k}\}$ will be a discretization of the interface with a level of detail that increases with the fixed j-value, cf. Figure 1 (a) and (b). Also note that when the interface is closed, $x_{j,0} = x_{j,2^j}$. We assume that we start from a given fine discretization with $N = 2^J$ points on the interface, which thus corresponds to level J. We let $\boldsymbol{x}_j(t) = \{x_{j,k}(t)\}_{k=0}^{2^j}$ and next define the wavelet vectors

$$\boldsymbol{w}_{j+1}(t) = \boldsymbol{x}_{j+1}(t) - S\boldsymbol{x}_j(t). \tag{3.1}$$

This is done recursively and gives an alternative description of the interface in terms of $\boldsymbol{x}_0(t)$ together with $\boldsymbol{w}_j(t)$ for $j = 1, \dots, J$. The wavelet sequences $\boldsymbol{w}_j(t) =: \{w_{j,k}(t)\}_{k=0}^{2^j}$ can be computed from the original discretization $\boldsymbol{x}_J(t)$ at a $O(N)$ cost, where $N = 2^J$ is the number of discretization points. Similarly, with an inverse wavelet transform based on reversing the recursion in (3.1), the points $\boldsymbol{x}_J(t)$ can be computed from $\{\boldsymbol{w}_j\}$ and $\boldsymbol{x}_0$ at a $O(N)$ cost. The process is visualized in Figure 1. We can now insert (3.1) in (1.1) and get

$$\frac{d\boldsymbol{w}_{j+1}}{dt} = F(t, \boldsymbol{x}_{j+1}(t)) - SF(t, \boldsymbol{x}_j(t)) = F(t, S\boldsymbol{x}_j(t) + \boldsymbol{w}_{j+1}(t)) - SF(t, \boldsymbol{x}_j(t)).$$

Setting

$$G(t, \boldsymbol{x}, \boldsymbol{w}) = F(t, S\boldsymbol{x} + \boldsymbol{w}) - SF(t, \boldsymbol{x}),$$

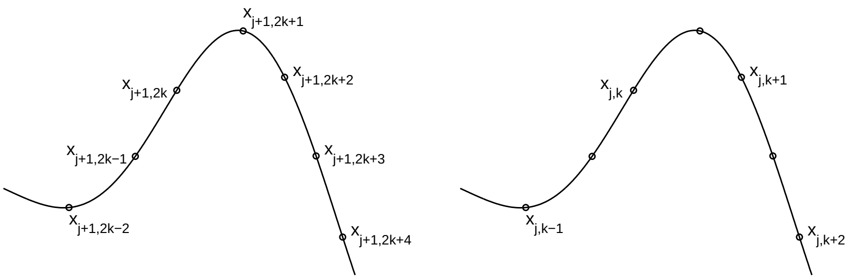
we thus have the following alternative system of ODEs

$$\frac{d\boldsymbol{w}_{j+1}(t)}{dt} = G(t, \boldsymbol{x}_j(t), \boldsymbol{w}_{j+1}(t)), \qquad \frac{d\boldsymbol{x}_0(t)}{dt} = F(t, \boldsymbol{x}_0(t)), \tag{3.2}$$

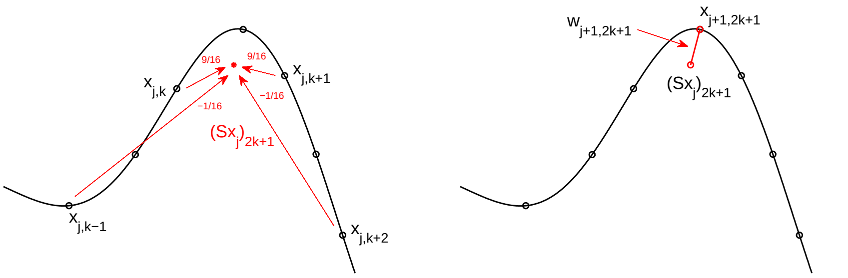
which together with (3.1) describe the dynamics of the system. Note that, when $\Omega = [0,1]$ we assume here, and in what follows, that S has been appropriately modified near boundaries.

The important property of the wavelet representation is the fast decay in j of the wavelet vectors $\{\boldsymbol{w}_j\}$ *and their time derivatives.* It can be shown that for smooth $x(t,s)$ they decay exponentially in j with a rate determined by the order of the subdivision scheme S (see Theorem 5.5). This gives the representation good compression properties and, as was shown in [22], it allows us to construct a fast interface tracking algorithm. This is discussed in the next section.

Remark 3.1. In [22] a particular version of the above algorithm was used for the initial data, where each wavelet vector $w_{j+1,2k+1}$ was chosen to be exactly normal to the line between its neighboring points on the previous level $x_{j,k+1} - x_{j,k}$. This gives a so-called *normal mesh*, [13, 8, 14, 21], which is a somewhat better starting



(a) Interface and elements of sequence $\boldsymbol{x}_{j+1}$. (b) Interface and elements of sequence $\boldsymbol{x}_j$.



(c) Construction of $S\boldsymbol{x}_j$ from $\boldsymbol{x}_j$. (d) Wavelet vector $w_{j+1,2k+1}$.

FIGURE 1. Relationship between the markers $\boldsymbol{x}_{j+1} = \{x_{j+1,k}\}$, $\boldsymbol{x}_j = \{x_{j,k}\}$, the subdivision sequence $S\boldsymbol{x}_j$ and the wavelet vectors in $\boldsymbol{w}_{j+1}$. The weights indicated next to the arrows in c) correspond to the Lagrange four-point scheme, for which $(S\boldsymbol{x}_j)_{2k+1} = (-x_{j,k-1} + 9x_{j,k} + 9x_{j,k+1} - x_{j,k+2})/16$.

position but is not necessary for the convergence of the algorithm. On the other hand, it makes the proofs more difficult since the parameterization induced by this choice of $w_{j+1,2k+1}$ has limited smoothness.

4. Numerical Methods

We construct high order methods based on the methods developed in [22]. The main idea is to solve (3.2) instead of (1.2), starting from a multiresolution representation of $\gamma(s)$. Since the time derivatives of $\boldsymbol{w}_j$ decay rapidly with j by Theorem 5.5 we can take longer time steps for larger j and thereby get a more efficient method. More precisely, we use the time step

$$\Delta t_j := m^j \Delta t,$$

on level j, where the integer $\Delta t_{j+1}/\Delta t_j = m \geq 2$ is the time step ratio, and Δt is a reference time step. A larger m means a smaller computational cost, but also a less accurate solution. The numerical approximations are denoted

$$\boldsymbol{x}_j^n \approx \boldsymbol{x}_j(t_{j,n}), \qquad \boldsymbol{w}_j^n \approx \boldsymbol{w}_j(t_{j,n}), \qquad t_{j,n} = n\Delta t_j,$$

where $\boldsymbol{w}_j^0 = \{w_{j,k}(0)\}$ and $\boldsymbol{x}_0^0 = \{x_{0,k}(0)\}$ is a multiresolution decomposition of the initial curve $x(0,s) = \gamma(s)$. To solve (3.1) and (3.2) we first solve the zeroth level

$\boldsymbol{x}_0$ directly with a high order explicit one-step ODE method ϕ_F with the reference time step $\Delta t = \Delta t_0$,

$$\boldsymbol{x}_0^{n+1} = \boldsymbol{x}_0^n + \Delta t \phi_F(\boldsymbol{x}_0^n, \Delta t).$$

For higher levels we use an explicit one-step (but in general multi-stage) method ϕ_G, of the following form

$$\boldsymbol{w}_j^{n+1} = \boldsymbol{w}_j^n + \Delta t_j \phi_G(t_{j,n+\eta_1}, \boldsymbol{x}_{j-1}^{(n+\eta_1)m}, \cdots, t_{j,n+\eta_s}, \boldsymbol{x}_{j-1}^{(n+\eta_s)m}, \boldsymbol{w}_j^n, \Delta t_j), \tag{4.1}$$

where $0 \le \eta_k \le 1$ and $\eta_k m \in \mathbb{Z}$ for all k. To pass from level j to $j+1$ we use the definition of $\boldsymbol{w}_j$,

$$\boldsymbol{x}_j^n = S\boldsymbol{x}_{j-1}^{nm} + \boldsymbol{w}_j^n. \tag{4.2}$$

We note that

$$\boldsymbol{x}_{j-1}^{(n+\eta_s)m} \approx \boldsymbol{x}_{j-1}(t_{j-1,(n+\eta_s)m}) = \boldsymbol{x}_{j-1}((n+\eta_s)\Delta t_j) = \boldsymbol{x}_{j-1}(t_{j,n+\eta_s}).$$

The form of ϕ_G is chosen such that it will include the explicit Runge–Kutta methods; ϕ_G could for instance be the classical fourth order Runge–Kutta method:

$$\begin{aligned}
\xi_1 &= G\left(t_{j,n}, \boldsymbol{x}_{j-1}^{nm}, \boldsymbol{w}_j^n\right), \\
\xi_2 &= G\left(t_{j,n+1/2}, \boldsymbol{x}_{j-1}^{(n+1/2)m}, \boldsymbol{w}_j^n + \frac{\Delta t_j}{2}\xi_1\right) \\
\xi_3 &= G\left(t_{j,n+1/2}, \boldsymbol{x}_{j-1}^{(n+1/2)m}, \boldsymbol{w}_j^n + \frac{\Delta t_j}{2}\xi_2\right) \\
\xi_4 &= G\left(t_{j,n+1}, \boldsymbol{x}_{j-1}^{(n+1)m}, \boldsymbol{w}_j^n + \Delta t_j \xi_3\right) \\
\boldsymbol{w}_j^{n+1} &= \boldsymbol{w}_j^n + \frac{\Delta t_j}{6}(\xi_1 + 2\xi_2 + 2\xi_3 + \xi_4).
\end{aligned}$$

In this case $s = 4$ and $\eta_1 = 0$, $\eta_2 = \eta_3 = 1/2$ and $\eta_4 = 1$. In Section 7 a number of different Runge–Kutta methods, listed in Table 3, are used in the numerical examples.

Since (3.1) and (3.2) can be solved level by level, starting from $j = 0$ we can write down an expression for the computational cost as follows. We assume that S can be applied fast so that the cost of computing $S\boldsymbol{x}$ is proportional to the number of elements of $\boldsymbol{x}$. Then computing $G(t, \boldsymbol{x}, \boldsymbol{w})$ also has a cost proportional to the number of elements of $\boldsymbol{x}$ (and $\boldsymbol{w}$). This means that the cost of applying ϕ_G in (4.1) and updating $\boldsymbol{x}_j$ in (4.2) is proportional to the number of elements of $\boldsymbol{w}_j$, i.e. to the number of points at level j, which is 2^j. To propagate the solution at level j to time T we need to take $T/\Delta t_j$ time steps, hence executing (4.1) and (4.2) each $T/\Delta t_j$ times. Suppose the interface is described by $N = 2^J$ points. Then the total cost for all levels would be proportional to

$$\sum_{j=0}^{J} \frac{T}{\Delta t_j} 2^j = \frac{T}{\Delta t} \sum_{j=0}^{J} (2/m)^j \le C \frac{T}{\Delta t} \begin{cases} \log_2 N, & m = 2, \\ 1, & m > 2. \end{cases}$$

Hence, for $m = 2$ we have the complexity $O(\log_2 N/\Delta t)$ while for $m > 2$ it is $O(1/\Delta t)$. This is of course significantly faster than the $O(N/\Delta t)$ complexity for the standard method. That the fast method is still accurate will be proved in the next section. Note also here that in addition to the complexity of propagating the wavelet vectors we normally also want to reconstruct the pointwise values of the interface which is an additional $O(N)$ cost.

Remark 4.1. The assumption that S can be applied fast is obviously true for local subdivision operator which has a finite size mask, like e.g. the Lagrange schemes used in Section 7. However, we note that this can also be true for other S, such as Fourier interpolation computed via FFT (at least upto a log factor). This is also tested in Section 7.

5. Analysis of the Methods

In this section we analyze the errors in the methods proposed in Section 4 for closed interfaces, where the time-evolution of $\{w_{j,k}\}$ and the node points $\{x_{j,k}\}$ are approximated by a time stepping method ϕ_G and the spatial approximation is done with a subdivision operator S. We then make the following precise assumptions about the method for the given G:

(A1) p-th order accuracy in time.
Given an exact solution $\boldsymbol{w}_j$, the local truncation error $\boldsymbol{\tau}_j^n$ of ϕ_G is defined by the relation

$$\boldsymbol{w}_j(t_{j,n+1}) = \boldsymbol{w}_j(t_{j,n}) + \Delta t_j \phi_G\Big(t_{j,n+\eta_1}, \boldsymbol{x}_{j-1}(t_{j,n+\eta_1}), \cdots, t_{j,n+\eta_s}, \boldsymbol{x}_{j-1}(t_{j,n+\eta_s}), \boldsymbol{w}_j(t_n), \Delta t_j\Big) + \boldsymbol{\tau}_j^n.$$

When $\boldsymbol{w}_j \in C^{p+1}([0,T];\mathbb{R}^d)$ it satisfies

$$\max_{0\le t_{j,n}\le T} \left|\boldsymbol{\tau}_j^n\right|_\infty \le C\Delta t_j^{p+1} \max_{\substack{0\le t\le T\\ 1\le \ell\le p+1}} \left|\frac{d^\ell \boldsymbol{w}_j(t)}{dt^\ell}\right|_\infty,$$

where the constant C is independent of j, Δt_j and n.

(A2) The time stepping methods satisfies the following Lipschitz type bound

$$\begin{aligned}&|\phi_G(t_1, \boldsymbol{x}_1, \cdots, t_s, \boldsymbol{x}_s, \boldsymbol{w}, \Delta t) - \phi_G(t_1, \tilde{\boldsymbol{x}}_1, \cdots, t_s, \tilde{\boldsymbol{x}}_s, \tilde{\boldsymbol{w}}, \Delta t)|_\infty \\ &\quad \le C\left(|\boldsymbol{w}|_\infty + \max_{1\le j\le s} |\Delta \boldsymbol{x}_j|_\infty\right) \max_{1\le j\le s} |\boldsymbol{x}_j - \tilde{\boldsymbol{x}}_j|_\infty \\ &\quad\quad + C\left(|\boldsymbol{w} - \tilde{\boldsymbol{w}}|_\infty + \max_{1\le j\le s} |\boldsymbol{x}_j - \tilde{\boldsymbol{x}}_j|_\infty^2\right),\end{aligned}$$

when $\Delta t \le 1$.

(A3) q-th order accuracy in space.
The subdivision scheme S is of order $q \ge 1$. It is local, stationary and interpolating.

(A4) Regularity of S.
There is an integer r such that $1 \le r \le q$ and

$$\left|S^{[r]}\right|_\infty < 2^r.$$

We can then prove the following theorem which shows that despite the low complexity derived in Section 4 the method is accurate, thus justifying the designation "fast."

Theorem 5.1. *Given a smooth initial closed curve $\gamma \in C^M(\mathbb{T};\mathbb{R}^d)$, a finite final time $T < \infty$, a smooth and bounded velocity field $F \in C_b^M([0,T]\times\mathbb{R}^d;\mathbb{R}^d)$ and a time step ratio m. The exact solution $x(t,s)$ exists and belongs to $C^M([0,T]\times\mathbb{T};\mathbb{R}^d)$. Suppose further that the assumptions (A1)–(A4) hold for the numerical method, with $p+q+1 \le M$. Let the reference time step Δt and the positive integer J be*

Table 2. Max norms of the Lagrange subdivision schemes, S and the derived schemes $S^{[1]}, \ldots, S^{[4]}$.

q	$\lvert S\rvert_\infty$	$\lvert S^{[1]}\rvert_\infty$	$\lvert S^{[2]}\rvert_\infty$	$\lvert S^{[3]}\rvert_\infty$	$\lvert S^{[4]}\rvert_\infty$
2	1.0000	1.0000	2.0000	n.a.	n.a.
4	1.2500	1.2500	2.0000	2.0000	4.0000
6	1.3906	1.3906	2.0000	2.0000	4.0000
8	1.4883	1.4883	2.0000	2.0000	4.0000

chosen such that the total number of points $N = 2^J$ and the final time $T = \Delta t_J = m^J \Delta t$. If also the time step ratio satisfies the stability condition

$$m < 2^{q/p}, \tag{5.1}$$

then

$$\left|\boldsymbol{x}_J^1 - \boldsymbol{x}_J(T)\right|_\infty \le C\Delta t^p. \tag{5.2}$$

where C is independent of J and Δt.

To prove this theorem we will first derive estimates of the exact solution in Theorem 5.5. We next show Theorem 5.7, which proves that (A3) and (A4) imply stability of the subdivision operator S. With these results in hand we can subsequently prove Theorem 5.1 in Section 5.3, where we also use a lemma on growth in a nonlinear recursion proved in Appendix A.

Remark 5.2. We will show later in Section 6 that standard explicit Runge–Kutta methods satisfy assumptions (A1) and (A2). Moreover, there are many subdivision schemes satisfying (A3) and (A4), for instance the Lagrange schemes; see Table 2.

Remark 5.3. In [22] a "barrier" was observed which prevented a fast method for time-stepping methods of order higher than two. The reason was that the method was based on the two-point subdivision scheme, with $q = 2$. From (5.1) we then have the condition $m < 2^{2/p}$, which says that m cannot be larger than one if $p > 2$. By introducing high order subdivision schemes, with larger q, the barrier is removed.

Remark 5.4. In [22] we showed that for the stability borderline case $m = 2^{q/p}$ one obtains an additional $\log N$ factor multiplying the error estimate when $m = p = q = 2$. We conjecture that the same thing will happen in the general case.

5.1. **Estimates of the Exact Solution.** Here we show that if the solution $x(t,s)$ is smooth then the wavelets and their time derivatives decay rapidly with the level j.

Theorem 5.5. *Suppose $x(t,s) \in C^{q+\ell}([0,T]\times\mathbb{T};\mathbb{R}^d)$ and assumption (A3) holds for S. Then*

$$\left|\frac{d^\ell \boldsymbol{w}_j(t)}{dt^\ell}\right|_\infty \le C_\ell(T)2^{-jq}, \quad |\Delta \boldsymbol{x}_j(t)|_\infty \le C(T)2^{-j}, \qquad 0 \le t \le T.$$

Proof. Suppose $\boldsymbol{x}_j = \{x_{j,k}\}$ is defined by $x_{j,k} = V(s_{j,k})$ where $s_{j,k} = k2^{-j}$ and $V = (v_1,\ldots,v_d)^T$ is a function belonging to $C^q(\mathbb{T};\mathbb{R}^d)$. Let $\boldsymbol{s}_j = \{s_{j,k}\}$. Since S is at least of order one, it reproduces linear functions. Hence, $\boldsymbol{s}_{j+1} = S\boldsymbol{s}_j$. Then the corresponding wavelet coefficients $\boldsymbol{w}_j = \{w_{j,k}\}$ are given by

$$\boldsymbol{w}_{j+1} = \boldsymbol{x}_{j+1} - S\boldsymbol{x}_j = V(\boldsymbol{s}_{j+1}) - SV(\boldsymbol{s}_j) = V(S\boldsymbol{s}_j) - SV(\boldsymbol{s}_j),$$

We can therefore apply Theorem 2.1 componentwise to V and obtain

$$|\boldsymbol{w}_{j+1}|_\infty \le C \max_{\substack{0\le r\le q\\ 1\le m\le d}} \left|v_m^{(r)}\right|_\infty 2^{-jq} = C \max_{0\le r\le q} \left|V^{(r)}\right|_\infty 2^{-jq},$$

since

$$\boldsymbol{s}_j^{[r]} = \begin{cases} 1, & r = 1, \\ 0, & r \ge 2. \end{cases}$$

Letting

$$v_{j,k}(t) = \partial_t^\ell x(t, s_{j,k})$$

we then have

$$\left|\frac{d^\ell \boldsymbol{w}_j}{dt^\ell}\right|_\infty = |\boldsymbol{v}_{j+1} - S\boldsymbol{v}_j|_\infty \le C \max_{0\le r\le q} |\partial_s^r v(t,\cdot)|_\infty 2^{-qj} = C \max_{0\le r\le q} \left|\partial_t^\ell \partial_s^r x(t,\cdot)\right|_\infty 2^{-qj}$$
$$\le C_\ell(T) 2^{-qj},$$

since $x \in C^{q+\ell}([0,T]\times\mathbb{T};\mathbb{R}^d)$. Finally,

$$|\Delta\boldsymbol{x}_j|_\infty = \sup_k |x(t, s_{j,k+1}) - x(t, s_{j,k})| \le |\partial_s x(t,\cdot)|_\infty 2^{-j}.$$

This proves the theorem. □

Remark 5.6. By making the assumption that $x(t,s)$ is smooth the proof of this theorem is significantly simplified compared to the corresponding theorem in [22]. There the initial curve γ had a normal parameterization, which was only Lipschitz continuous. Here we do not insist on a normal representation of γ. Therefore $x(t,s)$ will be smooth whenever the velocity field F is smooth and γ is smoothly parameterized.

5.2. **Subdivision Stability.** In this section we show that the subdivision operator S used for the spatial approximation is stable, that is, a small perturbation in the subdivision construction process gives only a small change in the final result. This will be necessary to be able to prove Theorem 5.1. Consider a sequence $\boldsymbol{x}_j$ that is approximately built using S, in the sense that in each step we have a bound

$$|\boldsymbol{x}_{j+1} - S\boldsymbol{x}_j|_\infty \le d_j, \qquad j \ge 0, \tag{5.3}$$

with d_j being a small perturbation. If $|S|_\infty \le 1$ we can easily estimate the size of $|\boldsymbol{x}_j|_\infty$ from

$$|\boldsymbol{x}_{j+1}|_\infty \le |\boldsymbol{x}_j|_\infty + d_j \le |\boldsymbol{x}_0|_\infty + \sum_{k=0}^{j} d_k,$$

and one can see that the construction is stable as long as the sum of the perturbations d_j is bounded. This was the strategy used in [22]. In the present paper, however, we cannot use the simple analysis above, because for almost all interesting high order subdivision schemes $|S|_\infty > 1$, for instance the Lagrange schemes; see Table 2. What saves us are corresponding bounds on the derived operators of the type in (A4), namely $\left|S^{[r]}\right|_\infty < 2^r$ for some $r \in \mathbb{Z}$ and $r \le q$, where q is the order of S. This holds for most reasonable schemes, including the Lagrange schemes, as seen in Table 2. From such a bound we can prove Theorem 5.7 below which is the same kind of stability result for high order subdivision scheme as Theorem 4.2 in [8]; the proof also uses the same technique. We should remark that in [8] and in many other papers, e.g. [20, 28, 26], stability results of this type are used to show smoothness of weakly nonlinear subdivision schemes. In that setting the nonlinear scheme is seen as a perturbation of a linear scheme "in proximity," and d_k is then the difference between the two when applied to $\boldsymbol{x}_j$.

Theorem 5.7. *Let* $\{\boldsymbol{x}_j\}$ *be approximately generated by the subdivision scheme* S *such that*

$$|\boldsymbol{x}_{j+1} - S\boldsymbol{x}_j|_\infty \le d_j, \qquad j \ge 0. \tag{5.4}$$

Suppose assumptions (A3) and (A4) hold for S*. Then there is a constant* C *independent of* j*,* d_j *and* $\boldsymbol{x}_0 \in \ell_\infty$ *such that*

$$|\boldsymbol{x}_j|_\infty \le C\left(|\boldsymbol{x}_0|_\infty + \sum_{k=0}^{j-1} d_k\right), \tag{5.5}$$

for $j > 0$*.*

Proof. By (A3) and (A4) we can choose $0 < \varepsilon < 1$ and a positive integer $r \le q$ so that

$$\left|S^{[r]}\right|_\infty \le 2^{r-\varepsilon}.$$

We claim that the following result holds. Let $\ell \in \mathbb{Z}$ satisfy $1 \le \ell \le r$. There is a constant C_ℓ, independent of $j \ge 0$, d_j and $\boldsymbol{x}_0^{[\ell]}$, such that

$$\left|\boldsymbol{x}_j^{[\ell]}\right|_\infty \le C_\ell 2^{j(\ell-\varepsilon)}\left(\left|\boldsymbol{x}_0^{[\ell]}\right|_\infty + v_{j-1}(\varepsilon)\right), \tag{5.6}$$

where $v_{-1}(\alpha) = 0$ and

$$v_j(\alpha) := \sum_{k=0}^{j} 2^{k\alpha} d_k, \qquad j \ge 0. \tag{5.7}$$

We will prove this by induction and start by proving that it is true for $\ell = r$. Let us define the residual sequences

$$\boldsymbol{r}_{j+1} := \boldsymbol{x}_{j+1} - S\boldsymbol{x}_j, \qquad j \ge 0;$$

observe that

$$\boldsymbol{r}_{j+1}^{[r]} = \boldsymbol{x}_{j+1}^{[r]} - S^{[r]}\boldsymbol{x}_j^{[r]}. \tag{5.8}$$

Thus, with $\mu = r - \varepsilon$,

$$\left|\boldsymbol{x}_{j+1}^{[r]}\right|_\infty \le \left|S^{[r]}\right|_\infty \left|\boldsymbol{x}_j^{[r]}\right|_\infty + \left|\boldsymbol{r}_{j+1}^{[r]}\right|_\infty \le 2^\mu \left|\boldsymbol{x}_j^{[r]}\right|_\infty + \left|\boldsymbol{r}_{j+1}^{[r]}\right|_\infty.$$

Applying this repeatedly we get

$$\left|\boldsymbol{x}_{j+1}^{[r]}\right|_\infty \le \left|\boldsymbol{r}_{j+1}^{[r]}\right|_\infty + \sum_{k=0}^{j-1} \left|\boldsymbol{r}_{k+1}^{[r]}\right|_\infty 2^{\mu(j-k)} + \left|\boldsymbol{x}_0^{[r]}\right|_\infty 2^{\mu(j+1)}. \tag{5.9}$$

Moreover, by definition

$$\left|\boldsymbol{r}_j^{[r]}\right|_\infty = 2^j \left|\Delta \boldsymbol{r}_j^{[r-1]}\right|_\infty \le 2^{j+1}\left|\boldsymbol{r}_j^{[r-1]}\right|_\infty.$$

By induction on this relationship and (5.4), we get

$$\left|\boldsymbol{r}_j^{[r]}\right|_\infty \le 2^{(j+1)r}|\boldsymbol{r}_j|_\infty \le c2^{(j-1)r}d_{j-1}, \qquad \forall j \ge 1, \tag{5.10}$$

with $c = 4^r$. Together (5.10) and (5.9) give us for $j \geq 0$,

$$\begin{aligned}\left|\boldsymbol{x}_{j+1}^{[r]}\right|_\infty &\leq c\left(2^{jr}d_j + \sum_{k=0}^{j-1} 2^{kr}2^{\mu(j-k)}d_k\right) + \left|\boldsymbol{x}_0^{[r]}\right|_\infty 2^{\mu(j+1)} \\ &= c2^{\mu j}v_j(r-\mu) + \left|\boldsymbol{x}_0^{[r]}\right|_\infty 2^{\mu(j+1)}.\end{aligned}$$

This agrees with (5.6) when $\ell = r \geq 1$ and $j \geq 0$ upon taking $C_r = \max(c, 2^\mu) = c$. Suppose now that (5.6) holds for some $2 \leq \ell \leq r$. Induction will then yield the result if we can prove that this implies (5.6) is true also for $\ell - 1 \geq 1$. To show this, we first fix an index $k =: k_{j+1}$, and construct a sequence of indices $\{k_s\}_{s=0}^j$ such that $k_s \in I_{k_{s+1}}$. Then

$$x_{j+1,k}^{[\ell-1]} = x_{0,k_0}^{[\ell-1]} + \sum_{s=0}^{j}\left(x_{s+1,k_{s+1}}^{[\ell-1]} - x_{s,k_s}^{[\ell-1]}\right)$$

and we can estimate

$$\left|\boldsymbol{x}_{j+1}^{[\ell-1]}\right|_\infty \leq \left|\boldsymbol{x}_0^{[\ell-1]}\right|_\infty + \sum_{s=0}^{j}\sup_k \max_{i\in I_k}\left|x_{s+1,k}^{[\ell-1]} - x_{s,i}^{[\ell-1]}\right|.$$

Moreover, since $\ell - 1 < q$ the order of $S^{[\ell-1]}$ is at least one, and we can use (2.2) in Proposition 1. Together with (5.10) and the hypothesis that (5.6) is true for q, we then get, with C' being the constant in Proposition 1,

$$\begin{aligned}&\sum_{s=0}^{j}\sup_k \max_{i\in I_k}\left|x_{s+1,k}^{[\ell-1]} - x_{s,i}^{[\ell-1]}\right| \\ &\quad\leq \sum_{s=0}^{j}\sup_k \max_{i\in I_k}\left|\left(S^{[\ell-1]}\boldsymbol{x}_s^{[\ell-1]}\right)_k - x_{s,i}^{[\ell-1]}\right| + \sum_{s=0}^{j}\left|\boldsymbol{r}_{s+1}^{[\ell-1]}\right|_\infty \\ &\quad\leq C'\sum_{s=0}^{j}2^{-s}\left|\boldsymbol{x}_s^{[\ell]}\right|_\infty + c\sum_{s=0}^{j}2^{s(\ell-1)}d_s \\ &\quad\leq C_\ell C'\sum_{s=0}^{j}2^{s(\ell-1-\varepsilon)}\left(\left|\boldsymbol{x}_0^{[\ell]}\right|_\infty + v_{s-1}(\varepsilon)\right) + c\sum_{s=0}^{j}2^{s(\ell-1-\varepsilon)}2^{s\varepsilon}d_s \\ &\quad\leq C_\ell C'\left(\left|\boldsymbol{x}_0^{[\ell]}\right|_\infty + v_j(\varepsilon)\right)\sum_{s=0}^{j}2^{s(\ell-1-\varepsilon)} + c2^{j(\ell-1-\varepsilon)}v_j(\varepsilon) \\ &\quad\leq C_{\ell-1}2^{j(\ell-1-\varepsilon)}\left(\left|\boldsymbol{x}_0^{[\ell]}\right|_\infty + v_j(\varepsilon)\right),\end{aligned}$$

with

$$C_{\ell-1} = \max\left(C_\ell C'\sum_{s=0}^{\infty}2^{-s(\ell-1-\varepsilon)}, c\right) = \max\left(\frac{C_\ell C'}{1-2^{-(\ell-1-\varepsilon)}}, c\right),$$

which is finite since $\ell - 1 - \varepsilon > 0$. This shows the claim (5.6). It remains to show the final estimate for $\ell = 0$. In the same way as above we obtain

$$\begin{aligned} |\boldsymbol{x}_{j+1}|_\infty &\le |\boldsymbol{x}_0|_\infty + \sum_{s=0}^{j} \sup_k \max_{i\in I_k} |x_{s+1,k} - x_{s,i}| \\ &\le |\boldsymbol{x}_0|_\infty + C' \sum_{s=0}^{j} 2^{-s} \left|\boldsymbol{x}_s^{[1]}\right|_\infty + \sum_{s=0}^{j} d_s. \end{aligned}$$

Applying (5.6) with $\ell = 1$ now gives us

$$\begin{aligned} \sum_{s=0}^{j} 2^{-s} \left|\boldsymbol{x}_s^{[1]}\right|_\infty &\le C_1 \sum_{s=0}^{j} 2^{-s\varepsilon} \left(\left|\boldsymbol{x}_0^{[1]}\right|_\infty + v_{s-1}(\varepsilon) \right) \\ &\le \frac{2C_1}{1-2^{-\varepsilon}} |\boldsymbol{x}_0|_\infty + C_1 \sum_{s=0}^{j} 2^{-s\varepsilon} v_{s-1}(\varepsilon), \end{aligned}$$

where we used the fact that $\left|\boldsymbol{x}_0^{[1]}\right|_\infty \le 2\,|\boldsymbol{x}_0|_\infty$. Moreover,

$$\sum_{s=0}^{j} 2^{-s\varepsilon} v_{s-1}(\varepsilon) = \sum_{s=1}^{j}\sum_{k=0}^{s-1} 2^{(k-s)\varepsilon} d_k = \sum_{k=0}^{j-1} d_k \sum_{s=1}^{j-k} 2^{-s\varepsilon} \le \frac{2^{-\varepsilon}}{1-2^{-\varepsilon}} \sum_{k=0}^{j-1} d_k.$$

This concludes the proof of (5.5) with

$$C = \max\left(1 + C' \frac{2C_1}{1-2^{-\varepsilon}}, C'C_1 \frac{2^{-\varepsilon}}{1-2^{-\varepsilon}} + 1\right) = 1 + \frac{2C'C_1}{1-2^{-\varepsilon}}.$$

□

5.3. **Proof of Theorem 5.1.** We now return to the proof of Theorem 5.1. The existence of the solution $\boldsymbol{x} \in C^{p+q+1}([0,T] \times \mathbb{T})$ follows from standard theory for ODEs. It means in particular that the local truncation error in (A1) is well-defined. Together with (A3) it also means that we can use Theorem 5.5 for derivatives up to order $p+1$.

Let

$$\boldsymbol{\varepsilon}_j^n = \boldsymbol{x}_j^n - \boldsymbol{x}_j(t_{j,n}), \qquad \boldsymbol{\varepsilon}_j = \max_{0\le t_{j,n}\le T} \left|\boldsymbol{\varepsilon}_j^n\right|_\infty,$$

and

$$\boldsymbol{\delta}_j^n = \boldsymbol{w}_j^n - \boldsymbol{w}_j(t_{j,n}), \qquad \boldsymbol{\delta}_j = \max_{0\le t_{j,n}\le T} \left|\boldsymbol{\delta}_j^n\right|_\infty.$$

We also let $\boldsymbol{\tau}_j = \max_n |\boldsymbol{\tau}_j^n|$ be the biggest local truncation error on level j. Moreover, we define

$$\phi_j^n := \phi_G\left(t_{j,n+\eta_1}, \boldsymbol{x}_{j-1}^{(n+\eta_1)m}, \cdots, t_{j,n+\eta_s}, \boldsymbol{x}_{j-1}^{(n+\eta_s)m}, \boldsymbol{w}_j^n, \Delta t_j\right)$$

and

$$\phi_j(t_{j,n}) := \phi_G\left(t_{j,n+\eta_1}, \boldsymbol{x}_{j-1}(t_{j,n+\eta_1}), \cdots, t_{j,n+\eta_s}, \boldsymbol{x}_{j-1}(t_{j,n+\eta_s}), \boldsymbol{w}_j(t_{j,n}), \Delta t_j\right).$$

We first estimate $\boldsymbol{\delta}_j^n$. We have

$$\boldsymbol{\delta}_j^{n+1} = \boldsymbol{\delta}_j^n + \Delta t_j\left[\phi_j^n - \phi_j(t_{j,n})\right] - \boldsymbol{\tau}_j^n$$

Denoting the constant in (A2) by C' we get,

$$\left|\phi_j(t_{j,n}) - \phi_j^n\right| \leq C' \left[\left(|\boldsymbol{w}_j(t_{j,n})|_\infty + \max_{1\leq k\leq s} |\Delta \boldsymbol{x}_{j-1}(t_{j,n+\eta_k})|_\infty \right) \max_{1\leq k\leq s} \left|\boldsymbol{\varepsilon}_{j-1}^{(n+\eta_k)m}\right|_\infty + \left|\boldsymbol{\delta}_j^n\right|_\infty + \max_{1\leq k\leq s} \left|\boldsymbol{\varepsilon}_{j-1}^{(n+\eta_k)m}\right|_\infty^2 \right]$$

By Theorem 5.5 there is a constant C'' such that $|\boldsymbol{w}_j|_\infty \leq C'' 2^{-jq}$ and $|\Delta \boldsymbol{x}_{j-1}|_\infty \leq C'' 2^{-j}$ for $0 \leq t \leq T$. This gives,

$$\left|\phi_j(t_{j,n}) - \phi_j^n\right| \leq c_0 \left(2^{-j} \max_{1\leq k\leq s} \left|\boldsymbol{\varepsilon}_{j-1}^{(n+\eta_k)m}\right|_\infty + \left|\boldsymbol{\delta}_j^n\right|_\infty + \max_{1\leq k\leq s} \left|\boldsymbol{\varepsilon}_{j-1}^{(n+\eta_k)m}\right|_\infty^2 \right),$$

where $c_0 = \max(2C'C'', C')$. We obtain,

$$\left|\boldsymbol{\delta}_j^{n+1}\right|_\infty \leq \left|\boldsymbol{\delta}_j^n\right|_\infty + c_0 \Delta t_j \left(2^{-j} \boldsymbol{\varepsilon}_{j-1} + \left|\boldsymbol{\delta}_j^n\right|_\infty + \boldsymbol{\varepsilon}_{j-1}^2 \right) + \boldsymbol{\tau}_j =: (1 + c_0 \Delta t_j) \left|\boldsymbol{\delta}_j^n\right|_\infty + Z_j,$$

with

$$Z_j = c_0 \Delta t_j \left(2^{-j} \boldsymbol{\varepsilon}_{j-1} + \boldsymbol{\varepsilon}_{j-1}^2 \right) + \boldsymbol{\tau}_j.$$

By induction, and using the fact that $\boldsymbol{\delta}_j^0 = \boldsymbol{0}$,

$$\left|\boldsymbol{\delta}_j^n\right|_\infty \leq Z_j \sum_{k=0}^{n-1} (1 + c_0 \Delta t_j)^k.$$

Since

$$\max_{0\leq t_{j,n}\leq T} (1 + c_0 \Delta t_j)^n \leq \max_{0\leq t_{j,n}\leq T} e^{c_0 n \Delta t_j} = \max_{0\leq t_{j,n}\leq T} e^{c_0 t_{j,n}} \leq e^{c_0 T},$$

we get

$$\boldsymbol{\delta}_j \leq \max_{0\leq t_{j,n}\leq T} Z_j \sum_{k=0}^{n-1} (1 + c_0 \Delta t_j)^k = \max_{0\leq t_{j,n}\leq T} Z_j \frac{(1 + c_0 \Delta t_j)^n - 1}{c_0 \Delta t_j} \leq c_1 \frac{Z_j}{\Delta t_j},$$

where $c_1 = (\exp(c_0 T) - 1)/c_0$. Next, we note that

$$\boldsymbol{\varepsilon}_{j+1}^n = S \boldsymbol{\varepsilon}_j^{mn} + \boldsymbol{\delta}_{j+1}^n.$$

By assumptions (A3) and (A4) we can use Theorem 5.7. We denote the constant in (5.5) by C''', which gives

$$\begin{aligned}
\boldsymbol{\varepsilon}_{j+1} &= \max_{0\leq t_{j+1,n}\leq T} \left|\boldsymbol{\varepsilon}_{j+1}^n\right|_\infty \leq \max_{0\leq t_{j+1,n}\leq T} C''' \left(\left|\boldsymbol{\varepsilon}_0^{m^{j+1} n}\right|_\infty + \sum_{k=0}^{j} \left|\boldsymbol{\delta}_{k+1}^n\right|_\infty \right) \\
&\leq C''' \left(\boldsymbol{\varepsilon}_0 + \sum_{k=0}^{j} \boldsymbol{\delta}_{k+1} \right) \leq C''' \left(\boldsymbol{\varepsilon}_0 + c_1 \sum_{k=0}^{j} \frac{Z_{k+1}}{\Delta t_{k+1}} \right) \\
&= C''' \left(\boldsymbol{\varepsilon}_0 + c_0 c_1 \sum_{k=0}^{j} \left(2^{-k-1} \boldsymbol{\varepsilon}_k + \boldsymbol{\varepsilon}_k^2 \right) + c_1 \sum_{k=0}^{j} \frac{\boldsymbol{\tau}_{k+1}}{\Delta t_{k+1}} \right) \\
&\leq c_2 \sum_{k=0}^{j} (2^{-k} \boldsymbol{\varepsilon}_k + \boldsymbol{\varepsilon}_k^2) + c_3 \sum_{k=1}^{j+1} \frac{\boldsymbol{\tau}_k}{\Delta t_k}, \qquad (5.11)
\end{aligned}$$

where $c_2 = C''' \max(1 + c_0c_1/2, c_0c_1)$ and $c_3 = C'''c_1$. Moreover, by assumption (A1) and Theorem 5.5

$$\boldsymbol{\tau}_j \le C'''' \Delta t_j^{p+1} \max_{\substack{0\le t\le T\\ 1\le \ell\le p+1}} \left| \frac{d^\ell \boldsymbol{w}_j(t)}{dt^\ell} \right|_\infty \le C''''C''\Delta t_j^{p+1} 2^{-jq},$$

where C'''' is the constant in (A1). Therefore, with $c_4 = c_3C''''C''$,

$$c_3 \sum_{k=1}^{j+1} \frac{\boldsymbol{\tau}_k}{\Delta t_k} \le c_4 \sum_{k=1}^{j+1} \Delta t_k^p 2^{-kq} = c_4 \Delta t^p \sum_{k=1}^{j+1} m^{kp} 2^{-kq} = c_4 \Delta t^p \sum_{k=1}^{j+1} (m^p 2^{-q})^k \le c_5 \Delta t^p,$$

where $c_5 = c_4/(1 - m^p 2^{-q})$, which is finite because of the condition (5.1) that $m < 2^{q/p}$. We will now use the following lemma on growth in nonlinear recursions, proved in Appendix A.

Lemma 5.8. *Let α, $\{a_n\}$ and b be positive real numbers. If*

$$y_{n+1} \le \sum_{j=0}^{n} (a_j y_j + \alpha y_j^2) + b, \qquad n \ge 0, \tag{5.12}$$

and

$$\sum_{j=1}^{\infty} a_j = A < \infty,$$

then there are constants C' and C'' independent of n, b and y_0 such that

$$y_n \le C' \max(b, y_0) \quad \text{whenever} \quad 0 \le n \le \frac{C''}{\max(b, y_0)}. \tag{5.13}$$

We apply the lemma to $\boldsymbol{\varepsilon}_j$ with $a_j = c_2 2^{-j}$, $\alpha = c_2$, $b = c_5 \Delta t^p$, and since $A = \sum_{k=1}^{\infty} a_k = c_2$ we obtain

$$\boldsymbol{\varepsilon}_j \le C' \max(\boldsymbol{\varepsilon}_0, c_5 \Delta t^p) \quad \text{whenever} \quad 0 \le j \le \frac{C''}{\max(\boldsymbol{\varepsilon}_0, c_5 \Delta t^p)}.$$

Furthermore, on the zeroth leve we use a standard p-th order ODE solver with step-size Δt, so $\boldsymbol{\varepsilon}_0 \le c_6 \Delta t^p$ for some constant c_6. Then $\max(\boldsymbol{\varepsilon}_0, c_5\Delta t^p) \le \max(c_5, c_6)\Delta t^p$. Since T is fixed, the time step Δt and the max level J couple as $\Delta t = \Delta t(J) = Tm^{-J}$. This means that the condition on j becomes $\max(c_5, c_6) J T^p m^{-Jp} \le C''$. As $m \ge 2$ this will always be satisfied for J large enough, say $J \ge J_0$. On the other hand, there are only a finite number of cases for which $J < J_0$, so the maximum error over these cases is bounded by some number D. The result (5.2) then follows with

$$C = \max\left(C'c_5, C'c_6, D \max_{0 \le J < J_0} \Delta t(J)^{-p} \right) = \max(C'c_5, C'c_6, Dm^{J_0p}/T^p).$$

This proves the theorem.

6. Validity of the Assumptions for Runge–Kutta Schemes

Here we verify that the assumptions on accuracy (A1) and stability (A2) in the main theorem are satisfied for standard Runge–Kutta schemes. Regarding the accuracy, we note that there are many investigations of the local truncation errors for Runge–Kutta methods, going back to Butcher [3] and Henrici [15] and more recently work by Albrecht [1] and Hosea [17]. However, the focus has traditionally been on the form of the leading error term, rather than the whole truncation error, and we have been unable to find a statement in the litterature similar to (A1), which holds in our simplified setting where F and all its derivatives are bounded. In Appendix B we therefore derive and prove the form of the truncation error for standard Runge–Kutta schemes in this setting. From this result we can then deduce (A1) under the regularity assumption that $F \in C_b^{p+1}(\mathbb{R}^+ \times \mathbb{R}^d; \mathbb{R}^d)$ for a p-th order method. We note that this is a weaker regularity requirement than what is used in Theorem 5.1, which means that for these Runge–Kutta schemes, Theorem 5.1 holds whenever (A3) and (A4) hold.

We consider an explicit p-th order s-stage Runge–Kutta scheme. For the fast method this means that $\phi_G(t_1, \boldsymbol{x}_1, \ldots, t_s, \boldsymbol{x}_s, \boldsymbol{w}, \Delta t)$ is defined as follows. First,

$$\xi_1 = G(t_1, \boldsymbol{x}_1, \boldsymbol{w}), \tag{6.1}$$

and for $k = 2, \ldots, s$,

$$\xi_k(t_1, \boldsymbol{x}_1, \ldots, t_k, \boldsymbol{x}_k, \boldsymbol{w}, \Delta t) = G\left(t_k, \boldsymbol{x}_k, \boldsymbol{w} + \Delta t \sum_{\ell=1}^{k-1} \alpha_{k,\ell}\xi_\ell\right). \tag{6.2}$$

Finally,

$$\phi_G = \sum_{k=1}^{s} \beta_k \xi_k. \tag{6.3}$$

We can then show the following theorem.

Theorem 6.1. *Suppose the coefficients $\alpha_{k,\ell}, \eta_j, \beta_k$ are chosen such that the Runge–Kutta method defined by (6.1), (6.2) and (6.3) is p-th order accurate with $p \geq 1$. Then, if $F \in C_b^{p+1}(\mathbb{R}^+ \times \mathbb{R}^d; \mathbb{R}^d)$ the method satisfies the assumptions (A1) and (A2) when applied to (3.2).*

Proof. We divide the proof into two parts, one for the accuracy (A1) and one for the stability (A2).
1) Accuracy. The assumption (A1) follows from Theorem B.1 in Appendix B if we can prove that the first $p+1$ derivatives in t and $\boldsymbol{w}$ of $H(t, \boldsymbol{w}) := G(t, \boldsymbol{x}(t), \boldsymbol{w})$ are bounded. We claim that all derivatives of H are of the same form: For any multi-index α and positive integer n,

$$\partial_t^n \partial_w^\alpha H(t, \boldsymbol{w}) = G_{n,\alpha}(t, \boldsymbol{x}(t), \boldsymbol{w}), \tag{6.4}$$

where $G_{n,\alpha}(t, \boldsymbol{x}, \boldsymbol{w}) \in C_b^{p+1-n-|\alpha|}(\mathbb{R}^+ \times \mathbb{R}^d \times \mathbb{R}^d; \mathbb{R}^d)$, i.e. all derivatives with respect to t, $\boldsymbol{x}$ and $\boldsymbol{w}$ are bounded as long as $n + |\alpha| \leq p+1$. Since

$$G(t, \boldsymbol{x}, \boldsymbol{w}) = F(t, S\boldsymbol{x} + \boldsymbol{w}) - SF(t, \boldsymbol{x}),$$

this is clearly true for $n = \alpha = 0$. Moreover, if it holds for n and α, then

$$\partial_t^n \partial_w^{\alpha+\alpha'} H(t, \boldsymbol{w}) = \partial_w^{\alpha'} G_{n,\alpha}(t, \boldsymbol{x}(t), \boldsymbol{w}) =: G_{n,\alpha+\alpha'}(t, \boldsymbol{x}(t), \boldsymbol{w}),$$

and

$$\begin{aligned}\partial_t^{n+1} \partial_w^\alpha H(t, \boldsymbol{w}) &= (\partial_t + \boldsymbol{x}'(t) \cdot \partial_x) G_{n,\alpha}(t, \boldsymbol{x}(t), \boldsymbol{w}) \\ &= (\partial_t + F(t, \boldsymbol{x}(t)) \cdot \partial_x) G_{n,\alpha}(t, \boldsymbol{x}(t), \boldsymbol{w}) =: G_{n+1,\alpha}(t, \boldsymbol{x}(t), \boldsymbol{w}).\end{aligned}$$

Using (6.4) and the assumption on F the claim follows by induction. This proves (A1).
2) Stability. Before proving this part of the theorem we derive a Lipschitz type bound for the function $G(t, x, w)$.

Lemma 6.2. *Suppose $F \in C_b^2(\mathbb{R}^+ \times \mathbb{R}^d;\mathbb{R}^d)$. Then,*

$$\begin{aligned}&|G(t,\boldsymbol{x},\boldsymbol{w}) - G(t,\tilde{\boldsymbol{x}},\tilde{\boldsymbol{w}})|_\infty \\ &\qquad \le C\left((|\boldsymbol{w}|_\infty + |\Delta\boldsymbol{x}|_\infty)\,|\boldsymbol{x}-\tilde{\boldsymbol{x}}|_\infty + |\boldsymbol{w}-\tilde{\boldsymbol{w}}|_\infty + |\boldsymbol{x}-\tilde{\boldsymbol{x}}|_\infty^2 \right),\end{aligned}$$

for some constant C independent of t, $\boldsymbol{x}$, $\tilde{\boldsymbol{x}}$, $\boldsymbol{w}$ and $\tilde{\boldsymbol{w}}$.

Proof. We fix the time t in the proof and can therefore for simplicity drop it in the notation. Let $\boldsymbol{y}(s) := \tilde{\boldsymbol{x}} + s(\boldsymbol{x}-\tilde{\boldsymbol{x}})$. Then

$$\begin{aligned}G(\boldsymbol{x},\boldsymbol{w}) - G(\tilde{\boldsymbol{x}},\boldsymbol{w}) &= \int_0^1 DF(S\boldsymbol{y}(s)+\boldsymbol{w})\cdot S(\boldsymbol{x}-\tilde{\boldsymbol{x}}) - SDF(\boldsymbol{y}(s))\cdot(\boldsymbol{x}-\tilde{\boldsymbol{x}})ds \\ &= \int_0^1 DF(S\boldsymbol{y}(s)+\boldsymbol{w})\cdot S(\boldsymbol{x}-\tilde{\boldsymbol{x}}) - DF(S\boldsymbol{y}(s))\cdot S(\boldsymbol{x}-\tilde{\boldsymbol{x}})ds \\ &+ \int_0^1 DF(S\boldsymbol{y}(s))\cdot S(\boldsymbol{x}-\tilde{\boldsymbol{x}}) - SDF(\boldsymbol{y}(s))\cdot(\boldsymbol{x}-\tilde{\boldsymbol{x}})ds.\end{aligned}$$

Moreover, since $F \in C_b^2$,

$$\begin{aligned}|DF(\boldsymbol{x}_1+\boldsymbol{w})\cdot\boldsymbol{x}_2 - DF(\boldsymbol{x}_1)\cdot\boldsymbol{x}_2|_\infty &= \left|\int_0^1 D^2F(\boldsymbol{x}_1+s\boldsymbol{w})(\boldsymbol{x}_2,\boldsymbol{w})ds\right|_\infty \\ &\le C\,|\boldsymbol{x}_2|_\infty\cdot|\boldsymbol{w}|_\infty\,.\end{aligned}$$

Let $F=(f_1,\ldots,f_d)^T$. By Proposition 2

$$\begin{aligned}&|DF(S\boldsymbol{y})\cdot S(\boldsymbol{x}-\tilde{\boldsymbol{x}}) - SDF(\boldsymbol{y})\cdot(\boldsymbol{x}-\tilde{\boldsymbol{x}})|_\infty \\ &\qquad \le \sum_{j=1}^d \left|\nabla f_j(S\boldsymbol{y})^T S(\boldsymbol{x}-\tilde{\boldsymbol{x}}) - S\nabla f_j(\boldsymbol{y})^T(\boldsymbol{x}-\tilde{\boldsymbol{x}})\right|_\infty \\ &\qquad \le C\,|\Delta\boldsymbol{y}|_\infty\,|\boldsymbol{x}-\tilde{\boldsymbol{x}}|_\infty\,.\end{aligned}$$

Thus, noting that $\Delta\boldsymbol{y} = \Delta\boldsymbol{x} + (1-s)\Delta(\boldsymbol{x}-\tilde{\boldsymbol{x}})$,

$$|G(\boldsymbol{x},\boldsymbol{w}) - G(\tilde{\boldsymbol{x}},\boldsymbol{w})|_\infty \le C\left(|S(\boldsymbol{x}-\tilde{\boldsymbol{x}})|_\infty\cdot|\boldsymbol{w}|_\infty + |\boldsymbol{x}-\tilde{\boldsymbol{x}}|_\infty\cdot\left[|\Delta\boldsymbol{x}|_\infty + |\boldsymbol{x}-\tilde{\boldsymbol{x}}|_\infty\right]\right)$$

Moreover,

$$G(\tilde{\boldsymbol{x}},\boldsymbol{w}) - G(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{w}}) = \int_0^1 DF(S\tilde{\boldsymbol{x}}+\tilde{\boldsymbol{w}}+s(\boldsymbol{w}-\tilde{\boldsymbol{w}}))\cdot(\boldsymbol{w}-\tilde{\boldsymbol{w}})ds \le C\,|\boldsymbol{w}-\tilde{\boldsymbol{w}}|_\infty\,.$$

These last two equations give the desired estimate. Finally, since all constants involved can be taken independent of t the result will not depend on t. □

By the regularity assumptions on F it is at least in C_b^2, so this lemma shows that (A2) holds for the simplest of Runge–Kutta schemes, namely the Forward Euler method. To prove it is true for higher order schemes we also need the following lemma, where again the required regularity of F follows from our assumptions.

Lemma 6.3. *Suppose $F \in C_b^1(\mathbb{R}^+ \times \mathbb{R}^d;\mathbb{R}^d)$. Then,*

$$|G(t,\boldsymbol{x},\boldsymbol{w})|_\infty \le C\left(|\boldsymbol{w}|_\infty + |\Delta\boldsymbol{x}|_\infty\right),$$

for some constant C independent of t, $\boldsymbol{x}$ and $\boldsymbol{w}$.

Proof. Let $F = (f_1, \ldots, f_d)^T$. We have

$$\begin{aligned} |G(t, \boldsymbol{x}, \boldsymbol{w})|_\infty &\le |F(t, S\boldsymbol{x} + \boldsymbol{w}) - F(t, S\boldsymbol{x})|_\infty + |F(t, S\boldsymbol{x}) - SF(t, \boldsymbol{x})|_\infty \\ &\le C\,|\boldsymbol{w}|_\infty + |F(t, S\boldsymbol{x}) - SF(t, \boldsymbol{x})|_\infty \\ &\le C\,|\boldsymbol{w}|_\infty + \sum_{j=1}^{d} |f_j(t, S\boldsymbol{x}) - Sf_j(t, \boldsymbol{x})|_\infty . \end{aligned}$$

From Proposition 2 with $\boldsymbol{y} = \mathbf{1}$ we furthermore get,

$$|f_j(t, S\boldsymbol{x}) - Sf_j(t, \boldsymbol{x})|_\infty \le C\,|Df_j(t, \cdot)|_\infty\,|\Delta \boldsymbol{x}|_\infty \le C\,|\Delta \boldsymbol{x}|_\infty ,$$

with C independent of t and $\boldsymbol{x}$. This proves the lemma. □

For the recursion in the remainder of the proof we will use the following lemma, which is proved in Appendix A.

Lemma 6.4. *Let a and $\{b_n\}$ be positive real numbers. If*

$$y_1 \le b_1, \qquad y_n \le a \sum_{j=1}^{n-1} y_j + b_n, \qquad n > 1, \tag{6.5}$$

then, for $n \ge 1$,

$$y_n \le \max(a, 1) e^{(n-1)a} \sum_{j=1}^{n} b_j. \tag{6.6}$$

We are now ready for the proof that the Runge–Kutta methods defined by (6.1), (6.2) and (6.3) satisfy (A2). Let $G_k := G(t_k, \boldsymbol{x}_k, \boldsymbol{w})$. Then by Lemma 6.2,

$$\begin{aligned} |\xi_k|_\infty &= \left| G\left(t_k, \boldsymbol{x}_k, \boldsymbol{w} + \Delta t \sum_{\ell=1}^{k-1} \alpha_{k,\ell} \xi_\ell \right) \right|_\infty \\ &\le |G_k|_\infty + \left| G\left(t_k, \boldsymbol{x}_k, \boldsymbol{w} + \Delta t \sum_{\ell=1}^{k-1} \alpha_{k,\ell} \xi_\ell \right) - G_k \right|_\infty \\ &\le |G_k|_\infty + C' \Delta t \sum_{\ell=1}^{k-1} \alpha_{k,\ell} |\xi_\ell|_\infty \le |G_k|_\infty + c_0 \sum_{\ell=1}^{k-1} |\xi_\ell|_\infty . \end{aligned}$$

where C' is the constant in Lemma 6.2 and $c_0 = C' \Delta t \max_{k,\ell} |\alpha_{k,\ell}|$. Furthermore, let C'' be the constant in Lemma 6.3 and $c_1 = \max(c_0, 1) \exp((s-1)c_0)$. By Lemma 6.3 and Lemma 6.4 with $a = c_0$ and $b_n = |G_n|_\infty$,

$$|\xi_k|_\infty \le c_1 \sum_{\ell=1}^{k} |G_\ell|_\infty \le s c_1 C'' \left(|\boldsymbol{w}|_\infty + \max_{1 \le j \le s} |\Delta \boldsymbol{x}_j|_\infty \right). \tag{6.7}$$

Next, we have by Lemma 6.2,

$$\begin{aligned} \left| \xi_1 - \tilde{\xi}_1 \right|_\infty &= |G(t_1, \boldsymbol{x}_1, \boldsymbol{w}) - G(t_1, \tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{w}})|_\infty \\ &\le C' \left((|\boldsymbol{w}|_\infty + |\Delta \boldsymbol{x}_1|_\infty) |\boldsymbol{x}_1 - \tilde{\boldsymbol{x}}_1|_\infty + |\boldsymbol{w} - \tilde{\boldsymbol{w}}|_\infty + |\boldsymbol{x}_1 - \tilde{\boldsymbol{x}}_1|_\infty^2 \right). \end{aligned}$$

For $k > 1$, by Lemma 6.2 and (6.7),

$$
\begin{aligned}
\left|\xi_k - \tilde{\xi}_k\right|_\infty &= |\xi_k(t_1, \boldsymbol{x}_1, t_2, \boldsymbol{x}_2, \cdots, t_k, \boldsymbol{x}_k, \boldsymbol{w}, \Delta t) - \xi_k(t_1, \tilde{\boldsymbol{x}}_1, t_2, \tilde{\boldsymbol{x}}_2, \cdots, t_k, \tilde{\boldsymbol{x}}_k, \tilde{\boldsymbol{w}}, \Delta t)|_\infty \\
&= \left| G\left(t_k, \boldsymbol{x}_k, \boldsymbol{w} + \Delta t \sum_{\ell=1}^{k-1} \alpha_{k,\ell}\xi_\ell\right) - G\left(t_k, \tilde{\boldsymbol{x}}_k, \tilde{\boldsymbol{w}} + \Delta t \sum_{\ell=1}^{k-1} \alpha_{k,\ell}\tilde{\xi}_\ell\right)\right|_\infty \\
&\leq C' \left[\left(\left| \boldsymbol{w} + \Delta t \sum_{\ell=1}^{k-1} \alpha_{k,\ell}\xi_\ell \right|_\infty + |\Delta \boldsymbol{x}_k|_\infty \right) |\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k|_\infty + |\boldsymbol{w} - \tilde{\boldsymbol{w}}|_\infty \right. \\
&\quad \left. + \Delta t \sum_{\ell=1}^{k-1} \alpha_{k,\ell} \left|\xi_\ell - \tilde{\xi}_\ell\right|_\infty + |\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k|_\infty^2 \right] \\
&\leq \left(C'(|\boldsymbol{w}|_\infty + |\Delta \boldsymbol{x}_k|_\infty) + c_0 \sum_{\ell=1}^{k-1} |\xi_\ell| \right) |\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k|_\infty + C' |\boldsymbol{w} - \tilde{\boldsymbol{w}}|_\infty \\
&\quad + c_0 \sum_{\ell=1}^{k-1} \left|\xi_\ell - \tilde{\xi}_\ell\right|_\infty + C' |\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k|_\infty^2 \\
&\leq (C' + s^2 c_0 c_1 C'') \left(|\boldsymbol{w}|_\infty + \max_{1\leq j\leq s} |\Delta \boldsymbol{x}_j|_\infty \right) |\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k|_\infty + C' |\boldsymbol{w} - \tilde{\boldsymbol{w}}|_\infty \\
&\quad + c_0 \sum_{\ell=1}^{k-1} \left|\xi_\ell - \tilde{\xi}_\ell\right|_\infty + C' |\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k|_\infty^2 .
\end{aligned}
$$

Again, by Lemma 6.4 applied to $|\xi_k - \tilde{\xi}_k|$,

$$
\begin{aligned}
\left|\xi_k - \tilde{\xi}_k\right|_\infty &\leq c_1 \sum_{\ell=1}^{k} \left[(C' + s^2 c_0 c_1 C'') \left(|\boldsymbol{w}|_\infty + \max_{1\leq j\leq s} |\Delta \boldsymbol{x}_j|_\infty \right) |\boldsymbol{x}_\ell - \tilde{\boldsymbol{x}}_\ell|_\infty \right. \\
&\quad \left. + C' |\boldsymbol{w} - \tilde{\boldsymbol{w}}|_\infty + C' |\boldsymbol{x}_\ell - \tilde{\boldsymbol{x}}_\ell|_\infty^2 \right] \\
&\leq s c_1 \left[(C' + s^2 c_0 c_1 C'') \left(|\boldsymbol{w}|_\infty + \max_{1\leq j\leq s} |\Delta \boldsymbol{x}_j|_\infty \right) \max_{1\leq j\leq s} |\boldsymbol{x}_j - \tilde{\boldsymbol{x}}_j|_\infty \right. \\
&\quad \left. + C' |\boldsymbol{w} - \tilde{\boldsymbol{w}}|_\infty + \max_{1\leq j\leq s} C' |\boldsymbol{x}_j - \tilde{\boldsymbol{x}}_j|_\infty^2 \right].
\end{aligned} \tag{6.8}
$$

Moreover,

$$
|\phi_G(t_1, \boldsymbol{x}_1, \ldots, t_s, \boldsymbol{x}_s, \boldsymbol{w}, \Delta t) - \phi_G(t_1, \tilde{\boldsymbol{x}}_1, \ldots, t_s, \tilde{\boldsymbol{x}}_s, \tilde{\boldsymbol{w}}, \Delta t)|_\infty \leq \max_k |\beta_k| \sum_{k=1}^{s} \left|\xi_k - \tilde{\xi}_k\right|_\infty .
$$

which together with (6.8) proves the validity of (A2) with the constant

$$
C = s c_1 (C' + s^2 c_0 c_1 C'') s \max_k |\beta_k|.
$$

This concludes the proof. □

7. Numerical Examples

In this section we present results of using the fast interface tracking (FIT) method, and verify numerically the theoretical results in Section 5. Errors are

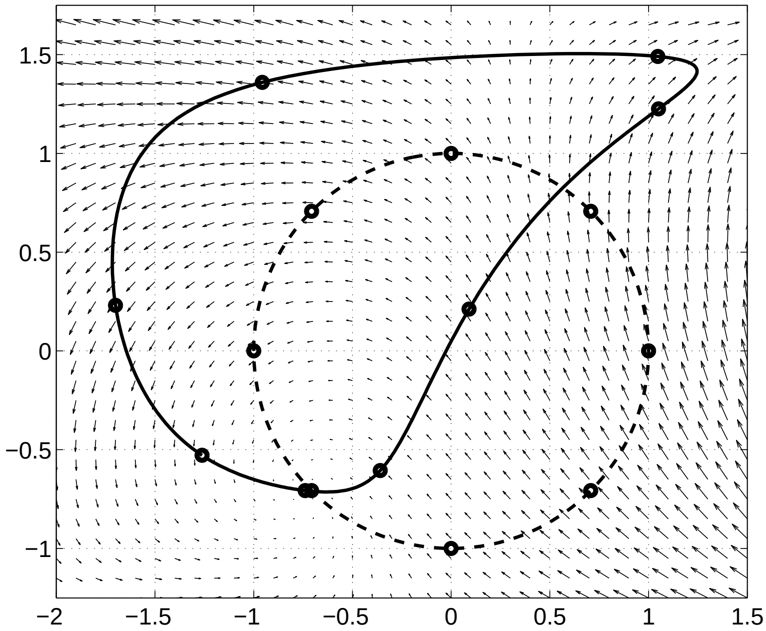


FIGURE 2. Solution example: curve plotted at $t = 0$ and $t = 1$. Vector field F overlaid.

compared with those of standard interface tracking (SIT) based on (1.2). We consider a test case where the velocity field is given by

$$F(x) = \begin{pmatrix} x_2 \sin(x_1) - \frac{1}{2} \\ (x_1 + 0.2)\cos(x_2) + 0.4 \end{pmatrix}, \qquad x = (x_1, x_2). \tag{7.1}$$

We let the initial curve be a circle $|\boldsymbol{x}| = 1$ and run the problem until time $t = 1$. The solution of this problem is shown in Figure 2, where the vector field (7.1) is overlaid.

In the implementation of the fast method the coarsest level that we use contains 2^{j_0} points, where $j_0 > 1$, and $\Delta t_j = \Delta t m^{j-j_0}$. We employ this slight modification of the method since it allows us to have reasonably large time steps on the coarsest level while still being able to explore the errors when we go to very fine levels, i.e. when taking J large; recall that $\Delta t_J = 1$ implies that $\Delta t = m^{j_0 - J}$. With very short time steps the local truncation errors in the highest order method, Runge–Kutta 4, will be of the same order as the round-off errors in double precision, which makes the results unreliable.

In the experiments we test Runge–Kutta methods of order $p = 1$, 2, 3 and 4, with the first order method being the standard Forward Euler. The parameters defining these schemes are given in Table 3.

We test spatial approximations of even orders: $q = 2$, 4, 6 and 8 given in Table 1. We also test a global approximation using Fourier interpolation and indicate this by $q = \infty$. Note that even though the corresponding subdivision operator S is non-local it can still be applied fast in $O(N \log N)$ time leading to a similar complexity as for the local subdivision operators considered in the analysis, cf. Remark 4.1. However, the bound (5.5) is not true for this S so the result in Theorem 5.1 does not

Table 3. Runge–Kutta tableaus for the schemes used. Standard notation is employed: η_j in left column, $\alpha_{k,\ell}$ in top right matrix and β_j in bottom row.

$$\begin{array}{c|c} 0 & \\ \hline & 1 \end{array}$$

Forward Euler

$$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Runge–Kutta 2

$$\begin{array}{c|ccc} 0 & & & \\ \frac{1}{2} & \frac{1}{2} & & \\ 1 & -1 & 2 & \\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{array}$$

Runge–Kutta 3

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

Runge–Kutta 4

Table 4. Errors with $m = 2$ and $N = 2048$ for various ODE-methods and spatial approximation orders.

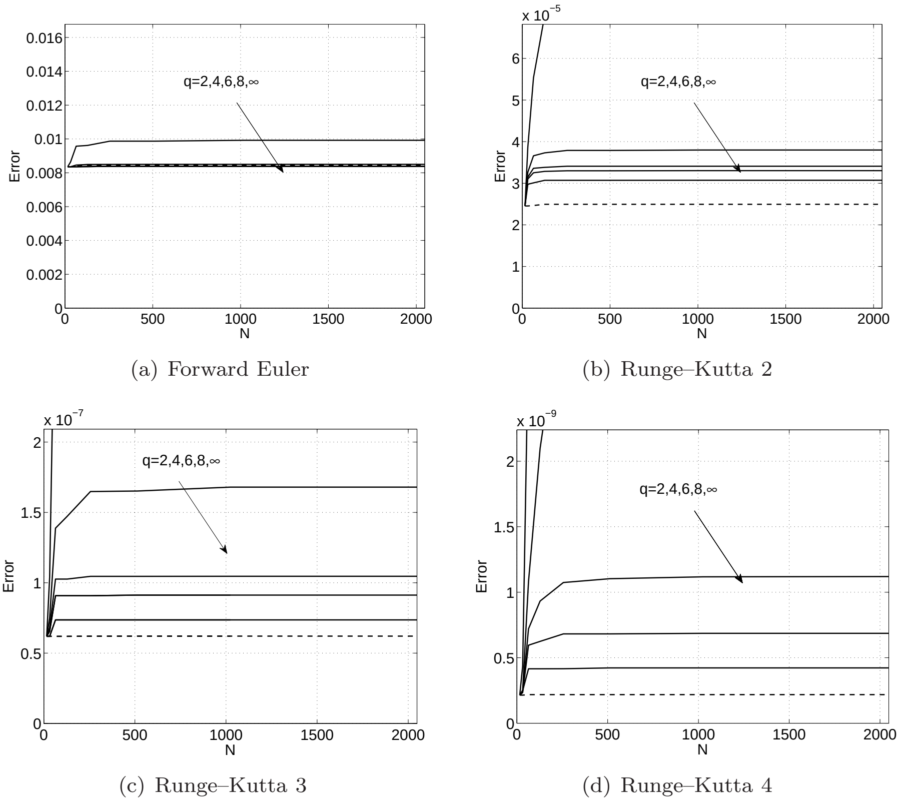
Method	p	$q = 2$	$q = 4$	$q = 6$	$q = 8$	$q = \infty$	SIT
Forward Euler	1	9.92e-03	8.50e-03	8.39e-03	8.38e-03	8.38e-03	8.46e-03
Runge–Kutta 2	2	1.13e-04	3.80e-05	3.41e-05	3.30e-05	3.07e-05	2.50e-05
Runge–Kutta 3	3	8.00e-06	1.68e-07	1.05e-07	9.12e-08	7.36e-08	6.20e-08
Runge–Kutta 4	4	2.09-e06	5.20e-09	1.12e-09	6.86e-10	4.22e-10	2.18e-10

apply, although in the experiments it performs very well. All errors are measured in max norm and compared with a well resolved numerical simulation.

In the first test we take $m = 2$, which corresponds to doubling the time step in each level. We use $j_0 = 4$, $J = 11$ and $\Delta t = 1/128$. The results are given in Figure 3 and Table 4. The dashed lines and the SIT column give the error when using standard interface tracking (1.2) with the same time step and number of points. The error in the fast interface tracking is higher but decreases with higher order spatial approximation q and remains well within a magnitude from the standard method when the stability condition $2^{q/p} > m = 2$, i.e. $p < q$, holds; it is in fact clear that when $p > q$ the error grows with N, while it remains bounded when $p < q$ as Theorem 5.1 predicts. For the borderline case $p = q$ the picture is less clear.

The same results are seen also in the second test, where we take $m = 4$, which corresponds to quadrupling the time step in each level. We use $j_0 = 5$, $J = 9$ and $\Delta t = 1/256$. The results are given in Figure 4 and Table 5. Here the stability condition $2^{q/p} > m$ implies $p < q/2$. Again, the error remains bounded when this condition is satisfied.

We have tested very smooth and nice problems here to verify the asymptotic properties of the numerical solutions. For more difficult problems (longer time) the difference between SIT and FIT is bigger since the smoothness of the s-parameterization in $x(t, s)$ typically deteriorates, which affects FIT but not SIT. For real problems one therefore also needs to add adaptivity. This is work in progress.



(a) Forward Euler

(b) Runge–Kutta 2

(c) Runge–Kutta 3

(d) Runge–Kutta 4

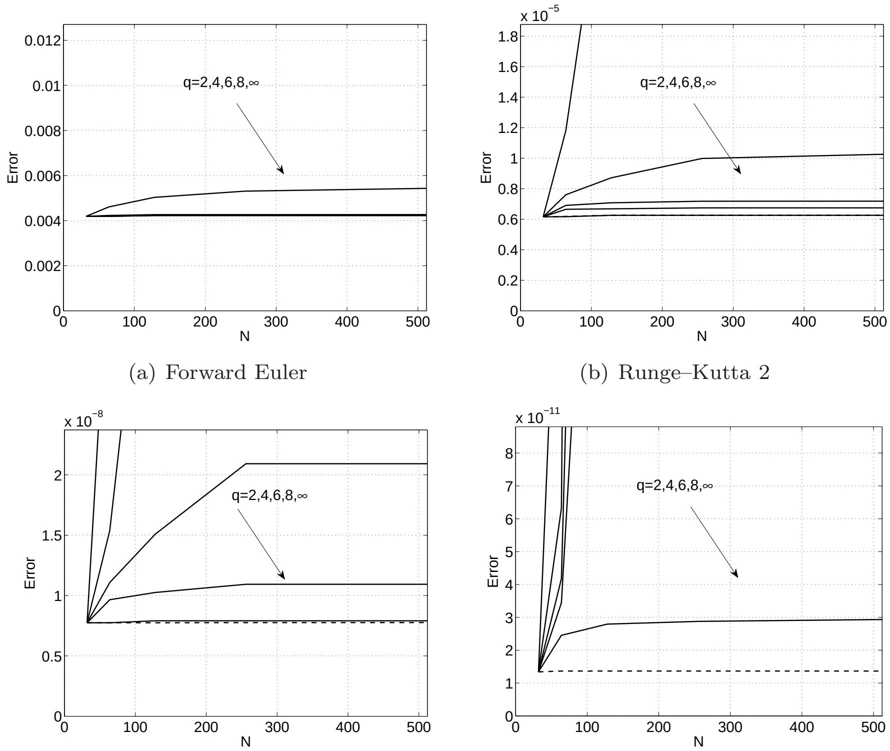
FIGURE 3. Error with $m = 2$ as a function of N for different ODE methods and spatial approximations (corresponding to $q = 2, 4, 6, 8, \infty$). Arrow indicates ordering of lines. (In frame (a) all lines but one are on top of each other.) The dashed line is the error of standard interface tracking.

TABLE 5. Errors with $m = 4$ and $N = 512$ for various ODE-methods and spatial approximation orders.

Method	p	$q = 2$	$q = 4$	$q = 6$	$q = 8$	$q = \infty$	SIT
Forward Euler	1	5.43e-03	4.27e-03	4.23e-03	4.22e-03	4.24e-03	4.24e-03
Runge–Kutta 2	2	2.30e-04	1.03e-05	7.19e-06	6.75e-06	6.26e-06	6.26e-06
Runge–Kutta 3	3	6.90e-05	3.58e-07	2.09e-08	1.09e-08	7.91e-09	7.76e-09
Runge–Kutta 4	4	2.80e-05	2.21e-07	5.50e-09	7.61e-10	2.93e-11	1.37e-11

8. ACKNOWLEDGEMENT

The author would like to thank Lexing Ying for his suggestion to use Fourier interpolation as predictor, i.e. the $q = \infty$ case in Section 7.



(a) Forward Euler

(b) Runge–Kutta 2

(c) Runge–Kutta 3

(d) Runge–Kutta 4

Figure 4. Error with $m = 4$ as a function of N for different ODE methods and spatial approximations (corresponding to $q = 2, 4, 6, 8, \infty$). Arrow indicates ordering of lines. (In frame (a) all lines but one are on top of each other.) The dashed line is the error of standard interface tracking.

References

[1] P. Albrecht. The Runge-Kutta theory in a nutshell. *SIAM J. Numer. Anal.*, 33(5):1712–1735, 1996.

[2] H.-J. Bungartz and M. Griebel. Sparse grids. *Acta Numerica*, 13:1–121, 2004.

[3] J. C. Butcher. Coefficients for the study of Runge-Kutta integration processes. *J. Austral. Math. Soc.*, 3:185–201, 1963.

[4] E. Candès and L. Ying. Fast geodesics computations with the phase flow method. *J. Comput. Phys.*, 220:6–18, 2006.

[5] A. S. Cavaretta, W. Dahmen, and C. A. Micchelli. Stationary subdivision. *Memoirs Amer. Math. Soc.*, 93(453), 1991.

[6] I. Daubechies and J. C. Lagarias. Two-scale difference equations I. Existence and global regularity of solutions. *SIAM J. Math. Anal.*, 22(5):1388–1410, 1991.

[7] I. Daubechies and J. C. Lagarias. Two-scale difference equations II. Local regularity, infinite products of matrices and fractals. *SIAM J. Math. Anal.*, 23(4):1031–1079, 1992.

[8] I. Daubechies, O. Runborg, and W. Sweldens. Normal multiresolution approximation of curves. *Constr. Approx.*, 20:399–463, 2004.

[9] G. Deslauriers and S. Dubuc. Interpolation dyadique. In *Fractals, dimensions non entières et applications*, pages 44–55. Masson, Paris, 1987.

[10] N. Dyn, D. Levin, and J. Gregory. A 4-point interpolatory subdivision scheme for curve design. *Comput. Aided Geom. Des.*, 4:257–268, 1987.

[11] J. Glimm, J. W. Grove, X. L. Li, K.-M. Shyue, Y. Zeng, and Q. Zhang. Three-dimensional front tracking. *SIAM J. Sci. Comput.*, 19(3):703–727, 1998.

[12] J. Glimm, E. Isaacson, D. Marchesin, and O. McBryan. Front tracking for hyperbolic systems. *Adv. Appl. Math*, 2:91–119, 1981.

[13] I. Guskov, K. Vidimce, W. Sweldens, and P. Schröder. Normal meshes. In *Computer Graphics (SIGGRAPH '00 Proceedings)*, pages 259–268, 2000.

[14] S. Harizanov, P. Oswald, and T. Shingel. Normal multi-scale transforms for curves. *Found. Comput. Math.*, 11(6):617–656, 2011.

[15] P. Henrici. *Discrete Variable Methods in Ordinary Differential Equations*. Wiley, 1962.

[16] C. W. Hirt and B. D. Nichols. Volume of fluid (VOF) method for the dynamics of free boundaries. *J. Comput. Phys.*, 39:201–225, 1981.

[17] M. E. Hosea. A new recurrence for computing Runge-Kutta truncation error coefficients. *SIAM J. Numer. Anal.*, pages 1989–2001, 1995.

[18] G. Lambaré, P. S. Lucio, and A. Hanyga. Two-dimensional multivalued traveltime and amplitude maps by uniform sampling of ray field. *Geophys. J. Int.*, 125:584–598, 1996.

[19] S. J. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.*, 79(1):12–49, 1988.

[20] P. Oswald. Smoothness of nonlinear subdivision schemes. *Adv. in Comp. Math.*, 20:401–423, 2004.

[21] P. Oswald. Normal multi-scale transforms for surfaces. In *Proceedings of the 7th international conference on Curves and Surfaces*, pages 527–542. Springer Verlag, 2012.

[22] O. Runborg. Fast interface tracking via a multiresolution representation of curves and surfaces. *Commun. Math. Sci.*, 7(2):365–398, 2009.

[23] N. Sepasian, A. Vilanova, L. Florack, and B.M. Ter Haar Romeny. A ray tracing method for geodesic based tractography in diffusion tensor images. In *CVPRW '08. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–6, 2008.

[24] S. O. Unverdi and G. Tryggvason. A front tracking method for viscous, incompressible, multifluid flows. *J. Comput. Phys.*, 100:25–37, 1992.

[25] V. Vinje, E. Iversen, and H. Gjøystdal. Traveltime and amplitude estimation using wavefront construction. *Geophysics*, 58(8):1157–1166, 1993.

[26] J. Wallner and N. Dyn. Convergence and C^1 analysis of subdivision schemes on manifolds by proximity. *Comput. Aided Geom. Des.*, 22:593–622, 2005.

[27] J. E. Welch, F. W. Harlow, J. P. Shannon, and B. J. Daly. The MAC method: A computing technique for solving viscous, incompressible, transient fluid flow problems involving free surfaces. Los Alalmos Scientific Laboratory Report LA–3425, 1966.

[28] G. Xie and T. P.-Y. Yu. Smoothness analysis of nonlinear subdivision schemes of homoegeneous and affine invariant type. *Constr. Approx.*, 22:219–254, 2005.

[29] C. Zenger. Sparse grids. In W. Hackbusch, editor, *Parallel Algorithms for Partial Differential Equations*, volume 31 of *Notes on Numerical Fluid Mechanics*, pages 241–251. Vieweg, 1991.

Appendix A. Growth Results

In this section we prove Lemma 5.8 and Lemma 6.4 which both give estimates of the growth in certain recursions.

We start with Lemma 6.4 but consider a slightly more general case where the assumption (6.5) is replaced by

$$y_{n+1} \leq \sum_{j=0}^{n} a_j y_j + b_{n+1}.$$

This more general case will be used in the proof of Lemma 5.8 below. We define the partial sum

$$s_n := \sum_{j=0}^{n} a_j y_j,$$

and get

$$s_{n+1} = s_n + a_{n+1}y_{n+1} \leq s_n + a_{n+1}\left(\sum_{j=0}^{n} a_j y_j + b_{n+1}\right) = (1+a_{n+1})s_n + a_{n+1}b_{n+1}.$$

A simple induction argument then leads to (see e.g. [22, Lemma 4.8]), for $n \geq 1$,

$$\begin{aligned} s_{n+1} &\leq a_{n+1}b_{n+1} + \sum_{j=1}^{n} a_j b_j \prod_{k=j}^{n}(1+a_{k+1}) + s_0 \prod_{k=0}^{n}(1+a_{k+1}) \\ &\leq a_{n+1}b_{n+1} + \sum_{j=1}^{n} a_j b_j \exp\left(\sum_{k=j}^{n} a_{k+1}\right) + s_0 \exp\left(\sum_{k=0}^{n} a_{k+1}\right), \end{aligned}$$

where we also used the fact that $1 + x \leq e^x$. Consequently, for $n \geq 1$,

$$\begin{aligned} y_{n+1} &\leq s_n + b_n \leq a_n b_n + \sum_{j=1}^{n-1} a_j b_j e^{\sum_{k=j}^{n-1} a_{k+1}} + a_0 y_0 e^{\sum_{k=0}^{n-1} a_{k+1}} + b_{n+1} \\ &\leq \left(a_0 y_0 + \sum_{j=1}^{n} a_j b_j + b_{n+1}\right) \exp\left(\sum_{k=1}^{n} a_k\right), \end{aligned} \tag{A.1}$$

which also holds for $n = 0$, with the sums taken to be zero. When $y_0 = 0$ and $a_j = a$ this gives the result (6.6) and proves Lemma 6.4.

For Lemma 5.8 we start by defining a sequence $\{\beta_j\}$ that will be used to majorize $\{y_j\}$ for a, large, initial part of the sequence. The constants C_j that we define below are all chosen such that they only depend on α and a_j, not on the sequence index n or the parameter that will eventually be small, namely

$$\varepsilon := \max(b, y_0).$$

More precisely, we define $\beta_0 = y_0$, and for $n \geq 0$,

$$\beta_{n+1} = e^{A+\alpha}\left((a_0 + \alpha\beta_0)y_0 + b\sum_{j=1}^{n}(a_j + \alpha\beta_j) + b\right) =: \sum_{j=0}^{n} \beta_j c_j + d_{n+1}, \tag{A.2}$$

where

$$c_j = \alpha e^{A+\alpha}\begin{cases} y_0, & j = 0, \\ b, & j \geq 1, \end{cases} \qquad d_{n+1} = e^{A+\alpha}\left(b\sum_{j=1}^{n} a_j + a_0 y_0 + b\right),$$

with the sum being zero for $n = 0$. Then, with $C_0 := \alpha\exp(A + \alpha)$ we can bound

$$c_j \leq \varepsilon C_0, \qquad d_{n+1} \leq \varepsilon e^{A+\alpha}\left(\sum_{j=1}^{n} a_j + a_0 + 1\right) \leq \varepsilon e^{A+\alpha}[(A+1) + a_0] =: \varepsilon C_1,$$

and (A.1) gives

$$\beta_{n+1} \leq \left(\varepsilon C_0 \beta_0 + \varepsilon^2 n C_0 C_1 + \varepsilon C_1\right) e^{\varepsilon n C_0} \leq \varepsilon C_2\left(1 + \varepsilon(n+1)\right) e^{\varepsilon n C_0}, \tag{A.3}$$

where $C_2 = \max(C_0, C_0C_1, C_2)$. We now let $N(\boldsymbol{y})$ denote the largest index n for which β_n majorizes y_n,

$$y_n \leq \beta_n, \qquad 0 \leq n \leq N(\boldsymbol{y}).$$

Then,

$$y_{n+1} \leq \sum_{j=0}^{n}(a_j + \alpha\beta_j)y_j + b, \qquad 0 \leq n \leq N(\boldsymbol{y}),$$

and we can apply (A.1) with $n = N(\boldsymbol{y})$,

$$\begin{aligned} y_{N(\boldsymbol{y})+1} \leq & \left((a_0 + \alpha\beta_0)y_0 + b \sum_{j=1}^{N(\boldsymbol{y})} (a_j + \alpha\beta_j) + b \right) \exp\left(\sum_{k=1}^{N(\boldsymbol{y})} (a_k + \alpha\beta_k) \right) \\ &= \beta_{N(\boldsymbol{y})+1} \exp\left(-A - \alpha + \sum_{k=1}^{N(\boldsymbol{y})} (a_k + \alpha\beta_k) \right) \leq \beta_{N(\boldsymbol{y})+1} \exp\left(-\alpha + \alpha \sum_{k=1}^{N(\boldsymbol{y})} \beta_k \right). \end{aligned}$$

Since by construction $\beta_{N(\boldsymbol{y})+1} < y_{N(\boldsymbol{y})+1}$, this shows that

$$\sum_{k=1}^{N(\boldsymbol{y})} \beta_k > 1.$$

We thus have

$$\begin{aligned} 1 < \sum_{k=1}^{N(\boldsymbol{y})} \beta_k &\leq \varepsilon C_2 \sum_{k=1}^{N(\boldsymbol{y})} (1+\varepsilon n) e^{\varepsilon(k-1)C_0} \leq \varepsilon C_2 e^{\varepsilon N(\boldsymbol{y})C_0} \sum_{k=1}^{N(\boldsymbol{y})} (1+\varepsilon n) \\ &= C_2 e^{\varepsilon N(\boldsymbol{y})C_0} \left(\varepsilon N(\boldsymbol{y}) + \varepsilon^2 \frac{1}{2} N(\boldsymbol{y})[N(\boldsymbol{y})+1] \right) \leq g(\varepsilon N(\boldsymbol{y})), \end{aligned}$$

with $g(x)$ being the function

$$g(x) = C_2(x + x^2)e^{xC_0}.$$

Here we also used the fact that $N(\boldsymbol{y}) \geq 1$; that $y_1 \leq \beta_1$ follows easily from (6.5) and (A.2). We note that $g(0) = 0$ and $g(x)$ is strictly increasing for $x > 0$. Therefore, for positive x, $g(x) > 1$ implies $x > x^*$, where x^* only depends on C_0 and C_2. Hence,

$$\varepsilon N(\boldsymbol{y}) > x^*.$$

Thus, for $n \leq x^*/\varepsilon < N(\boldsymbol{y})$, by the estimate (A.3),

$$y_n \leq \beta_n \leq \varepsilon C_2(1+\varepsilon n)e^{\varepsilon(n-1)C_0} \leq \varepsilon C_2(1+x^*)e^{x^*C_0}.$$

The final result (5.13) thus follows with $C' = C_2(1+x^*)\exp(x^*C_0)$ and $C'' = x^*$, proving Lemma 5.8.

Appendix B. Local Truncation Error for Runge–Kutta Schemes

We consider an explicit p-th order s-stage Runge–Kutta method for the problem

$$\frac{dy(t)}{dt} = F(t, y), \qquad y \in \mathbb{R}^d,$$

where $F : \mathbb{R}^+ \times \mathbb{R}^d \to \mathbb{R}^d$. The method is defined by the parameters η_j, $\alpha_{k,\ell}$ and β_k through the following steps

$$\xi_1 = F(t_n, y_n),$$

and for $k = 2, \ldots, s$,

$$\xi_k = F\left(t_n + \eta_j h, y_n + h \sum_{\ell=1}^{k-1} \alpha_{k,\ell}\xi_\ell \right).$$

Finally,

$$y_{n+1} = y_n + h\sum_{k=1}^{s} \beta_k \xi_k =: y_n + h\phi_F(h, t_n, y_n).$$

The local truncation error is defined as the residual when the exact solution is entered into the scheme, hence

$$\tau(t, h) = y(t+h) - y(t) - h\phi_F(h, t, y(t)). \tag{B.1}$$

For this method we can prove the following

Theorem B.1. *Suppose $F \in C_b^{p+1}(\mathbb{R}^+ \times \mathbb{R}^d; \mathbb{R}^d)$ and that the coefficients $\alpha_{k,\ell}, \eta_j, \beta_k$ are chosen such that the method above is p-th order accurate. Then, for $0 \le h \le h_0$*

$$|\tau(t_n, h)| \le Ch^{p+1} \max_{\substack{t_n \le t \le t_n + h \\ 1 \le j \le p+1}} |y^{(j)}(t)|,$$

where the constant C depends on h_0 and F but is independent of h and t_n.

The proof is based on two lemmas given below. For most of the steps $F \in C_b^p$ is a sufficient regularity condition, but in the last lemma $F \in C_b^{p+1}$ is needed. In severe places in the proof it is necessary that a solution $y(t)$ exists and belongs to at least C^{p+1}. This follows from standard ODE theory with the regularity assumption on F.

We can Taylor expand $y(t_n + h)$ and $\phi_F(h, t_n, y(t_n))$ in (B.1) around $h = 0$ to order $p + 1$ and p respectively,

$$\tau = \sum_{k=0}^{p} \frac{h^k y^{(k)}(t_n)}{k!} + \frac{h^{p+1}}{(p+1)!} Z_1 - y(t_n) - h\sum_{k=0}^{p-1} \frac{h^k \phi_F^{(k)}(0, t_n, y(t_n))}{k!} - \frac{h^{p+1}}{p!} Z_2,$$

and the remainder terms can be bounded by

$$|Z_1| \le \max_{t_n \le t \le t_n + h} \left| y^{(p+1)}(t) \right|, \qquad |Z_2| \le \max_{0 \le \tilde{h} \le h} \left| \phi_F^{(p)}(\tilde{h}, t_n, y(t_n)) \right|.$$

By the definition of a p-th order method all lower order terms in h must vanish and we are left with

$$\begin{aligned} |\tau| &= \frac{h^{p+1}}{(p+1)!} \left| Z_1 - (p+1)Z_2 \right| \\ &\le \frac{h^{p+1}}{(p+1)!} \max_{t_n \le t \le t_n + h} \left| y^{(p+1)}(t) \right| + \frac{h^{p+1}}{p!} \max_{0 \le \tilde{h} \le h} \left| \phi_F^{(p)}(\tilde{h}, t_n, y(t_n)) \right|. \end{aligned}$$

To show the result in the theorem, we therefore need to prove that

$$\max_{0 \le \tilde{h} \le h} \left| \phi_F^{(p)}(\tilde{h}, t_n, y(t_n)) \right| \le C \max_{\substack{t_n \le t \le t_n + h \\ 1 \le \ell \le p+1}} |y^{(\ell)}(t)|.$$

We start by introducing the functions

$$R_0(t_n) = F(t_n, y(t_n)), \qquad R_k(t, x_1, \ldots, x_k; t_n) = F\left(t_n + \eta_k t, y(t_n) + \sum_{\ell=1}^{k} \alpha_{k,\ell} x_\ell \right),$$

where $R_k : \mathbb{R} \times \mathbb{R}^{d \times k} \to \mathbb{R}^d$ and t_n is considered as a parameter. Then

$$\xi_k(h; t_n) = R_{k-1}(h, h\Xi_{k-1}(h); t_n), \qquad \Xi_k(h; t_n) = (\xi_1(h; t_n), \ldots, \xi_k(h; t_n))^T.$$

For most of the proof we consider a fixed t_n and we will frequently drop the dependence on t_n in the notation for these functions. We now show the following lemma, which gives the form of the derivatives of ξ with respect to h.

Lemma B.2. *If $F \in C_b^{\bar{p}}(\mathbb{R}^+ \times \mathbb{R}^d; \mathbb{R}^d)$ then for $0 \le p \le \bar{p}$,*

$$\frac{d^p \xi_k(h;t_n)}{dh^p} = \sum_{\ell=0}^{k-1}\sum_{q=0}^{p} S_{\ell,k-1,q,p}(h, h\Xi_{k-1}(h;t_n);t_n)\partial_t^q R_\ell(h, h\Xi_\ell(h;t_n);t_n), \tag{B.2}$$

for $k = 2, \ldots, s$, where $S_{\ell,k,q,p}(t, x; t_n) : \mathbb{R} \times \mathbb{R}^{d\times k} \to \mathbb{R}^{d\times d}$ are continuous and bounded in x for every fixed t, uniformly in t_n.

Proof. Since $F \in C_b^{\bar{p}}(\mathbb{R}^+ \times \mathbb{R}^d; \mathbb{R}^d)$ we have $R_k \in C_b^{\bar{p}}(\mathbb{R} \times \mathbb{R}^{d\times k}; \mathbb{R}^d)$, with bounds uniform in t_n. By construction (B.2) is true for $p = 0$ and $k = 2, \ldots, s$ with $S_{\ell,k,0,0} = \delta_{\ell,k} I$. Moreover, for $p = 1$ we get

$$\begin{aligned}\frac{d\xi_k(h)}{dh} &= \partial_t R_{k-1}(h, h\Xi_{k-1}(h)) + \sum_{\ell=1}^{k-1}[\partial_{x_\ell} R_{k-1}(h, h\Xi_{k-1}(h))](\xi_\ell(h) + h\xi'_\ell(h)) \\ &= \partial_t R_{k-1} + \sum_{\ell=1}^{k-1}[\partial_{x_\ell} R_{k-1}]R_{\ell-1} + h\sum_{\ell=1}^{k-1}[\partial_{x_\ell} R_{k-1}]\xi'_\ell.\end{aligned}$$

When $k = 2$ this reduces to $\xi'_2 = \partial_t R_1 + [\partial_{x_1} R_1]R_0$ since ξ_1 is independent of h. The $p = 1$, $k = 2$ case is thus of the form (B.2). If this is true upto $k \ge 2$, then

$$\frac{d\xi_{k+1}(h)}{dh} = \partial_t R_k + \sum_{\ell=1}^{k}[\partial_{x_\ell} R_k]R_{\ell-1} + \sum_{\ell=1}^{k}\sum_{r=0}^{\ell-1}\sum_{q=0}^{1} h[\partial_{x_\ell} R_k]S_{r,\ell-1,q,p}\partial_t^q R_r,$$

which shows, by induction, that (B.2) holds for $p = 1$ and $k = 2, \ldots, s$. Moreover, $S_{\ell,k,q,1}$ belongs to $C^{\bar{p}-1}$ and it is bounded in y (but not in t because of the h multiplying the second sum), uniformly in t_n. We claim that (B.2) holds with $S_{\ell,k,q,p} \in C^{\bar{p}-p}$ for all $0 \le p \le \bar{p}$. Indeed, if true upto some $p \in [1, \bar{p}-1]$ and $k \ge 2$,

$$\begin{aligned}\frac{d^{p+1}\xi_k(h)}{dh^{p+1}} &= \frac{d}{dh}\sum_{\ell=0}^{k-1}\sum_{q=0}^{p} S_{\ell,k-1,q,p}(h, h\Xi_\ell(h))\partial_t^q R_\ell(h, h\Xi_\ell(h)) \\ &= \sum_{\ell=1}^{k-1}\sum_{q=0}^{p} S_{\ell,k-1,q,p}(h, h\Xi_{k-1}(h))\partial_t^{q+1} R_\ell(h, h\Xi_\ell(h)) \\ &\quad + \sum_{\ell=1}^{k-1}\sum_{q=0}^{p} \partial_t S_{\ell,k,q,p}(h, h\Xi_{k-1}(h))\partial_t^q R_\ell(h, h\Xi_\ell(h)) \\ &\quad + h\sum_{\ell=1}^{k-1}[\partial_{x_\ell} D_{k-1,p}(h, h\Xi_{k-1}(h))]\xi'_\ell(h) \\ &\quad + \sum_{\ell=1}^{k-1}[\partial_{x_\ell} D_{k-1,p}(h, h\Xi_{k-1}(h))]\xi_\ell(h) \\ &=: E_1 + E_2 + E_3 + E_4,\end{aligned}$$

where

$$D_{k,p}(t,x;t_n) = \sum_{\ell=0}^{k}\sum_{q=0}^{p} S_{\ell,k,q,p}(t,x;t_n)\partial_t^q R_\ell(t,x;t_n) \ \in C^{\bar{p}-p}.$$

The terms E_1 and E_2 are clearly of the same form as (B.2), with the stipulated uniformity in t_n. So is E_4 since $\xi_\ell(h) = R_{\ell-1}(h, h\Xi_{\ell-1}(h))$. For E_3 we note that

$$\begin{aligned} &h[\partial_{x_\ell} D_{k-1,p}(h, h\Xi_{k-1}(h))]\xi'_\ell(h) \\ &\quad = h\sum_{r=0}^{\ell-1}\sum_{q=0}^{1}[\partial_{x_\ell} D_{k-1,p}(h, h\Xi_{k-1}(h))]S_{r,\ell-1,q,1}(h, h\Xi_{\ell-1}(h))\partial_t^q R_r(h, h\Xi_r(h)), \end{aligned}$$

which is again of the form (B.2). Since one t-derivative is taken in the E_2 term and one x-derivative in the E_3 and E_4 terms, the new matrix functions $S_{\ell,k,q,p+1}$ will at least belong to $C^{\bar{p}-p-1}$. This shows the lemma by induction. $\square$

Using this lemma we can write down an expression for the h-derivatives of ϕ_F. For $p \geq 1$, as long as $F \in C_b^p$,

$$\begin{aligned} \phi_F^{(p)}(h, t_n, y(t_n)) &= \sum_{k=1}^{s} \beta_k \frac{d^p \xi_k(h;t_n)}{dh^p} \\ &= \sum_{k=2}^{s}\sum_{\ell=0}^{k-1}\sum_{q=0}^{p} \beta_k S_{\ell,k-1,q,p}(h, h\Xi_{k-1}(h))\partial_t^q R_\ell(h, h\Xi_\ell(h)), \end{aligned}$$

which shows that for $0 \leq h \leq h_0$,

$$\left|\phi_F^{(p)}(h, t_n, y(t_n))\right| \leq C \max_{\substack{1\leq \ell\leq s \\ 0\leq q\leq p}} |\partial_t^q R_\ell(h, h\Xi_\ell(h))|,$$

where the constant C depends on h_0 but not on h and t_n. The result then follows from the following lemma.

Lemma B.3. *If $F \in C_b^{p+1}(\mathbb{R}^+ \times \mathbb{R}^d; \mathbb{R}^d)$ then, for $0 \leq q \leq p$ and $0 \leq h \leq h_0$,*

$$|\partial_t^q R_\ell(h, h\Xi_\ell(h;t_n);t_n)| \leq C \max_{\substack{t_n\leq t\leq t_n+h \\ 1\leq j\leq q+1}} |y^{(j)}(t)|,$$

where C depends on h_0 but is independent of h and t_n.

Proof. We claim that there are functions $S_{\ell,q}(t,x) \in C_b^{p+1-q}(\mathbb{R}\times\mathbb{R}^d; \mathbb{R}^{d\times d})$ such that for $0 \leq q \leq p$,

$$\partial_t^q F(t, y(t)) = \sum_{\ell=1}^{q} S_{\ell,q}(t, y(t))y^{(\ell)}(t) + y^{(q+1)}(t). \tag{B.3}$$

This is clearly true for $q = 0$. Moreover, if it is true up to q, then

$$\begin{aligned}
\partial_t^{q+1} F(t, y(t)) &= -\partial_t^q F_y(t, y(t))y'(t) + \frac{d}{dt}\partial_t^q F(t, y(t)) \\
&= -\partial_t^q F_y(t, y(t))y'(t) + \frac{d}{dt}\sum_{\ell=1}^{q} S_{\ell,q}(t, y(t))y^{(\ell)}(t) + y^{(q+2)}(t) \\
&= -\partial_t^q F_y(t, y(t))y'(t) + \sum_{\ell=1}^{q} S_{\ell,q}(t, y(t))y^{(\ell+1)}(t) + y^{(q+2)}(t). \\
&\quad + \sum_{\ell=1}^{q} \left[\partial_t S_{\ell,q}(t, y(t))y^{(\ell)}(t) + \partial_x S_{\ell,q}(t, y(t))(y'(t), y^{(\ell)}(t))\right],
\end{aligned}$$

where $\partial_x S_{\ell,q}(t, y(t))(z_1, z_2)$ is a bilinear form acting on $z_1, z_2 \in \mathbb{R}^d$. Hence, since $y'(t) = F(t, y)$, we can take

$$S_{\ell,q+1} = \begin{cases} -\partial_t^q F_y + \partial_t S_{\ell,q} + \partial_x S_{\ell,q}(F, \cdot), & \ell = 1, \\ S_{\ell-1,q} + \partial_t S_{\ell,q} + \partial_x S_{\ell,q}(F, \cdot), & 2 \le \ell \le q, \\ S_{q,q}, & \ell = q + 1 > 1, \end{cases} \qquad S_{0,q} \equiv 0.$$

These are sums of functions or derivatives of functions belonging to C_b^{p+1-q}, so $S_{\ell,q+1} \in C_b^{p-q}$, which shows (B.3) by induction. Hence, $|S_{\ell,q}|$ is bounded for $\ell \le q \le p$ by a number, say M_p, independent of t and x. Then when $k \ge 2$ and $q \le p$, since $F \in C_b^{p+1}$,

$$\begin{aligned}
\text{(B.4)} \quad |\partial_t^q R_k(h, h\Xi_k(h))| &= \left|\eta_k^q \partial_t^q F\left(t_n + \eta_k h, y(t_n) + h\sum_{\ell=1}^{k} \alpha_{k,\ell}\xi_\ell(h)\right)\right| \\
&\le |\partial_t^q F\left(t_n + \eta_k h, y(t_n + \eta_k h)\right)| \\
&\quad + \left|\partial_t^q F\left(t_n + \eta_k h, y(t_n) + h\sum_{\ell=1}^{k} \alpha_{k,\ell}\xi_\ell(h)\right)\right. \\
&\quad \left. - \partial_t^q F\left(t_n + \eta_k h, y(t_n + \eta_k h)\right)\right| \\
&\le |\partial_t^q F\left(t_n + \eta_k h, y(t_n + \eta_k h)\right)| \\
&\quad + |\partial_t^q \partial_y F|_\infty \left|y(t_n + \eta_k h) - y(t_n) - h\sum_{\ell=1}^{k} \alpha_{k,\ell}\xi_\ell(h)\right| \\
&\le M_p \sum_{\ell=1}^{q+1} |y^{(\ell)}(t_n + \eta_k h)| \\
&\quad + Ch_0 \max_{t_n \le t \le t_n+h} |y'(t)| + Ch_0 \sum_{\ell=1}^{k} |\xi_\ell(h)| \\
&\le C\left(\max_{\substack{t_n \le t \le t_n+h \\ 1 \le j \le q+1}} |y^{(j)}(t)| + \sum_{\ell=1}^{k} |\xi_\ell(h)|\right),
\end{aligned}$$

where C depends on h_0 but not on t or y. Taking $q = 0$ in this inequality we have, since $R_k = \xi_{k+1}$,

$$|\xi_{k+1}(h)| \leq C \left(\max_{t_n \leq t \leq t_n + h} |y'(t)| + \sum_{\ell=1}^{k} |\xi_\ell(h)| \right).$$

Since $\xi_1 = F(t_n, y(t_n)) = y'(t_n)$, we can conclude by Lemma 5.8 that

$$\max_{1 \leq \ell \leq s} |\xi_\ell(h)| \leq \max(C, 1) e^{(s-1)C} C \sum_{j=1}^{s} \max_{t_n \leq t \leq t_n + h} |y'(t)| \leq C' \max_{t_n \leq t \leq t_n + h} |y'(t)|$$

Inserting this into (B.4) proves the lemma. □